

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	SETSUFG -- Set up a start-up command file

```

1          .TITLE  TSKMN1 -- TSX-Plus Keyboard Monitor
2          .ENABL  LC
3          .DSABL  GBL
4 000000    .ASECT
5          =      42
6 000042    000042' .WORD  KMSTK          ; DEFINE INITIAL STACK POINTER
7          =      300
8 000300    000000G .WORD  MDT          ; # MINUTES OF SYSTEM UPTIME IF DEMO VERSION
9 000000    .CSECT  TSKMON
10 TSKMON:
11 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, 1984, 1985, 1986, 1987, 1988.
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, .GTIM, .DATE
39          .MCALL  .PRINT, .CLOSE, .LOOKUP
40          .MCALL  .WRITW, .ENTER, .EXIT
41          .MCALL  .SERR, .HERR, .FPROT, .GVAL, .PVAL
42 ;
43 ; Global definitions
44 ;
45          .GLOBL  INGEMT, INGADR, INPEMT, INPADR, IIBUF
46          .GLOBL  TSKMON, GREET, PRGRT, GRTFIN, KMNOT1
47          .GLOBL  LGOVER, BOTEMT, KEYBUF
48          .GLOBL  START, PBUFND
49          .GLOBL  REMNDR, KEYEND, KMSTK
50          .GLOBL  ALDEMT, DLCEMT, TALEMT, ALCDEV, CDBUF
51          .GLOBL  TSKMON, TSKMN1
52          .GLOBL  EM$NUK, ESC, UCIDEF
53          .GLOBL  GENMON, CDGEMT, CDGADR, CDPEMT, CDPADR
54 ;
55 ; Global references
56 ;
57          .GLOBL  KMONCE, KNOT, EXCJOB

```

58	. GLOBL	\$NOVLN, P2\$VIR, PRIVA2, PRIVS2, SC\$ERR, SUCF2
59	. GLOBL	INSSRC, II\$PRV, II\$NPV, AFCF, II\$FLG, EM\$NUC
60	. GLOBL	ISPF5, ISPF6, ISPF7, ISPF9, PRIVF0, SKPSPC, ISPF11
61	. GLOBL	LBSPRI, GETSYP, JPWDEV, JPWTYP, SBPSUF, CFSTRT
62	. GLOBL	ABRTAD, ABRTCD, CINFLG, INIUKE, R50KMN, ABRTOV
63	. GLOBL	CORUSR, LSW, SERFLG, KMPRMT
64	. GLOBL	UTRPAD, JSWLOC, MAXMEM, MAXPRI, JPWFLG
65	. GLOBL	\$KINIT, CFSTK, DFJMEM, \$SUCF
66	. GLOBL	SPUBUF, SXBPNT, LWINDO, CMDSPN, CMDRSM
67	. GLOBL	RUNCHN, RUNNAM
68	. GLOBL	CMDHD, \$SYSPS
69	. GLOBL	VEDIT, WILDFL, \$NOIN, \$NOWTT, \$CHACT
70	. GLOBL	PRIVCO, PRIVAO, PRIVSO, PVNPW, RUNARG
71	. GLOBL	JS\$KMN, JS\$ON, LMDNHD, EM\$OVL, II\$SZ, RSTPRV, EM\$SFP
72	. GLOBL	MSTALC, ALCERR, ABM24, ABM23, ABM22, ABM21, ABM20, SMONHD
73	. GLOBL	ABM17, ABM16, ABM15, ABM14, ABM13, ABM12, ABM11, JS\$LOG
74	. GLOBL	ABM10, ABM7, ABM6, ABM5, ABM4, ABM3, ABM2, ABM1, AB1
75	. GLOBL	AB2, AB3, AB4, AB5, AB6, AB10, AB11, AB12, AB13, AB14, AB15, AB16
76	. GLOBL	REGEN, STDNAM, IMVT10, IMVT52, IMADM3, IMHAZL, MONTXT, FILERM
77	. GLOBL	LOCMSG, TOOLNG, NOCF, NUMPRM, STRLEN, ABCMD, MISUCL
78	. GLOBL	URCMD, UNSUP, LINNTX, CMDINS, TM\$LN1
79	. GLOBL	TECO, EDIT, \$1STLG, \$DIBOL, HANCHN
80	. GLOBL	CMDYEL
81	. GLOBL	CMDRSY, CMDRUN, CMDRST, CMDSHO, CMDSET, CMDSDND
82	. GLOBL	CMDWHO, CMDFMT, CMDDAT, CMDTIM, CMDMNT, CMDMON
83	. GLOBL	CMDDMT, CMDDSP, CMDASN, CMDALC, CMDDLC, CMDREM
84	. GLOBL	CMDMEM, OPRCMD, CMDSPD, CMDKIL, CMDPAU, CMDDDET, CMDRCL
85	. GLOBL	CMDACC, CMDUSE, CMDINI, CMDSQZ, UCLCMD, CMDBOT, CMDNAC
86	. GLOBL	CMDSHT, PROFLG, CMDDEF
87	. GLOBL	LSW9, \$DEBUG, LSW11
88	. GLOBL	HAZEL, HAZLFL, HAZLNO, MDT
89	. GLOBL	LRDTIM
90	. GLOBL	\$VTESE
91	. GLOBL	UCISPC
92	. GLOBL	LAFSIZ, LFWLIM, ILSW2
93	. GLOBL	\$SLLET
94	. GLOBL	LSUCF, \$CCLRN, \$UKMON, \$UKMRN
95	. GLOBL	KL3CLR, \$PRGLK, LSW5
96	. GLOBL	LSTD, \$DETC
97	. GLOBL	LPROJ, LPROG, LUNAME, UPPN
98	. GLOBL	LCONTM
99	. GLOBL	USPLCH, SPLCHN
100	. GLOBL	CFBUF, CFEND, CCLSAV
101	. GLOBL	MINTIM, LSECPT
102	. GLOBL	OKFILE, OKFEND, \$CLTST, \$NOINT
103	. GLOBL	ERRSPC
104	. GLOBL	GENTOP, CMDOFF
105	. GLOBL	\$CTRLC, LSW2, UFORM, \$INDAB
106	. GLOBL	\$CFABT, INDSTA, INDERR
107	. GLOBL	UHIMEM
108	. GLOBL	LSW3, LSW2S, \$DUPRN
109	. GLOBL	LSCCA, \$RNIOP, \$SETRN
110	. GLOBL	\$ECHO, \$LC
111	. GLOBL	UCHAN, \$CFDCC, \$CFCC
112	. GLOBL	LNPRIM, \$NTGCC
113	. GLOBL	\$DOOFF, LRBFIL, CFIND, LPARNT
114	. GLOBL	CD\$SZ

115	. GLOBL	LTSCMD, LNSPAC, VU\$CL, UCLNAM
116	. GLOBL	\$CFOPN, PBFEND, CFSP, VUCLMC
117	. GLOBL	UFPTRP, \$UCLCF, VUCLOR
118	. GLOBL	\$UCLRN
119	. GLOBL	NSPLDV, \$UCLCM, \$UCLCL
120	. GLOBL	\$DEFER, CFCHAN, SCHAIN
121	. GLOBL	CFPNT, \$QUIET, DIABFL, KDOCIN, CFSQEZ
122	. GLOBL	DIABNO, VT52NO, LA36NO, LA36FL, POPCF
123	. GLOBL	LSW4, KL4CLR
124	. GLOBL	\$KED
125	. GLOBL	PRMPNT
126	. GLOBL	LSTPRM, PRMBUF, PRMEND, CFSPND
127	. GLOBL	LCOL, \$TECO
128	. GLOBL	\$WILD, ERRSEV, UERSEV, PASLIN
129	. GLOBL	LSTPL
130	. GLOBL	KMPRMT, MXPRMT
131	. GLOBL	SJEMT, RJEMT
132	. GLOBL	CINDAT
133	. GLOBL	TSXLN, TSXSIT, GRT1, TRGRET, LICTXT, SUPCOD, NAMTOP, SUMS, SUCS
134	. GLOBL	LPRG1, LPRG2
135	. GLOBL	LCBIT, LA36, LA120, VT52, VT100, DIABLO, QUME
136	. GLOBL	ADM3A, LTRMTP, LA12FL, LA12NO, VT52FL
137	. GLOBL	VT10FL, VT10NO, ADM3FL
138	. GLOBL	ADM3NO
139	. GLOBL	RESDEV, OKFAND, UKFNND
140	. GLOBL	VT2007, VT2008, VT20NO, VT20FL
141	. GLOBL	LSW6, \$SNWTT
142	. GLOBL	\$INDDF, \$INDRN, IN\$ACT, IN\$CNT, IN\$CMD, INDSAV
143	. GLOBL	LSW7, \$CFKIL
144	. GLOBL	MXJPRI
145	. GLOBL	LOGCHN
146	. GLOBL	UCLBLK, UCLDAT
147	. GLOBL	UC\$NDC, UC\$MDC
148	. GLOBL	PLOAD, DORUN, CMDFRM, CMDDSN, DATTIM, PRGALL
149	. GLOBL	LDCLEN, R50TSX, R50UCL, INDABT
150	. GLOBL	RDCMD, DKSAV, SYSAV, SEARCH
151	. GLOBL	FKILL, ABRTCF, ACRFN, XAREA, FPRINT
152	. GLOBL	PUSHCF, FILNAM, R50DIR, R50SY, R50IND, R50SAV
153	. GLOBL	INDACT, R50DUP, R50PIP, R50KED, R50K52, R50KEX, R50COM
154	. GLOBL	BLKO, RDERM, R50VIR, RUNEMT
155	. GLOBL	LDNAM, NOCIN, SIZVAL, ASKLN, BADCMD, KCSIBF
156	. GLOBL	ASDEX, KCSIMS, R50BUF, R50LDO, MNTDEV, DMTARG
157	. GLOBL	NOFLAG, ILLCMD
158	. GLOBL	R5OLD, R5ODSK, R5OLD7
159	. GLOBL	CSIMS2, R50NO, AMBOPT
160	. GLOBL	PRTDEC, DEVUNT, HANIDX, HNBUF
161	. GLOBL	CSIMS1, NOIND
162	. GLOBL	CRLF, R5OLOG
163	. GLOBL	SPLHLA
164	. GLOBL	CPUAH, CPUAL, CMDBUF, CALUCL
165	. GLOBL	PRTBUF
166	. GLOBL	EDTFIL, MNFLGS, MNBPC
167	. GLOBL	MNBASE, MNTOP, MONAR1, MONAR2
168	. GLOBL	ALDEX, ALDBLK
169	. GLOBL	SPGEMT
170	. GLOBL	MSGBUF, MSGEND
171	. GLOBL	YELEMT

172	. GLOBL	AUTHFN, OFFEMT, KILEMT
173	. GLOBL	DIVSOR
174	. GLOBL	SIZEMT, CSIMS4, MNTARG, HUPARG, R50TT
175	. GLOBL	KMNNAM, CCLNAM, CMDCCCL, CHKTTD
176	. GLOBL	OCTPRT, CSIMS3, OVLYEQ
177	. GLOBL	PRTR50, ALFN, R50DK
178	. GLOBL	DETARG, R50MON

```

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$TECO = 1 ; USE TECO AS DEFAULT EDITOR
26     000002 C$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
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 /JANFEBMARAPRMAYJUNJULAUGSEPOCTNOVDEC/
   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     ABTMSG: .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

```

```

;-----
; 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, CMDTIM
39	000172	CMDDEF MO*UNT, CMDMNT
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, CMDCCI
57	000302	CMDDEF ME*MORY, CMDMEM

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	NOAC*CESS, CMDNAC
70 000366	CMDDEF	US*E, CMDUSE
71 000372	CMDDEF	INI*TIALIZE, CMDINI
72 000376	CMDDEF	SQ*UEEZE, CMDSQZ
73 000402	CMDDEF	LIB*RARY, CMDCCCL
74 000406	CMDDEF	TE*CO, CMDCCCL
75 000412	CMDDEF	UC*L, UCLCMD
76 000416	CMDDEF	MU*NG, CMDCCCL
77 000422	CMDDEF	CL*OSE, RDCMD
78 000426	CMDDEF	GT, RDCMD
79 000432	CMDDEF	INS*TALL, CMDINS
80 000436	CMDDEF	LO*AD, RDCMD
81 000442	CMDDEF	SU*SPEND, CMDSPN
82 000446	CMDDEF	UNL*OAD, RDCMD
83 000452	CMDDEF	BO*OT, CMDBOT
84 000456	CMDDEF	\$ST*OP, CMDBOT
85 000462	CMDDEF	\$SH*UTDOWN, CMDSHT
86 000466	TBLEND	

Table of command keywords

1						
2	000234	000000	000000	000000	ASDEX: .WORD	0,0,0,0
	000242	000000				
3	000244	015270	012445		DKCOM: .RAD50	/DK COM/
4	000250	075250	073376		SYSAV: .RAD50	/SY SAV/
5	000254	015270	073376		DKSAV: .RAD50	/DK SAV/
6	000260	100040			R50TT: .RAD50	/TT /
7	000262	015270			R50DK: .RAD50	/DK /
8	000264	075250			R50SY: .RAD50	/SY /
9	000266	046537			R50LOG: .RAD50	/LOG/
10	000270	016003			R50DSK: .RAD50	/DSK/
11	000272	042614			R50KED: .RAD50	/KED/
12	000274	045130			R50K52: .RAD50	/K52/
13	000276	042640			R50KEX: .RAD50	/KEX/
14	000300	062570			R50PIP: .RAD50	/PIP/
15	000302	015172			R50DIR: .RAD50	/DIR/
16	000304	016130			R50DUP: .RAD50	/DUP/
17	000306	035164			R50IND: .RAD50	/IND/
18	000310	100020			R50TSX: .RAD50	/TSX/
19	000312	101704			R50UCL: .RAD50	/UCL/
20	000314	105372			R50VIR: .RAD50	/VIR/
21	000316	045640			R50LD: .RAD50	/LD /
22	000320	045676			R50LD0: .RAD50	/LD0/
23	000322	045705			R50LD7: .RAD50	/LD7/
24	000324	012445			R50COM: .RAD50	/COM/
25	000326	073376			R50SAV: .RAD50	/SAV/
26	000330	004056			ALDEX: .RAD50	/ALN/
27	000332	015270	004051	026760	ALDBLK: .RAD50	/DK ALIGN ALN/
	000340	004056				
28	000342	075250	003273	021113	AUTHFN: .RAD50	/SY ACCESSTSX/
	000350	100020				
29	000352	075250	100003	051646	KMNNAM: .RAD50	/SY TSKMONSAV/
	000360	073376				
30	000362	075250	011504	000000	CCLNAM: .RAD50	/SY CCL SAV/
	000370	073376				
31	000372	075250	035164	000000	INDNAM: .RAD50	/SY IND SAV/
	000400	073376				
32	000402	075250	045640	000000	LDNAM: .RAD50	/SY LD SYS/
	000410	075273				
33	000412	043327	053600		R50KMN: .RAD50	/KMON /
34	000416	075250	114730	000000	HNBUF: .RAD50	/SY XXX TSX/
	000424	100020				
35	000426	054730			R50NO: .RAD50	/NO /
36						
37	000430	000000			DEVUNT: .WORD	0
38	000432	000000			ASKLNM: .WORD	0
39	000434	000000			INDRFL: .WORD	0
40	000436	000000			QUOTFL: .WORD	0
41	000440				FILNAM: .BLKW	5
42	000452				XAREA: .BLKW	8
43	000472				R50BUF: .BLKW	2
44	000476	000000			CPUAH: .WORD	0
45	000500	000000			CPUAL: .WORD	0
46	000502	000000			HANIDX: .WORD	0
47	000504	000000	000000	000000	MNTDEV: .WORD	0,0,0,0
	000512	000000				
48	000514	000000			CMDEND: .WORD	0 ;Pointer past end of current command string

Table of command keywords

49	000516	000000	000000	000000	ALCDEV: .WORD	0,0,0,0 ; Name of dev being allocated or deallocated
	000524	000000				

Table of command keywords

1				;
2				; Emt arg block to run a program.
3				;
4	000526	000	126	RUNEMT: .BYTE 0,126
5	000530	000000		.WORD 0
6				;
7				; Emt arg block to start a spooler.
8				;
9	000532	000	126	SPGEMT: .BYTE 0,126
10	000534	000001		.WORD 1
11	000536	000000		.WORD 0
12				;
13				; Emt arg block to control detached jobs.
14				;
15	000540	000	132	DETARG: .BYTE 0,132
16	000542	000000		.WORD 0
17				;
18				; Emt argument block to set the size of the job
19				;
20	000544	000	141	SIZEMT: .BYTE 0,141
21	000546	000000		SIZVAL: .WORD 0
22				;
23				; Emt arg block to mount a file structure.
24				;
25	000550	000	134	MNTARG: .BYTE 0,134
26	000552	000504'		.WORD MNTDEV
27				;
28				; Emt arg block to dismount a file structure.
29				;
30	000554	000	135	DMTARG: .BYTE 0,135
31	000556	000504'		.WORD MNTDEV
32				;
33				; Emt arg block to copy the file status and privileges from a parent job
34				;
35	000560	001	160	CPYCXT: .BYTE 1,160
36	000562	000000		.WORD 0
37				;
38				; Emt arg block to send a message to a line.
39				;
40	000564	000	127	YELEMT: .BYTE 0,127
41	000566	000000		.WORD 0
42	000570	000572'		.WORD MSGBUF
43	000572			MSGBUF: .BLKB 84.
44	000716			MSGEND:
45				;
46				; Emt to allocate a device
47				;
48	000716	000	156	ALDEMT: .BYTE 0,156
49	000720	000516'		.WORD ALCDEV ;Pointer to device spec
50				;
51				; Emt to deallocate a device
52				;
53	000722	001	156	DLCEMT: .BYTE 1,156
54	000724	000516'		.WORD ALCDEV
55				;
56				; Emt to determine if a device is allocated to another user
57				;

58	000726	002	156	TALEMT: .BYTE	2,156	
59	000730	000516'			.WORD	ALCDEV
60						
61				; Emt argument block to read a cached device descriptor block into CDBUF		
62						
63	000732	000	126	CDGEMT: .BYTE	0,126	
64	000734	000010			.WORD	10 ; Subfunction code (peek)
65	000736	000000		CDGADR: .WORD	0	; Address of block within kernel
66	000740	0000000			.WORD	CD\$\$SZ ; Amt of data to get
67	000742	003362'			.WORD	CDBUF ; Destination buffer
68						
69				; Emt argument block to copy a cached device descriptor block from CDBUF		
70				; back into the kernel data area.		
71						
72	000744	000	126	CDPEMT: .BYTE	0,126	
73	000746	000011			.WORD	11 ; Subfunction code (poke)
74	000750	000000		CDPADR: .WORD	0	; Address of block within kernel
75	000752	0000000			.WORD	CD\$\$SZ
76	000754	003362'			.WORD	CDBUF ; Destination buffer
77						
78				; Emt argument block to move INSTALL entry from kernel into IIBUF		
79						
80	000756	000	126	INGEMT: .BYTE	0,126	
81	000760	000010			.WORD	10 ; Subfunction code (peek)
82	000762	000000		INGADR: .WORD	0	; Address of block within kernel
83	000764	0000000			.WORD	II\$\$SZ ; Amt of data to get
84	000766	000060'			.WORD	IIBUF ; Destination buffer
85						
86				; Emt argument block to store INSTALL entry into kernel data		
87						
88	000770	000	126	INPEMT: .BYTE	0,126	
89	000772	000011			.WORD	11 ; Subfunction code (poke)
90	000774	000000		INPADR: .WORD	0	; Address of block within kernel
91	000776	0000000			.WORD	II\$\$SZ ; Amt of data to move
92	001000	000060'			.WORD	IIBUF ; Source buffer
93						
94				; Emt arg block to log off the current job.		
95						
96	001002	000	126	OFFEMT: .BYTE	0,126	
97	001004	000002			.WORD	2
98	001006	000000			.WORD	0 ; 0==>Default time to drop DTR
99						
100				; Emt to update running copy of a handler.		
101						
102	001010	000	126	HUPARG: .BYTE	0,126	
103	001012	000003			.WORD	3
104	001014				.BLKW	2
105						
106				; Emt to reboot the system.		
107						
108	001020	000	126	BOTEMT: .BYTE	0,126	
109	001022	000004			.WORD	4
110	001024	004340'			.WORD	START
111						
112				; Emt to force logoff of a job.		
113						
114	001026	000	126	KILEMT: .BYTE	0,126	

115	001030	000005		.WORD	5	
116	001032	000000		.WORD	0	
117						
118				; Emt to suspend execution of a job		
119						
120	001034	000	126	SJEMT:	.BYTE	0,126
121	001036	000012		.WORD	12	
122	001040	000000		.WORD	0	
123						
124				; Emt to resume execution of a job		
125						
126	001042	001	126	RJEMT:	.BYTE	1,126
127	001044	000012		.WORD	12	
128	001046	000000		.WORD	0	
129						
130				; EMT arg block to set "hold" mode for a spooled log file		
131						
132	001050	0000	151	SPLHLA:	.BYTE	LOGCHN,151
133	001052	000000		.WORD	0	
134	001054	000001		.WORD	1	
135						
136				; EMT arg block to copy data from the context block of another job		
137				; into our context block.		
138						
139	001056	000	126	EMCXCP:	.BYTE	0,126
140	001060	000014		.WORD	14	
141	001062	000000		.WORD	0	; # of job we are copying from
142	001064	000000		.WORD	0	; Address of item being copied
143	001066	000000		.WORD	0	; Number of bytes to copy
144						
145				; EMT arg block to set the job execution priority		
146						
147	001070	000	150	PRIEMT:	.BYTE	0,150
148	001072	000000		.WORD	0	
149						
150				; EMT arg block to create a window for this job		
151						
152	001074	000	161	MAKWIN:	.BYTE	0,161
153	001076	001	001	.BYTE	1,1	
154	001100	120	020	.BYTE	80,16.	
155	001102	001	000	.BYTE	1,0	
156	001104	000000		.WORD	0	
157						
158				; EMT arg block to select window 1 as the current window		
159						
160	001106	001	161	MAPWIN:	.BYTE	1,161
161	001110	001	000	.BYTE	1,0	
162						
163				; EMT arg block to delete all temporary display windows for the job		
164						
165	001112	002	161	DELWIN:	.BYTE	2,161
166	001114	000	000	.BYTE	0,0	
167						
168				; Emt arg block to initiate performance monitoring.		
169						
170	001116	000	136	MONAR1:	.BYTE	0,136
171	001120	000000		MNBASE:	.WORD	0

Table of command keywords

172	001122	000000			MNTOP: .WORD 0
173	001124	000000			MNBPC: .WORD 0
174	001126	000000			MNFLGS: .WORD 0
175					;
176					; Emt arg block to start performance monitoring.
177					;
178	001130	001	136		MONAR2: .BYTE 1,136
179					;
180					; Emt arg block to broadcast job status change info to monitoring jobs
181					;
182	001132	002	157		GENMON: .BYTE 2,157
183	001134	000000			.WORD 0 ;Store value to broadcast here

```

1 001136          CMDBUF: . BLKW  66.
2 001342          CBFEND:                      ; END OF CMDBUF
3 001342          KEYBUF: . BLKW   8.           ; Used by SEARCH to hold keywords
4 001362          KEYEND:                      ; End of KEYBUF
5 001362          BLKO: . BLKW  512.
6          001362' KCSIBF = BLKO              ; OVERLAY WITH BLKO
7          001362' INBUF  = BLKO              ; OVERLAY WITH BLKO
8 003362          CDBUF: . BLKW  10.           ; Should be at least as large as CD$$SZ
9          ; TSKMON STACK
10 003406          . BLKW  100.
11          003716' KMSTK =
12          ;
13          ; BYTE DATA
14          ;
15          ; PRINT BUFFER
16 003716          PRTBUF: . BLKB  256.
17 004316          200 PBUFND: . BYTE  200
18 004317          000 NOFLAG: . BYTE   0
19 004320          000 DOLRAT: . BYTE   0           ; INDICATES #@ SCANNED
20 004321          000 DOTAT: . BYTE   0           ; INDICATES #@ SCANNED
21 004322          000 COLEQL: . BYTE   0           ; INDICATES := SCANNED
22 004323          000 NOUCL: . BYTE   0           ; Non-zero==>Don't call UCL for this command
23 004324          123      131      072 SYTXT: . ASCIIZ /SY: /
24          004327          000
25          004330          123      131      072 SYINTX: . ASCIIZ /SY: IND /
26          004333          111      116      104
27          004336          040      000
          ;
          . EVEN
          ;

```

Job Initialization

```

1          .SBTTL  Job Initialization
2          ;-----
3          ; ENTER TSKMON AT KMNSTR.
4          ;
5 004340 000240          START:  NOP          ;DEBUGGING BPT INSERTION POINT
6 004342 116701 000000G      MOVB      CORUSR,R1      ;GET USER INDEX #
7          ;
8          ; IF THE SYSTEM-WIDE ONCE-ONLY INITIALIZATION HASN'T BEEN DONE YET,
9          ; THEN DISABLE THE JOB SCHEDULER TO ENSURE THIS JOB GETS A CHANCE TO DO IT
10         ; BEFORE ANYBODY ELSE CAN RUN.
11         ;
12 004346 105767 000000G      TSTB      KMONCE          ;Has once only initialization been done yet?
13 004352 001402          BEQ      40$          ;Br if it has already been done
14 004354 110167 000000G      MOVB      R1,EXCJOB      ;If not, disable scheduling of any other jobs
15         ;
16         ; INITIALIZATION PERFORMED FOR LINE THE FIRST TIME
17         ; TSKMON IS ENTERED.
18         ;
19 004360 032761 000000G 000000G 40$:  BIT      ##KINIT,LSW(R1) ;IS THIS 1ST ENTRY FOR THIS LINE?
20 004366 001402          BEQ      10$          ;BR IF YES
21 004370 000167 001060      JMP      KMNOT1
22         ;
23         ; Make sure TSGEN hasn't been modified without relinking TSKMON.
24         ;
25 004374 026727 000000G 123456 10$:  CMP      GENTOP,#123456 ;DOES THIS LOOK LIKE THE TOP OF TSGEN?
26 004402 001406          BEQ      22$          ;BR IF OK
27 004404          FERR      #REGEN          ;NEED TO RELINK TSKMON
28 004420 016761 000000G 000000G 22$:  MOV      MINTIM,LCONTM(R1);SET JOB LOG-ON TIME
29 004426 042761 000000G 000000G      BIC      ##CFKIL,LSW6(R1) ;IN CASE OF ^C^C DURING GETSYP
30 004434 042761 000000G 000000G      BIC      ##CTRLC,LSW(R1) ;DO NOT DO CONTROL-C ABORT
31         ;
32         ; Initially set the job name to all blanks
33         ;
34 004442 010105          MOV      R1,R5          ;GET JOB INDEX #
35 004444 070527 000006      MUL      #6,R5          ;* 12. BYTES PER ENTRY
36 004450 062705 000000G      ADD      #LUNAME,R5      ;POINT TO USER NAME ENTRY FOR THIS LINE
37 004454 012704 000014      MOV      #12,R4
38 004460 112725 000040 24$:  MOVB      #'',(R5)+      ;SET USER NAME TO BLANKS
39 004464 077403          SOB      R4,24$
40         ;
41         ; Initialize cells in job context block
42         ;
43 004466 012767 000000G 000000G      MOV      #CFSTK,CFSP      ;COMMAND FILE STACK
44 004474 016700 000000G      MOV      DFJMEM,R0      ;GET MAX # KB JOB IS ALLOWED TO USE
45 004500 072027 000012      ASH      #10,R0          ;CONVERT TO ADDRESS
46 004504 001002          BNE      25$          ;BR IF DIDN'T OVERFLOW 64KB
47 004506 012700 177774      MOV      #177774,R0      ;SET MAX MEMORY AS 64KB
48 004512 010067 000000G 25$:  MOV      R0,MAXMEM      ;SET AS TOP OF MEMORY FOR JOB
49 004516 012767 000000G 000000G      MOV      #NAMTOP,UHIMEM ;HIGHEST LEGAL ADDRESS FOR JOB
50 004524 012767 000004G 000000G      MOV      #SPUBUF+4,SXBPNT;SPOOL FILE BUFFER
51 004532 112767 000000G 000000G      MOVB      #MAXPRI,MXJPRI ;SET MAXIMUM JOB PRIORITY
52 004540 112767 000056 000000G      MOVB      #',,KMPRMT      ;SET UP DEFAULT KMON PROMPT (". ")
53 004546 112767 000200 000001G      MOVB      #200,KMPRMT+1 ;TERMINATE PROMPT STRING
54 004554 012703 000000G      MOV      #STDNAM,R3      ;SET UP CURRENT (DEFAULT) SPOOL FORM NAME
55 004560 012704 000000G      MOV      #UFORM,R4
56 004564 012700 000006      MOV      #6,R0
57 004570 112324 3$:  MOVB      (R3)+,(R4)+

```

```

58 004572 077002          SOB      R0,3$
59 004574 112767 000000G 000000G      MOVB      #SC$ERR,ERRSEV ;ERROR ABORT SEVERITY LEVEL
60
61 ; Start job with all privileges granted and no access restrictions
62 ; Privileges may be changed by running LOGON or by use of
63 ; the SET PROCESS/AUTH/PRIV=(list) command.
64
65 004602 005067 000000G          CLR      RESDEV      ;Say no access restrictions
66 004606 012767 000000G 000000G      MOV      #OKFILE,OKFAND ;Init last access entry ptr
67 004614 012767 000000G 000000G      MOV      #OKFEND,OKFNND ;Init first noaccess entry ptr
68 004622 012700 000000G          MOV      #PVNPW,R0      ;Get # words in each privilege vector
69 004626 005002          CLR      R2      ;Set index for 1st word
70 004630 012703 177777          MOV      #177777,R3      ;Get all privilege flags turned on
71 004634 010362 000000G          77$: MOV      R3,PRIVA0(R2) ;Authorized privileges
72 004640 010362 000000G          MOV      R3,PRIVS0(R2) ;Set privileges
73 004644 010362 000000G          MOV      R3,PRIVF0(R2) ;Command file privileges
74 004650 010362 000000G          MOV      R3,PRIVC0(R2) ;Current (program) privileges
75 004654 062702 000002          ADD      #2,R2      ;Increment vector index
76 004660 077013          SOB      R0,77$      ;Set all privileges
77 004662 020127 000000G          CMP      R1,#LSTPL      ;Is this a primary line?
78 004666 003016          BGT      32$      ;Br if not
79 004670 032761 000000G 000000G      BIT      ##NOVLN,1LSW2(R1);Disallow subprocess usage?
80 004676 001412          BEQ      32$      ;Br if don't need to disallow subprocesses
81 004700 012700 000000G          MOV      #P2$VIR,R0      ;Get subprocess privilege flag
82 004704 040067 000000G          BIC      R0,PRIVA2      ;Remove subprocess privilege
83 004710 040067 000000G          BIC      R0,PRIVS2
84 004714 040067 000000G          BIC      R0,PRIVF0
85 004720 040067 000000G          BIC      R0,PRIVC0
86 004724 004767 000000G          32$: CALL      RSTPRV      ;Setup some other privilege flags
87
88 ; Open spool file channel
89
90 004730 105767 000000G          TSTB      NSPLDV      ;ARE THERE ANY SPOOLED DEVICES?
91 004734 001410          BEQ      28$      ;BR IF NOT
92 004736          .REOPEN #XAREA,#USPLCH,#SPLCHN ;OPEN CHANNEL FOR WRITES TO SPOOL FILE
93
94 ; Initialize TSXUCL data base for this job
95 ; (Say that there are no user-defined commands yet)
96
97 004756 016705 000000G          28$: MOV      UCLBLK,R5      ;Get # blocks in file per job
98 004762 001446          BEQ      2$      ;Br if TSXUCL data file not needed
99 004764          .LOOKUP #XAREA,#1,#UCLDAT ;Lookup TSXUCL data file
100 005004 103435          BCS      2$      ;Br if not there
101 005006 010100          MOV      R1,R0      ;Get job index number
102 005010 006200          ASR      R0      ;Convert to job number
103 005012 005300          DEC      R0      ;Make first job # 0
104 005014 070500          MUL      R0,R5      ;Compute block number of data for this job
105 005016 012702 001362'          MOV      #BLK0,R2      ;Point to data buffer
106 005022 012762 177777 000000G      MOV      #-1,UC$NDC(R2) ;Set flag saying TSXUCL should init data
107 005030 016762 000000G 000000G      MOV      VUCLMC,UC$MDC(R2) ;Set maximum allowed number of commands
108 005036          .WRITW #XAREA,#1,R2,#256,R5 ;Write out data for job
109 005072          .CLOSE #1      ;Close TSXUCL file
110
111 ; Set default editor
112
113 005100 126727 000000G 000000G      2$: CMPB      VEDIT,#EDIT      ;IS DEFAULT EDITOR EDIT?
114 005106 001413          BEQ      7$      ;BR IF YES

```

```

115 005110 126727 000000G 000000G      CMPB  VEDIT,#TECO      ; IS DEFAULT EDITOR TECO?
116 005116 001004      BNE  21$      ; BR IF NOT
117 005120 052761 000000G 000000G      BIS  21$      ; SET DEFAULT EDITOR FOR JOB
118 005126 000403      BR  7$
119 005130 052761 000000G 000000G 21$:  BIS  7$      ; DEFAULT EDITOR MUST BE KED
120      ;
121      ; Set default UCL FIRST/MIDDLE/LAST
122      ;
123 005136 126727 000000G 000001 7$:  CMPB  VUCLOR,#1      ; SET UCL FIRST?
124 005144 001003      BNE  30$      ; BR IF NOT
125 005146 052761 000000G 000000G      BIS  30$      ; SET UCL MIDDLE?
126 005154 126727 000000G 000002 30$:  CMPB  VUCLOR,#2      ; SET UCL MIDDLE?
127 005162 001003      BNE  29$      ; BR IF NOT
128 005164 052761 000000G 000000G      BIS  29$      ; SET UCL LAST?
129 005172 126727 000000G 000003 29$:  CMPB  VUCLOR,#3      ; SET UCL LAST?
130 005200 001003      BNE  27$      ; BR IF NOT
131 005202 052761 000000G 000000G      BIS  27$      ; SET UCL LAST?
132      ;
133      ; Set default SL mode
134      ;
135 005210 052761 000000G 000000G 27$:  BIS  27$      ; Default SL substitution on
136      ;
137      ; Set default wildcard flag
138      ;
139 005216 005727 000000G      31$:  TST   #WILDFL      ; EXPLICIT OR IMPLICIT WILDCARDS?
140 005222 001403      BEQ   8$      ; BR IF EXPLICIT
141 005224 052761 000000G 000000G      BIS  8$      ; SET IMPLICIT WILDCARD FLAG
142 005232 020127 000000G      8$:  CMP   R1,#LSTD      ; REAL OR VIRTUAL LINE?
143 005236 003022      BGT   12$      ; BR IF VIRTUAL
144 005240 020127 000000G      CMP   R1,#LSTPL      ; REAL OR DETACHED?
145 005244 003021      BGT   1$      ; BR DETACHED
146      ;
147      ; See if we need to get a system password (primary line only).
148      ;
149 005246 032761 000000G 000000G      BIT   #SYSPS,LSW2(R1) ; Do we need to accept a system password?
150 005254 001402      BEQ   13$      ; Br if not
151 005256 004767 000000G      CALL  GETSYP      ; Accept system password
152      ;
153      ; Mount SY device for this job (primary line only).
154      ;
155 005262 016767 172776 173214 13$:  MOV   R50SY,MNTDEV      ; SET SY AS DEVICE TO MOUNT
156 005270 012700 000550'      MOV   #MNTARG,R0      ; MOUNT SY
157 005274 104375      EMT   375
158      ;
159      ; Terminal dependent initialization for primary lines.
160      ;
161 005276 004767 004714      CALL  TRMINI      ; Do terminal-dependent initialization
162 005302 000402      BR  1$
163      ;
164      ; Initialization for subprocesses only.
165      ; Print n> to show which subprocess we are switching to.
166      ;
167 005304 004767 005170      12$:  CALL  VIRINI      ; Do virtual line initialization
168      ;
169      ; Do some initialization for detached jobs
170      ;
171 005310 032761 000000G 000000G 1$:  BIT   #DETCH,LSW(R1) ; Is this a detached job?

```

Job Initialization

```

172 005316 001403          BEQ      GRTINI          ;Br if not
173 005320 012761 000000C 000000G          MOV      #$ECHO!$LC!$DEFER,LSW2(R1) ;Init control flags
174                                     ;
175                                     ; Perform code which is executed only once each time TSX is started
176                                     ; and Print TSX greeting message
177                                     ;
178 005326 105767 000000G          GRTINI: TSTB      KMONCE          ;Has once only code been done yet?
179 005332 001402          BEQ      3$              ;Br if it has already been done
180 005334 004767 000000G          CALL      KMINIT          ;If not, call once only code
181 005340 004767 005774          3$:   CALL      PRTGRT          ;Print the TSX-Plus logon greeting
182 005344 016161 000000G 000000G          MOV      LSW2(R1),LSW2S(R1) ;SAVE IN CASE OF CTRL-C REENTRY
183                                     ;
184                                     ; If this job has a parent job, copy file context and privilege from parent
185                                     ;
186 005352 004767 005254          CALL      CPYPRN          ;Copy info from parent job
187                                     ;
188                                     ; Set flag saying job initialization has been done
189                                     ;
190 005356 052761 000000G 000000G          BIS      #$KINIT,LSW(R1) ;SAY LINE HAS BEEN INITIALIZED
191                                     ;
192                                     ; See if this line has an associated start-up command file.
193                                     ;
194 005364 020127 000000G          CMP      R1,#LSTD L          ;Is this a virtual line?
195 005370 101004          BHI      1$              ;Br if yes
196 005372 016102 000000G          MOV      LSUCF(R1),R2          ;IS THERE A START-UP COMMAND FILE?
197 005376 001407          BEQ      61$             ;BR IF NOT
198 005400 000402          BR       2$              ;
199 005402 012702 000000G          1$:   MOV      #SBPSUF,R2          ;Point to job context cell with file name
200 005406 105712          2$:   TSTB      (R2)          ;IS FILE NAME NULL?
201 005410 001402          BEQ      61$             ;BR IF YES
202 005412 004767 006324          CALL      SETSUF          ;Set up start-up command file for execution
203                                     ;
204                                     ; Broadcast status message to monitoring jobs telling them that
205                                     ; this job logged on.
206                                     ;
207 005416 012767 000000G 173510          61$:  MOV      #JS$ON,GENMON+2 ;Set logged-on status code
208 005424 012700 001132'          MOV      #GENMON,R0          ;Point to EMT argument block
209 005430 104375          EMT      375              ;Tell monitoring jobs that we are initiated
210 005432 005761 000000G          TST      LPARNT(R1)          ;Do we have a parent job?
211 005436 001406          BEQ      KMNOT1          ;Br if not
212 005440 012767 000000G 173466          MOV      #JS$LOG,GENMON+2 ;Set status code saying we have logged on
213 005446 012700 001132'          MOV      #GENMON,R0          ;Point to EMT argument block
214 005452 104375          EMT      375              ;Broadcast status code that job logged on

```

Entry to KMON

```

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 005454 016761 172732 000000G KMNOT1: MOV      R50KMN,LPRG1(R1); SET "KMON" AS RUNNING PROGRAM NAME
7 005462 016761 172726 000000G          MOV      R50KMN+2,LPRG2(R1)
8 005470 005067 000000G          CLR      UTRPAD          ; CLEAR USER TRAP CONTROL
9 005474 005067 000000G          CLR      UFPTRP          ; RESET FLOATING POINT TRAP CONTROL
10 005500 005061 000000G          CLR      LSCCA(R1)        ; RESET .SCCA TRAP CONTROL
11 005504 105067 000000G          CLRB     SERFLG          ; DO .HERR
12 005510 105067 000000G          CLRB     RUNARG          ; No argument string from RUN command
13 005514 005767 000000G          TST      UCHAN          ; DID USER DO A .CDFN?
14 005520 001404          BEQ      4$          ; BR IF NOT
15 005522 005067 000000G          CLR      UCHAN          ; UNDO THE .CDFN
16 005526 004767 000000G          CALL     PRGALL          ; PURGE ALL CHANNELS
17 005532 005061 000000G          4$: CLR      LTSCMD(R1)    ; NO PENDING SPECIAL COMMAND
18 005536 042761 000000G 000000G          BIC      $$SETRN,LSW9(R1); Say SETUP is no longer running
19 005544          .CLOSE    #RUNCHN          ; CLOSE PROGRAM SAV FILE
20 005552 004767 000000G          CALL     RSTPRV          ; Reset job privileges
21 005556 032761 000000G 000000G          BIT      $$DUPRN,LSW6(R1); WAS DUP THE PROGRAM THAT EXITED?
22 005564 001407          BEQ      7$          ; BR IF NOT
23 005566 042761 000000G 000000G          BIC      $$DUPRN,LSW6(R1); SAY DUP IS NO LONGER RUNNING
24 005574 004767 000000G          CALL     PRGALL          ; PURGE ALL CHANNELS
25 005600 004767 000000G          CALL     LDCLN          ; DO "LD CLEAN" OPERATION
26 005604 032761 000000G 000000G 7$: BIT      $$DOOFF,LSW(R1); SHOULD WE LOG USER OFF?
27 005612 001402          BEQ      6$          ; BRANCH IF NOT
28 005614 000167 000000G          JMP      CMDOFF          ; FORCE LOGOFF
29          ; CHECK FOR SYSTEM ABORT
30 005620 116702 000000G          6$: MOVB     ABRTCD,R2          ; WAS USER ABORTED?
31 005624 001467          BEQ      NOABRT          ; BRANCH IF NOT
32          ;
33          ; USER WAS ABORTED -- PUT OUT ABORT MESSAGE.
34          ;
35 005626 112767 000010 000000G          MOVB     #10,UERSEV          ; SAY ERROR SEVERITY LEVEL = SEVERE
36 005634 006302          ASL      R2          ; CVT ERROR CODE TO WORD INDEX
37 005636 016203 000176'          MOV      ABTMSG(R2),R3          ; GET ADDR OF ERROR MESSAGE
38 005642          .PRINT    #MONTXT
39 005650          .PRINT    R3
40          ; Print name of file that caused the error
41 005654 005767 000000G          TST      ERRSPC          ; Do we have a file spec to print?
42 005660 001414          BEQ      1$          ; Br if not
43 005662          .PRINT    #FILERM          ; Print heading message
44 005670 012703 001362'          MOV      #BLKO,R3          ; Point to buffer where result is to be stored
45 005674 012704 000000G          MOV      #ERRSPC,R4          ; Point to RAD50 file spec
46 005700 004767 000000G          CALL     EDTFIL          ; Convert file spec to ascii
47 005704          .PRINT    #BLKO          ; Print the file spec
48          ; PUT OUT LOCATION OF ABORTED INSTRUCTION.
49 005712          1$: .PRINT    #LOCMSG
50 005720 016702 000000G          MOV      ABRTAD,R2          ; GET ADDR OF ABORT INST
51 005724 004767 000000G          CALL     OCTPRT          ; PRINT THE OCTAL VALUE
52 005730 020227 120000          CMP      R2,#120000          ; IN OVERLAY REGION?
53 005734 000407          BR      10$          ; *** skip overlay this version ***
54          ;
55 005736          .PRINT    #OVLREQ          ; PRINT " OVERLAY = "
56 005744 016700 000000G          MOV      ABRTQV,R0          ; GET OVERLAY NAME
57 005750 004767 000000G          CALL     PRTR50          ; AND DISPLAY IT

```

Entry to KMON

58 005754			10\$:	. PRINT	#CRLF	
59 005762	005067	000000G		CLR	ABRTAD	; CLEAN ABORT ADDR
60 005766	105067	000000G		CLRB	ABRTCD	; AND CODE
61 005772	042737	004440 000000G		BIC	#004440, @#JSWLOC	; RELEASE CHAIN AREA, PASLIN, AND SCHAIN
62 006000	105067	000000G		CLRB	CINFLG	; KILL ANY CHAIN REQUEST

Entry to KMON

```

1
2      ; User did not abort. See if this is a .CHAIN request
3
4 006004 005067 000000G NOABRT: CLR ERRSPC ;CLEAR ANY ERROR FILE SPEC
5 006010 032761 000000C 000000G BIT ##CFKIL!$CFABT,LSW6(R1);REQUEST TO ABORT ALL COM FILES?
6 006016 001405 BEQ 1$ ;BR IF NOT
7 006020 105067 000000G CLRB CINFLG ;CLEAR ANY CHAIN REQUEST
8 006024 042737 000000C 000000G BIC #PASLIN!$CHAIN,@#JSWLOC ;KILL ANY PASSED COMMAND
9 006032 105767 000000G 1$: TSTB CINFLG ;IS THIS .CHAIN REQUEST?
10 006036 001402 BEQ NOCIN ;BRANCH IF NOT
11 006040 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 006044 012700 001112' NOCIN: MOV #DELWIN,RO ;Point to EMT argument block
18 006050 104375 EMT 375 ;Delete all temporary windows for job
19
20      ; Clean up various cells
21
22 006052 012761 000040 000000G MOV #BLANK,LRBFIL(R1);USE SPACE FOR RUBOUT FOR KMON
23 006060 016161 000000G 000000G MOV LSW2S(R1),LSW2(R1);RESET STATUS OF LSW2
24 006066 042761 000000G 000000G BIC #KL4CLR,LSW4(R1)
25 006074 042761 000000G 000000G BIC ##VTESE,LSW5(R1);TURN OFF VT52 ESC-LETTER ACTIVATION
26 006102 042761 000000C 000000G BIC ##DEBUG!$RNJOP,LSW9(R1);Not debugger, I/O page
27 006110 005061 000000G CLR LNSPAC(R1) ;NO SPECIAL ACTIVATION CHARS
28 006114 005061 000000G CLR LRDTIM(R1) ;CLEAR TT READ TIME-OUT VALUE
29 006120 005061 000000G CLR LAFSIZ(R1) ;CLEAR FIELD WIDTH ACTIVATION
30 006124 005061 000000G CLR LFWLIM(R1) ;CLEAR FIELD WIDTH LIMIT
31 006130 042761 000000G 000000G BIC #KL3CLR,LSW3(R1);CLEAR MISC BITS IN LSW3
32 006136 042761 000000G 000000G BIC ##CFDCC,LSW4(R1);CLEAR DEFERRED-CTRL-C FLAG
33 006144 042761 000000C 000000G BIC #<$NOWTT!$CHACT>,LSW5(R1);TURN OFF NO WAIT AND SINGLE CHAR ACT
34 006152 042761 000000G 000000G BIC ##NOINT,LSW7(R1);Reset non-interactive run switch
35 006160 042761 000000G 000000G BIC ##NTGCC,LSW9(R1);Clear non-terminating .GTLIN ctrl-C flag
36 006166 032761 000000G 000000G BIT ##SNWTT,LSW6(R1);DID USER DO "SET TT NOWAIT"?
37 006174 001403 BEQ 3$ ;BR IF NOT
38 006176 052761 000000G 000000G BIS ##NOWTT,LSW5(R1);SET NO-WAIT FLAG
39 006204 016700 000000G 3$: MOV CFSPND,RO ;WAS A COMMAND FILE SUSPENDED?
40 006210 001404 BEQ 4$ ;BR IF NOT
41 006212 004767 000000G CALL CFSTRT ;Restart it
42 006216 005067 000000G CLR CFSPND
43
44      ; Check if program specified an error severity level.
45
46 006222 156767 000000G 000000G 4$: BISB UERSEV,INDERR ;SAVE ERROR STATUS FOR IND
47 006230 126767 000000G 000000G CMPB UERSEV,ERRSEV ;DID PROGRAM SPECIFY ERROR SEVERITY?
48 006236 103413 BLD 1$ ;BR IF OK
49 006240 004767 000000G CALL ABRTCF ;SEVER ERROR -- ABORT COMMAND FILE
50 006244 042761 000000G 000000G BIC ##CCLRN,LSW5(R1);SAY CCL NOT RUNNING
51 006252 032761 000000G 000000G BIT ##INDAB,LSW7(R1);DOES HE WANT TO ABORT IND COMMAND FILES?
52 006260 001402 BEQ 1$ ;BR IF NOT
53 006262 004767 000000G CALL INDABT ;Abort IND execution
54 006266 105067 000000G 1$: CLRB UERSEV ;CLEAR ERROR STATUS
55
56      ; Purge all of user's channels
57

```

Entry to KMON

```

58 006272 004767 000000G          CALL PRGALL          ;PURGE ALL CHANNELS
59 006276 032761 000000G 000000G    BIT  ##CFKIL,LSW6(R1);Should we abort IND and command files?
60 006304 001403          BEQ  2$          ;Br if not
61 006306 004767 000000G          CALL INDABT          ;Abort IND and nested command files
62 006312 000406          BR  5$
63 006314 032761 000000G 000000G 2$: BIT  ##CFABT,LSW6(R1);SHOULD WE ABORT ALL ACTIVE COMMAND FILES?
64 006322 001405          BEQ  6$          ;BR IF NOT
65 006324 004767 000000G          CALL ABRTCF          ;ABORT ALL ACTIVE COMMAND FILES
66 006330 042761 000000G 000000G 5$: BIC  ##CCLRN,LSW5(R1);STOP EXECUTION OF CCL
67                                     ;
68                                     ; Tell any jobs that are monitoring us that we just entered TSKMON
69                                     ;
70 006336 005761 000000G          6$: TST  LMONHD(R1)      ;Are we being monitored?
71 006342 001003          BNE  7$          ;Br if yes
72 006344 005767 000000G          TST  SMONHD          ;Anyone monitoring all jobs?
73 006350 001406          BEQ  CKPASL          ;Br if not
74 006352 012700 001132'          7$: MOV  #GENMON,R0      ;Point to EMT argument block
75 006356 012760 000000G 0000002    MOV  #JS$KMN,2(R0)    ;Set status code
76 006364 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 006366 032737 000000C 000000G CKPASL: BIT      #PASLIN!SCHAIN,@#JSWLOC;DID PROGRAM PASS US A COMMAND?
6 006374 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 006376 032761 000000C 000000G      BIT      #$CCLRN!$INDRN,LSW5(R1);IS CCL OR IND RUNNING?
17 006404 001006      BNE      5$      ;BR IF YES
18 006406 032737 000000G 000000G      BIT      #SCHAIN,@#JSWLOC;SHOULD BE ABORT CURRENT COMMAND FILE?
19 006414 001002      BNE      5$      ;BR IF NOT
20 006416 004767 000000G      CALL     ABRTCF      ;ABORT ALL CURRENTLY OPEN COMMAND FILES
21 006422 005767 000010G 5$:      TST      CINDAT+10      ;DID HE PASS US A NULL COMMAND FILE?
22 006426 001531      BEQ      4$      ;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 006430 032761 000000G 000000G      BIT      #$UCLRN,LSW7(R1);Is TSXUCL program running?
31 006436 001432      BEQ      15$      ;Br if not
32 006440 126727 000012G 000077      CMPB     CINDAT+12,#'?      ;Is TSXUCL throwing command back to us?
33 006446 001026      BNE      15$      ;Br if not
34 006450 012703 000013G      MOV      #CINDAT+13,R3      ;Point to chain data area (past "?" char)
35 006454 012702 001136'      MOV      #CMDBUF,R2      ;Point to command buffer
36 006460 112322      16$:      MOVB     (R3)+,(R2)+      ;Move command to command buffer
37 006462 001376      BNE      16$      ;Loop till asciz null moved
38 006464 005302      DEC      R2      ;Make R2 point to null at end of command
39 006466 010267 172022      MOV      R2,CMDEND      ;Save pointer to end of command
40 006472 042761 000000G 000000G      BIC      #$UCLRN,LSW7(R1);Say TSXUCL program no longer running
41 006500 042737 000000C 000000G      BIC      #PASLIN!SCHAIN,@#JSWLOC ;Clear command-passed flag
42 006506 052737 000000G 000000G      BIS      #LCBIT,@#JSWLOC ;Enable lower-case input
43 006514 105267 175603      INCB     NOUCL      ;Set flag saying not to call TSXUCL again
44 006520 000167 001762      JMP      IDNCMD      ;Go process the command
45
46     ; If we have pending commands in the command file buffer, compress
47     ; them to make room for new commands.
48     ; If input is coming from a real command file we do not need to
49     ; compress since we will reread buffer when we hit end of new commands.
50
51 006524 032761 000000G 000000G 15$:      BIT      #$CFOPN,LSW4(R1);IS A COMMAND FILE OPEN?
52 006532 001403      BEQ      11$      ;BR IF NOT
53 006534 012705 001000G      MOV      #CFBUF+512,R5      ;SAY ENTIRE COMMAND FILE BUFFER IS FREE
54 006540 000403      BR      12$
55 006542 004767 000000G      11$:      CALL     CFSQEZ      ;COMPRESS INFO IN CURRENT COMMAND FILE BUFFER
56 006546 010005      MOV      R0,R5      ;SAVE ADDRESS OF END OF FREE SPACE IN BUFFER
57 006550 004767 000000G      12$:      CALL     PUSHCF      ;OPEN A NEW COMMAND FILE (PUSH CURRENT ONE)

```

Entry to KMON

```

58 ;
59 ; Move command line from chain area to command file buffer
60 ;
61 006554 012703 000010G      MOV      #CINDAT+10,R3      ;POINT TO CELL WITH CHAR COUNT
62 006560 012302              MOV      (R3)+,R2      ;GET COUNT OF # CHARS IN COMMAND
63 006562 020227 000270      CMP      R2,#<1000-510>    ;CHECK LENGTH OF COMMAND STRING
64 006566 003407              BLE      6$            ;BR IF COMMAND IS SMALL ENOUGH
65 006570 042737 000000C 000000G 10$: BIC      #PASLIN!$CHAIN,@#JSWLOC ;CLEAR FLAGS BEFORE ABORT
66 006576              FABORT    #TOOLNG             ;LINE TOO LONG ERROR MESSAGE
67 006606 012704 000000G      6$:  MOV      #CFBUF,R4      ;POINT TO START OF COMMAND FILE BUFFER
68 006612 112324              2$:  MOVB     (R3)+,(R4)+    ;MOVE COMMAND TO BUFFER
69 006614 001005              BNE      1$            ;BR IF NORMAL CHARACTER
70 ; Put cr-lf in place of asciz nulls
71 006616 112764 000015 177777 MOVB     #CR,-1(R4)
72 006624 112724 000012      MOVB     #LF,(R4)+
73 006630 077210              1$:  SOB      R2,2$            ;MOVE ALL CHARACTERS
74 006632 020405              CMP      R4,R5            ;DID WE OVERFLOW BUFFER SPACE?
75 006634 101355              BHI      10$            ;BR IF YES -- COMMAND TOO LONG
76 ;
77 ; Null fill remainder of buffer
78 ;
79 006636 020405              14$:  CMP      R4,R5            ;HAVE WE REACHED THE END OF THE BUFFER?
80 006640 103002              BHIS     13$            ;BR IF YES
81 006642 105024              CLRB     (R4)+          ;NULL FILL REST OF BUFFER
82 006644 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 006646 032761 000000C 000000G 13$: BIT      #$CCLRN!$INDRN,LSW5(R1) ;IS CCL OR IND RUNNING?
88 006654 001004              BNE      8$            ;BR IF YES
89 006656 032737 000000G 000000G      BIT      #SCHAIN,@#JSWLOC ;IS THIS A SPECIAL CHAIN EXIT?
90 006664 001403              BEQ      9$            ;BR IF NOT
91 006666 052761 000000G 000000G 8$:  BIS      #$QUIET,LSW4(R1) ;SET FLAG TO SUPPRESS LISTING COMMAND LINES
92 006674 032761 000000G 000000G 9$:  BIT      #$CCLRN,LSW5(R1) ;IS THIS AN EXPANDED CCL COMMAND?
93 006702 001403              BEQ      4$            ;BR IF NOT
94 006704 052761 000000G 000000G      BIS      #$CFCCCL,LSW4(R1) ;REMEMBER THAT THIS IS A CCL COMMAND
95 ; Clear flags which say CCL or IND is running.
96 006712 042761 000000C 000000G 4$:  BIC      #$CCLRN!$INDRN,LSW5(R1) ;CCL AND IND ARE NO LONGER RUNNING
97 006720 042737 000000C 000000G      BIC      #PASLIN!$CHAIN,@#JSWLOC ;CLEAR COMMAND-PASSED FLAGS
98 006726 042761 000000C 000000G      BIC      #$UCLRN!$UKMRN,LSW7(R1) ;SAY TSXUCL IS NO LONGER RUNNING
99 ;
100 ; See if we are exiting from a locked program
101 ;
102 006734 032761 000000G 000000G CKLK: BIT      #$PRGLK,LSW5(R1) ;IS A LOCKED PROGRAM EXITING?
103 006742 001402              BEQ      CKSF2          ;BR IF NOT
104 006744 000167 000000G      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 006750 105767 000000G      CKSF2: TSTB     SUCF2          ;Is there a pending secondary command file?
111 006754 001417              BEQ      NEWCMD          ;Br if not
112 006756 005767 000000G      TST      CFPNT          ;Are we currently in another command file?
113 006762 001014              BNE      NEWCMD          ;Br if yes -- Wait for it to finish
114 006764 042761 000000G 000000G      BIC      #$SUCF,LSW9(R1) ;Say we are finished with 1st startup file

```

Entry to KMON

115	006772	042761	000000G	000000G	BIC	##NDIN,LSW3(R1)	;Allow input to be accepted for line
116	007000	012702	000000G		MOV	#SUCF2,R2	;Point to name of secondary command file
117	007004	004767	004732		CALL	SETSU	;Set up command file
118	007010	105067	000000G		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 007014 052737 000000G 000000G NEWCMD: BIS      #LCBIT,@#JSWLOC ;ENABLE LOWER-CASE INPUT
6 007022 116701 000000G          MOV      CORUSR,R1      ;GET JOB INDEX NUMBER
7 007026 105761 000000G          TST      LCOL(R1)        ;ARE WE ALREADY AT LEFT MARGIN?
8 007032 001403          BEQ      RDCMD      ;BR IF YES -- NO CR-LF NEEDED
9 007034          .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 007042 116701 000000G RDCMD:  MOV      CORUSR,R1      ;GET JOB INDEX NUMBER
14 007046 012702 000000G          MOV      #INDSTA,R2      ;GET POINTER TO IND STATUS BYTE
15 007052 132712 000000G          BIT      #IN$ACT,@R2      ;IS IND ACTIVE?
16 007056 001406          BEQ      2$          ;BR IF NOT
17 007060 142712 000000C          BICB     #IN$ACT!IN$CMD,@R2 ;CLEAR STATUS FLAGS FOR IND
18 007064 152712 000000G          BISB     #IN$CNT,@R2      ;SAY WE ARE CONTINUING EXECUTION OF IND
19 007070 000167 002476          JMP      INDRUN          ;GO RUN IND PROGRAM
20
21          ; If user-written command interface program is active, call it to
22          ; get the next command.
23          ;
24 007074 032761 000000G 000000G 2$: BIT      #$UKMON,LSW7(R1);Are we to use user-written command program?
25 007102 001405          BEQ      3$          ;Br if not
26 007104 005767 000000G          TST      CFPNT          ;Are we getting commands from a command file?
27 007110 001002          BNE      3$          ;Br if yes -- Don't call user program till end
28 007112 000167 002162          JMP      CALUKM          ;Enter user command processor program
29
30          ; Read next command line
31          ;
32 007116          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 007136 012702 001136' PRSCMD: MOV      #CMDBUF,R2      ; MOVE FINISHED COMMAND HERE
7 007142 005067 171266      CLR      INDRFL      ; SAY NO INDIRECT FILE YET
8 007146 105067 171264      CLRB     QUOTFL      ; Say we are not in quoted string
9 007152 105067 175142      CLRB     DOLRAT      ; SAY HAVE NOT SEEN "$@"
10 007156 105067 175137      CLRB     DOTAT      ; SAY HAVE NOT SEEN "#@"
11 007162 105067 175134      CLRB     COLEQL      ; SAY HAVE NOT SEEN ":@"
12 007166 105067 175131      CLRB     NOUCL      ; SAY WE MAY CALL UCL FOR THIS COMMAND
13 007172 012704 001362' SCNCMD: MOV      #INBUF,R4      ; SCAN FROM HERE
14 007176 112400 8$:      MOVB     (R4)+,R0      ; GET NEXT CHAR FROM INPUT LINE
15 007200 001546      BEQ      4$      ; BR IF END OF LINE HIT
16 007202 120027 000040      CMPB     RO,#'      ; SKIP OVER LEADING SPACES
17 007206 001773      BEQ      8$
18 007210 120027 000011      CMPB     RO,#TAB      ; SKIP LEADING TABS
19 007214 001770      BEQ      8$
20 007216 120027 000014      CMPB     RO,#FF      ; SKIP LEADING FORM FEEDS
21 007222 001765      BEQ      8$
22 007224 000402      BR      9$      ; BEGIN SCANNING REAL COMMAND
23
24      ; Get next character from input line
25
26 007226 112400 6$:      MOVB     (R4)+,R0      ; GET NEXT CHAR FROM INPUT LINE
27 007230 001530      BEQ      11$      ; BR IF END OF LINE HIT
28
29      ; See if we are in a quoted string
30
31 007232 120067 171200 9$:      CMPB     RO,QUOTFL      ; Is this the terminating quote mark?
32 007236 001003      BNE      17$      ; Br if not
33 007240 105067 171172      CLRB     QUOTFL      ; Say not within quoted field now
34 007244 000511      BR      5$      ; Go store terminating quote character
35 007246 105767 171164 17$:      TSTB     QUOTFL      ; Are we inside a quoted string now?
36 007252 001106      BNE      5$      ; Br if yes -- Go store char without checking
37 007254 120027 000047      CMPB     RO,#47      ; Apostrophe character?
38 007260 001403      BEQ      19$      ; Br if yes
39 007262 120027 000042      CMPB     RO,#42      ; Quote character?
40 007266 001003      BNE      18$      ; Br if not
41 007270 110067 171142 19$:      MOVB     RO,QUOTFL      ; Remember we are inside a quoted string
42 007274 000475      BR      5$      ; Go store character without further checking
43
44      ; Check for start of comments
45
46 007276 120027 000041 18$:      CMPB     RO,#'!      ; START OF COMMENT FIELD?
47 007302 001503      BEQ      11$      ; BR IF YES
48
49      ; Check for :=
50
51 007304 105767 175012      TSTB     COLEQL      ; Have we already seen :=?
52 007310 001067      BNE      5$      ; Br if yes -- Ignore @'s after :=
53 007312 120027 000072      CMPB     RO,#':      ; Start of := sequence?
54 007316 001010      BNE      12$      ; Br if not
55 007320 110022      MOVB     RO,(R2)+      ; Store into result string
56 007322 112400      MOVB     (R4)+,R0      ; Get character following colon
57 007324 001472      BEQ      11$      ; Br if end of line hit

```

Get keyboard command

```

58 007326 120027 000075          CMPB    R0,#'=      ; Is this := ?
59 007332 001002          BNE     12$      ; Br if not
60 007334 105267 174762          INCB    COLEQL    ; Remember := seen within command string
61                               ;
62                               ;   Check for $@
63                               ;
64 007340 120027 000044      12$:    CMPB    R0,#'$      ; COULD THIS BE START OF "$@"?
65 007344 001014          BNE     10$      ; BR IF NOT
66 007346 005767 171062          TST     INDRFL    ; HAVE WE ALREADY SEEN @?
67 007352 001046          BNE     5$       ; BR IF YES
68 007354 121427 000100          CMPB    (R4),# '@     ; IS THIS "$@"?
69 007360 001043          BNE     5$       ; BR IF NOT
70 007362 010267 171046          MOV     R2,INDRFL    ; REMEMBER INDIRECT FILE NAME LOCATION
71 007366 105267 174726          INCB    DOLRAT    ; REMEMBER PREFIX WAS "$@"
72 007372 005204          INC      R4        ; SKIP PAST "$"
73 007374 000714          BR       6$       ; START GETTING FILE NAME
74                               ;
75                               ;   Check for #@
76                               ;
77 007376 120027 000043      10$:    CMPB    R0,#'#      ; COULD THIS BE START OF "#@"?
78 007402 001014          BNE     3$       ; BR IF NOT
79 007404 005767 171024          TST     INDRFL    ; HAVE WE ALREADY SEEN @?
80 007410 001027          BNE     5$       ; BR IF YES
81 007412 121427 000100          CMPB    (R4),# '@     ; IS THIS "#@"?
82 007416 001024          BNE     5$       ; BR IF NOT
83 007420 010267 171010          MOV     R2,INDRFL    ; REMEMBER INDIRECT FILE NAME LOCATION
84 007424 105267 174671          INCB    DOTAT     ; REMEMBER PREFIX WAS "#@"
85 007430 005204          INC      R4        ; POINT PAST "."
86 007432 000675          BR       6$       ; START GETTING FILE NAME
87                               ;
88                               ;   Check for @ and @@
89                               ;
90 007434 120027 000100      3$:     CMPB    R0,# '@     ; START OF INDIRECT FILE REFERENCE?
91 007440 001013          BNE     5$       ; BR IF NOT
92 007442 121427 000100          CMPB    (R4),# '@     ; Is this "@@"?
93 007446 001002          BNE     20$      ; Br if not
94 007450 005204          INC      R4        ; Skip past second at-sign
95 007452 000406          BR       5$       ; Translate "@@" to "@" and pass with command
96 007454 005767 170754      20$:    TST     INDRFL    ; ALREADY SEEN @ BEFORE?
97 007460 001003          BNE     5$       ; IF YES THEN IGNORE THIS ONE FOR NOW
98 007462 010267 170746          MOV     R2,INDRFL    ; SAVE POINTER TO START OF FILE NAME
99 007466 000657          BR       6$       ; DON'T BOTHER STORING @ IN CMDBUF
100                               ;
101                               ;   Move character to buffer
102                               ;
103 007470 020227 001342'      5$:     CMP     R2,#CBFEND    ; MAKE SURE WE DON'T OVERFLOW BUFFER
104 007474 103404          BLO     1$       ; BR IF OK
105 007476          FABORT    #ILLCMD      ; COMMAND TOO LONG
106 007506 110022      1$:     MOVB     R0,(R2)+    ; MOVE CHAR TO CMDBUF
107 007510 000646          BR       6$       ; GO GET REST OF LINE
108                               ;
109                               ;   Reached end of line -- strip off any trailing spaces and tabs
110                               ;
111 007512 010267 170776      11$:    MOV     R2,CMDEND    ; SAVE POINTER PAST END OF COMMAND STRING
112 007516 020227 001136'      4$:     CMP     R2,#CMDBUF    ; HAVE WE GONE PAST START OF BUFFER?
113 007522 001002          BNE     16$      ; BR IF NOT
114 007524 000167 177312          JMP     RDCMD      ; YES -- THIS IS A NULL COMMAND

```

Get keyboard command

```

115 007530 124227 000040      16$:  CMPB    -(R2),#'      ; IS NEXT CHARACTER A SPACE?
116 007534 001770              BEQ      4$              ; LOOP BACKWARD OVER SPACES
117 007536 121227 000011      CMPB    (R2),#TAB      ; IS THIS A TAB?
118 007542 001765              BEQ      4$              ; LOOP IF YES
119 007544 005202              INC      R2              ; POINT BEYOND LAST NON-BLANK CHARACTER
120
121                          ; See if command line is continued
122
123 007546 126227 177777 000055  CMPB    -1(R2),#'-      ; IS LINE CONTINUED?
124 007554 001013              BNE      GOTCML          ; BR IF NOT
125                          ; Line is continued -- get more
126 007556 005302              DEC      R2              ; POINT BACK TO -
127 007560              .GTLIN  #INBUF,#KMPRMT          ; READ COMMAND CONTINUATION LINE
128 007600 000167 177366      JMP      SCNCMD          ; CONTINUE SCANNING LINE

```

```

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 007604 105012      ;
7 007606 026727 170622 001136' GDTCL: CLRB      (R2)      ;store null at end of command
8 007614 001443      CMP      INDRFL,#CMDBUF ;Was "@" first character of command?
9 007616 105767 174500 BEQ      INDCMD      ;Br if yes -- Don't call UCL for this
10 007622 001411      TSTB     COLEQL      ;Was := seen within command line?
11 007624 005767 000000G BEQ      13$      ;Br if not
12 007630 001004      TST      UCLBLK      ;Are we supporting user-defined commands?
13 007632      BNE      1$      ;Br if yes
14 007642 000167 001306 FABORT  #EM$NUC      ;No user-defined commands allowed
15      ;
16      ; See if command line contains an indirect command file reference
17
18 007646 016703 170562 13$:     MOV      INDRFL,R3      ;Did we have indirect file reference?
19 007652 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 007654 126727 171256 000137 CMPB     CMDBUF,#'_' ;Was "_" specified as first char of command?
25 007662 001411      BEQ      15$      ;Br if yes -- Don't call TSXUCL for _command
26 007664 032761 000000G 000000G BIT      #$UCLCF,LSW7(R1);Should we call TSXUCL before normal commands?
27 007672 001412      BEQ      16$      ;Br if not
28 007674 005767 000000G TST      UCLBLK      ;Are we allowing user-defined commands?
29 007700 001407      BEQ      16$      ;Br if not
30 007702 000167 001246      JMP      CALUCL      ;Call TSXUCL to try to process this command
31 007706 112767 000040 171222 15$:  MOVB     #' ,CMDBUF ;Replace leading underscore with space
32 007714 105267 174403      INCB     NOUCL      ;Remember not to call TSXUCL for this command
33 007720 000167 000562      16$:     JMP      IDNCMD      ;Go try to identify the command

```

```

1      ;
2      ; Command line contains indirect file reference.
3      ;
4      ; Accrue indirect file name and try to open the file.
5      ;
6 007724 116701 000000G INDCMD: MOVB CORUSR,R1 ;GET JOB INDEX #
7 007730 016703 170500      MOV INDRFL,R3 ;GET POINTER TO COMMAND FILE NAME
8      ;
9      ; Accrue the command file spec
10     ;
11 007734 010304      MOV R3,R4 ;Save pointer to start of command file name
12 007736 012705 000244' MOV #DKCOM,R5 ;SET DEFAULT DEV AND EXT
13 007742 004767 000000G CALL ACRFN ;ACCRUE THE FILE NAME
14 007746 103002      BCC 4$ ;BR IF GOT NAME OK
15 007750 000167 177066      JMP RDCMD ;ERROR WHILE GETTING FILE NAME
16     ;
17     ; See if we should let IND process the indirect command file
18     ;
19 007754 026727 170454 001136' 4$: CMP INDRFL,#CMDBUF ;Was "@" first character of command?
20 007762 001031      BNE 1$ ;Br if not
21 007764 105767 174331      TSTB DOTAT ;WAS PREFIX "#@"?
22 007770 001407      BEQ 2$ ;BR IF NOT
23 007772 005767 000000G      TST INDSAV ;IS IND AVAILABLE ON SYSTEM?
24 007776 001013      BNE 3$ ;BR IF YES
25 010000      FABORT #NOIND ;IND IS NOT AVAILABLE
26 010010 032761 000000G 000000G 2$: BIT ##INDDF,LSW5(R1); IS IND TO BE CALLED BY DEFAULT?
27 010016 001413      BEQ 1$ ;BR IF NOT
28 010020 105767 174274      TSTB DOLRAT ;WAS COMMAND PREFIX "#@"?
29 010024 001010      BNE 1$ ;BR IF YES -- DON'T USE IND
30 010026 132767 000000G 000000G 3$: BITB #IN$ACT,INDSTA ;IS IND ACTIVE NOW?
31 010034 001004      BNE 1$ ;BR IF YES
32 010036 010403      MOV R4,R3 ;Get back pointer to start of file spec
33 010040 005005      CLR R5 ;NO DEFAULT DEVICE
34 010042 000167 001372      JMP INDINI ;GO ENTER IND
35     ;
36     ; Process this indirect command file directly rather than calling IND
37     ;
38 010046 016700 170366      1$: MOV FILNAM,R0 ;Get the device name
39 010052 004767 000000G      CALL CHKTID ;Is the device TT?
40 010056 103441      BCS NOICMD ;Br if yes -- Error
41 010060 004767 000000G      CALL PUSHCF ;PUSH CURRENT @FILE ON STACK
42 010064 112767 000001 000000G      MOVB #1,SERFLG ;DO .SERR
43 010072      . LOOKUP #XAREA,#CFCHAN,#FILNAM
44 010112 112767 000000 000000G      MOVB #0,SERFLG ;DO .HERR, DON'T CLEAR CARRY FLAG
45 010120 103024      BCC CFOPEN ;BR IF OPEN OK
46 010122 116701 000000G      MOVB CORUSR,R1 ;GET JOB INDEX NUMBER
47 010126 032761 000000G 000000G      BIT ##SUCF,LSW9(R1); ARE WE WITHIN A START-UP COMMAND FILE?
48 010134 001410      BEQ 5$ ;BR IF NOT
49 010136      FERR #NOCF ;PRINT ERROR MESSAGE
50 010152 000167 000000G      JMP CMDOFF ;LOG THE JOB OFF
51 010156 004767 000000G      5$: CALL POPCF ;POP PUSHED COMMAND FILE STATUS
52 010162      NOICMD: FABORT #NOCF ;COULDN'T OPEN @FILE
53     ;
54     ; We have successfully opened the indirect command file.
55     ;
56 010172 116701 000000G      CFOPEN: MOVB CORUSR,R1
57 010176 052761 000000G 000000G      BIS ##CFOPN,LSW4(R1); SAY CFCHAN IS OPEN

```

Get keyboard command

```

58
59 ; See if command file was installed with any privileges.
60
61 010204 004767 001726          CALL    INSCF          ;See if command file was installed
62
63 ; READ IN 1ST BLOCK FROM INDIRECT FILE
64
65 010210          . READW  #XAREA, #CFCHAN, #CFBUF, #256., #0
66 010246 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 010250 012767 000000G 000000G      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
76 010256 105713          CFPRM: TSTB     (R3)          ;HIT END OF COMMAND?
77 010260 001437          BEQ     RDREST          ;BR IF END HIT
78 010262 122327 000040          CMPB     (R3)+, #'      ;SKIP OVER LEADING SPACES
79 010266 001773          BEQ     CFPRM
80 ; SEE IF WE SHOULD USE SPACE OR \ AS PARAMETER DELIMITER
81 010270 012701 000040          MOV     #' , R1          ;ASSUME SPACE IS DELIMITER
82 010274 124327 000134          CMPB     -(R3), #' \      ;DOES HE WANT TO USE \?
83 010300 001001          BNE     8$              ;BR IF NOT -- USE SPACE
84 010302 112301          MOVB     (R3)+, R1          ;USE \ AS PARAM SEPARATOR
85 ; THERE ARE SOME PARAMETERS -- ACCRUE THEM
86 010304 012704 000000G      8$: MOV     #PRMPNT, R4          ;POINT TO PARAM POINTER CELLS
87 010310 012705 000000G      MOV     #PRMBUF, R5          ;POINT TO PARAMETER STRING BUFFER
88 010314 020427 000000G      7$: CMP     R4, #LSTPRM          ;TOO MANY PARAMETERS?
89 010320 103062          BHS      TOMPRM          ;BR IF TOO MANY
90 010322 010524          MOV     R5, (R4)+          ;SET PARAMETER STRING POINTER
91 010324 121301          6$: CMPB     (R3), R1          ;REACHED PARAM DELIMITER YET?
92 010326 001407          BEQ     4$              ;BR IF YES
93 010330 105713          TSTB     (R3)          ;HIT END OF COMMAND?
94 010332 001405          BEQ     4$              ;BR IF YES
95 010334 020527 000000G      CMP     R5, #PRMEND          ;HIT END OF PARAM STRING BUFFER?
96 010340 103056          BHS      PTL              ;BR IF PARAM STRING TOO LONG
97 010342 112325          MOVB     (R3)+, (R5)+        ;MOVE PARAMETER TO BUFFER
98 010344 000767          BR       6$
99 ; HIT END OF PARAMETER -- STORE NULL IN STRING TO MARK END.
100 010346 105025          4$: CLRB     (R5)+          ;FLAG END OF THIS PARAMETER STRING
101 010350 105723          TSTB     (R3)+          ;MORE TO ACCRUE?
102 010352 001360          BNE     7$              ;BR IF YES
103 010354 010567 000000G      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 010360 016702 170050      RDREST: MOV     INDRFL, R2          ;GET POINTER TO START OF @SPECIFICATION
113 010364 001436          BEQ     CFJMP          ;BR IF IMPLICIT @FILE EXECUTION
114 010366 020227 001136'      CMP     R2, #CMDBUF          ;IS @FILE 1ST THING IN COMMAND?

```

```

115 010372 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 010374 116701 000000G      MOVB      CORUSR,R1      ;GET USER INDEX #
121 010400 016146 000000G      MOV       LSW4(R1),-(SP)    ;SAVE COMMAND FILE STATUS FLAGS
122 010404 012705 000000G      MOV       $$QUIET,R5      ;GET QUIET FLAG INTO REGISTER
123 010410 050561 000000G      BIS       R5,LSW4(R1)      ;TURN QUIET ON
124 010414          .GT LIN   #INBUF      ;READ LINE FROM @FILE
125 010432 030526          BIT       R5,(SP)+      ;NEED TO SET OR RESET QUIET MODE?
126 010434 001002          BNE      10$      ;BR IF WANT TO LEAVE IT SET
127 010436 040561 000000G      BIC       R5,LSW4(R1)      ;RESET QUIET MODE
128 010442 005067 167766      10$:      CLR      INDRFL      ;SAY @FILE REFERENCE HAS BEEN RESOLVED
129 010446 105067 173646      CLRB      DOLRAT      ;SAY HAVEN'T SEEN "$@"
130 010452 105067 173643      CLRB      DOTAT      ;SAY HAVEN'T SEEN "#@"
131 010456 000167 176510      JMP       SCNCMD      ;GO GET REST OF COMMAND
132 010462 000167 176354      CFJMP:    JMP       RDCMD      ;GO READ NEXT COMMAND
133
134 010466          ; ERROR -- TOO MANY PARAMETERS
135          TOMPRM: FABORT #NUMPRM
136 010476          ; ERROR -- PARAMETER STRING TOO LONG
          PTL:      FABORT #STRLEN

```

Identify command

```

1
2
3
4
5
6
7
8
9
10 010506 012704 000000'
11 010512 012703 001136'
12 010516 004767 000000G
13 010522 103006
14
15 010524 005704
16 010526 001407
17 010530
18
19
20
21
22
23
24
25
26
27 010540 116701 000000G
28 010544 000134

```

```

      .SBTTL  Identify command
      -----
; AT THIS POINT A COMMAND LINE HAS BEEN ACCEPTED, CONTINUATION
; LINES READ, COMMENTS STRIPPED AND INDIRECT FILE REFERENCES
; RESOLVED.  THE RESULTING COMMAND IS STORED IN CMDBUF IN
; ASCIIZ FORM WITH R2 POINTING TO THE END OF THE COMMAND.
;
; TRY TO IDENTIFY IT AS A SYSTEM COMMAND.
;
IDNCMD: MOV      #CMDHDL,R4      ;POINT TO TABLE OF SYSTEM COMMANDS
        MOV      #CMDBUF,R3     ;POINT TO OUR COMMAND
        CALL     SEARCH         ;LOOK UP THE COMMAND KEYWORD
        BCC      FNDCMD         ;BR IF IDENTIFIED THE COMMAND
; ERROR DURING SEARCH.  SEE IF UNRECOGNIZED OR AMBIGUOUS.
        TST      R4             ;AMBIGUOUS OR UNRECOGNIZED?
        BEQ      NOCMD          ;BR IF UNRECOGNIZED
        FABORT    #ABCMD        ;AMBIGUOUS COMMAND
;
; WE HAVE A VALID SYSTEM COMMAND.
;
; BRANCH OFF TO COMMAND PROCESSING ROUTINE.
; AT THIS POINT THE FOLLOWING REGISTERS ARE SET UP:
;   R1 = USER INDEX NUMBER
;   R2 = ADDRESS OF END OF COMMAND STRING
;   R3 = ADDRESS OF START OF COMMAND ARGUMENT FIELD.
;
FNDCMD: MOVB     CORUSR,R1       ;GET USER INDEX #
        JMP      @(R4)+         ;ENTER COMMAND PROCESSING ROUTINE

```

Identify command

```

1
2 ; -----
3 ; We could not identify the command as a standard system command
4 ; so we try to identify it as an implicit command in the following way:
5 ;
6 ; 1. See if it is a user-defined command (if SET UCL MIDDLE).
7 ; 2. See if there is a command file on "DK:" with command name.
8 ; 3. See if there is a command file on "SY:" with command name.
9 ; 4. See if there is a SAV file on "SY:" with command name.
10 ; 5. See if there is a SY:UCL program to process the command (if UCL LAST).
11 ;
12 ; See if we should call TSXUCL to process this command.
13 010546 116701 000000G NOCMD: MOV B CORUSR,R1 ;GET CURRENT JOB INDEX NUMBER
14 010552 032761 000000G 000000G BIT #UCLCM,LSW7(R1);SET UCL MIDDLE?
15 010560 001410 BEQ 7$ ;BR IF NOT
16 010562 105767 173535 TSTB NOUCL ;DID UCL ALREADY REJECT THIS COMMAND?
17 010566 001005 BNE 7$ ;BR IF YES
18 010570 005767 000000G TST UCLBLK ;ARE WE ALLOWING USER-DEFINED COMMANDS?
19 010574 001402 BEQ 7$ ;BR IF NOT
20 010576 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 010602 012703 001136' 7$: MOV #CMDBUF,R3 ;POINT TO KEYWORD NAME
25 010606 012705 000244' MOV #DKCOM,R5 ;GET DEV AND EXT DEFAULTS
26 010612 004767 000000G CALL ACRFN ;ACCRUE THE FILE NAME
27 010616 103513 BCS 6$ ;BR IF ERROR IN GETTING FILE NAME
28 010620 016700 167614 MOV FILNAM,R0 ;Get the device name
29 010624 004767 000000G CALL CHKTTD ;Is it TT?
30 010630 103506 BCS 6$ ;Error if yes
31 010632 004767 000000G CALL PUSHCF ;PUSH CURRENT @FILE STATUS
32 010636 112767 000001 000000G MOV B #1,SERFLG ;DON'T ABORT ON ERRORS
33 010644 . LOOKUP #XAREA,#CFCHAN,#FILNAM
34 010664 112767 000000 000000G MOV B #0,SERFLG ;DO .HERR -- DON'T CLEAR CARRY FLAG
35 010672 103402 BCS 3$ ;BR IF NOT FOUND
36 010674 005005 CLR R5 ;SAY NO DEFAULT DEVICE NAME
37 010676 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 010700 016767 167360 167532 3$: MOV R5OSY,FILNAM ;CHANGE DEVICE NAME TO BE "SY"
42 010706 112767 000001 000000G MOV B #1,SERFLG ;DO .SERR
43 010714 . LOOKUP #XAREA,#CFCHAN,#FILNAM
44 010734 112767 000000 000000G MOV B #0,SERFLG ;DO .HERR, DON'T CLEAR CARRY FLAG
45 010742 103430 BCS 1$ ;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 010744 012705 004324' MOV #SYTXT,R5 ;GET DEFAULT DEVICE FOR IND FILE
51 010750 132767 000000G 000000G 4$: BITB #IN$ACT,CFIND ;IS IND ACTIVE NOW?
52 010756 001015 BNE 2$ ;BR IF ALREADY ACTIVE
53 010760 032761 000000G 000000G BIT #INDDF,LSW5(R1);IS IND WANTED?
54 010766 001411 BEQ 2$ ;BR IF NOT
55 010770 . PURGE #CFCHAN ;PURGE CHANNEL WE OPENED TO FILE
56 010776 004767 000000G CALL POPCF ;POP UP TO OLD FILE
57 011002 012703 001136' MOV #CMDBUF,R3 ;POINT TO START OF COMMAND LINE

```

Identify command

```

58 011006 000167 000426          JMP      INDINI          ;GD START UP IND
59
60          ; This is an implicit command file execution.
61
62 011012 052761 000000G 000000G 2$:      BIS      #$QUIET,LSW4(R1); ALWAYS SET QUIET IF IMPLICIT RUN
63 011020 000167 177146          JMP      CFOPEN          ;CONTINUE PROCESSING STARTUP OF @FILE
64
65          ; This is not an implicit @file call.
66
67 011024 004767 000000G          1$:      CALL     POPCF          ;REOPEN PREVIOUS @FILE
68
69          ; See if there is a program on "SY" with command name.
70
71 011030 012703 001136'          MOV      #CMDBUF,R3          ;POINT TO COMMAND KEYWORD
72 011034 012705 000250'          MOV      #SYSAV,R5          ;SET DEFAULT DEV AND EXT
73 011040 004767 000000G          CALL     ACRFN          ;ACCRUE FILE NAME
74 011044 103002          BCC      5$          ;BR IF GOT FILE NAME OK
75 011046 000167 175770          6$:      JMP      RDCMD          ;ERROR IN GETTING FILE NAME
76 011052 016700 167362          5$:      MOV      FILNAM,R0          ;Get the device name
77 011056 004767 000000G          CALL     CHKTTD          ;Is the device TT?
78 011062 103421          BCS      BADCMD          ;Error if yes
79 011064 112767 000001 000000G      MOV     #1,SERFLG          ;DO .SERR
80 011072          . LOOKUP #XAREA,#RUNCHN,#FILNAM
81 011112 112767 000000 000000G      MOV     #0,SERFLG          ;DO .HERR, DON'T CLEAR CARRY FLAG
82 011120 103406          BCS      TRYUCL          ;COULDN'T FIND PROGRAM
83 011122 000167 000000G          JMP      DORUN          ;START RUNNING THE PROGRAM
84
85 011126          BADCMD: FABORT #ILLCMD          ;INVALID COMMAND

```

Identify command

```

1          ; -----
2          ; See if there is a SY:TSXUCL program to process the command
3          ;
4 011136 032761 000000G 000000G TRYUCL: BIT    #$UCLCL,LSW7(R1); SHOULD WE CALL TSXUCL LAST?
5 011144 001451          BEQ    URERR      ;BR IF NOT
6 011146 105767 173151    TSTB    NOUCL     ;DID UCL ALREADY REJECT THIS COMMAND?
7 011152 001046          BNE    URERR      ;BR IF YES
8          ;
9          ; Call the TSXUCL program to process this command
10         ;
11 011154 105767 000000G CALUCL: TSTB    VU$CL      ;IS THERE A UCL PROGRAM?
12 011160 001443          BEQ    URERR      ;BR IF NOT
13         ;
14         ; UCL option is genned in. See if we can find UCL program.
15         ;
16 011162          . LOOKUP #XAREA, #RUNCHN, #UCLNAM ; TRY TO FIND SY:UCL. SAV
17 011202 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 011204 012703 000012G      MOV    #CINDAT+12,R3 ; POINT TO CHAIN AREA
23 011210 012704 001136'      MOV    #CMDBUF,R4   ; POINT TO COMMAND LINE BUFFER
24 011214 112423          1$:  MOVB    (R4)+,(R3)+ ; MOVE COMMAND TO CHAIN DATA AREA
25 011216 001376          BNE    1$           ; LOOP TILL ASCIZ NULL MOVED
26 011220 162703 000013G      SUB    #CINDAT+13,R3 ; COMPUTE LENGTH OF COMMAND LINE
27 011224 010367 000010G      MOV    R3,CINDAT+10 ; AND STORE LENGTH INTO 510
28 011230 012704 000000G      MOV    #UCLNAM,R4   ; POINT TO NAME OF UCL PROGRAM
29 011234 105267 000000G      INCB    CINFLG      ; SIMULATE .CHAIN
30 011240 116701 000000G      MOVB    CORUSR,R1   ; GET CURRENT JOB INDEX NUMBER
31 011244 052761 000000G 000000G BIS    #$UCLRN,LSW7(R1); SAY THAT UCL PROGRAM IS RUNNING
32 011252 005000          CLR    R0             ; NO RUN OPTION FLAGS FOR PLOAD
33 011254 000167 000000G      JMP     PLOAD      ; LOAD AND RUN UCL PROGRAM
34         ;
35         ; Cannot find TSXUCL program
36         ;
37 011260          9$:  FABORT  #MISUCL      ; SAY UCL IS MISSING
38         ;
39         ; Could not identify command
40         ;
41 011270          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 011300 116701 000000G CALUKM: MOVB  CORUSR,R1      ;Get job index number
6 011304 012704 000000G      MOV   #UCISPC,R4      ;Get pointer to program name for PLOAD
7 011310      .LOOKUP #XAREA,#RUNCHN,R4;Try to lookup program
8 011326 103406      BCS    9$      ;Br if cannot find program
9          ;
10         ; We found the program. Enter it.
11         ;
12 011330 052761 000000G 000000G      BIS    #$UKMRN,LSW7(R1);Say user command processor is running
13 011336 005000      CLR    RO      ;No run option flags for PLOAD
14 011340 000167 000000G      JMP    PLOAD      ;Enter the program
15         ;
16         ; Cannot find the program.
17         ;
18 011344 042761 000000G 000000G 9$:  BIC    #$UKMRN,LSW7(R1);Don't try to use the program again
19 011352      FABORT  #EM$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 011362   CMDIND:
7          ;
8          ;  See if IND is available
9          ;
10 011362   005767   000000G      TST      INDSAV      ;Is IND available on system?
11 011366   001004      BNE      1$      ;Br if yes
12 011370      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 011400   004767   000000G      1$:      CALL      SKPSPC      ;Skip over any spaces
18 011404   105713      TSTB      (R3)      ;Any file specified?
19 011406   001411      BEQ      4$      ;Br if not
20 011410   010304      MOV      R3,R4      ;Save pointer to start of command file
21 011412   012705   000244'      MOV      #DKCOM,R5      ;Set default device and extension
22 011416   004767   000000G      CALL      ACRFN      ;Accrue the file name
23 011422   103002      BCC      2$      ;Br if got file name ok
24 011424   000167   175412      JMP      RDCMD      ;Error accruing file name
25 011430   010403      2$:      MOV      R4,R3      ;Restore pointer to start of file spec
26          ;
27          ;  Start execution of IND
28          ;
29 011432   005005      4$:      CLR      R5      ;No default device string
30 011434   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         INDINI:
11         ;
12         ; Error if IND is already running
13         ;
14         011440 132767 000000G 000000G      BITB    #IN$ACT,INDSTA ;Is IND running now?
15         011446 001404                      BEQ     6$           ;Br if not
16         011450                      FABORT   #INDACT           ;IND is already active
17         ;
18         ; Set flag that says IND is being started
19         ;
20         011460 116701 000000G              6$:   MOVB    CORUSR,R1      ;Get job index number
21         011464 052761 000000G 000000G      BIS     #$INDRN,LSW5(R1);Say IND is running
22         ;
23         ; See if command file was installed with any privileges
24         ;
25         011472 004767 000440              CALL    INSCF          ;See if command file was installed
26         ;
27         ; Build command line of the form "SY:IND file-name"
28         ;
29         011476 012702 001362'             MOV     #BLK0,R2        ;Point to area where we will build command
30         ;
31         ; Insert "SY:IND "
32         ;
33         011502 012704 004330'             MOV     #SYINTX,R4      ;Point to "SY:IND " text string
34         011506 112422                     1$:   MOVB    (R4)+,(R2)+ ;Copy text string
35         011510 001376                     BNE     1$           ;Loop till null hit
36         011512 005302                     DEC     R2           ;Point back to null
37         ;
38         ; See if we need to insert default device name
39         ;
40         011514 005705                     TST     R5           ;Do we have a default device name?
41         011516 001403                     BEQ     2$           ;Br if not
42         011520 112522                     3$:   MOVB    (R5)+,(R2)+ ;Move in default device name
43         011522 001376                     BNE     3$           ;
44         011524 005302                     DEC     R2           ;Point back over terminating null
45         ;
46         ; Copy command text string
47         ;
48         011526 010304                     2$:   MOV     R3,R4      ;Get pointer to command buffer
49         011530 112422                     4$:   MOVB    (R4)+,(R2)+ ;Copy command text string
50         011532 001376                     BNE     4$           ;
51         ;
52         ; Now move string back to command buffer
53         ;
54         011534 012702 001362'             MOV     #BLK0,R2        ;Point to start of new string
55         011540 010304                     MOV     R3,R4          ;Point to destination area
56         011542 112224                     5$:   MOVB    (R2)+,(R4)+ ;Copy string
57         011544 001376                     BNE     5$           ;

```

INDINI -- Start IND program

```

58      ;
59      ; Now process like a R command
60      ;
61 011546 156767 000000G 000000G      BISB  UERSEV, INDERR ; Pass error sev level to IND
62 011554 142767 000000G 000000G      BICB  #IN$CMD, INDSTA ; Say no IND command pending
63 011562 012705 000250'      MOV   #SYSAV, R5 ; Point to default device (SY)
64 011566 000167 000000G      JMP   RUNNAM ; Enter RUN code
65      ;
66      ; Reopen channel to IND.SAV file
67      ;
68 011572      INDRUN: .REOPEN #XAREA, #RUNCHN, #INDSAV ; REOPEN CHANNEL 16 TO IND
69 011612 012704 000372'      MOV   #INDNAM, R4 ; POINT TO NAME OF IND
70 011616 105267 000000G      INCB  CINFLG ; SAY A .CHAIN IS IN PROGRESS
71 011622 156767 000000G 000000G      BISB  UERSEV, INDERR ; PASS ERROR SEV LEVEL TO IND
72 011630 116701 000000G      MOVB  CORUSR, R1 ; GET JOB INDEX NUMBER
73 011634 052761 000000G 000000G      BIS   ##INDRN, LSW5(R1) ; SAY IND IS RUNNING
74 011642 142767 000000G 000000G      BICB  #IN$CMD, INDSTA ; SAY NO IND COMMAND PENDING
75 011650 005000      CLR   RO ; No run option flags for PLOAD
76 011652 000167 000000G      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 011656 012703 000012G CMDCCCL: MOV      #CINDAT+12,R3  ; POINT TO CHAIN DATA AREA
12 011662 012704 001136'  MOV      #CMDBUF,R4    ; POINT TO COMMAND BUFFER
13 011666 112400          1$:  MOVB    (R4)+,R0    ; GET NEXT CHAR FROM COMMAND LINE
14 011670 120027 000011    CMPB    R0,#TAB      ; IS CHARACTER TAB?
15 011674 001403          BEQ      10$           ; BR IF YES
16 011676 120027 000014    CMPB    R0,#FF        ; FORM-FEED?
17 011702 001002          BNE      11$           ;
18 011704 112700 000040    10$:  MOVB    #' ,R0    ; TRANSLATE TAB AND FF TO SPACE
19 011710 120027 000141    11$:  CMPB    R0,#141   ; SEE IF IT IS A LOWER CASE LETTER
20 011714 103405          BLO      6$            ; BR IF NOT
21 011716 120027 000172    CMPB    R0,#172       ; LOWER CASE Z
22 011722 101002          BHI      6$            ; BR IF DELIMITER
23 011724 162700 000040    SUB      #40,R0       ; CONVERT LOWER-CASE UP UPPER-CASE
24 011730 110023          6$:  MOVB    R0,(R3)+    ; MOVE CHAR TO CHAIN AREA
25 011732 001355          BNE      1$            ; LOOP TILL ASCIZ NULL MOVED
26          ; NOW SET SOME FLAGS IN LOCATION 510
27 011734 005003          CLR      R3            ; FORM FLAG WORD IN R3
28 011736 016104 000000G  MOV      LSW5(R1),R4    ; PICK UP STATUS WORD
29 011742 032704 000000C  BIT      #$TECO!$KED,R4 ; IS DEFAULT EDITOR TECO OR KED?
30 011746 001417          BEQ      2$            ; BR IF NOT (MUST BE EDIT THEN)
31 011750 032704 000000G  BIT      #$TECO,R4      ; IS IT TECO?
32 011754 001403          BEQ      7$            ; BR IF NOT
33 011756 052703 000001  BIS      #$TECO,R3      ; TELL CCL TO USE TECO
34 011762 000411          BR       2$            ;
35 011764 026127 000000G 000000G 7$:  CMP      LTRMTP(R1),#VT52; IS TERMINAL TYPE VT52?
36 011772 001403          BEQ      8$            ; BR IF YES (EDITOR = K52)
37 011774 052703 000020  BIS      #$KED,R3       ; DEFAULT EDITOR = KED
38 012000 000402          BR       2$            ;
39 012002 052703 000040    8$:  BIS      #$K52,R3   ; DEFAULT EDITOR = K52
40 012006 032761 000000G 000000G 2$:  BIT      #$DIBOL,LSW6(R1); IS DIBOL OR DBL DEFAULT COMPILER?
41 012014 001402          BEQ      9$            ; BR IF DBL IS DEFAULT
42 012016 052703 000100  BIS      #$DIBL,R3      ; REMEMBER DIBOL IS DEFAULT
43 012022 032704 000000G  9$:  BIT      #$WILD,R4   ; WANT IMPLICIT/EXPLICIT WILDCARDS?
44 012026 001402          BEQ      3$            ; BR IF WANTS EXPLICIT
45 012030 052703 000002  BIS      #$WILD,R3      ; SET IMPLICIT-WILDCARD FLAG
46 012034 032704 000000G  3$:  BIT      #$CLTST,R4  ; DO CCL IN TEST MODE?
47 012040 001402          BEQ      5$            ; BR IF NOT
48 012042 052703 000010  BIS      #$TEST,R3      ; SET TEST MODE FLAG
49 012046 032761 000000G 000000G 5$:  BIT      #$QUIET,LSW4(R1); ARE COMMAND FILES IN QUIET MODE?
50 012054 001402          BEQ      4$            ; BR IF NOT
51 012056 052703 000004  BIS      #$QUIT,R3      ; SET QUIET FLAG
52 012062 110337 000010G  4$:  MOVB    R3,@#CINDAT+10 ; STORE FLAG WORD INTO CHAIN DATA AREA
53 012066 110137 000011G  MOVB    R1,@#CINDAT+11 ; STORE USER INDEX # IN 511 FOR CCL
54          ; Reopen channel 16 with CCL file status saved during initialization.
55 012072          .REOPEN #XAREA,#RUNCHN,#CCLSAV ; OPEN CHANNEL TO CCL.SAV FILE
56 012112 012704 000362'  MOV      #CCLNAM,R4    ; POINT TO CELL WITH CCL NAME
57 012116 105267 000000G  INCB     CINFLG        ; SIMULATE CHAIN

```

58	012122	052761	000000G	000000G	BIS	##CCLRN,LSW5(R1);REMEMBER CCL.SAV IS RUNNING
59	012130	005000			CLR	RO ;No run option flags for PLOAD
60	012132	000167	000000G		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 012136 010246 INSCF:  MOV      R2, -(SP)
13         ;
14         ; See if this command file is installed
15         ;
16 012140 012700 000440'      MOV      #FILNAM, R0      ; Point to file spec
17 012144 004767 000000G      CALL     INSSRC          ; See if file is in install table
18 012150 103420              BCS      9$              ; Br if not
19         ;
20         ; Command file is installed.
21         ; Apply any privilege changes.
22         ;
23 012152 012702 000000C      MOV      #2*(PVNPW-1), R2 ; Get index to last privilege word
24 012156 056262 000000C 000000G 1$:  BIS      II$PRV+IIBUF(R2), PRIVFO(R2) ; Set some flags
25 012164 046262 000000C 000000G      BIC      II$NPV+IIBUF(R2), PRIVFO(R2) ; Clear some flags
26 012172 162702 000002      SUB      #2, R2          ; More to do?
27 012176 002367              BGE      1$              ; Loop if yes
28 012200 056767 000001C 000000G      BIS      II$FLG+IIBUF, AF CF ; Set command file attribute flags
29 012206 004767 000000G      CALL     RSTPRV          ; Set current attributes for command file
30         ;
31         ; Finished
32         ;
33 012212 012602 9$:  MOV      (SP)+, R2
34 012214 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 012216 010246 TRMINI: MOV      R2, -(SP)
10         ;
11         ; Get initial LSW2 flags
12         ;
13 012220 016100 0000000 MOV      LTRMTP(R1), R0 ; GET TERMINAL TYPE FLAGS
14 012224 016102 0000000 MOV      LSW2(R1), R2 ; GET LSW2 FLAGS
15         ;
16 012230 032700 0000000 ; VT100
17 012234 001410         BIT      #VT100, R0 ; IS THIS A VT100 TERMINAL?
18 012236 042702 0000000 BEQ      14$ ; BR IF NOT
19 012242 052702 0000000 BIC      #VT10ND, R2 ; SET FLAGS FOR LSW2
20 012246         BIS      #VT10FL, R2
21 012254 000503         .PRINT  #IMVT10 ; CLEAR SCREEN
22         BR      15$
23 012256 032700 0000000 ; VT200
24 012262 001410 14$: BIT      #VT2007!VT2008, R0 ; VT200 terminal?
25 012264 042702 0000000 BEQ      32$ ; Br if not
26 012270 052702 0000000 BIC      #VT20ND, R2 ; Set flags for VT200
27 012274         BIS      #VT20FL, R2
28 012302 000470         .PRINT  #IMVT10 ; Clear screen
29         BR      15$
30 012304 032700 0000000 ; VT52
31 012310 001410 32$: BIT      #VT52, R0 ; IS THIS A VT52 TERMINAL?
32 012312         BEQ      16$ ; BR IF NOT
33 012320 042702 0000000 .PRINT  #IMVT52 ; CLEAR SCREEN
34 012324 052702 0000000 BIC      #VT52ND, R2 ; SET FLAGS IN LSW2
35 012330 000455         BIS      #VT52FL, R2
36         BR      15$
37 012332 032700 0000000 ; ADM3A
38 012336 001410 16$: BIT      #ADM3A, R0 ; IS THIS AN ADM3A TERMINAL?
39 012340         BEQ      17$ ; BR IF NOT
40 012346 042702 0000000 .PRINT  #IMADM3 ; CLEAR SCREEN
41 012352 052702 0000000 BIC      #ADM3ND, R2 ; SET LSW2 FLAGS
42 012356 000442         BIS      #ADM3FL, R2
43         BR      15$
44 012360 032700 0000000 ; LA36
45 012364 001405 17$: BIT      #LA36, R0 ; IS THIS AN LA36?
46 012366 042702 0000000 BEQ      18$ ; BR IF NOT
47 012372 052702 0000000 BIC      #LA36ND, R2 ; SET FLAGS
48 012376 000432         BIS      #LA36FL, R2
49         BR      15$
50 012400 032700 0000000 ; LA120
51 012404 001405 18$: BIT      #LA120, R0 ; IS THIS AN LA120?
52 012406 042702 0000000 BEQ      19$ ; BR IF NOT
53 012412 052702 0000000 BIC      #LA12ND, R2 ; SET FLAGS
54 012416 000422         BIS      #LA12FL, R2
55         BR      15$
56 012420 032700 0000000 ; Hazeltine
57 012424 001410 19$: BIT      #HAZEL, R0 ; HAZELTINE TERMINAL?
                    BEQ      20$ ; BR IF NOT

```

TRMINI -- Perform terminal-dependent initialization

```
58 012426          .PRINT  #IMHAZL          ;CLEAR SCREEN
59 012434  042702  000000G      BIC      #HAZLND,R2      ;SET TERMINAL CONTROL FLAGS
60 012440  052702  000000G      BIS      #HAZLFL,R2
61 012444  000407          BR      15$
62          ; DIABLE & QUME
63 012446  032700  000000C  20$:  BIT      #DIABLO!QUME,R0 ;DIABLO OR QUME TERMINAL?
64 012452  001410          BEQ      1$      ;BR IF NOT
65 012454  042702  000000G      BIC      #DIABND,R2      ;SET FLAGS
66 012460  052702  000000G      BIS      #DIABFL,R2
67          ; Store updated LSW2 flags.
68 012464  010261  000000G  15$:  MOV      R2,LSW2(R1)
69 012470  010261  000000G      MOV      R2,LSW2S(R1)
70          ;
71          ; Finished
72          ;
73 012474  012602  1$:  MOV      (SP)+,R2
74 012476  000207          RETURN
```

VIRINI -- Virtual line initialization

```

1
2
3
4
5
6
7
8
9
10 012500 010246
11 012502 010346
12 012504 010446
13
14
15
16 012506 016102 000000G
17
18
19
20
21 012512 005762 000000G
22 012516 001413
23 012520 012700 001074'
24 012524 110260 000007
25 012530 106260 000007
26 012534 104375
27 012536 103403
28 012540 012700 001106'
29 012544 104375
30
31
32
33 012546 016204 000000G
34 012552 005003
35 012554 005203
36 012556 120124
37 012560 001375
38 012562 062703 000060
39 012566
40 012574
41 012602
42
43
44
45 012612 016102 000000G
46 012616 004767 000000G
47
48
49
50 012622 012604
51 012624 012603
52 012626 012602
53 012630 000207

```

```

.SBTTL VIRINI -- Virtual line initialization
;-----
; Perform initialization for virtual lines during job startup.
; If the primary line is using display windows then create a display
; window for the virtual line. Otherwise print n>.
;
; Inputs:
; R1 = Job index number
;
VIRINI: MOV R2,-(SP)
        MOV R3,-(SP)
        MOV R4,-(SP)
;
; Get job index number of our primary job
;
        MOV LNPRIM(R1),R2 ;GET PRIMARY LINE INDEX NUMBER
;
; If primary line is using a display window, try to create one for
; this job.
;
        TST LWINDO(R2) ;Is primary line using a display window?
        BEQ 1$ ;Br if not
        MOV #MAKWIN,R0 ;Point to make-window argument block
        MOVB R2,7(R0) ;Set index # of primary job
        ASRB 7(R0) ;Convert index to job number
        EMT 375 ;Try to create a window
        BCS 1$ ;Br if unable to create a window
        MOV #MAPWIN,R0 ;Point to map-window argument block
        EMT 375 ;Select this window
;
; Print n>
;
1$: MOV LSECPT(R2),R4 ;GET ADDRESS OF TABLE WITH VIRTUAL LINES
    CLR R3 ;COUNT VIRTUAL LINE #'S
13$: INC R3
    CMPB R1,(R4)+ ;LOOK UP OUR VIRTUAL LINE #
    BNE 13$
    ADD #'0,R3 ;CONVERT NUMBER TO ASCII
    .PRINT #CRLF ;PRINT CR-LF
    .TTYOUT R3 ;PRINT LINE #
    .TTYOUT #76 ;PRINT '>'
;
; Try to copy any key definitions from primary line
;
        MOV LNPRIM(R1),R2 ;Get our primary line number
        CALL INIUKD ;Copy any user-defined keys
;
; Finished
;
9$: MOV (SP)+,R4
    MOV (SP)+,R3
    MOV (SP)+,R2
    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 012632    010246
6 012634    010346
7 012636    010546
8          ;
9          ; Return immediately if there is no parent job
10         ;
11 012640    016102    000000G
12 012644    001002
13 012646    000167    000456
14         ;
15         ; Copy file context and privilege information
16         ;
17 012652    010267    165704
18 012656    006267    165700
19 012662    012700    000560'
20 012666    104375
21 012670    004767    000000G
22         ;
23         ; Copy execution priority
24         ;
25 012674    012700    001070'
26 012700    116260    000000G 000002
27 012706    104375
28         ;
29         ; Copy user name and PPN
30         ;
31 012710    010203
32 012712    010105
33 012714    070327    000006
34 012720    070527    000006
35 012724    062703    000000G
36 012730    062705    000000G
37 012734    012700    000014
38 012740    112325
39 012742    077002
40 012744    016261    000000G 000000G
41 012752    016261    000000G 000000G
42 012760    016167    000000G 000000G
43 012766    016167    000000G 000002G
44         ;
45         ; Copy flags from LSW2
46         ;
47 012774    016261    000000G 000000G
48 013002    016261    000000G 000000G
49         ;
50         ; Copy some flags from LSW5
51         ;
52 013010    016200    000000G
53 013014    042700    000000C
54 013020    042761    000000G 000000G
55 013026    050061    000000G
56         ;
57         ; Copy some flags from LSW6

```

```

CPYPRN: MOV      R2,-(SP)
        MOV      R3,-(SP)
        MOV      R5,-(SP)

        ;
        ; Return immediately if there is no parent job
        ;
        MOV      LPARNT(R1),R2 ;Get # of parent job
        BNE      2$            ;Br if there is a parent job
        JMP      9$            ;No parent job -- Nothing to copy

        ; Copy file context and privilege information
        ;
2$:      MOV      R2,CPYCXT+2    ;Get # of parent job
        ASR      CPYCXT+2      ;Convert to job number
        MOV      #CPYCXT,R0    ;Point to EMT arg block
        EMT      375           ;Copy context from parent job
        CALL     LDCLN         ;Reinit LD status

        ; Copy execution priority
        ;
        MOV      #PRIEMT,R0    ;Point to emt argument block
        MOVB     LBSPRI(R2),2(R0);Get base priority from parent job
        EMT      375           ;Set the job priority

        ; Copy user name and PPN
        ;
        MOV      R2,R3         ;Get # of parent job
        MOV      R1,R5         ;Get # of our job
        MUL      #6.,R3        ;Get offset to name of parent job
        MUL      #6.,R5        ;Get offset to name of our job
        ADD      #LUNAME,R3    ;Point to tables
        ADD      #LUNAME,R5
        MOV      #12.,R0       ;Get # chars to move
1$:      MOVB     (R3)+,(R5)+   ;Copy user names
        SOB      R0,1$
        MOV      LPROJ(R2),LPROJ(R1);Copy project number
        MOV      LPROG(R2),LPROG(R1);Copy programmer number
        MOV      LPROJ(R1),UPPN ;Set project number
        MOV      LPROG(R1),UPPN+2;Set programmer number

        ; Copy flags from LSW2
        ;
        MOV      LSW2(R2),LSW2(R1);Copy LSW2 flags
        MOV      LSW2S(R2),LSW2S(R1)

        ; Copy some flags from LSW5
        ;
        MOV      LSW5(R2),R0    ;Get flags from parent's LSW5
        BIC      #^C<ISPF5>,R0 ;Clear all but selected flags
        BIC      #ISPF5,LSW5(R1);Clear those flags in our LSW5
        BIS      R0,LSW5(R1)    ;Transfer flags from parent job

```

```

58
59 013032 016200 0000000 MOV LSW6(R2),R0 ;Get flags from parent's LSW6
60 013036 042700 0000000 BIC #^C<ISPF6>,R0 ;Clear all but selected flags
61 013042 042761 0000000 0000000 BIC #ISPF6,LSW6(R1) ;Clear those flags in our LSW6
62 013050 050061 0000000 BIS RO,LSW6(R1) ;Transfer flags from parent job
63
64 ; Copy some flags from LSW7
65
66 013054 016200 0000000 MOV LSW7(R2),R0 ;Get flags from parent's LSW7
67 013060 042700 0000000 BIC #^C<ISPF7>,R0 ;Clear all but selected flags
68 013064 042761 0000000 0000000 BIC #ISPF7,LSW7(R1) ;Clear those flags in our LSW7
69 013072 050061 0000000 BIS RO,LSW7(R1) ;Transfer flags from parent job
70
71 ; Copy some flags from LSW9
72
73 013076 016200 0000000 MOV LSW9(R2),R0 ;Get flags from parent's LSW9
74 013102 042700 0000000 BIC #^C<ISPF9>,R0 ;Clear all but selected flags
75 013106 042761 0000000 0000000 BIC #ISPF9,LSW9(R1) ;Clear those flags in our LSW9
76 013114 050061 0000000 BIS RO,LSW9(R1) ;Transfer flags from parent job
77
78 ; Copy some flags from LSW11
79
80 013120 016200 0000000 MOV LSW11(R2),R0 ;Get flags from parent's LSW11
81 013124 042700 0000000 BIC #^C<ISPF11>,R0 ;Clear all but selected flags
82 013130 042761 0000000 0000000 BIC #ISPF11,LSW11(R1) ;Clear those flags in our LSW11
83 013136 050061 0000000 BIS RO,LSW11(R1) ;Transfer flags from parent job
84
85 ; Copy Kmon prompt string
86
87 013142 010267 165714 MOV R2,EMCXCP+4 ;Set # of job we are copying from
88 013146 012767 0000000 165710 MOV #KMPRMT,EMCXCP+6 ;Set address of item
89 013154 012767 0000000 165704 MOV #<<MXPRMT+2>&76>,EMCXCP+8 ;Set # bytes to copy
90 013162 012700 001056' MOV #EMCXCP,RO ;Point to EMT arg block
91 013166 104375 EMT 375 ;Copy Kmon prompt string
92
93 ; Copy default printer form name
94
95 013170 012767 0000000 165666 MOV #UFORM,EMCXCP+6 ;Set address of item
96 013176 012767 000006 165662 MOV #6,EMCXCP+8 ;Set # of bytes to copy
97 013204 012700 001056' MOV #EMCXCP,RO ;Point to EMT arg block
98 013210 104375 EMT 375 ;Copy default form name
99
100 ; Copy User Command Interface (UCI) file spec
101
102 013212 012767 0000000 165644 MOV #UCISPC,EMCXCP+6 ;Set address of item to copy
103 013220 012767 000010 165640 MOV #8,EMCXCP+8 ;Set # bytes to copy
104 013226 012700 001056' MOV #EMCXCP,RO ;Point to EMT arg block
105 013232 104375 EMT 375 ;Copy UCI file spec
106
107 ; Copy subprocess start-up command file spec
108
109 013234 012767 0000000 165622 MOV #SBPSUF,EMCXCP+6 ;Set address of item to copy
110 013242 012767 000020 165616 MOV #16,EMCXCP+8 ;Set # bytes to copy
111 013250 012700 001056' MOV #EMCXCP,RO ;Point to EMT arg block
112 013254 104375 EMT 375 ;Copy subprocess start-up file spec
113
114 ; Copy print-window device information

```

```

115
116 013256 012767 000000G 165600      MOV    #JPWDEV,EMCXCP+6; Set address of print-window device
117 013264 012767 000002 165574      MOV    #2,EMCXCP+8.    ; Set # bytes to copy
118 013272 012700 001056'             MOV    #EMCXCP,R0      ; Point to EMT arg block
119 013276 104375                      EMT    375             ; Copy JPWDEV
120 013300 012767 000000G 165556      MOV    #JPWTYP,EMCXCP+6; Set address of print-window dev type
121 013306 012700 001056'             MOV    #EMCXCP,R0      ; Point to EMT arg block
122 013312 104375                      EMT    375             ; Copy JPWTYP
123 013314 012767 000000G 165542      MOV    #JPWFLG,EMCXCP+6; Set address of print-window flag word
124 013322 012700 001056'             MOV    #EMCXCP,R0      ; Point to EMT arg block
125 013326 104375                      EMT    375             ; Copy JPWFLG
126
127      ; Finihsed
128
129 013330 012605      9$:      MOV    (SP)+,R5
130 013332 012603      MOV    (SP)+,R3
131 013334 012602      MOV    (SP)+,R2
132 013336 000207      RETURN
  
```

```

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 013340 020127 000000G PRTGRT: CMP      R1,#LSTPL      ; IS THIS A REAL OR VIRTUAL LINE?
8 013344 003157          BGT      GRTVIR      ; BR IF VIRTUAL -- ABBREVIATE GREETING
9 013346 005727 000002  TST      #GREET      ; IS PRINTED GREETING WANTED?
10 013352 001403          BEQ      23$         ; BR IF NOT
11 013354          .PRINT  #GRT1             ; TSX NAME AND VERSION NUMBER
12 013362 032761 000000G 000000G 23$: BIT    #1STLG,LSW6(R1); IS THIS THE 1ST TIME THIS LINE LOGGED ON?
13 013370 001064          BNE      GRTFIN      ; BR IF NOT
14 013372 052761 000000G 000000G  BIS    #1STLG,LSW6(R1); REMEMBER 1ST LOGON HAS OCCURED
15 013400 020127 000000G  CMP      R1,#LSTPL      ; IS THIS A REAL OR VIRTUAL LINE?
16 013404 101056          BHI      GRTFIN      ; BR IF VIRTUAL -- SKIP LONG MESSAGE
17 013406 005727 000002  TST      #GREET      ; SHOULD WE PRINT GREETING MESSAGE?
18 013412 001453          BEQ      GRTFIN      ; BR IF NOT
19          ;
20          ; Calculate the message checksum.
21          ;
22 013414 012702 176532G          MOV      #SUMS-1246,R2 ; GET ADDRESS OF START OF MESSAGE
23 013420 005003          CLR      R3           ; FORM CHECKSUM IN R3
24 013422 012704 001362'          MOV      #BLKO,R4      ; TEMPORARILY STORE GREETING HERE
25 013426 112200          2$:  MOVB     (R2)+,R0        ; GET CHARACTER FROM MESSAGE
26 013430 060003          ADD      R0,R3         ; FORM CHECKSUM
27 013432 005400          NEG      R0           ; DECRYPT CHARACTER
28 013434 001424          BEQ      3$           ; BR IF END OF MESSAGE TEXT
29 013436 020227 000000G  CMP      R2,#LICTXT    ; CHECK FOR LICENSE TEXT
30 013442 103371          BHIS    2$           ; BR IF INTO THE LICENSE TEXT
31          ;
32          ; Check for various flags that indicated message variations.
33          ;
34 013444 105767 000000G          TSTB     PROFLG      ; ARE WE RUNNING ON A PROFESSIONAL?
35 013450 001003          BNE      4$           ; BR IF YES -- PRINT TRUNCATED GREETING
36 013452 005727 000001  TST      #GREET-1      ; CHECK FOR TRUNCATED GREETING MESSAGE
37 013456 001013          BNE      3$           ; BR IF FULL MESSAGE IS PRINTED
38 013460 020227 000000G  4$:  CMP      R2,#TRGRET    ; CHECK FOR TRUNCATED MESSAGE END
39 013464 103410          BLO      3$           ; CONTINUE MOVING THE GREETING MESSAGE
40 013466 101357          BHI      2$           ; GET THE NEXT CHARACTER
41 013470 112724 000056          MOVB     #'',(R4)+    ; PLACE PERIOD AT THE END OF TRUNCATED MESSAGE
42 013474 112724 000015          MOVB     #CR,(R4)+    ; PLACE CARRIAGE RETURN AFTER PERIOD
43 013500 112724 000012          MOVB     #LF,(R4)+    ; PLACE LINE-FEED AFTER PERIOD
44 013504 000750          BR       2$           ; GET THE NEXT CHARACTER
45 013506 110024          3$:  MOVB     R0,(R4)+    ; PUT IN TEMP BUFFER
46 013510 001346          BNE      2$           ; BR IF MORE OF MESSAGE TEXT
47 013512          1$:  .PRINT  #BLKO          ; PRINT GREETING MESSAGE
48 013520 005403          NEG      R3           ; DECRYPT CHECKSUM
49 013522 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 013524 020327 000000G          CMP      R3,#SUCS      ; CHECK CHECKSUM
54 013530 001404          BEQ      GRTFIN      ; BR IF OK
55          ; Checksum failure -- kill TSX
56 013532 005037 000100          CLR      @#100        ; DIE HORRIBLY...
57 013536 005037 000060          CLR      @#60

```

PRTGRT -- Print the logon greeting message

```

58      ;
59      ;   Checksum ok, set up site number.
60      ;
61 013542 105767 000000G   GRTFIN: TSTB    PROFLG      ;ARE WE RUNNING ON A PROFESSIONAL?
62 013546 001415          BEQ      1$          ;BR IF NOT
63      ;
64      ;   Print license number on a Professional
65      ;
66 013550          .PRINT  #TM$LN1          ;"License # = mmmm-TPS-"
67 013556 016705 000000G   MOV      TSXSIT,R5      ;Get the license number
68 013562 004767 000000G   CALL     PRTDEC      ;Print it
69 013566          .PRINT  #CRLF          ;Terminate the print line
70 013574 004767 000000G   CALL     DATTIM      ;Print date and time
71 013600 000441          BR       GRTVIR      ;Go print line number
72      ;
73      ;   Print license number if not on a professional
74      ;
75 013602 012767 000000G 000000G 1$: 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 013610 005727 000001     TST      #GREET-1      ;SHOULD WE PRINT DATE AND TIME?
80 013614 002451          BLT      GRTEND      ;BR IF NOT
81 013616 012704 001362'   MOV      #BLKO,R4      ;TEMPORARILY STORE GREETING HERE
82 013622 012702 000000G   MOV      #LICTXT,R2     ;OBTAIN LICENSE TEXT ADDRESS
83 013626 112200          21$: MOVB    (R2)+,R0     ;GET CHARACTER FROM MESSAGE
84 013630 005400          NEG      R0            ;DECRYPT CHARACTER
85 013632 110024          31$: MOVB    R0,(R4)+     ;PUT IN TEMP BUFFER
86 013634 001374          BNE      21$          ;BR IF MORE OF MESSAGE TEXT
87 013636 112724 000200   MOVB     #200,(R4)+     ;NO CR/LF FOLLOWING TEXT
88 013642          .PRINT  #BLKO          ;PRINT GREETING MESSAGE
89      ;
90      ;   If not supported ("S" or "s"), then print license is unsupported.
91      ;
92 013650 116700 000000G   PRTSUP: MOVB    SUPCOD,R0      ;OBTAIN THE SUPPORT CODE (SIGN EXTEND)
93 013654 005400          NEG      R0            ;DECRYPT THE CHARACTER
94 013656 120027 000123   CMPB     R0,#'S        ;CHECK FOR SUPPORTED LICENSE
95 013662 001406          BEQ      61$          ;BR IF LICENSE SUPPORTED
96 013664 120027 000163   CMPB     R0,#'s        ;CHECK FOR SUPPORTED LICENSE
97 013670 001403          BEQ      61$          ;BR IF LICENSE SUPPORTED
98 013672          .PRINT  #UNSUP          ;PRINT UNSUPPORTED LICENSE TEXT
99 013700 004767 000000G   61$: CALL     DATTIM      ;PRINT CURRENT DATE & TIME
100      ;
101      ;   Print line number.
102      ;
103 013704          GRTVIR: .PRINT  #LINNTX          ;DISPLAY LINE #
104 013712 010105          MOV      R1,R5
105 013714 006205          ASR      R5
106 013716 004767 000000G   CALL     PRTDEC
107 013722          .PRINT  #CRLF
108 013730          .TTYOUT #LF
109 013740 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 013742 010246 SETSUF: MOV     R2,-(SP)
10 013744 010346      MOV     R3,-(SP)
11          ;
12          ; Start execution of a new command file
13          ;
14 013746 004767 000000G      CALL    PUSHCF      ;SET UP FOR NEW COMMAND FILE
15          ;
16          ; Move command file name to buffer
17          ;
18 013752 012703 000000G      MOV     #CFBUF,R3      ;POINT TO COMMAND FILE BUFFER
19 013756 032761 000000G 000000G      BIT     #$DETC,LSW(R1) ;Is this a detached job?
20 013764 001004      BNE     63$      ;Br if yes
21 013766 112723 000136      MOVB    #'^,(R3)+      ;PUT '^@ AT FRONT OF COMMAND
22 013772 112723 000050      MOVB    #'',(R3)+
23 013776 112723 000100      63$: MOVB    #'@,(R3)+
24 014002 112223      5$: MOVB    (R2)+,(R3)+      ;MOVE FILE NAME TO COMMAND AREA
25 014004 001376      BNE     5$      ;LOOP TILL ALL MOVED
26 014006 112763 000015 177777      MOVB    #CR,-1(R3) ;END LINE WITH CR-LF
27 014014 112723 000012      MOVB    #LF,(R3)+
28 014020 105023      6$: CLRB    (R3)+      ;FILL REST OF BUFFER WITH NULLS
29 014022 020327 000000G      CMP     R3,#CFEND
30 014026 103774      BLO     6$
31          ;
32          ; Finished
33          ;
34 014030 012603      MOV     (SP)+,R3
35 014032 012602      MOV     (SP)+,R2
36 014034 000207      RETURN
37          ;
38          .END      START

```

Errors detected: 0

*** Assembler statistics

```

Work file reads: 0
Work file writes: 0
Size of work file: 11288 Words ( 45 Pages)
Size of core pool: 18176 Words ( 71 Pages)
Operating system: RT-11

```

```

Elapsed time: 00:01:07.19
,LP: TSKMN1=DK: TSKMN1/C/N: SYM

```

\$1STLG	1-79	29-12	29-14										
\$CCLRN	1-94	12-50	12-66	13-16	13-87	13-92	13-96	24-58					
\$CFABT	1-106	12-5	12-63										
\$CFCCCL	1-111	13-94											
\$CFDCC	1-111	12-32											
\$CFKIL	1-143	10-29	12-5	12-59									
\$CFOPN	1-116	13-51	17-57										
\$CHACT	1-69	12-33											
\$CLTST	1-102	24-46											
\$CTRLC	1-105	10-30											
\$DEBUG	1-87	12-26											
\$DEFER	1-120	10-173											
\$DETCH	1-96	10-171	30-19										
\$DIBOL	1-79	24-40											
\$DOOFF	1-113	11-26											
\$DUPRN	1-108	11-21	11-23										
\$ECHO	1-110	10-173											
\$INDAB	1-105	12-51											
\$INDDF	1-142	17-26	19-53										
\$INDRN	1-142	13-16	13-87	13-96	23-21	23-73							
\$KED	1-124	10-119	24-29										
\$KINIT	1-65	10-19	10-190										
\$LC	1-110	10-173											
\$NOIN	1-69	13-115											
\$NOINT	1-102	12-34											
\$NOVLN	1-58	10-79											
\$NOWTT	1-69	12-33	12-38										
\$NTGCC	1-112	12-35											
\$PRGLK	1-95	13-102											
\$QUIET	1-121	13-91	17-122	19-62	24-49								
\$RNIDP	1-109	12-26											
\$SETRN	1-109	11-18											
\$SLLET	1-93	10-135											
\$SNWTT	1-141	12-36											
\$SUCF	1-65	13-114	17-47										
\$SYSPS	1-68	10-149											
\$TECO	1-127	10-117	24-29	24-31									
\$UCLCF	1-117	10-125	16-26										
\$UCLCL	1-119	10-131	20-4										
\$UCLCM	1-119	10-128	19-14										
\$UCLRN	1-118	13-30	13-40	13-98	20-31								
\$UKMON	1-94	14-24	21-18										
\$UKMRN	1-94	13-98	21-12										
\$VTESE	1-90	12-25											
\$WILD	1-128	10-141	24-43										
... V1	10-92	10-92	10-99	10-99	10-99	10-108	10-108	10-109	10-109	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	10-92	10-92	10-92#	10-92#	10-99	10-99	10-99#	10-99#	10-108	10-108	10-108#	10-108#	
	10-109	10-109#	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	1-74	4-30			
AB10	1-75	4-37			
AB11	1-75	4-38			
AB12	1-75	4-39			
AB13	1-75	4-40			
AB14	1-75	4-41			
AB15	1-75	4-42			
AB16	1-75	4-43			
AB2	1-75	4-31			
AB3	1-75	4-32			
AB4	1-75	4-33			
AB5	1-75	4-34			
AB6	1-75	4-35			
ABCMD	1-77	18-17			
ABM1	1-74	4-28			
ABM10	1-74	4-21			
ABM11	1-73	4-20			
ABM12	1-73	4-19			
ABM13	1-73	4-18			
ABM14	1-73	4-17			
ABM15	1-73	4-16			
ABM16	1-73	4-15			
ABM17	1-73	4-14			
ABM2	1-74	4-27			
ABM20	1-72	4-13			
ABM21	1-72	4-12			
ABM22	1-72	4-11			
ABM23	1-72	4-10			
ABM24	1-72	4-9			
ABM3	1-74	4-26			
ABM4	1-74	4-25			
ABM5	1-74	4-24			
ABM6	1-74	4-23			
ABM7	1-74	4-22			
ABRTAD	1-62	11-50	11-59*		
ABRTCD	1-62	11-30	11-60*		
ABRTCF	1-151	12-49	12-65	13-20	
ABRTOV	1-62	11-56			
ABTMSG	4-29#	11-37			
ACRFN	1-151	17-13	19-26	19-73	22-22
ADM3A	1-136	26-37			
ADM3FL	1-137	26-41			
ADM3NO	1-138	26-40			
AFCF	1-59	25-28*			
ALCDEV	1-50	7-49#	8-49	8-54	8-59
ALCERR	1-72	4-7			
ALDBLK	1-168	7-27#			
ALDEMT	1-50	8-48#			
ALDEX	1-168	7-26#			
ALFN	1-177				
AMBOPT	1-159				
ASDEX	1-156	7-2#			
ASKLNM	1-155	7-38#			
AUTHFN	1-172	7-28#			
BADCMD	1-155	19-78	19-85#		
BELL	2-16#				

[illegible]

[illegible]

DKSAV	1-150	7-5#										
DLCMT	1-50	8-53#										
DMTARG	1-156	8-30#										
DOLRAT	9-19#	15-9*	15-71*	17-28	17-129*							
DORUN	1-148	19-83										
DOTAT	9-20#	15-10*	15-84*	17-21	17-130*							
EDIT	1-79	10-113										
EDTFIL	1-166	11-46										
EM\$NUC	1-59	16-13										
EM\$NUK	1-52	21-19										
EM\$OVL	1-71	4-5										
EM\$SFP	1-71	4-4										
EMCXP	8-139#	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			
ERRSEV	1-128	10-59*	12-47									
ERRSPC	1-103	11-41	11-45	12-4*								
ESC	1-52	2-19#										
EXCJOB	1-57	10-14*										
FF	2-18#	15-20	24-16									
FILERM	1-76	11-43										
FILNAM	1-152	7-41#	17-38	17-43	19-28	19-33	19-41*	19-43	19-76	19-80	25-16	
FKILL	1-151	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	1-151	10-27	17-49									
GENMON	1-53	8-182#	10-207*	10-208	10-212*	10-213	12-74					
GENTOP	1-104	10-25										
GETSYP	1-61	10-151										
GOTCML	15-124	16-6#										
GREET	1-46	2-9#	29-9	29-17	29-36	29-79						
GRT1	1-133	29-11										
GRTEND	29-80	29-109#										
GRTFIN	1-46	29-13	29-16	29-18	29-54	29-61#						
GRTINI	10-172	10-178#										
GRTVIR	29-8	29-71	29-103#									
HANCHN	1-79	2-21#										
HANIDX	1-160	7-46#										
HAZEL	1-88	26-56										
HAZLFL	1-88	26-60										
HAZLNO	1-88	26-59										
HNBUF	1-160	7-34#										
HUPARG	1-174	8-102#										
IDNCMD	13-44	16-33	18-10#									
II\$SZ	1-71	8-83	8-91									
II\$FLG	1-59	25-28										
II\$NPV	1-59	25-25										
II\$PRV	1-59	25-24										
IIBUF	1-45	3-9#	8-84	8-92	25-24	25-25	25-28					
ILLCMD	1-157	15-105	19-85									
ILSW2	1-92	10-79										
IMADM3	1-76	26-39										
IMHAZL	1-76	26-58										

IN\$CNT	1-142	14-18										
INBUF	9-7#	14-32	15-13	15-127	17-124							
INDABT	1-149	12-53	12-61									
INDACT	1-153	23-16										
INDCMD	16-8	16-19	17-6#									
INDERR	1-106	12-46*	23-61*	23-71*								
INDINI	17-34	19-58	22-30	23-10#								
INDNAM	7-31#	23-69										
INDRFL	7-39#	15-7*	15-66	15-70*	15-79	15-83*	15-96	15-98*	16-7	16-18	17-7	17-19
	17-112	17-128*										
INDRUN	14-19	23-68#										
INDSAV	1-142	17-23	22-10	23-68								
INDSTA	1-106	14-14	17-30	23-14	23-62*	23-74*						
INGADR	1-45	8-82#										
INGEMT	1-45	8-80#										
INIUKD	1-62	27-46										
INPADR	1-45	8-90#										
INPEMT	1-45	8-88#										
INSCF	17-61	23-25	25-12#									
INSSRC	1-59	25-17										
ISPF11	1-60	28-81	28-82									
ISPF5	1-60	28-53	28-54									
ISPF6	1-60	28-60	28-61									
ISPF7	1-60	28-67	28-68									
ISPF9	1-60	28-74	28-75									
JPWDEV	1-61	28-116										
JPWFLG	1-64	28-123										
JPWTYP	1-61	28-120										
JS\$KMN	1-71	12-75										
JS\$LOG	1-73	10-212										
JS\$ON	1-71	10-207										
JSWLOC	1-64	11-61*	12-8*	13-5	13-18	13-41*	13-42*	13-65*	13-89	13-97*	14-5*	
KCSIBF	1-155	9-6#										
KCSIMS	1-156	3-8#										
KDOCIN	1-121	12-11										
KEYBUF	1-47	9-3#										
KEYEND	1-49	9-4#										
KILEMT	1-172	8-114#										
KL3CLR	1-95	12-31										
KL4CLR	1-123	12-24										
KMINIT	1-57	10-180										
KMNNAM	1-175	7-29#										
KMNOT1	1-46	10-21	10-211	11-6#								
KMONCE	1-57	10-12	10-178									
KMPRMT	1-63	1-130	10-52*	10-53*	14-32	15-127	28-88					
KMSTK	1-6	1-49	9-11#									
L	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#

MAKWIN	8-152#	27-23											
MAPWIN	8-160#	27-28											
MAXMEM	1-64	10-48*											
MAXPRI	1-64	10-51											
MDT	1-8	1-88											
MINTIM	1-101	10-28											
MISUCL	1-77	20-37											
MNBASE	1-167	8-171#											
MNBPC	1-166	8-173#											
MNFLGS	1-166	8-174#											
MNTARG	1-174	8-25#	10-156										
MNTDEV	1-156	7-47#	8-26	8-31	10-155*								
MNTOP	1-167	8-172#											
MONAR1	1-167	8-170#											
MONAR2	1-167	8-178#											
MONTXT	1-76	11-38											
MSGBUF	1-170	8-42	8-43#										
MSGEND	1-170	8-44#											
MSTALC	1-72	4-6											
MXJPRI	1-144	10-51*											
MXPRMT	1-130	28-89											
NAMTOP	1-133	10-49											
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	6-85	6-85		
NEWCMD	13-111	13-113	14-5#										
NOABRT	11-31	12-4#											
NOCF	1-77	17-49	17-52										
NOCIN	1-155	12-10	12-17#										
NOCMD	18-16	19-13#											
NOFLAG	1-157	9-18#											
NOICMD	17-40	17-52#											
NOIND	1-161	17-25	22-12										
NOUCL	9-22#	13-43*	15-12*	16-32*	19-16	20-6							
NSPLDV	1-119	10-90											
NUMPRM	1-77	17-134											
OCTPRT	1-176	11-51											
OFFEMT	1-172	8-96#											
OKFAND	1-139	10-66*											
OKFEND	1-102	10-67											
OKFILE	1-102	10-66											
OKFNND	1-139	10-67*											
OPRCMD	1-84	6-61											
OVLREQ	1-176	11-55											
P2\$VIR	1-58	10-81											
PASLIN	1-128	12-8	13-5	13-41	13-65	13-97							

[illegible]

RESDEV	1-139	10-65*										
RJEMT	1-131	8-126#										
RSTPRV	1-71	10-86	11-20	25-29								
RUNARG	1-70	11-12*										
RUNCHN	1-67	11-19	11-19	19-80	19-80	20-16	20-16	21-7	21-7	23-68	23-68	24-55
	24-55											
RUNEMT	1-154	8-4#										
RUNNAM	1-67	23-64										
SBPSUF	1-61	10-199	28-109									
SC\$ERR	1-58	10-59										
SCHAIN	1-120	12-8	13-5	13-18	13-41	13-65	13-89	13-97				
SCNCMD	15-13#	15-128	17-131									
SEARCH	1-150	18-12										
SERFLG	1-63	11-11*	17-42*	17-44*	19-32*	19-34*	19-42*	19-44*	19-79*	19-81*		
SETSUF	10-202	13-117	30-9#									
SIZEMT	1-174	8-20#										
SIZVAL	1-155	8-21#										
SJEMT	1-131	8-120#										
SKPSPC	1-60	22-17										
SMONHD	1-72	12-72										
SPGEMT	1-169	8-9#										
SPLCHN	1-99	10-92										
SPLHLA	1-163	8-132#										
SPUBUF	1-66	10-50										
START	1-48	8-110	10-5#	30-38								
STDNAM	1-76	10-54										
STRLEN	1-77	17-136										
SUCF2	1-58	13-110	13-116	13-118*								
SUCS	1-133	29-53										
SUMS	1-133	29-22										
SUPCOD	1-133	29-92										
SXBPNT	1-66	10-50*										
SYINTX	9-24#	23-33										
SYSAV	1-150	7-4#	19-72	23-63								
SYTXT	9-23#	19-50										
TAB	2-17#	15-18	15-117	24-14								
TALEMT	1-50	8-58#										
TECO	1-79	10-115										
TM\$LN1	1-78	29-66										
TOMPRM	17-89	17-134#										
TOOLNG	1-77	13-66										
TRGRET	1-133	29-38										
TRMINI	10-161	26-9#										
TRYUCL	19-82	20-4#										
TSKMN1	1-11#	1-51										
TSKMON	1-10#	1-46	1-51									
TSXLN	1-133	29-75										
TSXSIT	1-133	29-67	29-75*									
UC\$MDC	1-147	10-107*										
UC\$NDC	1-147	10-106*										
UCHAN	1-111	11-13	11-15*									
UCIDF	1-52	3-4#										
UCISPC	1-91	21-6	28-102									
UCLBLK	1-146	10-97	16-11	16-28	19-18							
UCLCMD	1-85	6-75										
UCLDAT	1-146	10-99										

[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	10-92	10-99	10-108	17-43	17-65	19-33	19-43	19-80	20-16	21-7	23-68	24-55
...CM2	10-92	10-99	10-99	10-108	10-108	10-108	10-108	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	10-109	11-19	19-55									
...CM5	10-92	10-99	10-108	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	10-108	17-65										
.CLOSE	1-39#	10-109	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#	10-99	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#	10-92	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#	10-108										
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	6-85						
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#	10-27	17-49									
TBLDEF	5-26#	6-8										
TBLEND	5-52#	6-86										