

PERMUTED INDEX OF MERT AND UNIX PROGRAMMER'S MANUALS

Legend: (I-VIII) UNIX Sections; (a-g) MERT Sections

mreceive(a) get a message of type	-1 (acknowledgement)
dp(IV) DP-11	201 data-phone interface
diff3(I)	3-way differential file comparison
abort in newio(III)	abort process
	abort process
	abort(III) generate an IOT fault
	abs, fabs(III) absolute value
abs, fabs(III)	absolute value
setmap(a) set	access, mode and starting segmentation register
access(II) determine	accessibility of file
	access(II) determine accessibility of file
acct(V)	Accounting file
ac(VIII) login	accounting
	acct(V) Accounting file
queuemn(b) queue message with no	acknowledgement expected
mreceive(a) get a message of type -1	(acknowledgement)
	acp(e) asynchronous copy
alarm(II)	activate alarm clock timer
dn(IV) DN-11	ACU interface
	ac(VIII) login accounting
	adb(I) debugger
openseg(a)	add a segment id to the process segment table
addseg(a)	add a segment to the process address space
fork(c) change count on open files and	add capabilities
spacaloc(a) allocate space for segment;	add it to the proc. virtual addr. space
for segment; add it to the proc. virtual	addr. space...spacaloc(a) allocate space
addseg(a) add a segment to the process	address space
drop a segment from a process virtual	address space...dropseg(a)
getarg(b) get argument from SUP	address space
putarg(b) put argument into SUP	address space
remove a segment from a process virtual	address space...rmovseg(a)
iomap(b) map segid/offset to virtual	address
	addseg(a) add a segment to the process address space
	adduser(a) increment user count on a process
shift(I)	adjust Shell arguments
	admin(I) administer SCCS files
admin(I)	administer SCCS files
basename(I) strip filename	affixes
	agen(VI) generate associative memory drivers
alarm(II) activate	alarm clock timer
	alarm(II) activate alarm clock timer
falloc(d)	allocate contiguous file space
falloc(c)	allocate contiguous space for a file
calloc in newio(III)	allocate memory
alocmsg(b)	allocate message buffer
falloc(f)	allocate space for contiguous file
virtual addr. space...spacaloc(a)	allocate space for segment; add it to the proc.
break, brk, sbrk(II) change core	allocation
alloc(III) core	allocator
	alloc(III) core allocator
	alockseg(a) lock a segment in memory and set write back
	alocmsg(b) allocate message buffer
	alocseg(a) create a segment
isalpha in newio(III) test for	alphabetic
riteback(b) set	altered bit on a segment
yacc(I) yet	another compiler-compiler
write(I) write to	another user
	a.out(V) assembler and link editor output

	bc(I)	arbitrary precision interactive language
	atan, atan2(III)	arc tangent function
	ar(I)	archive and library maintainer
	ar(V)	archive (library) file format
	cpio(V) format of cpio	archive
	cpio(I) copy file	archives in and out
fmove(c)	move file into a contiguous	area
fmove(d)	move file into contiguous	area
fmove(f)	move file into contiguous	area
	getarg(b) get	argument from SUP address space
	putarg(b) put	argument into SUP address space
getarg, iargc(III)	get command	arguments from Fortran
	echo(I) echo	arguments
	eval in sh(I) evaluate	arguments
glob(VIII)	generate command	arguments
	shift(I) adjust Shell	arguments
		ar(I) archive and library maintainer
		ar(V) archive (library) file format
	ascii(VII) map of	ASCII character set
	atof in newio(III)	ASCII to float conversion
	atof(III) convert	ASCII to floating
	atoi(III) convert	ASCII to integer
gmtime(III)	convert date and time to	ASCII...ctime, localtime, ascii(VII) map of ASCII character set
		as(I) assembler
	help(I)	ask for help
	a.out(V)	assembler and link editor output
	as(I)	assembler
tkill(e)	terminate all processes	associated with a terminal
	agen(VI) generate	associative memory drivers
	dm(IV)	asynchronous communication device
	acp(e)	asynchronous copy
tty(IV)	interface to low speed	asynchronous devices including typewriters
	kl(IV) KL-11 or DL-11	asynchronous interface
	statio(f) get status of	asynchronous I/O
		atan, atan2(III) arc tangent function
	atan, atan2(III)	arc tangent function
	atchintr(b)	attach a process to an interrupt
	atof in newio(III)	ASCII to float conversion
	atof(III) convert	ASCII to floating
	atoi(III) convert	ASCII to integer
	atchintr(b)	attach a process to an interrupt
	attach(a)	attach process to interrupt vector
		attach(a) attach process to interrupt vector
	wait(I)	await completion of process
lock a segment in memory and set write		back...alockseg(a)
ungetc in newio(III) push character		back
writeseg(a) force a segment to be written		back
	join(VI) relational data	base operator
		basename(I) strip filename affixes
		bas(I) basic
	bas(I)	basic
intss in newio(III) test for tss or		batch
		bc(I) arbitrary precision interactive language
writeseg(a) force a segment to		be written back
	cb(VI) C	beautifier
	su(VIII)	become privileged user
mgetlim(a) get a message of type		between given limits
	riteback(b) set altered	bit on a segment
	sleep(a) set a	bit pattern to sleep on
	psleep(b) put process to sleep on	bit pattern
pwakeup(b) wake up processes sleeping on		bit pattern
	setdspac(a) set user-supervisor d-space	bits
	strip(I) remove symbols and relocation	bits
		bj(VI) the game of black jack
	bj(VI) the game of	black jack

sync(VIII) update the super	block
update(VIII) periodically update the super	block
	boot procedures(VIII) MERT startup
	break, brk, sbrk(II) change core allocation
	break in sh(I) exit from loop
	brk, sbrk(II) change core allocation
break,	buffer size
setbuf in newio(III) set	buffer
alocmsg(b) allocate message	buffered input
getc, getw, fopen(III)	buffered output
putc, putw, creat, flush(III)	buffer
flush in newio(III) flush	buffer
freemsg(b) free up message	build special file
mknod(VIII)	C beautifier
cb(VI)	C compiler
cc(I)	C program verifier
lint(I) a	calculate hypotenuse
hypot(III)	calculator
dc(I) desk	calendar
cal(VI) print	call, lcall, vcall(II) create and execute a new process
	call
indir(II) indirect system	calloc in newio(III) allocate memory
Intro-b(b) INTRO. TO KERNEL EMT	CALLS
Intro-f(f) INTRO. TO MERT UNIX SYSTEM	CALLS
	cal(VI) print calendar
fork(c) change count on open files and add	capabilities
delcap(c) delete	capability from process PCB
sendcpmsg(a) send a	capability message
islower in newio(III) test for lower	case
isupper in newio(III) test for upper	case
in newio(III) translate to lower	case...tolower
in newio(III) translate to upper	case...toupper
ierror(III)	catch Fortran errors
signal(II)	catch or ignore signals
trap in sh(I)	catch signals
	cat(I) concatenate and print
	cb(VI) C beautifier
	cc(I) C compiler
chdir,	cd(I) change working directory
floor,	ceil(III) floor and ceiling functions
floor, ceil(III) floor and	ceiling functions
	cfree in newio(III) deallocate memory
jobchg(a)	change control to next process
break, brk, sbrk(II)	change core allocation
fork(c)	change count on open files and add capabilities
passwd(I)	change login password
chmod(c)	change mode of file
chmod(II)	change mode of file
chmod(I)	change mode
chown(II)	change owner and group of a file
chown(c)	change owner of file
chown(VIII)	change owner
chroot(I)	change root directory for a command
chdir, cd(I)	change working directory
chdir(c)	change working directory
chdir(II)	change working directory
pipe(II) create an interprocess	channel
send events to processes on a control	channel...psignal(b)
ungetc in newio(III) push	character back
putchar(a) output characters to	character device driver
gsi(VI) interpret extended	character set on GSI terminal
ascii(VII) map of ASCII	character set
fgetc in newio(III) get	character
fputc in newio(III) put	character
getc in newio(III) get	character
getchar in newio(III) get	character

getchar(III) read	character
putc in newio(III) put	character
putchar, flush(III) write	character
putchar in newio(III) put	character
getchar(a) get	characters from kernel process
putchar(a) output	characters to character device driver
	chdir, cd(I) change working directory
	chdir(c) change working directory
	chdir(II) change working directory
fsck(VIII) file system consistency	check and interactive repair
qwait(f)	check for child process termination
icon(d) do consistency	check of i-nodes
check(VIII) file system consistency	check
file system directory consistency	check...dcheck(VIII)
file system storage consistency	check...icheck(VIII)
	check(VIII) file system consistency check
chess(VI) the game of	chess
	chess(VI) the game of chess
qwait(f) check for	child process termination
	chmod(c) change mode of file
	chmod(I) change mode
	chmod(II) change mode of file
	chown(c) change owner of file
	chown(II) change owner and group of a file
	chown(VIII) change owner
	chroot(I) change root directory for a command
clrevent(a)	clear event flag(s)
clri(VIII)	clear i-node
iclri(d)	clear i-node
cron(VIII)	clock daemon
alarm(II) activate alarm	clock timer
close(c)	close a file
close(II)	close a file
fclose in newio(III)	close file
	close(c) close a file
	close(II) close a file
	clrevent(a) clear event flag(s)
	clri(VIII) clear i-node
	cmp(I) compare two files
nmcode(c) get name	code for segment
	col(VI) filter reverse line feeds
getarg, largc(III) get	command arguments from Fortran
glob(VIII) generate	command arguments
nice(I) run a	command at low priority
exit(I) terminate	command file
nohup(I) run a	command immune to hangups
sh(I) shell	command programming language
chroot(I) change root directory for a	command
ktime(e) give detailed kernel time of a	command
system in newio(III) execute	command
test(I) condition	command
time(I) time a	command
	comm(I) print lines common to two files
comm(I) print lines	common to two files
dm(IV) asynchronous	communication device
du(IV) DU-11 synchronous	communication device
dc(IV) DC-11	communications interface
dh(IV) DH-11	communications multiplexer
diff(I) differential file	comparator
strcmp in newio(III)	compare strings
cmp(I)	compare two files
	compar(III) default comparison routine for qsort
compar(III) default	comparison routine for qsort
diff3(I) 3-way differential file	comparison
cc(I) C	compiler
yacc(I) yet another	compiler-compiler

fc(I)	Fortran	compiler
rc(VI)	Ratfor	compiler
wait(I)	await	completion of process
	cat(I)	concatenate and print
	icat(d)	concatenate i-node
	strcat	concatenate strings
	test(I)	condition command
	cwait(a)	conditional wait for event
msgport(f)	send message to a process	connected to a port
	fsck(VIII)	file system consistency check and interactive repair
	icon(d)	do consistency check of i-nodes
	check(VIII)	file system consistency check
dcheck(VIII)	file system directory	consistency check
icheck(VIII)	file system storage	consistency check
	getcsw(a)	get console switch register setting
	csw(II)	read console switches
	mkfs(VIII)	construct a file system
deroff(VI)	remove Troff and Eqn	constructs
	egrep(VI)	search a file for lines containing a pattern
	fgrep(VI)	search a file for lines containing keywords
	ls(I)	list contents of directory
	isnp(d)	snap i-node contents
	fmove(c)	move file into a contiguous area
	fmove(d)	move file into contiguous area
	fmove(f)	move file into contiguous area
	falloc(d)	allocate contiguous file space
	falloc(f)	allocate space for contiguous file
	falloc(c)	allocate contiguous space for a file
psignal(b)	send events to processes on a control channel	continue in sh(I) next iteration in loop
	init(VIII)	process control initialization
	jobchg(a)	change control to next process
	units(VI)	conversion program
	atof	in newio(III) ASCII to float conversion
floating point	to double precision integer	conversion...dto(I)(III)
	ecvt, fcvt(III)	output conversion
	fscanf	in newio(III) input conversion
	itol(III)	integer to long integer conversion
	locv(III)	long output conversion
double precision integer	to floating point	conversion...lto(I)(III)
	ltoi(III)	long integer to integer conversion
	scanf	in newio(III) input conversion
	sscanf	in newio(III) input conversion
	dd(I)	convert and copy a file
	atof(III)	convert ASCII to floating
	atoi(III)	convert ASCII to integer
	ctime, localtime, gmtime(III)	convert date and time to ASCII
	dd(I)	convert and copy a file
	cpall(I)	copy all files to a directory
	cpio(I)	copy file archives in and out
	copyseg(a)	make a copy of a segment
	strcpy	in newio(III) copy string
	acp(c)	asynchronous copy
	cp(I)	copy
	pcp(e)	physical copy
	copyseg(a)	make a copy of a segment
	uucp(VI)	unix-to-unix copy
	break, brk, sbrk(II)	change core allocation
	alloc(III)	core allocator
extract user core image	from process	core dump...xusr(e)
kdump(e)	dump system state into	core file
tdmp(e)	dump system state into	core file
	core(V)	format of core image file
	xusr(e)	extract user core image from process core dump
	mem, kmem, null(IV)	core memory
terminate a process and dump		core...(P-Mgr)MSTERM(c)

	core(V) format of core image file
	sin, cos(III) trigonometric functions
ulockid(a) decrement lock	count of segment
ulockseg(a) decrement lock	count of segment
adduser(a) increment user	count on a process
lockid(a) increment the lock	count on a segment
fork(c) change	count on open files and add capabilities
wc(I) word	count
	cpall(I) copy all files to a directory
	cp(I) copy
cpio(V) format of	cpio archive
	cpio(I) copy file archives in and out
	cpio(V) format of cpio archive
crash(VIII) what to do when the system	crashes
	crash(VIII) what to do when the system crashes
creat(c)	creat a new file
pcreat(f)	creat new process
	creat(c) creat a new file
creat(II)	create a new file
(P-Mgr)P_CREAT(c)	create a process from a file
alocseg(a)	create a segment
pipe(II)	create an interprocess channel
call, lcall, vcall(II)	create and execute a new process
tmpnam in newio(III)	create tmp name
	creat(II) create a new file
	cref(I) make cross reference listing
	cron(VIII) clock daemon
cref(I) make	cross reference listing
	crypt(I) encode/decode
	crypt(III) password encoding
	csw(II) read console switches
ASCII...	ctime, localtime, gmtime(III) convert date and time to
	cubic(VI) three dimensional tic-tac-toe
ftell in newio(III) get	current offset
lnxx(III) return name of	current terminal
spline(VI) interpolate smooth	curve
cut(VI)	cut out selected fields of each line of a file
	cut(VI) cut out selected fields of each line of a file
	cwait(a) conditional wait for event
	daemon
cron(VIII) clock	daemon
lpd(VIII) line printer	daemon
recdmn(d) reconfiguration	daemon
dp(IV) DP-11 201	data-phone interface
ctime, localtime, gmtime(III) convert	date and time to ASCII
time(II) get	date and time
mdate(c) modify	date of file
date(I) print and set the	date
	date(I) print and set the date
	db(VI) debug
dc(IV)	DC-11 communications interface
	dcheck(VIII) file system directory consistency check
	dc(I) desk calculator
	dc(IV) DC-11 communications interface
dmc(IV) network link with	DDCMP protocol
	dd(I) convert and copy a file
cfree in newio(III)	deallocate memory
db(VI)	debug
adb(I)	debugger
kdb(e) kernel	debugger
tp(V)	DEC/mag tape formats
growseg(a) increase or	decrease the size of a segment
ulockid(a)	decrement lock count of segment
ulockseg(a)	decrement lock count of segment
tp(I) manipulate	DECTape and magtape
tc(IV) TC-11/TU56	DECTape
compar(III)	default comparison routine for qsort

include(V) system data structure	definitions file
	delcap(c) delete capability from process PCB
	delcap(c) delete capability from process PCB
	dsw(I) delete interactively
	tail(I) deliver the last part of a file
delta(I) make an SCCS	delta
	delta(I) make an SCCS delta
mesg(I) permit or	deny messages
dequeue(m(b)	dequeue a message
dqtype(b)	dequeue a particular message type
	dequeue(m(b) dequeue a message
	deroff(VI) remove Troff and Eqn constructs
dup(II) duplicate an open file	descriptor
fileno in newio(III) get file	descriptor
mail(I) send mail to	designated users
	dc(I) desk calculator
	detchintr(b) detach a process from an interrupt
	detach(a) detach process from interrupt vector
	detach(a) detach process from interrupt vector
	ktime(e) give detailed kernel time of a command
	access(II) determine accessibility of file
	file(I) determine format of file
ioqueuem(b) send message to I/O	device driver
putchar(a) output characters to character	device driver
dr(IV) DR-11 general	device interface
tty(IV) interface to low speed asynchronous	devices including typewriters
	df(VIII) disk free
	dh(IV) DH-11 communications multiplexer
	sdh(IV) DH11 for Satellite Processor System
	dh(IV) DH-11 communications multiplexer
	diff3(I) 3-way differential file comparison
	diff(I) differential file comparator
	diff3(I) 3-way differential file comparison
	diff(I) differential file comparator
	cubic(VI) three dimensional tic-tac-toe
loginfo(II) login inform.: name,	dir, tty, post; udata
dir(V) format of	directories
dcheck(VIII) file system	directory consistency check
unlink(c) remove	directory entry
unlink(II) remove	directory entry
chroot(I) change root	directory for a command
pwd(I) working	directory name
mknod(c) make a	directory or a special file
mknod(II) make a	directory or a special file
chdir, cd(I) change working	directory
chdir(c) change working	directory
chdir(II) change working	directory
cpall(I) copy all files to a	directory
ls(I) list contents of	directory
mkdir(I) make a	directory
mval(I) move all files to a	directory
rmdir(I) remove	directory
	dirname(I) strip simple filename
	dir(V) format of directories
	tf(IV) Telefile disk driver
hs(IV) RH11/RS03-RS04 fixed-head	disk file
rf(IV) RF11/RS11 fixed-head	disk file
	df(VIII) disk free
	du(I) summarize disk usage
rk(IV) RK-11/RK03 (or RK05)	disk
rp(IV) RP-11/RP03 moving-head	disk
	umount(c) dismount file system
	umount(II) dismount file system
	umount(VIII) dismount file system
	prof(I) display profile data
kl(IV) KL-11 or	DL-11 asynchronous interface

	dmc(IV) network link with DDCMP protocol
	dm(IV) asynchronous communication device
dn(IV)	DN-11 ACU interface
	dn(IV) DN-11 ACU interface
man(I) print on-line	documentation
dtol(III) floating point to	double precision integer conversion
ltod(III)	double precision integer to floating point conversion
dp(IV)	DP-11 201 data-phone interface
	dp(IV) DP-11 201 data-phone interface
	dqtype(b) dequeue a particular message type
dr(IV)	DR-11 general device interface
	dr(IV) DR-11 general device interface
getty(a) get state of tty	driver process
setty(a) set state of tty	driver process
ioqueuem(b) send message to I/O device	driver
output characters to character device	driver...putchar(a)
agen(VI) generate associative memory	drivers
Intro(IV) INTROD. TO	DRIVERS
tf(IV) Telefile disk	driver
dropseg(a)	drop a segment from a process virtual address space
address space...	dropseg(a) drop a segment from a process virtual
setdspac(a) set user-supervisor	d-space bits
	dsw(I) delete interactively
	dtchintr(b) detach a process from an interrupt
conversion...	dtol(III) floating point to double precision integer
du(IV)	DU-11 synchronous communication device
	du(I) summarize disk usage
	du(IV) DU-11 synchronous communication device
(P-Mgr)MSTERM(c) terminate a process and	dump core
idmp(d)	dump i-node
kdmp(e)	dump system state into core file
tdmp(e)	dump system state into core file
dump(V) incremental	dump tape format
dump(VIII) incremental file system	dump
od(I) octal	dump
	dump(V) incremental dump tape format
	dump(VIII) incremental file system dump
extract user core image from process core	dump...xusr(e)
	dup(II) duplicate an open file descriptor
dup(II)	duplicate an open file descriptor
cut(VI) cut out selected fields of	each line of a file
echo(I)	echo arguments
	echo(I) echo arguments
	ecvt, fcvt(III) output conversion
end, etext,	edata(III) last locations in program
	ed(I) text editor
a.out(V) assembler and link	editor output
ed(I) text	editor
ld(I) link	editor
sed(I) stream	editor
	egrep(VI) search a file for lines containing a pattern
Intro-b(b) INTRO. TO KERNEL	EMT CALLS
Intro-a(a) INTRO. TO SUPERVISOR	EMT TRAPS
enevent(a)	enable event flag(s)
crypt(I)	encode/decode
crypt(III) password	encoding
	end, etext, edata(III) last locations in program
feof in newio(III)	end-of-file
	enevent(a) enable event flag(s)
nlist(III) get	entries from name list
unlink(c) remove directory	entry
unlink(II) remove directory	entry
run(e) run an	environment (superuser)
deroff(VI) remove Troff and	Eqn constructs
	eqn(I) typeset mathematics
perror, sys_errlist, sys_nerr,	errno(III) system error messages

ferror in newio(III)	error exit
errproc(e)	error logger
sys_nerr, errno(III)	system error messages...perror, sys_errlist,
ierror(III) catch Fortran	errors
spell(VI) find spelling	errors
end,	errproc(e) error logger
eval in sh(I)	etext, edata(III) last locations in program
clevent(a) clear	eval in sh(I) evaluate arguments
enevent(a) enable	evaluate arguments
event(a) send	event flag(s)
sendevent(b) send	event flag(s)
cwait(a) conditional wait for	event to a process
psignal(b) send	event to a process
sendev(f) send	event(a) send event to a process
waitev(f) wait for an	event
exec,	events to processes on a control channel
exec, execl, execv(II)	event(s)
exec in sh(I)	event
exec(c) open file for execution	exec, execl, execv(II) execute a file
execl, execv(II)	exec in sh(I) execute within shell
execute a file	exec(c) open file for execution
execute a new process	execl, execv(II) execute a file
execute command	execute a file
execute new process	exec, execl, execv(II) create and
execute non-local goto	system in newio(III)
execute within shell	execute(a)
execute(a) execute new process	execute new process
execution for an interval	reset, setexit(III)
execution for interval	execute non-local goto
execution for small interval	exec in sh(I)
execution indefinitely	execute within shell
execution profile	execute(a) execute new process
execution time profile	execution for an interval
execution	sleep(I) suspend
exec(c) open file for	sleep(II) stop
exec, execl,	qsleep(f) stop
execv(II) execute a file	pause(II) suspend
exit from loop	monitor(III) prepare
exit from subroutine	profil(II)
exit in newio(III) exit from subroutine	execution profile
exit	execution time profile
exit(I) terminate command file	exec(c) open file for
exit(II) terminate process	exec, execl,
expected...queuomn(b)	execv(II) execute a file
exp(III) exponential function	break in sh(I)
exponential function	exit in newio(III)
exponentiation	exit from subroutine
extended character set on GSI terminal	exit
extended TTY-37 type-box	exit(I) terminate command file
extract user core image from process core dump	exit(II) terminate process
fabs(III) absolute value	expected...queuomn(b)
faloc(c) allocate contiguous space for a file	exp(III) exponential function
faloc(d) allocate contiguous file space	exponential function
faloc(f) allocate space for contiguous file	exponentiation
fault	extended character set on GSI terminal
fc(I) Fortran compiler	extended TTY-37 type-box
fclose in newio(III) close file	extract user core image from process core dump
fcreat, fflush(III) buffered output	fabs(III) absolute value
fcvt(III) output conversion	faloc(c) allocate contiguous space for a file
feeds	faloc(d) allocate contiguous file space
feof in newio(III) end-of-file	faloc(f) allocate space for contiguous file
ferror in newio(III) error exit	fault
fflush in newio(III) flush buffer	fc(I) Fortran compiler
fflush(III) buffered output	fclose in newio(III) close file
fgetc in newio(III) get character	fcreat, fflush(III) buffered output
fgets in newio(III) get string	fcvt(III) output conversion
	feeds
	feof in newio(III) end-of-file
	ferror in newio(III) error exit
	fflush in newio(III) flush buffer
	fflush(III) buffered output
	fgetc in newio(III) get character
	fgets in newio(III) get string

	fgrep(VI) search a file for lines containing keywords
cut(VI) cut out selected	fields of each line of a file
cpio(I) copy	file archives in and out
diff(I) differential	file comparator
diff3(I) 3-way differential	file comparison
dup(II) duplicate an open	file descriptor
fileno in newio(III) get	file descriptor
grep(I) search a	file for a pattern
exec(c) open	file for execution
egrep(VI) search a	file for lines containing a pattern
fgrep(VI) search a	file for lines containing keywords
mkpt(VIII) make prototype	file for use by mkfs
pfile(g) process	file format produced by ldp
ar(V) archive (library)	file format
Intro-g(g) INTRO. TO MERT	FILE FORMATS
Intro(II) INTROD. TO MERT	FILE FORMATS
Intro(V) INTROD. TO	FILE FORMATS
fmove(c) move	file into a contiguous area
fmove(d) move	file into contiguous area
fmove(f) move	file into contiguous area
split(I) split a	file into pieces
Intro-fm(c) INTRO. TO	FILE MANAGER MESSAGES
setfil(III) specify Fortran	file name
tell(II) get	file offset
falloc(d) allocate contiguous	file space
openi(c) open	file specified by inode number
stat(c) get	file status
stat(II) get	file status
fsck(VIII)	file system consistency check and interactive repair
check(VIII)	file system consistency check
dcheck(VIII)	file system directory consistency check
dump(VIII) incremental	file system dump
init(c) initialize	file system manager
restor(VIII) incremental	file system restore
icheck(VIII)	file system storage consistency check
mtab(VII) mounted	file system table
Intro-d(d) INTRO. TO	FILE SYSTEM UTILITY PROGRAMS
fs(g) format of MERT	file system volume
fs(V) format of UNIX	file system volume
mkfs(VIII) construct a	file system
mount(c) mount	file system
mount(II) mount	file system
mount(VIII) mount	file system
recon(d) reconfigure	file system
umount(c) dismount	file system
umount(II) dismount	file system
umount(VIII) dismount	file system
ftrunc(c) truncate	file to given size
cut out selected fields of each line of a	file...cut(VI)
fclose in newio(III) close	file
fopen in newio(III) open	file
fread in newio(III) read from	file
freopen in newio(III) reopen	file
fwrite in newio(III) write to	file
	file(I) determine format of file
system data structure definitions	file...include(V)
basename(I) strip	filename affixes
dirname(I) strip simple	filename
	fileno in newio(III) get file descriptor
fork(c) change count on open	files and add capabilities
cpall(I) copy all	files to a directory
mval(I) move all	files to a directory
col(VI)	filter reverse line feeds
find(I)	find files
hyphen(VI)	find hyphenated words
wdleng in newio(III)	find machine word size

	typo(I)	find possible typos
	spell(VI)	find spelling errors
		find(I) find files
	tee(I) pipe	fitting
	hs(IV) RH11/RS03-RS04	fixed-head disk file
	rf(IV) RF11/RS11	fixed-head disk file
	crevent(a) clear event	flag(s)
	enevent(a) enable event	flag(s)
	atof in newio(III) ASCII to	float conversion
	pow(III)	floating exponentiation
	fmod(III)	floating modulo function
	ltod(III) double precision integer to	floating point conversion
	fptrap(III)	floating point interpreter
	dtol(III)	floating point to double precision integer conversion
	atof(III) convert ASCII to	floating
	floor, ceil(III)	floor and ceiling functions
		floor, ceil(III) floor and ceiling functions
	fflush in newio(III)	flush buffer
	putchar,	flush(III) write character
		fmod(III) floating modulo function
		fmove(c) move file into a contiguous area
		fmove(d) move file into contiguous area
		fmove(f) move file into contiguous area
		fopen in newio(III) open file
	getc, getw,	fopen(III) buffered input
	writeseq(a)	force a segment to be written back
		fork(c) change count on open files and add capabilities
		fork(II) spawn new process
	core(V)	format of core image file
	cpio(V)	format of cpio archive
	dir(V)	format of directories
	file(I) determine	format of file
	fs(g)	format of MERT file system volume
	sccsfile(V)	format of SCCS file
	fs(V)	format of UNIX file system volume
	pfile(g) process file	format produced by ldp
	tbl(VI)	format tables for nroff or troff
	ar(V) archive (library) file	format
	dump(V) incremental dump tape	format
	man(V) manual page	format
	Intro-c(c) INTRO. TO INTERPROCESS MESSAGE	FORMATS
	Intro-g(g) INTRO. TO MERT FILE	FORMATS
	Intro(ii) INTROD. TO MERT FILE	FORMATS
	Intro(V) INTROD. TO FILE	FORMATS
	tp(V) DEC/mag tape	formats
	printf(III)	formatted print
	fprintf in newio(III) print	formatted
	printf in newio(III) print	formatted
	sprintf in newio(III) print	formatted
	nroff, troff(I) text	formatters
	nroff, troff(I) text	formatters
	umac(VI) macros for	formatting manuscripts
	fc(I)	Fortran compiler
	ierror(III) catch	Fortran errors
	setfil(III) specify	Fortran file name
	iargc(III) get command arguments from	Fortran...getarg,
		fprintf in newio(III) print formatted
		fptrap(III) floating point interpreter
		fputc in newio(III) put character
		fputs in newio(III) put string
		fread in newio(III) read from file
	freemsg(b)	free up message buffer
	df(VIII) disk	free
		freemsg(b) free up message buffer
		freeseq(a) remove a segment ID from proc-sgm-table
		freopen in newio(III) reopen file

	fscanf in newio(III)	input conversion
interactive repair...	fsck(VIII)	file system consistency check and
	fseek in newio(III)	seek to offset
	fs(g)	format of MERT file system volume
	fsize(c)	get size of file
	fstat(c)	get status of open file
	fstat(II)	get status of open file
	fs(V)	format of UNIX file system volume
	ftell in newio(III)	get current offset
	ftrunc(c)	truncate file to given size
atan, atan2(III)		arc tangent function
exp(III)		exponential function
fmod(III)		floating modulo function
gamma(III)		log gamma function
floor, ceil(III)		floor and ceiling functions
sqrt(III)		square root function
sin, cos(III)		trigonometric functions
	fwrite in newio(III)	write to file
bj(VI)		the game of black jack
chess(VI)		the game of chess
wump(VI)		the game of hunt-the-wumpus
ttt(VI)		the game of tic-tac-toe
moo(VI)		guessing game
gamma(III)		log gamma function
	gamma(III)	log gamma function
dr(IV)		DR-11 general device interface
abort(III)		generate an IOT fault
agen(VI)		generate associative memory drivers
glob(VIII)		generate command arguments
ncheck(VIII)		generate names from i-numbers
lex(VI)		generate programs for simple lexical tasks
get(I)		generation from SCCS file
s-gen(e)		system generation program
rand, srand(III)		random number generator
gettype(a)		get a message of given type
mgettype(a)		get a message of given type
msgtype(a)		get a message of given type
mreceive(a)		get a message of type -1 (acknowledgement)
mgetlim(a)		get a message of type between given limits
getmsg(a)		get a message
receive(a)		get a message
getarg(b)		get argument from SUP address space
fgetc in newio(III)		get character
getc in newio(III)		get character
getchar in newio(III)		get character
getchar(a)		get characters from kernel process
getarg, iargc(III)		get command arguments from Fortran
getcsw(a)		get console switch register setting
ftell in newio(III)		get current offset
time(II)		get date and time
nlist(III)		get entries from name list
fileno in newio(III)		get file descriptor
tell(II)		get file offset
stat(c)		get file status
stat(II)		get file status
get(I)		get generation from SCCS file
getgid(II)		get group identifications
nmcode(c)		get name code for segment
getpw(III)		get name from UID
segname(b)		get name of segment
getpw in newio(III)		get password line
getpid, getppid(II)		get process identification
times(II)		get process times
segname(a)		get segment name
fsize(c)		get size of file
size-seg(a)		get size of segment

getty(a)	get state of tty driver process
statio(f)	get status of asynchronous I/O
fstat(c)	get status of open file
fstat(II)	get status of open file
fgets in newio(III)	get string
gets in newio(III)	get string
getime(b)	get system time
tty(I)	get terminal name
getime(a)	get time
timleft(b)	get time-out value for process
gtty(II)	get typewriter status
getuid(II)	get user identifications
getseg(f)	get user segment
getw in newio(III)	get word
getarg, iargc(III)	get command arguments from Fortran
getarg(b)	get argument from SUP address space
getc, getw, fopen(III)	buffered input
getc in newio(III)	get character
getchar in newio(III)	get character
getchar(a)	get characters from kernel process
getchar(III)	read character
getcsw(a)	get console switch register setting
getgid(II)	get group identifications
get(I)	get generation from SCCS file
getime(a)	get time
getime(b)	get system time
getmsg(a)	get a message
getpid, getppid(II)	get process identification
getpw in newio(III)	get password line
getpw(III)	get name from UID
gets in newio(III)	get string
getseg(f)	get user segment
getty(a)	get state of tty driver process
gettype(a)	get a message of given type
getty(VIII)	set typewriter mode
getuid(II)	get user identifications
getc,	getc, fopen(III) buffered input
	getw in newio(III) get word
ktime(e)	give detailed kernel time of a command
mgetlim(a)	get a message of type between given limits
ftruncate(c)	truncate file to given size
gettype(a)	get a message of given type
mgettype(a)	get a message of given type
msgtype(a)	get a message of given type
ctime, localtime,	glob(VIII) generate command arguments
reset, setexit(III)	gmtime(III) convert date and time to ASCII
greek(VII)	goto
	graphics for extended TTY-37 type-box
	greek(VII) graphics for extended TTY-37 type-box
	grep(I) search a file for a pattern
getgid(II) get	group identifications
setgid(II) set process	group ID
chown(II) change owner and	group of a file
newgrp(I) log in to a new	group
	growseg(a) increase or decrease the size of a segment
gsi(VI) interpret extended character set on	GSI terminal
	gsi(VI) interpret extended character set on GSI terminal
	gtty(II) get typewriter status
	guessing game
moo(VI)	hangups
nohup(I) run a command immune to	help
help(I) ask for	help(I) ask for help
	hmul(III) high-order product
wtmp(V) user login	history
	hmul(III) high-order product

	hs(IV) RH11/RS03-RS04 fixed-head disk file
	ht(IV) RH-11/TU-16 magtape interface
wump(VI) the game of	hunt-the-wumpus
hyphen(VI) find	hyphenated words
	hyphen(VI) find hyphenated words
hypot(III) calculate	hypotenuse
	hypot(III) calculate hypotenuse
getarg,	large(III) get command arguments from Fortran
	icat(d) concatenate i-node
	icheck(VIII) file system storage consistency check
	iclr(d) clear i-node
	icon(d) do consistency check of i-nodes
freeseq(a) remove a segment	ID from proc-sgm-table
openseq(a) add a segment	id to the process segment table
getpid, getppid(II) get process	identification
getgid(II) get group	identifications
getuid(II) get user	identifications
what(I)	identify SCCS files
	idmp(d) dump i-node
setgid(II) set process group	ID
setuid(II) set process user	ID
	ierror(III) catch Fortran errors
signal(II) catch or	ignore signals
core(V) format of core	image file
xusr(e) extract user core	image from process core dump
nohup(I) run a command	immune to hangups
	include(V) system data structure definitions file
interface to low speed asynchronous devices	including typewriters...tty(IV)
growseg(a)	increase or decrease the size of a segment
lockid(a)	increment the lock count on a segment
adduser(a)	increment user count on a process
dump(V)	incremental dump tape format
dump(VIII)	incremental file system dump
restor(VIII)	incremental file system restore
pause(II) suspend execution	indefinitely
ptx(VI) permuted	index
indir(II)	indirect system call
	indir(II) indirect system call
loginfo(II) login	inform.: name, dir, tty, post; udata
utmp(V) user	information
	inhibit(a) run process at priority one
	init(c) initialize file system manager
ttys(V) typewriter	initialization data
init(VIII) process control	initialization
init(c)	initialize file system manager
(P-Mgr)pinit(c)	initialize the process manager
	init(VIII) process control initialization
isnp(d) snap	i-node contents
openi(c) open file specified by	inode number
clri(VIII) clear	i-node
icat(d) concatenate	i-node
iclr(d) clear	i-node
idmp(d) dump	i-node
icon(d) do consistency check of	i-nodes
fscanf in newio(III)	input conversion
scanf in newio(III)	input conversion
sscanf in newio(III)	input conversion
queuem(b) queue message on	input queue
getc, getw, fopen(III) buffered	input
floating point to double precision	integer conversion...dtol(III)
itol(III) integer to long	integer conversion
ltoi(III) long integer to	integer conversion
ltod(III) double precision	integer to floating point conversion
ltoi(III) long	integer to integer conversion
itol(III)	integer to long integer conversion
atoi(III) convert ASCII to	integer

bc(I) arbitrary precision	interactive language
file system consistency check and	interactive repair...fsck(VIII)
dsw(I) delete	interactively
typewriters...tty(IV)	interface to low speed asynchronous devices including
dc(IV) DC-11 communications	interface
dn(IV) DN-11 ACU	interface
dp(IV) DP-11 201 data-phone	interface
dr(IV) DR-11 general device	interface
ht(IV) RH-11/TU-16 magtape	interface
kl(IV) KL-11 or DL-11 asynchronous	interface
tm(IV) TM-11/TU-10 magtape	interface
spline(VI)	interpolate smooth curve
gsi(VI)	interpret extended character set on GSI terminal
fptrap(III) floating point	interpreter
sno(VI) Snobol	interpreter
pipe(II) create an	interprocess channel
Intro-c(c) INTRO. TO	INTERPROCESS MESSAGE FORMATS
return(I) terminate profile or	interrupt processing routine
attach(a) attach process to	interrupt vector
detach(a) detach process from	interrupt vector
atchintr(b) attach a process to an	interrupt
dtchintr(b) detach a process from an	interrupt
qsleep(f) stop execution for small	interval
sleep(I) suspend execution for an	interval
sleep(II) stop execution for	interval
Intro-fm(c)	INTRO. TO FILE MANAGER MESSAGES
Intro-d(d)	INTRO. TO FILE SYSTEM UTILITY PROGRAMS
Intro-c(c)	INTRO. TO INTERPROCESS MESSAGE FORMATS
I/o-messages(c)	INTRO. TO I/O PROCESS-MESSAGES
Intro-b(b)	INTRO. TO KERNEL EMT CALLS
Intro-g(g)	INTRO. TO MERT FILE FORMATS
Intro-f(f)	INTRO. TO MERT UNIX SYSTEM CALLS
Intro-e(e)	INTRO. TO MERT UTILITY PROGRAMS
Process-mgr(c)	INTRO. TO PROCESS-MANAGER MESSAGES
Intro-a(a)	INTRO. TO SUPERVISOR EMT TRAPS
Intro(IV)	INTROD. TO DRIVERS
Intro(V)	INTROD. TO FILE FORMATS
Intro(II)	INTROD. TO MERT FILE FORMATS
Intro(III)	INTROD. TO SUBROUTINES
Intro(VIII)	INTROD. TO SYSTEM PROGRAMS
ncheck(VIII) generate names from	intss in newio(III) test for tss or batch
ioqueuem(b) send message to	i-numbers
ioqueuem(a) send an	I/O device driver
setio(f) set	I/O message
I/o-messages(c) INTRO. TO	I/O mode of file
newio(III) a new	I/O PROCESS-MESSAGES
iolock(b) lock segment for	IO subroutine package
	I/O
	iolock(b) lock segment for I/O
	iomap(b) map segid/offset to virtual address
	I/o-messages(c) INTRO. TO I/O PROCESS-MESSAGES
pio(c) physical	I/O
	ioqueuem(a) send an I/O message
	ioqueuem(b) send message to I/O device driver
statio(f) get status of asynchronous	I/O
abort(III) generate an	IOT fault
uniolock(b) unlock segment for	I/O
	isalpha in newio(III) test for alphabetic
	isdigit in newio(III) test for numeric
	islower in newio(III) test for lower case
	isnp(d) snap i-node contents
	isspace in newio(III) test for space
	isupper in newio(III) test for upper case
spacaloc(a) allocate space for segment; add	it to the proc. virtual addr. space
continue in sh(I) next	iteration in loop
	itol(III) integer to long integer conversion

bj(VI) the game of black	jack
	jobchg(a) change control to next process
	join(VI) relational data base operator
	kdb(e) kernel debugger
	kdmp(e) dump system state into core file
	kdb(e) kernel debugger
Intro-b(b) INTRO. TO	KERNEL EMT CALLS
kpkill(e) terminate a	kernel process (superuser)
kprc(g)	kernel process translation file
getchar(a) get characters from	kernel process
ktime(e) give detailed	kernel time of a command
search a file for lines containing	keywords...fgrep(VI)
	kill(I) terminate a process
	kill(II) send signal to a process
	kl(IV) KL-11 or DL-11 asynchronous interface
	kl(IV) KL-11 or DL-11 asynchronous interface
	mem, kmem, null(IV) core memory
	kpkill(e) terminate a kernel process (superuser)
	kprc(g) kernel process translation file
	ktime(e) give detailed kernel time of a command
bc(I) arbitrary precision interactive	language
sh(I) shell command programming	language
end, etext, edata(III)	last locations in program
tail(I) deliver the	last part of a file
call,	lcall, vcall(II) create and execute a new process
	ld(I) link editor
	ldp(e) load a process
pfile(g) process file format produced by	ldp
	ldu(e) load a user process with public libraries
strlen in newio(III) obtain string	length
lex(VI) generate programs for simple	lexical tasks
	lex(VI) generate programs for simple lexical tasks
ldu(e) load a user process with public	libraries
ar(V) archive	(library) file format
ar(I) archive and	library maintainer
get a message of type between given	limits...mgetlim(a)
read(I) read one	line at a time
col(VI) filter reverse	line feeds
cut(VI) cut out selected fields of each	line of a file
lpd(VIII)	line printer daemon
lpr(I)	line printer spooler
lp(IV)	line printer
getpw in newio(III) get password	line
comm(I) print	lines common to two files
egrep(VI) search a file for	lines containing a pattern
fgrep(VI) search a file for	lines containing keywords
uniq(I) report repeated	lines in a file
rev(VI) reverse	lines of a file
paste(VI) merge the same	lines of all files
a.out(V) assembler and	link editor output
ld(I)	link editor
link(c)	link to a file
link(II)	link to a file
dmc(IV) network	link with DD&MPP protocol
	link(c) link to a file
	link(II) link to a file
ln(I) make a	link
	lint(I) a C program verifier
ls(I)	list contents of directory
cref(I) make cross reference	listing
nlist(III) get entries from name	list
nm(I) print name	list
	ln(I) make a link
	lnxx(III) return name of current terminal
ldp(e)	load a process
(Mem-Mgr)load(c) to memory manager:	load a process

ldu(e)	load a user process with public libraries
ctime,	localtime, gmtime(III) convert date and time to ASCII
end, ctext, edata(III) last	locations in program
alockseg(a)	lock a segment in memory and set write back
lockseg(a)	lock a segment in memory
(Mem-Mgr)lock(c) to memory manager: process	lock a segment
ulockid(a) decrement	lock count of segment
unlockseg(a) decrement	lock count of segment
lockid(a) increment the	lock count on a segment
plock(f)	lock process in memory
iolock(b)	lock segment for I/O
	lock(f) semaphores (USG Version)
	lockid(a) increment the lock count on a segment
	lock(II) semaphore operations
	lockseg(a) lock a segment in memory
	locv(III) long output conversion
gamma(III)	log gamma function
newgrp(I)	log in to a new group
log(III) natural	logarithm
errproc(e) error	logger
	log(III) natural logarithm
ac(VIII)	login accounting
wtmp(V) user	login history
loginfo(II)	login inform.: name, dir, tty, post; udata
passwd(I) change	login password
	loginfo(II) login inform.: name, dir, tty, post; udata
	login(I) sign onto UNIX
itol(III) integer to	long integer conversion
ltoi(III)	long integer to integer conversion
lseek(III) seek using a	long offset
locv(III)	long output conversion
break in sh(I) exit from	loop
continuc in sh(I) next iteration in	loop
nice(I) run a command at	low priority
tty(IV) interface to	low speed asynchronous devices including typewriters
islower in newio(III) test for	lower case
tolower in newio(III) translate to	lower case
	lpd(VIII) line printer daemon
	lp(IV) line printer
	lpr(I) line printer spooler
	lseek(III) seek using a long offset
	ls(I) list contents of directory
conversion...	ltod(III) double precision integer to floating point
	ltoi(III) long integer to integer conversion
	m4(VI) macro processor
wdleng in newio(III) find	machine word size
m4(VI)	macro processor
tinac(VI) ms	macros for formatting manuscripts
mtm(I)	magnetic tape manipulation
ht(IV) RH-11/TU-16	magtape interface
tm(IV) TM-11/TU-10	magtape interface
tp(I) manipulate DECTape and	magtape
mail(I) send	mail to designated users
	mail(I) send mail to designated users
ar(I) archive and library	maintainer
copyseg(a)	make a copy of a segment
mknod(e)	make a directory or a special file
mknod(II)	make a directory or a special file
mkdir(I)	make a directory
ln(I)	make a link
punswap(a)	make a process non-swap
make(I)	make a program
sunswap(a)	make a segment non-swap
mktemp(III)	make a unique named temporary file
delta(I)	make an SCCS delta
cref(I)	make cross reference listing

	mkpt(VIII)	make prototype file for use by mkfs
		make(l) make a program
	(Mem-Mgr)load(c) to memory	manager: load a process
	Intro-fm(c) INTRO. TO FILE	MANAGER MESSAGES
	(Mem-Mgr)lock(c) to memory	manager: process lock a segment
	(Mem-Mgr)term(c) to memory	manager: terminate a process
	init(c) initialize file system	manager
	(P-Mgr)pinit(c) initialize the process	manager
		man(I) print on-line documentation
	tp(I)	manipulate DECTape and magtape
	mtm(I) magnetic tape	manipulation
	man(V)	manual page format
	tmac(VI) ms macros for formatting	manuscripts
		man(V) manual page format
	ascii(VII)	map of ASCII character set
	iomap(b)	map segid/offset to virtual address
	neqn(I) typeset	mathematics on terminal
	eqn(I) typeset	mathematics
		mdate(c) modify date of file
		mem, kmem, null(IV) core memory
		(Mem-Mgr)load(c) to memory manager: load a process
	segment...	(Mem-Mgr)lock(c) to memory manager: process lock a
		(Mem-Mgr)term(c) to memory manager: terminate a process
	alockseg(a) lock a segment in	memory and set write back
	agen(VI) generate associative	memory drivers
	(Mem-Mgr)load(c) to	memory manager: load a process
	(Mem-Mgr)lock(c) to	memory manager: process lock a segment
	(Mem-Mgr)term(c) to	memory manager: terminate a process
	calloc in newio(III) allocate	memory
	cfree in newio(III) deallocate	memory
	lockseg(a) lock a segment in	memory
	mem, kmem, null(IV) core	memory
	plock(f) lock process in	memory
	sort(I) sort or	merge files
	paste(VI)	merge the same lines of all files
		msg(I) permit or deny messages
		msg(III) write message on typewriter
	(P-Mgr)pwait(c)	message at termination of process-mgr-created process
	alocmsg(b) allocate	message buffer
	freemsg(b) free up	message buffer
	Intro-c(c) INTRO. TO INTERPROCESS	MESSAGE FORMATS
	sndmsgfrom(a) send a	message from a process
	gettype(a) get a	message of given type
	mgettype(a) get a	message of given type
	msgtype(a) get a	message of given type
	mreceive(a) get a	message of type -1 (acknowledgement)
	mgetlim(a) get a	message of type between given limits
	queuem(b) queue	message on input queue
	msg(III) write	message on typewriter
	sendport(a) send	message through port
	msgport(f) send	message to a process connected to a port
	msgsend(f) send	message to a process
	ioqueuem(b) send	message to I/O device driver
	dqtype(b) dequeue a particular	message type
	queuemn(b) queue	message with no acknowledgement expected
	dequeuem(b) dequeue a	message
	getmsg(a) get a	message
	ioqueuem(a) send an I/O	message
	messink(b) return a	message
	msgrecv(f) receive	message
	receive(a) get a	message
	msg(f) send and receive	messages (USG Version)
	sendcpmsg(a) send a capability	message
	sendmsg(a) send a	message
	Intro-fm(c) INTRO. TO FILE MANAGER	MESSAGES
	msg(I) permit or deny	messages

msg(II) send and receive	messages
sys_nerr, errno(III) system error	messages...perror, sys_errlist,
Process-mgr(c) INTRO. TO PROCESS-MANAGER	MESSAGES
	messink(b) return a message
	mgetlim(a) get a message of type between given limits
	mgettype(a) get a message of given type
	mkdir(I) make a directory
mkpt(VIII) make prototype file for use by	mkfs
	mkfs(VIII) construct a file system
	mknod(c) make a directory or a special file
	mknod(II) make a directory or a special file
	mknod(VIII) build special file
	mkpt(VIII) make prototype file for use by mkfs
	mktemp(III) make a unique named temporary file
setmap(a) set access,	mode and starting segmentation register
chmod(c) change	mode of file
chmod(II) change	mode of file
setio(f) set I/O	mode of file
stty(II) set	mode of typewriter
chmod(I) change	mode
getty(VIII) set typewriter	mode
mdate(c)	modify date of file
fmod(III) floating	modulo function
	monitor(III) prepare execution profile
	moo(VI) guessing game
mount(c)	mount file system
mount(II)	mount file system
mount(VIII)	mount file system
	mount(c) mount file system
tmac(VI)	ms macros for formatting manuscripts
mtab(VII)	mounted file system table
	mount(II) mount file system
	mount(VIII) mount file system
mval(I)	move all files to a directory
fmove(c)	move file into a contiguous area
fmove(d)	move file into contiguous area
fmove(f)	move file into contiguous area
mv(I)	move or rename a file
seek(II)	move read/write pointer
rp(IV) RP-11/RP03	moving-head disk
	mreceive(a) get a message of type -1 (acknowledgement)
	msg(f) send and receive messages (USG Version)
	msg(II) send and receive messages
	msgport(f) send message to a process connected to a port
	msgrecv(f) receive message
	msgsnd(f) send message to a process
	msgtype(a) get a message of given type
	mtab(VII) mounted file system table
	mtm(I) magnetic tape manipulation
dh(IV) DH-11 communications	multiplexer
	mval(I) move all files to a directory
	mv(I) move or rename a file
	nmcode(c) get name code for segment
loginfo(II) login inform.:	name, dir, tty, post; udata
getpw(III) get	name from UID
nlist(III) get entries from	name list
nm(I) print	name list
lnxx(III) return	name of current terminal
segname(b) get	name of segment
unblkseg(a) unblock a	named segment
mktemp(III) make a unique	named temporary file
pwd(I) working directory	name
ncheck(VIII) generate	names from i-numbers
segname(a) get segment	name
setfil(III) specify Fortran file	name
tmpnam in newio(III) create tmp	name

tty(I) get terminal	name
log(III)	natural logarithm
	ncheck(VIII) generate names from i-numbers
	neqn(I) typeset mathematics on terminal
dmc(IV)	network link with DDCMP protocol
	newgrp(I) log in to a new group
continue in sh(I)	next iteration in loop
jobchg(a) change control to	next process
	nice(I) run a command at low priority
	nice(II) set program priority
	nlist(III) get entries from name list
	nmcode(c) get name code for segment
	nm(I) print name list
queuemn(b) queue message with	no acknowledgement expected
	nohup(I) run a command immune to hangups
reset, setexit(III) execute	non-local goto
pswap(a) remove	non-swap status from a process
sswap(a) remove	non-swap status from a segment
punswap(a) make a process	non-swap
sunswap(a) make a segment	non-swap
tbl(VI) format tables for	nroff or troff
	nroff, troff(I) text formatters
	nroff, troff(I) text formatters
mem, kmem,	null(IV) core memory
rand, srand(III) random	number generator
openi(c) open file specified by inode	number
isdigit in newio(III) test for	numeric
size(I) size of an	object file
reloc(VIII) relocate	object files
strlen in newio(III)	obtain string length
od(I)	octal dump
	od(I) octal dump
fseek in newio(III) seek to	offset
ftell in newio(III) get current	offset
lseek(III) seek using a long	offset
tell(II) get file	offset
read(I) read	one line at a time
inhibit(a) run process at priority	one
man(I) print	on-line documentation
login(I) sign	onto UNIX
dup(II) duplicate an	open file descriptor
exec(c)	open file for execution
openi(c)	open file specified by inode number
fopen in newio(III)	open file
fstat(c) get status of	open file
fstat(II) get status of	open file
open(c)	open file
fork(c) change count on	open files and add capabilities
open(II)	open for reading or writing
	open(c) open file
	openi(c) open file specified by inode number
	open(II) open for reading or writing
	openseg(a) add a segment id to the process segment table
lock(II) semaphore	operations
join(VI) relational data base	operator
stty(I) set terminal	options
rk(IV) RK-11/RK03	(or RK05) disk
cut(VI) cut	out selected fields of each line of a file
putchar(a)	output characters to character device driver
ecvt, fcvt(III)	output conversion
locv(III) long	output conversion
a.out(V) assembler and link editor	output
putc, putw, fcreat, fflush(III) buffered	output
chown(II) change	owner and group of a file
chown(c) change	owner of file
chown(VIII) change	owner

newio(III) a new IO subroutine	package
man(V) manual	page format
readonly in sh(I) set	parameters to readonly
set in sh(I) set	parameters
tail(I) deliver the last	part of a file
dqtype(b) dequeue a	particular message type
	passwd(I) change login password
	passwd(V) password file
crypt(III)	password encoding
passwd(V)	password file
getpw in newio(III) get	password line
passwd(I) change login	password
	paste(VI) merge the same lines of all files
sleep(a) set a bit	pattern to sleep on
search a file for lines containing a	pattern...egrep(VI)
grep(I) search a file for a	pattern
psleep(b) put process to sleep on bit	pattern
wake up processes sleeping on bit	pattern...pwakeup(b)
wakeup all processes sleeping on a	pattern...wakeup(a)
	pause(II) suspend execution indefinitely
delcap(c) delete capability from process	PCB
	pcp(e) physical copy
	pcreat(f) creat new process
update(VIII)	periodically update the super block
mesg(I)	permit or deny messages
	permit(a) run process at priority zero
ptx(VI)	permuted index
error messages...	perror, sys_errlist, sys_nerr, errno(III) system
	pfile(g) process file format produced by ldp
pcp(e)	physical copy
pio(e)	physical I/O
split(I) split a file into	pieces
	pio(e) physical I/O
tee(I)	pipe fitting
	pipe(II) create an interprocess channel
	pkill(c) terminate a process (superuser)
	plock(f) lock process in memory
	(P-Mgr)MSTERM(c) terminate a process and dump core
	(P-Mgr)P_CREAT(c) create a process from a file
	(P-Mgr)pinit(c) initialize the process manager
	(P-Mgr)pwait(c) message at termination of
process-mgr-created process...	point conversion...ltod(III)
double precision integer to floating	point interpreter
fptrap(III) floating	point to double precision integer conversion
dtol(III) floating	pointer
seek(II) move read/write	port...msgport(f)
send message to a process connected to a	port
sendport(a) send message through	ports
sysproc(f) system	possible typos
typo(I) find	post; udata
loginfo(II) login inform.: name, dir, tty,	pow(III) floating exponentiation
	precision integer conversion
dtol(III) floating point to double	precision integer to floating point conversion
ltod(III) double	precision interactive language
bc(I) arbitrary	prepare execution profile
monitor(III)	pr(I) print file
	date(I) print and set the date
	cal(VI) print calendar
	pr(I) print file
fprintf in newio(III)	print formatted
printf in newio(III)	print formatted
sprintf in newio(III)	print formatted
comm(I)	print lines common to two files
nm(I)	print name list
man(I)	print on-line documentation
prt(I)	print SCCS file

cat(I) concatenate and	print
lpd(VIII) line	printer daemon
lpr(I) line	printer spooler
lp(IV) line	printer
	printf in newio(III) print formatted
	printf(III) formatted print
printf(III) formatted	print
setprior(a) set	priority of process
inhibit(a) run process at	priority one
permit(a) run process at	priority zero
nice(I) run a command at low	priority
nice(II) set program	priority
su(VIII) become	privileged user
allocate space for segment; add it to the	proc. virtual addr. space...spacalloc(a)
boot	procedures(VIII) MERT startup
abort in newio(III) abort	process
lcall, vcall(II) create and execute a new	process...call,
tkill(e) terminate all	processes associated with a terminal
psignal(b) send events to	processes on a control channel
wakeup(a) wakeup all	processes sleeping on a pattern
pwakeup(b) wake up	processes sleeping on bit pattern
return(I) terminate profile or interrupt	processing routine
Process-mgr(c) INTRO. TO	PROCESS-MANAGER MESSAGES
to memory manager: terminate a	process...(Mem-Mgr)term(c)
I/o-messages(c) INTRO. TO I/O	PROCESS-MESSAGES
(P-Mgr)pwait(c) message at termination of	process-mgr-created process
m4(VI) macro	processor
at termination of process-mgr-created	process...(P-Mgr)pwait(c) message
to system scheduler: terminate a	process...(System Scheduler)term(c)
freeseq(a) remove a segment ID from	proc-sgm-table
pfile(g) process file format	produced by ldp
hmul(III) high-order	product
	prof(I) display profile data
prof(I) display	profile data
return(I) terminate	profile or interrupt processing routine
monitor(III) prepare execution	profile
profil(II) execution time	profile
	profil(II) execution time profile
Intro-d(d) INTRO. TO FILE SYSTEM UTILITY	PROGRAMS
Intro-e(e) INTRO. TO MERT UTILITY	PROGRAMS
Intro(VIII) INTROD. TO SYSTEM	PROGRAMS
dmc(IV) network link with DDCMP	protocol
mkpt(VIII) make	prototype file for use by mkfs
	pri(I) print SCCS file
	ps(I) process status
	psignal(b) send events to processes on a control channel
	psleep(b) put process to sleep on bit pattern
	pstart(a) start process
	pswap(a) remove non-swap status from a process
	ptimer(b) set time-out value for process
	ptx(VI) permuted index
ldu(e) load a user process with	public libraries
	punswap(a) make a process non-swap
ungetc in newio(III)	push character back
putarg(b)	put argument into SUP address space
fputc in newio(III)	put character
putc in newio(III)	put character
putchar in newio(III)	put character
psleep(b)	put process to sleep on bit pattern
fputs in newio(III)	put string
puts in newio(III)	put string
putw in newio(III)	put word
	putarg(b) put argument into SUP address space
	putc in newio(III) put character
	putc, putw, fcreat, fflush(III) buffered output
	putchar, flush(III) write character

	putchar in newio(III)	put character
	putchar(a)	output characters to character device driver
	puts in newio(III)	put string
putc,	putw, fcreat, flush(III)	buffered output
	putw in newio(III)	put word
	pwakeup(b)	wake up processes sleeping on bit pattern
	pwd(I)	working directory name
	qsleep(f)	stop execution for small interval
compar(III)	default comparison routine for	qsort
		qsort(III) quicker sort
	queuem(b)	queue message on input queue
	queuemn(b)	queue message with no acknowledgement expected
	expected...	queuem(b) queue message on input queue
queuem(b)	queue message on input	queuemn(b) queue message with no acknowledgement
	qsort(III)	quicker sort
		qwait(f) check for child process termination
	rand, srand(III)	random number generator
	rand, srand(III)	random number generator
	rc(VI)	Ratfor compiler
		rc(VI) Ratfor compiler
	getchar(III)	read character
	csw(II)	read console switches
fread in newio(III)		read from file
	read(c)	read from file
	read(II)	read from file
	read(I)	read one line at a time
		read(c) read from file
		read(I) read one line at a time
		read(II) read from file
open(II)	open for	reading or writing
readonly in sh(I)	set parameters to	readonly in sh(I) set parameters to readonly
	seek(II)	move
		read/write pointer
	msgrecv(f)	receive message
msg(f)	send and	receive messages (USG Version)
msg(II)	send and	receive messages
		receive(a) get a message
		recon(d) reconfigure file system
	recdmn(d)	reconfiguration daemon
	recon(d)	reconfigure file system
cref(I)	make cross	reference listing
	reform(VI)	reformat text file
		reform(VI) reformat text file
getcsw(a)	get console switch	register setting
set access, mode and starting segmentation		register...setmap(a)
	join(VI)	relational data base operator
	reloc(VIII)	relocate object files
strip(I)	remove symbols and	relocation bits
		reloc(VIII) relocate object files
	rmovseg(a)	remove a segment from a process virtual address space
	freeseq(a)	remove a segment ID from proc-sgm-table
	unlink(c)	remove directory entry
	unlink(II)	remove directory entry
	rmdir(I)	remove directory
	pswap(a)	remove non-swap status from a process
	sswap(a)	remove non-swap status from a segment
	strip(I)	remove symbols and relocation bits
	deroff(VI)	remove Troff and Eqn constructs
	rm(I)	remove (unlink) files
mv(I)	move or	rename a file
freopen in newio(III)		reopen file
system consistency check and interactive		repair...fsck(VIII) file
	uniq(I)	report repeated lines in a file
	uniq(I)	report repeated lines in a file

restor(VIII) incremental file system	reset, setexit(III) execute non-local goto
	restore
	restor(VIII) incremental file system restore
messink(b)	return a message
rti(a)	return from trap
lnxx(III)	return name of current terminal
routine...	return(I) terminate profile or interrupt processing
col(VI) filter	reverse line feeds
rev(VI)	reverse lines of a file
	rev(VI) reverse lines of a file
	rew(I) rewind tape
	rewind in newio(III) rewind
rew(I)	rewind tape
rewind in newio(III)	rewind
rf(IV)	RF11/RS11 fixed-head disk file
	rf(IV) RF11/RS11 fixed-head disk file
hs(IV)	RH11/RS03-RS04 fixed-head disk file
ht(IV)	RH-11/TU-16 magtape interface
	riteback(b) set altered bit on a segment
rk(IV) RK-11/RK03 (or	RK05) disk
rk(IV)	RK-11/RK03 (or RK05) disk
	rk(IV) RK-11/RK03 (or RK05) disk
	rmdir(I) remove directory
	rm(I) remove (unlink) files
address space...	rmoveg(a) remove a segment from a process virtual
chroot(I) change	root directory for a command
sqrt(III) square	root function
compar(III) default comparison	routine for qsort
terminate profile or interrupt processing	routine...return(I)
rp(IV)	RP-11/RP03 moving-head disk
	rp(IV) RP-11/RP03 moving-head disk
	rti(a) return from trap
	nice(I) run a command at low priority
	nohup(I) run a command immune to hangups
	run(e) run an environment (superuser)
	inhibit(a) run process at priority one
	permit(a) run process at priority zero
	run(e) run an environment (superuser)
paste(VI) merge the	same lines of all files
sdh(IV) DH11 for	Satellite Processor System
break, brk,	sbrk(II) change core allocation
	scanf in newio(III) input conversion
delta(I) make an	SCCS delta
get(I) get generation from	SCCS file
prt(I) print	SCCS file
admin(I) administer	SCCS files
scsfile(V) format of	SCCS file
what(I) identify	SCCS files
	scsfile(V) format of SCCS file
(System Scheduler)term(c) to system	scheduler: terminate a process
process...(System	Scheduler)term(c) to system scheduler: terminate a
	sdh(IV) DH11 for Satellite Processor System
grep(I)	search a file for a pattern
egrep(VI)	search a file for lines containing a pattern
fgrep(VI)	search a file for lines containing keywords
	sed(I) stream editor
fseek in newio(III)	seek to offset
lseek(III)	seek using a long offset
	seek(II) move read/write pointer
iomap(b) map	segid/offset to virtual address
spacaloc(a) allocate space for	segment; add it to the proc. virtual addr. space
iolock(b) lock	segment for I/O
uniolock(b) unlock	segment for I/O
dropseg(a) drop a	segment from a process virtual address space
rmoveg(a) remove a	segment from a process virtual address space
freeseg(a) remove a	segment ID from proc-sgm-table

openseg(a)	add a segment id to the process segment table
alockseg(a)	lock a segment in memory and set write back
lockseg(a)	lock a segment in memory
segname(a)	get segment name
sunswap(a)	make a segment non-swap
openseg(a)	add a segment id to the process segment table
writeseg(a)	force a segment to be written back
addseg(a)	add a segment to the process address space
setmap(a)	set access, mode and starting segmentation register
	increase or decrease the size of a segment...growseg(a)
	to memory manager: process lock a segment...(Mem-Mgr)lock(c)
	segname(a) get segment name
	segname(b) get name of segment
cut(VI)	cut out selected fields of each line of a file
lock(II)	semaphore operations
lock(f)	semaphores (USG Version)
sendcpmsg(a)	send a capability message
sndmsgfrom(a)	send a message from a process
sendmsg(a)	send a message
ioqueuem(a)	send an I/O message
msg(f)	send and receive messages (USG Version)
msg(II)	send and receive messages
event(a)	send event to a process
sendevent(b)	send event to a process
psignal(b)	send events to processes on a control channel
sendev(f)	send event(s)
mail(I)	send mail to designated users
sendport(a)	send message through port
msgport(f)	send message to a process connected to a port
msgsend(f)	send message to a process
ioqueuem(b)	send message to I/O device driver
kill(II)	send signal to a process
	sendcpmsg(a) send a capability message
	sendevent(b) send event to a process
	sendev(f) send event(s)
	sendmsg(a) send a message
	sendport(a) send message through port
sleep(a)	set a bit pattern to sleep on
setmap(a)	set access, mode and starting segmentation register
riteback(b)	set altered bit on a segment
setbuf in newio(III)	set buffer size
	set in sh(I) set parameters
setio(f)	set I/O mode of file
stty(II)	set mode of typewriter
gsi(VI)	interpret extended character set on GSI terminal
readonly in sh(I)	set parameters to readonly
set in sh(I)	set parameters
setprior(a)	set priority of process
setgid(II)	set process group ID
setuid(II)	set process user ID
nice(II)	set program priority
setty(a)	set state of tty driver process
settime(b)	set system time
tabs(VII)	set tab stops
tabs(VI)	set tabs on terminal
stty(I)	set terminal options
date(I)	print and set the date
ptimer(b)	set time-out value for process
toutset(a)	set time-out
settime(a)	set time
time(II)	set time
getty(VIII)	set typewriter mode
setdspac(a)	set user-supervisor d-space bits
alockseg(a)	lock a segment in memory and set write back
ascii(VII)	map of ASCII character set
	setbuf in newio(III) set buffer size

	setdspac(a)	set user-supervisor d-space bits
reset,	setexit(III)	execute non-local goto
	setfil(III)	specify Fortran file name
	setgid(II)	set process group ID
	setime(a)	set time
	setime(b)	set system time
	setio(f)	set I/O mode of file
register...	setmap(a)	set access, mode and starting segmentation
	setprior(a)	set priority of process
getcsw(a)	get console switch register	setting
	setty(a)	set state of tty driver process
	setuid(II)	set process user ID
	sgen(e)	system generation program
shift(I)	adjust	Shell arguments
sh(I)		shell command programming language
exec in sh(I)	execute within	shell
	shift(I)	adjust Shell arguments
login(I)		sign onto UNIX
kill(II)	send	signal to a process
	signal(II)	catch or ignore signals
signal(II)	catch or ignore	signals
trap in sh(I)	catch	signals
dirname(I)	strip	simple filename
lex(VI)	generate programs for	simple lexical tasks
	sin, cos(III)	trigonometric functions
growseg(a)	increase or decrease the	size of a segment
size(I)		size of an object file
fsize(c)	get	size of file
sizeseg(a)	get	size of segment
ftrunc(c)	truncate file to given	size
	size(I)	size of an object file
	sizeseg(a)	get size of segment
setbuf in newio(III)	set buffer	size
wdleng in newio(III)	find machine word	size
psleep(b)	put process to	sleep on bit pattern
sleep(a)	set a bit pattern to	sleep on
	sleep(a)	set a bit pattern to sleep on
	sleep(I)	suspend execution for an interval
	sleep(II)	stop execution for interval
wakeup(a)	wakeup all processes	sleeping on a pattern
pwakeup(b)	wake up processes	sleeping on bit pattern
qsleep(f)	stop execution for	small interval
spline(VI)	interpolate	smooth curve
isnp(d)		snap i-node contents
	sndmsgfrom(a)	send a message from a process
sno(VI)		Snobol interpreter
	sno(VI)	Snobol interpreter
sort(I)		sort or merge files
	sort(I)	sort or merge files
qsort(III)	quicker	sort
the proc. virtual addr. space...	spacalloc(a)	allocate space for segment; add it to
falloc(c)	allocate contiguous	space for a file
falloc(f)	allocate	space for contiguous file
space...spacalloc(a)	allocate	space for segment; add it to the proc. virtual addr.
add a segment to the process address	space...addseg(a)	
a segment from a process virtual address	space...dropseg(a)	drop
falloc(d)	allocate contiguous file	space
getarg(b)	get argument from SUP address	space
isspace in newio(III)	test for	space
putarg(b)	put argument into SUP address	space
a segment from a process virtual address	space...rmovseg(a)	remove
segment; add it to the proc. virtual addr.	space...spacalloc(a)	allocate space for
fork(II)		spawn new process
mknod(c)	make a directory or a	special file
mknod(II)	make a directory or a	special file
mknod(VIII)	build	special file

openi(c) open file	specified by inode number
setfil(III)	specify Fortran file name
tty(IV) interface to low	speed asynchronous devices including typewriters
spell(VI) find	spelling errors
	spell(VI) find spelling errors
	spline(VI) interpolate smooth curve
split(I)	split a file into pieces
	split(I) split a file into pieces
lpr(I) line printer	spooler
	sprintf in newio(III) print formatted
	sqrt(III) square root function
sqrt(III)	square root function
rand,	rand(III) random number generator
	scanf in newio(III) input conversion
	sswap(a) remove non-swap status from a segment
pstart(a)	start process
startty(e)	start up tty
setmap(a) set access, mode and	starting segmentation register
	startty(e) start up tty
boot procedures(VIII) MERT	startup
	stat(c) get file status
kdmp(e) dump system	state into core file
tdmp(e) dump system	state into core file
getty(a) get	state of tty driver process
setty(a) set	state of tty driver process
	stat(II) get file status
	statio(f) get status of asynchronous I/O
pswap(a) remove non-swap	status from a process
sswap(a) remove non-swap	status from a segment
	statio(f) get status of asynchronous I/O
	fstat(c) get status of open file
	fstat(II) get status of open file
gtty(II) get typewriter	status
ps(I) process	status
stat(c) get file	status
stat(II) get file	status
	stime(II) set time
sleep(II)	stop execution for interval
qsleep(f)	stop execution for small interval
stoptty(e)	stop tty
tabs(VII) set tab	stops
	stoptty(e) stop tty
icheck(VIII) file system	storage consistency check
	streat in newio(III) concatenate strings
	streq in newio(III) compare strings
	streq in newio(III) copy string
	sed(I) stream editor
strlen in newio(III) obtain	string length
fgets in newio(III) get	string
fputs in newio(III) put	string
gets in newio(III) get	string
puts in newio(III) put	string
streat in newio(III) concatenate	strings
streq in newio(III) compare	strings
streq in newio(III) copy	string
basename(I)	strip filename affixes
dirname(I)	strip simple filename
	strip(I) remove symbols and relocation bits
	strlen in newio(III) obtain string length
include(V) system data	structure definitions file
	stty(I) set terminal options
	stty(II) set mode of typewriter
newio(III) a new IO	subroutine package
exit in newio(III) exit from	subroutine
Intro(III) INTROD. TO	SUBROUTINES
sum(I)	sum file

	sum(I) sum file
du(I)	summarize disk usage
	sunswap(a) make a segment non-swap
getarg(b) get argument from	SUP address space
putarg(b) put argument into	SUP address space
sync(VIII) update the	super block
update(VIII) periodically update the	super block
sync(c) update	super-block
sync(II) update	super-block
kppkill(e) terminate a kernel process	(superuser)
ppkill(e) terminate a process	(superuser)
run(e) run an environment	(superuser)
Intro-a(a) INTRO. TO	SUPERVISOR EMT TRAPS
sleep(I)	suspend execution for an interval
pause(II)	suspend execution indefinitely
	su(VIII) become privileged user
getcsw(a) get console	switch register setting
csw(II) read console	switches
strip(I) remove	symbols and relocation bits
	sync(c) update super-block
du(IV) DU-11	synchronous communication device
	sync(II) update super-block
	sync(VIII) update the super block
messages...perror,	sys_errlist, sys_nerr, errno(III) system error
perror, sys_errlist,	sys_nerr, errno(III) system error messages
	sysproc(a) system process
	sysproc(f) system ports
indir(II) indirect	system call
Intro-f(f) INTRO. TO MERT UNIX	SYSTEM CALLS
fsck(VIII) file	system consistency check and interactive repair
check(VIII) file	system consistency check
crash(VIII) what to do when the	system crashes
include(V)	system data structure definitions file
dcheck(VIII) file	system directory consistency check
dump(VIII) incremental file	system dump
sys_errlist, sys_nerr, errno(III)	system error messages...perror,
sgen(e)	system generation program
	system in newio(III) execute command
init(c) initialize file	system manager
sysproc(f)	system ports
sysproc(a)	system process
Intro(VIII) INTROD. TO	SYSTEM PROGRAMS
restor(VIII) incremental file	system restore
(System Scheduler)term(c) to	system scheduler: terminate a process
terminate a process...	(System Scheduler)term(c) to system scheduler:
kdump(e) dump	system state into core file
tdmp(e) dump	system state into core file
icheck(VIII) file	system storage consistency check
mtab(VII) mounted file	system table
getime(b) get	system time
setime(b) set	system time
Intro-d(d) INTRO. TO FILE	SYSTEM UTILITY PROGRAMS
fs(g) format of MERT file	system volume
fs(V) format of UNIX file	system volume
sdh(IV) DH11 for Satellite Processor	System
tabs(VII) set	tab stops
mtab(VII) mounted file system	table
add a segment id to the process segment	table...openseg(a)
tbl(VI) format	tables for nroff or troff
tabs(VI) set	tabs on terminal
	tabs(VI) set tabs on terminal
	tabs(VII) set tab stops
	tail(I) deliver the last part of a file
atan, atan2(III) arc	tangent function
dump(V) incremental dump	tape format
tp(V) DEC/mag	tape formats

mtm(I) magnetic	tape manipulation
rew(I) rewind	tape
generate programs for simple lexical	tasks...lex(VI)
	tbl(VI) format tables for nroff or troff
	TC-11/TU56 DECTape
tc(IV)	tc(IV) TC-11/TU56 DECTape
	tdmp(e) dump system state into core file
	tee(I) pipe fitting
	tf(IV) Telefile disk driver
	tell(II) get file offset
mktemp(III) make a unique named	temporary file
tty(I) get	terminal name
stty(I) set	terminal options
interpret extended character set on GSI	terminal...gsi(VI)
lnxx(III) return name of current	terminal
neqn(I) typeset mathematics on	terminal
tabs(VI) set tabs on	terminal
terminate all processes associated with a	terminal...tkill(e)
kpskill(e)	terminate a kernel process (superuser)
(P-Mgr)MSTERM(c)	terminate a process and dump core
pskill(e)	terminate a process (superuser)
kill(I)	terminate a process
(Mem-Mgr)term(c) to memory manager:	terminate a process
Scheduler)term(c) to system scheduler:	terminate a process...(System
tkill(e)	terminate all processes associated with a terminal
exit(I)	terminate command file
exit(II)	terminate process
return(I)	terminate profile or interrupt processing routine
wait(II) wait for process to	terminate
(P-Mgr)pswait(c) message at	termination of process-mgr-created process
qwait(f) check for child process	termination
wait in sh(I) wait for process	termination
isalpha in newio(III)	test for alphabetic
islower in newio(III)	test for lower case
isdigit in newio(III)	test for numeric
isspace in newio(III)	test for space
intss in newio(III)	test for tss or batch
isupper in newio(III)	test for upper case
	test(f) condition command
ed(I)	text editor
rform(VI) reformat	text file
nroff, troff(I)	text formatters
nroff, troff(I)	text formatters
	tf(IV) Telefile disk driver
cubic(VI)	three dimensional tic-tac-toe
sendport(a) send message	through port
cubic(VI) three dimensional	tic-tac-toe
ttt(VI) the game of	tic-tac-toe
time(I)	time a command
ktime(e) give detailed kernel	time of a command
profil(II) execution	time profile
localtime, gmtime(III) convert date and	time to ASCII...ctime,
gettime(a) get	time
gettime(b) get system	time
	time(I) time a command
	time(II) get date and time
ptimer(b) set	time-out value for process
timleft(b) get	time-out value for process
toutset(a) set	time-out
alarm(II) activate alarm clock	timer
read(I) read one line at a	time
setime(a) set	time
setime(b) set system	time
	times(II) get process times
stime(II) set	time
times(II) get process	times

time(II) get date and	time
terminal...	timeleft(b) get time-out value for process
tm(IV)	kill(e) terminate all processes associated with a TM-11/TU-10 magtape interface
tmpnam in newio(III) create	tmac(VI) ms macros for formatting manuscripts
	tm(IV) TM-11/TU-10 magtape interface
	tmp name
	tmpnam in newio(III) create tmp name
	tolower in newio(III) translate to lower case
	toupper in newio(III) translate to upper case
	toutset(a) set time-out
	tp(I) manipulate DECTape and magtape
	tp(V) DEC/mag tape formats
tolower in newio(III)	translate to lower case
toupper in newio(III)	translate to upper case
kprc(g) kernel process	translation file
tr(I)	transliterate
	trap in sh(I) catch signals
rti(a) return from	trap
Intro-a(a) INTRO. TO SUPERVISOR EMT	TRAPS
	tr(I) transliterate
	trigonometric functions
sin, cos(III)	Troff and Eqn constructs
deroff(VI) remove	troff(I) text formatters
nroff,	troff(I) text formatters
nroff,	troff
tbl(VI) format tables for nroff or	truncate file to given size
ftrunc(c)	tss or batch
intss in newio(III) test for	tty(VI) the game of tic-tac-toe
	tty driver process
getty(a) get state of	tty driver process
setty(a) set state of	tty, post; udata
loginfo(II) login inform.: name, dir,	TTY-37 type-box
greek(VII) graphics for extended	tty(I) get terminal name
	tty(IV) interface to low speed asynchronous devices
including typewriters...	tty
startty(e) start up	tty
stoptty(e) stop	ttys(V) typewriter initialization data
	two files
cmp(I) compare	two files
comm(I) print lines common to	type -1 (acknowledgement)
mreceive(a) get a message of	type between given limits
mgetlim(a) get a message of	type-box
greek(VII) graphics for extended TTY-37	type
dqtype(b) dequeue a particular message	type
gettype(a) get a message of given	type
mgettype(a) get a message of given	type
msgtype(a) get a message of given	type
	typeset mathematics on terminal
neqn(I)	typeset mathematics
eqn(I)	typewriter initialization data
ttys(V)	typewriter mode
getty(VIII) set	typewriter status
gty(II) get	typewriter
mesg(III) write message on	typewriter
stty(II) set mode of	typewriters...tty(IV) interface
to low speed asynchronous devices including	typo(I) find possible typos
	typos
typo(I) find possible	udata...loginfo(II)
login inform.: name, dir, tty, post;	UID
getpw(III) get name from	ulockid(a) decrement lock count of segment
	ulockseg(a) decrement lock count of segment
	umount(c) dismount file system
	umount(II) dismount file system
	umount(VIII) dismount file system
	unblkseg(a) unblock a named segment
unblkseg(a)	unblock a named segment

	ungetc in newio(III) push character back
	uniolock(b) unlock segment for I/O
	uniq(I) report repeated lines in a file
mktemp(III) make a	unique named temporary file
	units(VI) conversion program
rm(I) remove	(unlink) files
	unlink(c) remove directory entry
	unlink(II) remove directory entry
uniolock(b)	unlock segment for I/O
sync(c)	update super-block
sync(II)	update super-block
sync(VIII)	update the super block
update(VIII) periodically	update the super block
	update(VIII) periodically update the super block
isupper in newio(III) test for	upper case
toupper in newio(III) translate to	upper case
du(I) summarize disk	usage
mkpt(VIII) make prototype file for	use by mkfs
xusr(e) extract	user core image from process core dump
adduser(a) increment	user count on a process
getuid(II) get	user identifications
setuid(II) set process	user ID
utmp(V)	user information
wtmp(V)	user login history
ldu(e) load a	user process with public libraries
getseg(f) get	user segment
mail(I) send mail to designated	users
setdspac(a) set	user-supervisor d-space bits
su(VIII) become privileged	user
wall(I) write to all	users
wall(VIII) write to all	users
write(I) write to another	user
lock(f) semaphores	(USG Version)
msg(f) send and receive messages	(USG Version)
lseek(III) seek	using a long offset
Intro-d(d) INTRO. TO FILE SYSTEM	UTILITY PROGRAMS
Intro-e(e) INTRO. TO MERT	UTILITY PROGRAMS
	utmp(V) user information
	uucp(VI) unix-to-unix copy
ptimer(b) set time-out	value for process
timleft(b) get time-out	value for process
abs, fabs(III) absolute	value
call, lcall,	vcall(II) create and execute a new process
attach(a) attach process to interrupt	vector
detach(a) detach process from interrupt	vector
lint(I) a C program	verifier
space for segment; add it to the proc.	virtual addr. space...spacalloc(a) allocate
dropseg(a) drop a segment from a process	virtual address space
rmovscg(a) remove a segment from a process	virtual address space
iomap(b) map segid/offset to	virtual address
fs(g) format of MERT file system	volume
fs(V) format of UNIX file system	volume
waitev(f)	wait for an event
cwait(a) conditional	wait for event
wait in sh(I)	wait for process termination
wait(II)	wait for process to terminate
	wait in sh(I) wait for process termination
	waitev(f) wait for an event
	wait(I) await completion of process
	wait(II) wait for process to terminate
pwakeup(b)	wake up processes sleeping on bit pattern
wakeup(a)	wakeup all processes sleeping on a pattern
	wakeup(a) wakeup all processes sleeping on a pattern
	wall(I) write to all users
	wall(VIII) write to all users
	wc(I) word count

	wdleng in newio(III)	find machine word size
crash(VIII)		what to do when the system crashes
	what(I)	identify SCCS files
	who(I)	who is on the system
	who(I)	who is on the system
exec in sh(I)	execute	within shell
	wc(I)	word count
wdleng in newio(III)	find machine	word size
	getw in newio(III)	get word
	putw in newio(III)	put word
hyphen(VI)	find hyphenated	words
	pwd(I)	working directory name
chdir, cd(I)	change	working directory
	chdir(c)	change working directory
	chdir(II)	change working directory
lock a segment in memory and set		write back...alockseg(a)
	putchar, flush(III)	write character
	mesg(III)	write message on typewriter
	write(II)	write on a file
	wall(I)	write to all users
	wall(VIII)	write to all users
	write(I)	write to another user
fwrite in newio(III)		write to file
	write(c)	write to file
	write(c)	write to file
	write(I)	write to another user
	write(II)	write on a file
	writeseg(a)	force a segment to be written back
open(II)	open for reading or	writing
writeseg(a)	force a segment to be	written back
	wtmp(V)	user login history
	wump(VI)	the game of hunt-the-wumpus
	xusr(e)	extract user core image from process core dump
	yacc(I)	yet another compiler-compiler
	yacc(I)	yet another compiler-compiler
permit(a)	run process at priority	zero