

Table of contents

6-	1	Table of command keywords
10-	1	Job Initialization
11-	1	Entry to KMON
14-	1	Get keyboard command
18-	1	Identify command
21-	1	CALUKM -- Start user-written command processor
22-	1	CMDIND -- IND command
23-	1	INDINI -- Start IND program
24-	1	Process CCL commands
25-	1	INSCF -- See if a command file is installed with priv
26-	1	TRMINI -- Perform terminal-dependent initialization
27-	1	VIRINI -- Virtual line initialization
28-	1	CPYPRN -- Copy context info from parent job
29-	1	PRTGRT -- Print the logon greeting message
30-	1	SETSUF -- Set up a start-up command file

```

1          .TITLE  TSKMN1 -- TSX-Plus Keyboard Monitor
2          .ENABL  LC
3          .DSABL  QBL
4 000000    .ASECT
5          =      42
6 000042    000042' .WORD  KMSTK          ;DEFINE INITIAL STACK POINTER
7          =      300
8 000300    0000000 .WORD  MDT          ;# MINUTES OF SYSTEM UPTIME IF DEMO VERSION
9 000000    .CSECT  TSKMON
10 000000    TSKMON:
11 000000    TSKMN1:
12          ;
13          ; TSKMON is the terminal command processing module of the TSX
14          ; operating system.
15          ;
16          ; TSKMON is broken into three parts:
17          ;
18          ; TSKMN1 is the portion of TSKMON that contains the code to acquire
19          ; a command and branch off to the appropriate command processing routine.
20          ;
21          ; TSKMN2 contains the actual command processing routines.
22          ;
23          ; TSKMN3 contains subroutines.
24          ;
25          ; TSKMON runs in user mode and has memory mapping set up as follows:
26          ; 000000-040000 --> TSGEN
27          ; 140000-157777 --> Job context block
28          ; 160000-177777 --> Monitor vector (MONVEC)
29          ;
30          ; Copyright 1978, 1979, 1980, 1981, 1982, 1983.
31          ; S&H Computer Systems, Inc.
32          ; Nashville, Tennessee
33          ;
34          ;
35          .MCALL  .CSISPC, .TTOUTR, .SRESET
36          .MCALL  .READW, .TTYIN, .TTYOUT, .PURGE
37          .MCALL  .CSIGEN, .SAVEST, .REOPEN
38          .MCALL  .GTLIN, .GTIN, .DATE
39          .MCALL  .PRINT, .CLOSE, .LOOKUP
40          .MCALL  .WRITW, .ENTER, .EXIT
41          .MCALL  .SERR, .HERR, .FPROT, .OVAL, .PVAL
42          ;
43          ; Global definitions
44          ;
45          .GLOBL  INGEMT, INGADR, INPEMT, INPADR, IIBUF
46          .GLOBL  TSKMON, GREET, PRTGRT, GRTFIN, KMNOT1
47          .GLOBL  CTMSG, LGOVER, PPNMSG, BOTEMT, KEYBUF
48          .GLOBL  START, CPUMSG, CF2DEP, PBUFND, KMFTXT
49          .GLOBL  REMNDR, R5OCHR, KEYEND, KMSTK, BADBOT
50          .GLOBL  ALDEMT, DLCEMT, TALEMT, ALCDEV, CDBUF
51          .GLOBL  MONTAB, TSKMON, ASNOVF, AMBOPT, TSKMN1
52          .GLOBL  WRNHED, EM$NUK, ESC, CSHMSG, UCIDEF
53          .GLOBL  EM$DAA, GENMON, CDGEMT, CDGADR, CDPEMT, CDPADR
54          ;
55          ; Global references
56          ;
57          .GLOBL  $NOVLN, P2$VIR, PRIVA2, PRIVS2, SC$ERR, SUCF2

```

```

58 . GLOBL INSSRC, II$PRV, II$NPV, IIBUF, AFCE, II$FLO, EM$NUC
59 . GLOBL ISPF5, ISPF6, ISPF7, ISPF9, PRIVFO, SKPSPC, ISPF11
60 . GLOBL LBSPRI, GETSYP, JPWDEV, JPWTYP, SBPSUF, CFSTRT
61 . GLOBL ABRTAD, ABRTCD, CINFLG, $VNOTT, INIUKE, R50KMN, ABRTOV
62 . GLOBL CORUSR, LSW, SERFLG, IOABFL, KMPRMT, MXPRMT
63 . GLOBL UTRPAD, JSWLOC, ERRLOC, MAXMEM, MAXPRI, JPWFLG
64 . GLOBL USRSTK, $KINIT, CFSTK, MXJMEM, DFJMEM, $SUCF
65 . GLOBL SPUBUF, SXBPNT, MXJADR, LWINDO, CMDSPN, CMDRSM
66 . GLOBL TMTOTH, TMTOTL, TMUSRH, TMIOWH, RUNCHN, RUNNAM
67 . GLOBL TMSWTH, TMIDLH, TMIOH, TMSWPH, CMDHD, $SYSPS
68 . GLOBL VEDIT, WILDFL, $NOIN, $NOWTT, $HITTY, $CHACT
69 . GLOBL PO$DBG, PRIVCO, PRIVAO, PRIVSO, PO$SPV, PVNPW, RUNARG
70 . GLOBL JS$KMN, JS$ON, LMONHD, EM$OVL, II$$SZ, RSTPRV, EM$SFP
71 . GLOBL MSTALC, ALCERR, ABM24, ABM23, ABM22, ABM21, ABM20, SMONHD
72 . GLOBL ABM17, ABM16, ABM15, ABM14, ABM13, ABM12, ABM11, JS$LOG
73 . GLOBL ABM10, ABM7, ABM6, ABM5, ABM4, ABM3, ABM2, ABM1, AB1
74 . GLOBL AB2, AB3, AB4, AB5, AB6, AB10, AB11, AB12, AB13, AB14, AB15, AB16
75 . GLOBL REGEN, STDNAM, IMVT10, IMVT52, IMADM3, IMHAZL, MONTXT, FILERM
76 . GLOBL LOCMSG, TOOLNG, NOCF, NUMPRM, STRLEN, ABCMD, MISUCL
77 . GLOBL URCMD, TM$LNJ, UNSUP, LINNTX, CMDINS
78 . GLOBL TECO, EDIT, KED, K52, $1STLG, $DIBOL, HANCHN
79 . GLOBL SH$VAL, SH$NAM, SH$$SZ, SH$RTN, SH$FLO
80 . GLOBL SO$NVL, SO$OCT, SO$NO, HANENT, HANSIZ, CMDYEL
81 . GLOBL CMDRSY, CMDRUN, CMDRST, CMDSHO, CMDSET, CMDSDN
82 . GLOBL CMDWHO, CMDFMT, CMDDAT, CMDTIM, CMDMNT, CMDMON
83 . GLOBL CMDDMT, CMDDSP, CMDASN, CMDALC, CMDDLC, CMDREM
84 . GLOBL CMOMEM, OPRCMD, CMDSPO, CMDKIL, CMDPAU, CMDDDET, CMDRCL
85 . GLOBL CMDACC, CMDUSE, CMDINI, CMDSQZ, UCLCMD, CMDBOT, CMDBOT
86 . GLOBL CMDSHT, PROFLO, CMDDEF
87 . GLOBL LSW9, $DEBUG, LSW11
88 . GLOBL HAZEL, HAZFL, HAZLN, $MLOCK, MDT
89 . GLOBL LINBUF, LINNXT, LSTACT, PRGTOP, PRGSIZ, KMNHI
90 . GLOBL KMNTOP, KMNPGS, KMNSTK, KMNSTR, CXTAG
91 . GLOBL LINPNT, LINCNT, LACTIV, LRDTIM, CS$RON
92 . GLOBL LOTBUF, LOTNXT, LOTPNT, $VTESE
93 . GLOBL LOTSIZ, LOTSPC, LCOL, UCISPC, $MAPOK
94 . GLOBL LAFSIZ, LFWLIM, LINCUR, NUMON, ILSW2
95 . GLOBL $CARUP, $SLON, $SLTTY, $SLLET, $SLKED
96 . GLOBL LSUCF, $CCLRN, $UKMON, UKMNAM, $UKMRN
97 . GLOBL KL3CLR, $PRGLK, LSW5, PVON, $SPND
98 . GLOBL $TWFN, $TTFN, $OTFN, $IOFN, $OTLO
99 . GLOBL LSTD, FSTD, $DETCH, UMSYTP
100 . GLOBL $DISC, LPROJ, LPROG, LUNAME, UPPN
101 . GLOBL LCPUL, LCPUL, LCONTM, $CTRLS, $SPLJB
102 . GLOBL STPFLC, TOTON, USPLCH, SPLCHN
103 . GLOBL $INWT, $OTWT, $TMWT, $SFWT
104 . GLOBL $MSWT, CFBUF, CFEND, CCLSAV, KMNCHN
105 . GLOBL MINTIM, LSECT, MAXSEC, $EMTTR
106 . GLOBL OKFILE, OKFEND, $CLTST, $NOINT
107 . GLOBL LJSW, CTRLTT, NEWJSW, JSTKND, ERRSPC
108 . GLOBL USTART, GENTOP, BOTDEV, BOTUNI, CMDOFF
109 . GLOBL $CTRLC, LSW2, $INKMN, CHAIN, UFORM, $INDAB
110 . GLOBL MAXASN, $CFABT, INDSTA, INDERR
111 . GLOBL RUNDEV, LNBLKS, CXTBAS, CXTWDS, UHIMEM
112 . GLOBL ASNTBL, $DILUP, LNSBLK, RUNFLG
113 . GLOBL ASNEND, LSW3, LSW2S, $DUPRN
114 . GLOBL $FORM, $TAB, LSCCA, $CFSOT, $RNIOF, $SETRN

```

```

115 . GLOBL $PAGE, $SCOPE, $ECHO, $LC
116 . GLOBL UCHAN, $FORMO, $CFALL, $CFDCC, $CFCCCL
117 . GLOBL LNPRIM, LNMAP, CW$50H, CONFIG, $NTGCC
118 . GLOBL $DOOFF, NUCHN, LRBFIL, CFIND, LPARNT
119 . GLOBL C, CSW, C, DEVQ, C, SBLK, NLINES
120 . GLOBL CD$NAM, CD$DVU, CD$BAS, CD$JOB, CD$$SZ, CD$$UB
121 . GLOBL LTSCMD, LNSPAC, CFNEST, VU$CL, UCLNAM
122 . GLOBL $CFOPN, CFSEND, PBFEND, CFSP, $TTGAG, VUCLMC
123 . GLOBL UFPTRP, SDSFCB, SD$DEL, CFLFL4, $UCLCF, VUCLOR
124 . GLOBL SDFLAG, SD$FLK, SD$WFM, SDFORM, $UCLRN
125 . GLOBL SDBUF1, SDBLK, NSPLDV, LD$RON, $UCLCM, $UCLCL
126 . GLOBL LDNAME, LD$SIZE, LD$FLAG, LD$BASE, LD$DEV
127 . GLOBL $DEFER, CFCHAN, SCHAIN, LD$DEVX
128 . GLOBL CFPNT, CFBLK, $QUIET, DIABFL, KDOCIN, CFSQEZ
129 . GLOBL DIABNO, VT52NO, LA36NO, LA36FL, POPCF
130 . GLOBL LSW4, KL4CLR, SDSKIP, SDBU, SD$BAK
131 . GLOBL $INCOR, $KED
132 . GLOBL SF$BSY, SFFORM, SD$SNG, SFNMBL, NFRESB
133 . GLOBL SD$HLD, SF$HLD, CURPRM, PRMPNT, SF$1ST
134 . GLOBL LSTPRM, PRMBUF, PRMEND, CFSPND
135 . GLOBL SDFHD, SFFLAG, SFQLNK, CFHOLD
136 . GLOBL LCOL, $QTSET, $TECO, CD$TOP
137 . GLOBL $WILD, ERRSEV, UERSEV, PASLIN
138 . GLOBL LSTPL, SDCB, SDCBND
139 . GLOBL VQUAN1, VQUN1A, VQUAN2, VHIPCT
140 . GLOBL DCTRD, DCCRD, DCTWR, DCCWR
141 . GLOBL VCORTM, KMPRMT, MXPRMT
142 . GLOBL RDB, RDBEND, RT$NAM, RT$$SZ
143 . GLOBL SDCBSZ, LSTSL, LSTATE, SJEMT, RJEMT
144 . GLOBL TK1VAL, CINDAT, SYSDAT, SYTIMH, SYTIML
145 . GLOBL BASMAP, LOMAP, HIMAP, JCXPGS
146 . GLOBL TSXLN, TSXSIT, GRT1, TRGRET, LICTXT, SUPCOD, NAMTOP, SUMS, SUCS
147 . GLOBL LPRG1, LPRG2, S$QUSR, S$IQWT, S$SFWT
148 . GLOBL S$SPDB, S$SPCB, SFUSER, SFFILE
149 . GLOBL LCBIT, LA36, LA120, VT52, VT100, DIABLO, GUME
150 . GLOBL ADM3A, LTRMTP, LA12FL, LA12NO, VT52FL
151 . GLOBL VT10FL, VT10NO, QUMEFL, QUMEND, ADM3FL
152 . GLOBL ADM3NO, SYINDX, SYUNIT, NUMDEV, PNAME
153 . GLOBL OF$DEV, OF$UNT, OF$FIL, OF$FLG, SYNAME
154 . GLOBL OF$$SZ, OT$RON, RESDEV, $TAPE
155 . GLOBL KMNBAS, VT2007, VT2008, VT20NO, VT20FL
156 . GLOBL LSW6, $SNWTT, PF$SYS, PF$IOW
157 . GLOBL RSR, TSR, LMXNUM, LSTMX, MXDTR, ZCLR, MXCSR
158 . GLOBL $INDDF, $INDRN, IN$ACT, IN$CNT, IN$CMD, INDSAV
159 . GLOBL $PHONE, INVEC, LMXLN, MXVEC, $INIT, $DEAD
160 . GLOBL ITRMTP, LMXPRM, LSW7, $CFKIL, CFSTS, CF$IND, CF$QUT
161 . GLOBL CFABLV, MONVEC, MAXPRI, MXJPRI
162 . GLOBL LOGCHN, LOGFLG, LOGPTR, LOGBUF, LOGBLK
163 . GLOBL LF$OPN, LF$WRT, UCLBLK, UCLDAT
164 . GLOBL CSHHD, FC$CDX, FC$LNK, FD$NAM, UC$NDC, UC$MDC
165 . GLOBL PLOAD, DORUN, CMDFRM, CMDDSN, STLGCN, DATTIM, PRGALL
166 . GLOBL LDCLN, R50TSX, R50UCL, INDABT
167 . GLOBL CMDBUF, PAUMSG, RDCMD, DKSAV, SYSAV, CVTTAB, RUNHD, SEARCH
168 . GLOBL INVOP, FKILL, ABRTCF, ACRFN, XAREA, FILNAM, NOPRG, FPRINT
169 . GLOBL PUSHCF, TRMSTR, FILNAM, R50DIR, R50SY, R50IND, R50SAV
170 . GLOBL INDACT, R50DUP, R50PIP, R50KED, R50K52, R50KEX, R50COM
171 . GLOBL BLKO, RDERM, R50VIR, NOSTRT, RUNEMT, OVRCDR

```

172	. GLOBL	BADSAV, LDNAM, NOPRG, NOCIN, SIZVAL, ASKLNM, BADCMD, KCSIBF
173	. GLOBL	ASDEX, KCSIMS, QTRD50, R50BUF, R5OLD0, MNTDEV, DMTARG
174	. GLOBL	DEADEV, CHKMNT, CHKMTX, INFOMT, NOFLAG, MTOPHD, INVOPT, ILLCMD
175	. GLOBL	R5OLD, INVLDN, R5ODSK, ACRFIL, BDFNAM, LOGASN, MNTFUL, R5OLD7
176	. GLOBL	TBLOVF, SETHD, CSIMS2, CKPRIV, R5OND, AMBOPT, ACRDEC
177	. GLOBL	MAXAVL, PRTDEC, DEVUNT, PNAME, HANIDX, HNBUFF
178	. GLOBL	ACROCT, HANBSY, CSIMS1, MISSEQ, NOIND
179	. GLOBL	BADPMT, BADPRI, TOTXT, CRLF, HIPRI, STLQHD, LOGCLS, R5OLOG
180	. GLOBL	BDLGOP, SPLHLA, LDOPHD, PRTFIX, PRTSPC
181	. GLOBL	DLTXT, OCTFIX, PRTTTP, NATXT, NOTXT, YESTXT, NINTXT
182	. GLOBL	PRTUNM, SYHD1, SYHD2, PRTLN, SPACE2, DETTXT, SPACE3, RNMS
183	. GLOBL	SWPTX, LOCKTX, SPACE5, PRTDC3, KBMSG, DIVIDE, PRTDC2
184	. GLOBL	COLOO, CPUAH, CPUAL, PRTTMV, NOFIL, CMDBUF, CALUCL
185	. GLOBL	NOUDC, DEVHD1, ASNHD1, ASNHD2, SHMTH1, SHMTH2
186	. GLOBL	CVDVNM, SPACE6, PRTBUF, PRTFNM, NONEMS, NODAT, NOLDMT
187	. GLOBL	SUGARO, EDTFIL, RONTXT, NOTAVL, KBTX, MNFLGS, MNBPC
188	. GLOBL	DELSPC, MNBASE, MNTOP, MONHD, MONAR1, NOPMGN, PMBUSY, MONAR2
189	. GLOBL	NSWPMS, MAXMTX, CURMTX, CHKDLM, SPLHD, INVOPT
190	. GLOBL	DEVIDL, COAL, ALDEX, COAD, SPACTV, SPWFM, DEVIDL, SPSNG
191	. GLOBL	COAL, ALDEX, ALDBLK, COAD, SPACTV, SPWFM, DEVIDL
192	. GLOBL	SPSNG, SPFUL, SPCF, SPFLK, NOFIL, SPQEMT, NOOPTT
193	. GLOBL	BDLIN, MSGBUF, MSGEND, NOTON, GAGMSG, MHNSMS
194	. GLOBL	YELEMT, LINFRE, DJABMS, DLMSG, INVTIM, DMTALL
195	. GLOBL	SHTMSG, AUTHFN, SPLACT, DOSTOP, OFFEMT, KILEMT, UPTMMS
196	. GLOBL	TMTOTH, DIVSOR, TMTOTL, PRTPCT, SUM1, SUM2, SUM3, SUM4
197	. GLOBL	SUM5, SUM6, SUM7, OTHRON, SPLPND, STPASK, SRTSMS
198	. GLOBL	SIZEMT, ASNOVF, INVLDN, CSIMS4, MNTARG, HUPARG, R5OTT
199	. GLOBL	KMNNAM, CCLNAM, OTRMNT, CHKDEV, DMTSUB, CMDCCCL, CHKTTD
200	. GLOBL	SHOHD, SUBTXT, MNTTXT, SRTTXT, TOTMMS, UMSSMS, SSRMAP
201	. GLOBL	TSXSMS, USRMMS, JCXSMS, DZTXT, OCTPRT, CSIMS3, OVLYEQ
202	. GLOBL	PRTR50, PRTDAT, PRTTOD, PRTTIM, INVDEV, ALFN, R5ODK
203	. GLOBL	DETHD, DETARG, RUNMS, NOFRDL, R5OMON, INV DAT, MUL32, COAF

```

1      ;
2      ; Assembly parameters
3      ;
4      ; Greeting message control option
5      ; If GREET=2, full greeting message is printed.
6      ; If GREET=1, truncated greeting message is printed.
7      ; If GREET=0, greeting message is not printed. (SHOULD NEVER BE USED)
8      ;
9      000002      GREET      =      2
10     ;
11     ; Assembly constants
12     ;
13     000012      LF          =      12      ; LINE FEED
14     000015      CR          =      15      ; CARRIAGE RETURN
15     000040      BLANK       =      40      ; ASCII SPACE
16     000007      BELL        =      07      ; ASCII BELL
17     000011      TAB         =      11      ; HORIZONTAL TAB
18     000014      FF          =      14      ; FORM FEED
19     000033      ESC         =      33      ; Escape character
20     000400      BLKWDS      =      256.    ; # OF WORDS IN DISK BLOCK
21     000016      HANCHN      =      16      ; Channel used to access handler files
22     ;
23     ; Flags which are passed to CCL in chain location 510.
24     ;
25     000001      C$TECD      =      1      ; USE TECO AS DEFAULT EDITOR
26     000002      C$WILD      =      2      ; USE IMPLICIT WILDCARD NAMES
27     000004      C$QUIT      =      4      ; SET TT QUIET IS IN EFFECT
28     000010      C$TEST      =      10     ; DISPLAY CCL GENERATED COMMANDS
29     000020      C$KED        =      20     ; DEFAULT EDITOR IS KED
30     000040      C$K52       =      40     ; DEFAULT EDITOR IS K52
31     000100      C$DIBL      =      100    ; DEFAULT TO DIBOL RATHER THAN DBL

```

```

1      ;
2      ; General data area
3      ;
4 000000 075250 102405 057760 UCIDEF: .RAD50 /SY UKMON SAV/ ;Default program for UCI
    000006 073376
5 000010 000000 000000 DIVSOR: .WORD 0,0
6 000014 000000 000000 REMNDR: .WORD 0,0
7 000020 037266 023112 050572 R50MON: .RAD50 /JANFEBMARAPR MAYJUNJUL AUGSEP OCTNOV DEC/
    000026 004322 050601 040726
    000034 040724 004617 073630
    000042 057114 054756 014713
8 000050 000000G 000000G 000000G KCSIMS: .WORD CSIMS1, CSIMS2, CSIMS3, CSIMS4
    000056 000000G
9 000060 IIBUF: .BLKW 14.

```

```

1
2      ; Abort error message index table.
3      ;
4 000114 00000000      . WORD  EM$SFP      ; -31
5 000116 00000000      . WORD  EM$OVL      ; -30
6 000120 00000000      . WORD  MSTALC      ; -27
7 000122 00000000      . WORD  ALCERR      ; -26
8 000124 00000000      . WORD  LGDOVER     ; -25
9 000126 00000000      . WORD  ABM24       ; -24
10 000130 00000000     . WORD  ABM23       ; -23
11 000132 00000000     . WORD  ABM22       ; -22
12 000134 00000000     . WORD  ABM21       ; -21
13 000136 00000000     . WORD  ABM20       ; -20
14 000140 00000000     . WORD  ABM17       ; -17
15 000142 00000000     . WORD  ABM16       ; -16
16 000144 00000000     . WORD  ABM15       ; -15
17 000146 00000000     . WORD  ABM14       ; -14
18 000150 00000000     . WORD  ABM13       ; -13
19 000152 00000000     . WORD  ABM12       ; -12
20 000154 00000000     . WORD  ABM11       ; -11
21 000156 00000000     . WORD  ABM10       ; -10
22 000160 00000000     . WORD  ABM7        ; -7
23 000162 00000000     . WORD  ABM6        ; -6
24 000164 00000000     . WORD  ABM5        ; -5
25 000166 00000000     . WORD  ABM4        ; -4
26 000170 00000000     . WORD  ABM3        ; -3
27 000172 00000000     . WORD  ABM2        ; -2
28 000174 00000000     . WORD  ABM1        ; -1
29 000176 00000000     . WORD  0           ; 0
30 000200 00000000     . WORD  AB1         ; +1
31 000202 00000000     . WORD  AB2         ; +2
32 000204 00000000     . WORD  AB3         ; +3
33 000206 00000000     . WORD  AB4         ; +4
34 000210 00000000     . WORD  AB5         ; +5
35 000212 00000000     . WORD  AB6         ; +6
36 000214 00000000     . WORD  RDERM       ; +7
37 000216 00000000     . WORD  AB10        ; +10
38 000220 00000000     . WORD  AB11        ; +11
39 000222 00000000     . WORD  AB12        ; +12
40 000224 00000000     . WORD  AB13        ; +13
41 000226 00000000     . WORD  AB14        ; +14
42 000230 00000000     . WORD  AB15        ; +15
43 000232 00000000     . WORD  AB16        ; +16

```

ABTMSG:

```

;-----
; MACRO TO CAUSE A FATAL ERROR MESSAGE TO BE PRINTED.
;

```

```

. MACRO FERR MSG
MOV R5, -(SP)
MOV MSG, R5
CALL FPRINT
MOV (SP)+, R5
. ENDM FERR

```

```

;-----
; MACRO TO PRINT A FATAL ERROR MESSAGE, CLEAN UP
; AND THEN JUMP TO RDCMD.
;

```

```

. MACRO FABORT MSG
MOV MSG, R5
JMP FKILL
. ENDM FABORT

```

```

;-----
; MACRO TO START A STANDARD OPTION TABLE.
; NAME = 1 TO 4 CHARACTER TABLE NAME.
; NA = NUMBER OF ARGUMENTS PER TABLE ENTRY.
;

```

```

. MACRO TBLDEF NAME, NA
NARGS = NA
. CSECT CMDVK1
NAME'HD: . WORD 2*NA
. ENDM TBLDEF

```

```

;-----
; MACRO TO ENTER AN OPTION TEXT NAME AND A SET OF PARAMETERS
; INTO THE CURRENTLY OPEN TABLE.
; STRNG = ASCII NAME
; A, B, C = SET OF OPTION PARAMETERS TO STORE IN TABLE WITH NAME.
;

```

```

. MACRO CMDDEF STRNG, A, B, C
. CSECT NAMEK1
L =
. ASCIZ /STRNG/
. CSECT CMDVK1
. WORD L ; POINTER TO NAME STRING
. WORD A
. IIF GE, <NARGS-2> . WORD B
. IIF GE, <NARGS-3> . WORD C
. ENDM CMDDEF

```

```

;-----
; MACRO TO END A SET OF TABLE ENTRIES.
;

```

```

. MACRO TBLEND
. CSECT CMDVK1
. WORD 0
. CSECT TSKMON
. ENDM TBLEND

```

Table of command keywords

	1	SBTTL Table of command keywords
2		-----
3		; Define commands
4		
5		; Arg 1 = command string name
6		; Arg 2 = processing routine
7		
8	000234	TBLDEF CMD, 1
9	000002	CMDDEF R, CMDRSY
10	000006	CMDDEF RU*N, CMDRUN
11	000012	CMDDEF REN*AME, CMDCCCL
12	000016	CMDDEF REM*OVE, CMDREM
13	000022	CMDDEF RES*UME, CMDRSM
14	000026	CMDDEF RESE*T, CMDRST
15	000032	CMDDEF COP*Y, CMDCCCL
16	000036	CMDDEF DIR*ECTORY, CMDCCCL
17	000042	CMDDEF ED*IT, CMDCCCL
18	000046	CMDDEF TY*PE, CMDCCCL
19	000052	CMDDEF PRI*NT, CMDCCCL
20	000056	CMDDEF SH*OW, CMDSHO
21	000062	CMDDEF SE*T, CMDSET
22	000066	CMDDEF SEN*D, CMDSND
23	000072	CMDDEF YE*LL, CMDYEL
24	000076	CMDDEF W*HO, CMDWHO
25	000102	CMDDEF SY*STAT, CMDWHO
26	000106	CMDDEF DEL*ETE, CMDCCCL
27	000112	CMDDEF COM*PILE, CMDCCCL
28	000116	CMDDEF LIN*K, CMDCCCL
29	000122	CMDDEF EX*ECUTE, CMDCCCL
30	000126	CMDDEF COB*OL, CMDCCCL
31	000132	CMDDEF FOR*TRAN, CMDCCCL
32	000136	CMDDEF FORM, CMDFRM
33	000142	CMDDEF FORMA*T, CMDFMT
34	000146	CMDDEF DIB*OL, CMDCCCL
35	000152	CMDDEF MA*CRO, CMDCCCL
36	000156	CMDDEF MAK*E, CMDCCCL
37	000162	CMDDEF DA*TE, CMDDAT
38	000166	CMDDEF TI*ME, CMTIM
39	000172	CMDDEF MO*UNT, CMTMNT
40	000176	CMDDEF MON*ITOR, CMDMON
41	000202	CMDDEF DIS*MOUNT, CMDDMT
42	000206	CMDDEF DISP*LAY, CMDDSP
43	000212	CMDDEF DEF*INE, CMDDEF
44	000216	CMDDEF OF*F, CMDOFF
45	000222	CMDDEF LOG*OFF, CMDOFF
46	000226	CMDDEF KJ*OB, CMDOFF
47	000232	CMDDEF BY*E, CMDOFF
48	000236	CMDDEF AS*SIGN, CMDASN
49	000242	CMDDEF DEA, CMDDSN
50	000246	CMDDEF DEAS*SIGN, CMDDSN
51	000252	CMDDEF ALLO*CATE, CMDALC
52	000256	CMDDEF DEAL*LOCATE, CMDDLCL
53	000262	CMDDEF IND, CMDIND
54	000266	CMDDEF DIF*FERENCES, CMDCCCL
55	000272	CMDDEF DU*MP, CMDCCCL
56	000276	CMDDEF H*ELP, CMDCCCL
57	000302	CMDDEF ME*MORY, CMDMEM

Table of command keywords

58 000306	CMDDEF	BA*CKUP, CMDCCCL
59 000312	CMDDEF	PRO*TECT, CMDCCCL
60 000316	CMDDEF	UNP*ROTECT, CMDCCCL
61 000322	CMDDEF	OP*ERATOR, OPRCMD
62 000326	CMDDEF	SP*OOL, CMDSPD
63 000332	CMDDEF	CR*EATE, CMDCCCL
64 000336	CMDDEF	KI*LL, CMDKIL
65 000342	CMDDEF	PA*USE, CMDPAU
66 000346	CMDDEF	REC*ALL, CMDRCL
67 000352	CMDDEF	DET*ACH, CMDDET
68 000356	CMDDEF	AC*CESS, CMDACC
69 000362	CMDDEF	US*E, CMDUSE
70 000366	CMDDEF	INI*TIALIZE, CMDINI
71 000372	CMDDEF	SQ*UEEZE, CMDSQZ
72 000376	CMDDEF	LIB*RARY, CMDCCCL
73 000402	CMDDEF	TE*CO, CMDCCCL
74 000406	CMDDEF	UC*L, UCLCMD
75 000412	CMDDEF	MU*NG, CMDCCCL
76 000416	CMDDEF	CL*OSE, RDCMD
77 000422	CMDDEF	GT, RDCMD
78 000426	CMDDEF	INS*TALL, CMDINS
79 000432	CMDDEF	LQ*AD, RDCMD
80 000436	CMDDEF	SU*SPEND, CMDSPN
81 000442	CMDDEF	UNL*QAD, RDCMD
82 000446	CMDDEF	BO*OT, CMDBOT
83 000452	CMDDEF	\$ST*OP, CMDBOT
84 000456	CMDDEF	\$SH*UTDOWN, CMDSHT
85 000462	TBLEND	

Table of command keywords

1						
2	000234	000000	000000	000000	ASDEX:	WORD 0,0,0,0
	000242	000000				
3	000244	075250	012445		SYCOM:	RAD50 /SY COM/
4	000250	015270	012445		DKCOM:	RAD50 /DK COM/
5	000254	075250	073376		SYSAB:	RAD50 /SY SAV/
6	000260	015270	073376		DKSAV:	RAD50 /DK SAV/
7	000264	100040			R50TT:	RAD50 /TT /
8	000266	015270			R50DK:	RAD50 /DK /
9	000270	075250			R50SY:	RAD50 /SY /
10	000272	046537			R50LDG:	RAD50 /LDG/
11	000274	016003			R50DSK:	RAD50 /DSK/
12	000276	042614			R50KED:	RAD50 /KED/
13	000300	045130			R50K52:	RAD50 /K52/
14	000302	042640			R50KEX:	RAD50 /KEX/
15	000304	062570			R50PIP:	RAD50 /PIP/
16	000306	015172			R50DIR:	RAD50 /DIR/
17	000310	016130			R50DUP:	RAD50 /DUP/
18	000312	035164			R50IND:	RAD50 /IND/
19	000314	100020			R50TSX:	RAD50 /TSX/
20	000316	101704			R50UCL:	RAD50 /UCL/
21	000320	105372			R50VIR:	RAD50 /VIR/
22	000322	045640			R50LD:	RAD50 /LD /
23	000324	045676			R50LD0:	RAD50 /LD0/
24	000326	045705			R50LD7:	RAD50 /LD7/
25	000330	012445			R50COM:	RAD50 /COM/
26	000332	073376			R50SAV:	RAD50 /SAV/
27	000334	015270	016003		MNTDEF:	RAD50 /DK DSK/
28	000340	004056			ALDEX:	RAD50 /ALN/
29	000342	015270	004051	026760	ALDBLK:	RAD50 /DK ALIGN ALN/
	000350	004056				
30	000352	075250	003273	021113	AUTHFN:	RAD50 /SY ACCESST5X/
	000360	100020				
31	000362	075250	100003	051646	KMNNAM:	RAD50 /SY TSKMONSAV/
	000370	073376				
32	000372	075250	011504	000000	CCLNAM:	RAD50 /SY CCL SAV/
	000400	073376				
33	000402	075250	035164	000000	INDNAM:	RAD50 /SY IND SAV/
	000410	073376				
34	000412	075250	045640	000000	LDNAM:	RAD50 /SY LD SYS/
	000420	075273				
35	000422	043327	053600		R50KMN:	RAD50 /KMON /
36	000426	075250	114730	000000	HNBUF:	RAD50 /SY XXX TSX/
	000434	100020				
37	000436	054730			R50ND:	RAD50 /ND /
38						
39	000440	000000			DEVUNT:	WORD 0
40	000442	000000			ASKLNM:	WORD 0
41	000444	000000			INDRFL:	WORD 0
42	000446	000000			QUOTFL:	WORD 0
43	000450				FILNAM:	BLKW 5
44	000462				XAREA:	BLKW 8
45	000502				R50BUF:	BLKW 2
46	000506				LSTFRM:	BLKW 3
47	000514	000000			CPUAH:	WORD 0
48	000516	000000			CPUAL:	WORD 0
49	000520	000000			HANIDX:	WORD 0

Table of command keywords

50	000522	000000	000000	000000	MNTDEV: .WORD	0,0,0,0	
	000530	000000					
51	000532	000000			CMDEND: .WORD	0	;Pointer past end of current command string
52	000534	000000	000000	000000	ALCDEV: .WORD	0,0,0,0	;Name of dev being allocated or deallocated
	000542	000000					

Table of command keywords

1				
2				; Emt arg block to run a program.
3				
4	000544	000	126	RUNEMT: .BYTE 0,126
5	000546	000000		.WORD 0
6				
7				; Emt arg block to start a spooler.
8				
9	000550	000	126	SPGEMT: .BYTE 0,126
10	000552	000001		.WORD 1
11	000554	000000		.WORD 0
12				
13				; Emt arg block to control detached jobs.
14				
15	000556	000	132	DETARG: .BYTE 0,132
16	000560	000000		.WORD 0
17				
18				; Emt argument block to set the size of the job
19				
20	000562	000	141	SIZEMT: .BYTE 0,141
21	000564	000000		SIZVAL: .WORD 0
22				
23				; Emt arg block to mount a file structure.
24				
25	000566	000	134	MNTARG: .BYTE 0,134
26	000570	000522'		.WORD MNTDEV
27				
28				; Emt arg block to dismount a file structure.
29				
30	000572	000	135	DMTARG: .BYTE 0,135
31	000574	000522'		.WORD MNTDEV
32				
33				; Emt arg block to clean out all entries in directory cache
34				
35	000576	001	135	CSHCLN: .BYTE 1,135
36				
37				; Emt arg block to copy the file status and privileges from a parent job
38				
39	000600	001	160	CPYCXT: .BYTE 1,160
40	000602	000000		.WORD 0
41				
42				; Emt arg block to send a message to a line.
43				
44	000604	000	127	YELEMT: .BYTE 0,127
45	000606	000000		.WORD 0
46	000610	000612'		.WORD MSGBUF
47	000612			MSGBUF: .BLKB 84.
48	000736			MSGEND:
49				
50				; Emt to allocate a device
51				
52	000736	000	156	ALDEMT: .BYTE 0,156
53	000740	000534'		.WORD ALCDEV ;Pointer to device spec
54				
55				; Emt to deallocate a device
56				
57	000742	001	156	DLCENT: .BYTE 1,156

Table of command keywords

58	000744	000534'		.WORD	ALCDEV	
59						
60						; Emt to determine if a device is allocated to another user
61						
62	000746	0002	156	TALEMT:	.BYTE 2,156	
63	000750	000534'			.WORD ALCDEV	
64						
65						; Emt argument block to read a cached device descriptor block into CDBUF
66						
67	000752	000	126	CDGEMT:	.BYTE 0,126	
68	000754	000010			.WORD 10	; Subfunction code (peek)
69	000756	000000		CDGADR:	.WORD 0	; Address of block within kernel
70	000760	0000000			.WORD CD\$\$SZ	; Amt of data to get
71	000762	003402'			.WORD CDBUF	; Destination buffer
72						
73						; Emt argument block to copy a cached device descriptor block from CDBUF
74						; back into the kernel data area.
75						
76	000764	000	126	CDPEMT:	.BYTE 0,126	
77	000766	000011			.WORD 11	; Subfunction code (poke)
78	000770	000000		CDPADR:	.WORD 0	; Address of block within kernel
79	000772	0000000			.WORD CD\$\$SZ	
80	000774	003402'			.WORD CDBUF	; Destination buffer
81						
82						; Emt argument block to move INSTALL entry from kernel into IIBUF
83						
84	000776	000	126	INGEMT:	.BYTE 0,126	
85	001000	000010			.WORD 10	; Subfunction code (peek)
86	001002	000000		INGADR:	.WORD 0	; Address of block within kernel
87	001004	0000000			.WORD II\$\$SZ	; Amt of data to get
88	001006	000060'			.WORD IIBUF	; Destination buffer
89						
90						; Emt argument block to store INSTALL entry into kernel data
91						
92	001010	000	126	INPEMT:	.BYTE 0,126	
93	001012	000011			.WORD 11	; Subfunction code (poke)
94	001014	000000		INPADR:	.WORD 0	; Address of block within kernel
95	001016	0000000			.WORD II\$\$SZ	; Amt of data to move
96	001020	000060'			.WORD IIBUF	; Source buffer
97						
98						; Emt arg block to log off the current job.
99						
100	001022	000	126	OFFEMT:	.BYTE 0,126	
101	001024	000002			.WORD 2	
102	001026	000000			.WORD 0	; 0==>Default time to drop DTR
103						
104						; Emt to update running copy of a handler.
105						
106	001030	000	126	HUPARG:	.BYTE 0,126	
107	001032	0000003			.WORD 3	
108	001034				.BLKW 2	
109						
110						; Emt to reboot the system.
111						
112	001040	000	126	BOTEMT:	.BYTE 0,126	
113	001042	0000004			.WORD 4	
114	001044	004362'			.WORD START	

Table of command keywords

115				
116				; Emt to force logoff of a job.
117				
118	001046	000	126	KILEMT: .BYTE 0,126
119	001050	000005		.WORD 5
120	001052	000000		.WORD 0
121				
122				; Emt to suspend execution of a job
123				
124	001054	000	126	SJEMT: .BYTE 0,126
125	001056	000012		.WORD 12
126	001060	000000		.WORD 0
127				
128				; Emt to resume execution of a job
129				
130	001062	001	126	RJEMT: .BYTE 1,126
131	001064	000012		.WORD 12
132	001066	000000		.WORD 0
133				
134				; EMT arg block to set "hold" mode for a spooled log file
135				
136	001070	0000	151	SPLHLA: .BYTE LOGCHN,151
137	001072	000000		.WORD 0
138	001074	000001		.WORD 1
139				
140				; EMT arg block to copy data from the context block of another job
141				; into our context block.
142				
143	001076	000	126	EMCXCP: .BYTE 0,126
144	001100	000014		.WORD 14
145	001102	000000		.WORD 0 ;# of job we are copying from
146	001104	000000		.WORD 0 ;Address of item being copied
147	001106	000000		.WORD 0 ;Number of bytes to copy
148				
149				; EMT arg block to set the job execution priority
150				
151	001110	000	150	PRIEMT: .BYTE 0,150
152	001112	000000		.WORD 0
153				
154				; EMT arg block to create a window for this job
155				
156	001114	000	161	MAKWIN: .BYTE 0,161
157	001116	001	001	.BYTE 1,1
158	001120	120	020	.BYTE 80.,16.
159	001122	001	000	.BYTE 1,0
160	001124	000000		.WORD 0
161				
162				; EMT arg block to select window 1 as the current window
163				
164	001126	001	161	MAPWIN: .BYTE 1,161
165	001130	001	000	.BYTE 1,0
166				
167				; EMT arg block to delete all temporary display windows for the job
168				
169	001132	002	161	DELWIN: .BYTE 2,161
170	001134	000	000	.BYTE 0,0
171				

Table of command keywords

172				; Emt arg block to initiate performance monitoring.
173				;
174	001136	000	136	MONAR1: .BYTE 0,136
175	001140	000000		MNBASIE: .WORD 0
176	001142	000000		MNTOP: .WORD 0
177	001144	000000		MNBPC: .WORD 0
178	001146	000000		MNFLGS: .WORD 0
179				;
180				; Emt arg block to start performance monitoring.
181				;
182	001150	001	136	MONAR2: .BYTE 1,136
183				;
184				; Emt arg block to broadcast job status change info to monitoring jobs
185				;
186	001152	002	157	GENMON: .BYTE 2,157
187	001154	000000		.WORD 0 ;Store value to broadcast here

```

1 001156          CMDBUF: . BLKW  66.
2 001362          CBFEND:          ; END OF CMDBUF
3 001362          KEYBUF: . BLKW   8.      ; Used by SEARCH to hold keywords
4 001402          KEYEND:          ; End of KEYBUF
5 001402          BLKO: . BLKW  512.
6          001402' KCSIBF =          BLKO      ; OVERLAY WITH BLKO
7          001402' INBUF  =          BLKO      ; OVERLAY WITH BLKO
8 003402          CDBUF: . BLKW  10.      ; Should be at least as large as CD$$SZ
9          ; TSKMON STACK
10 003426          . BLKW  100.
11          003736' KMSTK =          .
12          ;
13          ; BYTE DATA
14          ;
15          ; PRINT BUFFER
16 003736          PRTBUF: . BLKB  256.
17 004336          200          PBUFND: . BYTE  200
18 004337          000          NOFLAG: . BYTE   0
19 004340          000          DOLRAT: . BYTE   0      ; INDICATES #@ SCANNED
20 004341          000          DOTAT: . BYTE   0      ; INDICATES #@ SCANNED
21 004342          000          COLEQL: . BYTE   0      ; INDICATES := SCANNED
22 004343          000          NOUCL: . BYTE   0      ; Non-zero==>Don't call UCL for this command
23 004344          000          NUMMNT: . BYTE   0      ; Counts number of mounted devices
24 004345          123          131          072          SYTXT: . ASCIZ  /SY: /
25          004350          000
26          004351          123          131          072          SYINTX: . ASCIZ  /SY: IND /
27          004354          111          116          104
28          004357          040          000
          ;
          . EVEN
          ;

```

```

1          .SBTTL  Job Initialization
2          ;-----
3          ; ENTER TSKMON AT KMNSTR.
4          ;
5 004362 000240          START:  NOP          ;DEBUGGING BPT INSERTION POINT
6 004364 116701 0000000  MOV     CORUSR,R1    ;GET USER INDEX #
7          ;
8          ; INITIALIZATION PERFORMED FOR LINE THE FIRST TIME
9          ; TSKMON IS ENTERED.
10         ;
11 004370 032761 0000000 0000000  BIT     %%KINIT,LSW(R1) ;IS THIS 1ST ENTRY FOR THIS LINE?
12 004376 001407          BEQ     10$          ;BR IF YES
13 004400 000167 001026          JMP     KMNOT1
14         ;
15         ; Make sure TSGEN hasn't been modified without relinking TSKMON.
16         ;
17 004404 026727 0000000 123456 10$:  CMP     GENTOP,#123456 ;DOES THIS LOOK LIKE THE TOP OF TSGEN?
18 004412 001406          BEQ     22$          ;BR IF OK
19 004414          FERR     #REGEN          ;NEED TO RELINK TSKMON
20 004430 016761 0000000 0000000 22$:  MOV     MINTIM,LCONTM(R1);SET JOB LOG-ON TIME
21 004436 042761 0000000 0000000  BIC     %%CFKIL,LSW6(R1) ;IN CASE OF ^C^C DURING GETSYP
22 004444 042761 0000000 0000000  BIC     %%CTRLC,LSW(R1) ;DO NOT DO CONTROL-C ABORT
23         ;
24         ; Initially set the job name to all blanks
25         ;
26 004452 010105          MOV     R1,R5          ;GET JOB INDEX #
27 004454 070527 0000006          MUL     #6,R5          ;* 12. BYTES PER ENTRY
28 004460 062705 0000000          ADD     #LUNAME,R5      ;POINT TO USER NAME ENTRY FOR THIS LINE
29 004464 012704 000014          MOV     #12.,R4
30 004470 112725 000040          24$:  MOVB    #'',(R5)+      ;SET USER NAME TO BLANKS
31 004474 077403          SOB     R4,24$
32         ;
33         ; Initialize cells in job context block
34         ;
35 004476 012767 0000000 0000000  MOV     #CFSTK,CFSP      ;COMMAND FILE STACK
36 004504 016700 0000000          MOV     DFJMEM,R0      ;GET MAX # KB JOB IS ALLOWED TO USE
37 004510 072027 000012          ASH     #10.,R0        ;CONVERT TO ADDRESS
38 004514 001002          BNE     25$          ;BR IF DIDN'T OVERFLOW 64KB
39 004516 012700 177774          MOV     #177774,R0      ;SET MAX MEMORY AS 64KB
40 004522 010067 0000000          25$:  MOV     R0,MAXMEM      ;SET AS TOP OF MEMORY FOR JOB
41 004526 012767 0000000 0000000  MOV     #NAMTOP,UHIMEM ;HIGHEST LEGAL ADDRESS FOR JOB
42 004534 012767 0000040 0000000  MOV     #SPUBUF+4,SXBPNT;SPOOL FILE BUFFER
43 004542 112767 0000000 0000000  MOVB    #MAXPRI,MXJPRI ;SET MAXIMUM JOB PRIORITY
44 004550 112767 000056 0000000  MOVB    #'. ,KMPRMT   ;SET UP DEFAULT KMON PROMPT (".")
45 004556 112767 000200 0000010  MOVB    #200,KMPRMT+1 ;TERMINATE PROMPT STRING
46 004564 012703 0000000          MOV     #STDNAM,R3     ;SET UP CURRENT (DEFAULT) SPOOL FORM NAME
47 004570 012704 0000000          MOV     #UFORM,R4
48 004574 012700 0000006          MOV     #6,R0
49 004600 112324          3$:  MOVB    (R3)+,(R4)+
50 004602 077002          SOB     R0,3$
51 004604 112767 0000000 0000000  MOVB    #SC$ERR,ERRSEV ;ERROR ABORT SEVERITY LEVEL
52         ;
53         ; Start job with all privileges granted.
54         ; Privileges may be changed by running LOGON or by use of
55         ; the SET PROCESS/AUTH/PRIV=(list) command.
56         ;
57 004612 012700 0000000          MOV     #PVNPW,R0      ;Get # words in each privilege vector

```

Job Initialization

```

58 004616 005002          CLR      R2          ;Set index for 1st word
59 004620 012703 177777    MOV      #177777,R3    ;Get all privilege flags turned on
60 004624 010362 0000000    77$:    MOV      R3,PRIVA0(R2) ;Authorized privileges
61 004630 010362 0000000    MOV      R3,PRIVS0(R2) ;Set privileges
62 004634 010362 0000000    MOV      R3,PRIVF0(R2) ;Command file privileges
63 004640 010362 0000000    MOV      R3,PRIVC0(R2) ;Current (program) privileges
64 004644 062702 000002    ADD      #2,R2          ;Increment vector index
65 004650 077013          SOB      R0,77$         ;Set all privileges
66 004652 020127 0000000    CMP      R1,#LSTPL     ;Is this a primary line?
67 004656 003016          BGT      32$           ;Br if not
68 004660 032761 0000000 0000000    BIT      ##NOVLN,LSW2(R1);Disallow subprocess usage?
69 004666 001412          BEQ      32$           ;Br if don't need to disallow subprocesses
70 004670 012700 0000000    MOV      #P2$VIR,R0     ;Get subprocess privilege flag
71 004674 040067 0000000    BIC      R0,PRIVA2     ;Remove subprocess privilege
72 004700 040067 0000000    BIC      R0,PRIVS2
73 004704 040067 0000000    BIC      R0,PRIVF0
74 004710 040067 0000000    BIC      R0,PRIVC0
75 004714 004767 0000000    32$:    CALL     RSTPRV      ;Setup some other privilege flags
76                                ;
77                                ; Open spool file channel
78                                ;
79 004720 105767 0000000    TSTB     NSPLDV         ;ARE THERE ANY SPOOLED DEVICES?
80 004724 001410          BEQ      29$           ;BR IF NOT
81 004726          .REOPEN #XAREA,#USPLCH,#SPLCHN ;OPEN CHANNEL FOR WRITES TO SPOOL FILE
82                                ;
83                                ; Initialize TSXUCL data base for this job
84                                ; (Say that there are no user-defined commands yet)
85                                ;
86 004746 016705 0000000    28$:    MOV      UCLBLK,R5     ;Get # blocks in file per job
87 004752 001446          BEQ      2$           ;Br if TSXUCL data file not needed
88 004754          .LOOKUP #XAREA,#1,#UCLDAT ;Lookup TSXUCL data file
89 004774 103435          BCS      2$           ;Br if not there
90 004776 010100          MOV      R1,R0          ;Get job index number
91 005000 006200          ASR      R0            ;Convert to job number
92 005002 005300          DEC      R0            ;Make first job # 0
93 005004 070500          MUL      R0,R5          ;Compute block number of data for this job
94 005006 012702 001402'    MOV      #BLK0,R2      ;Point to data buffer
95 005012 012762 177777 0000000    MOV      #-1,UC$NDC(R2) ;Set flag saying TSXUCL should init data
96 005020 016762 0000000 0000000    MOV      VUCLMC,UC$MDC(R2) ;Set maximum allowed number of commands
97 005026          .WRITW #XAREA,#1,R2,#256.,R5 ;Write out data for job
98 005062          .CLOSE #1                    ;Close TSXUCL file
99                                ;
100                               ; Set default editor
101                               ;
102 005070 126727 0000000 0000000 2$:    CMPB     VEDIT,#EDIT    ;IS DEFAULT EDITOR EDIT?
103 005076 001413          BEQ      7$           ;BR IF YES
104 005100 126727 0000000 0000000    CMPB     VEDIT,#TECO     ;IS DEFAULT EDITOR TECO?
105 005106 001004          BNE      21$          ;BR IF NOT
106 005110 052761 0000000 0000000    BIS      #$TECO,LSW5(R1) ;SET DEFAULT EDITOR FOR JOB
107 005116 000403          BR       7$           ;
108 005120 052761 0000000 0000000 21$:    BIS      #$KED,LSW5(R1) ;DEFAULT EDITOR MUST BE KED
109                                ;
110                               ; Set default UCL FIRST/MIDDLE/LAST
111                               ;
112 005126 126727 0000000 0000001 7$:    CMPB     VUCLOR,#1      ;SET UCL FIRST?
113 005134 001003          BNE      30$          ;BR IF NOT
114 005136 052761 0000000 0000000    BIS      #$UCLCF,LSW7(R1)

```

Job Initialization

```

115 005144 126727 0000000 0000002 30$: CMPB VUCLOR,#2 ;SET UCL MIDDLE?
116 005152 001003 BNE 29$ ;BR IF NOT
117 005154 052761 0000000 0000000 BIS ##UCLCM,LSW7(R1)
118 005162 126727 0000000 0000003 29$: CMPB VUCLOR,#3 ;SET UCL LAST?
119 005170 001003 BNE 27$ ;BR IF NOT
120 005172 052761 0000000 0000000 BIS ##UCLCL,LSW7(R1)
121 ;
122 ; Set default SL mode
123 ;
124 005200 052761 0000000 0000000 27$: BIS ##SLLET,LSW7(R1);Default SL substitution on
125 ;
126 ; Set default wildcard flag
127 ;
128 005206 005727 0000000 31$: TST #WILDFL ;EXPLICIT OR IMPLICIT WILDCARDS?
129 005212 001403 BEQ 8$ ;BR IF EXPLICIT
130 005214 052761 0000000 0000000 BIS ##WILD,LSW5(R1) ;SET IMPLICIT WILDCARD FLAG
131 005222 020127 0000000 8$: CMP R1,#LSTD L ;REAL OR VIRTUAL LINE?
132 005226 003022 BGT 12$ ;BR IF VIRTUAL
133 005230 020127 0000000 CMP R1,#LSTPL ;REAL OR DETACHED?
134 005234 003021 BGT 1$ ;BR DETACHED
135 ;
136 ; See if we need to get a system password (primary line only).
137 ;
138 005236 032761 0000000 0000000 BIT ##SYSPS,LSW2(R1);Do we need to accept a system password?
139 005244 001402 BEQ 13$ ;Br if not
140 005246 004767 0000000 CALL GETSYP ;Accept system password
141 ;
142 ; Mount SY device for this job (primary line only).
143 ;
144 005252 016767 173012 173242 13$: MOV R50SY,MNTDEV ;SET SY AS DEVICE TO MOUNT
145 005260 012700 000566' MOV #MNTARG,R0 ;MOUNT SY
146 005264 104375 ENT 375
147 ;
148 ; Terminal dependent initialization for primary lines.
149 ;
150 005266 004767 004702 CALL TRMINI ;Do terminal-dependent initialization
151 005272 000402 BR 1$
152 ;
153 ; Initialization for subprocesses only.
154 ; Print n> to show which subprocess we are switching to.
155 ;
156 005274 004767 005156 12$: CALL VIRINI ;Do virtual line initialization
157 ;
158 ; Do some initialization for detached jobs
159 ;
160 005300 032761 0000000 0000000 1$: BIT ##DETCH,LSW(R1) ;Is this a detached job?
161 005306 001403 BEQ GRTINI ;Br if not
162 005310 012761 0000000 0000000 MOV ##ECHO!$LC!$DEFER,LSW2(R1) ;Init control flags
163 ;
164 ; Print TSX greeting message
165 ;
166 005316 004767 005774 GRTINI: CALL PRTGRT ;Print the TSX-Plus logon greeting
167 005322 016161 0000000 0000000 MOV LSW2(R1),LSW2S(R1) ;SAVE IN CASE OF CTRL-C REENTRY
168 ;
169 ; If this job has a parent job, copy file context and privilege from parent
170 ;
171 005330 004767 005254 CALL CPYPRN ;Copy info from parent job

```

Job Initialization

```

172      ;
173      ; Set flag saying job initialization has been done
174      ;
175 005334 052761 0000000 0000000      BIS      ##KINIT,LSW(R1) ;SAY LINE HAS BEEN INITIALIZED
176      ;
177      ; See if this line has an associated start-up command file.
178      ;
179 005342 020127 0000000      CMP      R1,#LSTD1      ;Is this a virtual line?
180 005346 101004      BHI      1$      ;Br if yes
181 005350 016102 0000000      MOV      LSUCF(R1),R2      ;IS THERE A START-UP COMMAND FILE?
182 005354 001407      BEQ      61$      ;BR IF NOT
183 005356 000402      BR      2$
184 005360 012702 0000000      1$: MOV      #SBPSUF,R2      ;Point to job context cell with file name
185 005364 105712      2$: TSTB      (R2)      ;IS FILE NAME NULL?
186 005366 001402      BEQ      61$      ;BR IF YES
187 005370 004767 006324      CALL     SETSUF      ;Set up start-up command file for execution
188      ;
189      ; Broadcast status message to monitoring jobs telling them that
190      ; this job logged on.
191      ;
192 005374 012767 0000000 173552 61$: MOV      #JS$ON,GENMON+2 ;Set logged-on status code
193 005402 012700 001152'      MOV      #GENMON,R0      ;Point to EMT argument block
194 005406 104375      EMT      375      ;Tell monitoring jobs that we are initiated
195 005410 005761 0000000      TST      LPARNT(R1)      ;Do we have a parent job?
196 005414 001406      BEQ      KMNOT1      ;Br if not
197 005416 012767 0000000 173530      MOV      #JS$LOG,GENMON+2;Set status code saying we have logged on
198 005424 012700 001152'      MOV      #GENMON,R0      ;Point to EMT argument block
199 005430 104375      EMT      375      ;Broadcast status code that job logged on

```

```

1          .SBTTL  Entry to KMON
2          ;-----
3          ; End of initialization code that is performed during job login.
4          ; Begin processing that is performed each time KMON is entered.
5          ;
6 005432 016761 172764 0000000 KMNDT1: MOV      R50KMN,LPRG1(R1);SET "KMON" AS RUNNING PROGRAM NAME
7 005440 016761 172760 0000000      MOV      R50KMN+2,LPRG2(R1)
8 005446 005067 0000000      CLR      UTRPAD      ;CLEAR USER TRAP CONTROL
9 005452 005067 0000000      CLR      UFPTRP      ;RESET FLOATING POINT TRAP CONTROL
10 005456 005061 0000000      CLR      LSCCA(R1)    ;RESET .SCCA TRAP CONTROL
11 005462 105067 0000000      CLRB     SERFLG      ;DO .HERR
12 005466 105067 0000000      CLRB     RUNARG      ;No argument string from RUN command
13 005472 005767 0000000      TST      UCHAN      ;DID USER DO A .CDFN?
14 005476 001404      BEQ      4$      ;BR IF NOT
15 005500 005067 0000000      CLR      UCHAN      ;UNDO THE .CDFN
16 005504 004767 0000000      CALL     PRGALL      ;PURGE ALL CHANNELS
17 005510 005061 0000000      4$:  CLR      LTSCMD(R1) ;NO PENDING SPECIAL COMMAND
18 005514 042761 0000000 0000000      BIC      ##SETRN,LSW9(R1);Say SETUP is no longer running
19 005522      .CLOSE     #RUNCHN      ;CLOSE PROGRAM SAV FILE
20 005530 004767 0000000      CALL     RSTPRV      ;Reset job privileges
21 005534 032761 0000000 0000000      BIT      ##DUPRN,LSW6(R1);WAS DUP THE PROGRAM THAT EXITED?
22 005542 001407      BEQ      7$      ;BR IF NOT
23 005544 004767 0000000      CALL     PRGALL      ;PURGE ALL CHANNELS
24 005550 004767 0000000      CALL     LDCLN      ;DO "LD CLEAN" OPERATION
25 005554 042761 0000000 0000000      BIC      ##DUPRN,LSW6(R1);SAY DUP IS NO LONGER RUNNING
26 005562 032761 0000000 0000000 7$:  BIT      ##DOOFF,LSW(R1) ;SHOULD WE LOG USER OFF?
27 005570 001402      BEQ      6$      ;BRANCH IF NOT
28 005572 000167 0000000      JMP      CMDOFF      ;FORCE LOGOFF
29          ; CHECK FOR SYSTEM ABORT
30 005576 116702 0000000      6$:  MOVB     ABRTCD,R2      ;WAS USER ABORTED?
31 005602 001467      BEQ      NOABRT      ;BRANCH IF NOT
32          ;
33          ; USER WAS ABORTED -- PUT OUT ABORT MESSAGE.
34          ;
35 005604 112767 000010 0000000      MOVB     #10,VERSEV      ;SAY ERROR SEVERITY LEVEL = SEVERE
36 005612 006302      ASL      R2      ;CVT ERROR CODE TO WORD INDEX
37 005614 016203 000176'      MOV      ABTMSG(R2),R3      ;GET ADDR OF ERROR MESSAGE
38 005620      .PRINT     #MONTXT
39 005626      .PRINT     R3
40          ; Print name of file that caused the error
41 005632 005767 0000000      TST      ERRSPC      ;Do we have a file spec to print?
42 005636 001414      BEQ      1$      ;Br if not
43 005640      .PRINT     #FILERM      ;Print heading message
44 005646 012703 001402'      MOV      #BLKO,R3      ;Point to buffer where result is to be stored
45 005652 012704 0000000      MOV      #ERRSPC,R4      ;Point to RAD50 file spec
46 005656 004767 0000000      CALL     EDTFIL      ;Convert file spec to ascii
47 005662      .PRINT     #BLKO      ;Print the file spec
48          ; PUT OUT LOCATION OF ABORTED INSTRUCTION.
49 005670      1$:  .PRINT     #LOCMSG
50 005676 016702 0000000      MOV      ABRTAD,R2      ;GET ADDR OF ABORT INST
51 005702 004767 0000000      CALL     OCTPRT      ;PRINT THE OCTAL VALUE
52 005706 020227 120000      CMP      R2,#120000      ;IN OVERLAY REGION?
53 005712 000407      BR      10$      ;*** skip overlay this version ***
54          ;
55 005714      .PRINT     #OVLREQ      ;PRINT " OVERLAY = "
56 005722 016700 0000000      MOV      ABRTOV,R0      ;GET OVERLAY NAME
57 005726 004767 0000000      CALL     PRTR50      ;AND DISPLAY IT

```

58	005732			10#:	.PRINT	#CRLF	
59	005740	005067	00000000		CLR	ABRTAD	; CLEAN ABORT ADDR
60	005744	105067	00000000		CLRB	ABRTCD	; AND CODE
61	005750	042737	004440 00000000		BIC	#004440, @#JSWLOC	; RELEASE CHAIN AREA, PASLIN, AND SCHAIN
62	005756	105067	00000000		CLRB	CINFLG	; KILL ANY CHAIN REQUEST

Entry to KMON

```

1
2      ; User did not abort. See if this is a .CHAIN request
3
4 005762 005067 000000G NOABRT: CLR      ERRSPC      ;CLEAR ANY ERROR FILE SPEC
5 005766 032761 000000C 000000G      BIT      #$CFKIL!$CFABT,LSW6(R1) ;REQUEST TO ABORT ALL COM FILES?
6 005774 001405          BEQ      1$          ;BR IF NOT
7 005776 105067 000000G      CLRB     CINFLG      ;CLEAR ANY CHAIN REQUEST
8 006002 042737 000000C 000000G      BIC      #PASLIN!$CHAIN,@#JSWLOC ;KILL ANY PASSED COMMAND
9 006010 105767 000000G 1$:      TSTB     CINFLG      ;IS THIS .CHAIN REQUEST?
10 006014 001402          BEQ      NDCIN         ;BRANCH IF NOT
11 006016 000167 000000G      JMP      KDOCIN      ;GO DO THE CHAIN
12
13      ; Not .CHAIN request so enter normal KMON.
14
15      ; Delete any temporary display windows for this job
16
17 006022 012700 001132' NDCIN:  MOV      #DELWIN,R0      ;Point to EMT argument block
18 006026 104375          EMT      375          ;Delete all temporary windows for job
19
20      ; Clean up various cells
21
22 006030 012761 000040 000000G      MOV      #BLANK,LRBFIL(R1) ;USE SPACE FOR RUBOUT FOR KMON
23 006036 016161 000000G 000000G      MOV      LSW2S(R1),LSW2(R1);RESET STATUS OF LSW2
24 006044 042761 000000G 000000G      BIC      #KL4CLR,LSW4(R1)
25 006052 042761 000000G 000000G      BIC      #$VTESE,LSW5(R1);TURN OFF VT52 ESC-LETTER ACTIVATION
26 006060 042761 000000C 000000G      BIC      #$DEBUG!$RNIOP,LSW9(R1);Not debugger, I/O page
27 006066 005061 000000G      CLR      LNSPAC(R1)      ;NO SPECIAL ACTIVATION CHARS
28 006072 005061 000000G      CLR      LRDTIM(R1)      ;CLEAR TT READ TIME-OUT VALUE
29 006076 005061 000000G      CLR      LAFSIZ(R1)      ;CLEAR FIELD WIDTH ACTIVATION
30 006102 005061 000000G      CLR      LFWLIM(R1)      ;CLEAR FIELD WIDTH LIMIT
31 006106 042761 000000G 000000G      BIC      #KL3CLR,LSW3(R1);CLEAR MISC BITS IN LSW3
32 006114 042761 000000G 000000G      BIC      #$CFDCC,LSW4(R1);CLEAR DEFERRED-CTRL-C FLAG
33 006122 042761 000000C 000000G      BIC      #<$NOWTT!$CHACT>,LSW5(R1);TURN OFF NO WAIT AND SINGLE CHAR ACT
34 006130 042761 000000G 000000G      BIC      #$NOINT,LSW7(R1);Reset non-interactive run switch
35 006136 042761 000000G 000000G      BIC      #$NTGCC,LSW9(R1);Clear non-terminating .GTLIN ctrl-C flag
36 006144 032761 000000G 000000G      BIT      #$SNWTT,LSW6(R1);DID USER DO "SET TT NOWAIT"?
37 006152 001403          BEQ      3$          ;BR IF NOT
38 006154 052761 000000G 000000G      BIS      #$NOWTT,LSW5(R1);SET NO-WAIT FLAG
39 006162 016700 000000G 3$:      MOV      CFSPND,R0      ;WAS A COMMAND FILE SUSPENDED?
40 006166 001404          BEQ      4$          ;BR IF NOT
41 006170 004767 000000G      CALL     CFSTRT      ;Restart it
42 006174 005067 000000G      CLR      CFSPND
43
44      ; Check if program specified an error severity level.
45
46 006200 156767 000000G 000000G 4$:      BISB     UERSEV,INDERR      ;SAVE ERROR STATUS FOR IND
47 006206 126767 000000G 000000G      CMPB     UERSEV,ERRSEV      ;DID PROGRAM SPECIFY ERROR SEVERITY?
48 006214 103413          BLO      1$          ;BR IF OK
49 006216 004767 000000G      CALL     ABRTCF      ;SEVER ERROR -- ABORT COMMAND FILE
50 006222 042761 000000G 000000G      BIC      #CCLRN,LSW5(R1);SAY CCL NOT RUNNING
51 006230 032761 000000G 000000G      BIT      #$INDAB,LSW7(R1);DOES HE WANT TO ABORT IND COMMAND FILES?
52 006236 001402          BEQ      1$          ;BR IF NOT
53 006240 004767 000000G      CALL     INDABT      ;Abort IND execution
54 006244 105067 000000G 1$:      CLRB     UERSEV      ;CLEAR ERROR STATUS
55
56      ; Purge all of user's channels
57

```

Entry to KMON

```
58 006250 004767 0000000 CALL PRGALL ;PURGE ALL CHANNELS
59 006254 032761 0000000 0000000 BIT ##CFKIL,LSW6(R1);Should we abort IND and command files?
60 006262 001403 BEQ 2$ ;Br if not
61 006264 004767 0000000 CALL INDABT ;Abort IND and nested command files
62 006270 000406 BR 5$
63 006272 032761 0000000 0000000 2$: BIT ##CFABT,LSW6(R1);SHOULD WE ABORT ALL ACTIVE COMMAND FILES?
64 006300 001406 BEQ 6$ ;BR IF NOT
65 006302 004767 0000000 CALL ABRTCF ;ABORT ALL ACTIVE COMMAND FILES
66 006306 042761 0000000 0000000 5$: BIC ##CCLRN,LSW5(R1);STOP EXECUTION OF CCL
67 ;
68 ; Tell any jobs that are monitoring us that we just entered TSKMON
69 ;
70 006314 005761 0000000 6$: TST LMONHD(R1) ;Are we being monitored?
71 006320 001003 BNE 7$ ;Br if yes
72 006322 005767 0000000 TST SMONHD ;Anyone monitoring all jobs?
73 006326 001406 BEQ CKPASL ;Br if not
74 006330 012700 001152' 7$: MOV #GENMON,R0 ;Point to EMT argument block
75 006334 012760 0000000 000002 MOV #JS#KMN,2(R0) ;Set status code
76 006342 104375 EMT 375 ;Broadcast status message
```

Entry to KMON

```

1
2      ; See if program passed a set of command lines to Kmon
3      ; when it exited.
4
5 006344 032737 000000C 000000G CKPASL: BIT      #PASLIN!SCHAIN,@#JSWLOC;DID PROGRAM PASS US A COMMAND?
6 006352 001546      BEQ      4$      ;BR IF NOT
7
8      ; Program did pass a set of commands to Kmon.
9      ; Set it up to look like a fake command file.
10     ; That is, the commands are stored in the command file buffer so that
11     ; they are read as if they came from a command file, but no actual
12     ; file is open.
13
14     ; Determine if we should abort the currently open command file.
15
16 006354 032761 000000C 000000G      BIT      #CCLRN!$INDRN,LSW5(R1);IS CCL OR IND RUNNING?
17 006362 001006      BNE      5$      ;BR IF YES
18 006364 032737 000000G 000000G      BIT      #SCHAIN,@#JSWLOC;SHOULD BE ABORT CURRENT COMMAND FILE?
19 006372 001002      BNE      5$      ;BR IF NOT
20 006374 004767 000000G      CALL     ABRTCF      ;ABORT ALL CURRENTLY OPEN COMMAND FILES
21 006400 005767 000010G 5$:      TST      CINDAT+10      ;DID HE PASS US A NULL COMMAND FILE?
22 006404 001531      BEQ      4$      ;IF YES THEN WE ARE FINISHED
23
24     ; If command line is being passed to us by the TSXUCL program,
25     ; check to see if this is a case where TSXUCL could not recognize
26     ; the command and is passing it back to us for processing.
27     ; If so, just move the command to the command buffer and then proceed
28     ; with normal KMON command checking.
29
30 006406 032761 000000G 000000G      BIT      #UCLRN,LSW7(R1);Is TSXUCL program running?
31 006414 001432      BEQ      15$      ;Br if not
32 006416 126727 000012G 000077      CMPB     CINDAT+12,#'?' ;Is TSXUCL throwing command back to us?
33 006424 001026      BNE      15$      ;Br if not
34 006426 012703 000013G      MOV      #CINDAT+13,R3 ;Point to chain data area (past "?" char)
35 006432 012702 001156'      MOV      #CMDBUF,R2 ;Point to command buffer
36 006436 112322      16$:      MOVB     (R3)+,(R2)+ ;Move command to command buffer
37 006440 001376      BNE      16$      ;Loop till asciz null moved
38 006442 005302      DEC      R2 ;Make R2 point to null at end of command
39 006444 010267 172062      MOV      R2,CMDEND ;Save pointer to end of command
40 006450 042761 000000G 000000G      BIC      #UCLRN,LSW7(R1);Say TSXUCL program no longer running
41 006456 042737 000000C 000000G      BIC      #PASLIN!SCHAIN,@#JSWLOC ;Clear command-passed flag
42 006464 052737 000000G 000000G      BIS      #LCBIT,@#JSWLOC ;Enable lower-case input
43 006472 105267 175645      INCB     NOUCL ;Set flag saying not to call TSXUCL again
44 006476 000167 001762      JMP      IDNCMD ;Go process the command
45
46     ; If we have pending commands in the command file buffer, compress
47     ; them to make room for new commands.
48     ; If input is coming from a real command file we do not need to
49     ; compress since we will reread buffer when we hit end of new commands.
50
51 006502 032761 000000G 000000G 15$:      BIT      #CFOPN,LSW4(R1);IS A COMMAND FILE OPEN?
52 006510 001403      BEQ      11$      ;BR IF NOT
53 006512 012705 001000G      MOV      #CFBUF+512,R5 ;SAY ENTIRE COMMAND FILE BUFFER IS FREE
54 006516 000403      BR      12$
55 006520 004767 000000G 11$:      CALL     CFSQEZ ;COMPRESS INFO IN CURRENT COMMAND FILE BUFFER
56 006524 010005      MOV      R0,R5 ;SAVE ADDRESS OF END OF FREE SPACE IN BUFFER
57 006526 004767 000000G 12$:      CALL     PUSHCF ;OPEN A NEW COMMAND FILE (PUSH CURRENT ONE)

```

```

58      ;
59      ; Move command line from chain area to command file buffer
60      ;
61 006532 012703 0000100      MOV      #CINDAT+10,R3      ; POINT TO CELL WITH CHAR COUNT
62 006536 012302      MOV      (R3)+,R2      ; GET COUNT OF # CHARS IN COMMAND
63 006540 020227 000270      CMP      R2,#<1000-510>      ; CHECK LENGTH OF COMMAND STRING
64 006544 003407      BLE      6$      ; BR IF COMMAND IS SMALL ENOUGH
65 006546 042737 0000000 0000000 10$: BIC      #PASLIN!$CHAIN,@#JSWLOC ; CLEAR FLAGS BEFORE ABORT
66 006554      FABORT      #TOOLNG      ; LINE TOO LONG ERROR MESSAGE
67 006564 012704 0000000      6$: MOV      #CFBUF,R4      ; POINT TO START OF COMMAND FILE BUFFER
68 006570 112324      2$: MOV      (R3)+,(R4)+      ; MOVE COMMAND TO BUFFER
69 006572 001005      BNE      1$      ; BR IF NORMAL CHARACTER
70      ; Put cr-lf in place of asciz nulls
71 006574 112764 000015 177777      MOV      #CR,-1(R4)
72 006602 112724 000012      MOV      #LF,(R4)+
73 006606 077210      1$: SOB      R2,2$      ; MOVE ALL CHARACTERS
74 006610 020405      CMP      R4,R5      ; DID WE OVERFLOW BUFFER SPACE?
75 006612 101355      BHI      10$      ; BR IF YES -- COMMAND TOO LONG
76      ;
77      ; Null fill remainder of buffer
78      ;
79 006614 020405      14$: CMP      R4,R5      ; HAVE WE REACHED THE END OF THE BUFFER?
80 006616 103002      BHI      13$      ; BR IF YES
81 006620 105024      CLRB      (R4)+      ; NULL FILL REST OF BUFFER
82 006622 000774      BR      14$
83      ;
84      ; If special chain exit is being used, or if command is coming
85      ; from CCL or IND, then don't list the commands.
86      ;
87 006624 032761 0000000 0000000 13$: BIT      #CCLRN!$INDRN,LSW5(R1) ; IS CCL OR IND RUNNING?
88 006632 001004      BNE      8$      ; BR IF YES
89 006634 032737 0000000 0000000      BIT      #CHAIN,@#JSWLOC ; IS THIS A SPECIAL CHAIN EXIT?
90 006642 001403      BEQ      9$      ; BR IF NOT
91 006644 052761 0000000 0000000 8$: BIS      #QUIET,LSW4(R1) ; SET FLAG TO SUPPRESS LISTING COMMAND LINES
92 006652 032761 0000000 0000000 9$: BIT      #CCLRN,LSW5(R1) ; IS THIS AN EXPANDED CCL COMMAND?
93 006660 001403      BEQ      4$      ; BR IF NOT
94 006662 052761 0000000 0000000      BIS      #CFCCCL,LSW4(R1) ; REMEMBER THAT THIS IS A CCL COMMAND
95      ; Clear flags which say CCL or IND is running.
96 006670 042761 0000000 0000000 4$: BIC      #CCLRN!$INDRN,LSW5(R1) ; CCL AND IND ARE NO LONGER RUNNING
97 006676 042737 0000000 0000000      BIC      #PASLIN!$CHAIN,@#JSWLOC ; CLEAR COMMAND-PASSED FLAGS
98 006704 042761 0000000 0000000      BIC      #UCLRN!$UKMRN,LSW7(R1) ; SAY TSXUCL IS NO LONGER RUNNING
99      ;
100      ; See if we are exiting from a locked program
101      ;
102 006712 032761 0000000 0000000 CKLK: BIT      #PRGLK,LSW5(R1) ; IS A LOCKED PROGRAM EXITING?
103 006720 001402      BEQ      CKSF2      ; BR IF NOT
104 006722 000167 0000000      JMP      CMDOFF      ; EXIT FROM LOCKED PROGRAM==>LOGOFF
105      ;
106      ; See if we need to start execution of a secondary start-up command file
107      ; (The secondary start-up command file runs without privilege after the
108      ; initial start-up command file finishes).
109      ;
110 006726 105767 0000000      CKSF2: TSTB      SUCF2      ; Is there a pending secondary command file?
111 006732 001417      BEQ      NEWCMD      ; Br if not
112 006734 005767 0000000      TST      CFPNT      ; Are we currently in another command file?
113 006740 001014      BNE      NEWCMD      ; Br if yes -- Wait for it to finish
114 006742 042761 0000000 0000000      BIC      #SUCF,LSW9(R1) ; Say we are finished with 1st startup file

```

Entry to KMON

115	006750	042761	0000000	0000000	BIC	##NDIN,LSW3(R1)	; Allow input to be accepted for line
116	006756	012702	0000000		MOV	#SUCF2,R2	; Point to name of secondary command file
117	006762	004767	004732		CALL	SETSU	; Set up command file
118	006766	105067	0000000		CLRB	SUCF2	; Say secondary file no longer pending

Get keyboard command

```

1          .SBTTL  Get keyboard command
2
3          ; Print CR-LF to get to left margin if we are not already there.
4
5 006772 052737 0000000 0000000 NEWCMD: BIS      #LCBIT,@#JSWLOC ; ENABLE LOWER-CASE INPUT
6 007000 116701 0000000          MOV      CORUSR,R1      ; GET JOB INDEX NUMBER
7 007004 105761 0000000          TST      LCCL(R1)        ; ARE WE ALREADY AT LEFT MARGIN?
8 007010 001403          BEQ      RDCMD      ; BR IF YES -- NO CR-LF NEEDED
9 007012          .PRINT  #CRLF      ; PRINT CR-LF
10
11          ; See if IND is in control and we need to call it to get the next command
12
13 007020 116701 0000000 RDCMD:  MOV      CORUSR,R1      ; GET JOB INDEX NUMBER
14 007024 012702 0000000          MOV      #INDSTA,R2      ; GET POINTER TO IND STATUS BYTE
15 007030 132712 0000000          BIT      #IN$ACT,@R2      ; IS IND ACTIVE?
16 007034 001406          BEQ      2$          ; BR IF NOT
17 007036 142712 0000000          BICB     #IN$ACT!IN$CMD,@R2 ; CLEAR STATUS FLAGS FOR IND
18 007042 152712 0000000          BISB     #IN$CNT,@R2      ; SAY WE ARE CONTINUING EXECUTION OF IND
19 007046 000167 002476          JMP      INDRUN      ; GO RUN IND PROGRAM
20
21          ; If user-written command interface program is active, call it to
22          ; get the next command.
23
24 007052 032761 0000000 0000000 2$:  BIT      #$UXMON,LSW7(R1); Are we to use user-written command program?
25 007060 001405          BEQ      3$          ; Br if not
26 007062 005767 0000000          TST      CFPNT          ; Are we getting commands from a command file?
27 007066 001002          BNE      3$          ; Br if yes -- Don't call user program till end
28 007070 000167 002162          JMP      CALUKM          ; Enter user command processor program
29
30          ; Read next command line
31
32 007074          3$:  .GTLIN  #INBUF,#KMPRMT ; PROMPT FOR AND ACCEPT COMMAND LINE

```

Get keyboard command

```

1      ;
2      ; Input line is now in INBUF in asciz form.
3      ; Move line to CMDBUF while looking for start of comments or
4      ; indirect file reference.
5      ;
6 007114 012702 001156' PRSCMD: MOV      #CMDBUF,R2      ; MOVE FINISHED COMMAND HERE
7 007120 005067 171320      CLR      INDRFL      ; SAY NO INDIRECT FILE YET
8 007124 105067 171316      CLRB     QUOTFL      ; Say we are not in quoted string
9 007130 105067 175204      CLRB     DOLRAT      ; SAY HAVE NOT SEEN "$@"
10 007134 105067 175201     CLRB     DOTAT      ; SAY HAVE NOT SEEN "#@"
11 007140 105067 175176     CLRB     COLEQL     ; SAY HAVE NOT SEEN ":@"
12 007144 105067 175173     CLRB     NOUCL     ; SAY WE MAY CALL UCL FOR THIS COMMAND
13 007150 012704 001402' SCNCMD: MOV      #INBUF,R4      ; SCAN FROM HERE
14 007154 112400 8$:      MOVB     (R4)+,R0      ; GET NEXT CHAR FROM INPUT LINE
15 007156 001546      BEQ      4$      ; BR IF END OF LINE HIT
16 007160 120027 000040     CMPB     R0,#'      ; SKIP OVER LEADING SPACES
17 007164 001773      BEQ      8$
18 007166 120027 000011     CMPB     R0,#TAB      ; SKIP LEADING TABS
19 007172 001770      BEQ      8$
20 007174 120027 000014     CMPB     R0,#FF      ; SKIP LEADING FORM FEEDS
21 007200 001765      BEQ      8$
22 007202 000402      BR      9$      ; BEGIN SCANNING REAL COMMAND
23      ;
24      ; Get next character from input line
25      ;
26 007204 112400 6$:      MOVB     (R4)+,R0      ; GET NEXT CHAR FROM INPUT LINE
27 007206 001530      BEQ      11$      ; BR IF END OF LINE HIT
28      ;
29      ; See if we are in a quoted string
30      ;
31 007210 120067 171232 9$:      CMPB     R0,QUOTFL      ; Is this the terminating quote mark?
32 007214 001003      BNE      17$      ; Br if not
33 007216 105067 171224      CLRB     QUOTFL      ; Say not within quoted field now
34 007222 000511      BR      5$      ; Go store terminating quote character
35 007224 105767 171216 17$:     TSTB     QUOTFL      ; Are we inside a quoted string now?
36 007230 001106      BNE      5$      ; Br if yes -- Go store char without checking
37 007232 120027 000047     CMPB     R0,#47      ; Apostrophe character?
38 007236 001403      BEQ      19$      ; Br if yes
39 007240 120027 000042     CMPB     R0,#42      ; Quote character?
40 007244 001003      BNE      18$      ; Br if not
41 007246 110067 171174 19$:     MOVB     R0,QUOTFL      ; Remember we are inside a quoted string
42 007252 000475      BR      5$      ; Go store character without further checking
43      ;
44      ; Check for start of comments
45      ;
46 007254 120027 000041 18$:     CMPB     R0,#'!      ; START OF COMMENT FIELD?
47 007260 001503      BEQ      11$      ; BR IF YES
48      ;
49      ; Check for :=
50      ;
51 007262 105767 175054      TSTB     COLEQL      ; Have we already seen :=?
52 007266 001067      BNE      5$      ; Br if yes -- Ignore @'s after :=
53 007270 120027 000072     CMPB     R0,#':      ; Start of := sequence?
54 007274 001010      BNE      12$      ; Br if not
55 007276 110022      MOVB     R0,(R2)+      ; Store into result string
56 007300 112400      MOVB     (R4)+,R0      ; Get character following colon
57 007302 001472      BEQ      11$      ; Br if end of line hit

```

Get keyboard command

```

58 007304 120027 000075          CMPB    R0,#'='      ;Is this := ?
59 007310 001002          BNE      12$          ;Br if not
60 007312 105267 175024          INCB    COLEQL      ;Remember := seen within command string
61                                ;
62                                ; Check for $@
63                                ;
64 007316 120027 000044      12$:    CMPB    R0,#'$      ;COULD THIS BE START OF "$@"?
65 007322 001014          BNE      10$          ;BR IF NOT
66 007324 005767 171114          TST     INDRFL      ;HAVE WE ALREADY SEEN @?
67 007330 001046          BNE      5$          ;BR IF YES
68 007332 121427 000100          CMPB    (R4),#'@      ;IS THIS "$@"?
69 007336 001043          BNE      5$          ;BR IF NOT
70 007340 010267 171100          MOV     R2,INDRFL    ;REMEMBER INDIRECT FILE NAME LOCATION
71 007344 105267 174770          INCB    DOLRAT      ;REMEMBER PREFIX WAS "$@"
72 007350 005204          INC      R4              ;SKIP PAST "$"
73 007352 000714          BR       6$          ;START GETTING FILE NAME
74                                ;
75                                ; Check for #@
76                                ;
77 007354 120027 000043      10$:    CMPB    R0,#'#      ;COULD THIS BE START OF "#@"?
78 007360 001014          BNE      3$          ;BR IF NOT
79 007362 005767 171056          TST     INDRFL      ;HAVE WE ALREADY SEEN @?
80 007366 001027          BNE      5$          ;BR IF YES
81 007370 121427 000100          CMPB    (R4),#'@      ;IS THIS "#@"?
82 007374 001024          BNE      5$          ;BR IF NOT
83 007376 010267 171042          MOV     R2,INDRFL    ;REMEMBER INDIRECT FILE NAME LOCATION
84 007402 105267 174733          INCB    DOTAT       ;REMEMBER PREFIX WAS "#@"
85 007406 005204          INC      R4              ;POINT PAST "."
86 007410 000675          BR       6$          ;START GETTING FILE NAME
87                                ;
88                                ; Check for @ and @@
89                                ;
90 007412 120027 000100      3$:     CMPB    R0,#'@      ;START OF INDIRECT FILE REFERENCE?
91 007416 001013          BNE      5$          ;BR IF NOT
92 007420 121427 000100          CMPB    (R4),#'@      ;Is this "@@"?
93 007424 001002          BNE      20$         ;Br if not
94 007426 005204          INC      R4              ;Skip past second at-sign
95 007430 000406          BR       5$          ;Translate "@@" to "@" and pass with command
96 007432 005767 171006      20$:    TST     INDRFL      ;ALREADY SEEN @ BEFORE?
97 007436 001003          BNE      5$          ;IF YES THEN IGNORE THIS ONE FOR NOW
98 007440 010267 171000          MOV     R2,INDRFL    ;SAVE POINTER TO START OF FILE NAME
99 007444 000657          BR       6$          ;DON'T BOTHER STORING @ IN CMDBUF
100                                ;
101                                ; Move character to buffer
102                                ;
103 007446 020227 001362'      5$:     CMP     R2,#CBFEND    ;MAKE SURE WE DON'T OVERFLOW BUFFER
104 007452 103404          BLO      1$          ;BR IF OK
105 007454          FABORT    #ILLCMD      ;COMMAND TOO LONG
106 007464 110022      1$:     MOVB     R0,(R2)+      ;MOVE CHAR TO CMDBUF
107 007466 000646          BR       6$          ;GO GET REST OF LINE
108                                ;
109                                ; Reached end of line -- strip off any trailing spaces and tabs
110                                ;
111 007470 010267 171036      11$:    MOV     R2,CMDEND    ;SAVE POINTER PAST END OF COMMAND STRING
112 007474 020227 001156'      4$:     CMP     R2,#CMDBUF    ;HAVE WE GONE PAST START OF BUFFER?
113 007500 001002          BNE      16$         ;BR IF NOT
114 007502 000167 177312          JMP     RDCMD      ;YES -- THIS IS A NULL COMMAND

```

Get keyboard command

```
115 007506 124227 000040      16$:  CMPB  -(R2),#'      ; IS NEXT CHARACTER A SPACE?
116 007512 001770              BEQ    4$          ; LOOP BACKWARD OVER SPACES
117 007514 121227 000011      CMPB  (R2),#TAB    ; IS THIS A TAB?
118 007520 001765              BEQ    4$          ; LOOP IF YES
119 007522 005202              INC     R2          ; POINT BEYOND LAST NON-BLANK CHARACTER
120
121      ; See if command line is continued
122
123 007524 126227 177777 000055      CMPB  -1(R2),#'-    ; IS LINE CONTINUED?
124 007532 001013              BNE     GOTCML      ; BR IF NOT
125      ; Line is continued -- get more
126 007534 005302              DEC     R2          ; POINT BACK TO -
127 007536              .GT LIN #INBUF,#KMPRMT    ; READ COMMAND CONTINUATION LINE
128 007556 000167 177366              JMP     SCNCMD      ; CONTINUE SCANNING LINE
```

Get keyboard command

```

1
2      ; End of command found.
3      ; See if this command contains := which indicates this is a user
4      ; request to define a new command keyword.
5
6 007562 105012      GOTCML: CLRB      (R2)      ; store null at end of command
7 007564 026727 170654 001156'      CMP      INDRFL, #CMDBUF ; Was "@" first character of command?
8 007572 001443      BEQ      INDCMD      ; Br if yes -- Don't call UCL for this
9 007574 105767 174542      TSTB     COLEQL      ; Was := seen within command line?
10 007600 001411      BEQ      13$      ; Br if not
11 007602 005767 000000G      TST      UCLBLK      ; Are we supporting user-defined commands?
12 007606 001004      BNE      1$      ; Br if yes
13 007610      FABORT #EM$NUC      ; No user-defined commands allowed
14 007620 000167 001306      1$:      JMP      CALUCL      ; Go call UCL to process it
15
16      ; See if command line contains an indirect command file reference
17
18 007624 016703 170614      13$:      MOV      INDRFL, R3      ; Did we have indirect file reference?
19 007630 001024      BNE      INDCMD      ; Br if yes
20
21      ; If we are to call TSXUCL before normal processing, determine if we
22      ; should do it for this command.
23
24 007632 126727 171320 000137      CMPB     CMDBUF, #'_'      ; Was "_" specified as first char of command?
25 007640 001411      BEQ      15$      ; Br if yes -- Don't call TSXUCL for _command
26 007642 032761 000000G 000000G      BIT      #$UCLCF, LSW7(R1) ; Should we call TSXUCL before normal commands?
27 007650 001412      BEQ      16$      ; Br if not
28 007652 005767 000000G      TST      UCLBLK      ; Are we allowing user-defined commands?
29 007656 001407      BEQ      16$      ; Br if not
30 007660 000167 001246      JMP      CALUCL      ; Call TSXUCL to try to process this command
31 007664 112767 000040 171264      15$:      MOVB     #' ', CMDBUF      ; Replace leading underscore with space
32 007672 105267 174445      INCB     NOUCL      ; Remember not to call TSXUCL for this command
33 007676 000167 000562      16$:      JMP      IDNCMD      ; Go try to identify the command

```

Get keyboard command

```

1      ;
2      ; Command line contains indirect file reference.
3      ;
4      ; Accrue indirect file name and try to open the file.
5      ;
6 007702 116701 0000000 INDCMD: MOVB   CORUSR,R1      ;GET JOB INDEX #
7 007706 016703 170532      MOV     INDRFL,R3      ;GET POINTER TO COMMAND FILE NAME
8      ;
9      ; Accrue the command file spec
10     ;
11 007712 010304      MOV     R3,R4      ;Save pointer to start of command file name
12 007714 012705 000250'  MOV     #DKCOM,R5      ;SET DEFAULT DEV AND EXT
13 007720 004767 0000000  CALL    ACRFN      ;ACCRUE THE FILE NAME
14 007724 103002      BCC     4$      ;BR IF GOT NAME OK
15 007726 000167 177066  JMP     RDCMD      ;ERROR WHILE GETTING FILE NAME
16     ;
17     ; See if we should let IND process the indirect command file
18     ;
19 007732 026727 170506 001156' 4$:  CMP     INDRFL,#CMDBUF ;Was "@" first character of command?
20 007740 001031      BNE     1$      ;Br if not
21 007742 105767 174373  TSTB    DOTAT      ;WAS PREFIX "#@"?
22 007746 001407      BEQ     2$      ;BR IF NOT
23 007750 005767 0000000  TST     INDSAV      ;IS IND AVAILABLE ON SYSTEM?
24 007754 001013      BNE     3$      ;BR IF YES
25 007756      FABORT  #NOIND      ;IND IS NOT AVAILABLE
26 007766 032761 0000000 0000000 2$:  BIT     ##INDDF,LSW5(R1); IS IND TO BE CALLED BY DEFAULT?
27 007774 001413      BEQ     1$      ;BR IF NOT
28 007776 105767 174336  TSTB    DOLRAT      ;WAS COMMAND PREFIX "#@"?
29 010002 001010      BNE     1$      ;BR IF YES -- DON'T USE IND
30 010004 132767 0000000 0000000 3$:  BITB    #IN$ACT,INDSTA ;IS IND ACTIVE NOW?
31 010012 001004      BNE     1$      ;BR IF YES
32 010014 010403      MOV     R4,R3      ;Get back pointer to start of file spec
33 010016 005005      CLR     R5      ;NO DEFAULT DEVICE
34 010020 000167 001372  JMP     INDINI      ;GO ENTER IND
35     ;
36     ; Process this indirect command file directly rather than calling IND
37     ;
38 010024 016700 170420 1$:  MOV     FILNAM,R0      ;Get the device name
39 010030 004767 0000000  CALL    CHKTTD      ;Is the device TT?
40 010034 103441      BCS     NOICMD      ;Br if yes -- Error
41 010036 004767 0000000  CALL    PUSHCF      ;PUSH CURRENT @FILE ON STACK
42 010042 112767 000001 0000000  MOVB    #1,SERFLG      ;DO .SERR
43 010050      .LOOKUP #XAREA,#CFCHAN,#FILNAM
44 010070 112767 000000 0000000  MOVB    #0,SERFLG      ;DO .HERR, DON'T CLEAR CARRY FLAG
45 010076 103024      BCC     CFOPEN      ;BR IF OPEN OK
46 010100 116701 0000000  MOVB    CORUSR,R1      ;GET JOB INDEX NUMBER
47 010104 032761 0000000 0000000  BIT     ##SUCF,LSW9(R1); ARE WE WITHIN A START-UP COMMAND FILE?
48 010112 001410      BEQ     5$      ;BR IF NOT
49 010114      FERR    #NOCF      ;PRINT ERROR MESSAGE
50 010130 000167 0000000  JMP     CMDOFF      ;LOG THE JOB OFF
51 010134 004767 0000000 5$:  CALL    POPCF      ;POP PUSHED COMMAND FILE STATUS
52 010140  NOICMD:  FABORT  #NOCF      ;COULDN'T OPEN @FILE
53     ;
54     ; We have successfully opened the indirect command file.
55     ;
56 010150 116701 0000000  CFOPEN:  MOVB    CORUSR,R1
57 010154 052761 0000000 0000000  BIS     ##CFOPN,LSW4(R1); SAY CFCHAN IS OPEN

```

Get keyboard command

```

58 ;
59 ; See if command file was installed with any privileges.
60 ;
61 010162 004767 001726 CALL INSCF ; See if command file was installed
62 ;
63 ; READ IN 1ST BLOCK FROM INDIRECT FILE
64 ;
65 010166 . READW #XAREA, #CFCHAN, #CFBUF, #256, #0
66 010224 103003 BCC CFPRM ; BR IF READ OK
67 ; ERROR OCCURED ON COMMAND FILE READ.
68 ; THIS MUST MEAN THAT WE HAVE AN EMPTY COMMAND FILE.
69 ; SET BUFFER POINTER TO CAUSE US TO IGNORE THIS BUFFER FULL.
70 010226 012767 0000000 0000000 MOV #CFEND, CFPNT ; SAY BUFFER IS EMPTY
71 ;
72 ; SEE IF INDIRECT FILE HAS PARAMETERS.
73 ;
74 ; R3 NOW POINTS PAST END OF COMMAND FILE NAME.
75 ; SCAN ACROSS COMMAND LOOKING FOR 1ST PARAMETER
CFPRM: TSTB (R3) ; HIT END OF COMMAND?
76 010234 105713 BEQ RDREST ; BR IF END HIT
77 010236 001437 CMPB (R3)+, #' ; SKIP OVER LEADING SPACES
78 010240 122327 000040 BEQ CFPRM
79 010244 001773 ; SEE IF WE SHOULD USE SPACE OR \ AS PARAMETER DELIMITER
80 ;
81 010246 012701 000040 MOV #' , R1 ; ASSUME SPACE IS DELIMITER
82 010252 124327 000134 CMPB -(R3), #' \ ; DOES HE WANT TO USE \?
83 010256 001001 BNE B$ ; BR IF NOT -- USE SPACE
84 010260 112301 MOVB (R3)+, R1 ; USE \ AS PARAM SEPARATOR
85 ; THERE ARE SOME PARAMETERS -- ACCRUE THEM
86 010262 012704 0000000 B$: MOV #PRMPNT, R4 ; POINT TO PARAM POINTER CELLS
87 010266 012705 0000000 MOV #PRMBUF, R5 ; POINT TO PARAMETER STRING BUFFER
88 010272 020427 0000000 7$: CMP R4, #LSTPRM ; TOO MANY PARAMETERS?
89 010276 103062 BHS TOMPRM ; BR IF TOO MANY
90 010300 010524 MOV R5, (R4)+ ; SET PARAMETER STRING POINTER
91 010302 121301 6$: CMPB (R3), R1 ; REACHED PARAM DELIMITER YET?
92 010304 001407 BEQ 4$ ; BR IF YES
93 010306 105713 TSTB (R3) ; HIT END OF COMMAND?
94 010310 001405 BEQ 4$ ; BR IF YES
95 010312 020527 0000000 CMP R5, #PRMEND ; HIT END OF PARAM STRING BUFFER?
96 010316 103056 BHS PTL ; BR IF PARAM STRING TOO LONG
97 010320 112325 MOVB (R3)+, (R5)+ ; MOVE PARAMETER TO BUFFER
98 010322 000767 BR 6$
99 ; HIT END OF PARAMETER -- STORE NULL IN STRING TO MARK END.
100 010324 105025 4$: CLRB (R5)+ ; FLAG END OF THIS PARAMETER STRING
101 010326 105723 TSTB (R3)+ ; MORE TO ACCRUE?
102 010330 001360 BNE 7$ ; BR IF YES
103 010332 010567 0000000 MOV R5, PBFEND ; SAVE POINTER TO END OF PARAM STRING
104 ;
105 ; THE COMMAND FILE HAS BEEN OPENED AND ITS PARAMETERS HAVE
106 ; BEEN ACCRUED AND STORED AWAY.
107 ; IF THIS COMMAND FILE IS PART OF ANOTHER COMMAND (PART OF SAME LINE)
108 ; THEN GO AND READ THE REST OF THE COMMAND.
109 ; ELSE GO READ NEXT COMMAND LINE WHICH WILL COME FROM THE INDIRECT
110 ; COMMAND FILE WE JUST OPENED.
111 ;
112 010336 016702 170102 RDREST: MOV INDRFL, R2 ; GET POINTER TO START OF @SPECIFICATION
113 010342 001436 BEQ CFJMP ; BR IF IMPLICIT @FILE EXECUTION
114 010344 020227 001156' CMP R2, #CMDBUF ; IS @FILE 1ST THING IN COMMAND?

```

Get keyboard command

```

115 010350 001433          BEQ      CFJMP      ;BR IF YES
116
117          ; THIS INDIRECT FILE REFERENCE IS PART OF ANOTHER COMMAND SO
118          ; SUPPRESS THE LISTING OF THE 1ST LINE OF @FILE AS WE READ IT.
119
120 010352 116701 00000000      MOVB     CORUSR,R1      ;GET USER INDEX #
121 010356 016146 00000000      MOV      LSW4(R1),-(SP) ;SAVE COMMAND FILE STATUS FLAGS
122 010362 012705 00000000      MOV      ##QUIET,R5      ;GET QUIET FLAG INTO REGISTER
123 010366 050561 00000000      BIS      R5,LSW4(R1)      ;TURN QUIET ON
124 010372          .GTLIN     #INBUF      ;READ LINE FROM @FILE
125 010410 030526          BIT      R5,(SP)+      ;NEED TO SET OR RESET QUIET MODE?
126 010412 001002          BNE      10$      ;BR IF WANT TO LEAVE IT SET
127 010414 040561 00000000      BIC      R5,LSW4(R1)      ;RESET QUIET MODE
128 010420 005067 170020      10$:      CLR      INDRFL      ;SAY @FILE REFERENCE HAS BEEN RESOLVED
129 010424 105067 173710      CLR      DOLRAT      ;SAY HAVEN'T SEEN "$@"
130 010430 105067 173705      CLR      DOTAT      ;SAY HAVEN'T SEEN "#@"
131 010434 000167 176510      JMP      SCNCMD      ;GO GET REST OF COMMAND
132 010440 000167 176354      CFJMP:    JMP      RDCMD      ;GO READ NEXT COMMAND
133          ; ERROR -- TOO MANY PARAMETERS
134 010444      TOMPRM: FABORT #NUMPRM
135          ; ERROR -- PARAMETER STRING TOO LONG
136 010454      PTL:      FABORT #STRLEN

```

Identify command

```

1          .SBTTL  Identify command
2          -----
3          ; AT THIS POINT A COMMAND LINE HAS BEEN ACCEPTED, CONTINUATION
4          ; LINES READ, COMMENTS STRIPPED AND INDIRECT FILE REFERENCES
5          ; RESOLVED.  THE RESULTING COMMAND IS STORED IN CMDBUF IN
6          ; ASCII FORM WITH R2 POINTING TO THE END OF THE COMMAND.
7          ;
8          ; TRY TO IDENTIFY IT AS A SYSTEM COMMAND.
9          ;
10         IDNCMD: MOV      #CMDHD,R4          ; POINT TO TABLE OF SYSTEM COMMANDS
11                MOV      #CMDBUF,R3         ; POINT TO OUR COMMAND
12                CALL      SEARCH            ; LOOK UP THE COMMAND KEYWORD
13                BCC        FNDCMD           ; BR IF IDENTIFIED THE COMMAND
14          ; ERROR DURING SEARCH.  SEE IF UNRECOGNIZED OR AMBIGUOUS.
15                TST       R4                ; AMBIGUOUS OR UNRECOGNIZED?
16                BEQ        NOCMD            ; BR IF UNRECOGNIZED
17                FABORT     #ABCMD           ; AMBIGUOUS COMMAND
18          ;
19          ; WE HAVE A VALID SYSTEM COMMAND.
20          ;
21          ; BRANCH OFF TO COMMAND PROCESSING ROUTINE.
22          ; AT THIS POINT THE FOLLOWING REGISTERS ARE SET UP:
23          ;   R1 = USER INDEX NUMBER
24          ;   R2 = ADDRESS OF END OF COMMAND STRING
25          ;   R3 = ADDRESS OF START OF COMMAND ARGUMENT FIELD.
26          ;
27         FNDCMD: MOVB     CORUSR,R1          ; GET USER INDEX #
28                JMP       @(R4)+           ; ENTER COMMAND PROCESSING ROUTINE

```

Identify command

```

1          ; -----
2          ; We could not identify the command as a standard system command
3          ; so we try to identify it as an implicit command in the following way:
4          ;
5          ; 1. See if it is a user-defined command (if SET UCL MIDDLE).
6          ; 2. See if there is a command file on "DK:" with command name.
7          ; 3. See if there is a command file on "SY:" with command name.
8          ; 4. See if there is a SAV file on "SY:" with command name.
9          ; 5. See if there is a SY:UCL program to process the command (if UCL LAST).
10         ;
11         ; See if we should call TSXUCL to process this command.
12         ;
13 010524 116701 000000G NOCMD:  MOV  CORUSR,R1      ;GET CURRENT JOB INDEX NUMBER
14 010530 032761 000000G 000000G  BIT    ##UCLCM,LSW7(R1);SET UCL MIDDLE?
15 010536 001410          BEQ    7$              ;BR IF NOT
16 010540 105767 173577    TSTB   NOUCL          ;DID UCL ALREADY REJECT THIS COMMAND?
17 010544 001005          BNE    7$              ;BR IF YES
18 010546 005767 000000G    TST    UCLBLK        ;ARE WE ALLOWING USER-DEFINED COMMANDS?
19 010552 001402          BEQ    7$              ;BR IF NOT
20 010554 000167 000352    JMP    CALUCL        ;SEND COMMAND TO TSXUCL
21         ;
22         ; See if there is a command file on DK device with command name.
23         ;
24 010560 012703 001156'  7$:   MOV    #CMDBUF,R3      ;POINT TO KEYWORD NAME
25 010564 012705 000250'    MOV    #DKCOM,R5        ;GET DEV AND EXT DEFAULTS
26 010570 004767 000000G    CALL   ACRFN          ;ACCRUE THE FILE NAME
27 010574 103513          BCS    6$              ;BR IF ERROR IN GETTING FILE NAME
28 010576 016700 167646    MOV    FILNAM,R0        ;Get the device name
29 010602 004767 000000G    CALL   CHKTDD        ;Is it TT?
30 010606 103506          BCS    6$              ;Error if yes
31 010610 004767 000000G    CALL   PUSHCF        ;PUSH CURRENT @FILE STATUS
32 010614 112767 000001 000000G  MOV  #1,SERFLG    ;DON'T ABORT ON ERRORS
33 010622          .LOOKUP #XAREA,#CFCHAN,#FILNAM
34 010642 112767 000000 000000G  MOV  #0,SERFLG    ;DO .HERR -- DON'T CLEAR CARRY FLAG
35 010650 103402          BCS    3$              ;BR IF NOT FOUND
36 010652 005005          CLR    R5              ;SAY NO DEFAULT DEVICE NAME
37 010654 000424          BR     4$              ;GO SEE IF WE SHOULD RUN IND
38         ;
39         ; See if there is a command file on SY device with command name.
40         ;
41 010656 016767 167406 167564 3$:  MOV    R5OSY,FILNAM ;CHANGE DEVICE NAME TO BE "SY"
42 010664 112767 000001 000000G  MOV  #1,SERFLG    ;DO .SERR
43 010672          .LOOKUP #XAREA,#CFCHAN,#FILNAM
44 010712 112767 000000 000000G  MOV  #0,SERFLG    ;DO .HERR, DON'T CLEAR CARRY FLAG
45 010720 103430          BCS    1$              ;BR IF COMMAND IS NOT A COMMAND FILE
46         ;
47         ; We located a command file on DK or SY.
48         ; See if we should call IND to execute it.
49         ;
50 010722 012705 004345'    MOV    #SYTXT,R5      ;GET DEFAULT DEVICE FOR IND FILE
51 010726 132767 000000G 000000G 4$:  BITB   #IN$ACT,CFIND ;IS IND ACTIVE NOW?
52 010734 001015          BNE    2$              ;BR IF ALREADY ACTIVE
53 010736 032761 000000G 000000G  BIT    ##INDDF,LSW5(R1);IS IND WANTED?
54 010744 001411          BEQ    2$              ;BR IF NOT
55 010746          .PURGE #CFCHAN          ;PURGE CHANNEL WE OPENED TO FILE
56 010754 004767 000000G    CALL   POPCF        ;POP UP TO OLD FILE
57 010760 012703 001156'    MOV    #CMDBUF,R3      ;POINT TO START OF COMMAND LINE

```

Identify command

```

58 010764 000167 000426          JMP      INDINI          ; GO START UP IND
59                                ;
60                                ; This is an implicit command file execution.
61                                ;
62 010770 052761 0000000 0000000 2$: BIS      ##QUIET,LSW4(R1); ALWAYS SET QUIET IF IMPLICIT RUN
63 010776 000167 177146          JMP      CFOPEN          ; CONTINUE PROCESSING STARTUP OF @FILE
64                                ;
65                                ; This is not an implicit @file call.
66                                ;
67 011002 004767 0000000          1$: CALL     POPCF          ; REOPEN PREVIOUS @FILE
68                                ;
69                                ; See if there is a program on "SY" with command name.
70                                ;
71 011006 012703 001156'          MOV      #CMDBUF,R3        ; POINT TO COMMAND KEYWORD
72 011012 012705 000254'          MOV      #SYSAB,R5         ; SET DEFAULT DEV AND EXT
73 011016 004767 0000000          CALL     ACRFN            ; ACCRUE FILE NAME
74 011022 103002                  BCC      5$               ; BR IF GOT FILE NAME OK
75 011024 000167 175770          6$: JMP      RDCMD          ; ERROR IN GETTING FILE NAME
76 011030 016700 167414          5$: MOV      FILNAM,R0       ; Get the device name
77 011034 004767 0000000          CALL     CHKTTD           ; Is the device TT?
78 011040 103421                  BCS      BADCMD           ; Error if yes
79 011042 112767 000001 0000000  MOVB     #1,SERFLG        ; DO .SERR
80 011050                  . LOOKUP #XAREA,#RUNCHN,#FILNAM
81 011070 112767 000000 0000000  MOVB     #0,SERFLG        ; DO .HERR, DON'T CLEAR CARRY FLAG
82 011076 103406                  BCS      TRYUCL           ; COULDN'T FIND PROGRAM
83 011100 000167 0000000          JMP      DORUN           ; START RUNNING THE PROGRAM
84                                ;
85 011104          BADCMD: FABORT #ILLCMD          ; INVALID COMMAND

```

```

1
2 ; -----
3 ; See if there is a SY:TSXUCL program to process the command
4 011114 032761 0000000 0000000 TRYUCL: BIT    #UCLCL,LSW7(R1); SHOULD WE CALL TSXUCL LAST?
5 011122 001451                      BEQ      URERR      ;BR IF NOT
6 011124 105767 173213                TSTB    NOUCL      ;DID UCL ALREADY REJECT THIS COMMAND?
7 011130 001046                      BNE      URERR      ;BR IF YES
8
9 ; Call the TSXUCL program to process this command
10
11 011132 105767 0000000 CALUCL: TSTB    VU$CL      ; IS THERE A UCL PROGRAM?
12 011136 001443                      BEQ      URERR      ;BR IF NOT
13
14 ; UCL option is genned in. See if we can find UCL program.
15
16 011140                      . LOOKUP #XAREA, #RUNCHN, #UCLNAM ; TRY TO FIND SY: UCL. SAV
17 011160 103426                      BCS      9$          ;BR IF CAN'T FIND UCL PROGRAM
18
19 ; We found the UCL program.
20 ; Pass command line to it in chain area.
21
22 011162 012703 0000120                      MOV     #CINDAT+12,R3 ; POINT TO CHAIN AREA
23 011166 012704 001156'                      MOV     #CMDBUF,R4    ; POINT TO COMMAND LINE BUFFER
24 011172 112423                1$: MOV     (R4)+,(R3)+    ; MOVE COMMAND TO CHAIN DATA AREA
25 011174 001376                      BNE      1$          ; LOOP TILL ASCIZ NULL MOVED
26 011176 162703 0000130                      SUB     #CINDAT+13,R3 ; COMPUTE LENGTH OF COMMAND LINE
27 011202 010367 0000100                      MOV     R3,CINDAT+10 ; AND STORE LENGTH INTO 510
28 011206 012704 0000000                      MOV     #UCLNAM,R4    ; POINT TO NAME OF UCL PROGRAM
29 011212 105267 0000000                      INCB    CINFLG    ; SIMULATE .CHAIN
30 011216 116701 0000000                      MOV     CORUSR,R1    ; GET CURRENT JOB INDEX NUMBER
31 011222 052761 0000000 0000000                      BIS     #UCLRN,LSW7(R1); SAY THAT UCL PROGRAM IS RUNNING
32 011230 005000                      CLR      R0          ; NO RUN OPTION FLAGS FOR PLOAD
33 011232 000167 0000000                      JMP      PLOAD      ; LOAD AND RUN UCL PROGRAM
34
35 ; Cannot find TSXUCL program
36
37 011236                9$: FABORT  #MISUCL      ; SAY UCL IS MISSING
38
39 ; Could not identify command
40
41 011246                URERR: FABORT  #URCMD      ; UNRECOGNIZABLE COMMAND

```

CALUKM -- Start user-written command processor

```

1          .SBTTL  CALUKM -- Start user-written command processor
2          ;-----
3          ; Call user-written command interface program.
4          ;
5 011256 116701 0000000 CALUKM: MOVB  CORUSR,R1      ;Get job index number
6 011262 012704 0000000      MOV   #UCISPC,R4      ;Get pointer to program name for PLOAD
7 011266      .LOOKUP #XAREA,#RUNCHN,R4;Try to lookup program
8 011304 103406      BCS   9$      ;Br if cannot find program
9          ;
10         ; We found the program. Enter it.
11         ;
12 011306 052761 0000000 0000000      BIS   ##UKMRN,LSW7(R1);Say user command processor is running
13 011314 005000      CLR   R0      ;No run option flags for PLOAD
14 011316 000167 0000000      JMP   PLOAD      ;Enter the program
15         ;
16         ; Cannot find the program.
17         ;
18 011322 042761 0000000 0000000 9$:  BIC   ##UKMRN,LSW7(R1);Don't try to use the program again
19 011330      FABORT  #EM#NUK      ;Program not there

```

CMDIND -- IND command

```

1          .SBTTL  CMDIND -- IND command
2          ;-----
3          ;  A command of the form "IND file" has been entered.
4          ;  Call the IND program to process the indirect command file.
5          ;
6 011340   CMDIND:
7          ;
8          ;  See if IND is available
9          ;
10 011340   005767   0000000   TST      INDSAV      ;Is IND available on system?
11 011344   001004           BNE      1$          ;Br if yes
12 011346           FABORT   #NOIND      ;IND is not available
13          ;
14          ;  Accrue the file name so that we can check later to see if the
15          ;  command file was installed with any special privileges.
16          ;
17 011356   004767   0000000   1$:      CALL     SKPSPC      ;Skip over any spaces
18 011362   105713           TSTB      (R3)          ;Any file specified?
19 011364   001411           BEQ       4$          ;Br if not
20 011366   010304           MOV       R3,R4          ;Save pointer to start of command file
21 011370   012705   000250'   MOV       #DKCOM,R5      ;Set default device and extension
22 011374   004767   0000000   CALL     ACRFN          ;Accrue the file name
23 011400   103002           BCC       2$          ;Br if got file name ok
24 011402   000167   175412   JMP       RDCMD          ;Error accruing file name
25 011406   010403   2$:      MOV       R4,R3          ;Restore pointer to start of file spec
26          ;
27          ;  Start execution of IND
28          ;
29 011410   005005   4$:      CLR       R5            ;No default device string
30 011412   000167   000000   JMP       INDINI          ;Startup IND

```

INDINI -- Start IND program

```

1          .SBTTL  INDINI -- Start IND program
2          ;-----
3          ; Call IND to process an indirect command file.
4          ;
5          ; Inputs:
6          ;   R3 = Pointer to asciz command line.
7          ;   R5 = Pointer to default device string (asciz)
8          ;   FILNAM = File spec for command file being started
9          ;
10 011416  INDINI:
11          ;
12          ; Error if IND is already running
13          ;
14 011416  132767  0000000 0000000  BITB    #IN$ACT,INDSTA ;Is IND running now?
15 011424  001404          BEQ      6$          ;Br if not
16 011426          FABORT  #INDACT          ;IND is already active
17          ;
18          ; Set flag that says IND is being started
19          ;
20 011436  116701  0000000 6$:      MOVB    CORUSR,R1          ;Get job index number
21 011442  052761  0000000 0000000  BIS      #$INDRN,LSW5(R1); Say IND is running
22          ;
23          ; See if command file was installed with any privileges
24          ;
25 011450  004767  000440          CALL    INSCF          ;See if command file was installed
26          ;
27          ; Build command line of the form "SY:IND file-name"
28          ;
29 011454  012702  001402'      MOV      #BLK0,R2          ;Point to area where we will build command
30          ;
31          ; Insert "SY:IND "
32          ;
33 011460  012704  004351'      MOV      #SYINTX,R4          ;Point to "SY:IND " text string
34 011464  112422  1$:      MOVB    (R4)+,(R2)+          ;Copy text string
35 011466  001376          BNE      1$          ;Loop till null hit
36 011470  005302          DEC      R2          ;Point back to null
37          ;
38          ; See if we need to insert default device name
39          ;
40 011472  005705          TST      R5          ;Do we have a default device name?
41 011474  001403          BEQ      2$          ;Br if not
42 011476  112522  3$:      MOVB    (R5)+,(R2)+          ;Move in default device name
43 011500  001376          BNE      3$
44 011502  005302          DEC      R2          ;Point back over terminating null
45          ;
46          ; Copy command text string
47          ;
48 011504  010304  2$:      MOV      R3,R4          ;Get pointer to command buffer
49 011506  112422  4$:      MOVB    (R4)+,(R2)+          ;Copy command text string
50 011510  001376          BNE      4$
51          ;
52          ; Now move string back to command buffer
53          ;
54 011512  012702  001402'      MOV      #BLK0,R2          ;Point to start of new string
55 011516  010304          MOV      R3,R4          ;Point to destination area
56 011520  112224  5$:      MOVB    (R2)+,(R4)+          ;Copy string
57 011522  001376          BNE      5$

```

```

58      ;
59      ; Now process like a R command
60      ;
61 011524 156767 0000000 0000000      BISB  UERSEV,INDERR ;Pass error sev level to IND
62 011532 142767 0000000 0000000      BICB  #IN$CMD,INDSTA ;Say no IND command pending
63 011540 012705 000254'      MOV   #SYSAV,R5 ;Point to default device (SY)
64 011544 000167 0000000      JMP    RUNNAM ;Enter RUN code
65      ;
66      ; Reopen channel to IND.SAV file
67      ;
68 011550      INDRUN: .REOPEN #XAREA,#RUNCHN,#INDSAV ;REOPEN CHANNEL 16 TO IND
69 011570 012704 000402'      MOV   #INDNAM,R4 ;POINT TO NAME OF IND
70 011574 105267 0000000      INCB  CINFLG ;SAY A .CHAIN IS IN PROGRESS
71 011600 156767 0000000 0000000      BISB  UERSEV,INDERR ;PASS ERROR SEV LEVEL TO IND
72 011606 116701 0000000      MOVB  CORUSR,R1 ;GET JOB INDEX NUMBER
73 011612 052761 0000000 0000000      BIS   #$INDRN,LSW5(R1);SAY IND IS RUNNING
74 011620 142767 0000000 0000000      BICB  #IN$CMD,INDSTA ;SAY NO IND COMMAND PENDING
75 011626 005000      CLR   R0 ;No run option flags for PLOAD
76 011630 000167 0000000      JMP    PLOAD ;GO START RUNNING IND

```

```

1          .SBTTL  Process CCL commands
2          ;-----
3          ; THIS ROUTINE HANDLES CCL COMMANDS SUCH AS COMPILE, COPY, ETC.
4          ; IT MOVES THE CCL COMMAND TO THE CHAIN DATA AREA THEN CALLS
5          ; THE PROGRAM "SY:CCL.SAV" WHICH TRANSLATES THE CCL COMMAND
6          ; INTO A SERIES OF SIMPLE COMMANDS WHICH ARE RETURN TO KMON
7          ; IN THE CHAIN AREA WHEN CCL.SAV EXITS.
8          ;
9          ; MOVE CCL COMMAND TO CHAIN AREA.
10         ;
11 011634 012703 0000120 CMDCCCL: MOV      #CINDAT+12,R3  ;POINT TO CHAIN DATA AREA
12 011640 012704 001156'      MOV      #CMDBUF,R4    ;POINT TO COMMAND BUFFER
13 011644 112400      1$:  MOV      (R4)+,R0          ;GET NEXT CHAR FROM COMMAND LINE
14 011646 120027 000011      CMPB     R0,#TAB        ;IS CHARACTER TAB?
15 011652 001403      BEQ        10$                ;BR IF YES
16 011654 120027 000014      CMPB     R0,#FF         ;FORM-FEED?
17 011660 001002      BNE        11$
18 011662 112700 000040      10$:  MOV      #' ,R0      ;TRANSLATE TAB AND FF TO SPACE
19 011666 120027 000141      11$:  CMPB     R0,#141     ;SEE IF IT IS A LOWER CASE LETTER
20 011672 103405      BLO        6$                ;BR IF NOT
21 011674 120027 000172      CMPB     R0,#172        ;LOWER CASE Z
22 011700 101002      BHI        6$                ;BR IF DELIMITER
23 011702 162700 000040      SUB       #40,R0        ;CONVERT LOWER-CASE UP UPPER-CASE
24 011706 110023      6$:  MOV      R0,(R3)+         ;MOVE CHAR TO CHAIN AREA
25 011710 001355      BNE        1$                ;LOOP TILL ASCIZ NULL MOVED
26          ; NOW SET SOME FLAGS IN LOCATION 510
27 011712 005003      CLR        R3                ;FORM FLAG WORD IN R3
28 011714 016104 00000000    MOV      LSW5(R1),R4    ;PICK UP STATUS WORD
29 011720 032704 00000000    BIT      #$TECO!$KED,R4 ;IS DEFAULT EDITOR TECO OR KED?
30 011724 001417      BEQ        2$                ;BR IF NOT (MUST BE EDIT THEN)
31 011726 032704 00000000    BIT      #$TECO,R4      ;IS IT TECO?
32 011732 001403      BEQ        7$                ;BR IF NOT
33 011734 052703 0000001    BIS      #$C$TECO,R3    ;TELL CCL TO USE TECO
34 011740 000411      BR         2$
35 011742 026127 00000000 00000000 7$:  CMP      LTRMTP(R1),#VT52 ;IS TERMINAL TYPE VT52?
36 011750 001403      BEQ        8$                ;BR IF YES (EDITOR = K52)
37 011752 052703 0000020    BIS      #$C$KED,R3     ;DEFAULT EDITOR = KED
38 011756 000402      BR         2$
39 011760 052703 0000040      8$:  BIS      #$C$K52,R3 ;DEFAULT EDITOR = K52
40 011764 032761 00000000 00000000 2$:  BIT      #$DIBOL,LSW6(R1);IS DIBOL OR DBL DEFAULT COMPILER?
41 011772 001402      BEQ        9$                ;BR IF DBL IS DEFAULT
42 011774 052703 000100      BIS      #$C$DIBL,R3    ;REMEMBER DIBOL IS DEFAULT
43 012000 032704 00000000      9$:  BIT      #$WILD,R4 ;WANT IMPLICIT/EXPLICIT WILDCARDS?
44 012004 001402      BEQ        3$                ;BR IF WANTS EXPLICIT
45 012006 052703 0000002    BIS      #$C$WILD,R3    ;SET IMPLICIT-WILDCARD FLAG
46 012012 032704 00000000      3$:  BIT      #$CLTST,R4 ;DO CCL IN TEST MODE?
47 012016 001402      BEQ        5$                ;BR IF NOT
48 012020 052703 000010      BIS      #$C$TEST,R3   ;SET TEST MODE FLAG
49 012024 032761 00000000 00000000 5$:  BIT      #$QUIET,LSW4(R1);ARE COMMAND FILES IN QUIET MODE?
50 012032 001402      BEQ        4$                ;BR IF NOT
51 012034 052703 0000004    BIS      #$C$QUIT,R3    ;SET QUIET FLAG
52 012040 110337 0000100      4$:  MOV      R3,@#CINDAT+10 ;STORE FLAG WORD INTO CHAIN DATA AREA
53 012044 110137 0000110      MOV      R1,@#CINDAT+11 ;STORE USER INDEX # IN 511 FOR CCL
54          ; Reopen channel 16 with CCL file status saved during initialization.
55 012050      .REOPEN #XAREA,#RUNCHN,#CCLSAV ;OPEN CHANNEL TO CCL.SAV FILE
56 012070 012704 000372'      MOV      #CCLNAM,R4   ;POINT TO CELL WITH CCL NAME
57 012074 105267 00000000      INCB     CINFLG      ;SIMULATE CHAIN

```

Process CCL commands

58 012100	052761	0000000	0000000	BIS	##CCLRN,LSW5(R1);REMEMBER CCL.SAV IS RUNNING
59 012106	005000			CLR	RO ;No run option flags for PLOAD
60 012110	000167	0000000		JMP	PLOAD ;LOAD AND START CCL.SAV

INSCF -- See if a command file is installed with priv

```

1          .SBTTL  INSCF  -- See if a command file is installed with priv
2          ;-----
3          ; This routine is called when we are starting a command file to see
4          ; if the command file has been installed with any privileges.
5          ; If so, the privileges are applied to the command file and current
6          ; privileges for the job.
7          ; PUSHCF should be called before this routine.
8          ;
9          ; Inputs:
10         ; FILNAM = File spec for command file being started
11         ;
12 012114 010246 INSCF:  MOV      R2,-(SP)
13         ;
14         ; See if this command file is installed
15         ;
16 012116 012700 000450'      MOV      #FILNAM,R0      ;Point to file spec
17 012122 004767 0000000     CALL      INSSRC          ;See if file is in install table
18 012126 103420             BCS      9$              ;Br if not
19         ;
20         ; Command file is installed.
21         ; Apply any privilege changes.
22         ;
23 012130 012702 0000000     MOV      #2*(PVNPW-1),R2 ;Get index to last privilege word
24 012134 056262 0000000 0000000 1$:  BIS      II$PRV+IIBUF(R2),PRIVFO(R2) ;Set some flags
25 012142 046262 0000000 0000000     BIC      II$NPV+IIBUF(R2),PRIVFO(R2) ;Clear some flags
26 012150 162702 0000002     SUB      #2,R2          ;More to do?
27 012154 002367             BCC      1$              ;Loop if yes
28 012156 056767 0000000 0000000     BIS      II$FLG+IIBUF,AFCF ;Set command file attribute flags
29 012164 004767 0000000     CALL      RSTPRV          ;Set current attributes for command file
30         ;
31         ; Finished
32         ;
33 012170 012602             9$:  MOV      (SP)+,R2
34 012172 000207             RETURN

```

TRMINI -- Perform terminal-dependent initialization

```

1          .SBTTL  TRMINI -- Perform terminal-dependent initialization
2          ;-----
3          ; TRMINI is called during job start-up initialization to perform
4          ; terminal dependent initialization.
5          ;
6          ; Inputs:
7          ; R1 = Job index number.
8          ;
9 012174 010246 TRMINI: MOV      R2, -(SP)
10         ;
11         ; Get initial LSW2 flags
12         ;
13 012176 016100 0000000 MOV      LTRMTP(R1), R0 ; GET TERMINAL TYPE FLAGS
14 012202 016102 0000000 MOV      LSW2(R1), R2 ; GET LSW2 FLAGS
15         ; VT100
16 012206 032700 0000000 BIT      #VT100, R0 ; IS THIS A VT100 TERMINAL?
17 012212 001410 BEQ      14$ ; BR IF NOT
18 012214 042702 0000000 BIC      #VT10ND, R2 ; SET FLAGS FOR LSW2
19 012220 052702 0000000 BIS      #VT10FL, R2
20 012224 .PRINT #IMVT10 ; CLEAR SCREEN
21 012232 000503 BR      15$
22         ; VT200
23 012234 032700 0000000 14$: BIT      #VT2007!VT2008, R0 ; VT200 terminal?
24 012240 001410 BEQ      32$ ; Br if not
25 012242 042702 0000000 BIC      #VT20ND, R2 ; Set flags for VT200
26 012246 052702 0000000 BIS      #VT20FL, R2
27 012252 .PRINT #IMVT10 ; Clear screen
28 012260 000470 BR      15$
29         ; VT52
30 012262 032700 0000000 32$: BIT      #VT52, R0 ; IS THIS A VT52 TERMINAL?
31 012266 001410 BEQ      16$ ; BR IF NOT
32 012270 .PRINT #IMVT52 ; CLEAR SCREEN
33 012276 042702 0000000 BIC      #VT52ND, R2 ; SET FLAGS IN LSW2
34 012302 052702 0000000 BIS      #VT52FL, R2
35 012306 000455 BR      15$
36         ; ADM3A
37 012310 032700 0000000 16$: BIT      #ADM3A, R0 ; IS THIS AN ADM3A TERMINAL?
38 012314 001410 BEQ      17$ ; BR IF NOT
39 012316 .PRINT #IMADM3 ; CLEAR SCREEN
40 012324 042702 0000000 BIC      #ADM3ND, R2 ; SET LSW2 FLAGS
41 012330 052702 0000000 BIS      #ADM3FL, R2
42 012334 000442 BR      15$
43         ; LA36
44 012336 032700 0000000 17$: BIT      #LA36, R0 ; IS THIS AN LA36?
45 012342 001405 BEQ      18$ ; BR IF NOT
46 012344 042702 0000000 BIC      #LA36ND, R2 ; SET FLAGS
47 012350 052702 0000000 BIS      #LA36FL, R2
48 012354 000432 BR      15$
49         ; LA120
50 012356 032700 0000000 18$: BIT      #LA120, R0 ; IS THIS AN LA120?
51 012362 001405 BEQ      19$ ; BR IF NOT
52 012364 042702 0000000 BIC      #LA12ND, R2 ; SET FLAGS
53 012370 052702 0000000 BIS      #LA12FL, R2
54 012374 000422 BR      15$
55         ; Hazeltine
56 012376 032700 0000000 19$: BIT      #HAZEL, R0 ; HAZELTINE TERMINAL?
57 012402 001410 BEQ      20$ ; BR IF NOT

```

TRMINI -- Perform terminal-dependent initialization

```
58 012404          .PRINT  #IMHAZL          ; CLEAR SCREEN
59 012412  042702  0000000          BIC      #HAZLNO,R2          ; SET TERMINAL CONTROL FLAGS
60 012416  052702  0000000          BIS      #HAZLFL,R2
61 012422  000407          BR          15$
62          ; DIABLE & QUME
63 012424  032700  0000000 20$:      BIT      #DIABLO!QUME,R0 ; DIABLO OR QUME TERMINAL?
64 012430  001410          BEQ          1$          ; BR IF NOT
65 012432  042702  0000000          BIC      #DIABNO,R2          ; SET FLAGS
66 012436  052702  0000000          BIS      #DIABFL,R2
67          ; Store updated LSW2 flags.
68 012442  010261  0000000 15$:      MOV      R2,LSW2(R1)
69 012446  010261  0000000          MOV      R2,LSW2S(R1)
70          ;
71          ; Finished
72          ;
73 012452  012602          1$:      MOV      (SP)+,R2
74 012454  000207          RETURN
```

VIRINI -- Virtual line initialization

```

1          .SBTTL  VIRINI -- Virtual line initialization
2          ;-----
3          ; Perform initialization for virtual lines during job startup.
4          ; If the primary line is using display windows then create a display
5          ; window for the virtual line.  Otherwise print n>.
6          ;
7          ; Inputs:
8          ;   R1 = Job index number
9          ;
10         VIRINI: MOV     R2, -(SP)
11                 MOV     R3, -(SP)
12                 MOV     R4, -(SP)
13         ;
14         ; Get job index number of our primary job
15         ;
16         MOV     LNPRIM(R1), R2    ; GET PRIMARY LINE INDEX NUMBER
17         ;
18         ; If primary line is using a display window, try to create one for
19         ; this job.
20         ;
21         TST     LWINDO(R2)        ; Is primary line using a display window?
22         BEQ     1$               ; Br if not
23         MOV     #MAKWIN, R0       ; Point to make-window argument block
24         MOV     R2, 7(R0)         ; Set index # of primary job
25         ASRB    7(R0)            ; Convert index to job number
26         EMT     375              ; Try to create a window
27         BCS     1$               ; Br if unable to create a window
28         MOV     #MAPWIN, R0      ; Point to map-window argument block
29         EMT     375              ; Select this window
30         ;
31         ; Print n>
32         ;
33         1$: MOV     LSECPT(R2), R4 ; GET ADDRESS OF TABLE WITH VIRTUAL LINES
34                 CLR     R3        ; COUNT VIRTUAL LINE #'S
35         13$: INC     R3
36                 CMPB    R1, (R4)+ ; LOOK UP OUR VIRTUAL LINE #
37                 BNE     13$
38                 ADD     #'0, R3   ; CONVERT NUMBER TO ASCII
39                 .PRINT  #CRLF     ; PRINT CR-LF
40                 .TTYOUT R3        ; PRINT LINE #
41                 .TTYOUT #76      ; PRINT '>'
42         ;
43         ; Try to copy any key definitions from primary line
44         ;
45         MOV     LNPRIM(R1), R2    ; Get our primary line number
46         CALL    INIUKD           ; Copy any user-defined keys
47         ;
48         ; Finished
49         ;
50         7$: MOV     (SP)+, R4
51                 MOV     (SP)+, R3
52                 MOV     (SP)+, R2
53         RETURN

```

CPYPRN -- Copy context info from parent job

```

1          .SBTTL  CPYPRN -- Copy context info from parent job
2          ;-----
3          ; Copy context information from a parent job.
4          ;
5 012610    010246    CPYPRN: MOV      R2,-(SP)
6 012612    010346      MOV      R3,-(SP)
7 012614    010546      MOV      R5,-(SP)
8          ;
9          ; Return immediately if there is no parent job
10         ;
11 012616    016102    00000000      MOV      LPARNT(R1),R2      ;Get # of parent job
12 012622    001002      BNE      2$      ;Br if there is a parent job
13 012624    000167    000456      JMP      9$      ;No parent job -- Nothing to copy
14         ;
15         ; Copy file context and privilege information
16         ;
17 012630    010267    165746    2$: MOV      R2,CPYCXT+2      ;Get # of parent job
18 012634    006267    165742      ASR      CPYCXT+2      ;Convert to job number
19 012640    012700    000600'      MOV      #CPYCXT,R0      ;Point to EMT arg block
20 012644    104375      EMT      375      ;Copy context from parent job
21 012646    004767    00000000      CALL     LDCLN      ;Reinit LD status
22         ;
23         ; Copy execution priority
24         ;
25 012652    012700    001110'      MOV      #PRIEMT,R0      ;Point to emt argument block
26 012656    116260    00000000 000002      MOVVB   L BSPRI(R2),2(R0) ;Get base priority from parent job
27 012664    104375      EMT      375      ;Set the job priority
28         ;
29         ; Copy user name and PPN
30         ;
31 012666    010203      MOV      R2,R3      ;Get # of parent job
32 012670    010105      MOV      R1,R5      ;Get # of our job
33 012672    070327    0000006      MUL      #6.,R3      ;Get offset to name of parent job
34 012676    070527    0000006      MUL      #6.,R5      ;Get offset to name of our job
35 012702    062703    00000000      ADD      #LUNAME,R3      ;Point to tables
36 012706    062705    00000000      ADD      #LUNAME,R5
37 012712    012700    000014      MOV      #12.,R0      ;Get # chars to move
38 012716    112325    1$: MOVVB   (R3)+,(R5)+      ;Copy user names
39 012720    077002      SOB      R0,1$
40 012722    016261    00000000 00000000      MOV      LPROJ(R2),LPROJ(R1) ;Copy project number
41 012730    016261    00000000 00000000      MOV      LPROG(R2),LPROG(R1) ;Copy programmer number
42 012736    016167    00000000 00000000      MOV      LPROJ(R1),UPPN      ;Set project number
43 012744    016167    00000000 00000020      MOV      LPROG(R1),UPPN+2 ;Set programmer number
44         ;
45         ; Copy flags from LSW2
46         ;
47 012752    016261    00000000 00000000      MOV      LSW2(R2),LSW2(R1) ;Copy LSW2 flags
48 012760    016261    00000000 00000000      MOV      LSW2S(R2),LSW2S(R1)
49         ;
50         ; Copy some flags from LSW5
51         ;
52 012766    016200    00000000      MOV      LSW5(R2),R0      ;Get flags from parent's LSW5
53 012772    042700    00000000      BIC      #^C<ISPF5>,R0      ;Clear all but selected flags
54 012776    042761    00000000 00000000      BIC      #ISPF5,LSW5(R1) ;Clear those flags in our LSW5
55 013004    050061    00000000      BIS      R0,LSW5(R1)      ;Transfer flags from parent job
56         ;
57         ; Copy some flags from LSW6

```

```

58
59 013010 016200 0000000 MOV LSW6(R2),R0 ;Get flags from parent's LSW6
60 013014 042700 0000000 BIC #^C<ISPF6>,R0 ;Clear all but selected flags
61 013020 042761 0000000 0000000 BIC #ISPF6,LSW6(R1) ;Clear those flags in our LSW6
62 013026 050061 0000000 BIS R0,LSW6(R1) ;Transfer flags from parent job
63
64 ; Copy some flags from LSW7
65
66 013032 016200 0000000 MOV LSW7(R2),R0 ;Get flags from parent's LSW7
67 013036 042700 0000000 BIC #^C<ISPF7>,R0 ;Clear all but selected flags
68 013042 042761 0000000 0000000 BIC #ISPF7,LSW7(R1) ;Clear those flags in our LSW7
69 013050 050061 0000000 BIS R0,LSW7(R1) ;Transfer flags from parent job
70
71 ; Copy some flags from LSW9
72
73 013054 016200 0000000 MOV LSW9(R2),R0 ;Get flags from parent's LSW9
74 013060 042700 0000000 BIC #^C<ISPF9>,R0 ;Clear all but selected flags
75 013064 042761 0000000 0000000 BIC #ISPF9,LSW9(R1) ;Clear those flags in our LSW9
76 013072 050061 0000000 BIS R0,LSW9(R1) ;Transfer flags from parent job
77
78 ; Copy some flags from LSW11
79
80 013076 016200 0000000 MOV LSW11(R2),R0 ;Get flags from parent's LSW11
81 013102 042700 0000000 BIC #^C<ISPF11>,R0 ;Clear all but selected flags
82 013106 042761 0000000 0000000 BIC #ISPF11,LSW11(R1) ;Clear those flags in our LSW11
83 013114 050061 0000000 BIS R0,LSW11(R1) ;Transfer flags from parent job
84
85 ; Copy Kmon prompt string
86
87 013120 010267 165756 MOV R2,EMCXCP+4 ;Set # of job we are copying from
88 013124 012767 0000000 165752 MOV #KMPRMT,EMCXCP+6 ;Set address of item
89 013132 012767 0000000 165746 MOV #<<MXPRMT+2>>8,EMCXCP+8 ;Set # bytes to copy
90 013140 012700 001076' MOV #EMCXCP,R0 ;Point to EMT arg block
91 013144 104375 ENT 375 ;Copy Kmon prompt string
92
93 ; Copy default printer form name
94
95 013146 012767 0000000 165730 MOV #UFORM,EMCXCP+6 ;Set address of item
96 013154 012767 0000000 165724 MOV #6,EMCXCP+8 ;Set # of bytes to copy
97 013162 012700 001076' MOV #EMCXCP,R0 ;Point to EMT arg block
98 013166 104375 ENT 375 ;Copy default form name
99
100 ; Copy User Command Interface (UCI) file spec
101
102 013170 012767 0000000 165706 MOV #UCISPC,EMCXCP+6 ;Set address of item to copy
103 013176 012767 000010 165702 MOV #8,EMCXCP+8 ;Set # bytes to copy
104 013204 012700 001076' MOV #EMCXCP,R0 ;Point to EMT arg block
105 013210 104375 ENT 375 ;Copy UCI file spec
106
107 ; Copy subprocess start-up command file spec
108
109 013212 012767 0000000 165664 MOV #SBPSUF,EMCXCP+6 ;Set address of item to copy
110 013220 012767 000020 165660 MOV #16,EMCXCP+8 ;Set # bytes to copy
111 013226 012700 001076' MOV #EMCXCP,R0 ;Point to EMT arg block
112 013232 104375 ENT 375 ;Copy subprocess start-up file spec
113
114 ; Copy print-window device information

```

```
115 ;
116 013234 012767 0000000 165642      MOV      #JPWDEV,EMCSCP+6; Set address of print-window device
117 013242 012767 000002 165636      MOV      #2,EMCSCP+8. ; Set # bytes to copy
118 013250 012700 001076'           MOV      #EMCSCP,R0 ; Point to EMT arg block
119 013254 104375                   EMT      375 ; Copy JPWDEV
120 013256 012767 0000000 165620      MOV      #JPWTYP,EMCSCP+6; Set address of print-window dev type
121 013264 012700 001076'           MOV      #EMCSCP,R0 ; Point to EMT arg block
122 013270 104375                   EMT      375 ; Copy JPWTYP
123 013272 012767 0000000 165604      MOV      #JPWFLG,EMCSCP+6; Set address of print-window flag word
124 013300 012700 001076'           MOV      #EMCSCP,R0 ; Point to EMT arg block
125 013304 104375                   EMT      375 ; Copy JPWFLG
126 ;
127 ; Finihsed
128 ;
129 013306 012605      9#:      MOV      (SP)+,R5
130 013310 012603      MOV      (SP)+,R3
131 013312 012602      MOV      (SP)+,R2
132 013314 000207      RETURN
```

PRTGRT -- Print the logon greeting message

```

1          .SBTTL  PRTGRT -- Print the logon greeting message
2          ;-----
3          ; During logon processing, print the TSX-Plus logon greeting message.
4          ; The greeting message is different depending on the value of GREET and
5          ; the PROFLG (flag indicating a professional machine).
6          ;
7 013316 020127 0000000 PRTGRT: CMP R1,#LSTPL ; IS THIS A REAL OR VIRTUAL LINE?
8 013322 003157          BGT  ORTVIR  ; BR IF VIRTUAL -- ABBREVIATE GREETING
9 013324 005727 0000002 TST  #GREET  ; IS PRINTED GREETING WANTED?
10 013330 001403        BEQ  23$      ; BR IF NOT
11 013332          .PRINT #GRT1      ; TSX NAME AND VERSION NUMBER
12 013340 032761 0000000 0000000 23$: BIT ##1STLG,LSW6(R1); IS THIS THE 1ST TIME THIS LINE LOGGED ON?
13 013346 001064        BNE  GRTFIN  ; BR IF NOT
14 013350 052761 0000000 0000000 BIS ##1STLG,LSW6(R1); REMEMBER 1ST LOGON HAS OCCURED
15 013356 020127 0000000        CMP  R1,#LSTPL ; IS THIS A REAL OR VIRTUAL LINE?
16 013362 101056        BHI  GRTFIN  ; BR IF VIRTUAL -- SKIP LONG MESSAGE
17 013364 005727 0000002 TST  #GREET  ; SHOULD WE PRINT GREETING MESSAGE?
18 013370 001453        BEQ  GRTFIN  ; BR IF NOT
19          ;
20          ; Calculate the message checksum.
21          ;
22 013372 012702 1765320        MOV  #SUMS-1246,R2 ; GET ADDRESS OF START OF MESSAGE
23 013376 005003        CLR  R3        ; FORM CHECKSUM IN R3
24 013400 012704 001402'        MOV  #BLK0,R4    ; TEMPORARILY STORE GREETING HERE
25 013404 112200        2$: MOVB (R2)+,R0        ; GET CHARACTER FROM MESSAGE
26 013406 060003        ADD  R0,R3        ; FORM CHECKSUM
27 013410 005400        NEG  R0        ; DECRYPT CHARACTER
28 013412 001424        BEQ  3$        ; BR IF END OF MESSAGE TEXT
29 013414 020227 0000000        CMP  R2,#LICTXT ; CHECK FOR LICENSE TEXT
30 013420 103371        BHIS 2$        ; BR IF INTO THE LICENSE TEXT
31          ;
32          ; Check for various flags that indicated message variations.
33          ;
34 013422 105767 0000000        TSTB  PROFLG      ; ARE WE RUNNING ON A PROFESSIONAL?
35 013426 001003        BNE  4$        ; BR IF YES -- PRINT TRUNCATED GREETING
36 013430 005727 0000001 TST  #GREET-1        ; CHECK FOR TRUNCATED GREETING MESSAGE
37 013434 001013        BNE  3$        ; BR IF FULL MESSAGE IS PRINTED
38 013436 020227 0000000 4$: CMP  R2,#TRGRET    ; CHECK FOR TRUNCATED MESSAGE END
39 013442 103410        BLO  3$        ; CONTINUE MOVING THE GREETING MESSAGE
40 013444 101357        BHI  2$        ; GET THE NEXT CHARACTER
41 013446 112724 0000056 MOVB  #'',(R4)+      ; PLACE PERIOD AT THE END OF TRUNCATED MESSAGE
42 013452 112724 0000015 MOVB  #CR,(R4)+    ; PLACE CARRIAGE RETURN AFTER PERIOD
43 013456 112724 0000012 MOVB  #LF,(R4)+    ; PLACE LINE-FEED AFTER PERIOD
44 013462 000750        BR  2$        ; GET THE NEXT CHARACTER
45 013464 110024 3$: MOVB  R0,(R4)+      ; PUT IN TEMP BUFFER
46 013466 001346        BNE  2$        ; BR IF MORE OF MESSAGE TEXT
47 013470 1$: .PRINT  #BLK0        ; PRINT GREETING MESSAGE
48 013476 005403        NEG  R3        ; DECRYPT CHECKSUM
49 013500 005203        INC  R3
50          ; *****
51          ; If the following range of instructions are altered --
52          ; change the program that assigns micro TSX-Plus license numbers.
53 013502 020327 0000000        CMP  R3,#SUCS    ; CHECK CHECKSUM
54 013506 001404        BEQ  GRTFIN  ; BR IF OK
55          ; Checksum failure -- kill TSX
56 013510 005037 000100        CLR  @#100      ; DIE HORRIBLY...
57 013514 005037 000060        CLR  @#60

```

```

58      ;
59      ; Checksum ok, set up site number.
60      ;
61 013520 105767 0000000 GRTFIN: TSTB     PROFLG     ;ARE WE RUNNING ON A PROFESSIONAL?
62 013524 001415          BEQ      1$          ;BR IF NOT
63      ;
64      ; Print license number on a Professional
65      ;
66 013526          .PRINT  #TM$LN1          ;"License # = mmmm-TPS-"
67 013534 016705 0000000 MOV      TSXSIT,R5      ;Get the license number
68 013540 004767 0000000 CALL     PRTDEC          ;Print it
69 013544          .PRINT  #CRLF          ;Terminate the print line
70 013552 004767 0000000 CALL     DATTIM          ;Print date and time
71 013556 000441          BR       GRTVIR      ;Go print line number
72      ;
73      ; Print license number if not on a professional
74      ;
75 013560 012767 0000000 0000000 1$: MOV      #TSXLN,TSXSIT ;STORE SITE NUMBER
76      ; If the preceeding range of instructions are altered --
77      ; change the program that assigns micro TSX-Plus license numbers.
78      ;*****
79 013566 005727 0000001 TST      #GREET-1      ;SHOULD WE PRINT DATE AND TIME?
80 013572 002451          BLT      GRTEND          ;BR IF NOT
81 013574 012704 001402' MOV      #BLK0,R4          ;TEMPORARILY STORE GREETING HERE
82 013600 012702 0000000 MOV      #LICTXT,R2      ;OBTAIN LICENSE TEXT ADDRESS
83 013604 112200          21$: MOVB     (R2)+,R0      ;GET CHARACTER FROM MESSAGE
84 013606 005400          NEG      R0              ;DECRYPT CHARACTER
85 013610 110024          31$: MOVB     R0,(R4)+      ;PUT IN TEMP BUFFER
86 013612 001374          BNE      21$            ;BR IF MORE OF MESSAGE TEXT
87 013614 112724 000200 MOVB     #200,(R4)+      ;NO CR/LF FOLLOWING TEXT
88 013620          .PRINT  #BLK0          ;PRINT GREETING MESSAGE
89      ;
90      ; If not supported ("S" or "s"), then print license is unsupported.
91      ;
92 013626 116700 0000000 PRTSUP: MOVB     SUPCOD,R0      ;OBTAIN THE SUPPORT CODE (SIGN EXTEND)
93 013632 005400          NEG      R0              ;DECRYPT THE CHARACTER
94 013634 120027 000123 CMPB     R0,#'S          ;CHECK FOR SUPPORTED LICENSE
95 013640 001406          BEQ      61$            ;BR IF LICENSE SUPPORTED
96 013642 120027 000163 CMPB     R0,#'s          ;CHECK FOR SUPPORTED LICENSE
97 013646 001403          BEQ      61$            ;BR IF LICENSE SUPPORTED
98 013650          .PRINT  #UNSUP          ;PRINT UNSUPPORTED LICENSE TEXT
99 013656 004767 0000000 61$: CALL     DATTIM          ;PRINT CURRENT DATE & TIME
100     ;
101     ; Print line number.
102     ;
103 013662 GRTVIR: .PRINT  #LINNTX          ;DISPLAY LINE #
104 013670 010105 MOV      R1,R5
105 013672 006205 ASR      R5
106 013674 004767 0000000 CALL     PRTDEC          ;
107 013700          .PRINT  #CRLF          ;
108 013706          .TTYOUT #LF            ;
109 013716 000207 GRTEND: RETURN

```

SETSUF -- Set up a start-up command file

```

1          .SBTTL  SETSUF -- Set up a start-up command file
2          ;-----
3          ; Set up a start-up command file for execution by the job.
4          ;
5          ; Inputs:
6          ;   R1 = Job index number
7          ;   R2 = Pointer to asciz command file name string.
8          ;
9 013720 010246 SETSUF: MOV     R2,-(SP)
10 013722 010346      MOV     R3,-(SP)
11          ;
12          ; Start execution of a new command file
13          ;
14 013724 004767 0000000  CALL    PUSHCF      ;SET UP FOR NEW COMMAND FILE
15          ;
16          ; Move command file name to buffer
17          ;
18 013730 012703 0000000      MOV     #CFBUF,R3      ;POINT TO COMMAND FILE BUFFER
19 013734 032761 0000000 0000000  BIT     ##DETC,LSW(R1) ;Is this a detached job?
20 013742 001004      BNE     63$      ;Br if yes
21 013744 112723 000136      MOVB    #'^(R3)+      ;PUT '^(@ AT FRONT OF COMMAND
22 013750 112723 000050      MOVB    #'^(R3)+
23 013754 112723 000100      63$: MOVB    #'@(R3)+
24 013760 112223      5$: MOVB    (R2)+,(R3)+      ;MOVE FILE NAME TO COMMAND AREA
25 013762 001376      BNE     5$      ;LOOP TILL ALL MOVED
26 013764 112763 000015 177777  MOVB    #CR,-1(R3)      ;END LINE WITH CR-LF
27 013772 112723 000012      MOVB    #LF,(R3)+
28 013776 105023      6$: CLRB    (R3)+      ;FILL REST OF BUFFER WITH NULLS
29 014000 020327 0000000      CMP     R3,#CFEND
30 014004 103774      BLO     6$
31          ;
32          ; Finished
33          ;
34 014006 012603      MOV     (SP)+,R3
35 014010 012602      MOV     (SP)+,R2
36 014012 000207      RETURN
37          ;
38          .END    START

```

Errors detected: 0

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 11432 Words ( 45 Pages)
Size of core pool: 17720 Words ( 70 Pages)
Operating system: RT-11

```

Elapsed time: 00:01:37.58

DK: TSKMN1, LP: TSKMN1=DK: TSKMN1, MAC/C/N: SYM

\$1STLG	1-78	29-12	29-14					
\$CARUP	1-95							
\$CCLRN	1-96	12-50	12-66	13-16	13-87	13-92	13-96	24-58
\$CFABT	1-110	12-5	12-63					
\$CFALL	1-116							
\$CFCCCL	1-116	13-94						
\$CFDCC	1-116	12-32						
\$CFKIL	1-160	10-21	12-5	12-59				
\$CFOPN	1-122	13-51	17-57					
\$CFSOT	1-114							
\$CHACT	1-68	12-33						
\$CLTST	1-106	24-46						
\$CTRLC	1-109	10-22						
\$CTRLS	1-101							
\$DEAD	1-159							
\$DEBUG	1-87	12-26						
\$DEFER	1-127	10-162						
\$DETCH	1-99	10-160	30-19					
\$DIBOL	1-78	24-40						
\$DILUP	1-112							
\$DISCN	1-100							
\$DOOFF	1-118	11-26						
\$DUPRN	1-113	11-21	11-25					
\$ECHO	1-115	10-162						
\$EMTTR	1-105							
\$FORM	1-114							
\$FORMO	1-116							
\$HITTY	1-68							
\$INCOR	1-131							
\$INDAB	1-109	12-51						
\$INDDF	1-158	17-26	19-53					
\$INDRN	1-158	13-16	13-87	13-96	23-21	23-73		
\$INIT	1-159							
\$INKMN	1-109							
\$KED	1-131	10-108	24-29					
\$KINIT	1-64	10-11	10-175					
\$LC	1-115	10-162						
\$MAPOK	1-93							
\$MLOCK	1-88							
\$NDIN	1-68	13-115						
\$NDINT	1-106	12-34						
\$NOVLN	1-57	10-68						
\$NOWTT	1-68	12-33	12-38					
\$NTGCC	1-117	12-35						
\$PAGE	1-115							
\$PHONE	1-159							
\$PRGLK	1-97	13-102						
\$QTSET	1-136							
\$QUIET	1-128	13-91	17-122	19-62	24-49			
\$RNIOP	1-114	12-26						
\$SCOPE	1-115							
\$SETRN	1-114	11-18						
\$SLKED	1-95							
\$SLLET	1-95	10-124						
\$SLON	1-95							
\$SLTTY	1-95							

\$SNWTT	1-156	12-36										
\$SPLJB	1-101											
\$SUCF	1-64	13-114	17-47									
\$SYSPS	1-67	10-138										
\$TAB	1-114											
\$TAPE	1-154											
\$TECO	1-136	10-106	24-29	24-31								
\$TTGAG	1-122											
\$UCLCF	1-123	10-114	16-26									
\$UCLCL	1-125	10-120	20-4									
\$UCLCM	1-125	10-117	19-14									
\$UCLRN	1-124	13-30	13-40	13-98	20-31							
\$UKMON	1-96	14-24	21-18									
\$UKMRN	1-96	13-98	21-12									
\$VNOTT	1-61											
\$VTESE	1-92	12-25										
\$WILD	1-137	10-130	24-43									
... V1	10-81	10-81	10-88	10-88	10-88	10-97	10-97	10-98	10-98	11-19	11-19	11-38
	11-39	11-43	11-47	11-49	11-55	11-58	14-9	14-32	15-127	17-43	17-43	17-43
	17-65	17-65	17-124	19-33	19-33	19-33	19-43	19-43	19-43	19-55	19-80	19-80
	19-80	20-16	20-16	20-16	21-7	21-7	21-7	23-68	23-68	24-55	24-55	26-20
	26-27	26-32	26-39	26-58	27-39	27-40	27-41	29-11	29-47	29-66	29-69	29-88
	29-98	29-103	29-107	29-108								
... V2	10-81	10-81	10-81#	10-81#	10-88	10-88	10-88#	10-88#	10-97	10-97	10-97#	10-97#
	10-98	10-98#	11-19	11-19#	17-43	17-43	17-43#	17-43#	17-65	17-65	17-65#	17-65#
	19-33	19-33	19-33#	19-33#	19-43	19-43	19-43#	19-43#	19-55	19-55#	19-80	19-80
	19-80#	19-80#	20-16	20-16	20-16#	20-16#	21-7	21-7	21-7#	21-7#	23-68	23-68
	23-68#	23-68#	24-55	24-55	24-55#	24-55#						
AB1	1-73	4-30										
AB10	1-74	4-37										
AB11	1-74	4-38										
AB12	1-74	4-39										
AB13	1-74	4-40										
AB14	1-74	4-41										
AB15	1-74	4-42										
AB16	1-74	4-43										
AB2	1-74	4-31										
AB3	1-74	4-32										
AB4	1-74	4-33										
AB5	1-74	4-34										
AB6	1-74	4-35										
ABCMD	1-76	18-17										
ABM1	1-73	4-28										
ABM10	1-73	4-21										
ABM11	1-72	4-20										
ABM12	1-72	4-19										
ABM13	1-72	4-18										
ABM14	1-72	4-17										

[illegible]

C\$TEST	2-28#	24-48
C\$WILD	2-26#	24-45
C. CSW	1-119	
C. DEVG	1-119	
C. SBLK	1-119	
CALUCL	1-184	16-14 16-30 19-20 20-11#
CALUKM	14-28	21-5#
CBFEND	9-2#	15-103
CCLNAM	1-199	7-32# 24-56
CCLSAV	1-104	24-55
CD*\$SZ	1-120	8-70 8-79
CD*\$UB	1-120	
CD*BAS	1-120	
CD\$DVU	1-120	
CD\$JOB	1-120	
CD\$NAM	1-120	
CD\$TOP	1-136	
CDBUF	1-50	8-71 8-80 9-8#
CDGADR	1-53	8-69#
CDGEMT	1-53	8-67#
CDPADR	1-53	8-78#
CDPEMT	1-53	8-76#
CF\$IND	1-160	
CF\$QUT	1-160	
CF2DEP	1-48	
CFABLV	1-161	
CFBLK	1-128	
CFBUF	1-104	13-53 13-67 17-65 30-18
CFCHAN	1-127	17-43 17-43 17-65 17-65 19-33 19-33 19-43 19-43 19-55 19-55
CFEND	1-104	17-70 30-29
CFHOLD	1-135	
CFIND	1-118	19-51
CFJMP	17-113	17-115 17-132#
CFLFL4	1-123	
CFNEST	1-121	
CFOPEN	17-45	17-56# 19-63
CFPNT	1-128	13-112 14-26 17-70*
CFPRM	17-66	17-76# 17-79
CFSEND	1-122	
CFSP	1-122	10-35*
CFSPND	1-134	12-39 12-42*
CFSQEZ	1-128	13-55
CFSTK	1-64	10-35
CFSTRT	1-60	12-41
CFSTS	1-160	
CHAIN	1-109	
CHKDEV	1-199	
CHKDLM	1-189	
CHKMNT	1-174	
CHKMTX	1-174	
CHKTTD	1-199	17-39 19-29 19-77
CINDAT	1-144	13-21 13-32 13-34 13-61 20-22 20-26 20-27* 24-11 24-52* 24-53*
CINFLG	1-61	11-62* 12-7* 12-9 20-29* 23-70* 24-57*
CKLK	13-102#	
CKPASL	12-73	13-5#
CKPRIV	1-176	

CKSF2	13-103	13-110#										
CMDACC	1-85	6-68										
CMDALC	1-83	6-51										
CMDASN	1-83	6-48										
CMDBOT	1-85	1-85	6-82	6-83								
CMDBUF	1-167	1-184	9-1#	13-35	15-6	15-112	16-7	16-24	16-31*	17-19	17-114	18-11
	19-24	19-57	19-71	20-23	24-12							
CMDCCCL	1-199	6-11	6-15	6-16	6-17	6-18	6-19	6-26	6-27	6-28	6-29	6-30
	6-31	6-34	6-35	6-36	6-54	6-55	6-56	6-58	6-59	6-60	6-63	6-72
	6-73	6-75	24-11#									
CMDDAT	1-82	6-37										
CMDDEF	1-86	6-43										
CMDDET	1-84	6-67										
CMDDLCL	1-83	6-52										
CMDDMT	1-83	6-41										
CMDDSN	1-165	6-49	6-50									
CMDDSP	1-83	6-42										
CMDEND	7-51#	13-39*	15-111*									
CMDFMT	1-82	6-33										
CMDFRM	1-165	6-32										
CMDHD	1-67	6-8#	18-10									
CMDIND	6-53	22-6#										
CMDINI	1-85	6-70										
CMDINS	1-77	6-78										
CMDKIL	1-84	6-64										
CMDMEM	1-84	6-57										
CMDMNT	1-82	6-39										
CMDMON	1-82	6-40										
CMDOFF	1-108	6-44	6-45	6-46	6-47	11-28	13-104	17-50				
CMDPAU	1-84	6-65										
CMDRCL	1-84	6-66										
CMDREM	1-83	6-12										
CMDRSM	1-65	6-13										
CMDRST	1-81	6-14										
CMDRSY	1-81	6-9										
CMDRUN	1-81	6-10										
CMDSET	1-81	6-21										
CMDSHO	1-81	6-20										
CMDSHT	1-86	6-84										
CMDSND	1-81	6-22										
CMDSPN	1-65	6-80										
CMDSPD	1-84	6-62										
CMDSQZ	1-85	6-71										
CMDTIM	1-82	6-38										
CMDUSE	1-85	6-69										
CMDWHO	1-82	6-24	6-25									
CMDYEL	1-80	6-23										
COAD	1-190	1-191										
COAF	1-203											
COAL	1-190	1-191										
COL00	1-184											
COLEQL	9-21#	15-11*	15-51	15-60*	16-9							
CONFIG	1-117											
CORUSR	1-62	10-6	14-6	14-13	17-6	17-46	17-56	17-120	18-27	19-13	20-30	21-5
	23-20	23-72										
CPUAH	1-184	7-47#										

CPUAL	1-184	7-48#				
CPUMSG	1-48					
CPYCXT	8-39#	28-17*	28-18*	28-19		
CPYPRN	10-171	28-5#				
CR	2-14#	13-71	29-42	30-26		
CRLF	1-179	11-58	14-9	27-39	29-69	29-107
CS#RON	1-91					
CSHCLN	8-35#					
CSHHD	1-164					
CSHMSG	1-52					
CSIMS1	1-178	3-8				
CSIMS2	1-176	3-8				
CSIMS3	1-201	3-8				
CSIMS4	1-198	3-8				
CTMSG	1-47					
CTRLTT	1-107					
CURMTX	1-189					
CURPRM	1-133					
CVDVNM	1-186					
CVTTAB	1-167					
CW#50H	1-117					
CXTBAS	1-111					
CXTPAG	1-90					
CXTWDS	1-111					
DATTIM	1-165	29-70	29-99			
DCCRD	1-140					
DCCWR	1-140					
DCTRD	1-140					
DCTWR	1-140					
DEADEV	1-174					
DELSPC	1-188					
DELWIN	8-169#	12-17				
DETARG	1-203	8-15#				
DETHD	1-203					
DETTXT	1-182					
DEVHD1	1-185					
DEVIDL	1-190	1-190	1-191			
DEVUNT	1-177	7-39#				
DFJMEM	1-64	10-36				
DIABFL	1-128	26-66				
DIABLO	1-149	26-63				
DIABNO	1-129	26-65				
DIVIDE	1-183					
DIVSOR	1-196	3-5#				
DJABMS	1-194					
DKCOM	7-4#	17-12	19-25	22-21		
DKSAV	1-167	7-6#				
DLCEMT	1-50	8-57#				
DLMSG	1-194					
DLTXT	1-181					
DMTALL	1-194					
DMTARG	1-173	8-30#				
DMTSUB	1-199					
DOLRAT	9-19#	15-9*	15-71*	17-28	17-129*	
DORUN	1-165	19-83				
DOSTOP	1-195					

DOTAT	9-20#	15-10*	15-84*	17-21	17-130*							
DZTXT	1-201											
EDII	1-78	10-102										
EDTFIL	1-187	11-46										
EM#DAA	1-53											
EM#NUC	1-58	16-13										
EM#NUK	1-52	21-19										
EM#OVL	1-70	4-5										
EM#SFP	1-70	4-4										
EMCXP	8-143#	28-87*	28-88*	28-89*	28-90	28-95*	28-96*	28-97	28-102*	28-103*	28-104	28-109*
	28-110*	28-111	28-116*	28-117*	28-118	28-120*	28-121	28-123*	28-124			
ERRLOC	1-63											
ERRSEV	1-137	10-51*	12-47									
ERRSPC	1-107	11-41	11-45	12-4*								
ESC	1-52	2-19#										
FC#CDX	1-164											
FC#LNK	1-164											
FD#NAM	1-164											
FF	2-18#	15-20	24-16									
FILERM	1-75	11-43										
FILNAM	1-168	1-169	7-43#	17-38	17-43	19-28	19-33	19-41*	19-43	19-76	19-80	25-16
FKILL	1-168	13-66	15-105	16-13	17-25	17-52	17-134	17-136	18-17	19-85	20-37	20-41
	21-19	22-12	23-16									
FND CMD	18-13	18-27#										
FPRINT	1-168	10-19	17-49									
FSTD L	1-99											
GAGMSG	1-193											
GENMON	1-53	8-186#	10-192*	10-193	10-197*	10-198	12-74					
GENTOP	1-108	10-17										
GETSYP	1-60	10-140										
GUTCML	15-124	16-6#										
GREET	1-46	2-9#	29-9	29-17	29-36	29-79						
GRT1	1-146	29-11										
GRTEND	29-80	29-109#										
GRTFIN	1-46	29-13	29-16	29-18	29-54	29-61#						
GRTINI	10-161	10-166#										
GRTVIR	29-8	29-71	29-103#									
GTRD50	1-173											
HANBSY	1-178											
HANCHN	1-78	2-21#										
HANENT	1-80											
HANIDX	1-177	7-49#										
HANSIZ	1-80											
HAZEL	1-88	26-56										
HAZLFL	1-88	26-60										
HAZLNO	1-88	26-59										
HIMAP	1-145											
HIPRI	1-179											
HNBUFF	1-177	7-36#										
HUPARG	1-198	8-106#										
IDNCMD	13-44	16-33	18-10#									
II##SZ	1-70	8-87	8-95									
II#FLG	1-58	25-23										
II#NPV	1-58	25-25										
II#PRV	1-58	25-24										
IIBUF	1-45	1-58	3-9#	8-88	8-96	25-24	25-25	25-28				

[illegible]

[illegible]

LDPDEV	1-126				
LDSize	1-126				
LF	2-13#	13-72	29-43	29-108	30-27
LF\$OPN	1-163				
LF\$WRT	1-163				
LFWLIM	1-94	12-30*			
LGDOVER	1-47	4-8			
LICTXT	1-146	29-29	29-82		
LINBUF	1-89				
LINCNT	1-91				
LINCUR	1-94				
LINFRE	1-194				
LINNTX	1-77	29-103			
LINNXT	1-89				
LINPNT	1-91				
LJSW	1-107				
LMONHD	1-70	12-70			
LMXLN	1-159				
LMXNUM	1-157				
LMXPRM	1-160				
LNBLKS	1-111				
LNMAP	1-117				
LNPRIM	1-117	27-16	27-45		
LNSBLK	1-112				
LNSPAC	1-121	12-27*			
LOCKTX	1-183				
LOCMSG	1-76	11-49			
LOGASN	1-175				
LOGBLK	1-162				
LOGBUF	1-162				
LOGCHN	1-162	8-136			
LOGCLS	1-179				
LOGFLG	1-162				
LOGPTR	1-162				
LOMAP	1-145				
LOTBUF	1-92				
LOTNXT	1-92				
LOTPNT	1-92				
LOTSIZ	1-93				
LOTSPC	1-93				
LPARNT	1-118	10-175	28-11		
LPRG1	1-147	11-6*			
LPRG2	1-147	11-7*			
LPROG	1-100	28-41	28-41*	28-43	
LPROJ	1-100	28-40	28-40*	28-42	
LRBFIL	1-118	12-22*			
LRDTIM	1-91	12-28*			
LSCCA	1-114	11-10*			
LSECPT	1-105	27-33			
LSTACT	1-89				
LSTATE	1-143				
LSTDLL	1-99	10-131	10-177		
LSTFRM	7-46#				
LSTMX	1-157				
LSTPL	1-138	10-66	10-133	29-7	29-15
LSTPRM	1-134	17-88			

MXJPRI	1-161	10-43*										
MXPRMT	1-62	1-141	28-89									
MXVEC	1-159											
NAMTOP	1-146	10-41										
NARGS	6-8#	6-9	6-9	6-10	6-10	6-11	6-11	6-12	6-12	6-13	6-13	6-14
	6-14	6-15	6-15	6-16	6-16	6-17	6-17	6-18	6-18	6-19	6-19	6-20
	6-20	6-21	6-21	6-22	6-22	6-23	6-23	6-24	6-24	6-25	6-25	6-26
	6-26	6-27	6-27	6-28	6-28	6-29	6-29	6-30	6-30	6-31	6-31	6-32
	6-32	6-33	6-33	6-34	6-34	6-35	6-35	6-36	6-36	6-37	6-37	6-38
	6-38	6-39	6-39	6-40	6-40	6-41	6-41	6-42	6-42	6-43	6-43	6-44
	6-44	6-45	6-45	6-46	6-46	6-47	6-47	6-48	6-48	6-49	6-49	6-50
	6-50	6-51	6-51	6-52	6-52	6-53	6-53	6-54	6-54	6-55	6-55	6-56
	6-56	6-57	6-57	6-58	6-58	6-59	6-59	6-60	6-60	6-61	6-61	6-62
	6-62	6-63	6-63	6-64	6-64	6-65	6-65	6-66	6-66	6-67	6-67	6-68
	6-68	6-69	6-69	6-70	6-70	6-71	6-71	6-72	6-72	6-73	6-73	6-74
	6-74	6-75	6-75	6-76	6-76	6-77	6-77	6-78	6-78	6-79	6-79	6-80
	6-80	6-81	6-81	6-82	6-82	6-83	6-83	6-84	6-84			
NATXT	1-181											
NEWCMD	13-111	13-113	14-5#									
NEWJSW	1-107											
NFRESB	1-132											
NINTXT	1-181											
NLINES	1-119											
NOABRT	11-31	12-4#										
NOCF	1-76	17-49	17-52									
NOCIN	1-172	12-10	12-17#									
NOCMD	18-16	19-13#										
NODAT	1-186											
NOFIL	1-184	1-192										
NOFLAG	1-174	9-18#										
NOFRDL	1-203											
NOICMD	17-40	17-52#										
NOIND	1-178	17-25	22-12									
NOLDMT	1-186											
NONEMS	1-186											
NOOPTT	1-192											
NOPMGN	1-188											
NOPRG	1-168	1-172										
NOSTRT	1-171											
NOTAVL	1-187											
NOTON	1-193											
NOTXT	1-181											
NOUCL	9-22#	13-43*	15-12*	16-32*	19-16	20-6						
NOUDC	1-185											
NSPLDV	1-125	10-79										
NSWPMS	1-189											
NUCHN	1-118											
NUMDEV	1-152											
NUMMNT	9-23#											
NUMON	1-94											
NUMPRM	1-76	17-134										
OCTFIX	1-181											
OCTPRT	1-201	11-51										
OF\$\$SZ	1-154											
OF\$DEV	1-153											
OF\$FIL	1-153											

DF\$FLG	1-153					
DF\$UNT	1-153					
OFFEMT	1-195	8-100#				
OKFEND	1-106					
OKFILE	1-106					
OPRCMD	1-84	6-61				
OT\$RON	1-154					
OTHRON	1-197					
OTRMNT	1-199					
OVLVEQ	1-201	11-55				
OVRCOR	1-171					
PO\$DBG	1-69					
PO\$SPV	1-69					
P2\$VIR	1-57	10-70				
PASLIN	1-137	12-8	13-5	13-41	13-65	13-97
PAUMSQ	1-167					
PBFEND	1-122	17-103*				
PBUFND	1-48	9-17#				
PF\$IOW	1-156					
PF\$SYS	1-156					
PLDAD	1-165	20-33	21-14	23-76	24-60	
PMBUSY	1-188					
PNAME	1-152	1-177				
POPCF	1-129	17-51	19-56	17-67		
PPNMSG	1-47					
PRCALL	1-165	11-16	11-23	12-58		
PRGSIZ	1-89					
PRGTOP	1-89					
PRIEMT	8-151#	28-25				
PRIVA0	1-69	10-60*				
PRIVA2	1-57	10-71*				
PRIVC0	1-69	10-63*	10-74*			
PRIVF0	1-59	10-62*	10-73*	25-24*	25-25*	
PRIVS0	1-69	10-61*				
PRIVS2	1-57	10-72*				
PRMBUF	1-134	17-87				
PRMEND	1-134	17-95				
PRMPNT	1-133	17-86				
PROFLG	1-86	29-34	29-61			
PRSCMD	15-6#					
PRTBUF	1-186	9-16#				
PRTDAT	1-202					
PRTDC2	1-183					
PRTDC3	1-183					
PRTDEC	1-177	29-68	29-106			
PRTFIX	1-180					
PRTFNM	1-186					
PRTGRT	1-46	10-166	29-7#			
PRTLN	1-182					
P RTPCT	1-196					
PRTR50	1-202	11-57				
PRTSPC	1-180					
PRTSUP	29-92#					
PRTTIM	1-202					
PRTTMV	1-184					
PRTTOD	1-202					

[illegible]

[illegible]

SH\$VAL	1-79			
SHMTH1	1-185			
SHMTH2	1-185			
SHOHD	1-200			
SHTMSG	1-195			
SIZEMT	1-198	8-20#		
SIZVAL	1-172	8-21#		
SJEMT	1-143	8-124#		
SKPSPC	1-59	22-17		
SMONHD	1-71	12-72		
SO\$NO	1-80			
SO\$NVL	1-80			
SO\$DCT	1-80			
SPACE2	1-182			
SPACE3	1-182			
SPACE5	1-183			
SPACE6	1-186			
SPACTV	1-190	1-191		
SPCF	1-192			
SPFLK	1-192			
SPFUL	1-192			
SPGEMT	1-192	8-9#		
SPLACT	1-195			
SPLCHN	1-102	10-81		
SPLHD	1-189			
SPLHLA	1-180	8-136#		
SPLPND	1-197			
SPSNG	1-190	1-192		
SPUBUF	1-65	10-42		
SPWFM	1-190	1-191		
SRTSMS	1-197			
SRTTXT	1-200			
SSRMAP	1-200			
START	1-48	8-114	10-5#	30-38
STDNAM	1-75	10-46		
STLGCN	1-165			
STLGHD	1-179			
STPASK	1-197			
STPFLG	1-102			
STRLEN	1-76	17-136		
SUBARO	1-187			
SUBTXT	1-200			
SUCF2	1-57	13-110	13-116	13-118*
SUCS	1-146	29-53		
SUM1	1-196			
SUM2	1-196			
SUM3	1-196			
SUM4	1-196			
SUM5	1-197			
SUM6	1-197			
SUM7	1-197			
SUMS	1-146	29-22		
SUPCOD	1-146	29-92		
SWPTX	1-183			
SXBPNT	1-65	10-42*		
SYCOM	7-3#			

SYHD1	1-182						
SYHD2	1-182						
SYINDEX	1-152						
SYINTX	9-25#	23-33					
SYNAME	1-153						
SYSAV	1-167	7-5#	19-72	23-63			
SYSDAT	1-144						
SYTIMH	1-144						
SYTIML	1-144						
SYTXT	9-24#	19-50					
SYUNIT	1-152						
TAB	2-17#	15-18	15-117	24-14			
TALEMT	1-50	8-62#					
TBLOVF	1-176						
TECO	1-78	10-104					
TKIVAL	1-144						
TM\$LN1	1-77	29-66					
TMIDLH	1-67						
TMIOH	1-67						
TMIOWH	1-66						
TMSWPH	1-67						
TMSWTH	1-67						
TMTOTH	1-66	1-196					
TMTOTL	1-66	1-196					
TMUSRH	1-66						
TOMPRM	17-89	17-134#					
TOOLNG	1-76	13-66					
TOTMMS	1-200						
TOTON	1-102						
TOTXT	1-179						
TRGRET	1-146	29-38					
TRMINI	10-150	26-9#					
TRMSTR	1-169						
TRYUCL	19-82	20-4#					
TSKMNI	1-11#	1-51					
TSKMON	1-10#	1-46	1-51				
TSR	1-157						
TSXLN	1-146	29-75					
TSXSIT	1-146	29-67	29-75*				
TSXSMS	1-201						
UC\$MDC	1-164	10-96*					
UC\$NDC	1-164	10-95*					
UCHAN	1-116	11-13	11-15*				
UCIDEF	1-52	3-4#					
UCISPC	1-93	21-6	28-102				
UCLBLK	1-163	10-86	16-11	16-28	19-18		
UCLCMD	1-85	6-74					
UCLDAT	1-163	10-88					
UCLNAM	1-121	20-16	20-28				
VERSEV	1-137	11-35*	12-46	12-47	12-54*	23-61	23-71
UFORM	1-109	10-47	28-95				
UFPTRP	1-123	11-9*					
UHIMEM	1-111	10-41*					
UKMNAM	1-96						
UMSSMS	1-200						
UMSYTP	1-99						

[illegible]

...CM0	14-32	14-32	14-32	14-32	15-127	15-127	15-127	15-127	17-124	17-124	17-124	17-124
...CM1	10-81	10-88	10-97	17-43	17-65	19-33	19-43	19-80	20-16	21-7	23-68	24-55
...CM2	10-81	10-88	10-88	10-97	10-97	10-97	10-97	17-43	17-43	17-65	17-65	17-65
	17-65	19-33	19-33	19-43	19-43	19-80	19-80	20-16	20-16	21-7	21-7	23-68
	24-55											
...CM3	10-98	11-19	19-55									
...CM5	10-81	10-88	10-97	11-38	11-39	11-43	11-47	11-49	11-55	11-58	14-9	17-43
	17-65	19-33	19-43	19-80	20-16	21-7	23-68	24-55	26-20	26-27	26-32	26-39
	26-58	27-39	27-40	27-41	29-11	29-47	29-66	29-69	29-88	29-98	29-103	29-107
	29-108											
...CM7	10-97	17-65										
.CLOSE	1-39#	10-98	11-19									
.CSIGE	1-37#											
.CSISP	1-35#											
.DATE	1-38#											
.ENTER	1-40#											
.EXIT	1-40#											
.FPROT	1-41#											
.GTIM	1-38#											
.GTLIN	1-38#	14-32	15-127	17-124								
.GVAL	1-41#											
.HERR	1-41#											
.LOOKU	1-39#	10-88	17-43	19-33	19-43	19-80	20-16	21-7				
.PRINT	1-39#	11-38	11-39	11-43	11-47	11-49	11-55	11-58	14-9	26-20	26-27	26-32
	26-39	26-58	27-39	29-11	29-47	29-66	29-69	29-88	29-98	29-103	29-107	
.PURGE	1-36#	19-55										
.PVAL	1-41#											
.READW	1-36#	17-65										
.REOPE	1-37#	10-81	23-68	24-55								
.SAVES	1-37#											
.SERR	1-41#											
.SRESE	1-35#											
.TTOUT	1-35#											
.TTYIN	1-36#											
.TTYOU	1-36#	27-40	27-41	29-108								
.WRITW	1-40#	10-97										
CMDDEF	5-38#	6-9	6-10	6-11	6-12	6-13	6-14	6-15	6-16	6-17	6-18	6-19
	6-20	6-21	6-22	6-23	6-24	6-25	6-26	6-27	6-28	6-29	6-30	6-31
	6-32	6-33	6-34	6-35	6-36	6-37	6-38	6-39	6-40	6-41	6-42	6-43
	6-44	6-45	6-46	6-47	6-48	6-49	6-50	6-51	6-52	6-53	6-54	6-55
	6-56	6-57	6-58	6-59	6-60	6-61	6-62	6-63	6-64	6-65	6-66	6-67
	6-68	6-69	6-70	6-71	6-72	6-73	6-74	6-75	6-76	6-77	6-78	6-79
	6-80	6-81	6-82	6-83	6-84							
FABORT	5-16#	13-66	15-105	16-13	17-25	17-52	17-134	17-136	18-17	19-85	20-37	20-41
	21-19	22-12	23-16									
FERR	5-5#	10-19	17-49									
TBLDEF	5-26#	6-8										
TBLEN	5-52#	6-85										