

6-	1	SET commands
6-	2	Process
14-	1	Terminal
24-	1	CL
33-	1	Host

```

1          .TITLE  TSKST2 -- Keyboard SET Command routines
2          .ENABL  LC
3          .DSABL  GBL
4 000000   .CSECT  TSKST2
5 000000
6          TSKST2:
7          ;
8          ; TSKST2 is the portion of TSKMON that contains the code
9          ; to implement the SET command.
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, .TTOUTR, .SRESET
18         .MCALL .READW, .TTYIN, .TTYOUT, .PURGE
19         .MCALL .CSIGEN, .SAVEST, .REOPEN
20         .MCALL .GTLIN, .GTIM, .DATE, .SPFUN
21         .MCALL .PRINT, .CLOSE, .LOOKUP
22         .MCALL .WRITW, .ENTER, .EXIT
23         .MCALL .SERR, .HERR, .FPROT, .GVAL, .PVAL
24         ;
25         ; Global definitions
26         ;
27         .GLOBL SETCL, SETHST, SVT227, SVT228
28         .GLOBL SETPRC, SETTTY, STVRFY, STNOVR, SVT50, SVT100, SVT200
29         .GLOBL TSKST2, CMDHD, CMDOFF, KDOCIN, SKPSPC, UCLCMD
30         .GLOBL DORUN, CMDFRM, CMDDSN, STLGCN, DATTIM, PRGALL
31         .GLOBL DLCEMT, ALCDEV, CMDSHD, CMDSET, CMDWHO, CMDMEM, CMDUSE
32         ;
33         ; Global references
34         ;
35         .GLOBL EM$LAS, EM$LFD, EM$CLU, LWINDO, CLSFRS, LCXTBL, MXTTCT, EM$TMT
36         .GLOBL PA$GRC, PA$UKC, PA$DSC, PA$BLD, PA$ULN, PA$DWD, PA$HQL, PA$LET
37         .GLOBL EM$PTA, EM$PTU, SBPSUF, CHKCLU, EM$CLX, PA$BEL, EM$OPR, PA$NWD
38         .GLOBL SCNOPS, TM$XBK, CL$XLN, CKCLUS, ACRSTR, R5OKMN, SJEMT, RJEMT
39         .GLOBL PEKEMT, PEKADR, PEKSIZ, TM$NNR, CDBUF, CDGET, TM$IN1, TM$IN2
40         .GLOBL SYPSWD, EM$SPL, ACRTXT, JPWDEV, JPWTYP, JPWFLG, EM$BYP
41         .GLOBL EM$WCO, EM$WC1, EM$WC2, EM$WC3, EM$WCM, P2$CGR
42         .GLOBL CXTRMN, R$CHN, R$XCHN, CHNSIZ, C. USED, CL$EPN, CL$EPS, CLEOFS
43         .GLOBL EM$STL, EM$IST, CLSFEP, EM$IDR, VPRIDF, SETWRD
44         .GLOBL CLRPRV, OPTLST, PFSO, PFCO, PVNPW, PO$SPV, PRIVSO, PRVOPT
45         .GLOBL TM$PVA, TM$PVC, PRIVAO, EM$CNO, EM$CPD, EM$CAP, RSTPRV
46         .GLOBL CHKEQ, CKACQJ, PO$OPR, CKSYPV, P2$TRM, PRIVC2, PO$NAM, EM$NPR
47         .GLOBL INSTBL, INGADR, INGEMT, IIBUF, II$NAM, II$FLG, II$$SZ, EM$NAD
48         .GLOBL INSTBN, AF$SCA, AF$NOW, AF$MEM, PO$DBG, PRIVCO, PRVLST
49         .GLOBL ABRTAD, ABRTCD, CINFLG, $VNOTT, II$PRV, II$NPV, PO$BYP
50         .GLOBL TM$RD1, TM$RD2, TM$LCL, TM$GBL, SPACE1, RC$DOWN, RC$CNT
51         .GLOBL RC$EXC, RC$AGE, RC$AEP, RC$USE, RC$FLG, RC$GBL, RC$NAM
52         .GLOBL RC$LEN, RC$PVT, RCBBAS, RCBEND, RC$$SZ, SHRRCB, SHRRCN
53         .GLOBL EM$NPD, EM$IILN, EM$CIP, EM$NSF, EM$IUN, EM$CLN, EM$IVN
54         .GLOBL EM$IILN, EM$ACL, EM$TSL, EM$CLB, EM$NSL, EM$SLT, EM$SLW
55         .GLOBL SLKDON, SLKDOF, EM$UID, TM$PR1, TM$PR2, TM$LPR, TM$HPR
56         .GLOBL TM$HPE, TM$CNG, TM$CDS, TM$CEN, TRMHD1, TRMHD2, AT$DEV
57         .GLOBL OPRTXT, CLLINE, LCLTXT, REMTXT, TM$AUT, CLFREE, CLUNIT, CLVERS

```

```

58 . GLOBL TM$CLO, TM$CL1, TM$CL2, TM$CL3, TM$CL4, TM$CL5, TM$CL6
59 . GLOBL QHDMS1, QHDMS2, DVSHH1, DVSHH2, DVSHH3, SYASHD, DKASHD
60 . GLOBL TM$NAD, ALCHD1, ALCHD2, TM$NSD, TM$SDN, LNAME
61 . GLOBL CORUSR, LSW, $CTRL0, SERFLG, IOABFL, $CHACT, $STSNG
62 . GLOBL LSTHL, LCLUNT, FSTIOL, LSTIOL, CL$LIX, CW$PRO, CONF02
63 . GLOBL CL$RQH, CL$WQH, MAXALC, ALCTBL, ALCEND
64 . GLOBL AD$DVU, AD$JOB, AD$$SZ, UCIDEF, HANCHN
65 . GLOBL NEDCHR, LOUTIR, LINIR, LINRTS, CLOTIR
66 . GLOBL CO$DEF, CL$COL, LCDTYP, SOPALC, SOPDAT, SOPTIM
67 . GLOBL UTRPAD, JSWLOC, ERRLOC, MAXMEM, MAXPRI
68 . GLOBL USRSTK, $KINIT, CFSTK, MXJMEM, DFJMEM, EM$HNI
69 . GLOBL SPUBUF, SXBPNT, MXJADR, CLSFCH
70 . GLOBL TMTOTH, TMTOTL, TMUSRH, TMIOWH, LDMNT, EM$CSE
71 . GLOBL TMSWTH, TMIDLH, TMIOH, TMSWPH, LDCLEN
72 . GLOBL WILDFL, $NOIN, $NOWTT, $HITTY
73 . GLOBL TECO, EDIT, KED, K52, $1STLG, $DIBOL
74 . GLOBL SH$VAL, SH$NAM, SH$$SZ, SH$RTN, SH$FLG
75 . GLOBL SO$NVL, SO$OCT, SO$NO, HANENT, HANSIZ
76 . GLOBL H. CSR, H. VEC, DVSTAT, SFID, ACRSPD, HANPAR, LSTSPL
77 . GLOBL HAZEL, HAZLFL, HAZLNO, $MLOCK, MDT, GETKCH
78 . GLOBL LINBUF, LINNXT, LSTACT, PRGTOP, PRGSIZ, KMNHI
79 . GLOBL KMNTOP, KMNPOS, KMNSTK, KMNSTR, CXTPAG, FSTIOL
80 . GLOBL LINPNT, LINCNT, LACTIV, LRDTIM, CS$RON
81 . GLOBL LOTBUF, LOTNXT, LOTPNT, $VTESE
82 . GLOBL LOTSIZ, LOTSPC, LCOL, $SLKED, ESC, VDBFLG
83 . GLOBL LAFSIZ, LFWLIM, LINCUR, NUMON, ILSW2
84 . GLOBL $DBKMN
85 . GLOBL $CARUP, DOASGN, UKMNAM, $UKMON, LSW9
86 . GLOBL LSUCF, $CCLRN, VLDSYS, EM$NUK, S$QMIO
87 . GLOBL KL3CLR, $PRGLK, LSW5, PVON, S$SPND, $AUTO
88 . GLOBL S$TWFN, S$TTFN, S$OTFN, S$IOFN, S$OTLO
89 . GLOBL LSTDJ, FSTDJ, $DETCH, UMSYTP, S$TTSC
90 . GLOBL $DISCN, LPROJ, LPROG, LUNAME, S$RT, S$LOW
91 . GLOBL LCPUIH, LCPULO, LCONTM, $CTRLS, $SPLJB, TXTCL
92 . GLOBL STPFLO, TONON, USPLCH, SPLCHN, S$HICP
93 . GLOBL S$INWT, S$OTWT, S$TMWT, S$SFWT, S9600
94 . GLOBL S$MSWT, CFBUF, CFEND, CCLSAV, KMNCHN
95 . GLOBL MINTIM, LSECPT, MAXSEC, $EMTTR, VCSHNB
96 . GLOBL OKFILE, OKFEND, $CLTST, UCISPC, MHNSIZ
97 . GLOBL CASTBR, CASCBR, CASTBW, CASCUP, MHNSMS
98 . GLOBL CASTRO, CASTWO, CLTOTL, CO$DTR, CLSFSP
99 . GLOBL CO$CR, CO$FF, CO$FFO, CO$LC, CO$TAB, CO$CTL
100 . GLOBL CO$LFI, CO$LFO, CO$BNI, CO$BNO, CL$OPT
101 . GLOBL CL$LEN, CL$SKP, CL$WID, CL$LIN, PHYMEM
102 . GLOBL LJSW, CTRLTT, NEWJSW, JSTKND, VIMAGE
103 . GLOBL USTART, GENTOP, BOTDEV, BOTUNI, CSHALC
104 . GLOBL $CTRLC, LSW2, $INKMN, CHAIN, UFORM
105 . GLOBL $SG00, $SG03, LITIME
106 . GLOBL MAXASN, $CFABT, INDSTA, INDERR
107 . GLOBL RUNDEV, LNBLKS, CXTBAS, CXTWDS, UHIMEM
108 . GLOBL $DILUP, CSHDEV, CSHDVN, LNSBLK
109 . GLOBL LSW3, LSW2S, $DUPRN
110 . GLOBL $FORM, $TAB, LSCCA, $CFSOT, LOFSPC, R50COM
111 . GLOBL $PAGE, $SCOPE, $ECHO, $LC, $BBIT, CHKALC, $ALTER
112 . GLOBL UCHAN, $FORMO, $CFALL, $CFDCC, $CFCC
113 . GLOBL LNPRIM, LNMAP, CW$50H, CONFIG, $SUCF
114 . GLOBL $DOOFF, NUCHN, LRBFIL, CFIND, TALEMT

```

```

115 . GLOBL C. CSW, C. DEVQ, C. SBLK, NLINES, CO$BBT
116 . GLOBL CD$NAM, CD$DVU, CD$BAS, CD$JOB, CD$$SZ, CD$$UB
117 . GLOBL LTSCMD, LNSPAC, CFNEST, UCLNAM
118 . GLOBL $CFOPN, CFSEND, PBFEND, CFSP, $TTGAG
119 . GLOBL UFPTRP, SDSFCB, SD$DEL, CFLFL4, $UCLCF
120 . GLOBL SDFLAG, SD$FLK, SD$WFM, SDFORM, $UCLRN
121 . GLOBL SDBUF1, SDBLK, NSPLDV, LD$RON, $UCLCM, $UCLCL
122 . GLOBL LDNAME, LDSIZE, LDFLAG, LDBASE, LDPDEV
123 . GLOBL LSW8, $SQQ1, $SQQ1A, $SQQ1B, $SQQ1C, $SQQ2, $SGIIO, $SGHIO
124 . GLOBL $DEFER, CFCHAN, SCHAIN, LDDEVX, $SGALL
125 . GLOBL CFPNT, CFBLK, $QUIET, DIABFL
126 . GLOBL DIABNO, VT52NO, LA36NO, LA36FL
127 . GLOBL LSW4, KL4CLR, SDSKIP, SDBU, SD$BAK
128 . GLOBL $INCOR, $KED, VQUN1B, VINTIO, VQUN1C
129 . GLOBL SF$BSY, SFFORM, SD$SNG, SFNMBL, NFRESB
130 . GLOBL SD$HLD, SF$HLD, CURPRM, PRMPNT, SF$1ST
131 . GLOBL LSTPRM, PRMBUF, PRMEND, CFSPND
132 . GLOBL SDFHD, SFFLAG, SFQLNK, CFHOLD, LOGDVU, LOGBAS
133 . GLOBL LCOL, $QTSET, $TECO, CD$TOP, LOGCHK
134 . GLOBL $WILD, ERRSEV, UERSEV, PASLIN, LOGBAS, LOGDVU
135 . GLOBL LSTPL, SDCB, SDCBND, VQUANO, VQUAN3
136 . GLOBL VQUAN1, VQUN1A, VQUAN2, VHIPCT, VQUANO, VQUAN3
137 . GLOBL DCTRD, DCCRD, DCTWR, DCCWR, ASNSRC
138 . GLOBL VCORTM, NUMDCD, VNUMDC, KMPRMT, MXPRMT
139 . GLOBL RDB, RDBEND, RT$NAM, RT$$SZ, CLDEVX, SDDVU
140 . GLOBL SDNAME, SDCBSZ, LSTSL, LSTATE
141 . GLOBL TK1VAL, CINDAT, SYSDAT, SYTIMH, SYTIML
142 . GLOBL BASMAP, LOMAP, HIMAP, JCXPGS
143 . GLOBL SMRSIZ, SRTSIZ, CSHSIZ, TK1SEC
144 . GLOBL TSXLN, TSXSIT, GRT1, TRGRET, LICTXT, SUPCOD, NAMTOP, SUMS, SUCS
145 . GLOBL LPRG1, LPRG2, S$QUSR, S$IOWT, S$SFWT
146 . GLOBL S$SPDB, S$SPCB, SFUSER, SFFILE, VT200, VT2007, VT2008
147 . GLOBL LCBIT, LA36, LA120, VT52, VT100, DIABLO, QUME
148 . GLOBL ADM3A, LTRMTP, LA12FL, LA12NO, VT52FL
149 . GLOBL VT10FL, VT10NO, QUMEFL, QUMENO, ADM3FL
150 . GLOBL VT20FL, VT20NO
151 . GLOBL ADM3NO, SYINDEX, SYUNIT, NUMDEV, PNAME
152 . GLOBL OF$DEV, OF$UNT, OF$FIL, OF$FLG, SYNAME
153 . GLOBL OF$$SZ, OT$RON, RESDEV, $TAPE
154 . GLOBL KMNBAS, ODTBAS, $CTRLD
155 . GLOBL LSW6, $SNWTT, PF$SYS, PF$IOW, $DEBUG
156 . GLOBL RSR, TSR, LMXNUM, LSTMX, MXDTR, ZCLR, MXCSR
157 . GLOBL $INDDF, $INDRN, IN$ACT, IN$CNT, IN$CMD, INDSAV
158 . GLOBL $PHONE, INVEC, LMXLN, MXVEC, $INIT, $DEAD, $HARD
159 . GLOBL ITRMTP, LMXPRM, LSW7, $INDAB, CFSTS, CF$IND, CF$QUT
160 . GLOBL CFABLV, MONVEC, LBSPRI, MAXPRI, MXJPRI, LPRI, $SYSPS
161 . GLOBL LOGCHN, LOGFLG, LOGPTR, LOGBUF, LOGBLK
162 . GLOBL LF$OPN, LF$WRT, UCLBLK, UCLDAT
163 . GLOBL CSHHD, FC$CDX, FC$LNK, FD$NAM, UC$NDC, UC$MDC, CVTUC
164 . GLOBL CMDBUF, PAUMSQ, RDCMD, DKSAV, SYSAV, CVTTAB, RUNHD, SEARCH
165 . GLOBL INVOPT, FKILL, ABRTCF, ACRFN, XAREA, FILNAM, NOPRG, FPRINT
166 . GLOBL PUSHCF, TRMSTR, FILNAM, R50DIR, R50SY, R50IND, R50SAV
167 . GLOBL INDACT, R50DUP, R50PIP, R50KED, R50K52, R50KEX, R50TSX, R50UCL
168 . GLOBL BLKO, RDERM, R50VIR, NOSTRT, RUNEMT, DVRCOR
169 . GLOBL BADSAV, LDNAM, NOPRG, NDCIN, SIZVAL, ASKLNM, BADCMD, KCSIBF
170 . GLOBL ASDEX, KCSIMS, ASNOVF, GTRD50, R50BUF, R50LDO, MNTDEV, DMTARG
171 . GLOBL DEADEV, CHKMNT, CHKMTX, INFOMT, NOFLAG, MTOPHD, INVOPT, ILLCMD

```

```

172 . GLOBL R5OLD, INVLDN, R5ODSK, ACRFIL, BDFNAM, LOGASN, MNTFUL, R5OLD7
173 . GLOBL TBLOVF, SETHD, CSIMS2, CKPRIV, R5OND, AMBOPT, ACRDEC
174 . GLOBL MAXAVL, PRTDEC, DEVUNT, PNAME, HNBUF, CKTERM
175 . GLOBL ACROCT, HANBSY, CSIMS1, MISSEQ, NOIND, POPCF
176 . GLOBL BADPMT, BADPRI, TOTXT, CRLF, HIPRI, STLGH, LOGCLS, R5OLOG
177 . GLOBL BDLGOP, SPLHLA, NOCCL, LDOPHD, PRTPFX, PRTSPC
178 . GLOBL DLTXT, OCTFIX, PRTPPT, NATXT, SPDTX1, NOTXT, YESTXT, NINTXT
179 . GLOBL PRTUNM, SYHD1, SYHD2, PRTLN, SPACE2, DETTXT, SPACE3, RNMS
180 . GLOBL SWPTX, LOCKTX, SPACE5, PRTDC3, KBMSG, DIVIDE, PRTDC2
181 . GLOBL COLOO, CPUAH, CPUAL, PRRTMV, NOFIL, CMDBUF, CALUCL
182 . GLOBL NOUDC, DEVHD1, ASNHD1, ASNHD2, SHMTH1, SHMTH2, PRRTMD
183 . GLOBL CVDVNM, SPACE6, PRTPBU, PRTPFM, NONEMS, NODAT, NOLDMT
184 . GLOBL SUBARO, EDTFIL, RONTXT, NOTAVL, KBTX, MNFLGS, MNBPC
185 . GLOBL DELSPC, MNBASE, MNTOP, MONHD, MONAR1, NOPMGN, PMBUSY, MONAR2
186 . GLOBL NSWPMS, MAXMTX, CURMTX, CHKDLM, SPLHD, AMBOPT, INVOPT
187 . GLOBL DEVIDL, COAL, ALDEX, COAD, SPACTV, SPWFM, DEVIDL, SPSNG
188 . GLOBL COAL, ALDEX, ALDBLK, COAD, SPACTV, SPWFM, DEVIDL
189 . GLOBL SPSNG, SPFUL, SPCF, SPFLK, NOFIL, SPGEMT, NOOPTT
190 . GLOBL BDLIN, MSGBUF, MSGEND, NOTON, GAGMSG
191 . GLOBL LINFRE, DJABMS, DLMSG, INVTIM, DMTALL
192 . GLOBL SHTMSG, AUTHFN, SPLACT, DOSTOP, OFFEMT, KILEMT, UPTMMS
193 . GLOBL TMTOTH, DIVSOR, TMTOTL, PRTPCT, SUM1, SUM2, SUM3, SUM4
194 . GLOBL SUM5, SUM6, SUM7, OTHRON, SPLPND, STPASK, SRTSMS, CHKTTD
195 . GLOBL SIZEMT, ASNOVF, INVLDN, CSIMS4, MNTARG, HUPARG, R5OTT
196 . GLOBL KMNNAM, NOKMON, CCLNAM, OTRMNT, CHKDEV, DMTSUB, CMDCCCL
197 . GLOBL SHOHD, SUBTXT, MNTTXT, SRTTXT, TOTMMS, UMSSMS, SSRMAP
198 . GLOBL TSXSMS, USRMMS, JCSMS, DZTXT, OCTPRT
199 . GLOBL PRTR50, PRTPAT, PRTPOD, PRTPIM, INVDEV, ALFN, R5ODK
200 . GLOBL DETHD, DETARG, RUNMS, NOFRDL, R5OMON, INV DAT, MUL32, COAF
201 . GLOBL AR$PRJ, AR$PRG, AR$CON, AR$CNT, AR$CPH, AR$CPL, AR$UNM
202 . GLOBL AR$DMY, AR$$SZ, ARNRPB, $SLON, $SLTTY, $SLLET
203 . GLOBL PRTWRN, SLMXLN, VSLEDT, $LOFCF, CSHMSG
204 . GLOBL AF$HIE, AF$NOI, $NOINT, AF$PLK, AF$DBG
205 . GLOBL AF$IOP, $RNIOP

```

```
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         000054      COMMA   =      54      ; COMMA
11         000400      BLKWDS  =      256     ; # OF WORDS IN DISK BLOCK
12         132500      WLDNAM  =      132500 ; RAD50 /*/ (WILDCARD)
```

```

1      ; -----
2      ; Macro to cause a fatal error message to be printed.
3      ;
4      .MACRO FERR MSG
5      MOV R5, -(SP)
6      MOV MSG, R5
7      CALL FPRINT
8      MOV (SP)+, R5
9      .ENDM FERR
10     ;
11     ; -----
12     ; Macro to print a fatal error message, clean up
13     ; and then jump to RDCMD.
14     ;
15     .MACRO FABORT MSG
16     MOV MSG, R5
17     JMP FKILL
18     .ENDM FABORT
19     ;
20     ; -----
21     ; Macro to print a warning message
22     ;
23     .MACRO FWARN MSG
24     MOV R5, -(SP)
25     MOV MSG, R5
26     CALL PRTWRN
27     MOV (SP)+, R5
28     .ENDM FWARN
29     ;
30     ; -----
31     ; Macro to start a standard option table.
32     ; Name = 1 to 4 character table name.
33     ; NA = Number of arguments per table entry.
34     ;
35     .MACRO TBLDEF NAME, NA
36     NARGS = NA
37     .CSECT CMDVS2
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 NAMES2
49     L =
50     .ASCIZ /STRNG/
51     .CSECT CMDVS2
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  CMDVS2  
    .WORD   0  
    .CSECT  TSKST2  
    .ENDM   TBLEND
```



```

1          ; -----
2          ; Data areas
3          ;
4 000000 125017 074773          R50HST: .RAD50  /$HOST$/
5 000004 012256 000000 000000 CLDEV:  .RAD50  /CLN      /
   000012 000000
6 000014 012276          R50CLO: .RAD50  /CLO/
7 000016 013666          R50C10: .RAD50  /C10/
8 000020 000000          CPFLAG: .WORD   0
9          ;
10         ; Flags stored in CPFLAG
11         ;
12         CPFSUS =          1          ; Suspend process
13         CPFRES =          2          ; Resume process
14         CPFAUT =          4          ; Set authorized privileges
15         ;
16 000022          000          SETPRM: .BYTE   0          ; Indicates if SET is temp or perm
17 000023          000          CPPID:  .BYTE   0          ; Process ID for SET PROCESS command
18 000024          000          CPPRID: .BYTE   0          ; Priority value for SET PROCESS command
19 000025          000          CPNAME: .BLKB  12.         ; Name buffer for SET PROCESS command
20         .EVEN
21         ;
22         ; Emt to assign a CL line to a time-sharing line
23         ;
24 000042          000          155          CLAEMT: .BYTE   0,155
25 000044 000000          .WORD   0          ; CL unit number
26 000046 000000          .WORD   0          ; Line number
27         ;
28         ; Emt to clear XOFF status for a CL line (SET CLn XONBYPASS)
29         ;
30 000050          001          155          CLXEMT: .BYTE   1,155
31 000052 000000          .WORD   0          ; CL unit number
32         ;
33         ; Emt to completely reset a CL line (SET CLn RESETBYPASS)
34         ;
35 000054          002          155          CLREMT: .BYTE   2,155
36 000056 000000          .WORD   0          ; CL unit number
37         ;
38         ; Emt to cross connect our time-sharing line with a CL unit
39         ;
40 000060          000          126          TTXCL:  .BYTE   0,126
41 000062 000013          .WORD   13
42 000064 000000          .WORD   0
43         ;
44         ; Emt to set line speed
45         ;
46 000066          000          154          LSPEMT: .BYTE   0,154
47 000070 000000          .WORD   0          ; Line number
48 000072 000000          .WORD   0          ; Speed code
49         ;
50         ; Emt to reset XOFF status for a TT line
51         ;
52 000074          001          154          XONEMT: .BYTE   1,154
53 000076 000000          .WORD   0
54         ;
55         ; EMT to set/reset DTR for a line
56         ;

```

```

57 000100      002      154      DTREMT: .BYTE  2,154      ; Assume raise DTR (3,154 = drop DTR)
58 000102      000000      .WORD  0
59
60      ; Emt to set privilege flags
61
62 000104      001      150      PVSEMT: .BYTE  1,150
63 000106      002      001      .BYTE  2,1
64 000110      000000G      .WORD  PFS0
65 000112      000000      .WORD  0
66
67      ; Emt to clear privilege flags
68
69 000114      001      150      PVCEMT: .BYTE  1,150
70 000116      001      001      .BYTE  1,1
71 000120      000000G      .WORD  PFC0
72 000122      000000      .WORD  0
73
74      ; Emt to start an inactive line
75
76 000124      000      126      STAEMT: .BYTE  0,126
77 000126      000020      .WORD  20
78 000130      000000      .WORD  0
79
80      ; Emt to declare terminal type change to process windowing system
81
82 000132      007      161      WINSTT: .BYTE  7,161

```

```

1
2 ; -----
3 ; Options for SET TT command
4 TBLDEF TT, 3
5 CMDDEF ADM*3A, SADM3A, 0, 0
6 CMDDEF AL*TERNATE, SETTTB, LSW2, $ALTER
7 CMDDEF AUTO*BAUD, SAUTO, 0, 0
8 CMDDEF NOAUTO*BAUD, CLRTTP, LSW2, $AUTO
9 CMDDEF BIT*S, STCLEN, 0, 0
10 CMDDEF DEAD, SETDED, LSW3, $DEAD
11 CMDDEF NODEAD, CLRTTP, LSW3, $DEAD
12 CMDDEF DEC*WRITER, STLA36, 0, 0
13 CMDDEF DEF*ER, SETTTB, LSW2, $DEFER
14 CMDDEF NOD*EFER, CLRTTB, LSW2, $DEFER
15 CMDDEF DI*ABLO, SDIAB, 0, 0
16 CMDDEF DT*R, SETDTR, 0, 0
17 CMDDEF NODT*R, CLRDTR, 0, 0
18 CMDDEF E*CHO, SETTTB, LSW2, $ECHO
19 CMDDEF NOE*CHO, CLRTTB, LSW2, $ECHO
20 CMDDEF EIGHT*BIT, SETTTB, LSW2, $8BIT
21 CMDDEF NOEIGHT*BIT, CLRTTB, LSW2, $8BIT
22 CMDDEF FORM, SETTTB, LSW2, $FORM
23 CMDDEF NOFORM, CLRTTB, LSW2, $FORM
24 CMDDEF FORMO, SETTTB, LSW4, $FORMO
25 CMDDEF NOFORMO, CLRTTB, LSW4, $FORMO
26 CMDDEF G*AG, SETTTB, LSW7, $TTGAG
27 CMDDEF NOG*AG, CLRTTB, LSW7, $TTGAG
28 CMDDEF HAZEL*TINE, STHAZL, 0, 0
29 CMDDEF HO*LD, RDCMD, 0, 0
30 CMDDEF NOH*OLD, RDCMD, 0, 0
31 CMDDEF LA1*20, SLA120, 0, 0
32 CMDDEF LA3*6, STLA36, 0, 0
33 CMDDEF LC, SETTTB, LSW2, $LC
34 CMDDEF NOL*C, CLRTTB, LSW2, $LC
35 CMDDEF LE*NGTH, RDCMD, 0, 0
36 CMDDEF PAG*E, SETTTB, LSW2, $PAGE
37 CMDDEF NOP*AGE, CLRTTB, LSW2, $PAGE
38 CMDDEF PAR*ITY, SETPAR, 0, 0
39 CMDDEF NOPAR*ITY, STNOPR, 0, 0
40 CMDDEF PHO*NE, SETTTP, LSW2, $PHONE
41 CMDDEF NOPHO*NE, CLRTTP, LSW2, $PHONE
42 CMDDEF QUI*ET, SETQUT, 0, 0
43 CMDDEF NOQ*UIET, RSTQUT, 0, 0
44 CMDDEF QUM*E, SQUME, 0, 0
45 CMDDEF SC*OPE, SETTTB, LSW2, $SCOPE
46 CMDDEF NOSC*OPE, SETNSC, 0, 0
47 CMDDEF SEVEN*BIT, CLRTTB, LSW2, $8BIT
48 CMDDEF SI*NGLE, SETTTB, LSW6, $STSNG
49 CMDDEF NOSI*NGLE, CLRTTB, LSW6, $STSNG
50 CMDDEF SPE*ED, STTTSP, 0, 0
51 CMDDEF STA*RT, STTTST, 0, 0
52 CMDDEF STAN*DARD, CLRTTB, LSW2, $ALTER
53 CMDDEF STD, CLRTTB, LSW2, $ALTER
54 CMDDEF SYSP*ASSWORD, SETTTP, LSW2, $SYSPS
55 CMDDEF NOSYSP*ASSWORD, CLRTTP, LSW2, $SYSPS
56 CMDDEF SYSPS, SETTTP, LSW2, $SYSPS
57 CMDDEF NOSYSPS, CLRTTP, LSW2, $SYSPS

```

```

58 000652      CMDDEF  SYSP*WD, SETTTP, LSW2, $SYSPS
59 000662      CMDDEF  NOSYSP*WD, CLRTTP, LSW2, $SYSPS
60 000672      CMDDEF  TAB, SETTTB, LSW2, $TAB
61 000702      CMDDEF  NOTAB, CLRTTB, LSW2, $TAB
62 000712      CMDDEF  TAP*E, SETTTB, LSW2, $TAPE
63 000722      CMDDEF  NOTAP*E, CLRTTB, LSW2, $TAPE
64 000732      CMDDEF  TR*ANSLATE, STTRNS, 0, 0
65 000742      CMDDEF  VT1*00, SVT100, 24. , 0
66 000752      CMDDEF  VT2*00, SVT200, 24. , 0
67 000762      CMDDEF  VT220, SVT200, 24. , 0
68 000772      CMDDEF  VT2207, SVT227, 24. , 0
69 001002      CMDDEF  VT2208, SVT228, 24. , 0
70 001012      CMDDEF  VT240, SVT200, 24. , 0
71 001022      CMDDEF  VT241, SVT200, 24. , 0
72 001032      CMDDEF  VT5*0, SVI50, 12. , 0
73 001042      CMDDEF  VT5*2, SVT50, 24. , 0
74 001052      CMDDEF  W*AIT, SETTW, 0, 0
75 001062      CMDDEF  NOW*AIT, SETTNW, 0, 0
76 001072      CMDDEF  XON, SETXON, 0, 0
77 001102      CMDDEF  8BIT, SETTTB, LSW2, $8BIT
78 001112      CMDDEF  NO8BIT, CLRTTB, LSW2, $8BIT
79 001122      CMDDEF  7BIT, CLRTTB, LSW2, $8BIT
80 001132      TBLEND
81
82      ; Define options for SET TI [n] PARITY= command
83      ;
84 000134      TBLDEF  PAR, 3
85 001136      CMDDEF  E*VEN, STPAR, LP$PAR
86 001146      CMDDEF  O*DD, STPAR, LP$PAR!LP$ODD
87 001156      CMDDEF  N*ONE, STPAR, 0
88 001166      TBLEND
89
90      ;
91      ; Define options for the SET CLn command
92      ;
93 000134      TBLDEF  CLOP, 2
94 001172      CMDDEF  BIT*S, STCLBT, 0
95 001200      CMDDEF  CR, STCLOP, CO$CR
96 001206      CMDDEF  NOCR, CLCLOP, CO$CR
97 001214      CMDDEF  DTR, STCLOP, CO$DTR
98 001222      CMDDEF  NODTR, CLCLOP, CO$DTR
99 001230      CMDDEF  ENDP*AGES, STCLVL, CL$EPN
100 001236     CMDDEF  ENDS*TRING, CLESTR, 0
101 001244     CMDDEF  FORM, STCLOP, CO$FF
102 001252     CMDDEF  NOFORM, CLCLOP, CO$FF
103 001260     CMDDEF  FF, STCLOP, CO$FF
104 001266     CMDDEF  NOFF, CLCLOP, CO$FF
105 001274     CMDDEF  FORMO, STCLOP, CO$FFO
106 001302     CMDDEF  NOFORMO, CLCLOP, CO$FFO
107 001310     CMDDEF  GR*APH, STCLGR, 0.
108 001316     CMDDEF  NOGR*APH, STCLGR, 132.
109 001324     CMDDEF  LC, STCLOP, CO$LC
110 001332     CMDDEF  NOLC, CLCLOP, CO$LC
111 001340     CMDDEF  TAB, STCLOP, CO$TAB
112 001346     CMDDEF  NOTAB, CLCLOP, CO$TAB
113 001354     CMDDEF  CTRL, STCLOP, CO$CTL
114 001362     CMDDEF  NOCTRL, CLCLOP, CO$CTL

```

```

115 001370      CMDDEF  LEN*GTH, STCLVL, CL$LEN
116 001376      CMDDEF  LFIN, STCLOP, CO$LFI
117 001404      CMDDEF  NOLFIN, CLCLOP, CO$LFI
118 001412      CMDDEF  LFOUT, STCLOP, CO$LFO
119 001420      CMDDEF  NOLFOUT, CLCLOP, CO$LFO
120 001426      CMDDEF  LIN*E, STCLLN, 0
121 001434      CMDDEF  BININ, STCLOP, CO$BNI
122 001442      CMDDEF  NOBININ, CLCLOP, CO$BNI
123 001450      CMDDEF  BINOUT, STCLOP, CO$BNO
124 001456      CMDDEF  NOBINOUT, CLCLOP, CO$BNO
125 001464      CMDDEF  BBIT, STCLOP, CO$BBT
126 001472      CMDDEF  EIGHT*BIT, STCLOP, CO$BBT
127 001500      CMDDEF  NO8*BIT, CLCLOP, CO$BBT
128 001506      CMDDEF  NOEIGHT*BIT, CLCLOP, CO$BBT
129 001514      CMDDEF  7BIT, CLCLOP, CO$BBT
130 001522      CMDDEF  PAR*ITY, STCLPO, CLPRHD
131 001530      CMDDEF  NOPAR*ITY, STCLNP
132 001536      CMDDEF  SEVEN*BIT, CLCLOP, CO$BBT
133 001544      CMDDEF  TOP, CLTOP, 0
134 001552      CMDDEF  TR*ANSLATE, CLTRNS, 0
135 001560      CMDDEF  RES*ET, STCLRS, 0
136 001566      CMDDEF  SKI*P, STCLVL, CL$SKP
137 001574      CMDDEF  WID*TH, STCLVL, CL$WID
138 001602      CMDDEF  SP*EED, STCLSP, 0
139 001610      CMDDEF  VER*SION, STCLVR, 0
140 001616      CMDDEF  XON, STCLXN, 0
141              ; These two forms do not require allocation of the CL device, they
142              ; do their own separate checking for BYPASS privilege instead.
143              ; Leave them at the end of the table.
144 001624      CLNCKA:
145 001624      CMDDEF  XONB*YPASS, STCLXB, 0
146 001632      CMDDEF  RESETB*YPASS, STCLRB, 0
147 001640      TBLEND
148              ;
149              ; Define options for SET CL PARITY= command
150              ;
151 000134      TBLDEF  CLPR, 2
152 001644      CMDDEF  E*VEN, STCLPR, LP$PAR
153 001652      CMDDEF  O*DD, STCLPR, LP$PAR!LP$ODD
154 001660      CMDDEF  N*ONE, STCLPR, 0
155 001666      TBLEND
156              ;
157              ; Define options for the SET HOST command
158              ;
159 000134      TBLDEF  HOST, 1
160 001672      CMDDEF  DTE, HSTDTE
161 001676      CMDDEF  CL, HSTDTE
162 001702      CMDDEF  PO*RT, HSTPRT
163 001706      TBLEND
164              ;
165              ; Define options for SET PROCESS command
166              ;
167 000134      TBLDEF  CPOP, 2
168 001712      CMDDEF  PRIV*ILEGES, PRVOPT, 0
169 001720      CMDDEF  ID*ENTIFICATION, CPPPID, 0
170 001726      CMDDEF  NAM*E, CPPNAM, 0
171 001734      CMDDEF  PRIO*RITY, CPPPRI, 0

```

172 001742
173 001750
174 001756
175 001764

CMDDEF SUSP*END, CPPFLG, CPFSUS, 0
CMDDEF RES*UME, CPPFLG, CPFRES, 0
CMDDEF AU*THORIZED, CPPFLG, CPFAUT, 0
TBLEND

SET commands

```

1          .SBTTL SET commands
2          .SBTTL Process
3          ;-----
4          ; Process the SET PROCESS command.
5          ;
6 000134 SETPRC:
7          ;
8          ; Initialize cells which will hold values parsed by the command
9          ;
10 000134 116767 000000G 177661      MOVB   CORUSR, CPPID      ;Set process ID=our job
11 000142 105067 177657             CLRB   CPNAME          ;No name specified
12 000146 105067 177652             CLRB   CPPRIO         ;No priority
13 000152 005067 177642             CLR    CPFLAG         ;No flags
14 000156 004767 000000G             CALL   CLRPRV         ;No privileges
15          ;
16          ; Do command parsing
17          ;
18 000162 012704 001710'            MOV    #CPOPHD, R4      ;Point to option driver list
19 000166 004767 000000G            CALL   OPTLST         ;Parse the command
20          ;
21          ; Now process each option that was accrued
22          ;
23 000172 004767 000366             CALL   CPSNAM         ;See if process name was specified
24 000176 004767 000644             CALL   CPSPRI        ;See if priority was specified
25 000202 004767 000464             CALL   CPSPRV        ;See if privileges were specified
26 000206 004767 000676             CALL   CPSFLG        ;See if flags were specified
27          ;
28          ; Finished command
29          ;
30 000212 000167 000000G 9$:      JMP    RDCMD          ;Finished command

```

Process

```

1          ; -----
2          ;   SET PROCESS/IDENTIFICATION=job-number
3          ;
4 000216  010146  CPPPID: MOV      R1, -(SP)
5          ;
6          ;   Accrue the ID value
7          ;
8 000220  004767  000000G      CALL      ACRDEC
9          ;
10         ;   Make sure the ID value is valid
11        ;
12 000224  006301          ASL      R1          ; Convert to word table index
13 000226  001413          BEQ      10$          ; Invalid if zero
14 000230  020127  000000G  CMP      R1, #LSTSL      ; Is this a valid job #?
15 000234  101010          BHI      10$          ; Br if not
16 000236  032761  000000G  000000G  BIT      #$DILUP, LSW(R1) ; Is line logged on?
17 000244  001410          BEQ      11$          ; Br if not
18 000246  110167  177551      MOVVB   R1, CPPID      ; Save ID value
19        ;
20        ;   Finished
21        ;
22 000252  012601          MOV      (SP)+, R1
23 000254  000207          RETURN
24        ;
25        ;   Invalid ID
26        ;
27 000256          10$:   FABORT  #EM$ILN      ; Invalid job numver
28 000266          11$:   FABORT  #NOTON      ; Line not logged on

```


Process

```

1
2 ; -----
3 ; SET PROCESS/PRIORITY=value
4 000276 010146
5 000300 010546
6
7 ;
8 ; Accrue the priority value
9 000302 004767 000000G ; CALL ACRDEC ; Accrue priority value
10
11 ; See if the priority value is valid
12
13 000306 005701 ; TST R1 ; Must be > 0
14 000310 003406 ; BLE 10# ; Br if invalid
15 000312 120167 000000G ; CMPB R1,MXJPRI ; Is it within valid range for job?
16 000316 101003 ; BHI 10# ; Br if not
17 000320 110167 177500 ; MOVB R1,CPPRI0 ; Save priority value
18 000324 000424 ; BR 9#
19
20 ; Invalid priority value
21
22 000326 ; 10#: FERR #BADPRI ; Invalid priority value
23 000342 012705 000001 ; MOV #1,R5 ; Minimum prio value
24 000346 004767 000000G ; CALL PRTDEC
25 000352 ; .PRINT #TOTXT ; "to"
26 000360 116705 000000G ; MOVB MXJPRI,R5 ; Max possible
27 000364 004767 000000G ; CALL PRTDEC
28 000370 ; .PRINT #CRLF
29
30 ; Finished
31
32 000376 012605 ; 9#: MOV (SP)+,R5
33 000400 012601 ; MOV (SP)+,R1
34 000402 000207 ; RETURN

```

Process

```

1          ; -----
2          ;   SET PROCESS/NAME="string"
3          ;
4 000404 010246 CPPNAM: MOV     R2,-(SP)
5 000406 010446          MOV     R4,-(SP)
6          ;
7          ;   Make sure equal sign follows keyword
8          ;
9 000410 004767 000000G          CALL    CHKEQ          ;Check for equal sign
10 000414 012702 000025'          MOV     #CPNAME,R2      ;Point to buffer for name
11          ;
12          ;   See if string is enclosed in quote marks
13          ;
14 000420 004767 000000G          CALL    SKPSPC          ;Skip over any spaces
15 000424 111300          MOVVB   (R3),R0          ;Get 1st char of string
16 000426 120027 000042          CMPB   R0,#42          ;Quote mark?
17 000432 001403          BEQ    3$              ;Br if yes
18 000434 120027 000047          CMPB   R0,#47          ;Apostrophe?
19 000440 001014          BNE    1$              ;Br if not
20          ;
21          ;   String is enclosed in quote marks
22          ;
23 000442 004767 000000G 3$:   CALL    ACRSTR          ;Accrue a quoted string
24 000446 005700          TST    R0              ;Is string empty?
25 000450 001423          BEQ    2$              ;If empty, go blank fill it.
26 000452 020027 000014          CMP    R0,#12.         ;Is string too long?
27 000456 101026          BHI    10$             ;Br if too long
28 000460 012704 000000G          MOV    #BLKO,R4        ;Point to accrued string
29 000464 112422 4$:   MOVVB   (R4)+,(R2)+    ;Move to string buffer
30 000466 077002          SOB   R0,4$           ;
31 000470 000413          BR    2$              ;Go blank fill rest of buffer if needed
32          ;
33          ;   Name string is not quoted.
34          ;   Move name to buffer till we hit end of string
35          ;
36 000472 111300 1$:   MOVVB   (R3),R0          ;Get next char from string
37 000474 001411          BEQ    2$              ;Br if hit end of command
38 000476 120027 000057          CMPB   R0,#'/'        ;Start of next qualifier?
39 000502 001406          BEQ    2$              ;Br if yes
40 000504 005203          INC    R3              ;Point to next character in command
41 000506 020227 000041'          CMP    R2,#CPNAME+12. ;Buffer overflow?
42 000512 103010          BHIS  10$             ;Br if yes
43 000514 110022          MOVVB   R0,(R2)+      ;Store char into name buffer
44 000516 000765          BR    1$              ;
45          ;
46          ;   Fill remainder of buffer with blanks
47          ;
48 000520 020227 000041' 2$:   CMP    R2,#CPNAME+12. ;Filled buffer yet?
49 000524 103011          BHIS  9$              ;Br if yes
50 000526 112722 000040          MOVVB   #'',(R2)+    ;Pad with blanks
51 000532 000772          BR    2$              ;
52          ;
53          ;   Error: string is too long
54          ;
55 000534 10$:   FERR    #EM$STL          ;String is too long
56          ;
57          ;   Finished

```

```
58 ;  
59 000550 012604 9#: MOV (SP)+,R4  
60 000552 012602 MOV (SP)+,R2  
61 000554 000207 RETURN  
62 ;-----  
63 ; Set some flag for a SET PROCESS qualifier  
64 ;  
65 000556 051467 177236 CPPFLG: BIS (R4),CPFLAG ;Set specified flag bit  
66 000562 000207 RETURN
```

Process

```

1          ; -----
2          ; See if NAME qualifier was specified.
3          ;
4 000564   010146   CPSNAM: MOV     R1, -(SP)
5 000566   010246           MOV     R2, -(SP)
6          ;
7          ; See if name qualifier was specified
8          ;
9 000570   105767   177231           TSTB   CPNAME           ;Was name qualifier specified?
10 000574   001423           BEQ    9$              ;Br if not
11         ;
12         ; Make sure we are changing for our own process
13         ;
14 000576   116701   177221           MOVB   CPPID, R1       ;Get job index
15 000602   120167   000000G          CMPB   R1, CORUSR     ;Affecting our own process?
16 000606   001021           BNE    10$           ;Br if not
17 000610   032767   000000G 000000G          BIT    #PO$NAM, PRIVCO ;Are we authorized to change our name?
18 000616   001421           BEQ    11$           ;Br if not
19 000620   070127   000006           MUL   #6, R1         ;Each job has 12 character name
20 000624   062701   000000G          ADD   #LUNAME, R1    ;Point to name cell for our job
21 000630   012702   000025'          MOV   #CPNAME, R2    ;Point to name buffer
22 000634   012700   000014           MOV   #12, R0        ;Get # bytes to move
23 000640   112221           1$:   MOVB   (R2)+, (R1)+ ;Set name for job
24 000642   077002           SOB   R0, 1$
25         ;
26         ; Finished
27         ;
28 000644   012602           9$:   MOV   (SP)+, R2
29 000646   012601           MOV   (SP)+, R1
30 000650   000207           RETURN
31         ;
32         ; Cannot change name of another job
33         ;
34 000652           10$:  FABORT #EM$CND
35 000662           11$:  FABORT #EM$NPR

```

Process

```

1
2 ; -----
3 ; Change privilege flags for our process.
4 000672 010246 CPSPRV: MOV R2, -(SP)
5 000674 010346 MOV R3, -(SP)
6 000676 010446 MOV R4, -(SP)
7
8 ; See if any privilege changes were specified
9
10 000700 012702 000000G MOV #PFSO, R2 ;Flags to set
11 000704 012703 000000G MOV #PFCO, R3 ;Flags to clear
12 000710 012700 000000G MOV #PVNPW, R0 ;# words to check
13 000714 005722 1$: TST (R2)+ ;Any flags to set?
14 000716 001004 BNE 2$ ;Br if yes
15 000720 005723 TST (R3)+ ;Any flags to clear?
16 000722 001002 BNE 2$ ;Br if yes
17 000724 077005 SOB R0, 1$ ;Keep checking
18 000726 000433 BR 9$ ;No privilege changes requested
19
20 ; Make sure we are changing privileges for our job
21
22 000730 126767 177067 000000G 2$: CMPB CPPID, CORUSR ;Changing privilege for our job?
23 000736 001033 BNE 10$ ;Br if not
24
25 ; Set correct EMT function code depending on whether we are doing a
26 ; normal privilege change or are changing authorized privileges.
27
28 000740 012704 000001 MOV #1, R4 ;Assume normal priv change
29 000744 032767 000004 177046 BIT #CPFAUT, CPFLAG ;Changing authorized privileges?
30 000752 001402 BEQ 4$ ;Br if not
31 000754 012704 000002 MOV #2, R4 ;Code to change authorized privileges
32 000760 110467 177123 4$: MOVB R4, PVSEMT+3 ;Set code in EMT arg block
33 000764 110467 177127 MOVB R4, PVCEMT+3
34
35 ; Set any specified privileges
36
37 000770 005004 CLR R4 ;No privilege error detected
38 000772 012700 000104' MOV #PVSEMT, R0 ;Point to EMT arg block
39 000776 104375 EMT 375 ;Set privilege flags
40 001000 103001 BCC 3$ ;Br if no authorization violation
41 001002 005204 INC R4 ;Remember we had authorization violation
42
43 ; Clear any specified privileges
44
45 001004 012700 000114' 3$: MOV #PVCEMT, R0 ;Point to EMT arg block
46 001010 104375 EMT 375 ;Clear privilege flags
47 001012 005704 TST R4 ;Did we have an authorization violation?
48 001014 001010 BNE 11$ ;Br if yes
49
50 ; Finished
51
52 001016 012604 9$: MOV (SP)+, R4
53 001020 012603 MOV (SP)+, R3
54 001022 012602 MOV (SP)+, R2
55 001024 000207 RETURN
56
57 ; Can't change privilege for another job

```

58
59 001026
60
61
62
63 001036

;
10\$: FABORT #EM\$CPO ;Can't change privilege for another job
;
; Not authorized to give yourself this privilege
;
11\$: FABORT #EM\$CAP ;Can't authorize that privilege

Process

```

1
2 ; -----
3 ; See if we need to change priority for job.
4 001046 010246 CPSPRI: MOV R2, -(SP)
5 ;
6 ; See if a priority change was requested
7 ;
8 001050 105767 176750 TSTB CPPRID ;Was a priority specified
9 001054 001413 BEQ 9$ ;Br if not
10 001056 116702 176741 MOVB CPPID, R2 ;Get ID # of job we want to affect
11 001062 004767 000000G CALL CKACQJ ;Can we access that job?
12 001066 103406 BCS 9$ ;Br if not
13 ;
14 ; Change priority value
15 ;
16 001070 116762 176730 000000G MOVB CPPRID, LBSPRI(R2) ;Set base priority
17 001076 116762 176722 000000G MOVB CPPRID, LPRI(R2) ;Set running priority
18 ;
19 ; Finished
20 ;
21 001104 012602 9$: MOV (SP)+, R2
22 001106 000207 RETURN

```

Process

```

1          ; -----
2          ; See if we need to suspend or resume a job.
3          ;
4 001110 010246 CPSFLG: MOV      R2,-(SP)
5          ;
6          ; See if any flags were specified
7          ;
8 001112 032767 000003 176700          BIT      #<CPFSUS!CPFRES>,CPFLAG ;Suspend or resume requested?
9 001120 001423          BEQ      9$          ;Br if not
10         ;
11         ; Make sure we aren't suspending or resuming our own job
12         ;
13 001122 116702 176675          MOVB     CPPID,R2          ;Get ID # of job we want to affect
14 001126 120267 000000G        CMPB     R2,CORUSR        ;Is it our job?
15 001132 001420          BEQ      10$          ;Br if yes
16         ;
17         ; Make sure we can access the specified job
18         ;
19 001134 004767 000000G        CALL     CKACDJ          ;Can we access that job?
20 001140 103413          BCS      9$          ;Br if not
21         ;
22         ; Suspend or resume the specified job
23         ;
24 001142 012700 000000G        MOV      #RJEMT,R0        ;Point to resume-job arg block
25 001146 032767 000002 176644    BIT      #CPFRES,CPFLAG ;Does he want to resume the job?
26 001154 001002          BNE      1$          ;Br if yes
27 001156 012700 000000G        MOV      #SJEMT,R0        ;Point to suspend-job arg block
28 001162 010260 000004 1$:     MOV      R2,4(R0)        ;Set job # in EMT arg block
29 001166 104375          EMT      375          ;Suspend or resume the job
30         ;
31         ; Finished
32         ;
33 001170 012602 9$:     MOV      (SP)+,R2
34 001172 000207          RETURN
35         ;
36         ; Error -- Trying to suspend or resume our own job
37         ;
38 001174 10$:     FABORT  #EM$IDR          ;/ID parameter required

```


Terminal

```

1
2
3
4
5 001204
6
7
8
9 001204 105067 176612
10 001210 004767 000000G
11 001214 010305
12 001216 005002
13 001220 112500
14 001222 001413
15 001224 120027 000040
16 001230 001410
17 001232 120027 000060
18 001236 103435
19 001240 120027 000071
20 001244 101032
21 001246 005202
22 001250 000763
23 001252 005702
24 001254 001426
25
26
27
28 001256 004767 000000G
29
30
31
32 001262 006301
33 001264 001403
34 001266 020127 000000G
35 001272 101404
36 001274
37
38
39
40 001304 004767 000000G
41 001310 110167 176506
42
43
44
45
46
47 001314 032761 000000G 000000G
48 001322 001003
49 001324 016161 000000G 000000G
50
51
52
53
54
55 001332 012704 000000'
56 001336 004767 000000G
57

```

```

.SBTTL . Terminal
-----
; Process the SET TERMINAL command.
;
SETTTY:
;
; See if a line number was specified
;
CLR B SETPRM ; Assume no line number specified
CALL SKPSPC ; Skip over any spaces
MOV R3,R5 ; Save pointer to first operand
CLR R2 ; Say no digits seen yet
1$: MOV B (R5)+,R0 ; Get next char
BEQ 2$ ; Br if hit end of command line
CMP B R0,#40 ; Is this a space?
BEQ 2$ ; Br if yes
CMP B R0,#'0 ; Is this a digit?
BLO 3$ ; Br if not
CMP B R0,#'9 ; Br if not digit
BHI 3$ ; Br if not digit
INC R2 ; Remember we saw a digit
BR 1$ ; Scan first item and see if it is a number
2$: TST R2 ; Was first item a number?
BEQ 3$ ; Br if not
;
; There is a terminal number specified, accrue it
;
CALL ACRDEC ; Accrue the terminal number
;
; See if this is a valid terminal number
;
ASL R1 ; Convert line number to line index number
BEQ 4$ ; Zero is not valid line number
CMP R1,#LSTPL ; Is this a primary line number?
BLOS 5$ ; Br if yes
4$: FABORT #EM$ILN ; Invalid line number
;
; Require TERMINAL privilege to make any permanent change
;
5$: CALL CKTERM ; Require TERMINAL privilege to do this
MOV B R1,SETPRM ; Remember to make set permanent
;
; If we are changing line parameters for a line that is not
; currently logged on, initialize the LSW2 table from the initial
; flags for the line (ILSW2).
;
BIT #DILUP,LSW(R1) ; Is line logged on?
BNE 3$ ; Br if yes
MOV ILSW2(R1),LSW2(R1); Initialize LSW2 table
;
; We have the terminal line index number for which the command is being
; issued in R1.
; Process all qualifier specified with command.
;
3$: MOV #TTHD,R4 ; Get pointer to option table
CALL SCNOPS ; Process all of the command options
;

```

```
58 ; End of command reached
59 ;
60 001342 000167 000000G JMP RDCMD ;Finished SET TT command
```

Terminal

```

1          ; -----
2          ;   SET TERMINAL n START
3          ;
4 001346 004767 000000G STTTST: CALL   CKPRIV           ;Require OPER privilege
5          ;
6          ;   See if line is already started
7          ;
8 001352 032761 000000G 000000G          BIT    #$DILUP,LSW(R1) ;Is line already started?
9 001360 001404          BEQ     1$           ;Br if not
10 001362          FABORT  #EM$LAS          ;Line is already started
11         ;
12         ;   See if line is marked dead
13         ;
14 001372 032761 000000G 000000G 1$:    BIT    #$DEAD,LSW3(R1) ;Is line marked as dead?
15 001400 001404          BEQ     2$           ;Br if not
16 001402          FABORT  #EM$LFD          ;Line is dead
17         ;
18         ;   See if a CL unit is attached to the line
19         ;
20 001412 005761 000000G          2$:    TST    LCLUNT(R1)      ;Is line being used by a CL unit?
21 001416 002404          BLT     3$           ;Br if not
22 001420          FABORT  #EM$CLU          ;Line in use by a CL unit
23         ;
24         ;   Start the line
25         ;
26 001430 010167 176474          3$:    MOV    R1,STAEMT+4      ;Store line index into EMT arg block
27 001434 012700 000124'          MOV    #STAEMT,R0      ;Point to EMT arg block
28 001440 104375          EMT    375          ;Start the line
29 001442 000207          9$:    RETURN

```

Terminal

```

1          ; -----
2          ; SET TERMINAL [n] SPEED=n
3          ;
4 001444 004767 000000G STTTSP: CALL CKTERM          ;Require TERMINAL privilege
5 001450 004767 000000G CALL ACRSPD          ;Accrue speed value
6 001454 042761 000000G 000000G BIC ##AUTO,LSW2(R1) ;Say no autobaud on this line
7 001462 116100 000001G MOVB LMXPRM+1(R1),R0 ;Get control flags for line
8 001466 042700 000000G BIC #<LP$SPD>,R0 ;Clear speed flags
9 001472 050005 BIS R0,R5 ;Combine parity and length flags with speed
10 001474 010567 176372 STSP1: MOV R5,LSPEMT+4 ;Store speed, par, len code into EMT arg block
11 001500 010167 176364 MOV R1,LSPEMT+2 ;Store line # into EMT arg block
12 001504 006267 176360 ASR LSPEMT+2 ;Convert index # to line #
13 001510 012700 000066' MOV #LSPEMT,R0 ;Point to EMT arg block
14 001514 104375 EMT 375 ;Set the speed
15 001516 103402 BCS 2$ ;Br if error
16 001520 000167 000474 JMP BSO1
17          ;
18          ; Error on EMT to set speed
19          ;
20 001524 2$: FABORT #EM$ILN ;Say invalid line number

```

Terminal

```

1
2 ; -----
3 ; SET TT [n] XON
4 001534 004767 0000006 SETXON: CALL CKTERM ;Require TERMINAL privilege
5 001540 010167 176332 MOV R1,XONEMT+2 ;Set line # in EMT argument block
6 001544 006267 176326 ASR XONEMT+2 ;Convert index # to line #
7 001550 012700 000074' MOV #XONEMT,R0 ;Point to EMT argument block
8 001554 104375 EMT 375 ;Do the XON EMT
9 001556 103402 BCS 2$ ;Br if invalid line #
10 001560 000167 000434 JMP BSO1
11 001564 2$: FABORT #EM$ILN ;Say invalid line number

```

Terminal

```

1          ; -----
2          ; SET TT [n] [NO]DTR
3          ;
4 001574 112767 000002 176276 SETDTR: MOVB #2,DTREMT ;Set function to raise DTR
5 001602 000403          BR DTRCOM ;Br to common routine
6 001604 112767 000003 176266 CLRDR: MOVB #3,DTREMT ;Set function to lower DTR
7 001612 004767 000000G DTRCOM: CALL CKTERM ;Require TERMINAL privilege
8 001616 006201          ASR R1 ;Convert line index to line number
9 001620 010167 176256          MOV R1,DTREMT+2 ;Set in EMT arg block
10 001624 006301          ASL R1 ;Get back as line index
11 001626 012700 000100'          MOV #DTREMT,R0 ;Point to EMT arg block to
12 001632 104375          EMT 375 ;Raise or lower DTR
13 001634 103402          BCS 2$ ;Br on EMT error
14 001636 000167 000356          JMP BSD1
15 001642          2$: FABORT EM$ILN ;Say invalid line number

```

Terminal

```

1
2 ; -----
3 ; SET TERMINAL AUTOBAUD
4 001652 004767 000000G SAUTO: CALL CKTERM ;Require TERMINAL privilege
5 001656 052761 000000G 000000G BIS #$AUTO,LSW2(R1) ;Set flag saying to do autobaud select
6 001664 032761 000000G 000000G BIT #$DILUP,LSW(R1) ;Is anyone using the line now?
7 001672 001003 BNE 9$ ;Br if yes
8 001674 012705 000000G MOV #S9600,R5 ;Set line speed to 9600 baud
9 001700 000675 BR STSP1
10 001702 000167 000312 9$: JMP BSO1

```

Terminal

```

1
2 ; -----
3 ; SET TERMINAL PARITY={EVEN ODD NONE}
4 001706 010446 SETPAR: MOV R4, -(SP) ; Save current option table pointer
5 001710 004767 000000G CALL CKTERM ; Require TERMINAL privilege
6 001714 004767 000000G CALL SKPSPC ; Skip past any spaces
7 001720 121327 000075 CMPB (R3), #'= ; Equal sign following PARITY?
8 001724 001001 BNE 1$ ; Br if not
9 001726 005203 INC R3 ; Skip past equal sign
10 001730 012704 001134' 1$: MOV #PARHD, R4 ; Point to new option table
11 001734 004767 000000G CALL SETWRD ; Process EVEN, ODD, NONE word
12 001740 012604 MOV (SP)+, R4
13 001742 000207 RETURN

```


Terminal

```

1
2 ; -----
3 ; SET TERMINAL BITS=n
4 001744 004767 000000G STCLEN: CALL CKTERM ;Require TERMINAL privilege
5 001750 116105 000001G MOV LMXPRM+1(R1),R5 ;Get current line parameters
6 001754 010102 MOV R1,R2 ;Save line index number
7 001756 004767 000000G CALL ACRDEC ;Accrue parameter value
8 001762 042705 000000G BIC #LP$7BT,R5 ;Assume 8 bits wanted
9 001766 020127 000007 CMP R1,#7. ;Is value 7?
10 001772 001003 BNE 1$ ;Br if not
11 001774 052705 000000G BIS #LP$7BT,R5 ;Select 7 bits
12 002000 000407 BR 2$
13 002002 120127 000010 1$: CMPB R1,#8. ;Is value 8?
14 002006 001404 BEQ 2$ ;Br if yes
15 002010 FABORT #EM$ICL ;Invalid character length
16 002020 010201 2$: MOV R2,R1 ;Get back line index number
17 002022 000167 177446 JMP STSP1 ;Go set new length

```

Terminal

```

1
2 ; -----
3 ; SET TERMINAL PARITY={EVEN ODD NONE}
4 ;
5 ; Inputs:
6 ; R1 = line index number.
7 ; R4 = Pointer to work with parity flag bits.
8 ;
9 STPAR: CALL CKTERM ;Require TERMINAL privilege
10 MOVB LMXPRM+1(R1),R5 ;Get current flags for line
11 BIC #LP$PAR!LP$ODD,R5;Clear parity control flags
12 BISB @R4,R5 ;Set new flags
13 JMP STSP1 ;Go set value
14 ;
15 ; SET TERMINAL NOPARITY
16 ;
17 STNOPR: CALL CKTERM ;Require TERMINAL privilege
18 MOVB LMXPRM+1(R1),R5 ;Get current flags for line
19 BIC #LP$PAR!LP$ODD,R5;Clear parity control flags
20 JMP STSP1 ;Go set value
21 ; -----
22 ; SET TERMINAL [n] TRANSLATE=(ext=int,ext=int,...)
23 ;
24 STTRNS: MOV R1,-(SP) ;Save original line number
25 MOV LNPRIM(R1),R1 ;Make sure we have primary line number
26 CALL PRSTRN ;Parse the translate qualifier
27 MOV (SP)+,R1 ;Restore original line number
28 RETURN

```

Terminal

```

1          ; -----
2          ;   Set parameters for particular terminal types.
3          ;
4          ;   SET PARAMETERS FOR A VT50/VT52.
5          ;   SET VT50
6 002106  042761  000000G 000000G SVT50:  BIC      #VT52ND,LSW2(R1);CLEAR SOME FLAGS
7 002114  052761  000000G 000000G      BIS      #VT52FL,LSW2(R1);SAY TERMINAL IS A VT50
8 002122  012761  000000G 000000G      MOV      #VT52,LTRMTP(R1);SET TERMINAL TYPE
9 002130  000414                BR        BS02
10         ;
11        ;   SET PARAMETERS FOR LA36.
12        ;
13 002132  042761  000000G 000000G STLA36: BIC      #LA36ND,LSW2(R1);RESET VARIOUS FLAGS
14 002140  052761  000000G 000000G      BIS      #LA36FL,LSW2(R1)
15 002146  012761  000000G 000000G      MOV      #LA36,LTRMTP(R1);SET TERMINAL TYPE
16 002154  042761  000000G 000000G      BIC      #$SLON,LSW7(R1) ;Disalble SL editor for this terminal type
17        ;
18        ;   Set up terminal type information
19        ;
20 002162  120167  000000G                BS02:  CMPB     R1,CORUSR      ;Changing terminal type for our job?
21 002166  001006                BNE     1$              ;Br if not our job
22 002170  005761  000000G                TST     LWINDO(R1)      ;Is process windowing turned on for job?
23 002174  001403                BEQ     1$              ;Br if not
24 002176  012700  000132'                MOV     #WINSTT,R0     ;Point to EMT argument block
25 002202  104375                EMT     375            ;Tell windowing system about terminal type
26 002204  105767  175612                1$:   TSTB    SETPRM      ;Should we make the set permanent?
27 002210  001403                BEQ     BS01          ;Br if not
28 002212  016161  000000G 000000G      MOV     LTRMTP(R1),ITRMTP(R1) ;Set initial terminal type
29 002220  016161  000000G 000000G BS01:  MOV     LSW2(R1),LSW2S(R1) ;Save in case of ctrl-c reentry
30 002226  105767  175570                TSTB    SETPRM      ;Should we make the set permanent?
31 002232  001403                BEQ     SETJMP       ;Br if not
32 002234  016161  000000G 000000G      MOV     LSW2(R1),ILSW2(R1);Set initial flags for line
33 002242  000207                SETJMP: RETURN      ;Return to process next qualifier
34        ;
35        ;   LA120 parameters
36        ;
37 002244  042761  000000G 000000G SLA120: BIC      #LA12ND,LSW2(R1);SET FLAGS
38 002252  052761  000000G 000000G      BIS      #LA12FL,LSW2(R1)
39 002260  012761  000000G 000000G      MOV     #LA120,LTRMTP(R1);SET TERMINAL TYPE
40 002266  042761  000000G 000000G      BIC      #$SLON,LSW7(R1) ;Disalble SL editor for this terminal type
41 002274  000732                BR        BS02
42        ;
43        ;   VT100 parameters
44        ;
45 002276  042761  000000G 000000G SVT100: BIC      #VT10ND,LSW2(R1);SET FLAGS
46 002304  052761  000000G 000000G      BIS      #VT10FL,LSW2(R1)
47 002312  012761  000000G 000000G      MOV     #VT100,LTRMTP(R1);SET TERMINAL TYPE
48 002320  000720                STERM:  BR        BS02
49        ;
50        ;   VT200 parameters
51        ;
52 002322  042761  000000G 000000G SVT200: BIC      #VT20ND,LSW2(R1)
53 002330  052761  000000G 000000G      BIS      #VT20FL,LSW2(R1)
54 002336  012761  000000G 000000G      MOV     #VT200,LTRMTP(R1)
55 002344  000706                BR        BS02
56        ;
57        ;   VT200 parameters 7 bit controls

```

Terminal

```

58
59 002346 042761 000000G 000000G SVT227: BIC      #VT20ND,LSW2(R1)
60 002354 052761 000000G 000000G      BIS      #VT20FL,LSW2(R1)
61 002362 012761 000000G 000000G      MOV      #VT2007,LTRMTP(R1)
62 002370 000674                BR        BS02
63
64          ; VT200 parameters 8 bit controls
65
66 002372 042761 000000G 000000G SVT228: BIC      #VT20ND,LSW2(R1)
67 002400 052761 000000G 000000G      BIS      #VT20FL,LSW2(R1)
68 002406 012761 000000G 000000G      MOV      #VT2008,LTRMTP(R1)
69 002414 000662                BR        BS02
70
71          ; Hazeltine parameters
72
73 002416 042761 000000G 000000G STHAZL: BIC      #HAZLND,LSW2(R1);RESET SOME FLAGS
74 002424 052761 000000G 000000G      BIS      #HAZLFL,LSW2(R1);SET SOME FLAGS
75 002432 012761 000000G 000000G      MOV      #HAZEL,LTRMTP(R1);SET TERMINAL TYPE
76 002440 042761 000000G 000000G      BIC      #$SLON,LSW7(R1) ;Disalble SL editor for this terminal type
77 002446 000645                BR        BS02
78
79          ; ADM3A parameters
80
81 002450 042761 000000G 000000G SADM3A: BIC      #ADM3ND,LSW2(R1);SET FLAGS
82 002456 052761 000000G 000000G      BIS      #ADM3FL,LSW2(R1)
83 002464 012761 000000G 000000G      MOV      #ADM3A,LTRMTP(R1);SET TERMINAL TYPE
84 002472 042761 000000G 000000G      BIC      #$SLON,LSW7(R1) ;Disalble SL editor for this terminal type
85 002500 000630                BR        BS02
86
87          ; SET PARAMETERS FOR DIABLO-1620 TERMINAL
88
89          ; SQUME:
90 002502 042761 000000G 000000G SDIAB:  BIC      #DIABND,LSW2(R1);RESET VARIOUS FLAGS
91 002510 052761 000000G 000000G      BIS      #DIABFL,LSW2(R1);SAY THIS IS A DIABLO
92 002516 012761 000000G 000000G      MOV      #DIABLO,LTRMTP(R1);SET TERMINAL TYPE
93 002524 042761 000000G 000000G      BIC      #$SLON,LSW7(R1) ;Disalble SL editor for this terminal type
94 002532 000613                BR        BS02
95
96          ; SET TT QUIET
97          ; (Also, SET NOVERIFY)
98
99 002534 116701 000000G                STNOVR: MOV      CORUSR,R1                ;Get current job index number
100 002540 004767 000004                CALL      SETQUT                ;Do the set
101 002544 000167 000000G                JMP        RDCMD                ;Finished
102
103 002550 052761 000000G 000000G SETQUT: BIS      #$QTSET,LSW2(R1)        ;REMEMBER SET WAS DONE
104 002556 052761 000000G 000000G      BIS      #$QUIET,LSW4(R1)        ;HAVE IMMEDIATE EFFECT
105 002564 120167 000000G                CMPB     R1,CORUSR                ;Are we doing set for current job?
106 002570 001025                BNE      1$                      ;Br if not
107 002572                .GVAL     #XAREA,#<CFSTS-MONVEC> ;GET CURRENT COMMAND FILE FLAGS
108 002612 010002                MOV      R0,R2
109 002614 052702 000000G                BIS      #CF$QUT,R2                ;SET TT-QUIET FLAG
110 002620                .PVAL     #XAREA,#<CFSTS-MONVEC>; R2; STORE UPDATED FLAGS
111 002644 000167 177350                1$:    JMP        BS01
112
113          ; SET TT NOQUIET
114          ; (Also, SET VERIFY)

```

```

115 ;
116 002650 116701 000000G STVRFY: MOV B CORUSR, R1 ;Get current job index number
117 002654 004767 000004 CALL RSTQUT ;Do the SET
118 002660 000167 000000G JMP RDCMD ;Finished
119 ;
120 002664 042761 000000G 000000G RSTQUT: BIC ##QTSET, LSW2(R1) ;REMEMBER SET WAS DONE
121 002672 042761 000000G 000000G BIC ##QUIET, LSW4(R1) ;START LISTING NOW
122 002700 120167 000000G CMPB R1, CORUSR ;Are we doing set for current line?
123 002704 001025 BNE 1$ ;Br if not
124 002706 .GVAL #XAREA, #<<CFSTS-MONVEC> ;GET CURRENT COMMAND FILE FLAGS
125 002726 010002 MOV R0, R2
126 002730 042702 000000G BIC #CF$QUT, R2 ;CLEAR TT-QUIET FLAG
127 002734 .PVAL #XAREA, #<<CFSTS-MONVEC>, R2; STORE UPDATED FLAGS
128 002760 000167 177234 1$: JMP BS01
129 ;
130 ; SET TT NOSCOPE
131 ;
132 002764 042761 000000G 000000G SETNSC: BIC ##SCOPE, LSW2(R1) ;Say this is not a scope type terminal
133 002772 042761 000000G 000000G BIC ##SLON, LSW7(R1) ;Disable SL editor
134 003000 000167 177214 JMP BS01
135 ;
136 ; SET TT WAIT
137 ;
138 003004 042761 000000G 000000G SETTW: BIC ##SNWTT, LSW6(R1); UNDO SET TT NOWAIT
139 003012 042761 000000G 000000G BIC ##NOWTT, LSW5(R1)
140 003020 000207 RETURN
141 ;
142 ; SET TT NOWAIT
143 ;
144 003022 052761 000000G 000000G SETTNW: BIS ##SNWTT, LSW6(R1); SET TT NOWAIT
145 003030 052761 000000G 000000G BIS ##NOWTT, LSW5(R1)
146 003036 000207 RETURN
147 ;
148 ; SET TT DEAD
149 ;
150 003040 004767 000000G SETDED: CALL CKTERM ;Require TERMINAL privilege
151 003044 032761 000000G 000000G BIT ##DILUP, LSW(R1) ;Is line active now
152 003052 001406 BEQ SETTTB ;Br if not
153 003054 FABORT #RUNMS ;Can't kill an active line
154 ;
155 ; Perform terminal sets that only involve setting or clearing flag bits.
156 ;
157 003064 004767 000000G SETTTP: CALL CKTERM ;Require TERMINAL privilege
158 003070 012405 SETTTB: MOV (R4)+, R5 ;POINT TO TABLE
159 003072 060105 ADD R1, R5 ;POINT TO TABLE ENTRY FOR THIS USER
160 003074 051415 BIS @R4, @R5 ;SET THE DESIRED FLAG
161 003076 000167 177116 JMP BS01
162 ;
163 ; RESET A BIT IN A USER TABLE
164 ;
165 003102 004767 000000G CLRRTTP: CALL CKTERM ;Require TERMINAL privilege
166 003106 012405 CLRRTB: MOV (R4)+, R5 ;POINT TO TABLE
167 003110 060105 ADD R1, R5 ;POINT TO TABLE ENTRY FOR THIS USER
168 003112 041415 BIC @R4, @R5 ;RESET THE DESIRED FLAG
169 003114 000167 177100 JMP BS01

```

CL

```

1
2
3 ;----- SBTTL CL -----
4 ; Process the SET CLn command
5 ;
6 ; Inputs:
7 ; R2 = CL unit number
8 003120 004767 000000G SETCL: CALL CKTERM ;Require TERMINAL privilege for SET CL
9 003124 020227 000000G CMP R2,#CLTOTL ;Is this a valid unit?
10 003130 103404 BLD 3$ ;Br if ok
11 003132 FABORT #EM$IUN ;Invalid CL unit
12 003142 006302 3$: ASL R2 ;Convert unit # to word table index
13 ;
14 ; At this point, R2 contains the index into the CL unit tables.
15 ; Begin loop to process each specified command qualifier.
16 ;
17 003144 004767 000000G 1$: CALL SKPSPC ;Skip over any spaces
18 003150 111300 MOVB (R3),R0 ;Get next character from command
19 003152 001443 BEQ 15$ ;Br if hit the end of the command
20 003154 120027 000054 CMPB R0,#', ;Comma separator?
21 003160 001403 BEQ 4$ ;Br if yes
22 003162 120027 000057 CMPB R0,#'/ ;Slash separator?
23 003166 001005 BNE 5$ ;Br if not
24 003170 005203 4$: INC R3 ;Skip past separator
25 003172 004767 000000G CALL SKPSPC ;Skip over any spaces
26 003176 105713 TSTB (R3) ;Are we at the end of the command now?
27 003200 001424 BEQ 10$ ;Br if yes -- Error
28 ;
29 ; Process the next command qualifier
30 ;
31 003202 012704 001170' 5$: MOV #CLOPHD,R4 ;Point to option table
32 003206 004767 000000G CALL SEARCH ;Search for option keyword
33 003212 103413 BCS 11$ ;Br if cannot identify keyword
34 003214 020427 001624' CMP R4,#CLNCKA ;Does this set command need to check alloc?
35 003220 101004 BHI 51$ ;BR if not
36 003222 016700 000000G MOV R50BUF,R0 ;Get back RAD50 device name
37 003226 004767 000000G CALL CHKALC ;Forbid SET if device is allocated
38 003232 010246 51$: MOV R2,-(SP) ;Save unit index
39 003234 004734 CALL @(R4)+ ;Call routine to process qualifier
40 003236 012602 MOV (SP)+,R2 ;Get back unit number index
41 003240 000741 BR 1$ ;Go see if there are more qualifiers
42 ;
43 ; Invalid qualifier keyword
44 ;
45 003242 11$: FABORT #CSIMS1 ;Invalid option keyword
46 ;
47 ; Invalid command syntax
48 ;
49 003252 10$: FABORT #EM%CSE ;Command syntax error
50 ;
51 ; End of command reached
52 ;
53 003262 000167 000000G 15$: JMP RDCMD ;Finished command
54 ;
55 ; CL unit is not assigned to a line
56 ;
57 003266 STCLNL: FABORT #EM%CLN ;CL unit is not assigned to a line

```

CL

```

58 ;
59 ; Turn on some option
60 ;
61 003276 005762 000000G STCLOP: TST CL$LIX(R2) ; Is this CL unit assigned to a line?
62 003302 001771 BEQ STCLNL ; Br if not
63 003304 051462 000000G BIS (R4),CL$OPT(R2) ; Set desired option flag
64 003310 000207 RETURN
65 ;
66 ; Turn off some option
67 ;
68 003312 005762 000000G CLCLOP: TST CL$LIX(R2) ; Is this CL unit assigned to a line?
69 003316 001763 BEQ STCLNL ; Br if not
70 003320 041462 000000G BIC (R4),CL$OPT(R2) ; Reset the option
71 003324 000207 RETURN
72 ;
73 ; Set a parameter value
74 ;
75 003326 005762 000000G STCLVL: TST CL$LIX(R2) ; Is this CL unit assigned to a line?
76 003332 001755 BEQ STCLNL ; Br if not
77 003334 004767 000000G CALL ACRDEC ; Accrue decimal value
78 003340 011404 MOV (R4),R4 ; Get address of table to store value into
79 003342 060204 ADD R2,R4 ; Point to correct table entry
80 003344 010114 MOV R1,(R4) ; Store value into table
81 003346 000207 RETURN
82 ;
83 ; SET CL [NO]GRAPH
84 ;
85 003350 005762 000000G STCLGR: TST CL$LIX(R2) ; Is this CL unit assigned to a line?
86 003354 001744 BEQ STCLNL ; Br if not
87 003356 011462 000000G MOV (R4),CL$WID(R2) ; Set width to 0 (graph) or 132 (nograph)
88 003362 000207 RETURN
89 ;
90 ; SET CL TOP
91 ;
92 003364 005762 000000G CLTOP: TST CL$LIX(R2) ; Is this CL unit assigned to a line?
93 003370 001736 BEQ STCLNL ; Br if not
94 003372 005062 000000G CLR CL$LIN(R2) ; Say we are at top of the page
95 003376 000207 RETURN
96 ;
97 ; SET CL SPEED=value
98 ;
99 003400 016202 000000G STCLSP: MOV CL$LIX(R2),R2 ; Is this CL unit assigned to a line?
100 003404 001730 BEQ STCLNL ; Br if not
101 003406 004767 000000G CALL ACRSPD ; Accrue speed value
102 003412 116200 000001G MOVB LMXPRM+1(R2),R0 ; Get current flags for line
103 003416 042700 000000G BIC #LP$SPD,R0 ; Clear speed code
104 003422 050005 BIS R0,R5 ; Put in parity and length flags
105 003424 010567 174442 STCLS1: MOV R5,LSPEMT+4 ; Store speed code into EMT arg block
106 003430 006202 ASR R2 ; Convert line index # to line #
107 003432 010267 174432 MOV R2,LSPEMT+2 ; Store line # into EMT arg block
108 003436 012700 000066' MOV #LSPEMT,R0 ; Point to EMT arg block
109 003442 104375 EMT 375 ; Set the speed
110 003444 103401 BCS 2$ ; Br if error
111 003446 000207 RETURN
112 ;
113 ; Error on EMT to set speed
114 ;

```

CL

```

115 003450          2$:      FABORT  #EM$IUN          ;Say invalid unit number
116                ;
117                ; SET CLn XON
118                ;
119 003460  012705  000000G  STCLXN: MOV      #CLSFCH,R5          ;Get .SPFUN code
120 003464  000402          BR          CLSPFN          ;Do the .SPFUN
121                ;
122                ; SET CLn RESET
123                ;
124 003466  012705  000000G  STCLRS: MOV      #CLSFRS,R5          ;Get .SPFUN code
125                ;
126                ; Do a .SPFUN to a CL unit.
127                ; R5 = .SPFUN code.
128                ;
129 003472  005762  000000G  CLSPFN: TST      CL$LIX(R2)          ;Is this CL unit assigned to a line?
130 003476  001673          BEQ      STCLNL          ;Br if not
131 003500  004767  001140          CALL      CLLOOK          ;Open channel 1 to CL unit
132 003504          .SPFUN  #XAREA,#1,R5,#0,#0,#0 ;Do the function
133 003550  103404          BCS      2$          ;Br if error on .SPFUN
134 003552          .CLOSE  #1          ;Close CL channel
135 003560  000207          RETURN
136 003562          2$:      .PURGE  #1          ;Make sure channel is closed
137 003570          FABORT  #EM$IUN          ;Say invalid unit number
138                ;
139                ; SET CL PARITY driver routine
140                ;
141 003600  004767  000000G  STCLPD: CALL      SKPSPC          ;Skip any spaces
142 003604  122327  000075          CMPB     (R3)+,#'=          ;Should have an equal sign
143 003610  001401          BEQ      1$          ;
144 003612  005303          DEC     R3          ;Point to start of keyword
145 003614  012704  001642'  1$:      MOV      #CLPRHD,R4          ;Point to driver table
146 003620  004767  000000G  CALL     SETWRD          ;Process the option
147 003624  000207          RETURN

```


CL

```

1
2 ; -----
3 ; BYPASS forms of SET CLn XON and RESET commands.
4 ; These forms differ from the ordinary forms in that they use
5 ; EMT processing instead of SPFUN calls to the CL handler, and thus
6 ; do not have the overhead or access checking of getting a channel.
7 ;
8 ; SET CLn XONBYPASS Same as SET CLn XON, except uses EMT instead of SPFUN
9 ;
10 ; R2 contains CL unit index
11 ;
12 STCLXB: CALL CHKBCM ; Make common checks for SET CLn xxxBYPASS cmds
13 ASR R2 ; Convert CL unit index back into unit number
14 MOV #CLXEMT,RO ; Point to emt arg block to set cln xon
15 MOV R2,2(RO) ; Set CL unit number into arg block
16 EMT 375 ; Issue the command
17 RETURN
18 ;
19 ; SET CLn RESETBYPASS Same as SET CLn RESETBYPASS, except uses EMT method
20 ;
21 ; R2 contains CL unit index
22 ;
23 STCLRB: CALL CHKBCM ; Make common checks for SET CLn xxxBYPASS cmds
24 ASR R2 ; Convert CL unit index back into unit number
25 MOV #CLREMT,RO ; Point to EMT arg block to SET CLn RESET
26 MOV R2,2(RO) ; Set CL unit number into arg block
27 EMT 375
28 RETURN
29 ;
30 ; Common check routine for SET CLn XONBYPASS and RESETBYPASS
31 ; Inputs:
32 ; R2 contains CL unit index
33 ;
34 CHKBCM: BIT #PO$BYP,PRIVCO ; Are we privileged to use this command?
35 BNE 1$ ; BR if OK
36 FABORT #EM$BYP ; You must have BYPASS privilege for this cmd
37 TST CL$LIX(R2) ; Is specified CL unit connected to a line?
38 BNE 2$ ; Br if so
39 JMP STCLNL ; Else abort if not connected to a line
40 RETURN

```

CL

```

1          ; -----
2          ;   SET CL PARITY={EVEN ODD NONE}
3          ;
4          ;   Inputs:
5          ;   R2 = CL unit number index
6          ;   R4 = Pointer to word with parity control flags
7          ;
8 003726 011404 STCLPR: MOV      (R4),R4          ;Get parity flags
9 003730 016202 000000G STCLP1: MOV      CL$LIX(R2),R2      ;Get number of line CL unit is connected to
10 003734 001406      BEQ      9#          ;Br if not assigned to a line
11 003736 116205 000001G      MOVB     LMXPRM+1(R2),R5 ;Get current control flags for line
12 003742 042705 000000C      BIC     #LP$PAR!LP$ODD,R5 ;Clear parity control flags
13 003746 050405      BIS     R4,R5          ;Put in new flags
14 003750 000625      BR       STCLS1       ;Go set value for line
15 003752      9#:      FABORT  #EM$CLN      ;CL unit is not assigned to a line
16          ;
17          ;   SET CL NOPARITY
18          ;
19 003762 005004 STCLNP: CLR     R4          ;Clear parity flags
20 003764 000761      BR       STCLP1       ;Go do the set

```

CL

```

1
2
3
4 003766 016202 000000G
5 003772 001425
6 003774 004767 000000G
7 004000 116205 000001G
8 004004 042705 000000G
9 004010 020127 000007
10 004014 001003
11 004016 052705 000000G
12 004022 000407
13 004024 020127 000010
14 004030 001404
15 004032
16 004042 000167 177356
17 004046

```

```

; SET CL BITS=n
;
STCLBT: MOV     CL$LIX(R2),R2 ;Get index for line CL unit is assigned to
        BEQ     9$           ;Br if not assigned to a line
        CALL   ACRDEC        ;Accrue parameter value
        MOVB   LMXPRM+1(R2),R5 ;Get current flags for line
        BIC    #LP$7BT,R5    ;Assume 8 bits wanted
        CMP    R1,#7.        ;7 bits wanted?
        BNE    1$           ;Br if not
        BIS    #LP$7BT,R5    ;Set 7 bit flag
        BR     2$
1$:     CMP    R1,#8.        ;8 bits wanted?
        BEQ    2$           ;Br if yes
        FABORT #EM$ICL      ;Invalid character length
2$:     JMP    STCLS1        ;Go set value
9$:     FABORT #EM$CLN      ;CL unit not assigned to line

```

CL

```

1          ; -----
2          ;   SET CL LINE=11
3          ;
4 004056   STCLLN:
5          ;
6          ;   Accrue the line number
7          ;
8 004056   004767   000000G      CALL   ACRDEC           ;Accrue decimal value
9          ;
10         ;   Do the EMT to assign this CL unit to a line and check for errors.
11        ;
12 004062   006202           ASR    R2             ;Convert CL unit index to unit #
13 004064   010267   173754     MOV    R2,CLAEMT+2       ;Set CL unit number
14 004070   010167   173752     MOV    R1,CLAEMT+4       ;Set time-sharing line number
15 004074   012700   000042'    MOV    #CLAEMT,R0      ;Point to EMT argument block
16 004100   104375           EMT    375             ;Do the EMT
17 004102   103045           BCC   20$             ;Br if no error
18        ;
19        ;   An error occurred on the EMT.
20        ;   Print an error message
21        ;
22 004104   113702   000000G    MOVB  @#ERRLOC,R2     ;Get EMT error code
23 004110   120227   000002     CMPB  R2,#2           ;Invalid CL unit # ?
24 004114   001004           BNE   3$             ;Br if not
25 004116           FABORT  #EM$IUN       ;Invalid CL unit number
26 004126   120227   000003    3$:  CMPB  R2,#3           ;Invalid line number?
27 004132   001004           BNE   4$             ;Br if not
28 004134           FABORT  #EM$ILN       ;Invalid line number
29 004144   120227   000004    4$:  CMPB  R2,#4           ;Line already in use by a CL unit?
30 004150   001004           BNE   5$             ;Br if not
31 004152           FABORT  #EM$ACL       ;Line already being used by CL
32 004162   120227   000005    5$:  CMPB  R2,#5           ;Is line in use by a time-sharing user?
33 004166   001004           BNE   6$             ;Br if not
34 004170           FABORT  #EM$TSL       ;Line busy as time-sharing line
35 004200   120227   000006    6$:  CMPB  R2,#6           ;Is this CL unit currently busy?
36 004204   001004           BNE   20$            ;Br if not
37 004206           FABORT  #EM$CLB       ;This CL unit is busy
38        ;
39        ;   Finished
40        ;
41 004216   000207    20$:  RETURN

```

CL

```

1
2 ; -----
3 ; SET CL VERSION=n
4 004220 004767 000000G STCLVR: CALL ACRDEC ;Accrue parameter value
5 004224 120127 000017 CMPB R1,#15. ;Is this a valid argument
6 004230 002403 BLT 9$ ;Br if not
7 004232 110167 000000G MOVB R1,CLVERS ;Set version number
8 004236 000404 BR 10$ ;Return
9 004240 9$: FABORT #EM$IVN ;Invalid version number
10 ;
11 ; Finished
12 ;
13 004250 000207 10$: RETURN
14

```

CL

```

1          ; -----
2          ; SET CLn ENDSTRING='string'
3          ;
4 004252 005762 000000G CLESTR: TST     CL$LIX(R2)      ; Is this CL unit assigned to a line?
5 004256 001002          BNE     2$              ; Br if yes
6 004260 000167 177002   JMP     STCLNL          ; Br if not
7 004264 010446          2$:   MOV     R4,-(SP)
8 004266 010546          MOV     R5,-(SP)
9          ;
10         ; Accrue the string
11         ;
12 004270 004767 000000G         CALL    ACRSTR          ; Accrue the string
13         ;
14         ; Make sure the string is not too long
15         ;
16 004274 020027 000000G         CMP     RO,#CLEDFS      ; Is string too long?
17 004300 101046          BHI     10$              ; Br if yes
18         ;
19         ; Move string over in BLKO to allow room for ENDPAGE value in 1st word
20         ;
21 004302 012705 000000G         MOV     #BLKO,R5        ; Point to start of buffer
22 004306 005200          INC     RO              ; Get length of string with null
23 004310 060005          ADD     RO,R5          ; Point past last char in string
24 004312 010504          MOV     R5,R4
25 004314 062704 000002          ADD     #2,R4          ; Add 1 word offset
26 004320 114544          3$:   MOVB  -(R5),-(R4)    ; Move string over
27 004322 077002          SOB   RO,3$
28 004324 012767 000377 000000G   MOV     #377,BLKO      ; Set value for ENDPAGE (377=no change)
29         ;
30         ; Do lookup on CL unit
31         ;
32 004332 004767 000306          CALL    CLLOOK         ; Do lookup on channel 1 to CL unit
33         ;
34         ; Perform the .SPFUN to set the ENDSTRING
35         ;
36 004336          .SPFUN #XAREA,#1,#CLSFEP,#BLKO,#10.,#0
37 004402          .CLOSE #1          ; Close the channel
38         ;
39         ; Finished
40         ;
41 004410 012605          MOV     (SP)+,R5
42 004412 012604          MOV     (SP)+,R4
43 004414 000207          RETURN
44         ;
45         ; String is too long
46         ;
47 004416          10$:  FABORT #EM$STL      ; String too long

```

CL

```

1          ;-----
2          ; SET CLn TRANSLATE=(ext=int,ext=int,...)
3          ;
4 004426 016201 000000G CLTRNS: MOV     CL$LIX(R2),R1  ;Is this CL unit assigned to a line?
5 004432 001002          BNE     2$           ;Br if yes
6 004434 000167 176626   JMP     STCLNL          ;Br if not
7 004440 004767 000002   2$:   CALL    PRSTRN          ;Parse the translate qualifier
8 004444 000207          RETURN
9          ;-----
10         ;
11         ; Parse a translate qualifier of the form:
12         ; TRANSLATE=(ext=int,ext=int,...)
13         ; and store the translation table for the line.
14         ;
15         ; Inputs:
16         ; R1 = Line index number of line whose translation table is being set up.
17         ; R3 = Pointer past end of keyword
18         ;
19 004446 010446   PRSTRN: MOV     R4,-(SP)
20 004450 010546   MOV     R5,-(SP)
21         ;
22         ; Get pointer to translation table area for this line
23         ;
24 004452 016104 000000G MOV     LCXTBL(R1),R4  ;Get pointer to translation table for line
25 004456 001004          BNE     7$           ;Br if there is one
26 004460          FABORT  #EM#TMT          ;No translation table allocated
27         ;
28         ; See if equal sign and open paren follow keyword
29         ;
30 004470 004767 000000G 7$:   CALL    SKPSPC          ;Skip over any spaces
31 004474 122327 000075   CMPB   (R3)+,#'='     ;Does equal sign follow keyword?
32 004500 001406          BEQ    1$           ;Br if yes
33 004502 126327 177777 000050 CMPB   -1(R3),#'('    ;Did user omit equal sign?
34 004510 001413          BEQ    2$           ;Br if yes
35 004512 005303          DEC    R3           ;Point back to character
36 004514 000447          BR     3$           ;Clear the translation table
37 004516 004767 000000G 1$:   CALL    SKPSPC          ;Skip more spaces
38 004522 122327 000050   CMPB   (R3)+,#'('    ;Should have open paren
39 004526 001404          BEQ    2$           ;Br if got open paren
40 004530          FABORT  #EM#CSE          ;Invalid syntax
41         ;
42         ; Begin to accrue each value pair
43         ;
44 004540 005005   2$:   CLR     R5           ;Count pairs in R5
45         ;
46         ; See if we have reached the end of the list
47         ;
48 004542 004767 000000G 5$:   CALL    SKPSPC          ;Skip any spaces
49 004546 122327 000051   CMPB   (R3)+,#')'     ;At end of the list?
50 004552 001430          BEQ    3$           ;Br if yes
51 004554 005303          DEC    R3           ;Point back to character
52         ;
53         ; See if we have room for another value
54         ;
55 004556 020527 000000G CMP     R5,#MXTTCT     ;Room for another value?
56 004562 103404          BLD    4$           ;Br if yes
57 004564          FABORT  #EM#TMT          ;Too many translation values

```

CL

```

58 004574 005205      4$:      INC      R5          ;Count another value
59                    ;
60                    ; Accrue the external value
61                    ;
62 004576 004767 000000G      CALL      ACROCT      ;Accrue the external value
63 004602 110164 000001      MOVVB     R1,1(R4)    ;Store in high-order byte of word
64                    ;
65                    ; Accrue the internal value
66                    ;
67 004606 004767 000000G      CALL      ACROCT      ;Accrue the internal value
68 004612 110114      MOVVB     R1,(R4)    ;Store in low-order byte of word
69 004614 005724      TST      (R4)+      ;Point to next word
70 004616 004767 000000G      CALL      SKPSPC     ;Skip any spaces
71 004622 122327 000054      CMPB     (R3)+,#'    ;Comma separator?
72 004626 001745      BEQ      5$         ;Br if yes
73 004630 005303      DEC      R3         ;Point back to char we skipped
74 004632 000743      BR      5$         ;Loop and get rest of list
75                    ;
76                    ; We have reached the end of the list
77                    ;
78 004634 005014      3$:      CLR      (R4)          ;Store a zero to terminate the list
79                    ;
80                    ; Finished
81                    ;
82 004636 012605      MOV      (SP)+,R5
83 004640 012604      MOV      (SP)+,R4
84 004642 000207      RETURN

```


CL

```

1          ; -----
2          ; Perform a lookup on channel 1 to a specified CL unit.
3          ;
4          ; Inputs:
5          ; R2 = CL unit index number.
6          ;
7          ; Outputs:
8          ; Channel 1 opened to CL unit.
9          ;
10 004644 010246 CLLOOK: MOV R2, -(SP)
11          ;
12          ; Construct CL device name
13          ;
14 004646 006202          ASR R2          ;Get cl unit number
15 004650 020227 000007    CMP R2, #7.          ;Should this be a CL or C1 unit?
16 004654 101405          BLOS 2$          ;Br if CL
17 004656 162702 000010    SUB #8., R2          ;Remove C1 unit bias
18 004662 066702 173130    ADD R50C10, R2      ;Add "C10" to form device name
19 004666 000402          BR 1$
20 004670 066702 173120 2$: ADD R50CLO, R2      ;Add "CLO" to form device name
21 004674 010267 173104 1$: MOV R2, CLDEV      ;Save device name
22 004700          . LOOKUP #XAREA, #1, #CLDEV; Open channel to CL unit
23 004720 103402          BCS 10$          ;Br if error on lookup
24          ;
25          ; Finished
26          ;
27 004722 012602          MOV (SP)+, R2
28 004724 000207          RETURN
29          ;
30          ; Error on lookup to CL unit
31          ;
32 004726 10$: FABORT #EM$IUN      ;Invalid unit number

```

Host

```

1          .SBTTL          Host
2          ;-----
3          ; SET HOST/DTE=n
4          ; Cross connect time-sharint line with CL unit.
5          ;
6 004736  004767  000000G  SETHST: CALL    CKTERM          ;Require TERMINAL privilege for SET HOST
7          ;
8          ; Do command scanning
9          ;
10 004742  012767  177777  173114      MOV     #-1,TTXCL+4      ;Initially set CL unit to -1
11 004750  012704  001670'      MOV     #HOSTHD,R4      ;Point to option driver table
12 004754  004767  000000G      CALL    SCNOPS          ;Process the command options
13          ;
14          ; Check to make sure the CL unit is assigned to a line and is not
15          ; in use by another user.
16          ;
17 004760  016705  173100      MOV     TTXCL+4,R5      ;Get CL unit number
18 004764  020527  000000G      CMP     R5,#CLTOTL     ;Is this a valid unit number
19 004770  103404          BLO    1$              ;Br if yes
20 004772          FABORT  #EM$IUN      ;Invalid CL unit number
21 005002  006305          1$:   ASL     R5              ;Convert to CL index number
22 005004  010567  173054      MOV     R5,TTXCL+4
23 005010  005765  000000G      TST    CL$LIX(R5)     ;Is this CL unit connected to a line?
24 005014  001004          BNE    2$              ;Br if yes
25 005016          FABORT  #EM$CLN      ;Not connected to line
26 005026  010500          2$:   MOV     R5,R0        ;Get CL unit index
27 005030  004767  000000G      CALL    CKCLUS        ;See if CL unit in use by another job
28 005034  006300          ASL    R0              ;Any job using CL unit?
29 005036  001407          BEQ    3$              ;Br if not
30 005040  120067  000000G      CMPB   R0,CORUSR      ;Is it our job?
31 005044  001404          BEQ    3$              ;Br if yes
32 005046          FABORT  #EM$CLB      ;This CL unit is busy
33          ;
34          ; Perform the connection
35          ;
36 005056  116701  000000G  3$:   MOVB   CORUSR,R1      ;Get our job index number
37 005062  016761  172712  000000G      MOV    R5OHST,LPRG1(R1);Set name of running program to "$HOST$"
38 005070  016761  172706  000000G      MOV    R5OHST+2,LPRG2(R1)
39 005076  012700  000060'      MOV    #TTXCL,R0      ;Point to EMT argument block
40 005102  104375          EMT    375            ;Make the cross connection
41          ;
42          ; Print message saying cross connection broken
43          ;
44 005104  016761  000000G  000000G      MOV    R5OKMN,LPRG1(R1);Reset program name to KMON
45 005112  016761  000002G  000000G      MOV    R5OKMN+2,LPRG2(R1)
46 005120          .PRINT  #TM$XBK
47 005126  000167  000000G      JMP    RDCMD          ;Finished with command
48          ;
49          ; Process the /PORT=ddn qualifier
50          ;
51 005132  010246          HSTPRT: MOV    R2,-(SP)
52 005134  004767  000000G      CALL   CHKEQ          ;Equal sign should follow qualifier name
53 005140  004767  000000G      CALL   GTRD50         ;Accrue the device name
54 005144  122327  000072          CMPB   (R3)+,#'      ;Colon specified with device name?
55 005150  001401          BEQ    1$              ;Br if yes
56 005152  005303          DEC    R3              ;Backup pointer
57 005154  016700  000000G  1$:   MOV    R50BUF,R0      ;Get accrued device name

```

Host

```

58 005160 004767 000000G          CALL    ASNSRC          ;See if this is a logical device name
59 005164 103402                   BCS     2$              ;Br if not logical name
60 005166 016200 000000G          MOV     AT$DEV(R2),R0  ;Get physical device name
61 005172 004767 000000G          2$:    CALL    CHKCLU          ;Convert CL and C1 names to unit numbers
62 005176 103404                   BCS     10$            ;Br if not CL or C1 unit
63 005200 010067 172660          MOV     R0,TTXCL+4    ;Set CL unit number
64 005204 012602                   MOV     (SP)+,R2
65 005206 000207                   RETURN
66 005210                   10$:   FABORT  #EM$CLX          ;Port unit must be CL or C1 unit
67                               ;
68                               ; Process the /DTE=n SET HOST qualifier
69                               ;
70 005220 010146          HSTDTE: MOV     R1,-(SP)
71                               ;
72                               ; Accrue the CL unit number
73                               ;
74 005222 004767 000000G          CALL    ACRDEC          ;Accrue the CL unit number
75 005226 010167 172632          MOV     R1,TTXCL+4    ;Store CL unit index in EMT arg block
76 005232 012601          MOV     (SP)+,R1
77 005234 000207          RETURN
78
79                               .END

```

Errors detected: 0

*** Assembler statistics

Work file reads: 0
 Work file writes: 0
 Size of work file: 12016 Words (47 Pages)
 Size of core pool: 18176 Words (71 Pages)
 Operating system: RT-11

Elapsed time: 00:01:22.17
 ,LP: TSKST2=DK: TSKST2/C/N: SYM

\$1STLG	1-73						
\$BBIT	1-111	5-20	5-21	5-47	5-77	5-78	5-79
\$ALTER	1-111	5-6	5-52	5-53			
\$AUTO	1-87	5-8	16-6	19-5			
\$CARUP	1-85						
\$CCLRN	1-86						
\$CFABT	1-106						
\$CFALL	1-112						
\$CFCCCL	1-112						
\$CFDCC	1-112						
\$CFOPN	1-118						
\$CFSOT	1-110						
\$CHACT	1-61						
\$CLTST	1-96						
\$CTRLC	1-104						
\$CTRLD	1-154						
\$CTRLO	1-61						
\$CTRLS	1-91						
\$DBKMN	1-84						
\$DEAD	1-158	5-10	5-11	15-14			
\$DEBUG	1-155						
\$DEFER	1-124	5-13	5-14				
\$DETCH	1-89						
\$DIBOL	1-73						
\$DILUP	1-108	7-16	14-47	15-8	19-6	23-151	
\$DISCN	1-90						
\$DOOFF	1-114						
\$DUPRN	1-109						
\$ECHO	1-111	5-18	5-19				
\$EMTTR	1-95						
\$FORM	1-110	5-22	5-23				
\$FORMO	1-112	5-24	5-25				
\$HARD	1-158						
\$HITTY	1-72						
\$INCOR	1-128						
\$INDAB	1-159						
\$INDDF	1-157						
\$INDRN	1-157						
\$INIT	1-158						
\$INKMN	1-104						
\$KED	1-128						
\$KINIT	1-68						
\$LC	1-111	5-33	5-34				
\$LOFCF	1-203						
\$MLOCK	1-77						
\$NOIN	1-72						
\$NOINT	1-204						
\$NOWTT	1-72	23-139	23-145				
\$PAGE	1-111	5-36	5-37				
\$PHONE	1-158	5-40	5-41				
\$PRGLK	1-87						
\$QTSET	1-133	23-103	23-120				
\$QUIET	1-125	23-104	23-121				
\$RNIOP	1-205						
\$SCOPE	1-111	5-45	23-132				
\$SGALL	1-124						

DCCRD	1-137								
DCCWR	1-137								
DCTRD	1-137								
DCTWR	1-137								
DEADEV	1-171								
DELSPC	1-185								
DETARG	1-200								
DETHD	1-200								
DETTXT	1-179								
DEVHD1	1-182								
DEVIDL	1-187	1-187	1-188						
DEVUNT	1-174								
DFJMEM	1-68								
DIABFL	1-125	23-91							
DIABLO	1-147	23-92							
DIABNO	1-126	23-90							
DIVIDE	1-180								
DIVSOR	1-193								
DJABMS	1-191								
DKASHD	1-59								
DKSAV	1-164								
DLCEMT	1-30								
DLMSG	1-191								
DLTXT	1-178								
DMTALL	1-191								
DMTARG	1-170								
DMTSUB	1-196								
DOASGN	1-85								
DORUN	1-29								
DOSTOP	1-192								
DTRCOM	18-5	18-7#							
DTREMT	4-57#	18-4*	18-6*	18-9*	18-11				
DVSHH1	1-59								
DVSHH2	1-59								
DVSHH3	1-59								
DVSTAT	1-76								
DZTXT	1-198								
EDIT	1-73								
EDTFIL	1-184								
EM\$ACL	1-54	28-31							
EM\$BYP	1-39	25-35							
EM\$CAP	1-44	11-63							
EM\$CIP	1-53								
EM\$CLB	1-54	28-37	33-32						
EM\$CLN	1-53	24-57	26-15	27-17	33-25				
EM\$CLU	1-34	15-22							
EM\$CLX	1-36	33-66							
EM\$CNO	1-44	10-34							
EM\$CPO	1-44	11-59							
EM\$CSE	1-70	24-49	31-40						
EM\$HNI	1-68								
EM\$ICL	1-52	21-15	27-15						
EM\$IDR	1-42	13-38							
EM\$ILN	1-53	1-54	7-27	14-36	16-20	17-11	18-15	28-28	
EM\$IST	1-42								
EM\$IUN	1-53	24-11	24-115	24-137	28-25	32-32	33-20		

HNBUF	1-174		
HSTHD	5-159#	33-11	
HSTDTE	5-160	5-161	33-70#
HSTPRT	5-162	33-51#	
HUPARG	1-195		
II\$\$SZ	1-46		
II\$FLG	1-46		
II\$NAM	1-46		
II\$NPV	1-48		
II\$PRV	1-48		
IIBUF	1-46		
ILLCMD	1-171		
ILSW2	1-83	14-49	23-32*
IN\$ACT	1-157		
IN\$CMD	1-157		
IN\$CNT	1-157		
INDACT	1-167		
INDERR	1-106		
INDSAV	1-157		
INDSTA	1-106		
INFOMT	1-171		
INGADR	1-46		
INGEMT	1-46		
INSTBL	1-46		
INSTBN	1-47		
INVDAT	1-200		
INVDEV	1-199		
INVEC	1-158		
INVLDM	1-195		
INVLDN	1-172		
INVOPT	1-165	1-171	1-186
INVTIM	1-191		
IOABFL	1-61		
ITRMTP	1-159	23-28*	
JCXPGS	1-142		
JCXSMS	1-198		
JPWDEV	1-39		
JPWFLG	1-39		
JPWTYP	1-39		
JSTKND	1-102		
JSWLOC	1-67		
K52	1-73		
KBMSG	1-180		
KBTX	1-184		
KCSIBF	1-169		
KCSIMS	1-170		
KDOCIN	1-28		
KED	1-73		
KILEMT	1-192		
KL3CLR	1-87		
KL4CLR	1-127		
KMNBAS	1-154		
KMNCHN	1-94		
KMNHI	1-78		
KMNNAM	1-196		
KMNPGS	1-79		

LDSize	1-122							
LF	2-4#							
LF#OPN	1-162							
LF#WRT	1-162							
LFWLIM	1-83							
LICTXT	1-144							
LINBUF	1-78							
LINCNT	1-80							
LINCUR	1-83							
LINFRE	1-191							
LINIR	1-65							
LINNXT	1-78							
LINPNT	1-80							
LINRTS	1-65							
LITIME	1-105							
LJSW	1-102							
LMXLN	1-158							
LMXNUM	1-156							
LMXPRM	1-159	16-7	21-5	22-9	22-17	24-102	26-11	27-7
LNAME	1-60							
LNBLKS	1-107							
LNMAP	1-113							
LNPRIM	1-113	22-25						
LNSBLK	1-108							
LNSPAC	1-117							
LOCKTX	1-180							
LOFSPC	1-110							
LOGASN	1-172							
LOGBAS	1-132	1-134						
LOGBLK	1-161							
LOGBUF	1-161							
LOGCHK	1-133							
LOGCHN	1-161							
LOGCLS	1-176							
LOGDVU	1-132	1-134						
LOGFLG	1-161							
LOGPTR	1-161							
LOMAP	1-142							
LOTBUF	1-81							
LOTNXT	1-81							
LOTPNT	1-81							
LOTSIZ	1-82							
LOTSPC	1-82							
LOUTIR	1-65							
LP#7BT	1-52	21-8	21-11	27-8	27-11			
LP#ODD	1-52	5-86	5-153	22-10	22-18	26-12		
LP#PAR	1-52	5-85	5-86	5-152	5-153	22-10	22-18	26-12
LP#SPD	1-52	16-8	24-103					
LPRG1	1-145	33-37*	33-44*					
LPRG2	1-145	33-38*	33-45*					
LPRI	1-160	12-17*						
LPROG	1-90							
LPROJ	1-90							
LRBFIL	1-114							
LRDTIM	1-80							
LSCCA	1-110							

NOSTRT	1-168		
NOTAVL	1-184		
NOTON	1-190	7-28	
NOTXT	1-178		
NOUDC	1-182		
NSPLDV	1-121		
NSWPMS	1-186		
NUCHN	1-114		
NUMDCD	1-138		
NUMDEV	1-151		
NUMON	1-83		
OCTFIX	1-178		
OCTPRT	1-198		
ODTBAS	1-154		
OF##SZ	1-153		
OF\$DEV	1-152		
OF\$FIL	1-152		
OF\$FLG	1-152		
OF\$UNT	1-152		
OFFEMT	1-192		
OKFEND	1-96		
OKFILE	1-96		
OPRTXT	1-57		
OPTLST	1-43	6-19	
OT\$RON	1-153		
OTHRON	1-194		
OTRMNT	1-196		
OVRCOR	1-168		
PO\$BYP	1-48	25-33	
PO\$DBG	1-47		
PO\$NAM	1-45	10-17	
PO\$OPR	1-45		
PO\$SPV	1-43		
P2\$CGR	1-40		
P2\$TRM	1-45		
PA\$BEL	1-36		
PA\$BLD	1-35		
PA\$DSC	1-35		
PA\$DWD	1-35		
PA\$GRC	1-35		
PA\$HQL	1-35		
PA\$LET	1-35		
PA\$NWD	1-36		
PA\$UKC	1-35		
PA\$ULN	1-35		
PARHD	5-84#	20-10	
PASLIN	1-134		
PAUMSG	1-164		
PBFEND	1-118		
PEKADR	1-38		
PEKEMT	1-38		
PEKSIZ	1-38		
PF\$IOW	1-155		
PF\$SYS	1-155		
PFCO	1-43	4-71	11-11
PFSO	1-43	4-64	11-10

PHYMEM	1-101		
PMBUSY	1-185		
PNAME	1-151	1-174	
POPCF	1-175		
PRGALL	1-29		
PRGSIZ	1-78		
PRGTOP	1-78		
PRIVAO	1-44		
PRIVCO	1-47	10-17	25-33
PRIVC2	1-45		
PRIVSO	1-43		
PRMBUF	1-131		
PRMEND	1-131		
PRMPNT	1-130		
PROSLT	1-52		
PRSTRN	22-26	31-7	31-19#
PRTBUF	1-183		
PRTDAT	1-199		
PRTDC2	1-180		
PRTDC3	1-180		
PRTDEC	1-174	8-24	8-27
PRTFIX	1-177		
PRTFNM	1-183		
PRTLN	1-174		
P RTPCT	1-193		
PRTR50	1-199		
PRTSPC	1-177		
PRTTIM	1-199		
PRTTMD	1-182		
PRTTMV	1-181		
PRTTOD	1-199		
PRTTTP	1-178		
PRTUNM	1-179		
PRTWRN	1-203		
PRVLST	1-47		
PRVOPT	1-43	5-168	
PUSHCF	1-166		
PVCEMT	4-69#	11-33*	11-45
PVNPW	1-43	11-12	
PVON	1-87		
PVSEMT	4-62#	11-32*	11-38
QHDMS1	1-59		
QHDMS2	1-59		
QUME	1-147		
QUMEFL	1-149		
QUMENO	1-149		
R#CHN	1-41		
R#XCHN	1-41		
R50BUF	1-170	24-36	33-57
R50C10	4-7#	32-18	
R50CLO	4-6#	32-20	
R50COM	1-110		
R50DIR	1-166		
R50DK	1-199		
R50DSK	1-172		
R50DUP	1-167		

S#LOW	1-90				
S#MSWT	1-94				
S#OTFN	1-88				
S#OTLO	1-88				
S#OTWT	1-93				
S#QMIO	1-86				
S#QUSR	1-145				
S#RT	1-90				
S#SFWT	1-93	1-145			
S#SPCB	1-146				
S#SPDB	1-146				
S#SPND	1-87				
S#TMWT	1-93				
S#TTFN	1-88				
S#TTSC	1-89				
S#TWFN	1-88				
S9600	1-93	19-8			
SADM3A	5-5	23-81#			
SAUTO	5-7	19-4#			
SBPSUF	1-36				
SCHAIN	1-124				
SCNOPS	1-37	14-56	33-12		
SD\$BAK	1-127				
SD\$DEL	1-119				
SD\$FLK	1-120				
SD\$HLD	1-130				
SD\$SNG	1-129				
SD\$WFM	1-120				
SDBLK	1-121				
SDBU	1-127				
SDBUF1	1-121				
SDCB	1-135				
SDCBND	1-135				
SDCBSZ	1-140				
SDDVU	1-139				
SDFHD	1-132				
SDFLAG	1-120				
SDFORM	1-120				
SDIAB	5-15	23-90#			
SDNAME	1-140				
SDSFCB	1-119				
SDSKIP	1-127				
SEARCH	1-164	24-32			
SERFLG	1-61				
SETCL	1-26	24-8#			
SETDED	5-10	23-150#			
SETDTR	5-16	18-4#			
SETHD	1-173				
SETHST	1-26	33-6#			
SETJMP	23-31	23-33#			
SETNSC	5-46	23-132#			
SETPAR	5-38	20-4#			
SETPRC	1-27	6-6#			
SETPRM	4-16#	14-9*	14-41*	23-26	23-30
SETQUT	5-42	23-100	23-103#		
SETTNW	5-75	23-144#			

SUMS	1-144					
SUPCOD	1-144					
SVT100	1-27	5-65	23-45#			
SVT200	1-27	5-66	5-67	5-70	5-71	23-52#
SVT227	1-26	5-68	23-59#			
SVT228	1-26	5-69	23-66#			
SVT50	1-27	5-72	5-73	23-6#		
SWPTX	1-180					
SXBPNT	1-69					
SYASHD	1-59					
SYHD1	1-179					
SYHD2	1-179					
SYINDX	1-151					
SYNAME	1-152					
SYPSWD	1-39					
SYSAV	1-164					
SYSDAT	1-141					
SYTIMH	1-141					
SYTIML	1-141					
SYUNIT	1-151					
TAB	2-8#					
TALEMT	1-114					
TBLOVF	1-173					
TECO	1-73					
TK1SEC	1-143					
TK1VAL	1-141					
TM\$AUT	1-57					
TM\$CDS	1-56					
TM\$CEN	1-56					
TM\$CLO	1-58					
TM\$CL1	1-58					
TM\$CL2	1-58					
TM\$CL3	1-58					
TM\$CL4	1-58					
TM\$CL5	1-58					
TM\$CL6	1-58					
TM\$CNG	1-56					
TM\$GBL	1-49					
TM\$HPE	1-56					
TM\$HPR	1-55					
TM\$IN1	1-38					
TM\$IN2	1-38					
TM\$LCL	1-49					
TM\$LPR	1-55					
TM\$NAD	1-60					
TM\$NNR	1-38					
TM\$NSD	1-60					
TM\$PR1	1-55					
TM\$PR2	1-55					
TM\$PVA	1-44					
TM\$PVC	1-44					
TM\$RD1	1-49					
TM\$RD2	1-49					
TM\$SDN	1-60					
TM\$XBK	1-37					
TMIDLH	1-71					

TMIOH	1-71						
TMIOWH	1-70						
TMSWPH	1-71						
TMSWTH	1-71						
TMTOTH	1-70	1-193					
TMTOTL	1-70	1-193					
TMUSRH	1-70						
TOTMMS	1-197						
TOTON	1-92						
TOTXT	1-176	8-25					
TRGRET	1-144						
TRMHD1	1-56						
TRMHD2	1-56						
TRMSTR	1-166						
TSKST2	1-5#	1-28					
TSR	1-156						
TSXLN	1-144						
TSXSIT	1-144						
TSXSMS	1-198						
TTHD	5-4#	14-55					
TTXCL	4-40#	33-10*	33-17	33-22*	33-39	33-63*	33-75*
TXTCL	1-91						
UC#MDC	1-163						
UC#NDC	1-163						
UCHAN	1-112						
UCIDEF	1-64						
UCISPC	1-96						
UCLBLK	1-162						
UCLCMD	1-28						
UCLDAT	1-162						
UCLNAM	1-117						
UERSEV	1-134						
UFORM	1-104						
UFPTRP	1-119						
UHIMEM	1-107						
UKMNAM	1-85						
UMSSMS	1-197						
UMSYTP	1-89						
UPTMMS	1-192						
USPLCH	1-92						
USRMMS	1-198						
USRSTK	1-68						
USTART	1-103						
UTRPAD	1-67						
VCORTM	1-138						
VCSHNB	1-95						
VDBFLG	1-82						
VHIPCT	1-136						
VIMAGE	1-102						
VINTIO	1-128						
VLDSYS	1-86						
VNUMDC	1-138						
VPRIDF	1-42						
VQUANO	1-135	1-136					
VQUAN1	1-136						
VQUAN2	1-136						

VQUAN3	1-135	1-136						
VQUN1A	1-136							
VQUN1B	1-128							
VQUN1C	1-128							
VSLEDT	1-203							
VT100	1-147	23-47						
VT10FL	1-149	23-46						
VT10NO	1-149	23-45						
VT200	1-146	23-54						
VT2007	1-146	23-61						
VT2008	1-146	23-68						
VT20FL	1-150	23-53	23-60	23-67				
VT20NO	1-150	23-52	23-59	23-66				
VT52	1-147	23-8						
VT52FL	1-148	23-7						
VT52NO	1-126	23-6						
WILDFL	1-72							
WINSTT	4-82#	23-24						
WLDNAM	2-12#							
XAREA	1-165	23-107	23-110	23-124	23-127	24-132	30-36	32-22
XONEMT	4-52#	17-5*	17-6*	17-7				
YESTXT	1-178							
ZCLR	1-156							

Cross reference table (CREF V05.05)

... CM1	24-132	30-36	32-22									
... CM2	23-107	23-110	23-110	23-124	23-127	23-127	24-132	24-132	24-132	24-132	24-132	24-132
	30-36	30-36	30-36	30-36	30-36	32-22	32-22					
... CM3	24-134	24-136	30-37									
... CM5	8-25	8-28	23-107	23-110	23-124	23-127	24-132	30-36	32-22	33-46		
... CM6	23-107	23-110	23-124	23-127								
. CLOSE	1-20#	24-134	30-37									
. CSIGE	1-18#											
. CSISP	1-16#											
. DATE	1-19#											
. ENTER	1-21#											
. EXIT	1-21#											
. FPROT	1-22#											
. GTIM	1-19#											
. GTLIN	1-19#											
. GVAL	1-22#	23-107	23-124									
. HERR	1-22#											
. LOOKU	1-20#	32-22										
. PRINT	1-20#	8-25	8-28	33-46								
. PURGE	1-17#	24-136										
. PVAL	1-22#	23-110	23-127									
. READW	1-17#											
. REOPE	1-18#											
. SAVES	1-18#											
. SERR	1-22#											
. SPFUN	1-19#	24-132	30-36									
. SRESE	1-16#											
. TTOUT	1-16#											
. TTYIN	1-17#											
. TTYOU	1-17#											
. WRITW	1-21#											
CMDDEF	3-47#	5-5	5-6	5-7	5-8	5-9	5-10	5-11	5-12	5-13	5-14	5-15
	5-16	5-17	5-18	5-19	5-20	5-21	5-22	5-23	5-24	5-25	5-26	5-27
	5-28	5-29	5-30	5-31	5-32	5-33	5-34	5-35	5-36	5-37	5-38	5-39
	5-40	5-41	5-42	5-43	5-44	5-45	5-46	5-47	5-48	5-49	5-50	5-51
	5-52	5-53	5-54	5-55	5-56	5-57	5-58	5-59	5-60	5-61	5-62	5-63
	5-64	5-65	5-66	5-67	5-68	5-69	5-70	5-71	5-72	5-73	5-74	5-75
	5-76	5-77	5-78	5-79	5-85	5-86	5-87	5-94	5-95	5-96	5-97	5-98
	5-99	5-100	5-101	5-102	5-103	5-104	5-105	5-106	5-107	5-108	5-109	5-110
	5-111	5-112	5-113	5-114	5-115	5-116	5-117	5-118	5-119	5-120	5-121	5-122
	5-123	5-124	5-125	5-126	5-127	5-128	5-129	5-130	5-131	5-132	5-133	5-134
	5-135	5-136	5-137	5-138	5-139	5-140	5-145	5-146	5-152	5-153	5-154	5-160
	5-161	5-162	5-168	5-169	5-170	5-171	5-172	5-173	5-174			
FABORT	3-15#	7-27	7-28	10-34	10-35	11-59	11-63	13-38	14-36	15-10	15-16	15-22
	16-20	17-11	18-15	21-15	23-153	24-11	24-45	24-49	24-57	24-115	24-137	25-35
	26-15	27-15	27-17	28-25	28-28	28-31	28-34	28-37	29-9	30-47	31-26	31-40
	31-57	32-32	33-20	33-25	33-32	33-66						
FERR	3-4#	8-22	9-55									
FWARN	3-23#											
TBLDEF	3-35#	5-4	5-84	5-93	5-151	5-159	5-167					
TBLEND	3-61#	5-80	5-88	5-147	5-155	5-163	5-175					

TSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX
usTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusT S&H Computer Systems, Inc. SX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-
PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-Plu sTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTSX-PlusTS