

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  | DETACH command                 |
| 37- | 1  | DATE command                   |
| 38- | 1  | TIME command                   |
| 39- | 1  | RESET command                  |
| 40- | 1  | OFF command                    |
| 43- | 1  | KILL command                   |
| 43- | 8  | SUSPEND command                |
| 43- | 15 | RESUME command                 |
| 44- | 1  | BOOT and \$STOP commands       |
| 45- | 1  | \$SHUTDOWN command             |

```

1          .TITLE  TSKM2A -- Keyboard command routines
2          .ENABL  LC
3          .DSABL  GBL
4 000000   .CSECT  TSKM2A
5 000000
6
7          ;
8          ; TSKM2A is the portion of TSKMON that contains the actual code
9          ; to implement each keyboard command that is processed by TSKMON.
10         ;
11         ; Copyright 1978, 1979, 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988.
12         ; S&H Computer Systems, Inc.
13         ; Nashville, Tennessee
14         ;
15         ; Macro calls
16         ;
17         .MCALL .CSISPC, .CRRG, .ELRG
18         .MCALL .READW, .TTYIN, .TTYOUT, .TWAIT
19         .MCALL .CSIQEN
20         .MCALL .GTLIN
21         .MCALL .PRINT, .CLOSE, .LOOKUP, .CSTAT
22         .MCALL .WRITW, .ENTER
23         .MCALL .FPROT
24         ;
25         ; Global definitions
26         ;
27         .GLOBL TSKM2A, CMDOFF
28         .GLOBL CMDFRM, CMDDSN, PRGALL
29         .GLOBL DLCEMT, ALCDEV, CMDINS, CMDSPN, CMDRSM, CMDYEL
30         .GLOBL CMDRST, CMDSND, CMDFMT, CMDDAT
31         .GLOBL CMDTIM, CMDMNT, CMDMON, CMDDMT, CMDDSP, CMDASN
32         .GLOBL CMDALC, CMDDLC, OPRCMD, CMDSPO, CMDKIL, CMDREM
33         .GLOBL CMDPAU, CMDDDET, CMDINI, CMDSQZ
34         .GLOBL CMDBOT, CMDSHT
35         ;
36         ; Global references
37         ;
38         .GLOBL TM$SA1, TM$SA2, TM$SA3, TM$SA4, SKPSPC, SKPDLM
39         .GLOBL SWPCHN, SEGCHN, INDTSV, LWINDO
40         .GLOBL $SCCA, AF$CCA, AF$NPW
41         .GLOBL AF$DUP, AF$IND, AF$UCL, AF$SET, FORCEO
42         .GLOBL CFSTOP, CFSTRT, EM$ALC, EM$NFW, EM$SND, EM$DNR
43         .GLOBL SDFHD, AF$TPO, EM$CSE, WLDNAM
44         .GLOBL AF$PLK, AF$DBG, EM$LDI, EM$NLN
45         .GLOBL PO$NFW, PO$BYP, PO$ALC, CKSYPV, CKACDJ, PRIVSO
46         .GLOBL PFSO, PFCO, PVNPW, PRIVCO
47         .GLOBL CLRPRV, EM$FNI, EM$ITF, INPADR, INPEMT, OPTLST
48         .GLOBL CDBUF, CDGET, INGADR, INGEMT, IIBUF, PRVOPT
49         .GLOBL $VNOTT, SDNAME, DMYDEV
50         .GLOBL CORUSR, LSW, SERFLG
51         .GLOBL LSTSPL, SOPALC
52         .GLOBL SOPDAT, SOPTIM
53         .GLOBL ERRLOC
54         .GLOBL PO$DET, EM$DET, PO$SND
55         .GLOBL LPARNT
56         .GLOBL TMTOTH, TMUSRH, TMIDWH, LDMNT
57         .GLOBL R. GSIZ, RS. GBL, RS. PVT, R. NAME, RS. CRR, RS. EGR
58         .GLOBL EM$UAR, EM$UER, R. GSTS

```

```

58 . GLOBL TMSWTH, TMIDLH, TMIOH, TMSWPH, LDCLLEN
59 . GLOBL EM$IAD, EM$ATF, EM$NLD
60 . GLOBL EM$SSY, EM$NID, QHDMS1, QHDMS2, EM$OLO
61 . GLOBL GENMON, JS$OFF, AF$MEM, AF$BYA
62 . GLOBL EM$DAA, EM$DIU, CHKALC
63 . GLOBL $NOIN
64 . GLOBL AF$SCA
65 . GLOBL NUMON
66 . GLOBL DOASGN, LSW9
67 . GLOBL $PRGLK, LSW5, PVON
68 . GLOBL ALDEMT, TALEMT
69 . GLOBL LSTDLD, FSTDLD, $DETC
70 . GLOBL LPROJ, LPROG, LUNAME
71 . GLOBL LCPUHI, LCPULO, LCONTM, $CTRLS
72 . GLOBL STPFLG, TOTON, SPLCHN
73 . GLOBL CFBUF, CCLSAV, KMNCHN
74 . GLOBL MINTIM, LSECPT, MAXSEC
75 . GLOBL CASTBR, CASCBR, CASTBW, CASCUP
76 . GLOBL CASTRO, CASTWO
77 . GLOBL CTRLTT
78 . GLOBL BOTDEV
79 . GLOBL $INKMN, UFORM
80 . GLOBL $CFABT, INDERR
81 . GLOBL ASNTBL, $DILUP, CSHDEV, CSHDVN
82 . GLOBL AT$LOG, AT$SIZ, AT$DEV, AT$FIL, AT$$SZ, AT$EXT
83 . GLOBL ASNEND, LSW3
84 . GLOBL LOFSPC, R50COM
85 . GLOBL EM$FOE
86 . GLOBL LNPRIM, CW$50H, CONFIG
87 . GLOBL $DOOFF
88 . GLOBL C. CSW, C. DEVQ
89 . GLOBL CD$DVU, CD$BAS, CD$$SZ
90 . GLOBL $CFOPN, $TTGAG
91 . GLOBL SDSFCB, SD$DEL
92 . GLOBL SDFLAG, SD$FLK, SD$WFM, SDFORM, SD$FLG
93 . GLOBL LD$RON
94 . GLOBL LDNAME, LDSIZE, LDFLAG, LDBASE, LDPDEV
95 . GLOBL CFCHAN, LDDEVX
96 . GLOBL CFPNT, $QUIET
97 . GLOBL LSW4, SDSKIP, SDBU, SD$BAK
98 . GLOBL SD$SNG, NFRESB
99 . GLOBL SD$WID
100 . GLOBL CFSPND
101 . GLOBL LOGCHK
102 . GLOBL LSTPL, SDCB, SDCBND
103 . GLOBL DCTRD, DCCRD, DCTWR, DCCWR, ASNSRC
104 . GLOBL SDCBSZ, LSTSL
105 . GLOBL SYSDAT, SYTIMH, SYTIML
106 . GLOBL TK1SEC
107 . GLOBL RESDEV
108 . GLOBL LSW6, PF$SYS, PF$IOW
109 . GLOBL INDSAV
110 . GLOBL LSW7
111 . GLOBL LOGFLG
112 . GLOBL LF$WRT
113 . GLOBL CVTUC
114 . GLOBL CMDBUF, PAUMSG, RDCMD, CVTTAB, SEARCH

```

```

115 . GLOBL INVOPT, FKILL, ABRTCF, INDABT, XAREA, FPRINT
116 . GLOBL PUSHCF, FILNAM, R50SAV
117 . GLOBL BLKO
118 . GLOBL ASKLNM, BADCMD, KCSIBF
119 . GLOBL ASDEX, KCSIMS, ASNOVF, QTRD50, R50BUF, R50LDO, MNTDEV, DMTARG
120 . GLOBL DEADEV, CHKMNT, CHKMTX, INFOMT, NOFLAG, MTOPHD, ILLCMD
121 . GLOBL R5OLD, INVLDN, R5ODSK, ACRFIL, BDFNAM, LOGASN, MNTFUL, R5OLD7
122 . GLOBL CKPRIV, AMBOPT, ACRDEC
123 . GLOBL PRTDEC
124 . GLOBL ACROCT, CSIMS1, POPCF
125 . GLOBL CRLF, LOGCLS
126 . GLOBL DIVIDE
127 . GLOBL CPUAH, CPUAL
128 . GLOBL MNFLGS, MNBPC
129 . GLOBL DELSPC, MNBASE, MNTOP, MONHD, MONAR1, NOPMGN, PMBUSY, MONAR2
130 . GLOBL CHKDLM, SPLHD
131 . GLOBL DEVIDL, ALDEX, SPACTV, SPWFM, SPSNG
132 . GLOBL COAL, ALDBLK, COAD, SPFLG
133 . GLOBL SPFUL, SPCF, SPFLK, NOFIL, SPQEMT, NOOPTT, NSPFLG
134 . GLOBL BDLIN, MSQBUF, NUMTB1, MSGEND, NOTON, GAGMSG, SPWIDE, SPNARD
135 . GLOBL YELEMT, LINFRE, DJABMS, DLMSG, INVTIM, DMTALL
136 . GLOBL SHTMSG, AUTHFN, SPLACT, DOSTOP, OFFEMT, KILEMT
137 . GLOBL OTHRON, SPLPND, STPASK
138 . GLOBL INVLDM, CSIMS4, MNTARG
139 . GLOBL OTRMNT, CHKDEV, DMTSUB, CMDCCCL
140 . GLOBL SJEMT, RJEMT
141 . GLOBL PRTTIM, INVDEV, ALFN
142 . GLOBL DETHD, DETARG, RUNMS, NOFRDL, R50MON, INV DAT, MUL32, COAF
143 . GLOBL AR$PRJ, AR$PRG, AR$CON, AR$CNT, AR$CPH, AR$CPL
144 . GLOBL AR$$SZ, ARNRPB
145 . GLOBL PRTWRN, VLDSYS, $LOFCF
146 . GLOBL INSTBL, INSTBN, AF$NOW, AF$HIE, AF$NOI, II$$SZ
147 . GLOBL AF$IOP, II$NAM, II$FLG, II$PRV, II$NPV
148 . GLOBL NMSGBF, $NOABT, BADBOT

```

```
1          ;
2          ; Assembly constants
3          ;
4          000012      LF      =      12      ; LINE FEED
5          000015      CR      =      15      ; CARRIAGE RETURN
6          000040      BLANK   =      40      ; ASCII SPACE
7          000007      BELL    =      07      ; ASCII BELL
8          000011      TAB     =      11      ; HORIZONTAL TAB
9          000014      FF      =      14      ; FORM FEED
10         000400      BLKWDS  =      256.    ; # OF WORDS IN DISK BLOCK
11         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 |        |        | CINFO: .BLKW 6. ;Cstat information necessary for boot  |
| 8  | 000020 |        |        | REMRDB: .BLKW 5  |
| 9  | 000032 | 003344 |        | R50ADD: .RAD50 /ADD/                                   |
| 10 | 000034 | 014724 | 021145 | R50DEL: .RAD50 /DELETE/                                |
| 11 | 000040 | 070525 |        | R50REM: .RAD50 /REM/                                   |
| 12 | 000042 | 000000 |        | IATTRB: .WORD 0  |
| 13 |        |        |        | ;  |
| 14 |        |        |        | ; Byte data  |
| 15 |        |        |        | ;  |
| 16 | 000044 | 000    |        | DMTNLG: .BYTE 0  |
| 17 | 000045 | 000    |        | OFFNWF: .BYTE 0  |
| 18 |        |        |        | . EVEN   |
| 19 |        |        |        | ;  |
| 20 |        |        |        | ; Emt to delete a spool file                           |
| 21 |        |        |        | ;  |
| 22 | 000046 | 000    | 126    | DELEMT: .BYTE 0,126                                    |
| 23 | 000050 | 000007 |        | .WORD 7  |
| 24 | 000052 | 000000 |        | .WORD 0 ;File ID goes here                             |
| 25 |        |        |        | ;  |
| 26 |        |        |        | ; Emt to set spool HOLD mode                           |
| 27 |        |        |        | ;  |
| 28 | 000054 | 000    | 126    | SPHLEM: .BYTE 0,126                                    |
| 29 | 000056 | 000015 |        | .WORD 15   |
| 30 | 000060 | 000000 |        | .WORD 0 ;Address of SDCB                               |
| 31 |        |        |        | ;  |
| 32 |        |        |        | ; Emt to set spool NOHOLD mode                         |
| 33 |        |        |        | ;  |
| 34 | 000062 | 000    | 126    | SPNHEM: .BYTE 0,126                                    |
| 35 | 000064 | 000016 |        | .WORD 16   |
| 36 | 000066 | 000000 |        | .WORD 0 ;Address of SDCB                               |
| 37 |        |        |        | ;  |
| 38 |        |        |        | ; Time to wait for check spooler activity.             |
| 39 |        |        |        | ;  |
| 40 | 000070 | 000000 | 000170 | TIMSPL: .WORD 0,60.*2. ;2 second timer                 |



```
1 .SBTTL Tables of commands and options
2 ;-----
3 ; Define option switches for the INSTALL ADD command
4 ;
5 000074 TBLDEF INS, 2
6 000002 CMDDEF BYPASN, INSATR, AF$BYA
7 000010 CMDDEF DUP, INSATR, AF$DUP
8 000016 CMDDEF DEB*UG, INSATR, AF$DBG
9 000024 CMDDEF HI*GH, INSATR, AF$HIE
10 000032 CMDDEF IND, INSATR, AF$IND
11 000040 CMDDEF IO*MAP, INSATR, AF$IOP
12 000046 CMDDEF IO*PAGE, INSATR, AF$IOP
13 000054 CMDDEF LOCK, INSATR, AF$PLK
14 000062 CMDDEF MEM*LOCK, INSATR, AF$MEM
15 000070 CMDDEF NONI*INTERACTIVE, INSATR, AF$NOI
16 000076 CMDDEF NOWA*IT, INSATR, AF$NOW
17 000104 CMDDEF NOWI*NDOW, INSATR, AF$NPW
18 000112 CMDDEF PRIV*ILEGED, PRVOPT, 0
19 000120 CMDDEF SC*CA, INSATR, AF$CCA
20 000126 CMDDEF SETUP, INSATR, AF$SET
21 000134 CMDDEF SING*LECHAR, INSATR, AF$SCA
22 000142 CMDDEF T*RANSSPARENT, INSATR, AF$TPD
23 000150 CMDDEF TSXUCL, INSATR, AF$UCL
24 000156 TBLEND
25
26 ;-----
27 ; Define option switches for the MOUNT command
28 ;
29 000074 TBLDEF MTOP, 1
30 000162 CMDDEF WR*ITE, 0
31 000166 CMDDEF NOW*RITE, 1
32 000172 TBLEND
33
34 ;-----
35 ; Define option switches for the DISMOUNT command
36 ;
37 000074 TBLDEF DMTQ, 1
38 000176 CMDDEF LOG, DMTLQ
39 000202 CMDDEF NOLOG, DMTNLQ
40 000206 CMDDEF WARN, DMTLQ
41 000212 CMDDEF NOWARN, DMTNLQ
42 000216 TBLEND
43
44 ;-----
45 ; Define option switches for DETACH command
46 ;
47 000074 TBLDEF DET, 1
48 000222 CMDDEF CH*ECK, DETCHK
49 000226 CMDDEF KI*LL, DETKIL
50 000232 TBLEND
51
52 ;-----
53 ; Options for the MONITOR command.
54 ;
55 000074 TBLDEF MON, 1
56 000236 CMDDEF IO*WAIT, PF$IOW
57 000242 CMDDEF SY*STEM, PF$SYS
```

58 000246  
59  
60  
61  
62  
63 000074  
64 000252  
65 000256

TBLEND

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

```
1  
2 ; -----  
3 ; Define options for the SPOOL command  
4 ; Entry 1 = Option name  
5 ; Entry 2 = Processing routine  
6 000074 TBLDEF SPL, 1  
7 000262 CMDDEF A*LIGN, SPLALN  
8 000266 CMDDEF B*ACKUP, SPLBAK  
9 000272 CMDDEF D*ELETE, SPLDEL  
10 000276 CMDDEF FL*AGPAGE, SPLFLG  
11 000302 CMDDEF NOF*LAGPAGE, SPLNFL  
12 000306 CMDDEF F*ORM, SPLFRM  
13 000312 CMDDEF H*OLD, SPLHLD  
14 000316 CMDDEF NOH*OLD, SPLNHL  
15 000322 CMDDEF L*OCK, SPLLK  
16 000326 CMDDEF M*ULTIPLE, SPLMUL  
17 000332 CMDDEF NA*RROW, SPLNRW  
18 000336 CMDDEF SK*IP, SPLSKP  
19 000342 CMDDEF SI*NGLE, SPLSNG  
20 000346 CMDDEF ST*ATUS, SPLSTA  
21 000352 CMDDEF W*IDE, SPLWID  
22 000356 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 000074 016767 000000G 000000G CMDPAU: MOV      CFPNT,CFSPND      ;SUSPEND COMMAND FILE
9 000102 001441                      BEQ      9$              ;BR IF COMMAND FILE NOT RUNNING
10 000104 004767 000000G                      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 000110 032761 000000G 000000G          BIT      $$SCCA,LSW5(R1) ;Is control-C abort override in effect?
16 000116 001003                      BNE     3$              ;Br if yes -- Don't abort on control-C
17 000120 052761 000000G 000000G          BIS     $$CFABT,LSW6(R1);Abort command files if ctrl-C typed
18          ;
19          ; If TTY QUIET is set, print pause message for operator
20          ;
21 000126 032761 000000G 000000G 3$:    BIT      $$QUIET,LSW4(R1); IS QUIET SET ON?
22 000134 001403                      BEQ     2$              ;BR IF NOT
23 000136                      .PRINT   #CMDBUF          ;DISPLAY THE PAUSE COMMAND
24          ;
25          ; Print pause message and wait for response
26          ;
27 000144                      2$:    .PRINT   #PAUMSG          ;PRINT PAUSE MESSAGE
28 000152                      1$:    .TTYIN          ;ACCEPT A LINE OF INPUT
29 000156 120027 000012                      CMPB   RO,#LF
30 000162 001373                      BNE     1$
31          ;
32          ; Restart the command file
33          ;
34 000164 042761 000000G 000000G          BIC     $$CFABT,LSW6(R1);Clear command-file abort flag
35 000172 016700 000000G                      MOV     CFSPND,RO      ;RESTART COMMAND FILE
36 000176 004767 000000G                      CALL   CFSTRT
37 000202 005067 000000G                      CLR    CFSPND
38 000206 000167 000000G 9$:    JMP     RDCMD
39          ;
40          .SBTTL  DISPLAY command
41          ;-----
42          ; Display a line from within a command file.
43          ;
44 000212 124327 000040          CMDDSP: CMPB   -(R3),#'          ;BACKUP OVER LEADING SPACES
45 000216 001775                      BEQ    CMDDSP
46 000220 005203                      INC    R3              ;POINT TO FIRST SEPARATOR
47 000222 122327 000040          CMPB   (R3)+,#'          ;IS IT A SPACE?
48 000226 001401                      BEQ    1$              ;BR IF YES (SKIP IT)
49 000230 005303                      DEC    R3              ;POINT TO 1ST SEPARATOR
50 000232                      1$:    .PRINT   R3          ;PRINT THE ARGUMENT OF THE DISPLAY COMMAND
51 000236 000167 000000G          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 000242 004767 000000G  CMDASN: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
8 000246 105067 000000G      CLRB   ASKLNM      ; SET FLAG FOR LONG ASSIGN
9 000252 126227 177777 000072  CMPB   -(R2),#':    ; DOES LOGICAL NAME HAVE COLON?
10 000260 001001          BNE    5$          ; BR IF NOT
11 000262 105042          CLRB   -(R2)        ; CLEAR IT FOR NOW
12 000264 010201 5$:     MOV    R2,R1      ; POINT TO NULL AT END OF LINE
13 000266 020103 1$:     CMP    R1,R3      ; SCAN BACK TOWARDS FRONT
14 000270 101002          BHI    2$          ; BRANCH IF MORE TO DO
15 000272 000167 000000G      JMP    BADCMD      ; MUST HIT SOME DELIM
16 000276 114100 2$:     MOVB   -(R1),R0    ; GET NEXT CHAR BACK
17 000300 004767 000320      CALL  ASDELM      ; IS IT A DELIMITER?
18 000304 001370          BNE    1$          ; IF NOT KEEP SCANNING
19          ; WE SCANNED BACK TO A DELIMITER -- SEE WHERE IT IS
20          ; IN COMMAND.
21 000306 120027 000072      CMPB   R0,#':      ; REPLACE ': ' AND ' ' WITH '='
22 000312 001414          BEQ    3$          ;
23 000314 120027 000040      CMPB   R0,#'      ;
24 000320 001013          BNE    4$          ;
25 000322 020103 6$:     CMP    R1,R3      ; HAVE WE REACHED FRONT OF COMMAND LINE?
26 000324 101407          BLOS   3$          ; BR IF YES
27 000326 124127 000040      CMPB   -(R1),#'   ; SKIP OVER ALL CONSECUTIVE BLANKS
28 000332 001773          BEQ    6$          ;
29 000334 121127 000072      CMPB   (R1),#':   ; ALLOW "ASSIGN DX1: DK" CONSTRUCT
30 000340 001401          BEQ    3$          ;
31 000342 005201          INC    R1          ; POINT TO 1ST BLANK CHARACTER
32 000344 112711 000075 3$:     MOVB   #'=(, (R1) ;
33 000350 010100 4$:     MOV    R1,R0      ; HOW MANY CHARS IN NAME TO
34 000352 160300          SUB    R3,R0      ; LEFT OF DELIMITER?
35 000354 020027 000003      CMP    R0,#3      ; DEVICE NAME <= 3 CHARS
36 000360 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 000362 112722 000072  ASSMPL: MOVB   #'',(R2)+    ; PUT COLON AT END OF 2ND NAME
43 000366 105022          CLRB   (R2)+      ; NULL TO TERMINATE STRING
44 000370 010204          MOV    R2,R4      ; MAKE ROOM AT END OF 1ST NAME
45 000372 062704 000002      ADD    #2,R4      ; BY SLIDING CHARS OVER 2 PLACES
46 000376 020201 1$:     CMP    R2,R1      ; WORK BACK DOWN TO '='
47 000400 001402          BEQ    2$          ; BRANCH AFTER '=' MOVED
48 000402 114244          MOVB   -(R2),-(R4) ; SLIDE OVER TO RIGHT
49 000404 000774          BR    1$          ;
50 000406 112722 000072 2$:     MOVB   #'',(R2)+    ; PUT COLON AFTER 1ST DEVICE NAME
51 000412 112712 000101      MOVB   #'A,(R2)   ; PUT IN DUMMY FILE NAME 'A'.
52 000416 105267 000000G      INCB   ASKLNM      ; REMEMBER TO KILL FILE NAME LATER
53 000422 000403          BR    ASCIS      ; GO DO .CSISPC
54          ;
55          ; REQUEST ASSIGNMENT OF LOGICAL DEVICE TO NAMED FILE.
56          ;
57 000424 112722 000072  ASFID: MOVB   #'',(R2)+    ; PUT COLON AT END OF LOGICAL NAME

```

ASSIGN command

```

58 000430 105012          CLRB      (R2)          ;NULL TO END STRING
59
60          ; CALL CSI IN SPECIAL MODE TO PARSE ASSIGN COMMAND.
61
62 000432 010605  ASCIS:  MOV      SP,R5          ;SAVE STACK POINTER
63 000434          .CSISPC #KCSIBF,#ASDEX,R3
64 000450 010506          MOV      R5,SP          ;RESET TOP OF STACK
65 000452 103010          BCC     ASCSOK          ;BRANCH IF .CSISPC OK
66          ; ERROR OCCURED ON .CSISPC
67 000454 113705 000000G  CSIERR: MOVB   @#ERRLOC,R5      ;GET ERROR CODE
68 000460 006305          ASL     R5              ;CONVERT TO WORD INDEX
69 000462 016504 000000G  MOV     KCSIMS(R5),R4      ;GET ADDR OF ERROR MESSAGE
70 000466          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 000474 016700 000000G  ASCSOK: MOV    KCSIBF,R0      ;GET PHYSICAL DEVICE NAME FOR ASSIGN
82 000500 004767 000000G          CALL   ASNSRC          ;SEE IF IT IS ACTUALLY A LOGICAL DEV NAME
83 000504 103403          BCS    3$             ;BR IF NOT
84 000506 016267 000000G 000000G  MOV     AT$DEV(R2),KCSIBF;REPLACE LOGICAL NAME WITH REAL PHYSICAL NAME
85 000514 105767 000000G  3$:   TSTB   ASKLNМ        ;MUST WE KILL FILE NAME?
86 000520 001402          BEQ    1$             ;BRANCH IF NOT
87 000522 005067 000002G  CLR     <KCSIBF+2>       ;KILL THE NAME
88          ; Clear any previous assign with this logical name.
89 000526 016700 000036G  1$:   MOV    KCSIBF+30.,R0    ;GET LOGICAL DEVICE NAME
90 000532 004767 000000G          CALL   ASNSRC          ;SEE IF NAME IS ALREADY ASSIGNED
91 000536 103010          BCC    2$             ;BR IF ASSIGN ALREADY EXISTS (REUSE THIS ENTRY)
92          ; Search for a free assign block.
93 000540 005000          CLR    R0              ;SEARCH FOR FREE ASSIGN BLOCK
94 000542 004767 000000G          CALL   ASNSRC          ;SEE IF NAME IS ALREADY ASSIGNED
95 000546 103004          BCC    2$             ;BR IF FOUND ONE
96          ; ASSIGN TABLE IS FULL.
97 000550          FABORT  #ASNOVF
98
99          ; FOUND A FREE SLOT (POINTED TO BY R2) MOVE ASSIGN
100         ; INFO INTO BLOCK.
101
102 000560 016762 000036G 000000G  2$:   MOV    KCSIBF+30.,AT$LOG(R2);MOVE IN LOGICAL DEVICE NAME
103 000566 016762 000010G 000000G  MOV    KCSIBF+8.,AT$SIZ(R2);MOVE IN FILE LENGTH
104 000574 012703 000000G          MOV    #KCSIBF,R3
105 000600 012362 000000G          MOV    (R3)+,AT$DEV(R2);MOVE IN REAL DEVICE NAME
106 000604 012362 000000G          MOV    (R3)+,AT$FIL(R2);MOVE IN FILE NAME
107 000610 012362 000002G          MOV    (R3)+,AT$FIL+2(R2)
108 000614 011362 000000G          MOV    (R3),AT$EXT(R2) ;MOVE IN FILE EXTENSION
109 000620 000167 000000G          JMP    RDCMD

```

ASSIGN command

```

1          ; -----
2          ; ASDELM IS CALLED TO CHECK IF THE CHARACTER IN R0
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 000624   120027   000075   ASDELM:  CMPB    R0,#'=      ; CHECK EQUAL SIGN
8 000630   001405                BEQ     1$
9 000632   120027   000072       CMPB    R0,#':      ; CHECK COLON
10 000636   001402                BEQ     1$
11 000640   120027   000040       CMPB    R0,#'      ; CHECK SPACE
12 000644   000207   1$:        RETURN      ; RETURN WITH CC SET.

```

DEASSIGN command

```

1          .SBTTL  DEASSIGN command
2          ;-----
3          ;  PROCESS THE DEASSIGN COMMAND
4          ;
5 000646 004767 000000G  CMDDSN: CALL  CVTTAB      ; CONVERT TAB AND FF TO SPACES
6 000652 105713          TSTB   (R3)      ; DEASSIGN ALL OR ONE?
7 000654 001010          BNE    DEAS1     ; BR TO DEASSIGN ONE NAME
8          ;
9          ;  REQUEST TO DEASSIGN ALL LOGICAL NAMES
10         ;  (SIMPLY ZERO THE ASSIGN TABLE)
11         ;
12 000656 012702 000000G          MOV    #ASNTBL,R2    ; POINT TO START OF TABLE
13 000662 005022          1$: CLR    (R2)+
14 000664 020227 000000G          CMP    R2,#ASNEND    ; DONE ALL?
15 000670 103774          BLD    1$      ; BR IF NOT
16 000672 000167 000000G  DAJMP: JMP    RDCMD
17         ;
18         ;  REQUEST TO DEASSIGN A PARTICULAR LOGICAL DEVICE NAME
19         ;
20 000676 004767 000000G  DEAS1: CALL  GTRD50      ; ACCRUE THE DEVICE NAME
21 000702 016700 000000G          MOV    R50BUF,R0    ; PICK UP DEVICE NAME
22 000706 012702 000000G          MOV    #ASNTBL,R2    ; POINT TO START OF ASSIGN TABLE
23 000712 020062 000000G  1$:  CMP    R0,AT$LOG(R2) ; IS THIS ENTRY FOR THE DEVICE?
24 000716 001004          BNE    2$      ; BR IF NOT
25 000720 005062 000000G          CLR    AT$LOG(R2)    ; DEASSIGN THE LOGICAL NAME
26 000724 005062 000000G          CLR    AT$DEV(R2)    ; CLEAR PHYSICAL DEVICE NAME ALSO
27 000730 062702 000000G  2$:  ADD    #AT$$SZ,R2    ; POINT TO NEXT ASSIGN TABLE ENTRY
28 000734 020227 000000G          CMP    R2,#ASNEND    ; REACHED END OF TABLE?
29 000740 103764          BLD    1$      ; BR IF MORE TO DO
30 000742 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 000744  CMDALC:
8          ;
9          ; If no devices were specified, treat this like a SHOW ALLOCATE command
10         ;
11 000744 005067 177032          CLR      ALC1DV          ;Say no allocat device seen yet
12 000750 004767 000000G        CALL     CVTTAB          ;Convert tab and FF chars to spaces
13 000754 004767 000000G        CALL     SKPSPC          ;Skip over spaces
14 000760 105713                TSTB     (R3)          ;Any devices specified?
15 000762 001002                BNE      1$          ;Br if yes
16 000764 000167 000000G        JMP      SOPALC          ;Do SHOW ALLOCATE command
17         ;
18         ; See if we are authorized to allocate devices
19         ;
20 000770 032767 000000G 000000G 1$: BIT      #PO$ALC,PRIVCO ;Are we authorized to allocate devices?
21 000776 001004                BNE      10$          ;Br if yes
22 001000                FABORT  #EM$ALC          ;Not authorized for ALLOCATE
23         ;
24         ; Begin loop to get the name of each device to be allocated
25         ;
26 001010 004767 000000G        10$: CALL     SKPSPC          ;Skip over any spaces
27         ;
28         ; Accrue the next device name
29         ;
30 001014 004767 000000G        CALL     GTRD50          ;Accrue the device name
31 001020 122327 000072          CMPB    (R3)+,#'          ;Was a colon specified with the device name?
32 001024 001401                BEQ      6$          ;Br if yes
33 001026 005303                DEC     R3          ;Backup pointer
34 001030 010346                6$:  MOV     R3,-(SP)          ;Save command pointer
35 001032 004767 000000G        CALL     SKPDLM          ;Skip legal delimiters (blank(s), comma, EOL)
36 001036 012603                MOV     (SP)+,R3          ;Restore command pointer
37 001040 103435                BCS     41$          ;Invalid character following name
38 001042 016702 000000G        MOV     R50BUF,R2          ;Get the name of the device
39 001046 010267 000000G        MOV     R2,ALCDEV          ;Set name of device being allocated
40         ;
41         ; Save the first device name
42         ;
43 001052 005767 176724          TST     ALC1DV          ;Have we seen the first device yet?
44 001056 001002                BNE     2$          ;Br if yes
45 001060 010267 176716          MOV     R2,ALC1DV          ;Save the name of the 1st device
46         ;
47         ; Do the allocation for this device
48         ;
49 001064 012700 000000G        2$:  MOV     #ALDEMT,R0          ;Point to EMT arg block to do the allocation
50 001070 104375                EMT     375          ;Try to allocate the device
51 001072 103051                BCC     3$          ;Br if allocation was successful
52 001074 010005                MOV     R0,R5          ;Save returned error value
53         ;
54         ; An error occurred on the allocation. Print error message.
55         ;
56 001076 113702 000000G        MOVB    @#ERRLOC,R2          ;Get allocation error code
57 001102 120227 000001          CMPB    R2,#1          ;Device already allocated by someone else?

```

ALLOCATE command

```

58 001106 001007          BNE      4$          ;Br if not
59 001110                FERR     #EM$DAA      ;Device allocated by another job
60 001124 000427          BR       7$
61 001126 120227 000002   4$:     CMPB     R2,#2          ;Invalid device being allocated?
62 001132 001004          BNE     5$          ;Br if not
63 001134                FABORT  #EM$IAD      ;Invalid device for allocation
64 001144 120227 000003   5$:     CMPB     R2,#3          ;Allocation table full?
65 001150 001004          BNE     8$          ;Br if not
66 001152                FABORT  #EM$ATF      ;Allocation table full
67 001162 120227 000004   8$:     CMPB     R2,#4          ;Device in use by another user?
68 001166 001013          BNE     3$          ;Br if not
69 001170                FERR     #EM$DIU      ;Device is in use by another user
70 001204 004767 000000G  7$:     CALL     PRTDEC        ;Print the number of the job that has the dev
71 001210                .PRINT  #CRLF        ;End of line
72
73 ; See if there are more devices to be allocated
74
75 001216 004767 000000G  3$:     CALL     SKPSPC        ;Skip over any spaces
76 001222 122327 000054   CMPB     (R3)+,#'          ;Is there another device to allocate?
77 001226 001670          BEQ     10$         ;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 001230 105743          TSTB     -(R3)            ;Was a logical device name specified?
83 001232 001410          BEQ     9$          ;Br if not
84 001234 004767 000000G  CALL     GTRD50           ;Accrue the logical device name
85 001240 016705 000000G  MOV     R50BUF,R5        ;Get the logical device name
86 001244 016700 176532   MOV     ALC1DV,R0        ;Get the physical device name
87 001250 004767 000000G  CALL     DOASGN          ;Do the assignment
88
89 ; Finished
90
91 001254 000167 000000G  9$:     JMP     RDCMD        ;Finished with command

```

DEALLOCATE command

```

1          .SBTTL  DEALLOCATE command
2          ;-----
3          ; Deallocate an allocated device.
4          ;
5 001260 005067 176516          CMDDDL: CLR      ALC1DV      ;Clear deallocated device counter
6 001264 032767 000000G 000000G BIT      #PO$ALC,PRIVCO ;Are we authorized to allocate devices?
7 001272 001004          BNE      1$          ;Br if yes
8 001274          FABORT  #EM$ALC      ;Not authorized to deallocate
9          ;
10         ; See if /ALL was specified
11         ;
12 001304 004767 000000G      1$:      CALL      CVTTAB      ;Convert tab and FF chars to spaces
13 001310 004767 000000G      CALL      CVTUC       ;Convert lower case chars to upper case
14 001314 004767 000000G      CALL      SKSPC       ;Skip over spaces
15         ;
16         ; Find devices for deallocation.
17         ;
18 001320 111300          2$:      MOVVB   (R3),R0      ;Get 1st char of operand
19 001322 001473          BEQ      10$          ;No devices specified, deallocate all
20 001324 120027 000057      3$:      CMPB   R0,#'/'      ;Was a switch specified?
21 001330 001012          BNE      4$          ;Br if not
22 001332 005203          INC      R3          ;Skip over slash
23 001334 004767 000000G      CALL      SKSPC       ;Skip any spaces
24 001340 122327 000101      CMPB   (R3)+,#'A      ;"ALL"?
25 001344 001465          BEQ      12$          ;Br if yes
26 001346          FABORT  #INVOPT      ;Invalid option
27         ;
28         ; Deallocate a list of devices
29         ; Accrue the next device name
30         ;
31 001356 005267 176420      4$:      INC      ALC1DV      ;Count deallocated devices
32 001362 004767 000000G      CALL      SKSPC       ;Skip over any spaces
33 001366 004767 000000G      CALL      GTRD50      ;Accrue the device name
34 001372 122327 000072      CMPB   (R3)+,#':'      ;Was colon specified following name?
35 001376 001401          BEQ      5$          ;Br if yes
36 001400 005303          DEC      R3          ;Backup pointer
37 001402 004767 000000G      5$:      CALL      SKPDLM      ;Skip valid delimiters (blank(s), comma)
38 001406 103435          BCS     7$          ;Br if invalid character
39         ;
40         ; Deallocate the device specified.
41         ;
42 001410 016702 000000G      MOV     R50BUF,R2      ;Get the name of the device
43 001414 010267 000000G      MOV     R2,ALCDEV      ;Set name of device being allocated
44 001420 012700 000000G      MOV     #DLCEMT,R0     ;Point to EMT argument block
45 001424 104375          EMT     375          ;Do the deallocation
46 001426 103334          BCC     2$          ;Br if ok
47 001430 010005          MOV     R0,R5          ;Save job # if owned by someone else
48 001432 113700 000000G      MOVVB  @#ERRLOC,R0     ;Get error code
49 001436 120227 000001      CMPB   R2,#1          ;Device allocated by someone else?
50 001442 001014          BNE     6$          ;Br if not
51 001444          FERR   #EM$DAA      ;Device allocated by another job
52 001460 004767 000000G      CALL   PRTDEC        ;Print the number of the job that has the dev
53 001464          .PRINT #CRLF      ;End of line
54 001472 000712          BR     2$          ;
55 001474 120227 000002      6$:      CMPB   R2,#2          ;Invalid device being allocated?
56 001500 001307          BNE     2$          ;Br if not
57 001502          FABORT  #EM$IAD      ;Invalid device for allocation

```

```
58 ;  
59 ; Deallocate all devices allocated by this user  
60 ;  
61 001512 005767 176264 10#: TST ALC1DV ;Were any devices specified?  
62 001516 001006 BNE 20# ;Yes, end of command  
63 001520 005067 0000000 12#: CLR ALCDEV ;Say to deallocate all devices  
64 001524 012700 0000000 MOV #DLCEMT,RO ;Point to EMT argument block  
65 001530 104375 EMT 375 ;Do the deallocation  
66 001532 000400 BR 20#  
67 ;  
68 ; Finished  
69 ;  
70 001534 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 001540 004767 000000G  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 001544 004767 000000G  1$:   CALL  SKPSPC          ;Skip past leading spaces
13 001550 105713          ;TSTB   (R3)          ;Are we at the end of the command?
14 001552 001436          ;BEQ    9$          ;Br if yes
15         ;
16         ; Accrue the name of a device or region
17         ;
18 001554 004767 000000G          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 001560 005767 000002G          TST    R50BUF+2        ;Is name longer than 3 characters?
24 001564 001012          ;BNE    2$          ;Br if yes
25 001566 121327 000072          ;CMPB  (R3),#':'      ;Is name terminated with colon?
26 001572 001002          ;BNE    3$          ;Br if not
27 001574 105723          ;TSTB  (R3)+         ;Skip over colon
28 001576 000407          ;BR     5$          ;Must have been a device name
29 001600 016705 000000G  3$:   MOV    R50BUF,R5    ;Get 1st 3 chars of name
30 001604 004767 000000G          ;CALL  CHKDEV        ;See if this is the name of a device
31 001610 103002          ;BCC   5$          ;Br if device name
32         ;
33         ; This is a request to remove a named PLAS region.
34         ;
35 001612 004767 000036          2$:   CALL  REMRGN        ;Remove a region
36         ;
37         ; Skip over any separating comma
38         ;
39 001616 122327 000054          5$:   CMPB  (R3)+,#','    ;Is there a separating comma?
40 001622 001401          ;BEQ   4$          ;Br if yes
41 001624 005303          ;DEC   R3          ;Point back to skipped char
42 001626 105713          4$:   TSTB  (R3)          ;Reached end of command?
43 001630 001407          ;BEQ   9$          ;Br if yes
44 001632 121327 000040          ;CMPB  (R3),# '      ;Space separator?
45 001636 001742          ;BEQ   1$          ;Br if yes
46 001640          ;FABORT #EM#CSE      ;Command syntax error
47         ;
48         ; Finished
49         ;
50 001650 000167 000000G  9$:   JMP    RDCMD

```

```

1 ; -----
2 ; Eliminate a named PLAS region.
3 ;
4 ; Inputs:
5 ; R50BUF = Rad50 name of the region.
6 ;
7 001654 010246 REMRGN: MOV R2, -(SP)
8 ;
9 ; First attempt to attach to a private region with that name.
10 ;
11 001656 012702 000020' MOV #REMRDB, R2 ; Point to region definition block
12 001662 005062 000000G CLR R.GSIZ(R2) ; Clear size word
13 001666 012762 000000C 000000G MOV #<RS.GBL!RS.PVT>, R.GSTS(R2) ; Function = Attach private region
14 001674 016762 000000G 000000G MOV R50BUF, R.NAME(R2) ; Set name of region
15 001702 016762 000002G 000002G MOV R50BUF+2, R.NAME+2(R2)
16 001710 .CRRG #XAREA, R2 ; Attempt to attach to region
17 001726 032762 000000G 000000G BIT #RS.CRR, R.GSTS(R2) ; Were we able to attach?
18 001734 001025 BNE 1$ ; Br if yes
19 ;
20 ; Unable to attach to private region.
21 ; Try to attach to public region.
22 ;
23 001736 012762 000000G 000000G MOV #RS.GBL, R.GSTS(R2) ; Function = Attach global region
24 001744 .CRRG #XAREA, R2 ; Attempt to attach to it
25 001762 032762 000000G 000000G BIT #RS.CRR, R.GSTS(R2) ; Was attachment successful?
26 001770 001007 BNE 1$ ; Br if yes
27 ;
28 ; Error -- We are unable to attach to region
29 ;
30 001772 FERR #EM$UAR ; Unable to attach to region
31 002006 000421 BR 9$
32 ;
33 ; We successfully attached to the region.
34 ; Try to eliminate the region.
35 ;
36 002010 012762 000000G 000000G 1$: MOV #RS.EGR, R.GSTS(R2) ; Function = Eliminate region
37 002016 .ELRG #XAREA, R2 ; Try to eliminate the region
38 002034 103006 BCC 9$ ; Br if successful
39 ;
40 ; Error -- Unable to eliminate the region
41 ;
42 002036 FERR #EM$UER ; Unable to eliminate region
43 ;
44 ; Finished
45 ;
46 002052 012602 9$: MOV (SP)+, R2
47 002054 000207 RETURN

```

INSTALL Command

```

1          .SBTTL  INSTALL Command
2          ;-----
3          ; Process the INSTALL command.
4          ;
5 002056 004767 000000G  CMDINS: CALL  CVTTAB      ;Convert tabs to spaces
6 002062 004767 000000G          CALL  CVTUC      ;Convert to upper case
7          ;
8          ; Accrue first keyword and see if it is ADD or DELETE
9          ;
10         MOV     R3,R2      ;Save pointer to start of keyword
11 002070 004767 000000G  CALL  @TRD50      ;Accrue 1st keyword
12 002074 016700 000000G  MOV   R50BUF,R0      ;Get 1st 3 chars of keyword
13 002100 020067 175726   CMP   R0,R50ADD      ;Is keyword ADD?
14 002104 001421          BEQ   INSADD      ;Br if yes
15 002106 020067 175722   CMP   R0,R50DEL     ;Is keyword DELETE?
16 002112 001473          BEQ   INSDEL     ;Br if yes
17 002114 020067 175720   CMP   R0,R50REM     ;Is keyword REMOVE?
18 002120 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 002122 010300          MOV   R3,R0      ;Get pointer past end of keyword
24 002124 160200          SUB   R2,R0      ;Determine length of 1st keyword
25 002126 020027 000002   CMP   R0,#2      ;1st keyword longer than 2 chars?
26 002132 101404          BLOS  1$      ;Br if not
27 002134          FABORT #ILLCMD      ;Invalid command
28         ;
29         ; This must be a device handler name -- Ignore this command
30         ;
31 002144 000167 000000G  1$:  JMP   RDCMD      ;Finished installing device handler

```

```

1 ; -----
2 ; We are adding an image
3 ;
4 002150 004767 000000G INSADD: CALL CKSYPV ;Require SYSPRV privilege
5 002154 005067 175662 CLR IATTRB ;Clear all attribute flags
6 002160 004767 000000G CALL CLRPRV ;Reset privilege flag cells
7 002164 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 002170 004767 000144 CALL INSNAM ;Get name of image and look it up
12 002174 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 002176 004767 000450 CALL INSFRE ;Find free table entry
18 ;
19 ; Set up program name for free entry
20 ;
21 002202 012702 000000C 1$: MOV #II$NAM+IIBUF,R2;Point to target buffer
22 002206 012704 000000G MOV #FILNAM,R4 ;Point to name
23 002212 012700 000004 MOV #4,R0 ;Move 4 words
24 002216 012422 2$: MOV (R4)+,(R2)+ ;Move name to buffer
25 002220 077002 SOB R0,2$
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 002222 004767 000510 CALL INSOPT ;Process any options that follow name
31 ;
32 ; Move attribute and privilege flags into install entry
33 ;
34 002226 010546 MOV R5,-(SP) ;Save address of entry in install table
35 002230 016767 175606 000001C MOV IATTRB,II$FLG+IIBUF ;Set attribute flags for program
36 002236 012702 000000G MOV #PFSO,R2 ;Point to accrued privilege flags
37 002242 012703 000000C MOV #II$PRV+IIBUF,R3;Install entry flags
38 002246 012704 000000G MOV #PFCO,R4 ;Flags to clear when run
39 002252 012705 000000C MOV #II$NPV+IIBUF,R5;Install entry
40 002256 012700 000000G MOV #PVNPW,R0 ;Get # words to move
41 002262 012223 3$: MOV (R2)+,(R3)+ ;Move flags to be set
42 002264 012425 MOV (R4)+,(R5)+ ;Move flags to be cleared
43 002266 077003 SOB R0,3$
44 002270 012605 MOV (SP)+,R5 ;Recover address of entry in install table
45 ;
46 ; Add the entry to the install table
47 ;
48 002272 004767 000424 CALL INSPUT ;Add entry to table
49 ;
50 ; Finished
51 ;
52 002276 000167 000000G JMP RDCMD

```



INSTALL Command

```

1          ;
2          ; Process INSTALL DELETE command
3          ;
4 002302 004767 000000G  INSDEL: CALL  CKSYPV          ;Require SYSPRV privilege
5          ;
6          ; See if we can find program in install table
7          ;
8 002306 004767 000026   CALL  INSNAM          ;Accrue file spec and try to find in table
9 002312 103004         BCC  1$              ;Br if found file name in table
10 002314              FABORT #EM$FNI        ;File not installed
11         ;
12         ; Found entry, delete it
13         ;
14 002324 005067 000001C 1$: CLR  II$NAM+IIBUF      ;Say image is not installed
15 002330 004767 000366   CALL  INSPUT          ;Write entry back to table
16         ;
17         ; Finished
18         ;
19 002334 000167 000000G          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 | 002340 | 010246 |               | INSNAM: MOV | R2,-(SP)   |                                      |
| 14 | 002342 | 010446 |               |             | MOV        | R4,-(SP)                             |
| 18 | 002344 | 005002 |               | CLR         | R2         | ; Assume device name is not wild     |
| 19 | 002346 | 004767 | 000000G       | CALL        | SKPSPC     | ; Skip up to start of file spec      |
| 20 | 002352 | 121327 | 000052        | CMPB        | (R3),#'*   | ; Wildcard as device name?           |
| 21 | 002356 | 001013 |               | BNE         | 11\$       | ; Br if not                          |
| 22 | 002360 | 126327 | 000001 000072 | CMPB        | 1(R3),#'   | ; Is this the device name?           |
| 23 | 002366 | 001007 |               | BNE         | 11\$       | ; Br if not                          |
| 24 | 002370 | 012702 | 132500        | MOV         | #WLDNAM,R2 | ; Remember device name is wild       |
| 25 | 002374 | 112713 | 000040        | MOVB        | #',(R3)    | ; Blank out the device name          |
| 26 | 002400 | 112763 | 000040 000001 | MOVB        | #',1(R3)   |                                      |
| 30 | 002406 | 012704 | 000000G       | 11\$: MOV   | #R5OSAV,R4 | ; Get default file extension (SAV)   |
| 31 | 002412 | 005005 |               | CLR         | R5         | ; Say this is an input file          |
| 32 | 002414 | 004767 | 000000G       | CALL        | ACRFIL     | ; Accrue the file spec               |
| 33 | 002420 | 103510 |               | BCS         | 10\$       | ; Br if invalid file spec            |
| 34 | 002422 | 005767 | 000002G       | TST         | FILNAM+2   | ; Was a file name specified?         |
| 35 | 002426 | 001505 |               | BEQ         | 10\$       | ; Br if not                          |
| 36 | 002430 | 005702 |               | TST         | R2         | ; Is device name wild?               |
| 37 | 002432 | 001403 |               | BEQ         | 12\$       | ; Br if not                          |
| 38 | 002434 | 010267 | 000000G       | MOV         | R2,FILNAM  | ; Set wildcard as file name          |
| 39 | 002440 | 000431 |               | BR          | 5\$        |                                      |
| 43 | 002442 | 012705 | 000000G       | 12\$: MOV   | #FILNAM,R5 | ; Point to file spec                 |
| 44 | 002446 | 004767 | 000000G       | CALL        | LOGASN     | ; Do logical assignment              |
| 45 | 002452 | 010346 |               | MOV         | R3,-(SP)   | ; SAVE POINTER                       |
| 46 | 002454 | 010503 |               | MOV         | R5,R3      | ; GET COPY OF DEVICE POINTER         |
| 47 | 002456 | 004767 | 000000G       | CALL        | FORCE0     | ; MAKE BLANK UNIT = 0                |
| 48 | 002462 | 012603 |               | MOV         | (SP)+,R3   | ; RESTORE POINTER                    |
| 52 | 002464 | 016705 | 000000G       | MOV         | FILNAM,R5  | ; Get the device name                |
| 53 | 002470 | 004767 | 000000G       | CALL        | CHKDEV     | ; Convert to device index number     |
| 54 | 002474 | 103407 |               | BCS         | 6\$        | ; Br if invalid device               |
| 55 | 002476 | 020467 | 000000G       | CMP         | R4,LDDEVX  | ; Is this a logical disk?            |
| 56 | 002502 | 001010 |               | BNE         | 5\$        | ; Br if not                          |
| 57 | 002504 |        |               | FABORT      | #EM\$LDI   | ; Can't have installed program on LD |

INSTALL Command

```

58 002514          6$:      FABORT  #EM$IAD          ;Invalid device
59                ;
60                ; Try to find existing entry for file in install table
61                ;
62 002524 016705 000000G 5$:      MOV      INSTBL,R5          ;Point to 1st entry in table
63 002530 010567 000000G 1$:      MOV      R5,INGADR          ;Set address in EMT
64 002534 012700 000000G          MOV      #INGEMT,R0          ;Point to get EMT
65 002540 104375          EMT      375              ;Get next install table entry
66 002542 005760 000000C          TST      IIBUF+II$NAM(RO);Is this entry empty?
67 002546 001422          BEQ      2$              ;Br if yes
68 002550 012700 000006          MOV      #6,R0              ;Get index to last word of name
69 002554 026027 000000G 132500 3$:    CMP      FILNAM(RO),#WLDNAM ;Wildcard?
70 002562 001410          BEQ      13$             ;Br if yes
71 002564 026027 000000C 132500          CMP      IIBUF+II$NAM(RO),#WLDNAM ;Wildcard?
72 002572 001404          BEQ      13$             ;Br if yes
73 002574 026060 000000G 000000C          CMP      FILNAM(RO),IIBUF+II$NAM(RO) ;Compare names
74 002602 001004          BNE      2$              ;Br if this is not right entry
75 002604 162700 000002          13$:    SUB      #2,R0              ;More words to check?
76 002610 002361          BGE      3$              ;Br if yes
77 002612 000407          BR       4$              ;Found entry
78 002614 062705 000000G 2$:      ADD      #II$$SZ,R5          ;Point to next entry in table
79 002620 020567 000000G          CMP      R5,INSTBN          ;Reached end of table?
80 002624 103741          BLD      1$              ;Loop if not
81                ;
82                ; Cannot find entry in table
83                ;
84 002626 000261          SEC                      ;Signal failure on return
85 002630 000401          BR       9$
86                ;
87                ; We found entry for this name
88                ;
89 002632 000241          4$:      CLC                      ;Signal success on return
90                ;
91                ; Finished
92                ;
93 002634 012604          9$:      MOV      (SP)+,R4
94 002636 012602          MOV      (SP)+,R2
95 002640 000207          RETURN
96                ;
97                ; Invalid file spec
98                ;
99 002642          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 002652 016705 000000G  INSFRE: MOV     INSTBL,R5      ;Point to 1st install table entry
8 002656 010567 000000G  1$:  MOV     R5,INGADR      ;Set address for EMT
9 002662 012700 000000G      MOV     #INGEMT,R0      ;EMT to get assign table entry
10 002666 104375          EMT     375              ;Get next entry
11 002670 005767 000001C      TST     II$NAM+IIBUF     ;Is this entry free?
12 002674 001411          BEQ     2$              ;Br if found free entry
13 002676 062705 000000G      ADD     #II$SZ,R5       ;Point to next entry
14 002702 020567 000000G      CMP     R5,INSTBN      ;Reached end of table?
15 002706 103763          BLO     1$              ;Loop if not
16          ;
17          ; No free table entries
18          ;
19 002710          FABORT  #EM$ITF      ;Install table full
20          ;
21          ; Found free entry
22          ;
23 002720 000207          2$:  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 002722 010567 000000G  INSPUT: MOV      R5, INPADR      ;Set address for EMT
8 002726 012700 000000G      MOV      #INPEMT, R0      ;Point to EMT
9 002732 104375              EMT      375          ;Store install table entry
10 002734 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         002736 010446 INSOPT: MOV      R4, -(SP)
11         ;
12         ; Process list of options
13         ;
14         002740 012704 000000'      MOV      #INSHD, R4      ;Point to table of options
15         002744 004767 000000G      CALL     OPTLST        ;Process an option list
16         ;
17         ; Finished
18         ;
19         002750 012604      MOV      (SP)+, R4
20         002752 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         002754 051467 175062 INSATR: BIS      (R4), IATTRB    ;Set attribute flag
32         002760 000207      RETURN    ;Finished

```

MOUNT command

```

1          .SBTTL  MOUNT command
2          ;-----
3          ; Mount a file structure.
4          ; This command has two possible forms:
5          ;   MOUNT dev -- Simply enable directory caching
6          ;   MOUNT[/[NO]WRITE] LDn file-spec [logical-name] -- Mount logical disk
7          ;
8 002762 004767 000000G CMDMNT: CALL  CVTTAB          ; CONVERT TAB AND FF CHARS TO SPACES
9          ;
10         ; Determine if a command switch was specified
11         ;
12 002766 105067 000000G          CLRB  NOFLAG          ; ASSUME /NOWRITE NOT WANTED
13 002772 122327 000040          1$:  CMPB  (R3)+, #'          ; SKIP ANY LEADING SPACES
14 002776 001775                BEQ   1$
15 003000 124327 000057          CMPB  -(R3), #'/'          ; WAS A SWITCH SPECIFIED?
16 003004 001014                BNE   3$
17 003006 005203                INC   R3
18 003010 012704 000160'        MOV   #MTOPHD, R4        ; POINT TO TABLE OF SWITCHES
19 003014 004767 000000G        CALL  SEARCH          ; LOOK UP THE SWITCH
20 003020 103004                BCC   2$
21 003022                FABORT #INVOPT          ; INVALID SWITCH
22 003032 111467 000000G        2$:  MOVB  (R4), NOFLAG        ; REMEMBER IF NOWRITE SEEN
23         ;
24         ; Get device name
25         ;
26 003036 105713                3$:  TSTB  (R3)
27 003040 001004                BNE   12$
28 003042                FABORT #ILLCMD          ; INVALID COMMAND
29 003052 004767 000000G        12$: CALL  @TRD50          ; GET DEVICE NAME
30 003056 016702 000000G        MOV   R50BUF, R2        ; PICK UP DEVICE NAME
31 003062 020267 000000G        CMP   R2, R5OLD        ; IS NAME "LD"?
32 003066 001002                BNE   15$
33 003070 016702 000000G        MOV   R5OLD0, R2       ; TRANSLATE LD TO LDO
34 003074 010267 000000G        15$: MOV   R2, MNTDEV        ; SAVE DEVICE NAME
35 003100 122327 000072        CMPB  (R3)+, #'          ; WAS DEVICE NAME TERMINATED WITH COLON?
36 003104 001401                BEQ   4$
37 003106 005303                DEC   R3
38         ;
39         ; See if device we are mounting is allocated to another user
40         ;
41 003110 016767 000000G 000000G 4$:  MOV   MNTDEV, ALCDEV        ; Set device name for allocation EMT
42 003116 012700 000000G        MOV   #TALEMT, R0        ; Point to EMT argument block
43 003122 104375                EMT   375
44 003124 103022                BCC   5$
45 003126 123727 000000G 000001  CMPB  @#ERRLOC, #1        ; Device allocated to another user?
46 003134 001016                BNE   5$
47 003136 010005                MOV   R0, R5
48 003140                FERR  #EM#DAA        ; Device allocated to job
49 003154 004767 000000G        CALL  PRTDEC          ; Print # of job that owns device
50 003160                .PRINT #CRLF        ; End print line
51 003166 000167 000000G        JMP   RDCMD          ; Abort command
52         ;
53         ; See if there is a file name specified.
54         ;
55 003172 004767 000000G        5$:  CALL  SKPSPC          ; Skip any spaces
56 003176 105713                TSTB  (R3)
57 003200 001002                BNE   MNTLD          ; BR IF YES

```

58 003202 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 003206   105767   000000G   MNTLD:  TSTB    VLDSYS           ;WAS LD SUPPORT GENNED IN?
7 003212   001004                   BNE      18$           ;BR IF YES
8 003214                   FABORT  #EM$NLD       ;LD SUPPORT NOT GENNED IN
9          ;
10         ; Make sure the logical disk name is ok.
11        ;
12 003224   010200   18$:      MOV      R2,R0           ;GET LD DEVICE NAME
13 003226   004767   000412      CALL    LDDEVN        ;GET LD DEVICE INDEX NUMBER
14 003232   103004                   BCC     6$           ;BR IF OK
15 003234                   FABORT  #INVLDM       ;ERROR IF NAME IS NOT LD
16 003244   010002   6$:      MOV      R0,R2           ;CARRY INDEX IN R2
17        ;
18        ; Close the log file if it is on the logical disk
19        ;
20 003246   016705   000000G      MOV     MNTDEV,R5      ;GET NAME OF LOGICAL DEVICE
21 003252   004767   000000G      CALL    LOGCHK        ;SEE IF WE NEED TO CLOSE THE LOG FILE
22        ;
23        ; Tell system to stop doing directory caching for the device
24        ;
25 003256   112767   000001 000000G  MOVVB  #1,SERFLG      ;DO .SERR TO AVOID ABORT FOR ILLEGAL DEVICE
26 003264   012700   000000G      MOV     #DMTARG,R0    ;POINT TO EMT ARGUMENT BLOCK
27 003270   104375                   EMT     375           ;TELL SYSTEM TO STOP DOING CACHING
28 003272   105067   000000G      CLRB   SERFLG        ;DO .HERR
29        ;
30        ; Accrue the file name
31        ;
32 003276   012704   000000G      MOV     #R5ODSK,R4    ;SET DEFAULT FILE EXTENSION
33 003302   005005                   CLR     R5           ;SAY THIS IS AN INPUT FILE
34 003304   004767   000000G      CALL    ACRFIL        ;ACCRUE THE FILE SPEC
35 003310   103004                   BCC    13$          ;BR IF OK
36 003312   5$:      FABORT  #BDFNAM       ;INVALID FILE SPEC
37        ;
38        ; Translate logical file device name to physical device
39        ;
40 003322   012705   000000G   13$:    MOV     #FILNAM,R5  ;POINT TO FILE SPEC AREA
41 003326   004767   000000G      CALL    LOGASN        ;PERFORM ANY LOGICAL DEVICE ASSIGNMENT
42        ;
43        ; Make sure file is not on same LD as is being mounted
44        ;
45 003332   016700   000000G      MOV     FILNAM,R0     ;GET DEVICE THAT FILE IS ON
46 003336   004767   000302      CALL    LDDEVN        ;SEE IF FILE IS ON A LD DEVICE
47 003342   103406                   BCS    17$          ;BR IF NOT ON LD DEVICE
48 003344   020002                   CMP     R0,R2        ;IS IT ON LOWER # LD?
49 003346   103404                   BLD    17$          ;BR IF YES
50 003350                   FABORT  #INVLDM       ;INVALID LD ORDER
51        ;
52        ; Make sure the file name is not null
53        ;
54 003360   005767   000002G   17$:    TST     FILNAM+2      ;Is file name null?
55 003364   001752                   BEQ    5$           ;Br if yes -- Error
56        ;
57        ; Move file name into logical disk name table

```

MOUNT command

```

58
59 003366 010205          MOV      R2,R5          ;GET LOGICAL DISK UNIT #
60 003370 006305          ASL      R5              ;*4 WORDS PER ENTRY
61 003372 006305          ASL      R5
62 003374 062705 000000G  ADD      #LDNAME,R5     ;POINT TO ENTRY IN LOGICAL DISK TABLE
63 003400 012704 000000G  MOV      #FILNAM,R4     ;POINT TO FILE NAME BUFFER
64 003404 012425          MOV      (R4)+,(R5)+    ;MOVE NAME TO TABLE
65 003406 012425          MOV      (R4)+,(R5)+
66 003410 012425          MOV      (R4)+,(R5)+
67 003412 011415          MOV      (R4),(R5)
68
69          ; Set up flags for logical disk
70
71 003414 005004          CLR      R4
72 003416 105767 000000G  TSTB    NOFLAG          ;WAS NOWRITE SPECIFIED?
73 003422 001402          BEQ     9$              ;BR IF NOT
74 003424 052704 000000G  BIS     #LD$RON,R4      ;SET READ-ONLY FLAG
75 003430 010462 000000G  9$:     MOV      R4,LDFLAG(R2)
76
77          ; Lookup file and set up information about position of logical
78          ; disk on physical disk.
79
80 003434 004767 000000G  CALL    LDMNT           ;SET UP INFO ABOUT FILE FOR LOGICAL DISK
81 003440 103004          BCC     10$             ;BR IF FOUND FILE
82 003442          FABORT  #CSIMS4     ;CAN'T FIND FILE
83
84          ; Protect the file
85
86 003452 010205 10$:    MOV      R2,R5          ;GET LOGICAL DISK UNIT #
87 003454 006305          ASL      R5              ;* 4 WORDS PER ENTRY
88 003456 006305          ASL      R5
89 003460 062705 000000G  ADD      #LDNAME,R5     ;POINT TO ENTRY WITH FILE NAME
90 003464 112767 000001 000000G  MOVB    #1,SERFLG       ;DO .SERR TO AVOID ABORTS ON ERRORS
91 003472          .FPROT #XAREA,#1,R5,#1
92 003516 105067 000000G  CLRB    SERFLG          ;DO .HERR
93
94          ; See if a logical device name was specified following the file name
95
96 003522 122327 000040 16$:    CMPB    (R3)+,#'        ;SKIP BLANKS
97 003526 001775          BEQ     16$
98 003530 105743          TSTB    -(R3)           ;WAS A LOGICAL DEVICE NAME SPECIFIED?
99 003532 001410          BEQ     MNTCSH         ;BR IF NOT
100
101          ; Assign logical name to logical disk
102
103 003534 004767 000000G  CALL    GTRD50          ;ACCRUE LOGICAL DEVICE NAME
104 003540 016705 000000G  MOV     R50BUF,R5       ;GET LOGICAL DEVICE NAME
105 003544 016700 000000G  MOV     MNTDEV,R0        ;GET PHYSICAL DEVICE NAME
106 003550 004767 000000G  CALL    DOASGN          ;DO THE ASSIGNMENT
107
108          ; Initiate directory caching for the device whose name is in MNTDEV
109
110 003554 012700 000000G  MNTCSH: MOV    #MNTARG,R0 ;POINT TO MOUNT ARGUMENT BLOCK
111 003560 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 003564 004767 000000G ; CALL LDCLN ; DO SET LD CLEAN  
117 ;  
118 ; Finished with MOUNT command  
119 ;  
120 003570 000167 000000G ; JMP RDCMD
```

MOUNT command

```

1
2 ; -----
3 ; DOMNT is called to execute a mount or ,dismount EMT and check for
4 ; errors. If an error is detected, an error message is generated.
5 ;
6 ; Inputs:
7 ; RO = Address of EMT argument block for mount or dismount emt.
8 003574 104375 DOMNT: EMT 375 ;DO THE EMT (MOUNT OR DISMOUNT)
9 003576 103021 ; BCC 9# ;BR IF NO ERROR
10 ;
11 ; Error on mount or dismount emt.
12 ;
13 003600 123727 0000000 000001 CMPB @#ERRLOC,#1 ; MOUNT TABLE OVERFLOW?
14 003606 001007 BNE 2# ;BR IF NOT
15 003610 FWARN #MNTFUL ; MOUNT TABLE OVERFLOW
16 003624 000406 BR 9#
17 003626 2#: FERR #BDFNAM ; INVALID DEVICE NAME
18 ;
19 ; Finished
20 ;
21 003642 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 003644 020067 0000000 LDDEVN: CMP RO,R5OLD ;Is name "LD"?
36 003650 001002 BNE 1# ;Br if not
37 003652 016700 0000000 MOV R5OLD0,RO ;Replace "LD" by "LDO"
38 003656 020067 0000000 1#: CMP RO,R5OLD0 ;Is name in the range LDO to LD7?
39 003662 103410 BLD 2# ;Br if not
40 003664 020067 0000000 CMP RO,R5OLD7
41 003670 101005 BHI 2#
42 003672 166700 0000000 SUB R5OLD0,RO ;Get unit number only
43 003676 006300 ASL RO ;Convert to device index number
44 003700 000241 CLC ;Signal success on return
45 003702 000401 BR 3#
46 003704 000261 2#: SEC ;Signal failure on return
47 003706 000207 3#: RETURN

```

```

1          .SBTTL  DISMOUNT  command
2          ;-----
3          ; Dismount a file structure.
4          ;
5 003710   004767   000000G  CMDDMT: CALL    CVTTAB          ; CONVERT TAB AND FF CHARS TO SPACES
6          ;
7          ; See if any qualifier were specified following command keyword
8          ;
9 003714   105067   174124    CLRB    DMTNLG          ; Clear flag set if /NOLOG specified
10 003720   004767   000000G  CALL    SKPSPC         ; Skip over any spaces
11 003724   121327   000057    CMPB   (R3),#'/        ; Any qualifiers here?
12 003730   001004                BNE    5$              ; Br if not
13          ;
14          ; Process qualifier following command keyword
15          ;
16 003732   012704   000174'   MOV    #DMTQHD,R4      ; Point to parsing table
17 003736   004767   000000G  CALL    OPTLST         ; Process any qualifiers
18          ;
19          ; Skip leading spaces and make sure a device name was specified
20          ;
21 003742   004767   000000G  5$:   CALL    SKPSPC         ; SKIP LEADING SPACES
22 003746   105713                TSTB   (R3)            ; WAS A DEVICE NAME SPECIFIED
23 003750   001004                BNE    6$              ; BR IF YES
24 003752                7$:   FABORT  #ILLCMD          ; INVALID COMMAND IF NO DEVICE NAME
25          ;
26          ; Get name of device that is being dismounted
27          ;
28 003762   004767   000000G  6$:   CALL    @TRD50        ; ACCRUE THE DEVICE NAME
29 003766   016700   000000G  MOV    R50BUF,R0      ; GET DEVICE NAME
30 003772   001767                BEQ    7$              ; BR IF NO DEVICE NAME
31 003774   004767   000000G  2$:   CALL    ASNSRC        ; SEE IF IT IS A LOGICAL NAME
32 004000   103403                BCS    1$              ; BR IF NOT
33 004002   016200   000000G  MOV    AT$DEV(R2),R0  ; GET ASSIGNED NAME
34 004006   000772                BR     2$              ; ALLOW INDIRECT ASSIGNS
35 004010   020067   000000G  1$:   CMP    R0,R50LD     ; IS NAME "LD"?
36 004014   001002                BNE    4$              ; BR IF NOT
37 004016   016700   000000G  MOV    R50LD0,R0     ; TRANSLATE LD TO LDO
38 004022   010067   000000G  4$:   MOV    R0,MNTDEV    ; SAVE NAME OF DEVICE BEING DISMOUNTED
39          ;
40          ; Close the log file if it is on the device being dismounted
41          ;
42 004026   016705   000000G  MOV    MNTDEV,R5      ; GET NAME OF DEVICE BEING DISMOUNTED
43 004032   004767   000000G  CALL    LOGCHK        ; SEE IF LOG FILE IS ON THAT DEVICE
44          ;
45          ; Print information message if device is still mounted by other jobs
46          ;
47 004036   105767   174002    TSTB   DMTNLG         ; Was /NOLOG qualifier specified?
48 004042   001013                BNE    3$              ; Br if yes -- Suppress information message
49 004044   016705   000000G  MOV    MNTDEV,R5      ; GET NAME OF DEVICE BEING DISMOUNTED
50 004050   004767   000000G  CALL    CHKMNT        ; IS THIS DEVICE MOUNTED?
51 004054   103406                BCS    3$              ; BR IF NOT
52 004056   004767   000000G  CALL    CHKMTX        ; IS DEVICE MOUNTED BY ANYONE OTHER THAN US?
53 004062   103003                BCC    3$              ; BR IF NOT
54 004064                .PRINT  #INFDMT      ; STILL MOUNTED BY OTHER USERS
55          ;
56          ; Tell system to stop doing directory caching for the device
57          ;

```

DISMOUNT command

```

58 004072 112767 000001 000000G 3#:      MOVB    #1,SERFLG      ;DO .SERR TO AVDID ABORT FOR ILLEGAL DEVICE
59 004100 012700 000000G                MOV     #DMTARG,R0    ;POINT TO EMT ARGUMENT BLOCK
60 004104 104375                EMT     375           ;TELL SYSTEM TO STOP DOING CACHING
61 004106 105067 000000G                CLRB    SERFLG       ;DO .HERR
62                                     ;
63                                     ;   See if a logical disk is being dismantled
64                                     ;
65 004112 016702 000000G                MOV     MNTDEV,R2    ;GET DEVICE NAME
66 004116 020267 000000G                CMP     R2,R5OLD0    ;IS THIS A LOGICAL DISK?
67 004122 103420                BLD     9#           ;BR IF NOT
68 004124 020267 000000G                CMP     R2,R5OLD7
69 004130 101015                BHI     9#
70 004132 166702 000000G                SUB     R5OLD0,R2    ;GET UNIT #
71 004136 006302                ASL     R2           ;CONVERT TO WORD INDEX
72 004140 005062 000000G                CLR     LDPDEV(R2)  ;NO PHYSICAL DEVICE ASSIGNMENT
73 004144 006302                ASL     R2           ;GET INDEX INTO LDNAME TABLE
74 004146 006302                ASL     R2
75 004150 005062 000000G                CLR     LDNAME(R2)  ;NO FILE NAME
76 004154 016700 000000G                MOV     MNTDEV,R0    ;GET NAME OF LOGICAL DISK
77 004160 004767 000000G                CALL    DEADEV      ;DEASSIGN ANY LOGICAL NAMES
78                                     ;
79                                     ;   Finished
80                                     ;
81 004164 004767 000000G 9#:      CALL    LDCLN      ;RESET INFORMATION ABOUT LOGICAL DISKS
82 004170 000167 000000G                JMP     RDCMD       ;FINISHED WITH COMMAND
83                                     ;
84                                     ;   Process /NOLOG qualifier for DISMOUNT command
85                                     ;
86 004174 105267 173644                DMTNLQ: INCB    DMTNLG ;Remember qualifier specified
87 004200 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 004202   CMDINI:
9 004202   CMDSQZ:
10 004202  CMDFMT:
11          ;
12          ; See if user is authorized to use these commands
13          ;
14 004202 032767 000000C 000000G      BIT      #PO$NFW!PO$BYP,PRIVCO ;Authorized for these commands?
15 004210 001004                      BNE      13$                ;Br if yes
16 004212                      FABORT  #EM$NFW                ;Not authorized for this command
17          ;
18          ; Skip over any options specified with command keyword
19          ;
20 004222 004767 000000G      13$:    CALL      CVTTAB                ;Convert tabs to spaces
21 004226 121327 000057      CMPB     (R3),#/'          ;Were any keyword options specified?
22 004232 001005                      BNE      10$                ;Br if not
23 004234 112300      1$:    MOVB     (R3)+,R0                ;Get next character from command
24 004236 001407                      BEQ      15$                ;Br if no device was specified
25 004240 120027 000040      CMPB     R0,#40                ;Is this a space?
26 004244 001373                      BNE      1$                  ;Keep looking if not
27          ;
28          ; Accrue the device name
29          ;
30 004246 004767 000000G      10$:    CALL      SKSPSPC                ;Skip over any spaces
31 004252 105713                      TSTB     (R3)                ;Was a device name specified?
32 004254 001004                      BNE      14$                ;Br if yes
33 004256      15$:    FABORT  #EM$DNR                ;Device name is required
34 004266 004767 000000G      14$:    CALL      GTRD50                ;Accrue the device name
35 004272 016705 000000G      MOV      R50BUF,R5
36 004276 010567 000000G      MOV      R5,MNTDEV          ;Set name for mount/dismount
37 004302 010567 173472      MOV      R5,SQZDEV          ;Save name of device being squeezed
38 004306 010567 000000G      MOV      R5,ALCDEV          ;Set name for test-allocation EMT
39          ;
40          ; Disallow SQUEEZE/INITIALIZE/FORMAT of any disk with a system file
41          ;
42 004312 004767 000232      CALL      CKSYDV                ;See if device has any system files
43 004316 103004                      BCC      11$                ;Br if not
44 004320                      FABORT  #EM$SSY                ;Can't do this to SY
45          ;
46          ; See if this device is mounted by anyone
47          ;
48 004330 004767 000000G      11$:    CALL      CHKMNT                ;See if device is mounted by anyone
49 004334 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 004336 004767 000000G      CALL      CHKMTX                ;Is device mounted by any other users?
55 004342 103004                      BCC      2$                  ;Br if not
56 004344      3$:    FABORT  #OTRMNT                ;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 004354 012700 000000G 2$: MOV #MNTARG,R0 ;Remount the device
62 004360 104375 EMT 375
63 ;
64 ; See if this device is allocated to any other user
65 ;
66 004362 120467 000000G 8$: CMPB R4,LDDEVX ;Is this a logical disk?
67 004366 001404 BEQ 12$ ;Br if yes -- Don't check for allocation
68 004370 016700 000000G MOV ALCDEV,R0 ;Get RAD50 device name
69 004374 004767 000000G 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 004400 016705 173374 12$: MOV SQZDEV,R5 ;Get back device name
75 004404 004767 000000G CALL CHKDEV ;Convert device name to dev # & unit #
76 004410 020467 000000G CMP R4,LDDEVX ;Is this a logical disk?
77 004414 001406 BEQ 4$ ;Br if yes
78 004416 005002 CLR R2 ;Say base block # = 0
79 004420 012703 177777 MOV #177777,R3 ;Say top block = 177777
80 004424 000300 SWAB R0 ;Put unit # in high byte
81 004426 050004 BIS R0,R4 ;Get unit # and device # together
82 004430 000410 BR 5$
83 004432 006300 4$: ASL R0 ;Convert unit # to word table index
84 004434 016004 000000G MOV LDPDEV(R0),R4 ;Get physical dev # and unit #
85 004440 016002 000000G MOV LDBASE(R0),R2 ;Get base block of logical disk
86 004444 010203 MOV R2,R3
87 004446 066003 000000G ADD LDSIZE(R0),R3 ;Get top block number of logical disk
88 ;
89 ; Search for any mounted logical disks that are on the disk being squeezed
90 ;
91 004452 016705 000000G 5$: MOV CSHDEV,R5 ;Point to start of mount table
92 004456 010500 7$: MOV R5,R0 ;Get address of mount entry
93 004460 004767 000000G CALL CDGET ;Copy mount entry into CDBUF
94 004464 020467 000001C CMP R4,CDBUF+CD$DVU ;Is this device on same physical device?
95 004470 001014 BNE 6$ ;Br if not
96 004472 016700 000001C MOV CDBUF+CD$BAS,R0 ;Is this a logical disk?
97 004476 001411 BEQ 6$ ;Br if not
98 004500 020002 CMP R0,R2 ;See if logical disk is within disk being sqze
99 004502 101407 BLOS 6$ ;Br if not
100 004504 020003 CMP R0,R3 ;Compare upper range
101 004506 103005 BHIS 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 004510 004767 000000G CALL CHKMTX ;Anyone else have this logical disk mounted?
107 004514 103713 BCS 3$ ;Error if yes
108 ;
109 ; Dismount this logical disk and any files associated with it
110 ;
111 004516 004767 000000G CALL DMTSUB ;Do the dismount
112 ;
113 ; Check for more nested logical disks
114 ;

```



FORMAT command

```

115 004522 062705 000000G      6#:      ADD      #CD##SZ,R5      ;Point to next mount table entry
116 004526 020567 000000G      CMP      R5,CSHDVN      ;Reached end of table?
117 004532 103751                BLD      7#              ;Loop if not
118                                ;
119                                ; See if log file is opened to device being squeezed
120                                ; If so, close it.
121                                ;
122 004534 016705 173240      MOV      SQZDEV,R5      ;Get name of device being squeezed
123 004540 004767 000000G      CALL     LOGCHK        ;See if log file is on that device
124                                ;
125                                ; Call CCL to process the command
126                                ;
127 004544 000167 000000G      7#:      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 004550 010246
12 004552 010346
13 004554 010446
14 004556 010546
15
16 ; Convert device name to device index and unit number
17
18 004560 004767 000000G
19 004564 103422
20
21 ; At this point: R0 = unit number, R4 = device index number.
22 ; Begin loop to check each system file savestatus block.
23
24 004566 012705 004646'
25 004572 012503
26 004574 001416
27 004576 016302 000000G
28 004602 042702 177701
29 004606 020204
30 004610 001370
31 004612 116302 000000G
32 004616 042702 177770
33 004622 020200
34 004624 001362
35
36 ; This device contains a system file
37
38 004626 000261
39 004630 000401
40
41 ; This device does not contain a system file
42
43 004632 000241
44
45 ; Finished
46
47 004634 012605
48 004636 012604
49 004640 012603
50 004642 012602
51 004644 000207
52
53 ; List of addresses of savestatus blocks for system files
54
55 004646 000000G
56 004650 000000G
57 004652 000000G

```

```

;-----
; Check to see if a device being squeezed or initialized contains any
; system files.
;
; Inputs:
; R5 = Rad50 device name of device being squeezed or initialized
;
; Outputs:
; C-flag set ==> This device contains system files
CKSYDV: MOV R2, -(SP)
        MOV R3, -(SP)
        MOV R4, -(SP)
        MOV R5, -(SP)
;
; Convert device name to device index and unit number
;
        CALL CHKDEV ; Convert name to device index & unit #
        BCS B$ ; Br if device name not recognized
;
; At this point: R0 = unit number, R4 = device index number.
; Begin loop to check each system file savestatus block.
;
        MOV #SYFLVC, R5 ; Point to vector of savestatus addresses
1$: MOV (R5)+, R3 ; Point to next savestatus block
    BEQ B$ ; Br if reached end of list
    MOV C.CSW(R3), R2 ; Get 1st word of channel
    BIC #^C<76>, R2 ; Extract device index number
    CMP R2, R4 ; Does it match device being squeezed?
    BNE 1$ ; Br if not
    MOVB C.DEVQ(R3), R2 ; Get unit number
    BIC #^C<7>, R2 ; Extract unit number
    CMP R2, R0 ; Same as unit of device being squeezed?
    BNE 1$ ; Br if not
;
; This device contains a system file
;
        SEC ; Signal error on return
        BR 9$
;
; This device does not contain a system file
;
8$: CLC ; Signal no error
;
; Finished
;
9$: MOV (SP)+, R5
    MOV (SP)+, R4
    MOV (SP)+, R3
    MOV (SP)+, R2
    RETURN
;
; List of addresses of savestatus blocks for system files
;
SYFLVC: .WORD SWPCHN ; Swap file
        .WORD SEGCHN ; PLAS swap file
        .WORD SPLCHN ; Spool file

```

FORMAT command

|    |        |         |        |        |               |
|----|--------|---------|--------|--------|---------------|
| 58 | 004654 | 000000G | . WORD | KMNCHN | ; TSKMON. SAV |
| 59 | 004656 | 000000G | . WORD | CCLSAV | ; CCL. SAV    |
| 60 | 004660 | 000000G | . WORD | INDSAV | ; IND. SAV    |
| 61 | 004662 | 000000G | . WORD | INDTSV | ; TSXIND. TSX |
| 62 | 004664 | 000000  | . 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 004666 005067 000000G CMDMON: CLR      MNFLGS      ; INITIALIZE EMT ARG BLOCK
8 004672 005067 000000G      CLR      MNBPC
9 004676 004767 000000G      CALL     CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
10 004702 004767 000000G      CALL     DELSPC     ; DELETE SPACES FROM COMMAND LINE
11 004706 004767 000000G      CALL     ACROCT     ; ACCRUE BASE ADDRESS
12 004712 010167 000000G      MOV      R1, MNBASE  ; SET IN EMT ARG BLOCK
13 004716 122327 000054      CMPB    (R3)+, #' ,  ; COMMA SHOULD BE DELIMITER
14 004722 001075      BNE      3$
15 004724 004767 000000G      CALL     ACROCT     ; ACCRUE TOP ADDRESS
16 004730 010167 000000G      MOV      R1, MNTOP
17 004734 112300      MOVB    (R3)+, R0    ; GET DELIMITER
18 004736 001430      BEQ      1$          ; BR IF FINISHED WITH COMMAND
19 004740 120027 000054      CMPB    R0, #' / ,  ; DID HE SPECIFY NBPC?
20 004744 001006      BNE      2$          ; BR IF NOT
21 004746 004767 000000G      CALL     ACRDEC     ; ACCRUE NUMBER OF BYTES PER CELL
22 004752 010167 000000G      MOV      R1, MNBPC
23 004756 112300      4$:  MOVB    (R3)+, R0    ; GET DELIMITER
24 004760 001417      BEQ      1$          ; BR IF END OF COMMAND HIT
25 004762 120027 000057      2$:  CMPB    R0, #' /  ; GOT A SWITCH?
26 004766 001053      BNE      3$          ; BR IF NOT
27 004770 012704 000234'      MOV      #MONHD, R4  ; POINT TO TABLE OF SWITCH OPTIONS
28 004774 004767 000000G      CALL     SEARCH     ; SEARCH FOR SWITCH IN TABLE
29 005000 103004      BCC      5$          ; BR IF FOUND IT
30 005002      FABORT   #INVOPT  ; INVALID SWITCH
31 005012 051467 000000G      5$:  BIS      (R4), MNFLGS  ; SET BITS FOR SWITCH
32 005016 000757      BR      4$          ; GO CHECK FOR MORE SWITCHES
33                                     ;
34                                     ; Finished scanning command.
35                                     ; Initiate performance monitoring.
36                                     ;
37 005020 012700 000000G      1$:  MOV      #MONAR1, R0  ; POINT TO EMT ARG BLOCK
38 005024 104375      EMT      375          ; TRY TO INITIATE MONITORING
39 005026 103026      BCC      6$          ; BR IF OK
40 005030 105737 000000G      TSTB   @#ERRLOC     ; CHECK ERROR CODE
41 005034 001404      BEQ      7$          ; BR IF SOMEONE ELSE DOING MONITORING NOW
42 005036      FABORT   #NOPMGN  ; PM NOT GENNED IN
43 005046 010046      7$:  MOV      R0, -(SP)    ; SAVE # OF PM USER
44 005050      FERR     #PMBUSY  ; PM FACILITY IN USE BY ANOTHER USER
45 005064 012605      MOV      (SP)+, R5    ; GET # OF PM USER
46 005066 004767 000000G      CALL     PRTDEC     ; PRINT DECIMAL VALUE
47 005072      .PRINT  #CRLF    ; TERMINATE THE ERROR LINE
48 005100 000167 000000G      JMP      RDCMD
49 005104 012700 000000G      6$:  MOV      #MONAR2, R0  ; START MONITORING
50 005110 104375      EMT      375
51 005112 000167 000000G      JMP      RDCMD
52                                     ;
53                                     ; Invalid command syntax.
54                                     ;
55 005116      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 005126 004767 000000G  CMDSPD: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACE
9 005132 004767 000000G          CALL  CVTUC      ; Convert lower case chars to upper case
10 005136 105713          TSTB   (R3)        ; ANY DEVICE NAMED?
11 005140 001002          BNE    1$         ; BRANCH IF YES
12 005142 000167 000000G          JMP    BADCMD     ; MUST HAVE DEVICE NAME
13          ;
14          ; ACCRUE DEVICE NAME IN RAD50 FORM.
15          ;
16 005146 004767 000000G  1$:   CALL  QTRD50      ; ACCRUE THE NAME
17          ;
18          ; If device name is a logical name, translate to physical
19          ;
20 005152 016700 000000G          MOV    R50BUF,R0      ; Get device name
21 005156 004767 000000G          CALL  ASNSRC     ; See if this name is assigned
22 005162 103403          BCS   3$         ; Br if name is not assigned
23 005164 016267 000000G 000000G  MOV    AT$DEV(R2),R50BUF ; Set physical device name
24          ;
25          ; NOW TRY TO FIND SDCB WHICH MATCHES NAME.
26          ;
27 005172 010346          3$:   MOV    R3, -(SP)      ; SAVE POINTER
28 005174 012703 000000G          MOV    #R50BUF,R3     ; POINT TO DEVICE NAME
29 005200 004767 000000G          CALL  FORCE0      ; MAKE UNIT 0 IF NEEDED
30 005204 012603          MOV    (SP)+,R3     ; RESTORE POINTER
31 005206 012701 000000G          MOV    #SDCB,R1     ; POINT TO 1ST SDCB
32 005212 020127 000000G          4$:   CMP    R1,#SDCBND ; CHECKED ALL?
33 005216 103013          BHS   2$         ; BRANCH IF YES
34 005220 026127 000000G 000000G  CMP    SDNAME(R1),#DMYDEV ; Is this dummy device entry?
35 005226 001404          BEQ   5$         ; Br if yes
36 005230 026761 000000G 000000G  CMP    R50BUF,SDNAME(R1) ; DO NAMES MATCH?
37 005236 001407          BEQ   CSGETOP     ; BRANCH IF FOUND SDCB
38 005240 062701 000000G          5$:   ADD    #SDCBSZ,R1 ; GO CHECK NEXT SDCB
39 005244 000762          BR    4$
40          ;
41          ; CAN'T FIND SDCB FOR DEVICE.
42          ;
43 005246          2$:   FABORT #INVDEV
44          ;
45          ; FOUND SDCB (R1).
46          ; IDENTIFY OPTION.
47          ;
48 005256 112300          CSGETOP: MOVB   (R3)+,R0 ; SKIP OVER ANY DELIMITERS
49 005260 001424          BEQ   3$         ; Br at end of command string
50 005262 004767 000000G          CALL  CHKDLM
51 005266 103773          BCS   CSGETOP
52 005270 005303          DEC   R3
53 005272 012704 000260'          MOV    #SPLHD,R4 ; POINT TO TABLE OF SPOOL OPTIONS
54 005276 004767 000000G          CALL  SEARCH     ; LOOK UP OPTION
55 005302 103401          BCS   1$         ; BR IF ERROR
56          ;
57          ; FOUND OPTION. BRANCH OFF TO OPTION PROCESSING ROUTINE.

```

```
58 ; SDCB ADDRESS IS IN R1.  
59 ;  
60 005304 000134 ; JMP @(R4)+  
61 ;  
62 ; ERROR IN SEARCH  
63 ;  
64 005306 005704 1#: TST R4 ; ILLEGAL OR AMBIGUOUS  
65 005310 001404 BEQ 2# ; BR IF ILLEGAL  
66 005312 FABORT #AMBOPT ; AMBIGUOUS OPTION  
67 005322 2#: FABORT #INVOPT ; ILLEGAL OPTION  
68 ;  
69 ; Reached end of command string (usually when handling multiple options)  
70 ;  
71 005332 000167 0000006 3#: JMP RDCMD
```

SPOOL command

```

1          ;.....
2          ;
3          ; PROCESS THE 'DEL' (DELETE) OPTION.
4          ;
5 005336   SPLDEL:
6          ;
7          ; See if a file ID was specified
8          ;
9 005336   105713          TSTB      (R3)          ;Was an id specified?
10 005340   001425          BEQ       SPDLCR         ;Br if not
11 005342   121327   000054  CMPB      (R3),#',      ;Comma used as delimiter?
12 005346   001403          BEQ       1$          ;Br if yes
13 005350   121327   000075  CMPB      (R3),#'=     ;Equal sign used as delimiter?
14 005354   001001          BNE      2$          ;Br if not -- space must have been delimiter
15 005356   005203  1$:     INC       R3          ;Skip over comma or equal sign
16          ;
17          ; A file id was specified, accrue the id value
18          ;
19 005360   004767   000000G  2$:     CALL      ACRDEC         ;Accrue the value
20 005364   010167   172462          MOV      R1,DELEMT+4   ;Set file ID in EMT arg block
21 005370   012700   000046'  MOV      #DELEMT,R0    ;Point to EMT arg block
22 005374   104375          EMT      375           ;Try to delete the file
23 005376   103004          BCC      9$          ;Br if successful
24 005400          FABORT   #EM$NID      ;Cannot find file with that id
25 005410   000167   000000G  9$:     JMP       RDCMD
26          ;
27          ; Abort the file currently being printed by the spooled device
28          ;
29 005414   005761   000000G  SPDLCR: TST      SDSFCB(R1) ; IS DEVICE BUSY?
30 005420   001405          BEQ      SPNBSY        ; BRANCH IF NOT
31 005422   152761   000000G 000000G  BISB     #SD$DEL,SDFLAG(R1) ; SET DELETE FLAG
32 005430   000167   000000G  CSABT:  JMP      RDCMD
33 005434          SPNBSY: .PRINT  #DEVIDL
34 005442   000772          BR       CSABT

```

SPOOL command

```

1          ; -----
2          ;   PROCESS THE SPOOL 'ALIGN' OPTION.
3          ;   (SPOOL XX,ALIGN,<FILE NAME>)
4          ;
5 005444   004767   000000G   SPLALN: CALL    CKPRIV      ;Must have operator privilege
6 005450   122327   000054     .CMPB    (R3)+,#'      ;WAS FILE NAME GIVEN?
7 005454   001404     .BEQ    1$            ;BR IF YES
8 005456   9$:      .FABORT   #COAL
9          ;   PARSE ALIGNMENT FILE NAME
10 005466   010605   1$:      .MOV     SP,R5      ;SAVE SP
11 005470     .CSIGEN #ALDEX,#ALDEX,R3;OPEN ALIGNEMENT FILE - CHAN 3
12 005504   010506     .MOV    R5,SP        ;RESET SP
13 005506   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 005510   016167   000000G   000000G   .MOV     SDNAME(R1),ALDBLK;SET UP DEVICE NAME
18 005516     .ENTER  #XAREA,#1,#ALDBLK,#0;OPEN SPOOLED DEVICE
19 005542   103010     .BCC    2$          ;BR IF OPEN OK
20 005544     .PRINT  #COAD      ;CAN'T OPEN SPOOLED DEVICE
21 005552     .CLOSE  #3        ;CLOSE ALIGNMENT FILE
22 005560   000167   000000G   10$:     .JMP    RDCMD
23          ;
24          ;   WRITE 1ST RECORD TO SPOOLED FILE AND SPECIFY A
25          ;   FORM NAME OF "*****" WHICH MEANS ALIGNMENT FILE.
26          ;
27 005564   2$:     .WRITW  #XAREA,#1,#ALFN,#4,#0
28          ;
29          ;   NOW COPY ALIGNMENT FILE TO SPOOLED FILE
30          ;
31 005622   005002     .CLR    R2          ;INIT BLOCK #
32 005624   4$:     .READW  #XAREA,#3,#BLKO,#256.,R2
33 005662   103421     .BCS    3$          ;BR IF END OF FILE
34 005664     .WRITW  #XAREA,#1,#BLKO,#256.,R2
35 005722   005202     .INC    R2
36 005724   000737     .BR     4$
37          ;   HIT END OF ALIGNMENT FILE.   CLOSE BOTH CHANNELS.
38 005726   3$:     .CLOSE  #1
39 005734     .CLOSE  #3
40 005742   000706     .BR     10$
41          ;
42          ;   PROCESS THE 'LOCK' OPTION.
43          ;
44 005744   004767   000000G   SPLLK:  CALL    CKPRIV      ;Must have operator privilege
45 005750   052761   000000G   000000G   .BIS    #SD$FLK,SDFLAG(R1);SAY FORM IS LOCKED
46 005756   000405     .BR     SPLMT1      ;NOW DO MOUNT
47          ;
48          ;   PROCESS THE 'FORM' OPTION
49          ;
50 005760   004767   000000G   SPLFRM: CALL    CKPRIV      ;Must have operator privilege
51 005764   042761   000000G   000000G   .BIC    #SD$FLK,SDFLAG(R1);SAY FORM IS NOT LOCKED
52 005772   010102   SPLMT1: .MOV    R1,R2      ;POINT TO FORM NAME CELL
53 005774   062702   000000G     .ADD    #SDFORM,R2
54 006000   010204     .MOV    R2,R4        ;POINT TO END OF CELL
55 006002   062704   000006     .ADD    #6,R4
56 006006   005203     .INC    R3          ;SKIP OVER COMMA
57 006010   105713   3$:     .TSTB   (R3)      ;HIT END OF NAME?

```



SPOOL command

```

58 006012 001404          BEQ      1$          ; BR IF YES
59 006014 020204          CMP      R2,R4        ; COPIED 6 CHARS?
60 006016 103007          BHIS     2$          ; BR IF YES
61 006020 112322          MOVVB   (R3)+,(R2)+   ; COPY IN FORM NAME
62 006022 000772          BR       3$
63                          ; PAD END OF SHORT NAME WITH BLANKS
64 006024 020204          1$:    CMP      R2,R4        ; FILLED TO 6 CHARS?
65 006026 103003          BHIS     2$          ; BR IF YES
66 006030 112722 000040   MOVVB   #' ,(R2)+   ; PUT IN TRAILING SPACES
67 006034 000773          BR       1$
68                          ;
69                          ; NOW TRY TO START SPOOLER
70                          ;
71 006036 042761 000000G 000000G 2$:    BIC      #SD$WFM,SDFLAG(R1); SAY FORM MOUNT IS DONE
72 006044 010100          MOV      R1,R0        ; POINT TO SDCB
73 006046 004767 000630   CALL    SPOLGO        ; TRY TO START SPOOLER
74 006052 000167 000000G   JMP      RDCMD
75                          ;
76                          ; PROCESS THE 'SKIP' OPTION
77                          ;
78 006056 004767 000000G   SPLSKP: CALL    CKPRIV        ; Must have operator privilege
79 006062 005761 000000G   TST     SDSFCB(R1)    ; IS DEVICE BUSY?
80 006066 001415          BEQ      SPNBB        ; BR IF NOT
81                          ; ACCRUE NUMBER
82 006070 010104          MOV      R1,R4        ; SAVE SDCB ADDRESS
83 006072 005203          INC      R3          ; SKIP OVER COMMA
84 006074 004767 000000G   CALL    ACRDEC        ; ACCRUE DEC NUMBER
85 006100 010164 000000G   MOV      R1,SDSKIP(R4) ; SET SKIP COUNT IN SDCB
86 006104 000167 000000G   JMP      RDCMD
87                          ;
88                          ; PROCESS THE 'BACK' OPTION
89                          ;
90 006110 004767 000000G   SPLBAK: CALL    CKPRIV        ; Must have operator privilege
91 006114 005761 000000G   TST     SDSFCB(R1)    ; IS THE DEVICE BUSY
92 006120 001002          BNE     SPLBK1        ; BR IF YES
93 006122 000167 177306   SPNBB:  JMP      SPNBSY        ; GO GIVE ERROR MESSAGE
94 006126 005761 000000G   SPLBK1: TST     SDBU(R1)    ; ANY BLOCKS REMEMBERED YET?
95 006132 001403          BEQ     1$          ; BR IF NOTHING TO BACKUP TO
96 006134 052761 000000G 000000G   BIS     #SD$BAK,SDFLAG(R1); REQUEST BACK UP
97 006142 000167 000000G   1$:    JMP      RDCMD

```

SPOOL command

```

1          ;
2          ; PROCESS THE SPOOL 'STATUS' COMMAND
3          ;
4 006146 005761 000000G SPLSTA: TST SDSFCB(R1) ; IS SPOOLER BUSY?
5 006152 001404 BEQ 1$ ; BR IF NOT
6 006154 .PRINT #SPACTV
7 006162 000413 BR 4$
8 006164 032761 000000G 000000G 1$: BIT #SD$WFM, SDFLAG(R1); IS IT WAITING FOR A FORM MOUNT?
9 006172 001404 BEQ 2$
10 006174 .PRINT #SPWFM
11 006202 000403 BR 4$
12 006204 2$: .PRINT #DEVIDL
13          ; SEE IF IN SINGLE FILE MODE
14 006212 032761 000000G 000000G 4$: BIT #SD$SNG, SDFLAG(R1); IN SINGLE-FILE MODE?
15 006220 001403 BEQ 20$ ; BR IF NOT
16 006222 .PRINT #SPSNG
17          ; SEE IF SPOOL FILE IS FULL
18 006230 005767 000000G 20$: TST NFRESB ; IS SPOOL FILE FULL?
19 006234 001003 BNE 3$ ; BR IF NOT
20 006236 .PRINT #SPFUL
21          ; LIST CURRENT FORM NAME
22 006244 3$: .PRINT #SPCF
23 006252 010102 MOV R1, R2 ; POINT TO FORM NAME CELL
24 006254 062702 000000G ADD #SDFORM, R2
25 006260 012703 000000G MOV #6, R3 ; PRINT 6 CHARS
26 006264 112200 5$: MOVB (R2)+, R0
27 006266 .TTYOUT
28 006272 005303 DEC R3
29 006274 001373 BNE 5$
30          ; SEE IF CURRENT FORM IS LOCKED
31 006276 032761 000000G 000000G BIT #SD$FLK, SDFLAG(R1); IS FORM LOCKED?
32 006304 001403 BEQ 21$ ; BR IF NOT
33 006306 .PRINT #SPFLK
34          ; SEE IF IN FLAG PAGE MODE
35 006314 112700 000054 21$: MOVB #' , R0 ; PRINT COMMA
36 006320 .TTYOUT
37 006324 032761 000000G 000000G BIT #SD$FLG, SDFLAG(R1); IN FLAG PAGE MODE?
38 006332 001004 BNE 6$ ; BR IF YES
39 006334 .PRINT #NSPFLG ; "NOFLAGPAGE"
40 006342 000416 BR 7$
41 006344 6$: .PRINT #SPFLG ; "FLAGPAGE, "
42 006352 032761 000000G 000000G BIT #SD$WID, SDFLAG(R1); IS FLAG PAGE WIDE?
43 006360 001004 BNE 61$ ; BR IF YES
44 006362 .PRINT #SPNARD ; "NARROW"
45 006370 000403 BR 7$
46 006372 61$: .PRINT #SPWIDE ; "WIDE"
47          ; List info about files pending for this device.
48 006400 005761 000000G 7$: TST SDFHD(R1) ; ARE THERE ANY FILES PENDING FOR THIS DEVICE?
49 006404 001004 BNE 11$ ; BR IF YES
50 006406 .PRINT #NOFIL ; NO FILES...
51 006414 000410 BR 8$
52 006416 11$: .PRINT #QHDMS1 ; PRINT HEADING MESSAGE
53 006424 .PRINT #QHDMS2 ; UNDERLINE THE HEADING
54 006432 004767 000000G CALL LSTSPL ; LIST SPOOL FILES
55 006436 000167 000000G 8$: JMP RDCMD
56          ;
57          ; PROCESS THE 'SINGLE' OPTION

```

SPOOL command

```

58 ;
59 006442 004767 000000G SPLSNG: CALL CKPRIV ;Must have operator privilege
60 006446 052761 000000G 000000G BIS #SD$SNG,SDFLAG(R1);SET SINGLE-FILE MODE
61 006454 000167 000000G JMP RDCMD
62 ;
63 ; PROCESS THE 'MULTIPLE' OPTION
64 ;
65 006460 004767 000000G SPLMUL: CALL CKPRIV ;Must have operator privilege
66 006464 042761 000000G 000000G BIC #SD$SNG,SDFLAG(R1);RESET SINGLE-FILE MODE
67 006472 000167 000000G JMP RDCMD
68 ;
69 ; PROCESS THE SPOOL 'HOLD' COMMAND.
70 ; HOLD MEANS DON'T START OUTPUT UNTIL CHANNEL IS CLOSED.
71 ;
72 006476 004767 000000G SPLHLD: CALL CKPRIV ;Must have operator privilege
73 006502 010167 171352 MOV R1,SPHLEM+4 ;Set SDCB address in EMT arg block
74 006506 012700 000054' MOV #SPHLEM,R0 ;Point to EMT arg block
75 006512 104375 EMT 375 ;Set hold mode
76 006514 000167 000000G JMP RDCMD
77 ;
78 ; PROCESS THE SPOOL 'NOHOLD' COMMAND WHICH MEANS START OUTPUT
79 ; AS SOON AS A FILE IS CREATED.
80 ;
81 006520 004767 000000G SPLNHL: CALL CKPRIV ;Must have operator privilege
82 006524 010167 171336 MOV R1,SPNHEM+4 ;Set SDCB address in EMT arg block
83 006530 012700 000062' MOV #SPNHEM,R0 ;Point to EMT arg block
84 006534 104375 EMT 375 ;Set NOHOLD mode
85 006536 000167 000000G JMP RDCMD
86 ;
87 ; PROCESS THE 'FLAGPAGE' OPTION
88 ;
89 006542 004767 000000G SPLFLG: CALL CKPRIV ;Must have operator privilege
90 006546 052761 000000G 000000G BIS #SD$FLG,SDFLAG(R1);SET FLAG MODE
91 006554 000167 176476 JMP CSGTOP ;Go check for other options
92 ;
93 ; PROCESS THE 'NOFLAGPAGE' OPTION
94 ;
95 006560 004767 000000G SPLNFL: CALL CKPRIV ;Must have operator privilege
96 006564 042761 000000G 000000G BIC #SD$FLG,SDFLAG(R1);RESET FLAG MODE
97 006572 000167 176460 JMP CSGTOP ;Go check for other options
98 ;
99 ; PROCESS THE 'WIDE' OPTION
100 ;
101 006576 004767 000000G SPLWID: CALL CKPRIV ;Must have operator privilege
102 006602 052761 000000G 000000G BIS #SD$WID,SDFLAG(R1);SET FLAG MODE
103 006610 000167 176442 JMP CSGTOP ;Go check for other options
104 ;
105 ; PROCESS THE 'NARROW' OPTION
106 ;
107 006614 004767 000000G SPLNRW: CALL CKPRIV ;Must have operator privilege
108 006620 042761 000000G 000000G BIC #SD$WID,SDFLAG(R1);RESET FLAG MODE
109 006626 000167 176424 JMP CSGTOP ;Go check for other options
110 ;
111 ;-----
112 ; PROCESS THE 'FORM' COMMAND.
113 ;
114 006632 004767 000000G CMDFRM: CALL CVTTAB ;CONVERT TAB AND FF CHARS TO SPACE

```

SPOOL command

```

115 006636 004767 000000G          CALL    CVTUC          ;CONVERT ALL CHARS TO UPPER CASE
116 006642 012704 000000G          MOV     #UFORM,R4      ;POINT TO CELL WHICH HOLDS FORM NAME
117 006646 020302                   2$:    CMP     R3,R2      ;MOVED ALL OF NAME?
118 006650 103004                   BHS    1$              ;BR IF YES
119 006652 112324                   MOVB   (R3)+,(R4)+     ;MOVE FORM NAME TO UFORM
120 006654 020427 000006G          CMP     R4,#<UFORM+6> ;MOVED 6 CHARS?
121 006660 103772                   BLO    2$              ;BR IF NOT
122                                     ; SEE IF WE NEED TO PAD NAME WITH BLANKS
123 006662 020427 000006G          1$:    CMP     R4,#<UFORM+6> ;FILLED TO 6 CHARS?
124 006666 103003                   BHS    3$              ;BR IF FINISHED
125 006670 112724 000040                   MOVB   #' ,(R4)+      ;PUT IN TRAILING BLANKS
126 006674 000772                   BR     1$
127 006676 000167 000000G          3$:    JMP     RDCMD
128
129                                     ;-----
130                                     ; SPOLGO is called to start a spooler.
131
132                                     ; Inputs:
133                                     ; RO = SDCB address of spooler to be started.
134
135 006702 010046                   SPOLGO: MOV    RO,-(SP)
136 006704 010067 000004G          MOV    RO,SPGEMT+4    ;SET SDCB ADDRESS IN EMT ARG BLOCK
137 006710 012700 000000G          MOV    #SPGEMT,RO     ;POINT TO ARG BLOCK
138 006714 104375                   EMT    375             ;START THE SPOOLER
139 006716 012600                   MOV    (SP)+,RO
140 006720 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 006722 004767 000000G CMDYEL: CALL  CKPRIV      ;Require OPER privilege
7 006726 112767 000003 000000G      MOVB   #3,YELEM      ;Set the YELL and error flags
8 006734 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 006736 116701 000000G OPRCMD: MOVB   CTRLTT,R1      ;GET LINE INDEX # OF OPR TERMINAL
15 006742 001014          BNE    SNDGN      ;BR IF GOT ONE
16 006744          FABORT  #NOOPTT      ;NO OPERATOR'S CONSOLE
17
18        ;-----
19        ; PROCESS THE 'SEND' COMMAND.
20        ;
21 006754 105067 000000G CMDSND: CLRB   YELEM      ;This is not the YELL command
22 006760 111300          SNDCOM: MOVB   (R3),R0      ;GET 1ST CHAR OF COMMAND
23 006762 001420          BEQ    CSEXIT      ;BRANCH IF NULL COMMAND
24
25        ; ACCRUE LINE #.
26        ;
27 006764 004767 000544          CALL   SNDLIN      ;Accrue line # to send to
28 006770 005701          TST    R1          ;SEND TO SPECIFIC USER?
29 006772 003422          BLE    CSSCAN      ;BRANCH IF ALL OR DC
30 006774 020127 000000G SNDCGN: CMP    R1,#LSTSL      ;IS THIS A VALID LINE #?
31 007000 101003          BHI    LNUMER      ;BR IF NOT
32 007002 016101 000000G          MOV    LNPRIM(R1),R1      ;GET PRIMARY LINE #
33 007006 001010          BNE    CSCD      ;BRANCH IF VALID
34 007010          LNUMER: FWARN  #BDLIN      ;Invalid line number
35 007024 000167 000000G          CSEXIT: JMP   RDCMD
36 007030 032761 000000G 000000G CSCD:  BIT   #$DETC,LSW(R1) ;TRYING TO SEND TO DETACHED JOB?
37 007036 001364          BNE    LNUMER      ;BR IF YES
38
39        ; AT THIS POINT THE INDEX NUMBER OF THE LINE WE WISH TO
40        ; SEND TO IS IN R1.
41        ; PUT OUR LINE NUMBER AND USER NAME AT START OF MESSAGE BUFFER
42        ;
43 007040 032767 000000G 000000G CSSCAN: BIT   #PO$SND,PRIVCO ;Are we privileged to send messages?
44 007046 001004          BNE    10$          ;Br if yes
45 007050          FABORT  #EM$SND
46 007060 012704 000000G 10$:  MOV    #MSGBUF,R4      ;POINT TO MESSAGE BUFFER
47 007064 112724 000015          MOVB   #CR,(R4)+      ;PUT IN LEADING CR-LF-BELL
48 007070 112724 000012          MOVB   #LF,(R4)+
49 007074 112724 000007          MOVB   #BELL,(R4)+
50 007100 116705 000000G          MOVB   CORUSR,R5      ;GET OUR LINE NUMBER
51 007104 016505 000000G          MOV    LNPRIM(R5),R5      ;GET OUR PRIMARY LINE NUMBER
52 007110 010502          MOV    R5,R2
53 007112 062702 000000G          ADD    #NUMTB1,R2      ;POINT TO TABLE WITH LINE NUMBER CHARACTERS
54 007116 112224          MOVB   (R2)+,(R4)+      ;PUT IN LINE NUMBER
55 007120 112224          MOVB   (R2)+,(R4)+
56 007122 070527 000006          MUL   #6,R5          ;EACH JOB HAS A 12 CHAR USER NAME
57 007126 062705 000000G          ADD    #LUNAME,R5      ;POINT TO USER NAME ENTRY FOR THIS LINE

```

SEND command

```

58 007132 121527 000040      CMPB    (R5),#'      ; IS USER NAME BLANK?
59 007136 001416              BEQ     4$           ; BR IF YES
60 007140 112724 000040      MOVB   #' ,(R4)+    ; PUT IN A SPACE
61 007144 112724 000050      MOVB   #'(,(R4)+    ; AND OPEN PAREN AT START OF NAME
62 007150 012702 000014      MOV    #12,R2      ; GET LENGTH OF NAME
63 007154 112524              6$:    MOVB   (R5)+,(R4)+ ; GET NEXT CHAR OF USER NAME
64 007156 077202              SOB    R2,6$       ; MOVE REST OF NAME
65 007160 124427 000040      7$:    CMPB   -(R4),#BLANK ; TRIM OFF TRAILING SPACES
66 007164 001775              BEQ    7$
67 007166 005204              INC    R4           ; POINT BEYOND LAST NON-BLANK CHARACTER
68 007170 112724 000051      5$:    MOVB   #'),(R4)+ ; PUT IN CLOSE PAREN AT END OF NAME
69 007174 112724 000040      4$:    MOVB   #' ,(R4)+ ; PUT " -- " AT END OF NAME
70 007200 112724 000055      MOVB   #'-, (R4)+
71 007204 112724 000055      MOVB   #'-, (R4)+
72 007210 112724 000040      MOVB   #' ,(R4)+
73                               ;
74                               ; Now move message string to message buffer
75                               ;
76 007214 122327 000040      3$:    CMPB   (R3)+, #' ; SKIP OVER ANY BLANKS
77 007220 001775              BEQ    3$
78 007222 005303              DEC    R3           ; POINT TO 1ST CHAR OF MESSAGE
79 007224 112324              1$:    MOVB   (R3)+,(R4)+ ; MOVE MESSAGE TO OUR BUFFER
80 007226 001403              BEQ    2$           ; BRANCH WHEN END HIT
81 007230 020427 177776G     CMP    R4,#<MSGEND-2> ; DON'T OVERFLOW OUR BUFFER
82 007234 103773              BLD    1$
83 007236 005304              2$:    DEC    R4           ; PUT CR-LF AT END OF MESSAGE
84 007240 112724 000015      MOVB   #CR,(R4)+
85 007244 112724 000012      MOVB   #LF,(R4)+
86 007250 105014              CLRB   (R4)        ; PUT IN NULL TO SIGNAL END
87                               ;
88                               ; See who we should send the message to.
89                               ;
90 007252 005701              TST    R1           ; WHO IS IT GOING TO?
91 007254 001655              BEQ    LNUMER      ; SEND TO CONSOLE TERMINAL
92 007256 100450              BMI    SNDALL      ; SEND TO ALL LINES
93                               ;
94                               ; Send message to one terminal whose line index number
95                               ; is in R1.
96                               ;
97 007260 032761 000000G 000000G SNDONE: BIT    ##DILUP,LSW(R1) ; IS THAT LINE LOGGED ON?
98 007266 001005              BNE    1$           ; BRANCH IF YES
99 007270              .PRINT #NOTON
100 007276 000167 000000G     JMP    RDCMD
101 007302 032761 000000G 000000G 1$: BIT    #$TTGAG,LSW7(R1) ; IS LINE GAGGED?
102 007310 001414              BEQ    2$           ; BR IF NOT
103 007312 032761 000000G 000000G BIT    ##$INKMN,LSW4(R1) ; IS JOB IN KMON?
104 007320 001010              BNE    2$           ; BR IF YES
105 007322 105767 000000G     TSTB   YELEMT      ; Are we yelling?
106 007326 001005              BNE    2$           ; Br if yes -- Override gag
107 007330              .PRINT #GAGMSG    ; SAY LINE IS GAGGED
108 007336 000167 000000G     JMP    RDCMD
109 007342 012700 000000G     2$:    MOV    #YELEMT,R0 ; POINT TO EMT ARG BLOCK
110 007346 006201              ASR    R1           ; GET LINE NUMBER * 1
111 007350 010160 000002      MOV    R1,2(R0)    ; SET LINE # IN ARG BLOCK
112 007354 104375              EMT    375         ; SEND THE MESSAGE
113 007356 103006              BCC    3$           ; BR IF NO ERROR
114 007360              FWARN #NMSGBF

```

SEND command

```

115 007374 000167 000000G      3$:      JMP      RDCMD
116                               ;
117                               ; Send message to all terminals.
118                               ;
119 007400 012703 000000G      SNDALL:  MOV      #LSTPL,R3      ; GET INDEX # OF LAST LINE
120 007404 116702 000000G      MOVVB   CORUSR,R2      ; GET OUR LINE INDEX #
121 007410 016202 000000G      MOV     LNPRIM(R2),R2   ; GET OUR PRIMARY LINE INDEX #
122 007414 032763 000000G 000000G 2$:  BIT     #$DILUP,LSW(R3) ; IS THIS LINE LOGGED ON?
123 007422 001437                BEQ     1$              ; BRANCH IF NOT
124 007424 032763 000000G 000000G BIT     #$DETCH,LSW(R3) ; IS THIS A DETACH LINE?
125 007432 001033                BNE     1$              ; BR - DETACH LINES DON'T NEED MESSAGES
126 007434 020302                CMP     R3,R2           ; DON'T SEND MESSAGE TO OURSELF
127 007436 001431                BEQ     1$
128 007440 032763 000000G 000000G BIT     #$TTGAG,LSW7(R3); IS LINE GAGGED?
129 007446 001407                BEQ     3$              ; BR IF NOT
130 007450 105767 000000G      TSTB   YELEMT          ; Are we YELLing?
131 007454 001004                BNE     3$              ; Br if yes -- Override gag
132 007456 032763 000000G 000000G BIT     #$INKMN,LSW4(R3); IS JOB IN KMON NOW?
133 007464 001416                BEQ     1$              ; IF NOT THEN DON'T SEND MESSAGE
134 007466 012700 000000G      3$:     MOV     #YELEMT,R0   ; POINT TO EMT ARG BLOCK
135 007472 010301                MOV     R3,R1          ; GET LINE INDEX #
136 007474 006201                ASR     R1              ; GET LINE NUMBER
137 007476 010160 000002      MOV     R1,2(R0)       ; STORE LINE # IN EMT ARG BLOCK
138 007502 104375                EMT     375            ; SEND THE MESSAGE
139 007504 103006                BCC     1$              ; BR IF NO ERROR
140 007506                FWARN   #NMSGBF       ; REPORT NO MESSAGE BUFFERS
141 007522 162703 000002      1$:     SUB     #2,R3        ; MOVE ON TO NEXT LINE
142 007526 001332                BNE     2$              ; BRANCH IF MORE TO DO
143 007530 000167 000000G      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         007534 010546  SNDLIN: MOV      R5,-(SP)
11         ;
12         ;   See if a line number was specified
13         ;
14         007536 004767 000000G  CALL      SKPSPC      ;Skip over any spaces
15         007542 121327 000054  CMPB     (R3),#'      ;Start of line # field?
16         007546 001407  BEQ      1$           ;Br if yes
17         007550 121327 000057  CMPB     (R3),#'/     ;Start of line # field?
18         007554 001404  BEQ      1$           ;Br if yes
19         007556  FABORT  #EM$NLN      ;No line number was specified
20         ;
21         ;   See if we are sending to ALL, OPERATOR, or PARENT
22         ;
23         007566 005203 1$:      INC      R3           ;Point past delimiter
24         007570 004767 000000G  CALL      SKPSPC      ;Scan up to line number
25         007574 111300  MOVB     (R3),R0      ;Get 1st char of "number"
26         007576 020027 000141  CMP      R0,#141     ;Is this a lower-case letter?
27         007602 103402  BLD      2$           ;Br if not
28         007604 162700 000040  SUB      #40,R0      ;Convert lower-case to upper case
29         007610 120027 000101 2$:      CMPB     R0,#'A    ;Send to ALL?
30         007614 001003  BNE      3$           ;Br if not
31         007616 012701 177777  MOV      #-1,R1     ;Line # for All
32         007622 000424  BR       7$           ;
33         007624 120027 000117 3$:      CMPB     R0,#'O    ;Send to operator?
34         007630 001007  BNE      4$           ;Br if not
35         007632 116701 000000G  MOVB     CTRLTT,R1   ;Get operator line #
36         007636 001016  BNE      7$           ;
37         007640  FABORT  #NOOPTT      ;No operator terminal
38         007650 120027 000120 4$:      CMPB     R0,#'P    ;Send to parent job?
39         007654 001015  BNE      5$           ;Br if not
40         007656 116701 000000G  MOVB     CORUSR,R1   ;Get current job index #
41         007662 005761 000000G  TST      LPARNT(R1)  ;Is there a parent job?
42         007666 001402  BEQ      7$           ;Br if yes
43         007670 016101 000000G  MOV      LPARNT(R1),R1 ;Get # of parent job
44         007674 112300 7$:      MOVB     (R3)+,R0   ;Skip over rest of keyword
45         007676 001407  BEQ      9$           ;
46         007700 120027 000040  CMPB     R0,#'       ;Space?
47         007704 001373  BNE      7$           ;
48         007706 000403  BR       9$           ;
49         ;
50         ;   We must have an ordinary line number
51         ;
52         007710 004767 000000G 5$:      CALL      ACRDEC      ;Accrue line number
53         007714 006301  ASL      R1           ;Convert to line index number
54         ;
55         ;   Finished
56         ;
57         007716 012605 9$:      MOV      (SP)+,R5

```



58 007720 000207

RETURN

DETACH command

```

1          .SBTTL  DETACH command
2          ;-----
3          ; PROCESS THE DETACH COMMAND
4          ;
5 007722  004767  000000G  CMDDET: CALL  CVTTAB          ; CONVERT TAB AND FF CHARS TO SPACES
6 007726  032767  000000G 000000G BIT    #PO$DET, PRIVCO ; IS USER ALLOWED TO USE DETACHED JOBS?
7 007734  001004          BNE    1$          ; BR IF YES
8 007736          FABORT  #EM$DET        ; NOT ALLOWED
9          ;
10         ; SEE IF ANY SWITCHES WERE SPECIFIED WITH COMMAND
11         ;
12 007746  121327  000057  1$:    CMPB   (R3), #'/'      ; ANY SWITCHES?
13 007752  001062          BNE    DETGO        ; BR IF NOT -- START DETACHED JOB
14         ; ACCRUE AND IDENTIFY SWITCH
15 007754  005203          INC    R3          ; SKIP OVER /
16 007756  012704  000220'  MOV    #DETHD, R4      ; POINT TO TABLE OF SWITCHES
17 007762  004767  000000G  CALL  SEARCH          ; LOOK UP THE SWITCH
18 007766  103004          BCC   2$          ; BR IF FOUND OK
19 007770          FABORT  #INVOPT      ; INVALID SWITCH
20         ; ACCRUE LINE NUMBER AND SEE IF LEGAL
21 010000  004767  000000G  2$:    CALL  ACRDEC        ; ACCRUE THE DECIMAL NUMBER
22 010004  006301          ASL   R1          ; CONVERT TO LINE INDEX #
23 010006  020127  000000G  CMP   R1, #FSTDL      ; SEE IF IT IS A DETACHED LINE
24 010012  103004          BHIS  3$          ;
25 010014          4$:    FABORT  #BDLIN      ;
26 010024  020127  000000G  3$:    CMP   R1, #LSTDL      ;
27 010030  101371          BHI   4$          ;
28 010032  012700  000000G  MOV   #DETARG, R0     ; POINT TO DETACH EMT ARG BLOCK
29 010036  006201          ASR   R1          ; STORE DETACHED JOB #
30 010040  010167  000002G  MOV   R1, DETARG+2    ;
31         ; BRANCH OFF TO PROCESSING ROUTINE
32 010044  000134          JMP   @(R4)+         ; ENTER PROCESSING ROUTINE
33         ;
34         ; Process the /CHECK option.
35         ;
36 010046  112710  000001  DETCHK: MOVB  #1, (R0)   ; SET SUB-FUNCTION CODE FOR EMT
37 010052  104375          EMT   375          ; CHECK ON DETACHED JOB
38 010054  103404          BCS  DETIDL        ; BR IF JOB FINISHED
39 010056          .PRINT #RUNMS      ; JOB STILL RUNNING
40 010064  000403          BR   DETJMP        ;
41 010066          DETIDL: .PRINT #LINFRE ; LINE IS IDLE
42 010074  000167  000000G  DETJMP: JMP   RDCMD    ; FINISHED WITH COMMAND
43         ;
44         ; Process the /KILL option.
45         ;
46 010100  112710  000002  DETKIL: MOVB  #2, (R0)   ; SET SUB-FUNCTION CODE FOR EMT
47 010104  104375          EMT   375          ; KILL A DETACHED JOB
48 010106  103767          BCS  DETIDL        ; BR IF JOB NOT RUNNING
49 010110          .PRINT #DJABMS      ; PRINT CONFIRMATION
50 010116  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 010120 010302      DETGO:  MOV      R3,R2          ;Save pointer to start of file name string
7 010122 012704 000000G      MOV      #R50COM,R4      ;Point to word with default extension (COM)
8 010126 005005      CLR      R5              ;Say this is an input file
9 010130 004767 000000G      CALL     ACRFIL          ;Accrue the file spec
10 010134 103441      BCS     10$             ;Br if invalid file spec
11 010136      .LOOKUP #XAREA,#1,#FILNAM ;Try to open the file
12 010156 103434      BCS     11$             ;Br if cannot open file
13 010160      .CLOSE  #1          ;Close the file
14
15      ; Execute EMT to initiate the detached job.
16
17 010166 012700 000000G      MOV      #DETARG,R0      ;POINT TO DETACH EMT ARG BLOCK
18 010172 105010      CLR     (R0)            ;SET SUB-FUNCTION CODE FOR EMT
19 010174 010260 000002      MOV      R2,2(R0)       ;SET ADDRESS OF COMMAND FILE NAME ASCIZ STRING
20 010200 104375      EMT     375             ;START A DETACHED JOB
21 010202 103412      BCS     1$              ;BR IF NO FREE LINES
22
23      ; Tell user which line was used
24
25 010204 010005      MOV      R0,R5           ;GET DETACHED JOB LINE NUMBER
26 010206      .PRINT #DLMSG      ;PRINT HEADING
27 010214 004767 000000G      CALL     PRTDEC         ;DISPLAY LINE #
28 010220      .PRINT #CRLF      ;TERMINATE PRINT LINE
29 010226 000722      BR      DETJMP
30
31      ; No free lines
32
33 010230      1$:      FABORT #NOFRDL      ;NO FREE LINES
34
35      ; Invalid file spec
36
37 010240      10$:     FABORT #BDFNAM
38
39      ; Cannot open file
40
41 010250      11$:     FABORT #EM#FOE

```

DATE command

```

1          .SBTTL  DATE command
2          ;-----
3          ; Process the DATE command.
4          ;
5 010260 004767 000000G  CMDDAT: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
6 010264 105713          TSTB   (R3)        ; DOES HE WANT TO SHOW OR CHANGE THE DATE?
7 010266 001002          BNE    3$         ; BR TO SET THE DATE
8 010270 000167 000000G          JMP    SOPDAT      ; GO DISPLAY THE DATE
9          ;
10         ; Set a new system date value (operator privilege required).
11         ;
12 010274 004767 000000G  3$:   CALL  CKPRIV      ; MAKE SURE THIS USER IS PRIVILEGED
13         ; Get day value.
14 010300 004767 000000G          CALL  ACRDEC      ; ACCRUE DAY VALUE
15 010304 005701          TST   R1           ; CHECK RANGE OF VALUE
16 010306 003453          BLE   BADDAT      ; CAN'T BE ZERO
17 010310 020127 000037          CMP   R1,#31.     ; OR GREATER THAN 31
18 010314 101050          BHI   BADDAT
19 010316 072127 000005          ASH   #5,R1      ; POSITION FOR DATE WORD
20 010322 010105          MOV   R1,R5      ; CONSTRUCT DATE WORD IN R5
21 010324 122327 000055          CMPB (R3)+,#'-   ; HYPHEN SHOULD BE NEXT
22 010330 001042          BNE   BADDAT
23         ; Get month.
24 010332 004767 000000G          CALL  @TRD50     ; ACCRUE RAD50 MONTH VALUE
25 010336 016700 000000G          MOV   R50BUF,R0 ; GET RAD50 VALUE OF MONTH NAME
26 010342 012701 000001          MOV   #1,R1     ; FIRST MONTH IS # 1
27 010346 012702 000000G          MOV   #R50MON,R2 ; POINT TO TABLE OF MONTH NAMES
28 010352 020022          2$:   CMP   R0,(R2)+ ; LOOK UP NAME IN TABLE
29 010354 001405          BEQ   1$       ; BR IF FOUND
30 010356 005201          INC   R1       ; ADVANCE MONTH NUMBER
31 010360 020127 000014          CMP   R1,#12.  ; ONLY 12 MONTHS
32 010364 101772          BLOS  2$
33 010366 000423          BR    BADDAT   ; BAD MONTH NAME
34 010370 072127 000012          1$:   ASH   #10.,R1 ; POSITION MONTH VALUE
35 010374 050105          BIS   R1,R5    ; AND OR INTO DATE WORD
36 010376 122327 000055          CMPB (R3)+,#'-   ; SHOULD HAVE ANOTHER HYPHEN
37 010402 001015          BNE   BADDAT
38         ; Get year
39 010404 004767 000000G          CALL  ACRDEC      ; GET YEAR VALUE
40 010410 020127 000143          CMP   R1,#99.   ; CHECK FOR REASONABLE UPPER LIMIT
41 010414 101010          BHI   BADDAT    ; BR IF TOO HIGH
42 010416 162701 000110          SUB   #72.,R1   ; YEAR VALUE IS BIASED BY 72
43 010422 003405          BLE   BADDAT    ; BR IF TOO SMALL
44 010424 050105          BIS   R1,R5    ; FORM DATE WORD
45         ; Set system date value.
46 010426 010567 000000G          MOV   R5,SYSDAT ; SET NEW SYSTEM DATE VALUE
47 010432 000167 000000G          JMP   RDCMD     ; FINISHED WITH COMMAND
48         ; Invalid date value.
49 010436 152767 000010 000000G BADDAT: BISB  #10,INDERR ; SAVE ERROR LEVEL = SEVERE FOR IND
50 010444          FABORT #INV DAT ; INVALID DATE

```

TIME command

```

1          .SBTTL  TIME command
2          ;-----
3          ; Process the TIME command.
4          ;
5 010454  004767  000000G  CMDTIM: CALL  CVTTAB      ; CONVERT TAB AND FF CHARS TO SPACES
6 010460  105713                TSTB   (R3)      ; DOES HE WANT TO SHOW OR CHANGE THE TIME?
7 010462  001002                BNE    2$          ; BR TO SET THE TIME
8 010464  000167  000000G      JMP    SOPTIM     ; GO SHOW THE TIME
9          ;
10         ; Set new system time (requires operator privilege)
11         ;
12 010470  004767  000000G  2$:   CALL  CKPRIV     ; MAKE SURE USER HAS OPERATOR PRIVILEGE
13         ; Get hour value.
14 010474  004767  000000G      CALL  ACRDEC     ; ACCRUE HOUR VALUE
15 010500  020127  000030      CMP    R1,#24.   ; ONLY 24 HOUR IN A DAY
16 010504  103053                BHIS  BADTIM
17 010506  010104                MOV   R1,R4
18 010510  070427  000074      MUL   #60.,R4   ; CONVERT TO # MINUTES
19 010514  122327  000072      CMPB  (R3)+,#'  ; COLON SHOULD BE THE DELIMITER
20 010520  001045                BNE  BADTIM
21         ; Get minute value.
22 010522  004767  000000G      CALL  ACRDEC     ; ACCRUE MINUTE VALUE
23 010526  020127  000074      CMP    R1,#60.  ; ONLY 60 MINUTES PER HOUR
24 010532  103040                BHIS  BADTIM
25 010534  060105                ADD   R1,R5     ; COMBINE WITH HOUR VALUE
26 010536  005504                ADC   R4
27 010540  012700  000074      MOV   #60.,R0  ; CONVERT # MINUTES TO # SECONDS
28 010544  004767  000000G      CALL  MUL32
29         ; See if seconds are specified
30 010550  122327  000072      CMPB  (R3)+,#'  ; DELIMITER FOR SECONDS?
31 010554  001011                BNE  3$         ; NO, IGNORE SECONDS
32         ; Get second value
33 010556  004767  000000G      CALL  ACRDEC     ; ACCRUE SECOND VALUE
34 010562  020127  000074      CMP    R1,#60.  ; NO MORE THAN 60 SECONDS PER MINUTE
35 010566  103022                BHIS  BADTIM
36 010570  060105                ADD   R1,R5     ; COMBINE WITH HOURS AND MINUTES
37 010572  005504                ADC   R4
38 010574  012700  000074      MOV   #60.,R0  ; 60 HZ CLOCK TICKS PER SECOND
39         ; Convert time to # clock ticks.
40 010600  032767  000000G 000000G 3$:   BIT   #CW$50H,CONFIG ; 50 OR 60 HZ CLOCK?
41 010606  001402                BEQ   1$         ; BR IF 60 HZ
42 010610  012700  000062      MOV   #50.,R0  ; SET FOR 50 HZ CLOCK
43 010614  004767  000000G 1$:   CALL  MUL32     ; CONVERT # SECONDS TO # CLOCK TICKS
44         ; Set new system time.
45 010620  010467  000000G      MOV   R4,SYTIMH ; HIGH-ORDER TIME VALUE
46 010624  010567  000000G      MOV   R5,SYTIML ; LOW-ORDER TIME VALUE
47 010630  000167  000000G      JMP   RDCMD     ; FINISHED
48         ; Bad time value entered
49 010634  152767  000010 000000G BADTIM: BISB  #10,INDERR ; SAVE ERROR LEVEL = SEVERE FOR IND
50 010642                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 010652   004767   000000G  CMDRST: CALL   CKSYPV           ;Require SYSPRV privilege
7          ; Reset data cache statistics
8          CLR     DCTRD           ;TOTAL NUMBER OF READS FROM SHARED FILES
9 010656   005067   000000G  CLR     DCCRD           ;NUMBER OF READS FOUND IN CACHE
10 010662   005067   000000G  CLR     DCTWR           ;TOTAL NUMBER OF WRITES TO SHARED FILES
11 010666   005067   000000G  CLR     DCCWR           ;NUMBER OF WRITES THAT UPDATE CACHE DATA
12          ; Reset 32-bit data cells
13 010676   012701   010720'  MOV     #RSTVEC,R1       ;POINT TO VECTOR OF ADDRESSES TO CLEAR
14 010702   012102           1$:  MOV     (R1)+,R2       ;GET ADDRESS OF A CELL TO ZERO
15 010704   001403           BEQ     2$                ;BR IF HIT END OF LIST
16 010706   005022           CLR     (R2)+            ;CLEAR HIGH-ORDER WORD
17 010710   005012           CLR     (R2)            ;CLEAR LOW-ORDER WORD
18 010712   000773           BR      1$
19 010714   000167   000000G  2$:  JMP     RDCMD         ;FINISHED
20          ;
21          ; Vector of 32-bit cells to clear.
22          ;
23 010720   000000G  RSTVEC: .WORD   TMTOTH
24 010722   000000G  .WORD   TMUSRH
25 010724   000000G  .WORD   TMSWTH
26 010726   000000G  .WORD   TMIOH
27 010730   000000G  .WORD   TMSWPH
28 010732   000000G  .WORD   TMIOWH
29 010734   000000G  .WORD   TMIDLH
30 010736   000000G  .WORD   CASTRO
31 010740   000000G  .WORD   CASTBR
32 010742   000000G  .WORD   CASCBR
33 010744   000000G  .WORD   CASTWO
34 010746   000000G  .WORD   CASTBW
35 010750   000000G  .WORD   CASCUP
36 010752   000000G  .WORD   0

```

OFF command

```

1          .SBTTL  OFF command
2          ;-----
3          ; LOG OFF.
4          ;
5 010754   116701   000000G   CMDOFF: MOVB   CORUSR,R1       ;GET USER INDEX #
6          ;
7          ; See if any qualifiers were specified with command
8          ;
9 010760   105067   167061     CLRB   OFFNWF       ;Clear NOWARN flag
10 010764   012704   000250'    MOV    #OFFHD,R4     ;Point to parsing table
11 010770   004767   000000G    CALL   OPTLST       ;Process any qualifiers
12          ;
13          ; See if this is a primary process logging off with active subprocesses
14          ;
15 010774   020127   000000G    CMP    R1,#LSTPL    ;Is this a primary process?
16 011000   101027           BHI    14$          ;Br if not
17 011002   105767   167037     TSTB   OFFNWF       ;Was /NOWARN qualifier specified?
18 011006   001024           BNE    14$          ;Br if yes -- Don't worry about subprocesses
19 011010   105767   000000G    TSTB   STPFLG      ;Is this a STOP request?
20 011014   001021           BNE    14$          ;Br if yes -- Don't worry about subprocesses
21 011016   032761   000000G 000000G  BIT    $$PRGLK,LSW5(R1);Did we exit a locked program?
22 011024   001015           BNE    14$          ;Br if yes -- Don't worry about subprocesses
23 011026   032761   000000G 000000G  BIT    $$NOIN,LSW3(R1);Are we allowed to have terminal input?
24 011034   001011           BNE    14$          ;Br if not
25 011036   032761   000000G 000000G  BIT    $$LOFCF,LSW9(R1);Have we already done a logAoff command file?
26 011044   001005           BNE    14$          ;Do logoff if so
27 011046   004767   001142     CALL   CKOFSP       ;See if we need to warn about subprocesses
28 011052   103002           BCC    14$          ;Br if we should logoff
29 011054   000167   000000G    JMP    RDCMD       ;Abort the logoff
30          ;
31          ; Do the logoff.
32          ;
33 011060   052761   000000G 000000G 14$:   BIS    $$NOIN,LSW3(R1);DON'T ALLOW ABORT OF LOGOFF
34 011066   005067   000000G    CLR    RESDEV      ;RELEASE ALL DEVICE/FILE ACCESS RESTRICTIONS
35          ;
36          ; Close all of user's files.
37          ;
38 011072   004767   000000G    CALL   PRGALL       ;PURGE ALL OF USER'S CHANNELS
39 011076   004767   000000G    CALL   INDABT       ;Abort IND and nested command files
40 011102   004767   000000G    CALL   ABRTCF       ;CLOSE THE CONTROL FILE
41 011106   052761   000000G 000000G  BIS    $$NOIN,LSW3(R1);CLEARED BY ABRTCF(POPCF), SET IT AGAIN
42          ;
43          ; Determine if we need to invoke a logoff command file
44          ;
45 011114   005767   000000G    TST    LOFSPC      ;Is there a logoff command file?
46 011120   001471           BEQ    3$           ;Br if not
47 011122   004767   000000G    CALL   PUSHCF       ;Prepare to open new command file
48 011126   112767   000001  000000G  MOVB   #1,SERFLG    ;Do .SERR
49 011134           .LOOKUP #XAREA,#CFCHAN,#LOFSPC ;Try to open the command file
50 011154   112767   000000  000000G  MOVB   #0,SERFLG    ;Do .HERR but don't affect carry flag
51 011162   103013           BCC    10$          ;Br if open was ok
52 011164   004767   000000G 11$:   CALL   POPCF        ;Pop command file status
53 011170   005067   000000G    CLR    LOFSPC      ;No logoff command file
54 011174           FERR   #EM$OLD     ;Cannot open logoff command file
55 011210   000435           BR     3$           ;
56 011212           10$:   .READW #XAREA,#CFCHAN,#CFBUF,#256.,#0 ;Read 1st block of file
57 011250   103745           BCS   11$          ;Br if read error

```

OFF command

```

58 011252 052761 000000G 000000G      BIS      %#CFOPN,LSW4(R1); Say CFCHAN is open
59 011260 052761 000000G 000000G      BIS      %#LOFCF,LSW9(R1); Say we are processing logoff command file
60 011266 042761 000000G 000000G      BIC      %#DOOFF,LSW(R1) ; Say not doing logoff processing yet
61 011274 005067 000000G      CLR      LDFSPC      ; Say logoff command file has been started
62 011300 000167 000000G      JMP      RDCMD      ; Go process 1st command in command file
63
64      ; Enable privilege during logoff processing
65
66 011304 012702 000000G      3$:      MOV      #PRIVCO,R2      ; Point to current privilege word
67 011310 012703 000000G      MOV      #PRIVSO,R3      ; Point to set privileges
68 011314 012700 000000G      MOV      #PVNPW,R0      ; Get # privilege words
69 011320 012722 177777      12$:     MOV      #177777,(R2)+    ; Set all privilege flags
70 011324 012723 177777      MOV      #177777,(R3)+
71 011330 077005      SOB      R0,12$
72 011332 052767 000000G 000000G      BIS      #LF$WRT,LOGFLG ; ENABLE WRITES TO LOG FILE
73
74      ; Dismount all devices that were mounted by this job.
75
76 011340 004767 000000G      CALL     DMTALL      ; Dismount all mounted devices
77
78      ; Print usage statistics for line
79
80 011344 032761 000000G 000000G      BIT      %#VNOTT,LSW(R1) ; IS THIS JOB CONNECTED TO A TERMINAL?
81 011352 001033      BNE      7$          ; BR IF NOT
82 011354 016100 000000G      MOV      LNPRIM(R1),R0   ; GET PRIMARY LINE #
83 011360 042760 000000G 000000G      BIC      %#CTRLS,LSW3(R0); MAKE SURE OUTPUT IS ENABLED
84 011366 020127 000000G      CMP      R1,#LSTDL      ; Is this a subprocess?
85 011372 101403      BLOS     13$         ; Br if not
86 011374 005760 000000G      TST      LWINDO(R0)     ; Does primary process have windowing on?
87 011400 001020      BNE      7$          ; Br if yes -- Don't display logoff time
88 011402      13$:     .PRINT  #CRLF
89 011410 105767 000000G      TSTB     STPFLG      ; IS SYSTEM BEING SHUT DOWN?
90 011414 001403      BEQ      5$          ; BR IF NOT
91 011416      .PRINT  #SHTMSG     ; PRINT SHUT-DOWN MESSAGE
92 011424 004767 000000G      5$:      CALL     PRTTIM      ; PRINT USAGE INFO
93 011430      .TTYOUT #LF          ; PUT OUT EXTRA LF
94 011440 000420      BR      TIMUP
95      ; IF PRTTIM DIDN'T DO CPU CALCULATION WE MUST DO IT HERE.
96 011442 016104 000000G      7$:      MOV      LCPUHI(R1),R4   ; GET HIGH-ORDER CPU TIME (CLOCK TICKS)
97 011446 016105 000000G      MOV      LCPULO(R1),R5   ; GET LOW-ORDER CPU TIME (CLOCK TICKS)
98 011452 016703 000000G      MOV      TK1SEC,R3      ; GET # CLOCK TICKS PER SECOND
99 011456 004767 000000G      CALL     DIVIDE      ; CONVERT TIME VALUE TO # SECONDS
100 011462 012700 000012      MOV      #10.,R0        ; CONVERT TIME VALUE TO 0.1 SECOND
101 011466 004767 000000G      CALL     MUL32        ; UNITS BY MULTIPLYING BY 10
102 011472 010467 000000G      MOV      R4,CPUAH      ; SAVE HIGH-ORDER PART (0.1 SEC)
103 011476 010567 000000G      MOV      R5,CPUAL      ; SAVE LOW-ORDER PART (0.1 SEC)
104
105      ; If user logged on, then accumulate his connect time
106
107 011502 016105 000000G      TIMUP:   MOV      LPROJ(R1),R5 ; DO WE HAVE A PPN FOR USER?
108 011506 001535      BEQ      NOTIM        ; BR IF NOT -- DIDN'T LOG ON
109
110      ; DO A DEASSIGN
111 011510 012702 000000G      MOV      #ASNTBL,R2    ; CLEAR ALL OF HIS ASSIGNS
112 011514 005022      1$:      CLR      (R2)+
113 011522 103774      CMP      R2,#ASNEND
114      BLO      1$
      ; Try to open accounting file

```



OFF command

```

115 011524          . LOOKUP #XAREA,#1,#AUTHFN
116 011544 103004   . BCC 2$ ;BR IF OPEN OK
117 011546          . PRINT #COAF ;PRINT WARNING MESSAGE
118 011554 000512   BR NOTIM
119                ; Search file for user's ppn entry
120 011556 005002   2$: CLR R2 ;SET BLOCK # TO 0
121 011560 016103 000000G . MOV LPROG(R1),R3 ;GET PROGRAMMER #
122 011564          6$: . READW #XAREA,#1,#BLKO,#256.,R2
123 011622 103464   . BCS 3$ ;BR IF END OF FILE HIT
124 011624 004767 000544 . CALL CVTBUF ;COMPLEMENT CONTENTS OF BUFFER
125 011630 012704 000000G . MOV #BLKO,R4 ;SEARCH FOR RECORD IN BLOCK
126 011634 012700 000000G . MOV #ARNRPB,R0 ;# RECORDS PER BLOCK
127 011640 020564 000000G 5$: . CMP R5,AR$PRJ(R4) ;DO PROJECT #'S MATCH?
128 011644 001003   . BNE 8$ ;BR IF NOT
129 011646 020364 000000G . CMP R3,AR$PRG(R4) ;HOW ABOUT PROGRAMMER NUMBERS
130 011652 001406   . BEQ 4$ ;BR IF WE FOUND THE RECORD
131 011654 062704 000000G 8$: . ADD #AR$$SZ,R4 ;KEEP LOOKING IN BLOCK
132 011660 005300   . DEC R0 ;ANY MORE RECORDS IN THIS BLOCK?
133 011662 001366   . BNE 5$ ;BR IF YES
134 011664 005202   . INC R2 ;GO READ NEXT BLOCK
135 011666 000736   . BR 6$
136                ; Found record. Add in new connect time
137 011670 016700 000000G 4$: . MOV MINTIM,R0 ;GET CURRENT TIME
138 011674 166100 000000G . SUB LCONTM(R1),R0 ;CALCULATE CONNECT TIME
139 011700 005200   . INC R0
140 011702 060064 000000G 7$: . ADD R0,AR$CON(R4) ;ACCUMULATE CONNECT TIME
141                ; COUNT NUMBER OF SESSIONS
142 011706 005264 000000G . INC AR$CNT(R4) ;INC SESSION COUNTER
143                ; ACCUMULATE CPU TIME
144 011712 066764 000000G 000000G . ADD CPUAL,AR$CPL(R4) ;ADD IN LOW-ORDER CPU TIME
145 011720 005564 000000G . ADC AR$CPH(R4) ;ADD CARRY
146 011724 066764 000000G 000000G . ADD CPUAH,AR$CPH(R4) ;ADD IN HIGH-ORDER PART
147                ; WRITE UPDATED ACCOUNTING RECORD BACK TO FILE
148 011732 004767 000436 . CALL CVTBUF ;COMPLEMENT CONTENTS OF BUFFER
149 011736          . WRITW #XAREA,#1,#BLKO,#256.,R2
150 011774          3$: . CLOSE #1
151                ;
152                ; Close job's log file
153                ;
154 012002 004767 000000G NOTIM: . CALL LOGCLS ;CLOSE LOG FILE
155                ;
156                ; Broadcast message to any monitoring jobs telling them we are logging off
157                ;
158 012006 012700 000000G . MOV #GENMON,R0 ;Point to emt argument block
159 012012 012760 000000G 000002 . MOV #JS$OFF,2(R0) ;Set status code
160 012020 104375   . EMT 375 ;Broadcast status message
161                ;
162                ; Enter TSX to finish log off.
163                ;
164 012022 105767 000000G . TSTB STPFLG ;ARE WE DOING A SYSTEM SHUTDOWN?
165 012026 001460   . BEQ 1$ ;BR IF NOT
166 012030 003046   . BGT 15$ ;Br if $STOP or BOOT specified
167                ;
168                ; Shutdown was specified.
169                ;
170 012032 052761 000000G 000000G . BIS ##NOABT,LSW9(R1);Don't allow job abort
171 012040 120127 000000G . CMPB R1,#LSTPL ;Is this a detached job?

```

OFF command

```

172 012044 103403          BLD      10$          ;Br if not detached
173 012046 120127 000000G  CMPB    R1,#LSTD L    ;In the detached range?
174 012052 101435          BLOS   15$          ;Yes, job is detached
175 012054 126727 000000G 000001 10$:  CMPB    PVON,#1     ;Are we the last primary/virtual job?
176 012062 003031          BGT     15$          ;Br if more jobs exist
177
178 ; Kill all jobs (only detached jobs should be active).
179 ;
180 012064 012702 000000G  MOV     #LSTSL,R2    ;GET # OF LAST LINE
181 012070 120267 000000G 11$:  CMPB    R2,CORUSR   ;IS THIS OUR LINE?
182 012074 001405          BEQ     12$          ;BR IF YES
183 012076 012700 000000G  MOV     #KILEMT,R0   ;POINT TO KILL EMT ARG BLOCK
184 012102 010260 000004  MOV     R2,4(R0)     ;SET # OF LINE TO KILL
185 012106 104375          EMT     375         ;KILL THE LINE
186 012110 162702 000002 12$:  SUB     #2,R2       ;DO NEXT LINE
187 012114 001365          BNE     11$
188
189 ; Determine if the spooler is still active.
190 ;
191 012116 004767 000000G 13$:  CALL    SPLACT      ;Is the spooler still active?
192 012122 103011          BCC     15$          ;Br if spooler is not active.
193 012124          .TWAIT #XAREA,#TIMSPL ;Wait 2 seconds -- check again
194 012144 000764          BR      13$
195
196 ; $STOP or BOOT issued.
197 ;
198 012146 042761 000000G 000000G 15$:  BIC     ##NOABT,LSW9(R1); Allow job abort (necessary for log off)
199 012154 126727 000000G 000001  CMPB    TONON,#1     ;Are we the last job on system?
200 012162 003002          BGT     1$          ;Br if more jobs exist
201 012164 000167 000000G  JMP     DOSTOP      ;GO STOP THE SYSTEM
202 012170 005061 000000G 1$:  CLR     LPROJ(R1)   ;CLEAR PROJECT/PROGRAMMER #
203 012174 005061 000000G  CLR     LPROG(R1)
204 012200 012700 000000G  MOV     #OFFEMT,R0   ;POINT TO LOG-OFF EMT ARG BLOCK
205 012204 104375          EMT     375         ;LOG OFF THIS JOB

```

OFF command

```

1
2 ; -----
3 ; Subroutine called if the /NOWARN qualifier is specified with the OFF
4 ; command.
5 012206 105267 165633 OFFNWN: INCB OFFNWF ;Set NOWARN flag
6 012212 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 012214 010246 CKOFSP: MOV R2, -(SP)
20 012216 010546 MOV R5, -(SP)
21 ;
22 ; See if this process has any active subprocesses
23 ;
24 012220 005005 CLR R5 ;Count active subprocesses in R5
25 012222 016102 000000G MOV LSECT(R1),R2 ;Point to table of subprocess #'s
26 012226 012700 000000G MOV #MAXSEC,R0 ;Get max # subprocesses
27 012232 001454 BEQ 7$ ;Br if none to check
28 012234 105722 1$: TSTB (R2)+ ;Is this subprocess active?
29 012236 001401 BEQ 4$ ;Br if not
30 012240 005205 INC R5 ;Count # active subprocesses
31 012242 077004 4$: SOB R0,1$ ;Loop to check all
32 012244 005705 TST R5 ;Any active subprocesses?
33 012246 001446 BEQ 7$ ;Br if not
34 ;
35 ; There are active subprocesses.
36 ; Print warning message.
37 ;
38 012250 020527 000001 CMP R5,#1 ;One active subprocess?
39 012254 003004 BGT 2$ ;Br if more than one
40 012256 .PRINT #TM$SA1 ;You have 1 active subprocess
41 012264 000410 BR 5$
42 012266 .PRINT #TM$SA3 ;You have
43 012274 004767 000000G CALL PRTDEC ;Print # subprocesses
44 012300 .PRINT #TM$SA4 ;active subprocesses
45 ;
46 ; Read response
47 ;
48 012306 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 012326 012702 000000G MOV #BLKO,R2 ;Point to response buffer
54 012332 112200 3$: MOVB (R2)+,R0 ;Get next char of response
55 012334 001411 BEQ 8$ ;Br if hit end of line
56 012336 120027 000040 CMPB R0,#40 ;Ignore spaces
57 012342 001773 BEQ 3$

```

OFF command

```

58 012344 120027 000131          CMPB    RO,#'Y          ;Is response Yes?
59 012350 001405                BEQ     7#              ;Br if yes
60 012352 120027 000171          CMPB    RO,#'y          ;Allow lower-case too
61 012356 001402                BEQ     7#
62                               ;
63                               ;  Abort the logoff
64                               ;
65 012360 000261                8#:    SEC              ;Signal abort on return
66 012362 000401                BR      9#
67                               ;
68                               ;  Allow the logoff
69                               ;
70 012364 000241                7#:    CLC              ;Signal to logoff
71                               ;
72                               ;  Finished
73                               ;
74 012366 012605                9#:    MOV    (SP)+,R5
75 012370 012602                MOV    (SP)+,R2
76 012372 000207                RETURN

```

OFF command

```

1
2 ; -----
3 ; Complement the contents of BLKO buffer.
4 ; We do this to mildly encrypt the authorization file.
5 ;
5 012374 010146 CVTBUF: MOV R1,-(SP)
6 012376 010246 MOV R2,-(SP)
7 012400 012701 000000G MOV #BLKO,R1 ;POINT TO BUFFER
8 012404 012702 000400 MOV #256.,R2 ;GET # WORDS TO COMPLEMENT
9 012410 005121 1$: COM (R1)+ ;COMPLEMENT CONTENTS OF BUFFER
10 012412 077202 SOB R2,1$
11 012414 012602 MOV (SP)+,R2
12 012416 012601 MOV (SP)+,R1
13 012420 000207 RETURN

```

KILL command

```

1           .SBTTL  KILL command
2           ;-----
3           ; The KILL command is used to abort the specified job.
4           ;
5 012422    012705  000000G  CMDKIL: MOV      #KILEMT,R5      ;Point to Kill EMT arg block
6 012426    000405                BR          JOBCOM
7
8           .SBTTL  SUSPEND command
9           ;-----
10          ; Suspend the execution of a specified job
11          ;
12 012430    012705  000000G  CMDSPN: MOV      #SJEMT,R5      ;Point to suspend EMT arg block
13 012434    000402                BR          JOBCOM
14
15          .SBTTL  RESUME command
16          ;-----
17          ; Resume the execution of a suspended job
18          ;
19 012436    012705  000000G  CMDRSM: MOV      #RJEMT,R5      ;Point to resume EMT arg block
20          ;
21          ; Accrue the job number
22          ;
23 012442    004767  000000G  JOBCOM: CALL     CVTTAB          ;convert tab and FF chars to spaces
24 012446    004767  000000G          CALL     ACRDEC          ;accrue decimal value
25 012452    006301                ASL      R1          ;Convert # to job index number
26 012454    001403                BEQ      1$          ;Br if invalid line number
27 012456    020127  000000G          CMP      R1,#LSTSL      ;Is this a valid line number
28 012462    101404                BLOS    2$          ;Br if ok
29 012464                1$:      FABORT  #BDLIN      ;Invalid line number
30          ;
31          ; See if we are privileged to affect this job
32          ;
33 012474    010102                2$:      MOV      R1,R2      ;Get job # to R2 for CKACDJ
34 012476    004767  000000G          CALL     CKACDJ          ;Can we access this job
35 012502    103404                BCS     9$          ;Br if not
36          ;
37          ; Execute EMT to affect the job
38          ;
39 012504    010500                MOV      R5,R0          ;Point to EMT arg list
40 012506    010160  000004          MOV      R1,4(R0)      ;Set job number
41 012512    104375                EMT     375            ;Perform the function
42          ;
43          ; Finished
44          ;
45 012514    000167  000000G  9$:      JMP      RDCMD

```

```

1          .SBTTL  BOOT and $STOP commands
2          ;-----
3          ; Reboot RT-11.
4          ;
5 012520 004767 000000G  CMDDBOT: CALL  CKPRIV      ;USER MUST HAVE OPERATOR COMMAND PRIVILEGE
6 012524 004767 000000G          CALL  CVTTAB      ;CONVERT TAB AND FF CHARS TO SPACES
7          ;
8          ; See if a boot device was specified.
9          ;
10 012530 005067 000000G          CLR    BOTDEV      ;ASSUME NO BOOT DEVICE WILL BE SPECIFIED
11 012534 105713          TSTB   (R3)        ;WAS A BOOT DEVICE SPECIFIED?
12 012536 001450          BEQ    7$          ;BR IF NOT -- USE SYSTEM DEVICE
13          ; Accrue device name.
14 012540 004767 000000G          CALL  GTRD50      ;ACCRUE RAD50 DEVICE NAME
15 012544 016767 000000G 000000G  MOV    R50BUF,FILNAM  ;SAVE RAD50 NAME OF BOOT DEVICE
16 012552 005067 000002G          CLR    FILNAM+2      ;Clear remaining file specification
17 012556 005067 000004G          CLR    FILNAM+4
18 012562 005067 000006G          CLR    FILNAM+6
19 012566          .LOOKUP #XAREA,#1,#FILNAM;Open a channel to the boot device
20 012606 103510          BCS    NODEV        ;Br if lookup is unsuccessful
21 012610          .CSTAT  #XAREA,#1,#CINFO;Obtain channel information from boot device
22 012630 016767 165162 000000G  MOV    CINFO+12,BOTDEV ;Transfer RAD50 device name to boot device
23 012636 066767 165152 000000G  ADD    CINFO+10,BOTDEV ;Add unit number to boot device name
24 012644 062767 000036 000000G  ADD    #36,BOTDEV     ;Convert unit number to rad50
25 012652          .CLOSE  #1          ;Close channel to boot device
26          ;
27          ; Make sure system is idle.
28          ;
29          ; See if any other users are logged on.
30 012660 005005 7$: CLR    R5          ;IDLE FLAG
31 012662 126727 000000G 000001  CMPB   NUMON,#1      ;ARE WE ONLY JOB LOGGED ON?
32 012670 003404          BLE    1$          ;BR IF YES
33 012672          .PRINT  #OTHRON      ;OTHER USERS LOGGED ON
34 012700 005205          INC    R5          ;REMEMBER NOT IDLE
35          ; See if the spooler is idle.
36 012702 004767 000000G 1$: CALL  SPLACT      ;SEE IF SPOOLER IS ACTIVE
37 012706 103004          BCC   2$          ;BR IF SPOOLER IS IDLE
38 012710          .PRINT  #SPLPND      ;THERE ARE PENDING SPOOL FILES
39 012716 005205          INC    R5          ;REMEMBER SYSTEM IS NOT IDLE
40          ; See if we need to ask for confirmation.
41 012720 005705 2$: TST    R5          ;WAS SYSTEM IDLE?
42 012722 001420          BEQ    10$         ;BR IF YES
43 012724          .PRINT  #STPASK      ;ASK FOR CONFIRMATION
44 012732 5$: .TTYIN          ;ACCEPT INPUT LINE
45 012736 120027 000131          CMPB   R0,#'Y        ;DID HE SAY YES?
46 012742 001410          BEQ    10$         ;BR IF YES
47 012744 120027 000171          CMPB   R0,#171      ;LOWER-CASE 'Y'
48 012750 001405          BEQ    10$         ;BR IF YES
49 012752 120027 000012          CMPB   R0,#LF       ;EAT REST OF LINE
50 012756 001365          BNE    5$          ;
51 012760 000167 000000G          JMP    RDCMD        ;GO GET NEXT COMMAND
52          ;
53          ; Force logoff of all users.
54          ;
55 012764 112767 000001 000000G 10$: MOVB   #1,STPFLG     ;SET FLAG SAYING SYSTEM IS BEING STOPPED
56 012772 012702 000000G          MOV    #LSTSL,R2    ;GET # OF LAST LINE
57 012776 120267 000000G 11$: CMPB   R2,CORUSR     ;IS THIS OUR LINE?

```

```
58 013002 001405          BEQ      12$          ;BR IF YES
59 013004 012700 000000G   MOV      #KILEMT,R0      ;POINT TO KILL EMT ARG BLOCK
60 013010 010260 000004     MOV      R2,4(R0)        ;SET # OF LINE TO KILL
61 013014 104375          EMT      375             ;KILL THE LINE
62 013016 162702 000002   12$:    SUB      #2,R2     ;DO NEXT LINE
63 013022 001365          BNE      11$
64                          ;
65                          ; Now enter logoff processing for our line.
66                          ; It will jump to DOSTOP after it finishes.
67                          ;
68 013024 000167 175724     JMP      CMDOFF          ;LOG OFF OUR LINE
69                          ;
70                          ; Invalid boot device - could not locate boot device
71                          ;
72 013030  NODEV:  FABORT  #BADBOT      ;Say bad boot device
73
```



#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 013040 004767 000000G CMD$HT: CALL CKPRIV ;USER MUST HAVE OPERATOR PRIV
7 013044 112767 177777 000000G MOV$ # -1, STPFLG ;SAY SYSTEM SHUTDOWN TAKING PLACE
8 013052 005067 000000G CLR BOTDEV ;REBOOT FROM SYSTEM DISK
9 013056 000167 000000G JMP RDCMD
10 000001 .END
```

Errors detected: 0

\*\*\* Assembler statistics

Work file reads: 0  
Work file writes: 0  
Size of work file: 10920 Words ( 43 Pages)  
Size of core pool: 18176 Words ( 71 Pages)  
Operating system: RT-11

Elapsed time: 00:01:50.40  
,LP: TSKM2A=DK: TSKM2A/C/N: SYM



|          |       |         |         |        |        |        |       |        |      |
|----------|-------|---------|---------|--------|--------|--------|-------|--------|------|
| AR\$\$SZ | 1-144 | 40-131  |         |        |        |        |       |        |      |
| AR\$CNT  | 1-143 | 40-142* |         |        |        |        |       |        |      |
| AR\$CON  | 1-143 | 40-140* |         |        |        |        |       |        |      |
| AR\$CPH  | 1-143 | 40-145* | 40-146* |        |        |        |       |        |      |
| AR\$CPL  | 1-143 | 40-144* |         |        |        |        |       |        |      |
| AR\$PRG  | 1-143 | 40-129  |         |        |        |        |       |        |      |
| AR\$PRJ  | 1-143 | 40-127  |         |        |        |        |       |        |      |
| ARNRPB   | 1-144 | 40-126  |         |        |        |        |       |        |      |
| ASCIS    | 8-53  | 8-62#   |         |        |        |        |       |        |      |
| ASCOK    | 8-65  | 8-81#   |         |        |        |        |       |        |      |
| ASDELM   | 8-17  | 9-7#    |         |        |        |        |       |        |      |
| ASDEX    | 1-119 | 8-63    |         |        |        |        |       |        |      |
| ASFID    | 8-36  | 8-57#   |         |        |        |        |       |        |      |
| ASKLNM   | 1-118 | 8-8*    | 8-52*   | 8-85   |        |        |       |        |      |
| ASNEND   | 1-83  | 10-14   | 10-28   | 40-112 |        |        |       |        |      |
| ASNOVF   | 1-119 | 8-97    |         |        |        |        |       |        |      |
| ASNSRC   | 1-103 | 8-82    | 8-90    | 8-94   | 25-31  | 29-21  |       |        |      |
| ASNTBL   | 1-81  | 10-12   | 10-22   | 40-110 |        |        |       |        |      |
| ASSMPL   | 8-42# |         |         |        |        |        |       |        |      |
| AT\$\$SZ | 1-82  | 10-27   |         |        |        |        |       |        |      |
| AT\$DEV  | 1-82  | 8-84    | 8-105*  | 10-26* | 25-33  | 29-23  |       |        |      |
| AT\$EXT  | 1-82  | 8-108*  |         |        |        |        |       |        |      |
| AT\$FIL  | 1-82  | 8-106*  | 8-107*  |        |        |        |       |        |      |
| AT\$LOG  | 1-82  | 8-102*  | 10-23   | 10-25* |        |        |       |        |      |
| AT\$SIZ  | 1-82  | 8-103*  |         |        |        |        |       |        |      |
| AUTHFN   | 1-136 | 40-115  |         |        |        |        |       |        |      |
| BADBOT   | 1-148 | 44-72   |         |        |        |        |       |        |      |
| BADCMD   | 1-118 | 8-15    | 29-12   |        |        |        |       |        |      |
| BADDAT   | 37-16 | 37-18   | 37-22   | 37-33  | 37-37  | 37-41  | 37-43 | 37-49# |      |
| BADTIM   | 38-16 | 38-20   | 38-24   | 38-35  | 38-49# |        |       |        |      |
| BDFNAM   | 1-121 | 18-99   | 23-36   | 24-17  | 36-37  |        |       |        |      |
| BDLIN    | 1-134 | 33-34   | 35-25   | 43-29  |        |        |       |        |      |
| BELL     | 2-7#  | 33-49   |         |        |        |        |       |        |      |
| BLANK    | 2-6#  | 33-65   |         |        |        |        |       |        |      |
| BLKO     | 1-117 | 31-32   | 31-34   | 40-122 | 40-125 | 40-149 | 41-48 | 41-53  | 42-7 |
| BLKWDS   | 2-10# |         |         |        |        |        |       |        |      |
| BOTDEV   | 1-78  | 44-10*  | 44-22*  | 44-23* | 44-24* | 45-8*  |       |        |      |
| C. CSW   | 1-88  | 27-27   |         |        |        |        |       |        |      |
| C. DEVQ  | 1-88  | 27-31   |         |        |        |        |       |        |      |
| CASCBR   | 1-75  | 39-32   |         |        |        |        |       |        |      |
| CASCUP   | 1-75  | 39-35   |         |        |        |        |       |        |      |
| CASTBR   | 1-75  | 39-31   |         |        |        |        |       |        |      |
| CASTBW   | 1-75  | 39-34   |         |        |        |        |       |        |      |
| CASTRO   | 1-76  | 39-30   |         |        |        |        |       |        |      |
| CASTWO   | 1-76  | 39-33   |         |        |        |        |       |        |      |
| CCLSAV   | 1-73  | 27-59   |         |        |        |        |       |        |      |
| CD\$\$SZ | 1-89  | 26-115  |         |        |        |        |       |        |      |
| CD\$BAS  | 1-89  | 26-96   |         |        |        |        |       |        |      |
| CD\$DVU  | 1-89  | 26-94   |         |        |        |        |       |        |      |
| CDBUF    | 1-47  | 26-94   | 26-96   |        |        |        |       |        |      |
| CDGET    | 1-47  | 26-93   |         |        |        |        |       |        |      |
| CFBUF    | 1-73  | 40-56   |         |        |        |        |       |        |      |
| CFCHAN   | 1-95  | 40-49   | 40-49   | 40-56  | 40-56  |        |       |        |      |
| CFPNT    | 1-96  | 7-8     |         |        |        |        |       |        |      |
| CFSPND   | 1-100 | 7-8*    | 7-35    | 7-37*  |        |        |       |        |      |
| CFSTOP   | 1-41  | 7-10    |         |        |        |        |       |        |      |







Cross reference table (CREF V05.05)

|         |       |        |         |         |         |        |       |       |       |       |      |       |
|---------|-------|--------|---------|---------|---------|--------|-------|-------|-------|-------|------|-------|
| INVLDM  | 1-138 | 23-50  |         |         |         |        |       |       |       |       |      |       |
| INVLDN  | 1-121 | 23-15  |         |         |         |        |       |       |       |       |      |       |
| INVOPT  | 1-115 | 12-26  | 22-21   | 28-30   | 29-67   | 35-19  |       |       |       |       |      |       |
| INVTIM  | 1-135 | 38-50  |         |         |         |        |       |       |       |       |      |       |
| JOB COM | 43-6  | 43-13  | 43-23#  |         |         |        |       |       |       |       |      |       |
| JS\$OFF | 1-61  | 40-159 |         |         |         |        |       |       |       |       |      |       |
| KCSIBF  | 1-118 | 8-63   | 8-81    | 8-84*   | 8-87*   | 8-89   | 8-102 | 8-103 | 8-104 |       |      |       |
| KCSIMS  | 1-119 | 8-69   |         |         |         |        |       |       |       |       |      |       |
| KILEMT  | 1-136 | 40-183 | 43-5    | 44-59   |         |        |       |       |       |       |      |       |
| KMNCHN  | 1-73  | 27-58  |         |         |         |        |       |       |       |       |      |       |
| L       | 5-6   | 5-6#   | 5-7     | 5-7#    | 5-8     | 5-8#   | 5-9   | 5-9#  | 5-10  | 5-10# | 5-11 | 5-11# |
|         | 5-12  | 5-12#  | 5-13    | 5-13#   | 5-14    | 5-14#  | 5-15  | 5-15# | 5-16  | 5-16# | 5-17 | 5-17# |
|         | 5-18  | 5-18#  | 5-19    | 5-19#   | 5-20    | 5-20#  | 5-21  | 5-21# | 5-22  | 5-22# | 5-23 | 5-23# |
|         | 5-30  | 5-30#  | 5-31    | 5-31#   | 5-38    | 5-38#  | 5-39  | 5-39# | 5-40  | 5-40# | 5-41 | 5-41# |
|         | 5-48  | 5-48#  | 5-49    | 5-49#   | 5-56    | 5-56#  | 5-57  | 5-57# | 5-64  | 5-64# | 6-7  | 6-7#  |
|         | 6-8   | 6-8#   | 6-9     | 6-9#    | 6-10    | 6-10#  | 6-11  | 6-11# | 6-12  | 6-12# | 6-13 | 6-13# |
|         | 6-14  | 6-14#  | 6-15    | 6-15#   | 6-16    | 6-16#  | 6-17  | 6-17# | 6-18  | 6-18# | 6-19 | 6-19# |
|         | 6-20  | 6-20#  | 6-21    | 6-21#   |         |        |       |       |       |       |      |       |
| LCONTM  | 1-71  | 40-138 |         |         |         |        |       |       |       |       |      |       |
| LCPUHI  | 1-71  | 40-96  |         |         |         |        |       |       |       |       |      |       |
| LCPULO  | 1-71  | 40-97  |         |         |         |        |       |       |       |       |      |       |
| LD\$RON | 1-93  | 23-74  |         |         |         |        |       |       |       |       |      |       |
| LDBASE  | 1-94  | 26-85  |         |         |         |        |       |       |       |       |      |       |
| LDCLN   | 1-58  | 23-116 | 25-81   |         |         |        |       |       |       |       |      |       |
| LDDEVN  | 23-13 | 23-46  | 24-35#  |         |         |        |       |       |       |       |      |       |
| LDDEVX  | 1-95  | 18-55  | 26-66   | 26-76   |         |        |       |       |       |       |      |       |
| LDFLAG  | 1-94  | 23-75* |         |         |         |        |       |       |       |       |      |       |
| LDMNT   | 1-55  | 23-80  |         |         |         |        |       |       |       |       |      |       |
| LDNAME  | 1-94  | 23-62  | 23-89   | 25-75*  |         |        |       |       |       |       |      |       |
| LDPDEV  | 1-94  | 25-72* | 26-84   |         |         |        |       |       |       |       |      |       |
| LDSIZE  | 1-94  | 26-87  |         |         |         |        |       |       |       |       |      |       |
| LF      | 2-4#  | 7-29   | 33-48   | 33-85   | 40-93   | 44-49  |       |       |       |       |      |       |
| LF\$WRT | 1-112 | 40-72  |         |         |         |        |       |       |       |       |      |       |
| LINFRE  | 1-135 | 35-41  |         |         |         |        |       |       |       |       |      |       |
| LNPRIM  | 1-86  | 33-32  | 33-51   | 33-121  | 40-82   |        |       |       |       |       |      |       |
| LNUMER  | 33-31 | 33-34# | 33-37   | 33-91   |         |        |       |       |       |       |      |       |
| LOFSPC  | 1-84  | 40-45  | 40-49   | 40-53*  | 40-61*  |        |       |       |       |       |      |       |
| LOGASN  | 1-121 | 18-44  | 23-41   |         |         |        |       |       |       |       |      |       |
| LOGCHK  | 1-101 | 23-21  | 25-43   | 26-123  |         |        |       |       |       |       |      |       |
| LOGCLS  | 1-125 | 40-154 |         |         |         |        |       |       |       |       |      |       |
| LOGFLG  | 1-111 | 40-72* |         |         |         |        |       |       |       |       |      |       |
| LPARNT  | 1-54  | 34-41  | 34-43   |         |         |        |       |       |       |       |      |       |
| LPROG   | 1-70  | 40-121 | 40-203* |         |         |        |       |       |       |       |      |       |
| LPROJ   | 1-70  | 40-107 | 40-202* |         |         |        |       |       |       |       |      |       |
| LSECPT  | 1-74  | 41-25  |         |         |         |        |       |       |       |       |      |       |
| LSTD L  | 1-69  | 35-26  | 40-84   | 40-173  |         |        |       |       |       |       |      |       |
| LSTPL   | 1-102 | 33-119 | 40-15   | 40-171  |         |        |       |       |       |       |      |       |
| LSTSL   | 1-104 | 33-30  | 40-180  | 43-27   | 44-56   |        |       |       |       |       |      |       |
| LSTSPL  | 1-50  | 32-54  |         |         |         |        |       |       |       |       |      |       |
| LSW     | 1-49  | 33-36  | 33-97   | 33-122  | 33-124  | 40-60* | 40-80 |       |       |       |      |       |
| LSW3    | 1-83  | 40-23  | 40-33*  | 40-41*  | 40-83*  |        |       |       |       |       |      |       |
| LSW4    | 1-97  | 7-21   | 33-103  | 33-132  | 40-58*  |        |       |       |       |       |      |       |
| LSW5    | 1-67  | 7-15   | 40-21   |         |         |        |       |       |       |       |      |       |
| LSW6    | 1-108 | 7-17*  | 7-34*   |         |         |        |       |       |       |       |      |       |
| LSW7    | 1-110 | 33-101 | 33-128  |         |         |        |       |       |       |       |      |       |
| LSW9    | 1-66  | 40-25  | 40-59*  | 40-170* | 40-198* |        |       |       |       |       |      |       |









Cross reference table (CREF V05.05)

|        |        |         |        |        |        |        |        |       |       |       |       |       |  |
|--------|--------|---------|--------|--------|--------|--------|--------|-------|-------|-------|-------|-------|--|
| SPNHEM | 4-34#  | 32-82*  | 32-83  |        |        |        |        |       |       |       |       |       |  |
| SPOLGO | 31-73  | 32-135# |        |        |        |        |        |       |       |       |       |       |  |
| SPSNG  | 1-131  | 32-16   |        |        |        |        |        |       |       |       |       |       |  |
| SPWFM  | 1-131  | 32-10   |        |        |        |        |        |       |       |       |       |       |  |
| SPWIDE | 1-134  | 32-46   |        |        |        |        |        |       |       |       |       |       |  |
| SQZDEV | 4-5#   | 26-37*  | 26-74  | 26-122 |        |        |        |       |       |       |       |       |  |
| STPASK | 1-137  | 44-43   |        |        |        |        |        |       |       |       |       |       |  |
| STPFLG | 1-72   | 40-19   | 40-89  | 40-164 | 44-55* | 45-7*  |        |       |       |       |       |       |  |
| SWPCHN | 1-38   | 27-55   |        |        |        |        |        |       |       |       |       |       |  |
| SYFLVC | 27-24  | 27-55#  |        |        |        |        |        |       |       |       |       |       |  |
| SYSDAT | 1-105  | 37-46*  |        |        |        |        |        |       |       |       |       |       |  |
| SYTIMH | 1-105  | 38-45*  |        |        |        |        |        |       |       |       |       |       |  |
| SYTIML | 1-105  | 38-46*  |        |        |        |        |        |       |       |       |       |       |  |
| TAB    | 2-8#   |         |        |        |        |        |        |       |       |       |       |       |  |
| TALEMT | 1-68   | 22-42   |        |        |        |        |        |       |       |       |       |       |  |
| TIMSPL | 4-40#  | 40-193  |        |        |        |        |        |       |       |       |       |       |  |
| TIMUP  | 40-94  | 40-107# |        |        |        |        |        |       |       |       |       |       |  |
| TK1SEC | 1-106  | 40-98   |        |        |        |        |        |       |       |       |       |       |  |
| TM#SA1 | 1-37   | 41-40   |        |        |        |        |        |       |       |       |       |       |  |
| TM#SA2 | 1-37   | 41-48   |        |        |        |        |        |       |       |       |       |       |  |
| TM#SA3 | 1-37   | 41-42   |        |        |        |        |        |       |       |       |       |       |  |
| TM#SA4 | 1-37   | 41-44   |        |        |        |        |        |       |       |       |       |       |  |
| TMIDLH | 1-58   | 39-29   |        |        |        |        |        |       |       |       |       |       |  |
| TMIOH  | 1-58   | 39-26   |        |        |        |        |        |       |       |       |       |       |  |
| TMIOWH | 1-55   | 39-28   |        |        |        |        |        |       |       |       |       |       |  |
| TMSWPH | 1-58   | 39-27   |        |        |        |        |        |       |       |       |       |       |  |
| TMSWTH | 1-58   | 39-25   |        |        |        |        |        |       |       |       |       |       |  |
| TMTOTH | 1-55   | 39-23   |        |        |        |        |        |       |       |       |       |       |  |
| TMUSRH | 1-55   | 39-24   |        |        |        |        |        |       |       |       |       |       |  |
| TOTON  | 1-72   | 40-199  |        |        |        |        |        |       |       |       |       |       |  |
| TSKM2A | 1-5#   | 1-26    |        |        |        |        |        |       |       |       |       |       |  |
| UFORM  | 1-79   | 32-116  | 32-120 | 32-123 |        |        |        |       |       |       |       |       |  |
| VLDSYS | 1-145  | 23-6    |        |        |        |        |        |       |       |       |       |       |  |
| WLDNAM | 1-42   | 2-11#   | 18-24  | 18-69  | 18-71  |        |        |       |       |       |       |       |  |
| XAREA  | 1-115  | 14-16   | 14-24  | 14-37  | 23-91  | 31-18  | 31-27  | 31-32 | 31-34 | 36-11 | 40-49 | 40-56 |  |
|        | 40-115 | 40-122  | 40-149 | 40-193 | 44-19  | 44-21  |        |       |       |       |       |       |  |
| YELEMT | 1-135  | 33-7*   | 33-21* | 33-105 | 33-109 | 33-130 | 33-134 |       |       |       |       |       |  |

