

Table of contents

4-	1	Data areas
5-	1	Tables of commands and options
7-	1	PAUSE command
7-	40	DISPLAY command
8-	1	ASSIGN command
10-	1	DEASSIGN command
11-	1	ALLOCATE command
12-	1	DEALLOCATE command
13-	1	REMOVE command
15-	1	INSTALL Command
22-	1	MOUNT command
25-	1	DISMOUNT command
26-	1	INITIALIZE command
26-	2	SQUEEZE command
26-	3	FORMAT command
28-	1	MONITOR command
29-	1	SPOOL command
33-	1	SEND command
35-	1	ACCESS command
37-	1	DETACH command
39-	1	DATE command
40-	1	TIME command
41-	1	RESET command
42-	1	OFF command
45-	1	KILL command
45-	8	SUSPEND command
45-	15	RESUME command
46-	1	BOOT and \$STOP commands
47-	1	\$\$SHUTDOWN command

```

1          .TITLE  TSKM2A -- Keyboard command routines
2          .ENABL  LC
3          .DSABL  QBL
4 000000   .CSECT  TSKM2A
5 000000   TSKM2A:
6          ;
7          ; TSKM2A is the portion of TSKMON that contains the actual code
8          ; to implement each keyboard command that is processed by TSKMON.
9          ;
10         ; Copyright 1978, 1979, 1980, 1981, 1982, 1983, 1984.
11         ; S&H Computer Systems, Inc.
12         ; Nashville, Tennessee
13         ;
14         ; Macro calls
15         ;
16         .NCALL .CSISPC, .TTOUTR, .SRESET, .CRRG, .ELRG
17         .NCALL .READW, .TTYIN, .TTYOUT, .PURGE, .TWAIT
18         .NCALL .CSIEN, .SAVEST, .REOPEN
19         .NCALL .GTLIN, .GTIN, .DATE, .SPFUN
20         .NCALL .PRINT, .CLOSE, .LOOKUP
21         .NCALL .WRITW, .ENTER, .EXIT
22         .NCALL .SERR, .HERR, .FPROT, .QVAL, .PVAL
23         ;
24         ; Global definitions
25         ;
26         .GLOBL TSKM2A, CMDHD, CMDOFF, KDOCIN
27         .GLOBL PLOAD, CMDFRM, CMDDSN, STLGCN, DATTIM, PRGALL
28         .GLOBL DLCEMT, ALCDEV, CMDINS, CMDSPN, CMDRSM, CMDYEL
29         .GLOBL CMDRSY, CMDRST, CMDSND, CMDFMT, CMDDAT
30         .GLOBL CMDTIM, CMDMNT, CMDMON, CMDDMT, CMDDSP, CMDASN
31         .GLOBL CMDALC, CMDDL, OPRCMD, CMDSPO, CMDKIL, CMDREM
32         .GLOBL CMDPAU, CMDDET, CMDACC, CMDINI, CMDSQZ
33         .GLOBL CMDBOT, CMDSHT
34         ;
35         ; Global references
36         ;
37         .GLOBL TM$SA1, TM$SA2, TM$SA3, TM$SA4, SKPSPC, SKPDLM
38         .GLOBL SWPCHN, SEQCHN, INDTSV, LWINDO, $NOWIN, LSW11
39         .GLOBL $SCCA, AF$CCA, AF$CF, EM$NUC, PO$SYS, AF$NPW
40         .GLOBL AF$DUP, AF$IND, AF$UCL, AF$SET, INSSRC, FORCEO
41         .GLOBL CFSTOP, CFSTRT, EM$ALC, EM$NFW, EM$SND, EM$DNR
42         .GLOBL CFSQEZ, TOOLNG, SDFHD, AF$TPO, $TRNSP, EM$CSE, WLDNAM
43         .GLOBL EM$NAL, AF$PLK, AF$DBG, FIXPRV, EM$LDI, EM$NLN
44         .GLOBL PO$MEM, PO$NFW, PO$BYP, PO$ALC, CKSYPV, CKACQJ, PRIVSO
45         .GLOBL PFSO, PFCO, PVNPW, EM$NAD, PO$DBG, PRIVCO, II$PRV, II$NPV
46         .GLOBL CLRPRV, EM$FNI, EM$ITF, INPADR, INPEMT, OPTLST
47         .GLOBL CDBUF, CDQET, INGADR, INGEMT, IIBUF, PRVOPT
48         .GLOBL CINFLQ, $VNQTT, SDNAME, DMYDEV
49         .GLOBL CORUSR, LSW, $CTRLQ, SERFLG, IOABFL
50         .GLOBL LSTHL, LCLUNT, FSTIOL, LSTIOL, HANCHN
51         .GLOBL MAXALC
52         .GLOBL AD$DVU, AD$JOB, AD$SZ, LSTSPL, SOPALC
53         .GLOBL NEDCHR, LOUTIR, LINIR, LINRTS, CLOTIR
54         .GLOBL LCDTYP, SOPDAT, SOPTIM, PO$LQK
55         .GLOBL UTRPAD, JSWLOC, ERRLOC, MAXMEM, MAXPRI
56         .GLOBL PO$MEM, PO$DET, EM$DET, PO$SND, EM$MPV, P2$WRL
57         .GLOBL USRSTK, $KINIT, CFSTK, MXJMEM, DFJMEM, LPARNT

```

```

58 . GLOBL SPUBUF, SXBPNT, MXJADR
59 . GLOBL TMTOTH, TMTUTL, TMUSRH, TMIOWH, LDMNT
60 . GLOBL R, GSIZ, RS, GBL, RS, PVT, R, NAME, RS, CRR, RS, EGR
61 . GLOBL EM#UAR, EM#UER, R, GSTS, RSTPRV
62 . GLOBL TMSWTH, TMIDLH, TMIOH, TMSWPH, LDCLN
63 . GLOBL EM#NPD, EM#IAD, EM#ATF, EM#IAD, EM#NLD
64 . GLOBL EM#SSY, EM#NID, QHDMS1, QHDMS2, EM#NSF, EM#OLO
65 . GLOBL GENMON, JS$OFF, LMONHD, AF$MEM, AF$BYA, SMONHD
66 . GLOBL EM#DAA, EM#DIU, CHKALC, $CHACT, $STSNG
67 . GLOBL WILDFL, $NOIN, $NOWTT, $HITTY
68 . GLOBL TECO, EDIT, KED, K52, $1STLG, $DIBOL, AF$SCA
69 . GLOBL LINBUF, LINNXT, LSTACT, PRGTOP, PRGSIZ
70 . GLOBL CXTAG, FSI IOL
71 . GLOBL LAFSIZ, LFWLIM, LINCUR, NUMON, ILSW2
72 . GLOBL $DBKMN
73 . GLOBL $CARUP, DOASCN, UKMNAM, $UKMON, LSW9
74 . GLOBL LSUCF, $CCLRN, EM#NUK
75 . GLOBL KL3CLR, $PRGLK, LSW5, PVON
76 . GLOBL ALDEMT, TALEMT
77 . GLOBL LSTD, FSTD, $DETC, UMSYTP
78 . GLOBL $DISCN, LPROJ, LPROG, LUNAME
79 . GLOBL LCPUI, LCPULO, LCONTM, $CTRLS, $SPLJB
80 . GLOBL STPFLG, TOTON, USPLCH, SPLCHN
81 . GLOBL S$MSWT, CFBUF, CFEND, CCLSAV, KMNCHN
82 . GLOBL MINTIM, LSECPT, MAXSEC, $EMTTR, VCSHNB
83 . GLOBL OKFILE, OKFEND, $CLTST, UCISPC, MHNSIZ
84 . GLOBL CASTBR, CASCBR, CASTBW, CASCUP, MHNSMS
85 . GLOBL CASTRO, CASTWO, CLTOTL, CLSFSP
86 . GLOBL PHYMEM
87 . GLOBL LJSW, CTRLTT, NEWJSW, JSTKND, VIMAGE
88 . GLOBL USTART, GENTOP, BOTDEV, BOTUNI, CSHALC
89 . GLOBL $CTRLC, LSW2, $INKMN, CHAIN, UFORM
90 . GLOBL $SQO, $SQO3, LITIME
91 . GLOBL MAXASN, $CFABT, INDSTA, INDERR
92 . GLOBL LNBLKS, CXTBAS, CXTWDS, UHIMEM
93 . GLOBL ASNTBL, $DILUP, CSHDEV, CSHDVN, LNSBLK
94 . GLOBL AT$LOG, AT$SIZ, AT$DEV, AT$FIL, AT#$SZ, AT$EXT
95 . GLOBL ASNEND, LSW3, LSW2S, $DUPRN
96 . GLOBL $FORM, $TAB, LSCCA, $CFSOT, LOFSPC, R50COM
97 . GLOBL $PAGE, $SCOPE, $ECHO, $LC, $SETRN, EM#FOE
98 . GLOBL UCHAN, $FORMO, $CFALL, $CFDCC, $CFCL
99 . GLOBL LNPRIM, LNMAP, CW$50H, CONFIG, $SUCF
100 . GLOBL $DOOFF, NUCHN, LRBFIL, CFIND
101 . GLOBL C, CSW, C, DEVQ, C, SBLK, NLINES
102 . GLOBL CD$NAM, CD$DVU, CD$BAS, CD$JOB, CD#$SZ, CD#$UB
103 . GLOBL LTSCMD, LNSPAC, CFNEST, VSWPFL, UCLNAM
104 . GLOBL $CFOPN, CFSEND, PBFEND, CFSP, $TTGAG
105 . GLOBL UFPTRP, SDSFCB, SD$DEL, CFLFL4, $UCLCF
106 . GLOBL SDFLAG, SD$FLK, SD$WFM, SDFORM, $UCLRN
107 . GLOBL SDBUF1, SDBLK, SPLND, LD$RON, $UCLCM, $UCLCL
108 . GLOBL LDNAME, LDSIZE, LDFLAG, LDBASE, LDPDEV
109 . GLOBL LSW8, $SQO1, $SQO1A, $SQO1B, $SQO1C, $SQO2, $SQIIO, $SQHIO
110 . GLOBL $DEFER, CFCHAN, SCHAIN, LDDEVX, $SGALL
111 . GLOBL CFPNT, CFBLK, $QUIET
112 . GLOBL LSW4, KL4CLR, SDSKIP, SDBU, SD$BAK
113 . GLOBL $INCOR, $KED, VQUN1B, VINTIO, VQUN1C
114 . GLOBL SFFORM, SD$SNG, SFNMBL, NFRESB

```

115 . GLOBL SD#HLD, CURPRM, PRMPNT
 116 . GLOBL LSTPRM, PRMBUF, PRMEND, CFSPND
 117 . GLOBL CFHOLD, LOGDVU, LOGBAS
 118 . GLOBL LCOL, \$QTSET, \$TECO, CD\$TOP, LOGCHK
 119 . GLOBL \$WILD, ERRSEV, UERSEV, PASLIN, LOGBAS, LOGDVU
 120 . GLOBL LSTPL, SDCB, SDCBND, VQUANO, VQUAN3
 121 . GLOBL VQUAN1, VQUN1A, VQUAN2, VHIPCT, VQUANO, VQUAN3
 122 . GLOBL DCTRD, DCCRD, DCTWR, DCCWR, ASNSRC
 123 . GLOBL VCORTM, KMPRMT, MXPRMT
 124 . GLOBL RDB, RDBEND, RT\$NAM, RT\$\$SZ, CLDEVX, SDDVU
 125 . GLOBL SDCBSZ, LSTSL, LSTATE
 126 . GLOBL TKIVAL, CINDAT, SYSDAT, SYTIMH, SYTIML
 127 . GLOBL BASMAP, LOMAP, HIMAP, JCXPGS
 128 . GLOBL SMRSIZ, SRTSIZ, CSHSIZ, TK1SEC
 129 . GLOBL TSXLN, TSXSIT, GRT1, TRGRET, LICTXT, SUPCOD, NAMTOP
 130 . GLOBL SYINDX, SYUNIT, NUMDEV, PNAME
 131 . GLOBL OF\$DEV, OF\$UNT, OF\$FIL, OF\$FLG, SYNAME
 132 . GLOBL OF\$\$SZ, OT\$RON, RESDEV, \$TAPE
 133 . GLOBL KMNBAS, ODTBAS, \$CTRLD
 134 . GLOBL LSW6, \$SNWTT, PF\$SYS, PF\$IOW, \$DEBUG
 135 . GLOBL \$INDDF, \$INDRN, IN\$ACT, IN\$CNT, IN\$CMD, INDSAV
 136 . GLOBL \$PHONE, INVEC, LMXLN, MXVEC, \$INIT, \$DEAD, \$HARD
 137 . GLOBL ITRMTP, LMXPRM, LSW7, CFSTS, CF\$IND, CF\$QUT
 138 . GLOBL CFABLV, MONVEC, LBSPRI, MAXPRI, MXJPRI, LPRI
 139 . GLOBL LOGCHN, LOGFLG, LOGPTR, LOGBUF, LOGBLK
 140 . GLOBL LF\$OPN, LF\$WRT, UCLBLK, UCLDAT
 141 . GLOBL CSHHD, UC\$NDC, UC\$MDC, CVTUC
 142 . GLOBL CMDBUF, PAUMSG, RDCMD, DKSAY, SYSAV, CVTTAB, SEARCH
 143 . GLOBL INVOPT, FKILL, ABRTCF, INDABT, ACRFN, XAREA, FILNAM, NOPRG, FPRINT
 144 . GLOBL PUSHCF, TRMSTR, FILNAM, R5ODIR, R5OSY, R5OIND, R5OSAV
 145 . GLOBL INDACT, R5ODUP, R5OPIP, R5OKED, R5OK52, R5OKEX, R5OTSX, R5OUCL
 146 . GLOBL BLKO, RDERM, R5OVIR, NOSTRT, OVRCOR
 147 . GLOBL BADSAV, LDNAM, NOPRG, NDCIN, SIZVAL, ASKLN, BADCMD, KCSIBF
 148 . GLOBL ASDEX, KCSIMS, ASNQVF, QTRD50, R5OBUF, R5OLD0, MNTDEV, DMTARG
 149 . GLOBL DEADEV, CHKMNT, CHKMTX, INFOMT, NOFLAG, MTOPHD, INVOPT, ILLCMD
 150 . GLOBL R5OLD, INVLDN, R5ODSK, ACRFIL, BDFNAM, LOGASN, MNTFUL, R5OLD7
 151 . GLOBL TBLOVF, SETHD, CSIMS2, CKPRIV, R5ONO, AMBOPT, ACRDEC
 152 . GLOBL MAXAVL, PRTDEC, DEVUNT, PNAME, HANIDX, HNBUF
 153 . GLOBL ACROCT, HANBSY, CSIMS1, MISSEQ, NOIND, POPCF
 154 . GLOBL BADPMT, BADPRI, TOTXT, CRLF, HIPRI, STLQHD, LOGCLS, R5OLOG
 155 . GLOBL BDLGQP, SPLHLA, LDOPHD, PRTFIX, PRTSPC
 156 . GLOBL DLTXT, OCTFIX, PRTTTP, NATXT, NOTXT, YESTXT, NINTXT
 157 . GLOBL PRTUNM, SYHD1, SYHD2, PRTLN, DETTXT, RNMS
 158 . GLOBL SWPTX, LOCKTX, PRTDC3, KBMSG, DIVIDE, PRTDC2
 159 . GLOBL COLOO, CPUAH, CPUAL, PRTTMV, NOFIL, CMDBUF, CALUCL
 160 . GLOBL NOUDC, DEVHD1, ASNHD1, ASNHD2, SHMTH1, SHMTH2, PRTTMD
 161 . GLOBL CVDVNM, PRTBUF, PRTFNM, NONEMS, NODAT, NOLDMT
 162 . GLOBL SUBARO, EDTFIL, RONTXT, NOTAVL, KBTX, MNFLGS, MNBPC
 163 . GLOBL DELSPC, MNBASE, MNTOP, MONHD, MONAR1, NOPMGN, PMBUSY, MONAR2
 164 . GLOBL NSWPMS, MAXMTX, CURMTX, CHKDLM, SPLHD, AMBOPT, INVOPT
 165 . GLOBL DEVIDL, COAL, ALDEX, COAD, SPACTV, SPWFM, DEVIDL, SPSNG
 166 . GLOBL COAL, ALDEX, ALDBLK, COAD, SPACTV, SPWFM, DEVIDL
 167 . GLOBL SPSNG, SPFUL, SPCF, SPFLK, NOFIL, SPQEMT, NOOPTT
 168 . GLOBL BDLIN, MSGBUF, NUMTB1, MSGEND, NOTON, GAGMSG
 169 . GLOBL YELEM, LINFRE, DJABMS, DLMSG, INVTIM, DMTALL
 170 . GLOBL SHTMSG, AUTHFN, SPLACT, DOSTOP, OFFEMT, KILEMT, UPTMMS
 171 . GLOBL TMTOTH, DIVSOR, TMTOTL, PRTPCT

```
172 .GLOBL OTHRON, SPLPND, STPASK, SRTSMS
173 .GLOBL SIZEMT, ASNOVF, INVLDM, CSIMS4, MNTARG, HUPARG, R50TT
174 .GLOBL KMNNAM, CCLNAM, OTRMNT, CHKDEV, DMTSUB, CMDCCCL
175 .GLOBL SHOHD, SUBTXT, MNTTXX, SRTTXX, TOTMMS, UMSSMS, SSRMAP
176 .GLOBL TSXSMS, USRMMS, JCXSMS, DZTXX, OCTPRT, SJEMT, RJEMT
177 .GLOBL PRTR50, PRTDAT, PRTTOD, PRTTIM, INVDEV, ALFN, R50DK
178 .GLOBL DETHD, DETARG, RUNMS, NOFRDL, R50MON, INV DAT, MUL32, COAF
179 .GLOBL AR$PRJ, AR$PRG, AR$CON, AR$CNT, AR$CPH, AR$CPL, AR$UNM
180 .GLOBL AR$DMY, AR$$SZ, ARNRPB, $SLQN, $SLTTY, $SLLET
181 .GLOBL PRTWRN, SLMXLN, VLDSYS, $LOFCF, CSHMSG, CCSRV
182 .GLOBL INSTBL, INSTBN, AF$NOW, AF$HIE, AF$NOI, $NOINT, II$$SZ
183 .GLOBL AF$IOP, $RNIOP, $RNMLK, II$NAM, II$FLG, II$PRV, II$NPV
184 .GLOBL NMSGBF, $NOABT
```

```
1
2      ;
3      ; Assembly constants
4      ;
5      000012      LF      =      12      ; LINE FEED
6      000015      CR      =      15      ; CARRIAGE RETURN
7      000040      BLANK   =      40      ; ASCII SPACE
8      000007      BELL    =      07      ; ASCII BELL
9      000011      TAB     =      11      ; HORIZONTAL TAB
10     000014      FF      =      14      ; FORM FEED
11     000400      BLKWDS  =      256     ; # OF WORDS IN DISK BLOCK
12     132500      WLDNAM  =      132500  ; RAD50 /*/ (WILDCARD)
```

```

1      ; -----
2      ; Macro to cause a fatal error message to be printed.
3      ;
4      .MACRO FERR MSG
5      MOV R5, -(SP)
6      MOV MSG, R5
7      CALL FPRINT
8      MOV (SP)+, R5
9      .ENDM FERR
10     ;
11     ; -----
12     ; Macro to print a fatal error message, clean up
13     ; and then jump to RDCMD.
14     ;
15     .MACRO FABORT MSG
16     MOV MSG, R5
17     JMP FKILL
18     .ENDM FABORT
19     ;
20     ; -----
21     ; Macro to print a warning message
22     ;
23     .MACRO FWARN MSG
24     MOV R5, -(SP)
25     MOV MSG, R5
26     CALL PRTWRN
27     MOV (SP)+, R5
28     .ENDM FWARN
29     ;
30     ; -----
31     ; Macro to start a standard option table.
32     ; Name = 1 to 4 character table name.
33     ; NA = Number of arguments per table entry.
34     ;
35     .MACRO TBLDEF NAME, NA
36     NARGS = NA
37     .CSECT CMDV2A
38     NAME /HD: .WORD 2*NA
39     .ENDM TBLDEF
40     ;
41     ; -----
42     ; Macro to enter an option text name and a set of parameters
43     ; into the currently open table.
44     ; STRNG = Ascii name
45     ; A,B,C = Set of option parameters to store in table with name.
46     ;
47     .MACRO CMDDEF STRNG, A, B, C
48     .CSECT NAME2A
49     L =
50     .ASCIZ /STRNG/
51     .CSECT CMDV2A
52     .WORD L ; POINTER TO NAME STRING
53     .WORD A
54     .IIF GE, <NARGS-2> .WORD B
55     .IIF GE, <NARGS-3> .WORD C
56     .ENDM CMDDEF
57     ;

```

58
59
60
61
62
63
64
65

```
-----  
; Macro to end a set of table entries.  
;  
; .MACRO TBLEND  
; .CSECT CMDV2A  
; .WORD 0  
; .CSECT TSKM2A  
; .ENDM TBLEND
```

Data areas

```

1          .SBTTL  Data areas
2          ;-----
3          ; Data areas:
4          ;
5 000000  000000  SQZDEV: .WORD  0          ;Name of device being squeezed
6 000002  000000  ALC1DV: .WORD  0          ;Name of 1st device on allocation list
7 000004  073634  R50SET: .RAD50 /SET/
8 000006  102700  R50UP:  .RAD50 /UP /
9 000010          REMRDB: .BLKW  5
10 000022  003344  R50ADD: .RAD50 /ADD/
11 000024  014724  021145  R50DEL: .RAD50 /DELETE/
12 000030  070525  R50REM: .RAD50 /REM/
13 000032  000000  IATTRB: .WORD  0
14          ;
15          ; Byte data
16          ;
17 000034          000  DMTNLC: .BYTE  0
18 000035          000  OFFNWF: .BYTE  0
19 000036          000  YELFLC: .BYTE  0
20          .EVEN
21          ;
22          ; Emt to delete a spool file
23          ;
24 000040          000  126  DELEMT: .BYTE  0,126
25 000042  000007          .WORD  7
26 000044  000000          .WORD  0          ;File ID goes here
27          ;
28          ; Emt to set spool HOLD mode
29          ;
30 000046          000  126  SPHLEM: .BYTE  0,126
31 000050  000015          .WORD  15
32 000052  000000          .WORD  0          ;Address of SDCB
33          ;
34          ; Emt to set spool NOHOLD mode
35          ;
36 000054          000  126  SPNHEM: .BYTE  0,126
37 000056  000016          .WORD  16
38 000060  000000          .WORD  0          ;Address of SDCB
39          ;
40          ; Time to wait for check spooler activity.
41          ;
42 000062  000000  000170  TIMSPL: .WORD  0,60.*2.          ;2 second timer

```

Tables of commands and options

```

1          .SBTTTL  Tables of commands and options
2          ;-----
3          ; Define option switches for the INSTALL ADD command
4          ;
5          TBLDEF  INS, 2
6          CMDDEF  BYPASN, INSATR, AF$BYA
7          CMDDEF  DUP, INSATR, AF$DUP
8          CMDDEF  DEB*UG, INSATR, AF$DBG
9          CMDDEF  HI*GH, INSATR, AF$HIE
10         CMDDEF  IND, INSATR, AF$IND
11         CMDDEF  IO*MAP, INSATR, AF$IOP
12         CMDDEF  IO*PAGE, INSATR, AF$IOP
13         CMDDEF  LOCK, INSATR, AF$PLK
14         CMDDEF  MEM*LOCK, INSATR, AF$MEM
15         CMDDEF  NONI*INTERACTIVE, INSATR, AF$NOI
16         CMDDEF  NOWA*IT, INSATR, AF$NOW
17         CMDDEF  NOWI*NDOW, INSATR, AF$NPW
18         CMDDEF  PRIV*ILEGED, PRVOPT, 0
19         CMDDEF  SC*CA, INSATR, AF$CCA
20         CMDDEF  SETUP, INSATR, AF$SET
21         CMDDEF  SING*LECHAR, INSATR, AF$SCA
22         CMDDEF  T*RANSSPARENT, INSATR, AF$TPO
23         CMDDEF  TSXUCL, INSATR, AF$UCL
24         TBLEND
25
26         ;-----
27         ; Define option switches for the MOUNT command
28         ;
29         TBLDEF  MTOP, 1
30         CMDDEF  WR*ITE, 0
31         CMDDEF  NOW*RITE, 1
32         TBLEND
33
34         ;-----
35         ; Define option switches for the DISMOUNT command
36         ;
37         TBLDEF  DMTQ, 1
38         CMDDEF  LOG, DMTLQ
39         CMDDEF  NOLOG, DMTNLQ
40         CMDDEF  WARN, DMTLQ
41         CMDDEF  NOWARN, DMTNLQ
42         TBLEND
43
44         ;-----
45         ; Define option switches for DETACH command
46         ;
47         TBLDEF  DET, 1
48         CMDDEF  CH*ECK, DETCHK
49         CMDDEF  KI*LL, DETKIL
50         TBLEND
51
52         ;-----
53         ; Options for the MONITOR command.
54         ;
55         TBLDEF  MON, 1
56         CMDDEF  IO*WAIT, PF$IOW
57         CMDDEF  SY*STEM, PF$SYS

```

Tables of commands and options

58 000246
59
60
61
62
63 000066
64 000252
65 000256

TBLEND

```
-----  
; Options for the OFF (BYE, KJOB, LOGOFF) commands.  
;  
TBLDEF OFF,1  
CMDDEF N*QWARN,OFF,NWN  
TBLEND
```

Tables of commands and options

```

1
2 ; -----
3 ; Define options for the SPOOL command
4 ; Entry 1 = Option name
5 ; Entry 2 = Processing routine
6 000066 TBLDEF SPL, 1
7 000262 CMDDEF A*LIGN, SPLALN
8 000266 CMDDEF B*ACKUP, SPLBAK
9 000272 CMDDEF D*ELETE, SPLDEL
10 000276 CMDDEF F*ORM, SPLFRM
11 000302 CMDDEF H*OLD, SPLHLD
12 000306 CMDDEF NOH*OLD, SPLNHL
13 000312 CMDDEF L*OCK, SPLLK
14 000316 CMDDEF M*ULTIPLE, SPLMUL
15 000322 CMDDEF SK*IP, SPLSKP
16 000326 CMDDEF SI*NGLE, SPLSNG
17 000332 CMDDEF ST*ATUS, SPLSTA
18 000336 TBLEND

```

PAUSE command

```

1          .SBTTL  PAUSE command
2          ;-----
3          ; THE PAUSE COMMAND IS USED TO SUSPEND EXECUTION TEMPORARILY
4          ; WHILE PROCESSING A COMMAND FILE.
5          ; WHEN THE PAUSE COMMAND IS FOUND EXECUTION IS SUSPENDED UNTIL
6          ; THE USER ENTERS A CARRIAGE RETURN FROM THE KEYBOARD.
7          ;
8 000066 016767 0000000 0000000 CMDPAU: MOV     CFPNT,CFSPND  ;SUSPEND COMMAND FILE
9 000074 001441          BEQ     9$                ;BR IF COMMAND FILE NOT RUNNING
10 000076 004767 0000000          CALL    CFSTOP      ;Suspend file
11          ;
12          ; Set command-file-abort flag so that command file execution will
13          ; be aborted if control-C is typed in response to the pause.
14          ;
15 000102 032761 0000000 0000000          BIT     ##SCCA,LSW5(R1) ;Is control-C abort override in effect?
16 000110 001003          BNE     3$                ;Br if yes -- Don't abort on control-C
17 000112 052761 0000000 0000000          BIS     ##CFABT,LSW6(R1);Abort command files if ctrl-C typed
18          ;
19          ; If TTY QUIET is set, print pause message for operator
20          ;
21 000120 032761 0000000 0000000 3$:   BIT     ##QUIET,LSW4(R1); IS QUIET SET ON?
22 000126 001403          BEQ     2$                ;BR IF NOT
23 000130          .PRINT  #CMDBUF      ;DISPLAY THE PAUSE COMMAND
24          ;
25          ; Print pause message and wait for response
26          ;
27 000136          2$:   .PRINT  #PAUMSG      ;PRINT PAUSE MESSAGE
28 000144          1$:   .TTYIN          ;ACCEPT A LINE OF INPUT
29 000150 120027 000012          CMPB    RO,#LF
30 000154 001373          BNE     1$
31          ;
32          ; Restart the command file
33          ;
34 000156 042761 0000000 0000000          BIC     ##CFABT,LSW6(R1);Clear command-file abort flag
35 000164 016700 0000000          MOV     CFSPND,RO      ;RESTART COMMAND FILE
36 000170 004767 0000000          CALL    CFSTRT
37 000174 005067 0000000          CLR     CFSPND
38 000200 000167 0000000 9$:   JMP     RDCMD
39          ;
40          .SBTTL  DISPLAY command
41          ;-----
42          ; Display a line from within a command file.
43          ;
44 000204 124327 000040  CMDDSP:  CMPB    -(R3),#'          ;BACKUP OVER LEADING SPACES
45 000210 001775          BEQ     CMDDSP
46 000212 005203          INC     R3                ;POINT TO FIRST SEPARATOR
47 000214 122327 000040  CMPB    (R3)+,#'          ;IS IT A SPACE?
48 000220 001401          BEQ     1$                ;BR IF YES (SKIP IT)
49 000222 005303          DEC     R3                ;POINT TO 1ST SEPARATOR
50 000224          1$:   .PRINT  R3                ;PRINT THE ARGUMENT OF THE DISPLAY COMMAND
51 000230 000167 0000000          JMP     RDCMD

```

ASSIGN command

```

1          . SBTTL  ASSIGN command
2          ;-----
3          ;  PROCESS THE 'ASSIGN' COMMAND.
4          ;
5          ;  SEE IF FULL FILE ID IS SPECIFIED OR JUST DEVICE NAME.
6          ;
7 000234 004767 0000000  CMDASN: CALL  CVTTAR      ; CONVERT TAB AND FF CHARS TO SPACES
8 000240 105067 0000000      CLRB   ASKLNM      ; SET FLAG FOR LONG ASSIGN
9 000244 126227 177777 000072  CMPB   -(R2),#':    ; DOES LOGICAL NAME HAVE COLON?
10 000252 001001      BNE    5$          ; BR IF NOT
11 000254 105042      CLRB   -(R2)        ; CLEAR IT FOR NOW
12 000256 010201 5$:   MOV    R2,R1      ; POINT TO NULL AT END OF LINE
13 000260 020103 1$:   CMP    R1,R3      ; SCAN BACK TOWARDS FRONT
14 000262 101002      BHI    2$          ; BRANCH IF MORE TO DO
15 000264 000167 0000000  JMP    BADCMD      ; MUST HIT SOME DELIM
16 000270 114100 2$:   MOVB   -(R1),R0     ; GET NEXT CHAR BACK
17 000272 004767 000320  CALL   ASDELM      ; IS IT A DELIMITER?
18 000276 001370      BNE    1$          ; IF NOT KEEP SCANNING
19          ;  WE SCANNED BACK TO A DELIMITER -- SEE WHERE IT IS
20          ;  IN COMMAND.
21 000300 120027 000072      CMPB   R0,#':      ; REPLACE ': ' AND ' ' WITH '='
22 000304 001414      BEQ    3$          ;
23 000306 120027 000040      CMPB   R0,#'      ;
24 000312 001013      BNE    4$          ;
25 000314 020103 6$:   CMP    R1,R3      ; HAVE WE REACHED FRONT OF COMMAND LINE?
26 000316 101407      BLOS   3$          ; BR IF YES
27 000320 124127 000040  CMPB   -(R1),#'    ; SKIP OVER ALL CONSECUTIVE BLANKS
28 000324 001773      BEQ    6$          ;
29 000326 121127 000072  CMPB   (R1),#':    ; ALLOW "ASSIGN DX1: DK" CONSTRUCT
30 000332 001401      BEQ    3$          ;
31 000334 005201      INC    R1          ; POINT TO 1ST BLANK CHARACTER
32 000336 112711 000075 3$:   MOVB   #'=(R1)  ;
33 000342 010100 4$:   MOV    R1,R0      ; HOW MANY CHARS IN NAME TO
34 000344 160300      SUB    R3,R0      ; LEFT OF DELIMITER?
35 000346 020027 000003  CMP    R0,#3      ; DEVICE NAME <= 3 CHARS
36 000352 003021      BGT    ASFID      ; LONGER=>FULL FILE NAME.
37          ;
38          ;  SIMPLE ASSIGN OF PHYSICAL TO LOGICAL DEVICE.
39          ;  REFORMAT COMMAND TO 'DEV:A=DEV:' SO CSISPC CAN
40          ;  HANDLE IT.
41          ;
42 000354 112722 000072  ASSMPL: MOVB   #'',(R2)+  ; PUT COLON AT END OF 2ND NAME
43 000360 105022      CLRB   (R2)+      ; NULL TO TERMINATE STRING
44 000362 010204      MOV    R2,R4      ; MAKE ROOM AT END OF 1ST NAME
45 000364 062704 000002  ADD    #2,R4      ; BY SLIDING CHARS OVER 2 PLACES
46 000370 020201 1$:   CMP    R2,R1      ; WORK BACK DOWN TO '='
47 000372 001402      BEQ    2$          ; BRANCH AFTER '=' MOVED
48 000374 114244      MOVB   -(R2),-(R4) ; SLIDE OVER TO RIGHT
49 000376 000774      BR    1$          ;
50 000400 112722 000072 2$:   MOVB   #'',(R2)+  ; PUT COLON AFTER 1ST DEVICE NAME
51 000404 112712 000101  MOVB   #'A,(R2)    ; PUT IN DUMMY FILE NAME 'A'.
52 000410 105267 0000000  INCB   ASKLNM      ; REMEMBER TO KILL FILE NAME LATER
53 000414 000403      BR    ASCIS      ; GO DO .CSISPC
54          ;
55          ;  REQUEST ASSIGNMENT OF LOGICAL DEVICE TO NAMED FILE.
56          ;
57 000416 112722 000072  ASFID:  MOVB   #'',(R2)+  ; PUT COLON AT END OF LOGICAL NAME

```

ASSIGN command

```

58 000422 105012          CLR      (R2)          ;NULL TO END STRING
59
60          ; CALL CSI IN SPECIAL MODE TO PARSE ASSIGN COMMAND.
61
62 000424 010605          ASCIS:  MOV      SP,R5          ;SAVE STACK POINTER
63 000426          .CSISPC #KCSIBF,#ASDEX,R3
64 000442 010506          MOV      R5,SP          ;RESET TOP OF STACK
65 000444 103010          BCC      ASCSOK          ;BRANCH IF .CSISPC OK
66          ; ERROR OCCURED ON .CSISPC
67 000446 113705 000000G  CSIERR: MOV     @#ERRLOC,R5      ;GET ERROR CODE
68 000452 006305          ASL      R5              ;CONVERT TO WORD INDEX
69 000454 016504 000000G  MOV      KCSIMS(R5),R4      ;GET ADDR OF ERROR MESSAGE
70 000460          FABORT  R4          ;PRINT ERROR MESSAGE
71
72          ; Move the assign information into the assign table.
73          ; The physical device (and possibly file name)
74          ; description is in KCSIBF in the area for chan #0.
75          ; The logical device block is in the space for chan #3.
76
77          ; If "physical" device name used in assign statement is actually
78          ; a logical name (previously assigned), convert it to the corresponding
79          ; physical device name.
80
81 000466 016700 000000G  ASCSOK: MOV     KCSIBF,R0      ;GET PHYSICAL DEVICE NAME FOR ASSIGN
82 000472 004767 000000G  CALL     ASNSRC          ;SEE IF IT IS ACTUALLY A LOGICAL DEV NAME
83 000476 103403          BCS      3$             ;BR IF NOT
84 000500 016267 000000G 000000G  MOV      AT#DEV(R2),KCSIBF;REPLACE LOGICAL NAME WITH REAL PHYSICAL NAME
85 000506 105767 000000G  3$:  TSTB   ASKLNAM        ;MUST WE KILL FILE NAME?
86 000512 001402          BEQ      1$             ;BRANCH IF NOT
87 000514 005067 000002G  CLR      <KCSIBF+2>      ;KILL THE NAME
88          ; Clear any previous assign with this logical name.
89 000520 016700 000036G  1$:  MOV     KCSIBF+30.,R0    ;GET LOGICAL DEVICE NAME
90 000524 004767 000000G  CALL     ASNSRC          ;SEE IF NAME IS ALREADY ASSIGNED
91 000530 103010          BCC      2$             ;BR IF ASSIGN ALREADY EXISTS (REUSE THIS ENTRY)
92          ; Search for a free assign block.
93 000532 005000          CLR      R0              ;SEARCH FOR FREE ASSIGN BLOCK
94 000534 004767 000000G  CALL     ASNSRC          ;SEE IF NAME IS ALREADY ASSIGNED
95 000540 103004          BCC      2$             ;BR IF FOUND ONE
96          ; ASSIGN TABLE IS FULL.
97 000542          FABORT  #ASNOVF
98
99          ; FOUND A FREE SLOT (POINTED TO BY R2) MOVE ASSIGN
100         ; INFO INTO BLOCK.
101
102 000552 016762 000036G 000000G  2$:  MOV     KCSIBF+30.,AT#LOG(R2);MOVE IN LOGICAL DEVICE NAME
103 000560 016762 000010G 000000G  MOV     KCSIBF+8.,AT#SIZ(R2);MOVE IN FILE LENGTH
104 000566 012703 000000G  MOV     #KCSIBF,R3
105 000572 012362 000000G  MOV     (R3)+,AT#DEV(R2);MOVE IN REAL DEVICE NAME
106 000576 012362 000000G  MOV     (R3)+,AT#FIL(R2);MOVE IN FILE NAME
107 000602 012362 000002G  MOV     (R3)+,AT#FIL+2(R2)
108 000606 011362 000000G  MOV     (R3),AT#EXT(R2) ;MOVE IN FILE EXTENSION
109 000612 000167 000000G  JMP     RDCMD

```

ASSIGN command

```

1          ; -----
2          ; ASDELM IS CALLED TO CHECK IF THE CHARACTER IN RO
3          ; IS ONE OF THE CHARACTERS ':', '=', OR ' '.
4          ; ON RETURN THE CONDITION CODE IS SET FOR BEQ/BNE.
5          ; ALL REGISTERS ARE PRESERVED.
6          ;
7 000616   120027   000075   ASDELM:  CMPB    RO, #'=      ; CHECK EQUAL SIGN
8 000622   001405                   BEQ     1$
9 000624   120027   000072                   CMPB    RO, #' :      ; CHECK COLON
10 000630   001402                   BEQ     1$
11 000632   120027   000040                   CMPB    RO, #' '      ; CHECK SPACE
12 000636   000207   1$:      RETURN      ; RETURN WITH CC SET.

```

DEASSIGN command

```

1          .SBTTL  DEASSIGN command
2          ;-----
3          ;  PROCESS THE DEASSIGN COMMAND
4          ;
5 000640   004767   0000000  CMDDSN:  CALL    CVTTAB      ; CONVERT TAB AND FF TO SPACES
6 000644   105713                   TSTB    (R3)        ; DEASSIGN ALL OR ONE?
7 000646   001010                   BNE     DEAS1      ; BR TO DEASSIGN ONE NAME
8          ;
9          ;  REQUEST TO DEASSIGN ALL LOGICAL NAMES
10         ;  (SIMPLY ZERO THE ASSIGN TABLE)
11         ;
12 000650   012702   0000000          MOV     #ASNTBL,R2    ; POINT TO START OF TABLE
13 000654   005022 1$:  CLR     (R2)+
14 000656   020227   0000000          CMP     R2,#ASNEND  ; DONE ALL?
15 000662   103774                   BLO    1$          ; BR IF NOT
16 000664   000167   0000000  DAJMP:  JMP     RDCMD
17         ;
18         ;  REQUEST TO DEASSIGN A PARTICULAR LOGICAL DEVICE NAME
19         ;
20 000670   004767   0000000  DEAS1:  CALL    GTRD50    ; ACCRUE THE DEVICE NAME
21 000674   016700                   MOV     R50BUF,R0   ; PICK UP DEVICE NAME
22 000700   012702   0000000          MOV     #ASNTBL,R2  ; POINT TO START OF ASSIGN TABLE
23 000704   020062   0000000 1$:  CMP     R0,AT$LOG(R2) ; IS THIS ENTRY FOR THE DEVICE?
24 000710   001004                   BNE    2$          ; BR IF NOT
25 000712   005062   0000000          CLR     AT$LOG(R2)  ; DEASSIGN THE LOGICAL NAME
26 000716   005062   0000000          CLR     AT$DEV(R2)  ; CLEAR PHYSICAL DEVICE NAME ALSO
27 000722   062702   0000000 2$:  ADD     #AT$$SZ,R2  ; POINT TO NEXT ASSIGN TABLE ENTRY
28 000726   020227   0000000          CMP     R2,#ASNEND  ; REACHED END OF TABLE?
29 000732   103764                   BLO    1$          ; BR IF MORE TO DO
30 000734   000753                   BR     DAJMP

```

ALLOCATE command

```

1          .SBTTL  ALLOCATE command
2          ;-----
3          ; Allocate a device for exclusive access by this job.
4          ; The form of this command is:
5          ;   ALLOCATE dev[:],... logical_name
6          ;
7 000736  CMDALC:
8          ;
9          ; If no devices were specified, treat this like a SHOW ALLOCATE command
10         ;
11 000736  005067  177040          CLR      ALC1DV      ; Say no allocat device seen yet
12 000742  004767  0000000      CALL     CVTTAB      ; Convert tab and FF chars to spaces
13 000746  004767  0000000      CALL     SKPSPC      ; Skip over spaces
14 000752  105713          TSTB     (R3)        ; Any devices specified?
15 000754  001002          BNE      1$          ; Br if yes
16 000756  000167  0000000      JMP      SOPALC      ; Do SHOW ALLOCATE command
17         ;
18         ; See if we are authorized to allocate devices
19         ;
20 000762  032767  0000000 0000000 1$:  BIT      #PO$ALC,PRIVCO ; Are we authorized to allocate devices?
21 000770  001004          BNE      10$          ; Br if yes
22 000772          FABORT   #EM$ALC      ; Not authorized for ALLOCATE
23         ;
24         ; Begin loop to get the name of each device to be allocated
25         ;
26 001002  004767  0000000      10$:  CALL     SKPSPC      ; Skip over any spaces
27         ;
28         ; Accrue the next device name
29         ;
30 001006  004767  0000000          CALL     GTRD50      ; Accrue the device name
31 001012  122327  0000072          CMPB     (R3)+,#' : ; Was a colon specified with the device name?
32 001016  001401          BEQ      6$          ; Br if yes
33 001020  005303          DEC     R3          ; Backup pointer
34 001022  010346          6$:  MOV     R3,-(SP)    ; Save command pointer
35 001024  004767  0000000          CALL     SKPDLM      ; Skip legal delimiters (blank(s), comma, EOL)
36 001030  012603          MOV     (SP)+,R3     ; Restore command pointer
37 001032  103435          BCS     41$          ; Invalid character following name
38 001034  016702  0000000          MOV     R50BUF,R2    ; Get the name of the device
39 001040  010267  0000000          MOV     R2,ALCDEV    ; Set name of device being allocated
40         ;
41         ; Save the first device name
42         ;
43 001044  005767  176732          TST     ALC1DV      ; Have we seen the first device yet?
44 001050  001002          BNE     2$          ; Br if yes
45 001052  010267  176724          MOV     R2,ALC1DV    ; Save the name of the 1st device
46         ;
47         ; Do the allocation for this device
48         ;
49 001056  012700  0000000      2$:  MOV     #ALDEMT,R0 ; Point to EMT arg block to do the allocation
50 001062  104375          EMT     375          ; Try to allocate the device
51 001064  103051          BCC     3$          ; Br if allocation was successful
52 001066  010005          MOV     R0,R5       ; Save returned error value
53         ;
54         ; An error occurred on the allocation. Print error message.
55         ;
56 001070  113702  0000000          MOVB    @#ERRLOC,R2 ; Get allocation error code
57 001074  120227  000001          CMPB    R2,#1        ; Device already allocated by someone else?

```

ALLOCATE command

```

58 001100 001007          BNE      4$          ;Br if not
59 001102                FERR     #EM#DAA      ;Device allocated by another job
60 001116 000427          BR       7$          ;
61 001120 120227 000002   4$:     CMPB     R2,#2      ;Invalid device being allocated?
62 001124 001004          BNE      5$          ;Br if not
63 001126                41$:    FABORT  #EM#IAD      ;Invalid device for allocation
64 001136 120227 000003   5$:     CMPB     R2,#3      ;Allocation table full?
65 001142 001004          BNE      8$          ;Br if not
66 001144                FABORT  #EM#ATF      ;Allocation table full
67 001154 120227 000004   8$:     CMPB     R2,#4      ;Device in use by another user?
68 001160 001013          BNE      3$          ;Br if not
69 001162                FERR     #EM#DIU      ;Device is in use by another user
70 001176 004767 000000G  7$:     CALL     PRTDEC     ;Print the number of the job that has the dev
71 001202                .PRINT  #CRLF      ;End of line
72                ;
73                ; See if there are more devices to be allocated
74                ;
75 001210 004767 000000G  3$:     CALL     SKPSPC     ;Skip over any spaces
76 001214 122327 000054          CMPB     (R3)+,#'      ;Is there another device to allocate?
77 001220 001670          BEQ      10$         ;Loop if yes
78                ;
79                ; We have finished allocating all devices.
80                ; See if we need to assign a logical name to the first device we allocated.
81                ;
82 001222 105743          TSTB     -(R3)        ;Was a logical device name specified?
83 001224 001410          BEQ      9$          ;Br if not
84 001226 004767 000000G  CALL     GTRD50       ;Accrue the logical device name
85 001232 016705 000000G  MOV     R50BUF,R5     ;Get the logical device name
86 001236 016700 176540          MOV     ALC1DV,R0     ;Get the physical device name
87 001242 004767 000000G  CALL     DOASGN       ;Do the assignment
88                ;
89                ; Finished
90                ;
91 001246 000167 000000G  9$:     JMP      RDCMD     ;Finished with command

```

DEALLOCATE command

```

1          .SBTTL  DEALLOCATE command
2          ;-----
3          ; Deallocate an allocated device.
4          ;
5 001252  005067  176524  CMD DLC: CLR      ALC1DV      ;Clear deallocated device counter
6 001256  032767  000000G 000000G BIT      #PO$ALC,PRIVCO ;Are we authorized to allocate devices?
7 001264  001004          BNE      1$          ;Br if yes
8 001266          FABORT  #EM$ALC      ;Not authorized to deallocate
9          ;
10         ; See if /ALL was specified
11         ;
12 001276  004767  000000G 1$:      CALL      CVTTAB      ;Convert tab and FF chars to spaces
13 001302  004767  000000G      CALL      CVTUC        ;Convert lower case chars to upper case
14 001306  004767  000000G      CALL      SKPSPC       ;Skip over spaces
15         ;
16         ; Find devices for deallocation.
17         ;
18 001312  111300 2$:      MOV B      (R3),R0      ;Get 1st char of operand
19 001314  001473      BEQ      10$          ;No devices specified, deallocate all
20 001316  120027  000057 3$:      CMP B      R0,#'/'      ;Was a switch specified?
21 001322  001012      BNE      4$          ;Br if not
22 001324  005203      INC      R3          ;Skip over slash
23 001326  004767  000000G      CALL      SKPSPC       ;Skip any spaces
24 001332  122327  000101      CMP B      (R3)+,#'A      ;"ALL"?
25 001336  001465      BEQ      12$          ;Br if yes
26 001340          FABORT  #INVOPT      ;Invalid option
27         ;
28         ; Deallocate a list of devices
29         ; Accrue the next device name
30         ;
31 001350  005267  176426 4$:      INC      ALC1DV      ;Count deallocated devices
32 001354  004767  000000G      CALL      SKPSPC       ;Skip over any spaces
33 001360  004767  000000G      CALL      @TRD50       ;Accrue the device name
34 001364  122327  000072      CMP B      (R3)+,#':      ;Was colon specified following name?
35 001370  001401      BEQ      5$          ;Br if yes
36 001372  005303      DEC      R3          ;Backup pointer
37 001374  004767  000000G 5$:      CALL      SKPDLM       ;Skip valid delimiters (blank(s), comma)
38 001400  103435      BCS      7$          ;Br if invalid character
39         ;
40         ; Deallocate the device specified.
41         ;
42 001402  016702  000000G      MOV      R50BUF,R2      ;Get the name of the device
43 001406  010267  000000G      MOV      R2,ALCDEV      ;Set name of device being allocated
44 001412  012700  000000G      MOV      #DLCEMT,R0     ;Point to EMT argument block
45 001416  104375      EMT      375           ;Do the deallocation
46 001420  103334      BCC      2$          ;Br if ok
47 001422  010005      MOV      R0,R5        ;Save job # if owned by someone else
48 001424  113700  000000G      MOV B      @#ERRLOC,R0  ;Get error code
49 001430  120227  000001      CMP B      R2,#1       ;Device allocated by someone else?
50 001434  001014      BNE      6$          ;Br if not
51 001436          FERR      #EM$DAA      ;Device allocated by another job
52 001452  004767  000000G      CALL      PRTDEC       ;Print the number of the job that has the dev
53 001456          .PRINT  #CRLF        ;End of line
54 001464  000712      BR      2$          ;
55 001466  120227  000002 6$:      CMP B      R2,#2       ;Invalid device being allocated?
56 001472  001307      BNE      2$          ;Br if not
57 001474          FABORT  #EM$IAD      ;Invalid device for allocation

```

DEALLOCATE command

```

58      ;
59      ; Deallocate all devices allocated by this user
60      ;
61 001504 005767 176272 10$:   TST   ALC1DV   ;Were any devices specified?
62 001510 001006          BNE   20$   ;Yes, end of command
63 001512 005067 0000000 12$:   CLR   ALCDEV   ;Say to deallocate all devices
64 001516 012700 0000000      MOV   #DLCEMT,R0  ;Point to EMT argument block
65 001522 104375          EMT   375   ;Do the deallocation
66 001524 000400          BR    20$
67      ;
68      ; Finished
69      ;
70 001526 000167 0000000 20$:   JMP   RDCMD   ;Finished with command

```

REMOVE command

```

1          .SBTTL  REMOVE command
2          ;-----
3          ; The REMOVE command has two functions:
4          ; (1) To remove a device handler from memory. This function is a NOP
5          ; under TSX-Plus;
6          ; (2) To remove a named PLAS region.
7          ;
8 001532 004767 0000000  CMDREM: CALL  CVTTAB          ;Convert tab and FF chars to spaces
9          ;
10         ; Quit if we have reached the end of the command line
11         ;
12 001536 004767 0000000 1$:      CALL  SKPSPC          ;Skip past leading spaces
13 001542 105713          TSTB   (R3)          ;Are we at the end of the command?
14 001544 001436          BEQ    9$          ;Br if yes
15         ;
16         ; Accrue the name of a device or region
17         ;
18 001546 004767 0000000          CALL  GTRD50          ;Accrue device or region name
19         ;
20         ; If the name is longer than 3 characters then it must be a region
21         ; name, otherwise check to see if it is a device name.
22         ;
23 001552 005767 0000020          TST    R50BUF+2          ;Is name longer than 3 characters?
24 001556 001012          BNE    2$          ;Br if yes
25 001560 121327 0000072          CMPB   (R3),#':'          ;Is name terminated with colon?
26 001564 001002          BNE    3$          ;Br if not
27 001566 105723          TSTB   (R3)+          ;Skip over colon
28 001570 000407          BR     5$          ;Must have been a device name
29 001572 016705 0000000 3$:      MOV    R50BUF,R5          ;Get 1st 3 chars of name
30 001576 004767 0000000          CALL  CHKDEV          ;See if this is the name of a device
31 001602 103002          BCC    5$          ;Br if device name
32         ;
33         ; This is a request to remove a named PLAS region.
34         ;
35 001604 004767 0000036 2$:      CALL  REMRGN          ;Remove a region
36         ;
37         ; Skip over any separating comma
38         ;
39 001610 122327 0000054 5$:      CMPB   (R3)+,#','          ;Is there a separating comma?
40 001614 001401          BEQ    4$          ;Br if yes
41 001616 005303          DEC    R3          ;Point back to skipped char
42 001620 105713          4$:      TSTB   (R3)          ;Reached end of command?
43 001622 001407          BEQ    9$          ;Br if yes
44 001624 121327 0000040          CMPB   (R3),#'          ;Space separator?
45 001630 001742          BEQ    1$          ;Br if yes
46 001632          FABORT  #EM#CSE          ;Command syntax error
47         ;
48         ; Finished
49         ;
50 001642 000167 0000000 9$:      JMP    RDCMD

```

REMOVE command

```

1          ; -----
2          ; Eliminate a named PLAS region.
3          ;
4          ; Inputs:
5          ; R50BUF = Rad50 name of the region.
6          ;
7 001646 010246 REMRGN: MOV      R2,-(SP)
8          ;
9          ; First attempt to attach to a private region with that name.
10         ;
11 001650 012702 000010'      MOV      #REMRDB,R2      ;Point to region definition block
12 001654 005062 000000G      CLR      R.GSIZ(R2)      ;Clear size word
13 001660 012762 000000C 000000G  MOV      #<RS.GBL!RS.PVT>,R.GSTS(R2) ;Function = Attach private region
14 001666 016762 000000G 000000G  MOV      R50BUF,R.NAME(R2) ;Set name of region
15 001674 016762 000002G 000002G  MOV      R50BUF+2,R.NAME+2(R2)
16 001702          .CRRG     #XAREA,R2      ;Attempt to attach to region
17 001720 032762 000000G 000000G  BIT      #RS.CRR,R.GSTS(R2) ;Were we able to attach?
18 001726 001025          BNE      1$      ;Br if yes
19         ;
20         ; Unable to attach to private region.
21         ; Try to attach to public region.
22         ;
23 001730 012762 000000G 000000G  MOV      #RS.GBL,R.GSTS(R2) ;Function = Attach global region
24 001736          .CRRG     #XAREA,R2      ;Attempt to attach to it
25 001754 032762 000000G 000000G  BIT      #RS.CRR,R.GSTS(R2) ;Was attachment successful?
26 001762 001007          BNE      1$      ;Br if yes
27         ;
28         ; Error -- We are unable to attach to region
29         ;
30 001764          FERR     #EM$UAR      ;Unable to attach to region
31 002000 000421          BR      9$
32         ;
33         ; We successfully attached to the region.
34         ; Try to eliminate the region.
35         ;
36 002002 012762 000000G 000000G 1$:  MOV      #RS.EGR,R.GSTS(R2) ;Function = Eliminate region
37 002010          .ELRG     #XAREA,R2      ;Try to eliminate the region
38 002026 103006          BCC      9$      ;Br if successful
39         ;
40         ; Error -- Unable to eliminate the region
41         ;
42 002030          FERR     #EM$UER      ;Unable to eliminate region
43         ;
44         ; Finished
45         ;
46 002044 012602 9$:      MOV      (SP)+,R2
47 002046 000207          RETURN

```

INSTALL Command

```

1          .SBTTL  INSTALL Command
2          ;-----
3          ; Process the INSTALL command.
4          ;
5 002050  004767  0000000  CMDINS: CALL  CVTTAB      ;Convert tabs to spaces
6 002054  004767  0000000          CALL  CVTUC       ;Convert to upper case
7          ;
8          ; Accrue first keyword and see if it is ADD or DELETE
9          ;
10 002060  010302          MOV    R3,R2        ;Save pointer to start of keyword
11 002062  004767  0000000  CALL  GTRD50      ;Accrue 1st keyword
12 002066  016700  0000000  MOV    R50BUF,R0   ;Get 1st 3 chars of keyword
13 002072  020067  175724   CMP    R0,R50ADD    ;Is keyword ADD?
14 002076  001421          BEQ    INSADD       ;Br if yes
15 002100  020067  175720   CMP    R0,R50DEL    ;Is keyword DELETE?
16 002104  001473          BEQ    INSDEL       ;Br if yes
17 002106  020067  175716   CMP    R0,R50REM    ;Is keyword REMOVE?
18 002112  001470          BEQ    INSDEL       ;Br if yes
19          ;
20          ; Keyword is not ADD or DELETE.
21          ; Error if 1st keyword is longer than 2 characters.
22          ;
23 002114  010300          MOV    R3,R0        ;Get pointer past end of keyword
24 002116  160200          SUB    R2,R0        ;Determine length of 1st keyword
25 002120  020027  0000002  CMP    R0,#2       ;1st keyword longer than 2 chars?
26 002124  101404          BLOS  1$           ;Br if not
27 002126          FABORT #ILLCMD      ;Invalid command
28          ;
29          ; This must be a device handler name -- Ignore this command
30          ;
31 002136  000167  0000000  1$:  JMP    RDCMD      ;Finished installing device handler

```

INSTALL Command

```

1          ; -----
2          ; We are adding an image
3          ;
4 002142 004767 000000G  INSADD: CALL  CKSYPV      ;Require SYSPRV privilege
5 002146 005067 175660   CLR    IATTRB      ;Clear all attribute flags
6 002152 004767 000000G   CALL  CLRPRV      ;Reset privilege flag cells
7 002156 004767 000546   CALL  INSOPT      ;Process any options that follow ADD
8          ;
9          ; Get the name of the image and see if it is currently in table
10         ;
11 002162 004767 000144   CALL  INSNAM      ;Get name of image and look it up
12 002166 103002          BCC    1$          ;Br if name already in table
13         ;
14         ; Name is not currently in table.
15         ; Try to find a free slot in the table.
16         ;
17 002170 004767 000450   CALL  INSFRE      ;Find free table entry
18         ;
19         ; Set up program name for free entry
20         ;
21 002174 012702 000000C  1$:  MOV    #II$NAM+IIBUF,R2;Point to target buffer
22 002200 012704 000000G   MOV    #FILNAM,R4  ;Point to name
23 002204 012700 000004   MOV    #4,R0       ;Move 4 words
24 002210 012422          2$:  MOV    (R4)+,(R2)+  ;Move name to buffer
25 002212 077002          SOB    R0,2$
26         ;
27         ; At this point, R5 points to buffer where install entry is to be made
28         ; Process any options that follow the file name
29         ;
30 002214 004767 000510   CALL  INSOPT      ;Process any options that follow name
31         ;
32         ; Move attribute and privilege flags into install entry
33         ;
34 002220 010546          MOV    R5,-(SP)    ;Save address of entry in install table
35 002222 016767 175604 000000C  MOV    IATTRB,II$FLG+IIBUF ;Set attribute flags for program
36 002230 012702 000000G   MOV    #PFS0,R2   ;Point to accrued privilege flags
37 002234 012703 000000C   MOV    #II$PRV+IIBUF,R3;Install entry flags
38 002240 012704 000000G   MOV    #PFC0,R4   ;Flags to clear when run
39 002244 012705 000000C   MOV    #II$NPV+IIBUF,R5;Install entry
40 002250 012700 000000G   MOV    #PVNPW,R0  ;Get # words to move
41 002254 012223          3$:  MOV    (R2)+,(R3)+  ;Move flags to be set
42 002256 012425          MOV    (R4)+,(R5)+  ;Move flags to be cleared
43 002260 077003          SOB    R0,3$
44 002262 012605          MOV    (SP)+,R5    ;Recover address of entry in install table
45         ;
46         ; Add the entry to the install table
47         ;
48 002264 004767 000424   CALL  INSPUT      ;Add entry to table
49         ;
50         ; Finished
51         ;
52 002270 000167 000000G   JMP    RDCMD

```

INSTALL Command

```

1          ;
2          ; Process INSTALL DELETE command
3          ;
4 002274 004767 0000000  INSDEL: CALL    CKSYPV          ;Require SYSPRV privilege
5          ;
6          ; See if we can find program in install table
7          ;
8 002300 004767 000026   CALL    INSNAM          ;Accrue file spec and try to find in table
9 002304 103004         BCC    1$              ;Br if found file name in table
10 002306                FABORT  #EM#FNI       ;File not installed
11         ;
12         ; Found entry, delete it
13         ;
14 002316 005067 0000000 1$:   CLR    II#NAM+IIBUF  ;Say image is not installed
15 002322 004767 000366   CALL    INSPUT          ;Write entry back to table
16         ;
17         ; Finished
18         ;
19 002326 000167 0000000                JMP    RDCMD

```

INSTALL Command

```

1
2 ; -----
3 ; Accrue a file spec for a file that is being installed and look the
4 ; name up in the install table.
5 ;
6 ; Inputs:
7 ;   R3 = Pointer to start of file spec.
8 ;
9 ; Outputs:
10 ;   C-flag cleared ==> Found entry in install table.
11 ;   C-flag set ==> Cannot find entry in install table.
12 ;   R5 = Address of entry in install table if found.
13 ;
14 ;
15 ;
16 ; Determine if wildcard ("*") specified as device name
17 ;
18 ;
19 ;
20 ;
21 ;
22 ;
23 ;
24 ;
25 ;
26 ;
27 ;
28 ; Accrue the file spec
29 ;
30 ;
31 ;
32 ;
33 ;
34 ;
35 ;
36 ;
37 ;
38 ;
39 ;
40 ;
41 ; Translate logical file device to physical device with unit #
42 ;
43 ;
44 ;
45 ;
46 ;
47 ;
48 ;
49 ;
50 ; Error if logical disk (LD) specified for device
51 ;
52 ;
53 ;
54 ;
55 ;
56 ;
57 ;

```

13	002332	010246			INSNAM: MOV	R2,-(SP)	
14	002334	010446				MOV	R4,-(SP)
18	002336	005002				CLR	R2 ; Assume device name is not wild
19	002340	004767	0000000			CALL	SKPSPC ; Skip up to start of file spec
20	002344	121327	000052			CMPB	(R3),#'* ; Wildcard as device name?
21	002350	001013				BNE	11\$; Br if not
22	002352	126327	000001	000072		CMPB	1(R3),#' ; Is this the device name?
23	002360	001007				BNE	11\$; Br if not
24	002362	012702	132500			MOV	#WLDNAM,R2 ; Remember device name is wild
25	002366	112713	000040			MOVB	#',(R3) ; Blank out the device name
26	002372	112763	000040	000001		MOVB	#',1(R3)
30	002400	012704	0000000		11\$:	MOV	#R50SAV,R4 ; Get default file extension (SAV)
31	002404	005005				CLR	R5 ; Say this is an input file
32	002406	004767	0000000			CALL	ACRFIL ; Accrue the file spec
33	002412	103510				BCS	10\$; Br if invalid file spec
34	002414	005767	0000020			TST	FILNAM+2 ; Was a file name specified?
35	002420	001505				BEQ	10\$; Br if not
36	002422	005702				TST	R2 ; Is device name wild?
37	002424	001403				BEQ	12\$; Br if not
38	002426	010267	0000000			MOV	R2,FILNAM ; Set wildcard as file name
39	002432	000431				BR	5\$
43	002434	012705	0000000		12\$:	MOV	#FILNAM,R5 ; Point to file spec
44	002440	004767	0000000			CALL	LOGASN ; Do logical assignment
45	002444	010346				MOV	R3,-(SP) ; SAVE POINTER
46	002446	010503				MOV	R5,R3 ; GET COPY OF DEVICE POINTER
47	002450	004767	0000000			CALL	FORCE0 ; MAKE BLANK UNIT = 0
48	002454	012603				MOV	(SP)+,R3 ; RESTORE POINTER
52	002456	016705	0000000			MOV	FILNAM,R5 ; Get the device name
53	002462	004767	0000000			CALL	CHKDEV ; Convert to device index number
54	002466	103407				BCS	6\$; Br if invalid device
55	002470	020467	0000000			CMP	R4,LDDEVX ; Is this a logical disk?
56	002474	001010				BNE	5\$; Br if not
57	002476					FABORT	#EM#LDI ; Can't have installed program on LD

INSTALL Command

```

58 002506          6$:      FABORT  #EM#IAD          ;Invalid device
59                ;
60                ; Try to find existing entry for file in install table
61                ;
62 002516  016705  000000G  5$:      MOV      INSTBL,R5          ;Point to 1st entry in table
63 002522  010567  000000G  1$:      MOV      R5,INGADR          ;Set address in EMT
64 002526  012700  000000G          MOV      #INGEMT,R0          ;Point to get EMT
65 002532  104375          EMT      375          ;Get next install table entry
66 002534  005760  000000C          TST      IIBUF+II$NAM(R0); Is this entry empty?
67 002540  001422          BEQ      2$          ;Br if yes
68 002542  012700  0000006          MOV      #6,R0          ;Get index to last word of name
69 002546  026027  000000G  132500  3$:      CMP      FILNAM(R0),#WLDNAM ;Wildcard?
70 002554  001410          BEQ      13$         ;Br if yes
71 002556  026027  000000C  132500          CMP      IIBUF+II$NAM(R0),#WLDNAM ;Wildcard?
72 002564  001404          BEQ      13$         ;Br if yes
73 002566  026060  000000G  000000C          CMP      FILNAM(R0),IIBUF+II$NAM(R0) ;Compare names
74 002574  001004          BNE      2$          ;Br if this is not right entry
75 002576  162700  0000002          13$:     SUB      #2,R0          ;More words to check?
76 002602  002361          BGE      3$          ;Br if yes
77 002604  000407          BR       4$          ;Found entry
78 002606  062705  000000G          2$:      ADD      #II#$SZ,R5          ;Point to next entry in table
79 002612  020567  000000G          CMP      R5,INSTBN          ;Reached end of table?
80 002616  103741          BLO      1$          ;Loop if not
81                ;
82                ; Cannot find entry in table
83                ;
84 002620  000261          SEC          ;Signal failure on return
85 002622  000401          BR       9$
86                ;
87                ; We found entry for this name
88                ;
89 002624  000241          4$:      CLC          ;Signal success on return
90                ;
91                ; Finished
92                ;
93 002626  012604          9$:      MOV      (SP)+,R4
94 002630  012602          MOV      (SP)+,R2
95 002632  000207          RETURN
96                ;
97                ; Invalid file spec
98                ;
99 002634          10$:     FABORT  #BDFNAM          ;Invalid file spec

```

INSTALL Command

```

1 ;-----
2 ; Try to find a free entry in the install table.
3 ;
4 ; Outputs:
5 ; R5 = Address of free entry in install table.
6 ;
7 002644 016705 0000000 INSFRE: MOV INSTBL,R5 ;Point to 1st install table entry
8 002650 010567 0000000 1$: MOV R5,INGADR ;Set address for EMT
9 002654 012700 0000000 MOV #INGEMT,R0 ;EMT to get assign table entry
10 002660 104375 EMT 375 ;Get next entry
11 002662 005767 0000000 TST II$NAM+IIBUF ;Is this entry free?
12 002666 001411 BEQ 2$ ;Br if found free entry
13 002670 062705 0000000 ADD #II$SZ,R5 ;Point to next entry
14 002674 020567 0000000 CMP R5,INSTBN ;Reached end of table?
15 002700 103763 BLO 1$ ;Loop if not
16 ;
17 ; No free table entries
18 ;
19 002702 FABORT #EM$ITF ;Install table full
20 ;
21 ; Found free entry
22 ;
23 002712 000207 2$: RETURN

```

INSTALL Command

```
1 ;-----  
2 ; Write the current install entry into the install table.  
3 ;  
4 ; Inputs:  
5 ; R5 = Address in install table where entry is to go.  
6 ;  
7 002714 010567 0000000 INSPUT: MOV R5, INPADR ;Set address for EMT  
8 002720 012700 0000000 MOV #INPEMT, R0 ;Point to EMT  
9 002724 104375 EMT 375 ;Store install table entry  
10 002726 000207 RETURN
```

INSTALL Command

```

1 ;-----
2 ; Process command options specified with an INSTALL command.
3 ;
4 ; Inputs:
5 ; R3 = Pointer to option list.
6 ;
7 ; Outputs:
8 ; R3 = Points past end of option list
9 ;
10 002730 010446 INSOPT: MOV R4, -(SP)
11 ;
12 ; Process list of options
13 ;
14 002732 012704 000000' MOV #INSHD, R4 ;Point to table of options
15 002736 004767 000000G CALL OPTLST ;Process an option list
16 ;
17 ; Finished
18 ;
19 002742 012604 MOV (SP)+, R4
20 002744 000207 RETURN
21 ;-----
22 ;
23 ; Subroutine called to set a specific install attribute flag
24 ;
25 ; Inputs:
26 ; R4 = Pointer to option table entry.
27 ;
28 ; Outputs:
29 ; IATTRB = Combined attribute flags
30 ;
31 002746 051467 175060 INSATR: BIS (R4), IATTRB ;Set attribute flag
32 002752 000207 RETURN ;Finished

```


58 003174 000167 000346

JMP MNTCSH

;NO FILE NAME -- GO ENABLE DIRECTORY CACHING

MOUNT command

```

1          ;
2          ; This is a MOUNT command with a file name.
3          ; We are mounting a logical disk.
4          ; Make sure Logical Disk support was genned into system.
5          ;
6 003200 105767 0000000 MNTLD: TSTB VLDSYS ; WAS LD SUPPORT GENNED IN?
7 003204 001004          BNE 18$ ; BR IF YES
8 003206          FABORT #EM$NLD ; LD SUPPORT NOT GENNED IN
9          ;
10         ; Make sure the logical disk name is ok.
11         ;
12 003216 010200 18$: MOV R2,R0 ; GET LD DEVICE NAME
13 003220 004767 000412 CALL LDDEVN ; GET LD DEVICE INDEX NUMBER
14 003224 103004 BCC 6$ ; BR IF OK
15 003226          FABORT #INVLDN ; ERROR IF NAME IS NOT LD
16 003236 010002 6$: MOV R0,R2 ; CARRY INDEX IN R2
17         ;
18         ; Close the log file if it is on the logical disk
19         ;
20 003240 016705 0000000 MOV MNTDEV,R5 ; GET NAME OF LOGICAL DEVICE
21 003244 004767 0000000 CALL LOGCHK ; SEE IF WE NEED TO CLOSE THE LOG FILE
22         ;
23         ; Tell system to stop doing directory caching for the device
24         ;
25 003250 112767 000001 0000000 MOV#B #1,SERFLG ; DO .SERR TO AVOID ABORT FOR ILLEGAL DEVICE
26 003256 012700 0000000 MOV #DMTARG,R0 ; POINT TO EMT ARGUMENT BLOCK
27 003262 104375 EMT 375 ; TELL SYSTEM TO STOP DOING CACHING
28 003264 105067 0000000 CLRB SERFLG ; DO .HERR
29         ;
30         ; Accrue the file name
31         ;
32 003270 012704 0000000 MOV #R5ODSK,R4 ; SET DEFAULT FILE EXTENSION
33 003274 005005 CLR R5 ; SAY THIS IS AN INPUT FILE
34 003276 004767 0000000 CALL ACRFIL ; ACCRUE THE FILE SPEC
35 003302 103004 BCC 13$ ; BR IF OK
36 003304 5$: FABORT #BDFNAM ; INVALID FILE SPEC
37         ;
38         ; Translate logical file device name to physical device
39         ;
40 003314 012705 0000000 13$: MOV #FILNAM,R5 ; POINT TO FILE SPEC AREA
41 003320 004767 0000000 CALL LOGASN ; PERFORM ANY LOGICAL DEVICE ASSIGNMENT
42         ;
43         ; Make sure file is not on same LD as is being mounted
44         ;
45 003324 016700 0000000 MOV FILNAM,R0 ; GET DEVICE THAT FILE IS ON
46 003330 004767 000302 CALL LDDEVN ; SEE IF FILE IS ON A LD DEVICE
47 003334 103406 BCS 17$ ; BR IF NOT ON LD DEVICE
48 003336 020002 CMP R0,R2 ; IS IT ON LOWER # LD?
49 003340 103404 BLD 17$ ; BR IF YES
50 003342          FABORT #INVLDM ; INVALID LD ORDER
51         ;
52         ; Make sure the file name is not null
53         ;
54 003352 005767 0000020 17$: TST FILNAM+2 ; Is file name null?
55 003356 001752 BEQ 5$ ; Br if yes -- Error
56         ;
57         ; Move file name into logical disk name table

```

MOUNT command

```

58
59 003360 010205          MOV      R2,R5          ;GET LOGICAL DISK UNIT #
60 003362 006305          ASL      R5              ;*4 WORDS PER ENTRY
61 003364 006305          ASL      R5
62 003366 062705 000000G  ADD      #LDNAME,R5     ;POINT TO ENTRY IN LOGICAL DISK TABLE
63 003372 012704 000000G  MOV      #FILNAM,R4     ;POINT TO FILE NAME BUFFER
64 003376 012425          MOV      (R4)+,(R5)+    ;MOVE NAME TO TABLE
65 003400 012425          MOV      (R4)+,(R5)+
66 003402 012425          MOV      (R4)+,(R5)+
67 003404 011415          MOV      (R4),(R5)
68
69          ; Set up flags for logical disk
70
71 003406 005004          CLR      R4
72 003410 105767 000000G  TSTB    NOFLAG         ;WAS NOWRITE SPECIFIED?
73 003414 001402          BEQ     9$             ;BR IF NOT
74 003416 052704 000000G  BIS     #LD#RON,R4     ;SET READ-ONLY FLAG
75 003422 010462 000000G  9$:     MOV      R4,LDFLAG(R2)
76
77          ; Lookup file and set up information about position of logical
78          ; disk on physical disk.
79
80 003426 004767 000000G  CALL    LDMNT          ;SET UP INFO ABOUT FILE FOR LOGICAL DISK
81 003432 103004          BCC     10$           ;BR IF FOUND FILE
82 003434          FABORT  #CSIMS4    ;CAN'T FIND FILE
83
84          ; Protect the file
85
86 003444 010205          10$:    MOV      R2,R5          ;GET LOGICAL DISK UNIT #
87 003446 006305          ASL      R5              ;* 4 WORDS PER ENTRY
88 003450 006305          ASL      R5
89 003452 062705 000000G  ADD      #LDNAME,R5     ;POINT TO ENTRY WITH FILE NAME
90 003456 112767 000001 000000G  MOVVB   #1,SERFLG      ;DO .SERR TO AVOID ABORTS ON ERRORS
91 003464          .FPROT #XAREA,#1,R5,#1
92 003510 105067 000000G  CLR     SERFLG         ;DO .HERR
93
94          ; See if a logical device name was specified following the file name
95
96 003514 122327 000040          16$:    CMPB   (R3)+,#'      ;SKIP BLANKS
97 003520 001775          BEQ     16$
98 003522 105743          TSTB   -(R3)          ;WAS A LOGICAL DEVICE NAME SPECIFIED?
99 003524 001410          BEQ     MNTCSH        ;BR IF NOT
100
101          ; Assign logical name to logical disk
102
103 003526 004767 000000G  CALL    GTRD50         ;ACCRUE LOGICAL DEVICE NAME
104 003532 016705 000000G  MOV     R50BUF,R5     ;GET LOGICAL DEVICE NAME
105 003536 016700 000000G  MOV     MNTDEV,R0     ;GET PHYSICAL DEVICE NAME
106 003542 004767 000000G  CALL    DOASGN         ;DO THE ASSIGNMENT
107
108          ; Initiate directory caching for the device whose name is in MNTDEV
109
110 003546 012700 000000G  MNTCSH: MOV    #MNTARG,R0 ;POINT TO MOUNT ARGUMENT BLOCK
111 003552 004767 000010          CALL    DOMNT         ;ENABLE CASHING FOR DEVICE
112
113          ; Now do a SET LD CLEAN operation to make sure all logical disks
114          ; are set up correctly

```

MOUNT command

```
115 ;  
116 003556 004767 0000000 CALL LDCLN ; DO SET LD CLEAN  
117 ;  
118 ; Finished with MOUNT command  
119 ;  
120 003562 000167 0000000 JMP RDCMD
```

MOUNT command

```

1 ;-----
2 ; DOMNT is called to execute a mount or ,dismount EMT and check for
3 ; errors. If an error is detected, an error message is generated.
4 ;
5 ; Inputs:
6 ; RO = Address of EMT argument block for mount or dismount emt.
7 ;
8 003566 104375 DOMNT: EMT 375 ;DO THE EMT (MOUNT OR DISMOUNT)
9 003570 103021 BCC 9$ ;BR IF NO ERROR
10 ;
11 ; Error on mount or dismount emt.
12 ;
13 003572 123727 0000000 000001 CMPB @#ERRLOC,#1 ;MOUNT TABLE OVERFLOW?
14 003600 001007 BNE 2$ ;BR IF NOT
15 003602 FWARN #MNTFUL ;MOUNT TABLE OVERFLOW
16 003616 000406 BR 9$
17 003620 2$: FERR #BDFNAM ;INVALID DEVICE NAME
18 ;
19 ; Finished
20 ;
21 003634 000207 9$: RETURN
22 ;-----
23 ;
24 ; LDDEVN is called to determine if a device name is of the form LDn and
25 ; if so, to determine the unit number.
26 ;
27 ; Inputs:
28 ; RO = Device name (rad50)
29 ;
30 ; Outputs:
31 ; C-flag cleared ==> Device is LDn
32 ; C-flag set ==> Device is not LDn
33 ; RO = LD device index number (2*n)
34 ;
35 003636 020067 0000000 LDDEVN: CMP RO,R5OLD ;Is name "LD"?
36 003642 001002 BNE 1$ ;Br if not
37 003644 016700 0000000 MOV R5OLD0,RO ;Replace "LD" by "LDO"
38 003650 020067 0000000 1$: CMP RO,R5OLD0 ;Is name in the range LDO to LD7?
39 003654 103410 BLD 2$ ;Br if not
40 003656 020067 0000000 CMP RO,R5OLD7
41 003662 101005 BHI 2$
42 003664 166700 0000000 SUB R5OLD0,RO ;Get unit number only
43 003670 006300 ASL RO ;Convert to device index number
44 003672 000241 CLC ;Signal success on return
45 003674 000401 BR 3$
46 003676 000261 2$: SEC ;Signal failure on return
47 003700 000207 3$: RETURN

```

DISMOUNT command

```

1          .SBTTL  DISMOUNT  command
2          ;-----
3          ; Dismount a file structure.
4          ;
5 003702  004767  0000000  CMDDMT: CALL    CVTTAB          ; CONVERT TAB AND FF CHARS TO SPACES
6          ;
7          ; See if any qualifier were specified following command keyword
8          ;
9 003706  105067  174122    CLRFB    DMTNLG          ; Clear flag set if /NOLOG specified
10 003712  004767  0000000  CALL    SKPSPC          ; Skip over any spaces
11 003716  121327  000057    CMPB    (R3),#'/        ; Any qualifiers here?
12 003722  001004          BNE     5$              ; Br if not
13          ;
14          ; Process qualifier following command keyword
15          ;
16 003724  012704  000174'  MOV     #DMTQHD,R4      ; Point to parsing table
17 003730  004767  0000000  CALL    OPTLST          ; Process any qualifiers
18          ;
19          ; Skip leading spaces and make sure a device name was specified
20          ;
21 003734  004767  0000000  5$:    CALL    SKPSPC          ; SKIP LEADING SPACES
22 003740  105713          TSTB    (R3)            ; WAS A DEVICE NAME SPECIFIED
23 003742  001004          BNE     6$              ; BR IF YES
24 003744          7$:    FABORT  #ILLCMD          ; INVALID COMMAND IF NO DEVICE NAME
25          ;
26          ; Get name of device that is being dismounted
27          ;
28 003754  004767  0000000  6$:    CALL    GTRD50          ; ACCRUE THE DEVICE NAME
29 003760  016700  0000000  MOV     R50BUF,R0      ; GET DEVICE NAME
30 003764  001767          BEQ     7$              ; BR IF NO DEVICE NAME
31 003766  004767  0000000  2$:    CALL    ASNSRC          ; SEE IF IT IS A LOGICAL NAME
32 003772  103403          BCS     1$              ; BR IF NOT
33 003774  016200  0000000  MOV     AT$DEV(R2),R0  ; GET ASSIGNED NAME
34 004000  000772          BR     2$              ; ALLOW INDIRECT ASSIGNS
35 004002  020067  0000000  1$:    CMP     R0,R50LD      ; IS NAME "LD"?
36 004006  001002          BNE     4$              ; BR IF NOT
37 004010  016700  0000000  MOV     R50LD0,R0      ; TRANSLATE LD TO LDO
38 004014  010067  0000000  4$:    MOV     R0,MNTDEV      ; SAVE NAME OF DEVICE BEING DISMOUNTED
39          ;
40          ; Close the log file if it is on the device being dismounted
41          ;
42 004020  016705  0000000  MOV     MNTDEV,R5      ; GET NAME OF DEVICE BEING DISMOUNTED
43 004024  004767  0000000  CALL    LOGCHK          ; SEE IF LOG FILE IS ON THAT DEVICE
44          ;
45          ; Print information message if device is still mounted by other jobs
46          ;
47 004030  105767  174000    TSTB    DMTNLG          ; Was /NOLOG qualifier specified?
48 004034  001013          BNE     3$              ; Br if yes -- Suppress information message
49 004036  016705  0000000  MOV     MNTDEV,R5      ; GET NAME OF DEVICE BEING DISMOUNTED
50 004042  004767  0000000  CALL    CHKMNT          ; IS THIS DEVICE MOUNTED?
51 004046  103406          BCS     3$              ; BR IF NOT
52 004050  004767  0000000  CALL    CHKMTX          ; IS DEVICE MOUNTED BY ANYONE OTHER THAN US?
53 004054  103003          BCC     3$              ; BR IF NOT
54 004056          .PRINT  #INFOMT          ; STILL MOUNTED BY OTHER USERS
55          ;
56          ; Tell system to stop doing directory caching for the device
57          ;

```

DISMOUNT command

```

58 004064 112767 000001 0000006 3#:      MOV      #1,SERFLG      ;DO .SERR TO AVOID ABORT FOR ILLEGAL DEVICE
59 004072 012700 0000000      MOV      #DMTARG,R0    ;POINT TO EMT ARGUMENT BLOCK
60 004076 104375      EMT      375           ;TELL SYSTEM TO STOP DOING CACHING
61 004100 105067 0000000      CLRB     SERFLG       ;DO .HERR
62                                     ;
63                                     ; See if a logical disk is being dismantled
64                                     ;
65 004104 016702 0000000      MOV      MNTDEV,R2    ;GET DEVICE NAME
66 004110 020267 0000000      CMP      R2,R5OLD0    ;IS THIS A LOGICAL DISK?
67 004114 103420      BLO      9#           ;BR IF NOT
68 004116 020267 0000000      CMP      R2,R5OLD7
69 004122 101015      BHI      9#
70 004124 166702 0000000      SUB      R5OLD0,R2    ;GET UNIT #
71 004130 006302      ASL      R2           ;CONVERT TO WORD INDEX
72 004132 005062 0000000      CLR      LDPDEV(R2)   ;NO PHYSICAL DEVICE ASSIGNMENT
73 004136 006302      ASL      R2           ;GET INDEX INTO LDNAME TABLE
74 004140 006302      ASL      R2
75 004142 005062 0000000      CLR      LDNAME(R2)   ;NO FILE NAME
76 004146 016700 0000000      MOV      MNTDEV,R0    ;GET NAME OF LOGICAL DISK
77 004152 004767 0000000      CALL     DEADEV       ;DEASSIGN ANY LOGICAL NAMES
78                                     ;
79                                     ; Finished
80                                     ;
81 004156 004767 0000000      9#:      CALL     LDCLN      ;RESET INFORMATION ABOUT LOGICAL DISKS
82 004162 000167 0000000      JMP      RDCMD        ;FINISHED WITH COMMAND
83                                     ;
84                                     ; Process /NOLOG qualifier for DISMOUNT command
85                                     ;
86 004166 105267 173642      DMTNLQ: INCB     DMTNLQ ;Remember qualifier specified
87 004172 000207      DMTLQ@: RETURN

```

INITIALIZE command

```

1          .SBTTL  INITIALIZE command
2          .SBTTL  SQUEEZE command
3          .SBTTL  FORMAT command
4          ;-----
5          ; Disallow INITIALIZE and SQUEEZE commands if any other users have
6          ; the specified device mounted.
7          ;
8 004174   CMDINI:
9 004174   CMDSQZ:
10 004174  CMDFMT:
11          ;
12          ; See if user is authorized to use these commands
13          ;
14 004174  032767  000000C 000000G      BIT      #PO$NFW!PO$BYP,PRIVCO  ;Authorized for these commands?
15 004202  001004          BNE      13$          ;Br if yes
16 004204          FABORT  #EM$NFW          ;Not authorized for this command
17          ;
18          ; Skip over any options specified with command keyword
19          ;
20 004214  004767  000000G      13$:    CALL      CVTTAB          ;Convert tabs to spaces
21 004220  121327  000057      CMPB     (R3),#/'          ;Were any keyword options specified?
22 004224  001000          BNE      10$          ;Br if not
23 004226  112300          1$:     MOVB     (R3)+,R0          ;Get next character from command
24 004230  001407          BEQ      15$          ;Br if no device was specified
25 004232  120027  000040      CMPB     R0,#40          ;Is this a space?
26 004236  001373          BNE      1$          ;Keep looking if not
27          ;
28          ; Accrue the device name
29          ;
30 004240  004767  000000G      10$:    CALL      SKPSPC          ;Skip over any spaces
31 004244  105713          TSTB     (R3)          ;Was a device name specified?
32 004246  001004          BNE      14$          ;Br if yes
33 004250          15$:    FABORT  #EM$DNR          ;Device name is required
34 004260  004767  000000G      14$:    CALL      GTRD50          ;Accrue the device name
35 004264  016705  000000G      MOV      R5,OBUFF,R5
36 004270  010567  000000G      MOV      R5,MNTDEV          ;Set name for mount/dismount
37 004274  010567  173500      MOV      R5,SQZDEV          ;Save name of device being squeezed
38 004300  010567  000000G      MOV      R5,ALCDEV          ;Set name for test-allocation EMT
39          ;
40          ; Disallow SQUEEZE/INITIALIZE/FORMAT of any disk with a system file
41          ;
42 004304  004767  000232          CALL      CKSYDV          ;See if device has any system files
43 004310  103004          BCC      11$          ;Br if not
44 004312          FABORT  #EM$SSY          ;Can't do this to SY
45          ;
46          ; See if this device is mounted by anyone
47          ;
48 004322  004767  000000G      11$:    CALL      CHKMNT          ;See if device is mounted by anyone
49 004326  103412          BCS      8$          ;Br if not
50          ;
51          ; This device is in the mount table (R5=pointer to entry).
52          ; See if this device is mounted by any other users
53          ;
54 004330  004767  000000G          CALL      CHKMTX          ;Is device mounted by any other users?
55 004334  103004          BCC      2$          ;Br if not
56 004336          3$:     FABORT  #QTRMNT          ;Command not legal if mounted by others
57          ;

```

FORMAT command

```

58      ; Dismount and remount this device to clean out all file entries
59      ; associated with this device.
60      ;
61 004346 012700 0000000 2$:      MOV      #MNTARG,R0      ;Remount the device
62 004352 104375      EMT      375
63      ;
64      ; See if this device is allocated to any other user
65      ;
66 004354 120467 0000000 8$:      CMPB     R4,LDDEVX      ;Is this a logical disk?
67 004360 001404      BEQ      12$              ;Br if yes -- Don't check for allocation
68 004362 016700 0000000      MOV      ALCDEV,R0      ;Get RAD50 device name
69 004366 004767 0000000      CALL     CHKALC        ;See if device is allocated to anyone else
70      ;
71      ; Remove from mount table any logical disks that are on the device
72      ; being squeezed or initialized.
73      ;
74 004372 016705 173402 12$:      MOV      SQZDEV,R5      ;Get back device name
75 004376 004767 0000000      CALL     CHKDEV       ;Convert device name to dev # & unit #
76 004402 020467 0000000      CMP      R4,LDDEVX    ;Is this a logical disk?
77 004406 001406      BEQ      4$              ;Br if yes
78 004410 005002      CLR      R2              ;Say base block # = 0
79 004412 012703 177777      MOV      #177777,R3    ;Say top block = 177777
80 004416 000300      SWAB     R0              ;Put unit # in high byte
81 004420 050004      BIS      R0,R4         ;Get unit # and device # together
82 004422 000410      BR      5$              ;
83 004424 006300      4$:      ASL      R0              ;Convert unit # to word table index
84 004426 016004 0000000      MOV      LDPDEV(R0),R4 ;Get physical dev # and unit #
85 004432 016002 0000000      MOV      LDBASE(R0),R2 ;Get base block of logical disk
86 004436 010203      MOV      R2,R3
87 004440 066003 0000000      ADD      LDSIZE(R0),R3 ;Get top block number of logical disk
88      ;
89      ; Search for any mounted logical disks that are on the disk being squeezed
90      ;
91 004444 016705 0000000 5$:      MOV      CSHDEV,R5      ;Point to start of mount table
92 004450 010500 7$:      MOV      R5,R0         ;Get address of mount entry
93 004452 004767 0000000      CALL     CDGET        ;Copy mount entry into CDBUF
94 004456 020467 0000000      CMP      R4,CDBUF+CD$DVU ;Is this device on same physical device?
95 004462 001014      BNE      6$              ;Br if not
96 004464 016700 0000000      MOV      CDBUF+CD$BAS,R0 ;Is this a logical disk?
97 004470 001411      BEQ      6$              ;Br if not
98 004472 020002      CMP      R0,R2         ;See if logical disk is within disk being sqze
99 004474 101407      BLOS     6$              ;Br if not
100 004476 020003      CMP      R0,R3         ;Compare upper range
101 004500 103005      BHS      6$
102      ;
103      ; We found a logical disk within the disk being squeezed.
104      ; See if anyone else has that logical disk mounted.
105      ;
106 004502 004767 0000000      CALL     CHKMTX       ;Anyone else have this logical disk mounted?
107 004506 103713      BCS      3$              ;Error if yes
108      ;
109      ; Dismount this logical disk and any files associated with it
110      ;
111 004510 004767 0000000      CALL     DMTSUB       ;Do the dismount
112      ;
113      ; Check for more nested logical disks
114      ;

```

FORMAT command

```

115 004514 062705 0000000 6#: ADD #CD##5Z,R5 ;Point to next mount table entry
116 004520 020567 0000000 CMP R5,CSHDVN ;Reached end of table?
117 004524 103751 BLO 7# ;Loop if not
118 ;
119 ; See if log file is opened to device being squeezed
120 ; If so, close it.
121 ;
122 004526 016705 173246 MOV SQZDEV,R5 ;Get name of device being squeezed
123 004532 004767 0000000 CALL LOGCHK ;See if log file is on that device
124 ;
125 ; Call CCL to process the command
126 ;
127 004536 000167 0000000 9#: JMP CMDCCCL ;Call CCL to process the command

```

FORMAT command

```

1
2 ; -----
3 ; Check to see if a device being squeezed or initialized contains any
4 ; system files.
5 ;
6 ; Inputs:
7 ; R5 = Rad50 device name of device being squeezed or initialized
8 ;
9 ; Outputs:
10 ; C-flag set ==> This device contains system files
11 004542 010246 CKSYDV: MOV R2, -(SP)
12 004544 010346 MOV R3, -(SP)
13 004546 010446 MOV R4, -(SP)
14 004550 010546 MOV R5, -(SP)
15 ;
16 ; Convert device name to device index and unit number
17 ;
18 004552 004767 0000000 CALL CHKDEV ;Convert name to device index & unit #
19 004556 103422 BCS B$ ;Br if device name not recognized
20 ;
21 ; At this point: R0 = unit number, R4 = device index number.
22 ; Begin loop to check each system file savestatus block.
23 ;
24 004560 012705 004640' MOV #SYFLVC, R5 ;Point to vector of savestatus addresses
25 004564 012503 1$: MOV (R5)+, R3 ;Point to next savestatus block
26 004566 001416 BEQ B$ ;Br if reached end of list
27 004570 016302 0000000 MOV C, CSW(R3), R2 ;Get 1st word of channel
28 004574 042702 177701 BIC #^C<76>, R2 ;Extract device index number
29 004600 020204 CMP R2, R4 ;Does it match device being squeezed?
30 004602 001370 BNE 1$ ;Br if not
31 004604 116302 0000000 MOVB C, DEV0(R3), R2 ;Get unit number
32 004610 042702 177770 BIC #^C<7>, R2 ;Extract unit number
33 004614 020200 CMP R2, R0 ;Same as unit of device being squeezed?
34 004616 001362 BNE 1$ ;Br if not
35 ;
36 ; This device contains a system file
37 ;
38 004620 000261 SEC ;Signal error on return
39 004622 000401 BR 9$
40 ;
41 ; This device does not contain a system file
42 ;
43 004624 000241 8$: CLC ;Signal no error
44 ;
45 ; Finished
46 ;
47 004626 012605 9$: MOV (SP)+, R5
48 004630 012604 MOV (SP)+, R4
49 004632 012603 MOV (SP)+, R3
50 004634 012602 MOV (SP)+, R2
51 004636 000207 RETURN
52 ;
53 ; List of addresses of savestatus blocks for system files
54 ;
55 004640 0000000 SYFLVC: .WORD SWPCHN ;Swap file
56 004642 0000000 .WORD SEGCHN ;PLAS swap file
57 004644 0000000 .WORD SPLCHN ;Spool file

```

FORMAT command

58	004646	0000000	.WORD	KMNCHN	;TSKMON.SAV
59	004650	0000000	.WORD	CCLSAV	;CCL.SAV
60	004652	0000000	.WORD	INDSAV	;IND.SAV
61	004654	0000000	.WORD	INDTSV	;TSXIND.TSX
62	004656	0000000	.WORD	0	;End of list

MONITOR command

```

1          .SBTTL  MONITOR  command
2          ;-----
3          ; The MONITOR command is used to initiate performance monitoring.
4          ; The form of the command is:
5          ; MONITOR base-address,top-address[, #-bytes-per-cell]/switches
6          ;
7 004660 005067 0000000  CMDMON: CLR      MNFLGS      ; INITIALIZE EMT ARG BLOCK
8 004664 005067 0000000          CLR      MNBPC
9 004670 004767 0000000          CALL    CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
10 004674 004767 0000000         CALL    DELSPC      ; DELETE SPACES FROM COMMAND LINE
11 004700 004767 0000000         CALL    ACROCT      ; ACCRUE BASE ADDRESS
12 004704 010167 0000000         MOV     R1,MNBASE    ; SET IN EMT ARG BLOCK
13 004710 122327 000054          CMPB   (R3)+,#'      ; COMMA SHOULD BE DELIMITER
14 004714 001075                BNE    3$
15 004716 004767 0000000         CALL    ACROCT      ; ACCRUE TOP ADDRESS
16 004722 010167 0000000         MOV     R1,MNTOP
17 004726 112300                MOVB   (R3)+,R0      ; GET DELIMITER
18 004730 001430                BEQ    1$            ; BR IF FINISHED WITH COMMAND
19 004732 120027 000054          CMPB   R0,#'        ; DID HE SPECIFY NBPC?
20 004736 001006                BNE    2$            ; BR IF NOT
21 004740 004767 0000000         CALL    ACRDEC      ; ACCRUE NUMBER OF BYTES PER CELL
22 004744 010167 0000000         MOV     R1,MNBPC
23 004750 112300                4$:  MOVB   (R3)+,R0      ; GET DELIMITER
24 004752 001417                BEQ    1$            ; BR IF END OF COMMAND HIT
25 004754 120027 000057          2$:  CMPB   R0,#'/      ; GOT A SWITCH?
26 004760 001053                BNE    3$            ; BR IF NOT
27 004762 012704 000234'         MOV     #MONHD,R4   ; POINT TO TABLE OF SWITCH OPTIONS
28 004766 004767 0000000         CALL    SEARCH      ; SEARCH FOR SWITCH IN TABLE
29 004772 103004                BCC    5$            ; BR IF FOUND IT
30 004774                FABORT  #INVOPT    ; INVALID SWITCH
31 005004 051467 0000000          5$:  BIS     (R4),MNFLGS ; SET BITS FOR SWITCH
32 005010 000757                BR     4$            ; GO CHECK FOR MORE SWITCHES
33          ;
34          ; Finished scanning command.
35          ; Initiate performance monitoring.
36          ;
37 005012 012700 0000000          1$:  MOV     #MONAR1,R0 ; POINT TO EMT ARG BLOCK
38 005016 104375                EMT    375           ; TRY TO INITIATE MONITORING
39 005020 103026                BCC    6$            ; BR IF OK
40 005022 105737 0000000         TSTB  @#ERRLOC      ; CHECK ERROR CODE
41 005026 001404                BEQ    7$            ; BR IF SOMEONE ELSE DOING MONITORING NOW
42 005030                FABORT  #NOPMON    ; PM NOT GENNED IN
43 005040 010046                7$:  MOV     R0,-(SP)    ; SAVE # OF PM USER
44 005042                FERR    #PMBUSY    ; PM FACILITY IN USE BY ANOTHER USER
45 005056 012605                MOV     (SP)+,R5    ; GET # OF PM USER
46 005060 004767 0000000         CALL    PRTDEC      ; PRINT DECIMAL VALUE
47 005064                .PRINT #CRLF      ; TERMINATE THE ERROR LINE
48 005072 000167 0000000         JMP    RDCMD
49 005076 012700 0000000          6$:  MOV     #MONAR2,R0 ; START MONITORING
50 005102 104375                EMT    375
51 005104 000167 0000000         JMP    RDCMD
52          ;
53          ; Invalid command syntax.
54          ;
55 005110                3$:  FABORT  #CSIMS1

```

SPOOL command

```

1          .SBTTL  SPOOL command
2          ;-----
3          ; THE SPOOL COMMAND IS USED TO CONTROL THE OPERATION
4          ; OF SPOOLED DEVICES.
5          ; THE FORM OF THE COMMAND IS:
6          ;   SPOOL DEV,OPTION,OPTION
7          ;
8 005120  004767  0000000  CMDSPD: CALL   CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACE
9 005124  004767  0000000          CALL   CVTUC       ; Convert lower case chars to upper case
10 005130  105713          TSTB   (R3)        ; ANY DEVICE NAMED?
11 005132  001002          BNE    1$          ; BRANCH IF YES
12 005134  000167  0000000          JMP    BADCMD      ; MUST HAVE DEVICE NAME
13          ;
14          ; ACCRUE DEVICE NAME IN RAD50 FORM.
15          ;
16 005140  004767  0000000  1$:   CALL   GTRD50      ; ACCRUE THE NAME
17          ;
18          ; If device name is a logical name, translate to physical
19          ;
20 005144  016700  0000000          MOV    R50BUF,R0    ; Get device name
21 005150  004767  0000000          CALL  ASNSRC      ; See if this name is assigned
22 005154  103403          BCS   3$          ; Br if name is not assigned
23 005156  016267  0000000 0000000  MOV    AT$DEV(R2),R50BUF ; Set physical device name
24          ;
25          ; NOW TRY TO FIND SDCB WHICH MATCHES NAME.
26          ;
27 005164  010346          3$:   MOV    R3,-(SP)    ; SAVE POINTER
28 005166  012703  0000000          MOV    #R50BUF,R3   ; POINT TO DEVICE NAME
29 005172  004767  0000000          CALL  FORCE0        ; MAKE UNIT 0 IF NEEDED
30 005176  012603          MOV    (SP)+,R3     ; RESTORE POINTER
31 005200  012701  0000000          MOV    #SDCB,R1    ; POINT TO 1ST SDCB
32 005204  020127  0000000          4$:   CMP    R1,#SDCBND ; CHECKED ALL?
33 005210  103013          BHS   2$          ; BRANCH IF YES
34 005212  026127  0000000 0000000  CMP    SDNAME(R1),#DMYDEV ; Is this dummy device entry?
35 005220  001404          BEQ   5$          ; Br if yes
36 005222  026761  0000000 0000000  CMP    R50BUF,SDNAME(R1) ; DO NAMES MATCH?
37 005230  001407          BEQ   CSGETOP     ; BRANCH IF FOUND SDCB
38 005232  062701  0000000          5$:   ADD    #SDCBSZ,R1 ; GO CHECK NEXT SDCB
39 005236  000762          BR    4$
40          ;
41          ; CAN'T FIND SDCB FOR DEVICE.
42          ;
43 005240          2$:   FABORT  #INVDEV
44          ;
45          ; FOUND SDCB (R1).
46          ; IDENTIFY OPTION.
47          ;
48 005250  112300          CSGETOP: MOVB   (R3)+,R0 ; SKIP OVER ANY DELIMITERS
49 005252  004767  0000000          CALL  CHKDLH
50 005256  103774          BCS   CSGETOP
51 005260  005303          DEC   R3
52 005262  012704  000260'          MOV    #SPLHD,R4   ; POINT TO TABLE OF SPOOL OPTIONS
53 005266  004767  0000000          CALL  SEARCH      ; LOOK UP OPTION
54 005272  103401          BCS   1$          ; BR IF ERROR
55          ;
56          ; FOUND OPTION. BRANCH OFF TO OPTION PROCESSING ROUTINE.
57          ; SDCB ADDRESS IS IN R1.

```

SPUOL command

```
58 ;
59 005274 000134 ;          JMP      @(R4)+
60 ;
61 ;  ERROR IN SEARCH
62 ;
63 005276 005704 1#:      TST      R4          ; ILLEGAL OR AMBIGUOUS
64 005300 001404      BEQ      2#          ; BR IF ILLEGAL
65 005302              FABORT  #AMBOPT    ; AMBIGUOUS OPTION
66 005312              2#:      FABORT  #INVOPT ; ILLEGAL OPTION
```

SPOOL command

```

1          ; .....
2          ;
3          ; PROCESS THE 'DEL' (DELETE) OPTION.
4          ;
5 005322   SPLDEL:
6          ;
7          ; See if a file ID was specified
8          ;
9 005322   105713          TSTB      (R3)          ;Was an id specified?
10 005324   001425          BEQ       SPDLCR         ;Br if not
11 005326   121327   000054   CMPB      (R3),#',          ;Comma used as delimiter?
12 005332   001403          BEQ       1$           ;Br if yes
13 005334   121327   000075   CMPB      (R3),#'=        ;Equal sign used as delimiter?
14 005340   001001          BNE      2$           ;Br if not -- space must have been delimiter
15 005342   005203          1$:      INC       R3           ;Skip over comma or equal sign
16          ;
17          ; A file id was specified, accrue the id value
18          ;
19 005344   004767   000000G   2$:      CALL      ACRDEC          ;Accrue the value
20 005350   010167   172470          MOV       R1,DELEMT+4      ;Set file ID in EMT arg block
21 005354   012700   000040'          MOV       #DELEMT,R0      ;Point to EMT arg block
22 005360   104375          EMT      375             ;Try to delete the file
23 005362   103004          BCC      9$           ;Br if successful
24 005364          FABORT   #EM#NID          ;Cannot find file with that id
25 005374   000167   000000G   9$:      JMP       RDCMD
26          ;
27          ; Abort the file currently being printed by the spooled device
28          ;
29 005400   005761   000000G   SPDLCR: TST      SDSFCB(R1)      ;IS DEVICE BUSY?
30 005404   001405          BEQ      SPNBSY          ;BRANCH IF NOT
31 005406   152761   000000G 000000G   BISB     #SD$DEL,SDFLAG(R1) ;SET DELETE FLAG
32 005414   000167   000000G   CSABT:  JMP      RDCMD
33 005420          SPNBSY: .PRINT #DEVIDL
34 005426   000772          BR       CSABT

```

SPOOL command

```

1          ; -----
2          ; PROCESS THE SPOOL 'ALIGN' OPTION.
3          ; (SPOOL XX, ALIGN, <FILE NAME>)
4          ;
5 005430 004767 0000000 SPLALN: CALL CKPRIV ;Must have operator privilege
6 005434 122327 000054      CMPB (R3)+, #' ;WAS FILE NAME GIVEN?
7 005440 001404      BEQ 1$ ;BR IF YES
8 005442      9$: FABORT #COAL
9          ; PARSE ALIGNMENT FILE NAME
10 005452 010605      1$: MOV SP, R5 ;SAVE SP
11 005454      .CSIGEN #ALDEX, #ALDEX, R3; OPEN ALIGNEMENT FILE - CHAN 3
12 005470 010506      MOV R5, SP ;RESET SP
13 005472 103763      BCS 9$ ;BRANCH IF .CSIGEN ERROR
14          ; ALIGNEMENT FILE IS NOW OPEN ON CHANNEL 3
15          ; OPEN SPOOLED DEVICE ON CHANNEL 1
16          ;
17 005474 016167 0000000 0000000      MOV SDNAME(R1), ALDBLK; SET UP DEVICE NAME
18 005502      .ENTER #XAREA, #1, #ALDBLK, #0; OPEN SPOOLED DEVICE
19 005526 103010      BCC 2$ ;BR IF OPEN OK
20 005530      .PRINT #COAD ;CAN'T OPEN SPOOLED DEVICE
21 005536      .CLOSE #3 ;CLOSE ALIGNMENT FILE
22 005544 000167 0000000      10$: JMP RDCMD
23          ;
24          ; WRITE 1ST RECORD TO SPOOLED FILE AND SPECIFY A
25          ; FORM NAME OF "*****" WHICH MEANS ALIGNMENT FILE.
26          ;
27 005550      2$: .WRITW #XAREA, #1, #ALFN, #4, #0
28          ;
29          ; NOW COPY ALIGNMENT FILE TO SPOOLED FILE
30          ;
31 005606 005007      CLR R2 ;INIT BLOCK #
32 005610      4$: .READW #XAREA, #3, #BLKO, #256, R2
33 005646 103421      BCS 3$ ;BR IF END OF FILE
34 005650      .WRITW #XAREA, #1, #BLKO, #256, R2
35 005706 005207      INC R2
36 005710 000737      BR 4$
37          ; HIT END OF ALIGNMENT FILE. CLOSE BOTH CHANNELS.
38 005712      3$: .CLOSE #1
39 005720      .CLOSE #3
40 005726 000706      BR 10$
41          ;
42          ; PROCESS THE 'LOCK' OPTION.
43          ;
44 005730 004767 0000000 SPLLK: CALL CKPRIV ;Must have operator privilege
45 005734 052761 0000000 0000000      BIS #SD#FLK, SDFLAG(R1); SAY FORM IS LOCKED
46 005742 000405      BR SPLMT1 ;NOW DO MOUNT
47          ;
48          ; PROCESS THE 'FORM' OPTION
49          ;
50 005744 004767 0000000 SPLFRM: CALL CKPRIV ;Must have operator privilege
51 005750 042761 0000000 0000000      BIC #SD#FLK, SDFLAG(R1); SAY FORM IS NOT LOCKED
52 005756 010107      SPLMT1: MOV R1, R2 ;POINT TO FORM NAME CELL
53 005760 062702 0000000      ADD #SDFORM, R2
54 005764 010204      MOV R2, R4 ;POINT TO END OF CELL
55 005766 062704 000006      ADD #6, R4
56 005772 005203      INC R3 ;SKIP OVER COMMA
57 005774 105713      3$: TSTB (R3) ;HIT END OF NAME?

```

SPOOL command

```

58 005776 001404          BEQ      1#           ;BR IF YES
59 006000 020204          CMP      R2,R4         ;COPIED 6 CHARS?
60 006002 103007          BHIS     2#           ;BR IF YES
61 006004 112322          MOVB    (R3)+,(R2)+   ;COPY IN FORM NAME
62 006006 000772          BR      3#
63                          ; PAD END OF SHORT NAME WITH BLANKS
64 006010 020204          1#:    CMP      R2,R4         ;FILLED TO 6 CHARS?
65 006012 103003          BHIS     2#           ;BR IF YES
66 006014 112722 000040    MOVB    #' ,(R2)+   ;PUT IN TRAILING SPACES
67 006020 000773          BR      1#
68                          ;
69                          ; NOW TRY TO START SPOOLER
70                          ;
71 006022 042761 0000000 0000000 2#:    BIC      #SD$WFH,SDFLAG(R1);SAY FORM MOUNT IS DONE
72 006030 010100          MOV     R1,R0         ;POINT TO SDCB
73 006032 004767 000464    CALL    SPOLGO        ;TRY TO START SPOOLER
74 006036 000167 0000000    JMP     RDCMD
75                          ;
76                          ; PROCESS THE 'SKIP' OPTION
77                          ;
78 006042 004767 0000000    SPLSKP: CALL    CKPRIV      ;Must have operator privilege
79 006046 005761 0000000    TST     SDSFCB(R1)    ;IS DEVICE BUSY?
80 006052 001410          BEQ     SPNBB         ;BR IF NOT
81                          ; ACCRUE NUMBER
82 006054 010104          MOV     R1,R4         ;SAVE SDCB ADDRESS
83 006056 005203          INC     R3            ;SKIP OVER COMMA
84 006060 004767 0000000    CALL    ACRDEC        ;ACCRUE DEC NUMBER
85 006064 010164 0000000    MOV     R1,SDSKIP(R4) ;SET SKIP COUNT IN SDCB
86 006070 000167 0000000    JMP     RDCMD
87                          ;
88                          ; PROCESS THE 'BACK' OPTION
89                          ;
90 006074 004767 0000000    SPLBAK: CALL    CKPRIV      ;Must have operator privilege
91 006100 005761 0000000    TST     SDSFCB(R1)    ;IS THE DEVICE BUSY
92 006104 001002          BNE     SPLBK1        ;BR IF YES
93 006106 000167 177306    SPNBB:  JMP     SPNBSY    ;GO GIVE ERROR MESSAGE
94 006112 005761 0000000    SPLBK1: TST     SDBU(R1)   ;ANY BLOCKS REMEMBERED YET?
95 006116 001403          BEQ     1#           ;BR IF NOTHING TO BACKUP TO
96 006120 052761 0000000 0000000    BIS     #SD$BAK,SDFLAG(R1);REQUEST BACK UP
97 006126 000167 0000000    1#:    JMP     RDCMD

```

SPOOL command

```

1
2 ; PROCESS THE SPOOL 'STATUS' COMMAND
3 ;
4 006132 005761 0000000 SPLSTA: TST SDFSFCB(R1) ; IS SPOOLER BUSY?
5 006136 001404 BEQ 1$ ; BR IF NOT
6 006140 .PRINT #SPACTV
7 006146 000413 BR 4$
8 006150 032761 0000000 0000000 1$: BIT #SD$WFM,SDFLAG(R1); IS IT WAITING FOR A FORM MOUNT?
9 006156 001404 BEQ 2$
10 006160 .PRINT #SPWFM
11 006166 000403 BR 4$
12 006170 2$: .PRINT #DEVIDL
13 ; SEE IF IN SINGLE FILE MODE
14 006176 032761 0000000 0000000 4$: BIT #SD$SNG,SDFLAG(R1); IN SINGLE-FILE MODE?
15 006204 001403 BEQ 20$ ; BR IF NOT
16 006206 .PRINT #SPSNG
17 ; SEE IF SPOOL FILE IS FULL
18 006214 005767 0000000 20$: TST NFRESB ; IS SPOOL FILE FULL?
19 006220 001003 BNE 3$ ; BR IF NOT
20 006222 .PRINT #SPFUI
21 ; LIST CURRENT FORM NAME
22 006230 3$: .PRINT #SPCF
23 006236 010102 MOV R1,R2 ; POINT TO FORM NAME CELL
24 006240 062702 0000000 ADD #SDFORM,R2
25 006244 012703 0000006 MOV #6,R3 ; PRINT 6 CHARS
26 006250 112200 5$: MOVB (R2)+,R0
27 006252 .TTYOUT
28 006256 005303 DEC R3
29 006260 001373 BNE 5$
30 ; SEE IF CURRENT FORM IS LOCKED
31 006262 032761 0000000 0000000 BIT #SD$FLK,SDFLAG(R1); IS FORM LOCKED?
32 006270 001404 BEQ 6$ ; BR IF NOT
33 006272 .PRINT #SPFLK
34 006300 000403 BR 7$
35 006302 6$: .PRINT #CRLF
36 ; List info about files pending for this device.
37 006310 005761 0000000 7$: TST SDFHD(R1) ; ARE THERE ANY FILES PENDING FOR THIS DEVICE?
38 006314 001004 BNE 11$ ; BR IF YES
39 006316 .PRINT #NOFIL ; NO FILES...
40 006324 000410 BR 8$
41 006326 11$: .PRINT #QHMS1 ; PRINT HEADING MESSAGE
42 006334 .PRINT #QHMS2 ; UNDERLINE THE HEADING
43 006342 004767 0000000 CALL LSTSPL ; LIST SPOOL FILES
44 006346 000167 0000000 8$: JMP RDCMD
45 ;
46 ; PROCESS THE 'SINGLE' OPTION
47 ;
48 006352 004767 0000000 SPLSNG: CALL CKPRIV ; Must have operator privilege
49 006356 052761 0000000 0000000 BIS #SD$SNG,SDFLAG(R1); SET SINGLE-FILE MODE
50 006364 000167 0000000 JMP RDCMD
51 ;
52 ; PROCESS THE 'MULTIPLE' OPTION
53 ;
54 006370 004767 0000000 SPLMUL: CALL CKPRIV ; Must have operator privilege
55 006374 042761 0000000 0000000 BIC #SD$SNG,SDFLAG(R1); RESET SINGLE-FILE MODE
56 006402 000167 0000000 JMP RDCMD
57 ;

```

SPOOL command

```

58      ; PROCESS THE SPOOL 'HOLD' COMMAND.
59      ; HOLD MEANS DON'T START OUTPUT UNTIL CHANNEL IS CLOSED.
60      ;
61 006406 004767 0000000 SPLHLD: CALL CKPRIV      ;Must have operator privilege
62 006412 010167 171434      MOV R1,SPHLEM+4      ;Set SDCB address in EMT arg block
63 006416 012700 000046'      MOV #SPHLEM,R0      ;Point to EMT arg block
64 006422 104375      EMT 375      ;Set hold mode
65 006424 000167 0000000      JMP RDCMD
66      ;
67      ; PROCESS THE SPOOL 'NOHOLD' COMMAND WHICH MEANS START OUTPUT
68      ; AS SOON AS A FILE IS CREATED.
69      ;
70 006430 004767 0000000 SPLNHL: CALL CKPRIV      ;Must have operator privilege
71 006434 010167 171420      MOV R1,SPNHEM+4      ;Set SDCB address in EMT arg block
72 006440 012700 000054'      MOV #SPNHEM,R0      ;Point to EMT arg block
73 006444 104375      EMT 375      ;Set NOHOLD mode
74 006446 000167 0000000      JMP RDCMD
75      ;
76      ;-----
77      ; PROCESS THE 'FORM' COMMAND.
78      ;
79 006452 004767 0000000 CMDFRM: CALL CVTTAB      ;CONVERT TAB AND FF CHARS TO SPACE
80 006456 004767 0000000      CALL CVTUC      ;CONVERT ALL CHARS TO UPPER CASE
81 006462 012704 0000000      MOV #UFORM,R4      ;POINT TO CELL WHICH HOLDS FORM NAME
82 006466 020302      2$: CMP R3,R2      ;MOVED ALL OF NAME?
83 006470 103004      BHS 1$      ;BR IF YES
84 006472 112324      MOVB (R3)+,(R4)+      ;MOVE FORM NAME TO UFORM
85 006474 020427 0000060      CMP R4,#<UFORM+6>      ;MOVED 6 CHARS?
86 006500 103772      BLD 2$      ;BR IF NOT
87      ; SEE IF WE NEED TO PAD NAME WITH BLANKS
88 006502 020427 0000060      1$: CMP R4,#<UFORM+6>      ;FILLED TO 6 CHARS?
89 006506 103003      BHS 3$      ;BR IF FINISHED
90 006510 112724 000040      MOVB #' ,(R4)+      ;PUT IN TRAILING BLANKS
91 006514 000772      BR 1$
92 006516 000167 0000000      3$: JMP RDCMD
93      ;
94      ;-----
95      ; SPOOLGO is called to start a spooler.
96      ;
97      ; Inputs:
98      ; R0 = SDCB address of spooler to be started.
99      ;
100 006522 010046      SPOOLGO: MOV R0,-(SP)
101 006524 010067 0000040      MOV R0,SPGEMT+4      ;SET SDCB ADDRESS IN EMT ARG BLOCK
102 006530 012700 0000000      MOV #SPGEMT,R0      ;POINT TO ARG BLOCK
103 006534 104375      EMT 375      ;START THE SPOOLER
104 006536 012600      MOV (SP)+,R0
105 006540 000207      RETURN

```

SEND command

```

1          .SBTTL  SEND command
2          ;-----
3          ; Process the YELL command which overrides TT gag but requires
4          ; operator privilege.
5          ;
6 006542  004767  0000000  CMDYEL: CALL    CKPRIV      ;Require OPER privilege
7 006546  112767  000003  0000000  MOVV    #3,YELEM  ;Set the YELL and error flags
8 006554  000411                BR      SNDCOM    ;Enter SEND command code
9
10         ;-----
11        ; Process the OPERATOR command which functions as a send
12        ; command to the operators console.
13        ;
14 006556  116701  0000000  OPRCMD: MOVV    CTRLTT,R1  ;GET LINE INDEX # OF OPR TERMINAL
15 006562  001014                BNE     SNDCN      ;BR IF GOT ONE
16 006564                FABORT  #NOOPTT  ;NO OPERATOR'S CONSOLE
17
18        ;-----
19        ; PROCESS THE 'SEND' COMMAND.
20        ;
21 006574  105067  0000000  CMDSND: CLRB   YELEM      ;This is not the YELL command
22 006600  111300                SNDCOM: MOVV    (R3),R0    ;GET 1ST CHAR OF COMMAND
23 006602  001420                BEQ     CSEXIT     ;BRANCH IF NULL COMMAND
24
25        ; ACCRUE LINE #.
26        ;
27 006604  004767  000544                CALL    SNDLIN      ;Accrue line # to send to
28 006610  005701                TST    R1          ;SEND TO SPECIFIC USER?
29 006612  003422                BLE    CSSCAN      ;BRANCH IF ALL OR OC
30 006614  020127  0000000  SNDCN:  CMP    R1,#LSTSL  ;IS THIS A VALID LINE #?
31 006620  101003                BHI    LNUMER      ;BR IF NOT
32 006622  016101  0000000  MOV     LNPRIM(R1),R1  ;GET PRIMARY LINE #
33 006626  001010                BNE    CSCD        ;BRANCH IF VALID
34 006630                LNUMER: FWARN  #BDLIN   ;Invalid line number
35 006644  000167  0000000  CSEXIT: JMP    RDCMD    ;
36 006650  032761  0000000  0000000  CSCD:  BIT    #DETCH,LSW(R1) ;TRYING TO SEND TO DETACHED JOB?
37 006656  001364                BNE    LNUMER      ;BR IF YES
38
39        ; AT THIS POINT THE INDEX NUMBER OF THE LINE WE WISH TO
40        ; SEND TO IS IN R1.
41        ; PUT OUR LINE NUMBER AND USER NAME AT START OF MESSAGE BUFFER
42        ;
43 006660  032767  0000000  0000000  CSSCAN: BIT    #PO$SND,PRIVCO ;Are we privileged to send messages?
44 006666  001004                BNE    10$        ;Br if yes
45 006670                FABORT  #EM$SND
46 006700  012704  0000000  10$:   MOV     #MSGBUF,R4  ;POINT TO MESSAGE BUFFER
47 006704  112724  000015                MOVV   #CR,(R4)+      ;PUT IN LEADING CR-LF-BELL
48 006710  112724  000012                MOVV   #LF,(R4)+
49 006714  112724  000007                MOVV   #BELL,(R4)+
50 006720  116705  0000000                MOVV   CORUSR,R5      ;GET OUR LINE NUMBER
51 006724  016505  0000000                MOV    LNPRIM(R5),R5  ;GET OUR PRIMARY LINE NUMBER
52 006730  010502                MOV    R5,R2
53 006732  062702  0000000                ADD    #NUMTB1,R2    ;POINT TO TABLE WITH LINE NUMBER CHARACTERS
54 006736  112224                MOVV   (R2)+,(R4)+    ;PUT IN LINE NUMBER
55 006740  112224                MOVV   (R2)+,(R4)+
56 006742  070527  000006                MUL    #6,R5         ;EACH JOB HAS A 12 CHAR USER NAME
57 006746  062705  0000000                ADD    #LUNAME,R5    ;POINT TO USER NAME ENTRY FOR THIS LINE

```

SEND command

```

58 006752 121527 000040          CMPB    (R5),#'          ; IS USER NAME BLANK?
59 006756 001416                BEQ     4$              ; BR IF YES
60 006760 112724 000040          MOVVB  #' ,(R4)+        ; PUT IN A SPACE
61 006764 112724 000050          MOVVB  #'( ,(R4)+      ; AND OPEN PAREN AT START OF NAME
62 006770 012702 000014          MOV    #12 ,R2        ; GET LENGTH OF NAME
63 006774 112524                6$:    MOVVB  (R5)+,(R4)+  ; GET NEXT CHAR OF USER NAME
64 006776 077202                SOB     R2,6$         ; MOVE REST OF NAME
65 007000 124427 000040          7$:    CMPB   -(R4),#BLANK ; TRIM OFF TRAILING SPACES
66 007004 001775                BEQ     7$
67 007006 005204                INC     R4            ; POINT BEYOND LAST NON-BLANK CHARACTER
68 007010 112724 000051          5$:    MOVVB  #' ),(R4)+  ; PUT IN CLOSE PAREN AT END OF NAME
69 007014 112724 000040          4$:    MOVVB  #' , (R4)+  ; PUT " -- " AT END OF NAME
70 007020 112724 000055          MOVVB  #'-, (R4)+
71 007024 112724 000055          MOVVB  #'-, (R4)+
72 007030 112724 000040          MOVVB  #' , (R4)+
73                                ;
74                                ; Now move message string to message buffer
75                                ;
76 007034 122327 000040          3$:    CMPB   (R3)+, #'   ; SKIP OVER ANY BLANKS
77 007040 001775                BEQ     3$
78 007042 005303                DEC     R3            ; POINT TO 1ST CHAR OF MESSAGE
79 007044 112324                1$:    MOVVB  (R3)+,(R4)+  ; MOVE MESSAGE TO OUR BUFFER
80 007046 001403                BEQ     2$            ; BRANCH WHEN END HIT
81 007050 020427 177776        CMP     R4,#CMSGEND-2$ ; DON'T OVERFLOW OUR BUFFER
82 007054 103773                BLO    1$
83 007056 005304                2$:    DEC     R4            ; PUT CR-LF AT END OF MESSAGE
84 007060 112724 000015          MOVVB  #CR,(R4)+
85 007064 112724 000012          MOVVB  #LF,(R4)+
86 007070 105014                CLR    (R4)          ; PUT IN NULL TO SIGNAL END
87                                ;
88                                ; See who we should send the message to.
89                                ;
90                                TST     R1            ; WHO IS IT GOING TO?
91 007074 001655                BEQ     LNUMER        ; SEND TO CONSOLE TERMINAL
92 007076 100450                BMI    SNDALL        ; SEND TO ALL LINES
93                                ;
94                                ; Send message to one terminal whose line index number
95                                ; is in R1.
96                                ;
97 007100 032761 000000 000000  SNDONE: BIT    #$DILUP,LSW(R1) ; IS THAT LINE LOGGED ON?
98 007106 001005                BNE    1$            ; BRANCH IF YES
99 007110                .PRINT #NOTON
100 007116 000167 000000        JMP    RDCMD
101 007122 032761 000000 000000  1$:    BIT    #$TTGAG,LSW7(R1) ; IS LINE GAGGED?
102 007130 001414                BEQ     2$            ; BR IF NOT
103 007132 032761 000000 000000  BIT    #$INKMN,LSW4(R1) ; IS JOB IN KMDN?
104 007140 001010                BNE    2$            ; BR IF YES
105 007142 105767 000000        TSTB   YELEMT        ; Are we yelling?
106 007146 001005                BNE    2$            ; Br if yes -- Override gag
107 007150                .PRINT #GAGMSG      ; SAY LINE IS GAGGED
108 007156 000167 000000        JMP    RDCMD
109 007162 012700 000000        2$:    MOV    #YELEMT,R0  ; POINT TO EMT ARG BLOCK
110 007166 006201                ASR    R1            ; GET LINE NUMBER * 1
111 007170 010160 000002        MOV    R1,2(R0)     ; SET LINE # IN ARG BLOCK
112 007174 104375                EMT    375          ; SEND THE MESSAGE
113 007176 103006                BCC    3$            ; BR IF NO ERROR
114 007200                FWARN  #NMSGDF

```

SEND command

```

115 007214 000167 0000000 3$: JMP RDCMD
116 ;
117 ; Send message to all terminals.
118 ;
119 007220 012703 0000000 SNDALL: MOV #LSTPL,R3 ;GET INDEX # OF LAST LINE
120 007224 116702 0000000 MOVB CORUSR,R2 ;GET OUR LINE INDEX #
121 007230 016202 0000000 MOV LNPRIM(R2),R2 ;GET OUR PRIMARY LINE INDEX #
122 007234 032763 0000000 0000000 2$: BIT #DILUP,LSW(R3) ;IS THIS LINE LOGGED ON?
123 007242 001437 BEQ 1$ ;BRANCH IF NOT
124 007244 032763 0000000 0000000 BIT #DETCH,LSW(R3) ;IS THIS A DETACH LINE?
125 007252 001033 BNE 1$ ;BR - DETACH LINES DON'T NEED MESSAGES
126 007254 020302 CMP R3,R2 ;DON'T SEND MESSAGE TO OURSELF
127 007256 001431 BEQ 1$
128 007260 032763 0000000 0000000 BIT #TTGAG,LSW(R3) ;IS LINE GAGGED?
129 007266 001407 BEQ 3$ ;BR IF NOT
130 007270 105767 0000000 TSTB YELEMT ;Are we YELLing?
131 007274 001004 BNE 3$ ;Br if yes -- Override gag
132 007276 032763 0000000 0000000 BIT #INKMN,LSW(R3) ;IS JOB IN KMON NOW?
133 007304 001416 BEQ 1$ ;IF NOT THEN DON'T SEND MESSAGE
134 007306 012700 0000000 3$: MOV #YELEMT,R0 ;POINT TO EMT ARG BLOCK
135 007312 010301 MOV R3,R1 ;GET LINE INDEX #
136 007314 006201 ASR R1 ;GET LINE NUMBER
137 007316 010160 0000002 MOV R1,2(R0) ;STORE LINE # IN EMT ARG BLOCK
138 007322 104375 EMT 375 ;SEND THE MESSAGE
139 007324 103006 BCC 1$ ;BR IF NO ERROR
140 007326 FWARN #NMSGBF ;REPORT NO MESSAGE BUFFERS
141 007342 162703 0000002 1$: SUB #2,R3 ;MOVE ON TO NEXT LINE
142 007346 001332 BNE 2$ ;BRANCH IF MORE TO DO
143 007350 000167 0000000 JMP RDCMD

```

SEND command

```

1          ; -----
2          ;   Accrue the line number associated with a SEND command
3          ;
4          ;   Inputs:
5          ;   R3 = Points past end of SEND keyword.
6          ;
7          ;   Outputs:
8          ;   R1 = Index number of job to send to (-1 ==> Send to all jobs)
9          ;
10         007354  010546  SNDLIN: MOV      R5,-(SP)
11         ;
12         ;   See if a line number was specified
13         ;
14         007356  004767  0000006  CALL      SKPSPC      ;Skip over any spaces
15         007362  121327  000054   CMPB     (R3),#'     ;Start of line # field?
16         007366  001407          BEQ      1$          ;Br if yes
17         007370  121327  000057   CMPB     (R3),#'/    ;Start of line # field?
18         007374  001404          BEQ      1$          ;Br if yes
19         007376          FABORT  #EM$NLN   ;No line number was specified
20         ;
21         ;   See if we are sending to ALL, OPERATOR, or PARENT
22         ;
23         007406  005203  1$:      INC      R3          ;Point past delimiter
24         007410  004767  0000006  CALL      SKPSPC      ;Scan up to line number
25         007414  111300          MOVVB    (R3),R0      ;Get 1st char of "number"
26         007416  020027  000141   CMP      R0,#141     ;Is this a lower-case letter?
27         007422  103402          BLD      2$          ;Br if not
28         007424  162700  000040   SUB      #40,R0      ;Convert lower-case to upper case
29         007430  120027  000101  2$:      CMPB     R0,#'A    ;Send to ALL?
30         007434  001003          BNE      3$          ;Br if not
31         007436  012701  177777   MOV      #-1,R1      ;Line # for All
32         007442  000424          BR       7$          ;
33         007444  120027  000117  3$:      CMPB     R0,#'O    ;Send to operator?
34         007450  001007          BNE      4$          ;Br if not
35         007452  116701  0000006  MOVVB    CTRLTT,R1   ;Get operator line #
36         007456  001016          BNE      7$          ;
37         007460          FABORT  #NOOPTT   ;No operator terminal
38         007470  120027  000120  4$:      CMPB     R0,#'P    ;Send to parent job?
39         007474  001015          BNE      5$          ;Br if not
40         007476  116701  0000006  MOVVB    CORUSR,R1   ;Get current job index #
41         007502  005761  0000006  TST      LPARNT(R1)  ;Is there a parent job?
42         007506  001402          BEQ      7$          ;Br if yes
43         007510  016101  0000006  MOV      LPARNT(R1),R1 ;Get # of parent job
44         007514  112300  7$:      MOVVB    (R3)+,R0    ;Skip over rest of keyword
45         007516  001407          BEQ      9$          ;
46         007520  120027  000040   CMPB     R0,#'      ;Space?
47         007524  001373          BNE      7$          ;
48         007526  000403          BR       9$          ;
49         ;
50         ;   We must have an ordinary line number
51         ;
52         007530  004767  0000006  5$:      CALL      ACRDEC      ;Accrue line number
53         007534  006301          ASL      R1          ;Convert to line index number
54         ;
55         ;   Finished
56         ;
57         007536  012605  9$:      MOV      (SP)+,R5

```

58 007540 000207

RETURN

ACCESS command

```

1          . SBTTL  ACCESS command
2          ;-----
3          ;  PROCESS THE ACCESS COMMAND
4          ;
5 007542  032761  0000000 0000000 CMDACC: BIT    ##$SUCF,LSW9(R1) ; ARE WE INSIDE INITIAL COMMAND FILE?
6 007550  001010                BNE     15$          ; BR IF YES
7 007552  032767  0000000 0000000 BIT     #P0$SYS,PRIVCO ; Does user have SYSPRV privilege?
8 007560  001004                BNE     15$          ; Br if yes
9 007562                FABORT  #EM$NSF          ; ONLY LEGAL IN START-UP COMMAND FILES
10 007572  004767  0000000          15$: CALL   CVTTAB          ; CONVERT TAB AND FF CHARS TO SPACES
11 007576  012704  0000000          MOV    #OKFILE,R4      ; POINT TO TABLE OF FILE NAMES
12 007602  105713                3$: TSTB   (R3)          ; REACHED END OF COMMAND LINE?
13 007604  001002                BNE     1$          ; BR IF NOT
14 007606  000167  0000000          JMP    RDCMD          ; FINISHED COMMAND
15 007612  020427  0000000          1$: CMP    R4,#OKFEND    ; TABLE OVERFLOW?
16 007616  103404                BLO    16$          ; BR IF NOT
17 007620                FABORT  #TBLOVF        ; TABLE OVERFLOW
18 007630  005764  0000000          16$: TST   OF$FIL(R4)   ; SEARCH FOR FREE SLOT
19 007634  001403                BEQ    4$          ; BR IF FOUND ONE
20 007636  062704  0000000          ADD    #OF$$SZ,R4    ; CHECK NEXT SLOT
21 007642  000763                BR     1$
22          ;
23          ;  Initialize table entry
24          ;
25 007644  005267  0000000          4$: INC    RESDEV      ; ANOTHER AUTHORIZED DEVICE-FILE
26 007650  012702  132500          MOV    #WLDNAM,R2    ; GET WILD-CARD FILE SPEC VALUE
27 007654  010264  0000000          MOV    R2,OF$FIL(R4) ; SET FILE NAME TO WILD CARD
28 007660  010264  0000020          MOV    R2,OF$FIL+2(R4)
29 007664  010264  0000040          MOV    R2,OF$FIL+4(R4)
30 007670  112764  177777  0000000 MOVVB  #-1,OF$DEV(R4) ; WILDCARD THE DEVICE NAME TOO
31 007676  112764  177777  0000000 MOVVB  #-1,OF$UNT(R4)
32          ;
33          ;  Accrue device and file name.
34          ;
35 007704  004767  0000000          CALL   GTRD50        ; SCAN FIRST NAME
36 007710  121327  000072          CMPB   (R3),#':      ; IS COLON THE TERMINATING CHARACTER?
37 007714  001036                BNE     5$          ; BR IF NOT
38          ;
39          ;  We just accrued the device name.
40          ;  See if this is a logical name that needs to be converted to physical.
41          ;
42 007716  016700  0000000          MOV    R50BUF,R0     ; Get specified device name
43 007722  020027  132500          CMP    R0,#WLDNAM    ; Wildcard?
44 007726  001417                BEQ    6$          ; Br if wildcard device name
45 007730  004767  0000000          CALL   ASNSRC        ; See if name is in assign table
46 007734  103403                BCS    8$          ; Br if name not in assign table
47 007736  016267  0000000 0000000 MOV    AT$DEV(R2),R50BUF; Replace logical dev name with physical name
48          ;
49          ;  Convert the name to a device table index number.
50          ;
51 007744  016701  0000000          8$: MOV    R50BUF,R1    ; GET THE RAD50 DEVICE NAME
52 007750  004767  000270          CALL   CVTDEV        ; CONVERT DEVICE NAME TO INDEX NUMBER
53 007754  103473                BCS    ACBDFS        ; BR IF BAD FILE NAME
54 007756  110164  0000000          MOVVB  R1,OF$DEV(R4) ; SET DEVICE INDEX NUMBER
55 007762  110264  0000000          MOVVB  R2,OF$UNT(R4) ; SET UNIT NUMBER
56          ;
57          ;  Get file name.

```

ACCESS command

```

58 ;
59 007766 005203 6$: INC R3 ;SKIP PAST COLON
60 007770 111300 MOVB (R3),R0 ;GET FIRST CHAR OF FILE NAME
61 007772 120027 000052 CMPB R0,#'*' ;WILD CARD?
62 007776 001403 BEQ 7$ ;BR IF YES
63 010000 004767 0000000 CALL CHKDLM ;1ST CHAR OF NAME SHOULDN'T BE DELIMITER
64 010004 103426 BCS 10$ ;BR IF IT IS DELIMITER
65 010006 004767 0000000 7$: CALL GTRD50 ;ACCRUE FILE NAME
66 010012 016700 0000000 5$: MOV R50BUF,R0 ;GET RAD50 FILE NAME
67 010016 001452 BEQ ACBDFS ;BR IF NULL NAME
68 010020 020027 132500 CMP R0,#WLDNAM ;WILDCARD NAME?
69 010024 001403 BEQ 9$ ;BR IF YES
70 010026 010064 0000000 MOV R0,OF$FIL(R4) ;SET FILE NAME IN FILE SPEC
71 010032 016764 0000020 0000020 MOV R50BUF+2,OF$FIL+2(R4);SET 2ND HALF OF NAME
72 ; See if extension was specified.
73 010040 121327 000056 9$: CMPB (R3),#'.' ;WAS FILE EXTENSION SPECIFIED?
74 010044 001006 BNE 10$ ;BR IF NOT
75 010046 005203 INC R3 ;SKIP PAST PERIOD
76 010050 004767 0000000 CALL GTRD50 ;ACCRUE THE EXTENSION
77 010054 016764 0000000 0000040 MOV R50BUF,OF$FIL+4(R4);SET EXTENSION
78 ;
79 ; See if any switch was specified with file spec.
80 ;
81 010062 121327 000057 10$: CMPB (R3),#'/' ;WAS A SWITCH SPECIFIED?
82 010066 001020 BNE 11$ ;BR IF NOT
83 010070 116300 000001 MOVB 1(R3),R0 ;GET 1ST CHAR OF SWITCH NAME
84 010074 120027 000122 CMPB R0,#'R' ;"READ" ONLY?
85 010100 001403 BEQ 13$ ;BR IF YES
86 010102 120027 000162 CMPB R0,#'r' ;CHECK LOWER-CASE SPELLING
87 010106 001052 BNE ACBDSW ;BR IF INVALID SWITCH
88 010110 152764 0000000 0000000 13$: BISB #0T$RON,OF$FLG(R4);SET READ-ONLY FLAG IN FILE SPEC
89 010116 005203 12$: INC R3 ;SKIP OVER /
90 010120 111300 MOVB (R3),R0 ;GET NEXT CHAR OF SWITCH NAME
91 010122 004767 0000000 CALL CHKDLM ;SKIP UP TO NEXT DELIMITER
92 010126 103373 BCC 12$
93 ;
94 ; Finished getting a file spec.
95 ;
96 010130 062704 0000000 11$: ADD #OF$#SZ,R4 ;POINT TO NEXT SLOT IN ACCESS TABLE
97 010134 004767 000236 CALL SKPD2 ;SKIP OVER TERMINATING DELIMITER
98 010140 000167 177436 JMP 3$ ;GO GET NEXT FILE SPEC IF ANY
99 ;
100 ; Error -- Invalid file spec.
101 ;
102 010144 ACBDFS: FWARN #CSIMS2 ;Invalid device
103 010160 005367 0000000 DEC RESDEV ;Remove this entry from the access table
104 010164 005064 0000000 CLR OF$FIL(R4) ;Say entry is unused
105 010170 105064 0000000 CLRB OF$DEV(R4)
106 010174 105064 0000000 CLRB OF$UNT(R4)
107 010200 112300 1$: MOVB (R3)+,R0 ;Get next character
108 010202 001412 BEQ 2$ ;Br if hit end of command
109 010204 120027 000040 CMPB R0,#' ' ;Is this character a space?
110 010210 001403 BEQ 3$ ;Br if yes
111 010212 120027 000054 CMPB R0,#', ' ;Is this character a comma?
112 010216 001370 BNE 1$ ;Loop if not
113 010220 116701 0000000 3$: MOVB CORUSR,R1 ;Get back job index number
114 010224 000167 177312 JMP CMDACC ;Go back and process the rest of the command

```

ACCESS command

115 010230 000167 0000000
116
117
118
119 010234

2#: JMP RDCMD
;
; Error -- Bad switch
;
ACBDSW: FABORT #INVOP

ACCESS command

```

1          ; -----
2          ; CVTDEV is called to convert a RAD50 device name to the corresponding
3          ; device table index number and unit number.
4          ;
5          ; Inputs:
6          ; R1 = RAD50 device name.
7          ;
8          ; Outputs:
9          ; R1 = Device table index number for device.
10         ; R2 = Unit number.
11         ; C-flag set on return if invalid device name.
12         ;
13 010244  CVTDEV:
14         ;
15         ; Get device name and split off unit number.
16         ;
17 010244 005000      CLR      R0          ; SET FOR DIVIDE
18 010246 071027 000050  DIV      #50,R0      ; SPLIT OFF LOW-ORDER RAD50 CHARACTER
19 010252 012702 177777  MOV      #-1,R2      ; ASSUME NO UNIT NUMBER SPECIFIED
20 010256 005701      TST      R1          ; WAS A UNIT NUMBER SPECIFIED?
21 010260 001406      BEQ      6$          ; BR IF NOT
22 010262 162701 000036  SUB      #36,R1      ; CONVERT RAD50 DIGIT TO BINARY VALUE
23 010266 010102      MOV      R1,R2      ; PUT UNIT NUMBER IN R2
24 010270 020227 000011  CMP      R2,#9      ; MAKE SURE UNIT NUMBER IN RANGE 0-9.
25 010274 101027      BHI      8$          ; BR IF INVALID UNIT NUMBER
26 010276 070027 000050  6$:     MUL      #50,R0      ; GET DEVICE NAME WITHOUT UNIT #
27         ;
28         ; The RAD50 device name less unit number is now in R1.
29         ; The unit number is in R2.
30         ;
31         ; Translate SY and DK.
32         ;
33 010302 020167 000000G  CMP      R1,R50SY    ; IS DEVICE NAME SY?
34 010306 001403      BEQ      2$          ; BR IF YES
35 010310 020167 000000G  CMP      R1,R50DK    ; IS DEVICE NAME DK?
36 010314 001007      BNE      3$          ; BR IF NOT
37 010316 016701 000000G  2$:     MOV      SYINDX,R1 ; GET SY DEVICE INDEX NUMBER
38 010322 005702      TST      R2          ; WAS A UNIT NUMBER SPECIFIED?
39 010324 002016      BGE      4$          ; BR IF YES
40 010326 116702 000001G  MOVB    SYUNIT+1,R2 ; GET SY UNIT NUMBER
41 010332 000413      BR       4$          ;
42         ;
43         ; Look up device name in permanent device name table.
44         ;
45 010334 016700 000000G  3$:     MOV      NUMDEV,R0 ; GET INDEX OF LAST PERM NAME
46 010340 020160 000000G  5$:     CMP      R1,PNAME(R0) ; LOOK UP DEVICE NAME
47 010344 001405      BEQ      7$          ; BR IF FOUND
48 010346 162700 000002  SUB      #2,R0      ; CHECK NEXT ENTRY
49 010352 002372      BGE      5$          ;
50         ;
51         ; Invalid device name
52         ;
53 010354 000261  8$:     SEC          ; SIGNAL ERROR ON RETURN
54 010356 000405      BR       9$          ;
55         ;
56         ; Found device name.
57         ;

```

ACCESS command

```

58 010360 010001      7#:      MOV      R0,R1      ;GET DEVICE NAME TABLE INDEX
59 010362 005702      4#:      TST      R2              ;WAS A UNIT NUMBER SPECIFIED?
60 010364 002001              BGE      10#              ;BR IF YES
61 010366 005002              CLR      R2              ;CONVERT NO UNIT NUMBER TO # 0
62 010370 000241      10#:     CLC              ;SIGNAL NO ERROR ON RETURN
63
64                    ; Return
65
66 010372 000207      9#:      RETURN
67
68                    ; -----
69                    ; SUBROUTINE TO SKIP OVER COMMAS AND SPACES
70 010374 005203      SKPD1:   INC      R3              ;POINT TO NEXT CHAR
71 010376 121327 000040      SKPD2:   CMPB    (R3),#'      ;IS IT A SPACE
72 010402 001774              BEQ      SKPD1              ;KEEP SKIPPING IF YES
73 010404 121327 000054      CMPB    (R3),#','          ;IS IT A COMMA
74 010410 001771              BEQ      SKPD1              ;KEEP SKIPPING IF YES
75 010412 000207              RETURN

```

DETACH command

```

1          . SBTTL  DETACH command
2          ; -----
3          ; PROCESS THE DETACH COMMAND
4          ;
5 010414  004767  0000000  CMDDET: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
6 010420  032767  0000000 0000000 BIT    #PO$DET,PRIVCO ; IS USER ALLOWED TO USE DETACHED JOBS?
7 010426  001004          BNE     1#      ; BR IF YES
8 010430          FABORT  #EM$DET      ; NOT ALLOWED
9          ;
10         ; SEE IF ANY SWITCHES WERE SPECIFIED WITH COMMAND
11         ;
12 010440  121327  000057  1#:     CMPB   (R3),#'/      ; ANY SWITCHES?
13 010444  001062          BNE     DETGO      ; BR IF NOT -- START DETACHED JOB
14         ; ACCRUE AND IDENTIFY SWITCH
15 010446  005203          INC     R3      ; SKIP OVER /
16 010450  012704  000220'  MOV    #DETHD,R4      ; POINT TO TABLE OF SWITCHES
17 010454  004767  0000000  CALL  SEARCH          ; LOOK UP THE SWITCH
18 010460  103004          BCC    2#      ; BR IF FOUND OK
19 010462          FABORT  #INVOPT      ; INVALID SWITCH
20         ; ACCRUE LINE NUMBER AND SEE IF LEGAL
21 010472  004767  0000000  2#:   CALL  ACRDEC      ; ACCRUE THE DECIMAL NUMBER
22 010476  006301          ASL    R1      ; CONVERT TO LINE INDEX #
23 010500  020127  0000000  CMP    R1,#FSTDL      ; SEE IF IT IS A DETACHED LINE
24 010504  103004          BHIS   3#      ;
25 010506          4#:   FABORT  #BDLIN      ;
26 010516  020127  0000000  3#:   CMP    R1,#LSTDL      ;
27 010522  101371          BHI    4#      ;
28 010524  012700  0000000  MOV    #DETARG,R0      ; POINT TO DETACH EMT ARG BLOCK
29 010530  006201          ASR    R1      ; STORE DETACHED JOB #
30 010532  010167  0000020  MOV    R1,DETARG+2      ;
31         ; BRANCH OFF TO PROCESSING ROUTINE
32 010536  000134          JMP    @(R4)+      ; ENTER PROCESSING ROUTINE
33         ;
34         ; Process the /CHECK option.
35         ;
36 010540  112710  000001  DETCHK: MOVB  #1,(R0)      ; SET SUB-FUNCTION CODE FOR EMT
37 010544  104375          EMT    375      ; CHECK ON DETACHED JOB
38 010546  103404          BCS   DETIDL      ; BR IF JOB FINISHED
39 010550          .PRINT #RUNMS      ; JOB STILL RUNNING
40 010556  000403          BR    DETJMP      ;
41 010560          DETIDL: .PRINT #LINFRE      ; LINE IS IDLE
42 010566  000167  0000000  DETJMP: JMP   RDCMD      ; FINISHED WITH COMMAND
43         ;
44         ; Process the /KILL option.
45         ;
46 010572  112710  000002  DETKIL: MOVB  #2,(R0)      ; SET SUB-FUNCTION CODE FOR EMT
47 010576  104375          EMT    375      ; KILL A DETACHED JOB
48 010600  103767          BCS   DETIDL      ; BR IF JOB NOT RUNNING
49 010602          .PRINT #DJABMS      ; PRINT CONFIRMATION
50 010610  000766          BR    DETJMP      ; FINISHED

```

DETACH command

```

1      ;
2      ; Start a new detached job.
3      ; R3 = Pointer to ASCIZ string with detached job command file name.
4      ; Try to open the command file to make sure it exists.
5      ;
6 010612 010302 DETQD:  MOV     R3,R2           ;Save pointer to start of file name string
7 010614 012704 0000000  MOV     #R5OCOM,R4       ;Point to word with default extension (COM)
8 010620 005005      CLR     R5             ;Say this is an input file
9 010622 004767 0000000  CALL    ACRFIL          ;Accrue the file spec
10 010626 103441     BCS     10$            ;Br if invalid file spec
11 010630      .LOOKUP #XAREA,#1,#FILNAM ;Try to open the file
12 010650 103434     BCS     11$            ;Br if cannot open file
13 010652      .CLOSE  #1             ;Close the file
14      ;
15      ; Execute EMT to initiate the detached job.
16      ;
17 010660 012700 0000000  MOV     #DETARG,R0      ;POINT TO DETACH EMT ARG BLOCK
18 010664 105010     CLR    (R0)           ;SET SUB-FUNCTION CODE FOR EMT
19 010666 010260 0000002  MOV     R2,2(R0)       ;SET ADDRESS OF COMMAND FILE NAME ASCIZ STRING
20 010672 104375     EMT     375           ;START A DETACHED JOB
21 010674 103412     BCS     1$             ;BR IF NO FREE LINES
22      ;
23      ; Tell user which line was used
24      ;
25 010676 010005     MOV     R0,R5           ;GET DETACHED JOB LINE NUMBER
26 010700      .PRINT  #DLMSG        ;PRINT HEADING
27 010706 004767 0000000  CALL    PRTDEC         ;DISPLAY LINE #
28 010712      .PRINT  #CRLF        ;TERMINATE PRINT LINE
29 010720 000722     BR      DETJMP
30      ;
31      ; No free lines
32      ;
33 010722 1$:      FABORT #NOFRDL      ;NO FREE LINES
34      ;
35      ; Invalid file spec
36      ;
37 010732 10$:     FABORT #BDFNAM
38      ;
39      ; Cannot open file
40      ;
41 010742 11$:     FABORT #EM$FUE

```

DATE command

```

1          . SBTTL  DATE command
2          ; -----
3          ; Process the DATE command.
4          ;
5 010752  004767  0000000  CMDDAT: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
6 010756  105713                TSTB   (R3)      ; DOES HE WANT TO SHOW OR CHANGE THE DATE?
7 010760  001002                BNE    3$         ; BR TO SET THE DATE
8 010762  000167  0000000  JMP    SOPDAT     ; GO DISPLAY THE DATE
9          ;
10         ; Set a new system date value (operator privilege required).
11         ;
12 010766  004767  0000000  3$:   CALL  CKPRIV     ; MAKE SURE THIS USER IS PRIVILEGED
13         ; Get day value.
14 010772  004767  0000000  CALL  ACRDEC     ; ACCRUE DAY VALUE
15 010776  005701                TST    R1        ; CHECK RANGE OF VALUE
16 011000  003453                BLE    BADDAT    ; CAN'T BE ZERO
17 011002  020127  000037  CMP    R1, #31.   ; OR GREATER THAN 31
18 011006  101050                BHI    BADDAT
19 011010  072127  000005  ASH    #5, R1     ; POSITION FOR DATE WORD
20 011014  010105                MOV    R1, R5     ; CONSTRUCT DATE WORD IN R5
21 011016  122327  000055  CMPB  (R3)+, #'-  ; HYPHEN SHOULD BE NEXT
22 011022  001042                BNE    BADDAT
23         ; Get month.
24 011024  004767  0000000  CALL  GTRD50     ; ACCRUE RAD50 MONTH VALUE
25 011030  016700  0000000  MOV    R50BUF, R0 ; GET RAD50 VALUE OF MONTH NAME
26 011034  012701  000001  MOV    #1, R1     ; FIRST MONTH IS # 1
27 011040  012702  0000000  MOV    #R50MON, R2 ; POINT TO TABLE OF MONTH NAMES
28 011044  020022  2$:   CMP    R0, (R2)+   ; LOOK UP NAME IN TABLE
29 011046  001405                BEQ    1$         ; BR IF FOUND
30 011050  005201                INC    R1         ; ADVANCE MONTH NUMBER
31 011052  020127  000014  CMP    R1, #12.   ; ONLY 12 MONTHS
32 011056  101772                BLOS  2$         ;
33 011060  000423                BR     BADDAT     ; BAD MONTH NAME
34 011062  072127  000012  1$:   ASH    #10, R1  ; POSITION MONTH VALUE
35 011066  050105                BIS    R1, R5     ; AND OR INTO DATE WORD
36 011070  122327  000055  CMPB  (R3)+, #'-  ; SHOULD HAVE ANOTHER HYPHEN
37 011074  001015                BNE    BADDAT
38         ; Get year
39 011076  004767  0000000  CALL  ACRDEC     ; GET YEAR VALUE
40 011102  020127  000143  CMP    R1, #99.   ; CHECK FOR REASONABLE UPPER LIMIT
41 011106  101010                BHI    BADDAT     ; BR IF TOO HIGH
42 011110  162701  000110  SUB    #72, R1    ; YEAR VALUE IS BIASED BY 72
43 011114  003405                BLE    BADDAT     ; BR IF TOO SMALL
44 011116  050105                BIS    R1, R5     ; FORM DATE WORD
45         ; Set system date value.
46 011120  010567  0000000  MOV    R5, SYSDAT ; SET NEW SYSTEM DATE VALUE
47 011124  000167  0000000  JMP    RDCMD     ; FINISHED WITH COMMAND
48         ; Invalid date value.
49 011130  152767  000010  0000000 BADDAT: BISB  #10, INDERR ; SAVE ERROR LEVEL = SEVERE FOR IND
50 011136                FABORT #INVDAT   ; INVALID DATE

```

TIME command

```

1          .SBTTL  TIME command
2          ;-----
3          ; Process the TIME command.
4          ;
5 011146  004767  0000000  CMDTIM: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
6 011152  105713          TSTB   (R3)      ; DOES HE WANT TO SHOW OR CHANGE THE TIME?
7 011154  001002          BNE    2$      ; BR TO SET THE TIME
8 011156  000167  0000000  JMP    SOPTIM     ; GO SHOW THE TIME
9          ;
10         ; Set new system time (requires operator privilege)
11         ;
12 011162  004767  0000000  2$:  CALL  CKPRIV     ; MAKE SURE USER HAS OPERATOR PRIVILEGE
13         ; Get hour value.
14 011166  004767  0000000  CALL  ACRDEC      ; ACCRUE HOUR VALUE
15 011172  020127  000030  CMP   R1,#24.    ; ONLY 24 HOUR IN A DAY
16 011176  103053          BHS   BADTIM
17 011200  010104          MOV   R1,R4
18 011202  070427  000074  MUL   #60.,R4    ; CONVERT TO # MINUTES
19 011206  122327  000072  CMPB  (R3)+,#':  ; COLON SHOULD BE THE DELIMITER
20 011212  001045          BNE   BADTIM
21         ; Get minute value.
22 011214  004767  0000000  CALL  ACRDEC      ; ACCRUE MINUTE VALUE
23 011220  020127  000074  CMP   R1,#60.    ; ONLY 60 MINUTES PER HOUR
24 011224  103040          BHS   BADTIM
25 011226  060105          ADD   R1,R5      ; COMBINE WITH HOUR VALUE
26 011230  005504          ADC   R4
27 011232  012700  000074  MOV   #60.,R0    ; CONVERT # MINUTES TO # SECONDS
28 011236  004767  0000000  CALL  MUL32
29         ; See if seconds are specified
30 011242  122327  000072  CMPB  (R3)+,#':  ; DELIMITER FOR SECONDS?
31 011246  001011          BNE   3$      ; NO, IGNORE SECONDS
32         ; Get second value
33 011250  004767  0000000  CALL  ACRDEC      ; ACCRUE SECOND VALUE
34 011254  020127  000074  CMP   R1,#60.    ; NO MORE THAN 60 SECONDS PER MINUTE
35 011260  103022          BHS   BADTIM
36 011262  060105          ADD   R1,R5      ; COMBINE WITH HOURS AND MINUTES
37 011264  005504          ADC   R4
38 011266  012700  000074  MOV   #60.,R0    ; 60 HZ CLOCK TICKS PER SECOND
39         ; Convert time to # clock ticks.
40 011272  032767  0000000 0000000 3$:  BIT   #CW$50H,CONFIO ; 50 OR 60 HZ CLOCK?
41 011300  001402          BEQ   1$      ; BR IF 60 HZ
42 011302  012700  000062  MOV   #50.,R0    ; SET FOR 50 HZ CLOCK
43 011306  004767  0000000  1$:  CALL  MUL32      ; CONVERT # SECONDS TO # CLOCK TICKS
44         ; Set new system time.
45 011312  010467  0000000  MOV   R4,SYTIMH  ; HIGH-ORDER TIME VALUE
46 011316  010567  0000000  MOV   R5,SYTIML  ; LOW-ORDER TIME VALUE
47 011322  000167  0000000  JMP   RDCMD      ; FINISHED
48         ; Bad time value entered
49 011326  152767  000010  0000000  BADTIM: BISR   #10,INDERR ; SAVE ERROR LEVEL = SEVERE FOR IND
50 011334          FABORT  #INVTIM ; INVALID TIME VALUE

```

RESET command

```

1          .SBTTL  RESET command
2          ;-----
3          ; The RESET command resets system usage statistics.
4          ; Operator privilege is required.
5          ;
6 011344  004767  0000000  CMDRST: CALL   CKSYPV           ;Require SYSPRV privilege
7          ; Reset data cache statistics
8 011350  005067  0000000          CLR   DCTRD           ;TOTAL NUMBER OF READS FROM SHARED FILES
9 011354  005067  0000000          CLR   DCCRD           ;NUMBER OF READS FOUND IN CACHE
10 011360  005067  0000000         CLR   DCTWR           ;TOTAL NUMBER OF WRITES TO SHARED FILES
11 011364  005067  0000000         CLR   DCCWR           ;NUMBER OF WRITES THAT UPDATE CACHE DATA
12          ; Reset 32-bit data cells
13 011370  012701  011412'         MOV   #RSTVEC,R1       ;POINT TO VECTOR OF ADDRESSES TO CLEAR
14 011374  012102          1$:  MOV   (R1)+,R2       ;GET ADDRESS OF A CELL TO ZERO
15 011376  001403          BEQ   2$              ;BR IF HIT END OF LIST
16 011400  005022          CLR   (R2)+           ;CLEAR HIGH-ORDER WORD
17 011402  005017          CLR   (R2)            ;CLEAR LOW-ORDER WORD
18 011404  000773          BR    1$              ;
19 011406  000167  0000000  2$:  JMP   RDCMD           ;FINISHED
20          ;
21          ; Vector of 32-bit cells to clear.
22          ;
23 011412  0000000  RSTVEC: .WORD  TMTOTH
24 011414  0000000          .WORD  TMUSRH
25 011416  0000000          .WORD  TMSWTH
26 011420  0000000          .WORD  TMIOH
27 011422  0000000          .WORD  TMSWPH
28 011424  0000000          .WORD  TMIOWH
29 011426  0000000          .WORD  TMIDLH
30 011430  0000000          .WORD  CASTRO
31 011432  0000000          .WORD  CASTBR
32 011434  0000000          .WORD  CASCBR
33 011436  0000000          .WORD  CASTWO
34 011440  0000000          .WORD  CASTBW
35 011442  0000000          .WORD  CASCUP
36 011444  0000000          .WORD  0

```

OFF command

```

1          .SBTTTL  OFF command
2          ;-----
3          ; LOG OFF.
4          ;
5 011446   116701   000000G  CMDOFF:  MOVB    CORUSR,R1      ;GET USER INDEX #
6          ;
7          ; See if any qualifiers were specified with command
8          ;
9 011452   105067   166357      CLRB    OFFNWF      ;Clear NOWARN flag
10 011456   012704   000250'     MOV     #OFFHD,R4    ;Point to parsing table
11 011462   004767   000000G     CALL   OPTLST      ;Process any qualifiers
12         ;
13         ; See if this is a primary process logging off with active subprocesses
14         ;
15 011466   020127   000000G     CMP     R1,#LSTPL   ;Is this a primary process?
16 011472   101024      BHI    14$         ;Br if not
17 011474   105767   166335      TSTB   OFFNWF      ;Was /NOWARN qualifier specified?
18 011500   001021      BNE    14$         ;Br if yes -- Don't worry about subprocesses
19 011502   032761   000000G 000000G  BIT    ##PRGLK,LSW5(R1);Did we exit a locked program?
20 011510   001015      BNE    14$         ;Br if yes -- Don't worry about subprocesses
21 011512   032761   000000G 000000G  BIT    ##NOIN,LSW3(R1);Are we allowed to have terminal input?
22 011520   001011      BNE    14$         ;Br if not
23 011522   032761   000000G 000000G  BIT    ##LOFCF,LSW9(R1);Have we already done a logAoff command file?
24 011530   001005      BNE    14$         ;Do logoff if so
25 011532   004767   001142      CALL   CKOFSP      ;See if we need to warn about subprocesses
26 011536   103002      BCC    14$         ;Br if we should logoff
27 011540   000167   000000G     JMP    RDCMD      ;Abort the logoff
28         ;
29         ; Do the logoff.
30         ;
31 011544   052761   000000G 000000G 14$:   BIS    ##NOIN,LSW3(R1);DON'T ALLOW ABORT OF LOGOFF
32 011552   005067   000000G     CLR    RESDEV     ;RELEASE ALL DEVICE/FILE ACCESS RESTRICTIONS
33         ;
34         ; Close all of user's files.
35         ;
36 011556   004767   000000G     CALL   PRGALL     ;PURGE ALL OF USER'S CHANNELS
37 011562   004767   000000G     CALL   INDABT     ;Abort IND and nested command files
38 011566   004767   000000G     CALL   ABRTCF     ;CLOSE THE CONTROL FILE
39 011572   052761   000000G 000000G  BIS    ##NOIN,LSW3(R1);CLEARED BY ABRTCF(POPCF), SET IT AGAIN
40         ;
41         ; Determine if we need to invoke a logoff command file
42         ;
43 011600   005767   000000G     TST    LOFSPC     ;Is there a logoff command file?
44 011604   001471      BEQ    3$         ;Br if not
45 011606   004767   000000G     CALL   PUSHCF     ;Prepare to open new command file
46 011612   112767   000001  000000G  MOVB   #1,SERFLG   ;Do .SERR
47 011620      .LOOKUP  #XAREA,#CFCHAN,#LOFSPC ;Try to open the command file
48 011640   112767   000000  000000G  MOVB   #0,SERFLG   ;Do .HERR but don't affect carry flag
49 011646   103013      BCC    10$        ;Br if open was ok
50 011650   004767   000000G 11$:   CALL   POPCF      ;Pop command file status
51 011654   005067   000000G     CLR    LOFSPC     ;No logoff command file
52 011660      FERR    #EM$OLD   ;Cannot open logoff command file
53 011674   000435      BR     3$         ;
54 011676      10$:   .READW  #XAREA,#CFCHAN,#CFBUF,#256,#0 ;Read 1st block of file
55 011734   103745      BCS    11$        ;Br if read error
56 011736   052761   000000G 000000G  BIS    ##CFOPN,LSW4(R1);Say CFCHAN is open
57 011744   052761   000000G 000000G  BIS    ##LOFCF,LSW9(R1);Say we are processing logoff command file

```

OFF command

```

58 011752 042761 0000000 0000000      BIC    ##DOOFF,LSW(R1) ; Say not doing logoff processing yet
59 011760 005067 0000000      CLR    LDFSPC          ; Say logoff command file has been started
60 011764 000167 0000000      JMP    RDCMD          ; Go process 1st command in command file
61                                     ;
62                                     ; Enable privilege during logoff processing
63                                     ;
64 011770 012702 0000000      3$:   MOV    #PRIVCO,R2   ; Point to current privilege word
65 011774 012703 0000000      MOV    #PRIVSO,R3     ; Point to set privileges
66 012000 012700 0000000      MOV    #PVNPW,R0      ; Get # privilege words
67 012004 012722 177777      12$:  MOV    #177777,(R2)+ ; Set all privilege flags
68 012010 012723 177777      MOV    #177777,(R3)+
69 012014 077005      SOB    R0,12$
70 012016 052767 0000000 0000000      BIS    #LF$WRT,LOGFLG ; ENABLE WRITES TO LOG FILE
71                                     ;
72                                     ; Dismount all devices that were mounted by this job.
73                                     ;
74 012024 004767 0000000      CALL   DMTALL         ; Dismount all mounted devices
75                                     ;
76                                     ; Print usage statistics for line
77                                     ;
78 012030 032761 0000000 0000000      BIT    #VNOTT,LSW(R1) ; IS THIS JOB CONNECTED TO A TERMINAL?
79 012036 001033      BNE    7$            ; BR IF NOT
80 012040 016100 0000000      MOV    LNPRIM(R1),R0  ; GET PRIMARY LINE #
81 012044 042760 0000000 0000000      BIC    ##CTRLS,LSW3(R0); MAKE SURE OUTPUT IS ENABLED
82 012052 020127 0000000      CMP    R1,#LSTDL     ; Is this a subprocess?
83 012056 101403      BLOS   13$          ; Br if not
84 012060 005760 0000000      TST   LWINDO(R0)     ; Does primary process have windowing on?
85 012064 001020      BNE    7$            ; Br if yes -- Don't display logoff time
86 012066      13$:  .PRINT #CRLF
87 012074 105767 0000000      TSTB  STPFLG        ; IS SYSTEM BEING SHUT DOWN?
88 012100 001403      BEQ    5$            ; BR IF NOT
89 012102      .PRINT #SHTMSG    ; PRINT SHUT-DOWN MESSAGE
90 012110 004767 0000000      5$:   CALL   PRTTIM      ; PRINT USAGE INFO
91 012114      .TTYOUT #LF       ; PUT OUT EXTRA LF
92 012124 000420      BR    TIMUP
93                                     ; IF PRTTIM DIDN'T DO CPU CALCULATION WE MUST DO IT HERE.
94 012126 016104 0000000      7$:   MOV    LCPUHI(R1),R4 ; GET HIGH-ORDER CPU TIME (CLOCK TICKS)
95 012132 016105 0000000      MOV    LCPULO(R1),R5  ; GET LOW-ORDER CPU TIME (CLOCK TICKS)
96 012136 016703 0000000      MOV    TK1SEC,R3     ; GET # CLOCK TICKS PER SECOND
97 012142 004767 0000000      CALL   DIVIDE        ; CONVERT TIME VALUE TO # SECONDS
98 012146 012700 000012      MOV    #10.,R0      ; CONVERT TIME VALUE TO 0.1 SECOND
99 012152 004767 0000000      CALL   MUL32         ; UNITS BY MULTIPLYING BY 10
100 012156 010467 0000000      MOV    R4,CPUAH     ; SAVE HIGH-ORDER PART (0.1 SEC)
101 012162 010567 0000000      MOV    R5,CPUAL     ; SAVE LOW-ORDER PART (0.1 SEC)
102                                     ;
103                                     ; If user logged on, then accumulate his connect time
104                                     ;
105 012166 016105 0000000      TIMUP: MOV    LPROJ(R1),R5 ; DO WE HAVE A PPN FOR USER?
106 012172 001535      BEQ    NOTIM        ; BR IF NOT -- DIDN'T LOG ON
107                                     ; DO A DEASSIGN
108 012174 012702 0000000      MOV    #ASNTBL,R2   ; CLEAR ALL OF HIS ASSIGNS
109 012200 005022      1$:   CLR    (R2)+
110 012202 020227 0000000      CMP    R2,#ASNEND
111 012206 103774      BLO   1$
112                                     ; Try to open accounting file
113 012210      .LOOKUP #XAREA,#1,#AUTHFN
114 012230 103004      BCC   2$            ; BR IF OPEN OK

```

```

115 012232          .PRINT #COAF          ;PRINT WARNING MESSAGE
116 012240 000512    BR          NOTIM
117                ; Search file for user's ppn entry
118 012242 005002    2$: CLR          R2          ;SET BLOCK # TO 0
119 012244 016103 0000000 MOV        LPRG(R1),R3      ;GET PROGRAMMER #
120 012250          6$: .READW #XAREA,#1,#BLKO,#256.,R2
121 012306 103464    BCS          3$          ;BR IF END OF FILE HIT
122 012310 004767 000544 CALL        CVTBUF          ;COMPLEMENT CONTENTS OF BUFFER
123 012314 012704 0000000 MOV        #BLKO,R4          ;SEARCH FOR RECORD IN BLOCK
124 012320 012700 0000000 MOV        #ARNRPB,R0        ;# RECORDS PER BLOCK
125 012324 020564 0000000 5$: CMP        R5,AR$PRJ(R4)    ;DO PROJECT #'S MATCH?
126 012330 001003    BNE          8$          ;BR IF NOT
127 012332 020364 0000000 CMP        R3,AR$PRG(R4)    ;HOW ABOUT PROGRAMMER NUMBERS
128 012336 001406    BEQ          4$          ;BR IF WE FOUND THE RECORD
129 012340 062704 0000000 8$: ADD        #AR#$SZ,R4      ;KEEP LOOKING IN BLOCK
130 012344 005300    DEC          R0          ;ANY MORE RECORDS IN THIS BLOCK?
131 012346 001366    BNE          5$          ;BR IF YES
132 012350 005202    INC          R2          ;GO READ NEXT BLOCK
133 012352 000736    BR          6$
134                ; Found record. Add in new connect time
135 012354 016700 0000000 4$: MOV        MINTIM,R0      ;GET CURRENT TIME
136 012360 166100 0000000 SUB        LCONTM(R1),R0    ;CALCULATE CONNECT TIME
137 012364 005200    INC          R0
138 012366 060064 0000000 7$: ADD        R0,AR$CON(R4)    ;ACCUMULATE CONNECT TIME
139                ; COUNT NUMBER OF SESSIONS
140 012372 005264 0000000 INC        AR$CNT(R4)      ;INC SESSION COUNTER
141                ; ACCUMULATE CPU TIME
142 012376 066764 0000000 0000000 ADD        CPUAL,AR$CPL(R4)  ;ADD IN LOW-ORDER CPU TIME
143 012404 005564 0000000 ADC        AR$CPH(R4)      ;ADD CARRY
144 012410 066764 0000000 0000000 ADD        CPUAH,AR$CPH(R4) ;ADD IN HIGH-ORDER PART
145                ; WRITE UPDATED ACCOUNTING RECORD BACK TO FILE
146 012416 004767 000436 CALL        CVTBUF          ;COMPLEMENT CONTENTS OF BUFFER
147 012422          .WRITW #XAREA,#1,#BLKO,#256.,R2
148 012460          3$: .CLOSE #1
149                ;
150                ; Close job's log file
151                ;
152 012466 004767 0000000 NOTIM: CALL    LOGCLS          ;CLOSE LOG FILE
153                ;
154                ; Broadcast message to any monitoring jobs telling them we are logging off
155                ;
156 012472 012700 0000000 MOV        #GENMON,R0      ;Point to emt argument block
157 012476 012760 0000000 000002 MOV        #JS$OFF,2(R0)    ;Set status code
158 012504 104375    EMT          375          ;Broadcast status message
159                ;
160                ; Enter TSX to finish log off.
161                ;
162 012506 105767 0000000 TSTB       STPFLG          ;ARE WE DOING A SYSTEM SHUTDOWN?
163 012512 001460    BEQ          1$          ;BR IF NOT
164 012514 003046    BGT          15$         ;Br if $STOP or BOOT specified
165                ;
166                ; Shutdown was specified.
167                ;
168 012516 052761 0000000 0000000 BIS        ##NOABT,LSW7(R1);Don't allow job abort
169 012524 120127 0000000 CMPB       R1,#LSTPL        ;Is this a detached job?
170 012530 103403    BLO          10$         ;Br if not detached
171 012532 120127 0000000 CMPB       R1,#LSTDLL       ;In the detached range?

```

OFF command

```

172 012536 101435          BLOS  15$          ;Yes, job is detached
173 012540 126727 0000000 0000001 10$:  CMPB  PVON,#1      ;Are we the last primary/virtual job?
174 012546 003031          BGT   15$          ;Br if more jobs exist
175                          ;
176                          ; Kill all jobs (only detached jobs should be active).
177                          ;
178 012550 012702 0000000          MOV   #LSTSL,R2     ;GET # OF LAST LINE
179 012554 120267 0000000 11$:  CMPB  R2,CORUSR     ;IS THIS OUR LINE?
180 012560 001405          BEQ   12$          ;BR IF YES
181 012562 012700 0000000          MOV   #KILEMT,RO    ;POINT TO KILL EMT ARG BLOCK
182 012566 010260 0000004          MOV   R2,4(R0)      ;SET # OF LINE TO KILL
183 012572 104375          EMT   375          ;KILL THE LINE
184 012574 162702 0000002 12$:  SUB   #2,R2        ;DO NEXT LINE
185 012600 001365          BNE   11$          ;
186                          ;
187                          ; Determine if the spooler is still active.
188                          ;
189 012602 004767 0000000 13$:  CALL  SPLACT        ;Is the spooler still active?
190 012606 103011          BCC   15$          ;Br if spooler is not active.
191 012610          .TWAIT #XAREA,#TIMSPL ;Wait 2 seconds -- check again
192 012630 000764          BR   13$          ;
193                          ;
194                          ; $STOP or BOOT issued.
195                          ;
196 012632 042761 0000000 0000000 15$:  BIC   #NOABT,LSW9(R1); Allow job abort (necessary for log off)
197 012640 126727 0000000 0000001          CMPB  TONON,#1      ;Are we the last job on system?
198 012646 003002          BGT   1$           ;Br if more jobs exist
199 012650 000167 0000000          JMP   DOSTOP        ;GO STOP THE SYSTEM
200 012654 005061 0000000 1$:   CLR   LPROJ(R1)    ;CLEAR PROJECT/PROGRAMMER #
201 012660 005061 0000000          CLR   LPROG(R1)
202 012664 012700 0000000          MOV   #OFFEMT,RO   ;POINT TO LOG-OFF EMT ARG BLOCK
203 012670 104375          EMT   375          ;LOG OFF THIS JOB

```

OFF command

```

1          ; -----
2          ; Subroutine called if the /NOWARN qualifier is specified with the OFF
3          ; command.
4          ;
5 012672   105267   165137   OFFNWN: INCB   OFFNWF           ; Set NOWARN flag
6 012676   000207           RETURN
7          ;
8          ; -----
9          ; Subroutine called to determine if primary process has any active
10         ; subprocesses and print warning message if so.
11         ;
12         ; Inputs:
13         ; R1 = Process index number
14         ;
15         ; Outputs:
16         ; C-flag cleared ==> Proceed with logoff.
17         ; C-flag set    ==> Abort the logoff.
18         ;
19 012700   010246   CKOFSP: MOV     R2, -(SP)
20 012702   010546           MOV     R5, -(SP)
21         ;
22         ; See if this process has any active subprocesses
23         ;
24 012704   005005           CLR     R5           ; Count active subprocesses in R5
25 012706   016102   0000000 MOV    LSECT(R1),R2 ; Point to table of subprocess #'s
26 012712   012700   0000000 MOV    #MAXSEC,R0  ; Get max # subprocesses
27 012716   001454           BEQ    7$           ; Br if none to check
28 012720   105722   1$: TSTB   (R2)+      ; Is this subprocess active?
29 012722   001401           BEQ    4$           ; Br if not
30 012724   005205           INC    R5           ; Count # active subprocesses
31 012726   077004   4$: SOB    R0,1$     ; Loop to check all
32 012730   005705           TST   R5           ; Any active subprocesses?
33 012732   001446           BEQ    7$           ; Br if not
34         ;
35         ; There are active subprocesses.
36         ; Print warning message.
37         ;
38 012734   020527   000001   CMP    R5,#1       ; One active subprocess?
39 012740   003004           BGT   2$           ; Br if more than one
40 012742           .PRINT #TM$SA1 ; You have 1 active subprocess
41 012750   000410           BR    5$
42 012752           .PRINT #TM$SA3 ; You have
43 012760   004767   0000000 CALL  PRTDEC       ; Print # subprocesses
44 012764           .PRINT #TM$SA4 ; active subprocesses
45         ;
46         ; Read response
47         ;
48 012772           5$: .GTLIN #BLKO,#TM$SA2 ; Print prompt and read response
49         ;
50         ; If first non-blank letter of response is "Y" then logoff,
51         ; otherwise abort the logoff.
52         ;
53 013012   012702   0000000 MOV    #BLKO,R2    ; Point to response buffer
54 013016   112200   3$: MOVB  (R2)+,R0  ; Get next char of response
55 013020   001411           BEQ   8$           ; Br if hit end of line
56 013022   120027   000040   CMPB  R0,#40      ; Ignore spaces
57 013026   001773           BEQ   3$

```

OFF command

```

58 013030 120027 000131          CMPB   RO,#'Y          ;Is response Yes?
59 013034 001405                BEQ    7#              ;Br if yes
60 013036 120027 000171          CMPB   RO,#'y          ;Allow lower-case too
61 013042 001402                BEQ    7#
62                                ;
63                                ; Abort the logoff
64                                ;
65 013044 000261                B#:#   SEC            ;Signal abort on return
66 013046 000401                BR     9#
67                                ;
68                                ; Allow the logoff
69                                ;
70 013050 000241                7#:#   CLC            ;Signal to logoff
71                                ;
72                                ; Finished
73                                ;
74 013052 012605                9#:#   MOV    (SP)+,R5
75 013054 012602                MOV    (SP)+,R2
76 013056 000207                RETURN

```

OFF command

```

1
2 ; -----
3 ; Complement the contents of BLKO buffer.
4 ; We do this to mildly encrypt the authorization file.
5 013060 010146 CVTBUF: MOV R1, -(SP)
6 013062 010246 MOV R2, -(SP)
7 013064 012701 0000000 MOV #BLKO, R1 ; POINT TO BUFFER
8 013070 012702 000400 MOV #256., R2 ; GET # WORDS TO COMPLEMENT
9 013074 005121 1#: COM (R1)+ ; COMPLEMENT CONTENTS OF BUFFER
10 013076 07720? SOB R2, 1#
11 013100 01260? MOV (SP)+, R2
12 013102 01260? MOV (SP)+, R1
13 013104 00020? RETURN

```

KILL command

```

1          .SBTTL  KILL command
2          ;-----
3          ; The KILL command is used to abort the specified job.
4          ;
5 013106  012705  0000000  CMDKIL: MOV      #KILEMT,R5      ;Point to Kill EMT arg block
6 013112  000405          BR          JOBCOM
7
8          .SBTTL  SUSPEND command
9          ;-----
10         ; Suspend the execution of a specified job
11         ;
12 013114  012705  0000000  CMDSPN: MOV      #SJEMT,R5      ;Point to suspend EMT arg block
13 013120  000402          BR          JOBCOM
14
15         .SBTTL  RESUME command
16         ;-----
17         ; Resume the execution of a suspended job
18         ;
19 013122  012705  0000000  CMDRSM: MOV      #RJEMT,R5      ;Point to resume EMT arg block
20         ;
21         ; Accrue the job number
22         ;
23 013126  004767  0000000  JOBCOM: CALL     CVTTAB          ;convert tab and FF chars to spaces
24 013132  004767  0000000          CALL     ACRDEC          ;accrue decimal value
25 013136  006301          ASL      R1          ;Convert # to job index number
26 013140  001403          BEQ      1$          ;Br if invalid line number
27 013142  020127  0000000          CMP      R1,#LSTSL        ;Is this a valid line number
28 013146  101404          BLOS    2$          ;Br if ok
29 013150          1$: FABORT #BDLIN      ;Invalid line number
30         ;
31         ; See if we are privileged to affect this job
32         ;
33 013160  010102          2$: MOV      R1,R2          ;Get job # to R2 for CKACDJ
34 013162  004767  0000000          CALL     CKACDJ          ;Can we access this job
35 013166  103404          BCS     9$          ;Br if not
36         ;
37         ; Execute EMT to affect the job
38         ;
39 013170  010500          MOV      R5,R0          ;Point to EMT arg list
40 013172  010160  0000004          MOV      R1,4(R0)       ;Set job number
41 013176  104375          EMT     375          ;Perform the function
42         ;
43         ; Finished
44         ;
45 013200  000167  0000000  9$: JNP      RDCMD

```

BOOT and \$STOP commands

```

1          .SBTTTL  BOOT and $STOP commands
2          ;-----
3          ; Reboot RT-11.
4          ;
5 013204  004767  0000000  CMDBOT: CALL  CKPRIV  ;USER MUST HAVE OPERATOR COMMAND PRIVILEGE
6 013210  004767  0000000          CALL  CVTTAB  ;CONVERT TAB AND FF CHARS TO SPACES
7          ;
8          ; See if a boot device was specified.
9          ;
10 013214  005067  0000000          CLR    BOTDEV  ;ASSUME NO BOOT DEVICE WILL BE SPECIFIED
11 013220  105713          TSTB   (R3)    ;WAS A BOOT DEVICE SPECIFIED?
12 013222  001405          BEQ    7$      ;BR IF NOT -- USE SYSTEM DEVICE
13          ; Accrue device name.
14 013224  004767  0000000          CALL  CTRD50  ;ACCRUE RAD50 DEVICE NAME
15 013230  016767  0000000 0000000  MOV    R50BUF,BOTDEV ;SAVE RAD50 NAME OF BOOT DEVICE
16          ;
17          ; Make sure system is idle.
18          ;
19          ; See if any other users are logged on.
20 013236  005005          7$:   CLR    R5      ;IDLE FLAG
21 013240  126727  0000000 0000001  CMPB  NUMON,#1  ;ARE WE ONLY JOB LOGGED ON?
22 013246  003404          BLE   1$      ;BR IF YES
23 013250          .PRINT #OTHRON ;OTHER USERS LOGGED ON
24 013256  005205          INC   R5      ;REMEMBER NOT IDLE
25          ; See if the spooler is idle.
26 013260  004767  0000000          1$:   CALL  SPLACT  ;SEE IF SPOOLER IS ACTIVE
27 013264  103004          BCC   2$      ;BR IF SPOOLER IS IDLE
28 013266          .PRINT #SPLPND ;THERE ARE PENDING SPOOL FILES
29 013274  005205          INC   R5      ;REMEMBER SYSTEM IS NOT IDLE
30          ; See if we need to ask for confirmation.
31 013276  005705          2$:   TST   R5      ;WAS SYSTEM IDLE?
32 013300  001420          BEQ   10$     ;BR IF YES
33 013302          .PRINT #STPASK ;ASK FOR CONFIRMATION
34 013310          5$:   .TTYIN  ;ACCEPT INPUT LINE
35 013314  120027  000131          CMPB  R0,#'Y   ;DID HE SAY YES?
36 013320  001410          BEQ   10$     ;BR IF YES
37 013322  120027  000171          CMPB  R0,#171 ;LOWER-CASE 'Y'
38 013326  001405          BEQ   10$     ;
39 013330  120027  000012          CMPB  R0,#LF   ;EAT REST OF LINE
40 013334  001365          BNE   5$      ;
41 013336  000167  0000000          JMP   RDCMD   ;GO GET NEXT COMMAND
42          ;
43          ; Force logoff of all users.
44          ;
45 013342  112767  000001  0000000  10$:  MOVB   #1,STPFLO ;SET FLAG SAYING SYSTEM IS BEING STOPPED
46 013350  012702  0000000          MOV   #LSTSL,R2 ;GET # OF LAST LINE
47 013354  120267  0000000          11$:  CMPB  R2,CORUSR ;IS THIS OUR LINE?
48 013360  001405          BEQ   12$     ;BR IF YES
49 013362  012700  0000000          MOV   #KILEMT,R0 ;POINT TO KILL EMT ARG BLOCK
50 013366  010260  0000004          MOV   R2,4(R0) ;SET # OF LINE TO KILL
51 013372  104375          EMT   375     ;KILL THE LINE
52 013374  162702  0000002          12$:  SUB   #2,R2  ;DO NEXT LINE
53 013400  001365          BNE   11$     ;
54          ;
55          ; Now enter logoff processing for our line.
56          ; It will jump to DOSTOP after it finishes.
57          ;

```

58 013402 000167 176040 JMP CMDOFF ;LOG OFF OUR LINE

#SHUTDOWN command

```
1 .SBTTL #SHUTDOWN command
2 ;-----
3 ; Do a gentle shutdown of system. Don't force jobs off but don't let
4 ; any log on either.
5 ;
6 013406 004767 0000000 CMDSHT: CALL CKPRIV ;USER MUST HAVE OPERATOR PRIV
7 013412 112767 177777 0000000 MOVE #-1,STPFLO ;SAY SYSTEM SHUTDOWN TAKING PLACE
8 013420 005067 0000000 CLR BOTDEV ;REBOOT FROM SYSTEM DISK
9 013424 000167 0000000 JMP RDCMD
10 000001 .END
```

Errors detected: 0

*** Assembler statistics

Work file reads: 0
Work file writes: 0
Size of work file: 12080 Words (48 Pages)
Size of core pool: 17920 Words (70 Pages)
Operating system: RT-11

Elapsed time: 00:01:40.48
DK: TSKM2A, LP: TSKM2A=DK: TSKM2A. MAC/C/N: SYM

\$1STLG	1-68			
\$CARUP	1-73			
\$CCLRN	1-74			
\$CFABT	1-91	7-17	7-34	
\$CFALL	1-98			
\$CFCCCL	1-98			
\$CFDCC	1-98			
\$CFOPN	1-104	42-56		
\$CFSOT	1-96			
\$CHACT	1-66			
\$CLTST	1-83			
\$CTRLC	1-89			
\$CTRLD	1-133			
\$CTRLO	1-49			
\$CTRLS	1-79	42-81		
\$DBKMN	1-72			
\$DEAD	1-136			
\$DEBUG	1-134			
\$DEFER	1-110			
\$DETCH	1-77	33-36	33-124	
\$DIBOL	1-68			
\$DILUP	1-93	33-97	33-122	
\$DISCN	1-78			
\$DOOFF	1-100	42-58		
\$DUPRN	1-95			
\$ECHO	1-97			
\$EMTTR	1-82			
\$FORM	1-96			
\$FORMO	1-98			
\$HARD	1-136			
\$HITTY	1-67			
\$INCOR	1-113			
\$INDDF	1-135			
\$INDRN	1-135			
\$INIT	1-136			
\$INKMN	1-89	33-103	33-132	
\$KED	1-113			
\$KINIT	1-57			
\$LC	1-97			
\$LOFCF	1-181	42-23	42-57	
\$NOABT	1-184	42-168	42-196	
\$NOIN	1-67	42-21	42-31	42-39
\$NOINT	1-182			
\$NOWIN	1-38			
\$NOWTT	1-67			
\$PAGE	1-97			
\$PHONE	1-136			
\$PRGLK	1-75	42-17		
\$QTSET	1-118			
\$QUIET	1-111	7-21		
\$RNIOP	1-183			
\$RNMLK	1-183			
\$SCCA	1-39	7-15		
\$SCOPE	1-97			
\$SETRN	1-97			
\$SGALL	1-110			

AF#IOP	1-183	5-11	5-12					
AF#MEM	1-65	5-14						
AF#NOI	1-182	5-15						
AF#NOW	1-182	5-16						
AF#NPW	1-39	5-17						
AF#PLK	1-43	5-13						
AF#SCA	1-68	5-21						
AF#SET	1-40	5-20						
AF#TPO	1-42	5-22						
AF#UCL	1-40	5-23						
AFCF	1-39							
ALC1DV	4-6#	11-11*	11-43	11-45*	11-86	12-5*	12-31*	12-61
ALCDEV	1-28	11-39*	12-43*	12-63*	22-41*	26-38*	26-68	
ALDBLK	1-166	31-17*	31-18					
ALDEMT	1-76	11-49						
ALDEX	1-165	1-166	31-11	31-11				
ALFN	1-177	31-27						
AMBOPT	1-151	1-164	29-65					
AR##SZ	1-180	42-129						
AR#CNT	1-179	42-140*						
AR#CON	1-179	42-138*						
AR#CPH	1-179	42-143*	42-144*					
AR#CPL	1-179	42-142*						
AR#DMY	1-180							
AR#PRG	1-179	42-127						
AR#PRJ	1-179	42-125						
AR#UNM	1-179							
ARNRFB	1-180	42-124						
ASCIS	8-53	8-62#						
ASCOK	8-65	8-81#						
ASDELM	8-17	9-7#						
ASDEX	1-148	8-63						
ASFID	8-36	8-57#						
ASKLNM	1-147	8-8*	8-52*	8-85				
ASNEND	1-95	10-14	10-28	42-110				
ASNHD1	1-160							
ASNHD2	1-160							
ASNOVF	1-148	1-173	8-97					
ASNSRC	1-122	8-82	8-90	8-94	25-31	29-21	35-45	
ASNTBL	1-93	10-12	10-22	42-108				
ASSMPL	8-42#							
AT##SZ	1-94	10-27						
AT#DEV	1-94	8-84	8-105*	10-26*	25-33	29-23	35-47	
AT#EXT	1-94	8-108*						
AT#FIL	1-94	8-106*	8-107*					
AT#LOG	1-94	8-102*	10-23	10-25*				
AT#SIZ	1-94	8-103*						
AUTHFN	1-170	42-113						
BADCMD	1-147	8-15	29-12					
BADDAT	39-16	39-18	39-22	39-33	39-37	39-41	39-43	39-49#
BADPMT	1-154							
BADPRI	1-154							
BADSAV	1-147							
BADTIM	40-16	40-20	40-24	40-35	40-49#			
BASMAP	1-127							
BDFNAM	1-150	18-99	23-36	24-17	38-37			

BDLGP	1-155								
BDLIN	1-168	33-34	37-25	45-29					
BELL	2-7#	33-49							
BLANK	2-6#	33-65							
BLKO	1-146	31-32	31-34	42-120	42-123	42-147	43-48	43-53	44-7
BLKWDS	2-10#								
BOTDEV	1-88	46-10*	46-15*	47-8*					
BOTUNI	1-88								
C. CSW	1-101	27-27							
C. DEVQ	1-101	27-31							
C. SBLK	1-101								
CALUCL	1-159								
CASCBR	1-84	41-32							
CASCUP	1-84	41-35							
CASTBR	1-84	41-31							
CASTBW	1-84	41-34							
CASTRO	1-85	41-30							
CASTWO	1-85	41-33							
CCLNAM	1-174								
CCLSAV	1-81	27-59							
CCSPRV	1-181								
CD##SZ	1-102	26-115							
CD##UB	1-102								
CD#BAS	1-102	26-76							
CD#DVU	1-102	26-74							
CD#JOB	1-102								
CD#NAM	1-102								
CD#TOP	1-118								
CDBUF	1-47	26-94	26-96						
CDGET	1-47	26-93							
CF#IND	1-137								
CF#QUT	1-137								
CFABLV	1-138								
CFBLK	1-111								
CFBUF	1-81	42-54							
CFCHAN	1-110	42-47	42-47	42-54	42-54				
CFEND	1-81								
CFHOLD	1-117								
CFIND	1-100								
CFLFL4	1-105								
CFNEST	1-103								
CFPNT	1-111	7-8							
CFSEND	1-104								
CFSP	1-104								
CFSPND	1-116	7-35*	7-35	7-37*					
CFSQEZ	1-42								
CFSTK	1-57								
CFSTOP	1-41	7-10							
CFSTRT	1-41	7-36							
CFSTS	1-137								
CHAIN	1-89								
CHKALC	1-66	26-69							
CHKDEV	1-174	13-30	18-53	26-75	27-18				
CHKDLM	1-164	29-49	35-63	35-91					
CHKMNT	1-149	25-50	26-48						
CHKMTX	1-149	25-52	26-54	26-106					

CINDAT	1-126												
CINFLG	1-48												
CKACQJ	1-44	45-34											
CKOFSP	42-25	43-19#											
CKPRIV	1-151	31-5	31-44	31-50	31-78	31-90	32-48	32-54	32-61	32-70	33-6	39-12	
	40-12	46-5	47-6										
CKSYDV	26-42	27-11#											
CKSYPV	1-44	16-4	17-4	41-6									
CLDEVX	1-124												
CLOTIR	1-53												
CLRPRV	1-46	16-6											
CLSFSP	1-85												
CLTOTL	1-85												
CMDACC	1-32	35-5#	35-114										
CMDALC	1-31	11-7#											
CMDASN	1-30	8-7#											
CMDBOT	1-33	46-5#											
CMDBUF	1-142	1-159	7-23										
CMDCCCL	1-174	26-127											
CMDDAT	1-29	39-5#											
CMDDET	1-32	37-5#											
CMDDLG	1-31	12-5#											
CMDDMT	1-30	25-5#											
CMDDSN	1-27	10-5#											
CMDDSP	1-30	7-44#	7-45										
CMDFMT	1-29	26-10#											
CMDFRM	1-27	32-79#											
CMDHD	1-26												
CMDINI	1-32	26-3#											
CMDINS	1-28	15-5#											
CMDKIL	1-31	45-5#											
CMDMNT	1-30	22-3#											
CMDMON	1-30	28-7#											
CMDOFF	1-26	42-5#	46-58										
CMDPAU	1-32	7-8#											
CMDREM	1-31	13-3#											
CMDRSM	1-28	45-19#											
CMDRST	1-29	41-6#											
CMDRSY	1-29												
CMDSHT	1-33	47-6#											
CMDSDND	1-29	33-21#											
CMDSPN	1-29	45-12#											
CMDSPQ	1-31	29-3#											
CMDSQZ	1-32	26-9#											
CMDTIM	1-30	40-5#											
CMDYEL	1-28	33-6#											
COAD	1-165	1-166	31-20										
COAF	1-178	42-115											
COAL	1-165	1-166	31-8										
COLOO	1-159												
CONFIG	1-99	40-40											
CORUSR	1-49	33-50	33-120	34-40	35-113	42-5	42-179	46-47					
CPUAH	1-159	42-100*	42-144										
CPUAL	1-159	42-101*	42-147										
CR	2-5#	33-47	33-84										
CRLF	1-154	11-71	12-53	22-50	28-47	32-35	38-28	42-86					

DLTXT	1-156												
DMTALL	1-169	42-74											
DMTARG	1-148	23-26	25-59										
DMTLGQ	5-38	5-40	25-87#										
DMTNLQ	4-17#	25-7#	25-47	25-86*									
DMTNLQ	5-39	5-41	25-86#										
DMTQHD	5-37#	25-16											
DNTSUB	1-174	26-111											
DMYDEV	1-48	29-34											
DOASGN	1-73	11-87	23-106										
DOMNT	23-111	24-0#											
DOSTOP	1-170	42-179											
DZTXT	1-176												
EDIT	1-68												
EDTFIL	1-162												
EM\$ALC	1-41	11-22	12-8										
EM\$ATF	1-63	11-66											
EM\$CSE	1-42	13-46											
EM\$DAA	1-66	11-59	12-51	22-48									
EM\$DET	1-56	37-8											
EM\$DIU	1-66	11-69											
EM\$DNR	1-41	26-33											
EM\$FNI	1-46	17-10											
EM\$FOE	1-97	38-41											
EM\$IAD	1-63	1-63	11-63	12-57	18-58								
EM\$ITF	1-46	19-19											
EM\$LDI	1-43	18-57											
EM\$MPV	1-56												
EM\$NAD	1-45												
EM\$NAL	1-43												
EM\$NFW	1-41	26-16											
EM\$NID	1-64	30-24											
EM\$NLD	1-63	23-8											
EM\$NLN	1-43	34-19											
EM\$NPD	1-63												
EM\$NSF	1-64	35-9											
EM\$NUC	1-39												
EM\$NUK	1-74												
EM\$OLO	1-64	42-52											
EM\$SND	1-41	33-45											
EM\$SSY	1-64	26-44											
EM\$UAR	1-61	14-30											
EM\$UER	1-61	14-42											
ERRLOC	1-55	8-67	11-56	12-48	22-45	24-13	28-40						
ERRSEV	1-119												
FF	2-9#												
FILNAM	1-143	1-144	16-22	18-34	18-38*	18-43	18-52	18-69	18-73	23-40	23-45	23-54	
	23-63	38-11											
FIXPRV	1-43												
FKILL	1-143	8-70	8-97	11-22	11-63	11-66	12-8	12-26	12-57	13-46	15-27	17-10	
	18-57	18-58	18-99	19-19	22-21	22-28	23-8	23-15	23-36	23-50	23-82	25-24	
	26-16	26-33	26-44	26-56	28-30	28-42	28-55	29-43	29-65	29-66	30-24	31-8	
	33-16	33-45	34-19	34-37	35-9	35-17	35-119	37-8	37-19	37-25	38-33	38-37	
	38-41	39-50	40-50	45-29									
FORCEO	1-40	18-47	29-29										
FPRINT	1-143	11-59	11-69	12-51	14-30	14-42	22-48	24-17	28-44	42-52			

Cross reference table (GREF V05.04)

FSTD L	1-77	37-20										
FSTIOL	1-50	1-70										
GAGMSG	1-168	33-107										
GENMON	1-65	42-156										
GENTOP	1-88											
GRT1	1-129											
GTRD50	1-148	10-20	11-30	11-84	12-33	13-18	15-11	22-29	23-103	25-28	26-34	29-16
	35-35	35-65	35-76	39-24	46-14							
HANBSY	1-153											
HANCHN	1-50											
HANIDX	1-152											
HIMAP	1-127											
HIPRI	1-154											
HNBUF	1-152											
HUPARG	1-173											
IATTRB	4-13#	16-5*	16-35	21-31*								
II##SZ	1-182	18-78	19-13									
II#FLG	1-183	16-35*										
II\$NAM	1-183	16-21	17-14*	18-66	18-71	18-73	19-11					
II\$NPV	1-45	1-183	16-39									
II\$PRV	1-45	1-183	16-37									
IIBUF	1-47	16-21	16-35*	16-37	16-39	17-14*	18-66	18-71	18-73	19-11		
ILLCMD	1-149	15-27	22-28	25-24								
ILSW2	1-71											
IN\$ACT	1-135											
IN\$CMD	1-135											
IN\$CNT	1-135											
INDABT	1-143	42-37										
INDACT	1-145											
INDERR	1-91	39-49*	40-49*									
INDSAV	1-135	27-60										
INDSTA	1-91											
INDTSV	1-38	27-61										
INFOMT	1-149	25-54										
INGADR	1-47	18-63*	19-8*									
INGEMT	1-47	18-64	19-9									
INPADR	1-46	20-7*										
INPEMT	1-46	20-8										
INSADD	15-14	16-4#										
INSATR	5-6	5-7	5-8	5-9	5-10	5-11	5-12	5-13	5-14	5-15	5-16	5-17
	5-19	5-20	5-21	5-22	5-23	21-31#						
INSDEL	15-16	15-18	17-4#									
INSFRE	16-17	19-7#										
INSHD	5-5#	21-14										
INSNAM	16-11	17-8	18-13#									
INSOPT	16-7	16-30	21-10#									
INSPUT	16-48	17-15	20-7#									
INSSRC	1-40											
INSTBL	1-182	18-62	19-7									
INSTBN	1-182	18-79	19-14									
INV DAT	1-178	39-50										
INVDEV	1-177	29-43										
INVEC	1-136											
INVLDM	1-173	23-50										
INVLDN	1-150	23-15										
INV OPT	1-143	1-149	1-164	12-26	22-21	28-30	29-66	35-119	37-19			

LINCUR	1-71						
LINFRE	1-169	37-41					
LINIR	1-53						
LINNXT	1-69						
LINRTS	1-53						
LITIME	1-90						
LJSW	1-87						
LMONHD	1-65						
LMXLN	1-136						
LMXPRM	1-137						
LNBLKS	1-92						
LNMAP	1-99						
LNPRIM	1-99	33-32	33-51	33-121	42-80		
LNSBLK	1-93						
LNSPAC	1-103						
LNUMER	33-31	33-34#	33-37	33-91			
LOCKTX	1-158						
LOFSPC	1-96	42-43	42-47	42-51*	42-59*		
LOGASN	1-150	18-44	23-41				
LOGBAS	1-117	1-119					
LOGBLK	1-139						
LOGBUF	1-139						
LOGCHK	1-118	23-21	25-43	26-123			
LOGCHN	1-139						
LOGCLS	1-154	42-152					
LOGDVU	1-117	1-119					
LOGFLG	1-139	42-70*					
LOGPTR	1-139						
LOMAP	1-127						
LOUTIR	1-53						
LPARNT	1-57	34-41	34-43				
LPRI	1-138						
LPROG	1-78	42-119	42-201*				
LPROJ	1-78	42-105	42-200*				
LRBFIL	1-100						
LSCCA	1-96						
LSECPT	1-82	43-25					
LSTACT	1-69						
LSTATE	1-125						
LSTDLD	1-77	37-26	42-82	42-171			
LSTHL	1-50						
LSTIOL	1-50						
LSTPL	1-120	33-119	42-15	42-169			
LSTPRM	1-116						
LSTSL	1-125	33-30	42-178	45-27	46-46		
LSTSPL	1-52	32-43					
LSUCF	1-74						
LSW	1-49	33-36	33-97	33-122	33-124	42-58*	42-78
LSW11	1-38						
LSW2	1-89						
LSW2S	1-95						
LSW3	1-95	42-21	42-31*	42-39*	42-81*		
LSW4	1-112	7-21	33-103	33-132	42-56*		
LSW5	1-75	7-15	42-19				
LSW6	1-134	7-17*	7-34*				
LSW7	1-137	33-101	33-128				

SPLSNG	6-16	32-48#			
SPLSTA	6-17	32-4#			
SPNBB	31-80	31-93#			
SPNBSY	30-30	30-30#	31-93		
SPNHEM	4-36#	32-71#	32-72		
SPOLGO	31-73	32-100#			
SPSNG	1-165	1-167	32-16		
SPUBUF	1-58				
SPWFM	1-165	1-166	32-10		
SQZDEV	4-5#	26-37#	26-74	26-122	
SRTSIZ	1-128				
SRTSMS	1-172				
SRTTXT	1-175				
SSRMAP	1-175				
STLGCN	1-27				
STLGHD	1-154				
STPASK	1-172	46-33			
STPFLG	1-80	42-87	42-162	46-45*	47-7*
SUBARD	1-162				
SUBTXT	1-175				
SUPCOD	1-129				
SWPCHN	1-38	27-55			
SWPTX	1-158				
SXBPNT	1-58				
SYFLVC	27-24	27-55#			
SYHDI	1-157				
SYHD2	1-157				
SYINDX	1-130	36-37			
SYNAME	1-131				
SYSAV	1-142				
SYSDAT	1-126	39-46#			
SYTIMH	1-126	40-45#			
SYTIML	1-126	40-46#			
SYUNIT	1-130	36-40			
TAB	2-8#				
TALEMT	1-76	22-42			
TBLOVF	1-151	35-17			
TECO	1-68				
TIMSPL	4-42#	42-191			
TIMUP	42-92	42-105#			
TK1SEC	1-128	42-96			
TK1VAL	1-126				
TM#SA1	1-37	43-40			
TM#SA2	1-37	43-48			
TM#SA3	1-37	43-42			
TM#SA4	1-37	43-44			
TNIDLH	1-62	41-29			
TNIOH	1-62	41-26			
TNIOWH	1-59	41-28			
TMSWPH	1-62	41-27			
TMSWTH	1-62	41-25			
TMTOTH	1-59	1-171	41-23		
TMTOTL	1-59	1-171			
TMUSRH	1-59	41-24			
TUOLNG	1-42				
TOTMMS	1-175				

... CM0	8-63	8-63	8-63	31-11	31-11	31-11	43-48	43-48	43-48	43-48			
... CM1	23-91	31-18	31-27	31-32	31-34	38-11	42-47	42-54	42-113	42-120	42-147		
... CM2	14-16	14-24	14-37	23-91	23-91	31-18	31-18	31-18	31-27	31-27	31-27	31-27	31-27
	31-32	31-32	31-32	31-32	31-34	31-34	31-34	31-34	38-11	38-11	42-47	42-47	
	42-54	42-54	42-54	42-54	42-113	42-113	42-120	42-120	42-120	42-120	42-147	42-147	
	42-147	42-147	42-191										
... CM3	31-21	31-38	31-39	38-13	42-148								
... CM5	7-23	7-27	7-50	11-71	12-53	14-16	14-24	14-37	22-50	23-91	25-54	28-47	
	30-33	31-18	31-20	31-27	31-32	31-34	32-6	32-10	32-12	32-16	32-20	32-22	
	32-27	32-33	32-35	32-39	32-41	32-42	33-99	33-107	37-39	37-41	37-49	38-11	
	38-26	38-28	42-47	42-54	42-86	42-89	42-91	42-113	42-115	42-120	42-147	42-191	
	43-40	43-42	43-44	46-23	46-28	46-33							
... CM6	14-16	14-24	14-37	42-191									
... CM7	31-27	31-32	31-34	42-54	42-120	42-147							
. CLOSE	1-20#	31-21	31-38	31-39	38-13	42-148							
. CRRG	1-16#	14-16	14-24										
. CSIGE	1-18#	31-11											
. CSISP	1-16#	8-63											
. DATE	1-19#												
. ELRG	1-16#	14-37											
. ENTER	1-21#	31-18											
. EXIT	1-21#												
. FPROT	1-22#	23-91											
. GTIM	1-19#												
. GTLIN	1-19#	43-48											
. GVAL	1-22#												
. HERR	1-22#												
. LOOKU	1-20#	38-11	42-47	42-113									
. PRINT	1-20#	7-23	7-27	7-50	11-71	12-53	22-50	25-54	28-47	30-33	31-20	32-6	
	32-10	32-12	32-16	32-20	32-22	32-33	32-35	32-39	32-41	32-42	33-99	33-107	
	37-39	37-41	37-49	38-26	38-28	42-86	42-89	42-115	43-40	43-42	43-44	46-23	
	46-28	46-33											
. PURGE	1-17#												
. PVAL	1-22#												
. READW	1-17#	31-32	42-54	42-120									
. REOPE	1-18#												
. SAVES	1-18#												
. SERR	1-22#												
. SPFUN	1-19#												
. SRESE	1-16#												
. TTOUT	1-16#												
. TTYIN	1-17#	7-28	46-34										
. TTYOU	1-17#	32-27	42-91										
. TWAIT	1-17#	42-191											
. WRITW	1-21#	31-27	31-34	42-147									
CMDDEF	3-47#	5-6	5-7	5-8	5-9	5-10	5-11	5-12	5-13	5-14	5-15	5-16	
	5-17	5-18	5-19	5-20	5-21	5-22	5-23	5-30	5-31	5-38	5-39	5-40	
	5-41	5-48	5-49	5-56	5-57	5-64	6-7	6-8	6-9	6-10	6-11	6-12	
	6-13	6-14	6-15	6-16	6-17								
FABORT	3-15#	8-70	8-97	11-22	11-63	11-66	12-8	12-26	12-57	13-46	15-27	17-10	
	18-57	18-58	18-99	19-19	22-21	22-28	23-8	23-15	23-36	23-50	23-82	25-24	
	26-16	26-33	26-44	26-56	28-30	28-42	28-55	29-43	29-65	29-66	30-24	31-8	
	33-16	33-45	34-19	34-37	35-9	35-17	35-119	37-8	37-19	37-25	38-33	38-37	
	38-41	39-50	40-50	45-29									
FERR	3-4#	11-59	11-69	12-51	14-30	14-42	22-48	24-17	28-44	42-52			
FWARN	3-23#	24-15	33-34	33-114	33-140	35-102							

TBLDEF	3-35#	5-5	5-29	5-37	5-47	5-55	5-63	6-6
TBLEND	3-61#	5-24	5-32	5-42	5-50	5-58	5-65	6-18