Table of contents

```
                        .TITLE  TSKM2C -- The RUN command
                        .ENABL  LC
                        .DSABL  GBL
                        .CSECT  TSKM2C
                TSKM2C:
                ;
                ;  TSKM2C is the portion of TSKMON that contains the code
                ;  to implement the RUN command.
                ;
                ;  Copyright 1986.
                ;  S&H Computer Systems, Inc.
                ;  Nashville, Tennessee
                ;
                ;
                ;  Macro calls
                ;
                        .MCALL  .LOOKUP,.READW,.CLOSE
                ;  Global definitions
                ;
                        .GLOBL  CMDRSY,CMDRUN,DORUN,RUNNAM,PLOAD,KDOCIN
                ;
                ;  Global references
                ;
                        .GLOBL  CMDOFF,NOPRG,FPRINT,SYNAME,INDSAV,NOIND
                        .GLOBL  $DEBUG,$HITTY,$INDRN,$NOINT,PO$DBG
                        .GLOBL  $NOWTT,$PRGLK,$RNIOP,$RNMLK,$TRNSP,$UCLRN
                        .GLOBL  AF$BYA,AF$CCA,AF$DBG,AF$DUP,AF$HIE,TRMSTR
                        .GLOBL  AF$IND,AF$IOP,AF$MEM,AF$NOI,AF$NOW,LSW4
                        .GLOBL  AF$PLK,AF$SCA,AF$SET,AF$TPO,AF$UCL,TRMSTR
                        .GLOBL  $CFOPN,CFBUF,CFSQEZ,PUSHCF,ACRFN,TOOLNG
                        .GLOBL  II$FLG,II$NPV,II$PRV,$QUIET
                        .GLOBL  DKSAV,SYSAV,CORUSR,EM$NPD,AF$NPWFILNAM
                        .GLOBL  VDBFLG,CINDAT,LOGASN,$STSNG,LSW6,FORCEO
                        .GLOBL  RUNARG,RUNCHN,RUNFLG,RUNDEV,INSSRC,PVNPW
                        .GLOBL  RUNHD,RUNEMT,JS$RUN,IIBUF
                        .GLOBL  LSW7,LITIME,PO$MEM,LSW2,LSW2S
                        .GLOBL  PRIVCO,EM$MPV,EM$NAL,PO$LOK
                        .GLOBL  CCSPRV,ABRTCF,PO$DBG,EM$NAD
                        .GLOBL  $NOWIN,LSW11,$DUPRN,IN$ACT,INDSTA
                        .GLOBL  IN$CNT,INDACT,UCLBLK,EM$NUC
                        .GLOBL  $SETRN,LSW9,RDERM,CINFLG,CVTTAB
                        .GLOBL  RSTPRV,LJSW,NEWJSW,R50VIR,VIMAGE,MAXMEM
                        .GLOBL  ODTBAS,VSWPFLMXJADR,PRGSIZ,PRGTOP
                        .GLOBL  USRSTK,NOSTRT,USTART,LPRG1,LPRG2,LMONHD
                        .GLOBL  SMONHD,GENMON,OVRCOR,BADSAV,LDNAM,LDCLEN
                        .GLOBL  NOCIN,SKPSPC,SEARCH,INVOPT,FKILL
                        .GLOBL  XAREA,FILNAM,LSW5,$SCCA,BLKO,BLKWDS
                        .GLOBL  LSW3,$CHACT,RDCMD,MXJADR,FIXPRV,AFCF
                ;
000000  000000  RUNOPS: .WORD   0                ;AF$xxx flags from RUN switches
```

```
                        ;-------------------------------------------------
                        ;  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 print a warning message
                        ;
                              .MACRO  FWARN   MSG
                              MOV     R5,-(SP)
                              MOV     MSG,R5
                              CALL    PRTWRN
                              MOV     (SP)+,R5
                              .ENDM   FWARN
                        ;
                        ;-------------------------------------------------------------
                        ;  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  CMDV2C
                        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  NAME2C
                        L     =       .
                              .ASCIZ  /STRNG/
                              .CSECT  CMDV2C
                              .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  CMDV2C
                                .WORD   0
                                .CSECT  TSKM2C
                                .ENDM   TBLEND
```

```
                        .SBTTL  Tables of commands and options
                ;-----------------------------------------------------------
                ;  Define option switches for the RUN command
                ;
                        TBLDEF  RUN,1
                        CMDDEF  B*YPASN,RNBYA
                        CMDDEF  D*EBUG,RNDB
                        CMDDEF  H*IGH,RNSHI
                        CMDDEF  I*OPAGE,RNSIOP
                        CMDDEF  L*OCK,RNSLK
                        CMDDEF  M*EMLOCK,RNSMLK
                        CMDDEF  NONI*NTERACTIVE,RNSNOI
                        CMDDEF  NOW*INDOWING,RNNPW
                        CMDDEF  O*DT,RNDB
                        CMDDEF  S*INGLECHAR,RNSCHR
                        CMDDEF  SC*CA,RNSCCA
                        CMDDEF  T*RANSPARENT,RNSTPO
                        TBLEND
```

```
                                .SBTTL  RUN command
                        ;-----------------------------------------------------------
                        ;  Process the 'R' (RUN) command.
                        ;
                        ;  Entry point for 'RUN' command
                        ;
000002  012705  000000G CMDRUN: MOV     #DKSAV,R5       ;SET DEFAULT DEV AND EXT
000006  000402                 BR      RUNPRS
                        ;
                        ;  Entry point for 'R' command
                        ;
000010  012705  000000G CMDRSY: MOV     #SYSAV,R5       ;SET DEFAULT DEV AND EXT
                        ;
                        ;  Begin scanning command line
                        ;
000014  004767  000000G RUNPRS: CALL    CVTTAB          ;CONVERT TABS TO SPACES IN COMMAND LINE
000020  005067  177754          CLR     RUNOPS          ;Clear option flag word
                        ;
                        ;  See if any switches were specified with run command
                        ;
000024  004767  000000G RUNSW:  CALL    SKPSPC          ;Skip over any spaces
000030  121327  000057          CMPB    (R3),#'/        ;ANY SWITCH?
000034  001100                 BNE     RUNNAM          ;BR IF NOT
000036  005203                 INC     R3              ;POINT PAST SLASH
000040  012704  000000'         MOV     #RUNHD,R4       ;POINT TO RUN SWITCH TABLE
000044  004767  000000G         CALL    SEARCH          ;ACCRUE AND LOOKUP SWITCH
000050  103004                 BCC     1$              ;BR IF SWITCH OK
000052                         FABORT  #INVOPT         ;BAD SWITCH
000062  116701  000000G 1$:     MOVB    CORUSR,R1       ;PUT JOB INDEX NUMBER IN R1
000066  000134                 JMP     @(R4)+          ;ENTER SWITCH PROCESSING ROUTINE
                        ;
                        ;  Process the /LOCK switch
                        ;
000070  052767  000000G 177702 RNSLK:  BIS     #AF$PLK,RUNOPS  ;Remember to lock program to line
000076  000752                 BR      RUNSW           ;GO CHECK FOR OTHER SWITCHES
                        ;
                        ;  Process the /MEMLOCK switch
                        ;
000100  052767  000000G 177672 RNSMLK: BIS     #AF$MEM,RUNOPS  ;Remember to lock program in memory
000106  000746                 BR      RUNSW           ;Go check for more switches
                        ;
                        ;  Process the /SINGLECHAR switch.
                        ;
000110  052767  000000C 177662 RNSCHR: BIS     #<AF$SCA!AF$NOW>,RUNOPS ;Enable single char activ & non-wait
000116  000742                 BR      RUNSW
                        ;
                        ;  Process the /TRANSPARENT switch.
                        ;
000120  052767  000000G 177652 RNSTPO: BIS     #AF$TPO,RUNOPS  ;Say to run with transparent output
000126  000736                 BR      RUNSW
                        ;
                        ;  Process the /BYPASS (bypass assign) option
                        ;
000130  052767  000000G 177642 RNBYA:  BIS     #AF$BYA,RUNOPS  ;Bypass assign commands
000136  000732                 BR      RUNSW
                        ;
                        ;  Process the /HIGH (high-efficiency) switch.
```

```
                                ;
000140  052767  000000G 177632  RNSHI:  BIS     #AF$HIE,RUNOPS  ;Enable high-efficiency mode
000146  000726                         BR      RUNSW
                                ;
                                ;  Process the /IOPAGE switch
                                ;
000150  052767  000000G 177622  RNSIOP: BIS     #AF$IOP,RUNOPS  ;Program is to be run with I/O page access
000156  000722                         BR      RUNSW
                                ;
                                ;  Process the /NONINTERACTIVE switch
                                ;
000160  052767  000000G 177612  RNSNOI: BIS     #AF$NOI,RUNOPS  ;Run program in non-interactive mode
000166  000716                         BR      RUNSW
                                ;
                                ;  Process the /DEBUG switch
                                ;
000170  105767  000000G         RNDB:   TSTB    VDBFLG          ;Was program debugger genned into the system?
000174  001004                         BNE     1$              ;Br if yes
000176                                  FABORT  #EM$NPD         ;No program debugger available
000206  052767  000000G 177564  1$:     BIS     #AF$DBG,RUNOPS  ;Remember to run program with debugger
000214  000703                         BR      RUNSW
                                ;
                                ;  Process the /SCCA switch
                                ;
000216  052767  000000G 177554  RNSCCA: BIS     #AF$CCA,RUNOPS  ;Remember to suppress control-C aborts
000224  000677                         BR      RUNSW
                                ;
                                ;  Process the /NOWINDOW switch
                                ;
000226  052767  000000G 177544  RNNPW:  BIS     #AF$NPW,RUNOPS  ;Remember to suspend windowing
000234  000673                         BR      RUNSW
```

```
                                        ;
                                        ;  Accrue name of program to run
                                        ;
000236  004767  000000G          RUNNAM: CALL    ACRFN           ;ACCRUE THE PROGRAM FILE NAME
000242  103002                          BCC     1$              ;BR IF GOT NAME OK
000244  000167  001254                  JMP     CLSEXT          ;BR IF ERROR IN GETTING FILE NAME
000250  004767  002240          1$:     CALL    CHKIND          ;SEE IF WE ARE RUNNING IND
000254                                  .LOOKUP #XAREA,#RUNCHN,#FILNAM ;TRY TO OPEN FILE
000274  103020                          BCC     DORUNX          ;BRANCH IF OPEN OK
000276  032767  000000G 177474          BIT     #AF$PLK,RUNOPS  ;Was program supposed to be locked?
000304  001402                          BEQ     2$              ;NO, GO AHEAD
000306  000167  000000G                  JMP     CMDOFF          ;LOCKED FILE NOT FOUND, LOG OFF
000312                          2$:     FERR    #NOPRG          ;PRINT ERROR MESSAGE
000326  000167  001172                  JMP     CLSEXT          ;GO CLEANUP AND GET NEXT COMMAND
000332  005067  177442          DORUN:  CLR     RUNOPS          ;NO RUN OPTION FLAGS
000336  116701  000000G          DORUNX: MOVB    CORUSR,R1       ;GET LINE INDEX #
000342  016161  000000G 000000G          MOV     LSW2(R1),LSW2S(R1);SAVE VALUE OF LSW2
                                        ;
                                        ;  See if an argument was specified with run command
                                        ;
000350  012767  000001  000010G          MOV     #1,CINDAT+10    ;Set 1 to go in 510 chain area
000356  005067  000012G                  CLR     CINDAT+12       ;Set null at run argument string
000362  105267  000000G                  INCB    RUNARG          ;Set flag saying RUN argument string present
000366  112300                  1$:     MOVB    (R3)+,R0        ;ANY ARGUMENT FIELD PRESENT?
000370  001502                          BEQ     RNOARG          ;BR IF NOT
000372  120027  000040                  CMPB    R0,#' '         ;IGNORE SPACES
000376  001773                          BEQ     1$
                                        ;
                                        ;  There is an argument field.
                                        ;  Move argument string to chain area.
                                        ;
000400  005303                          DEC     R3              ;Point to start of string
000402  010302                          MOV     R3,R2           ;Get pointer to argument string
000404  012704  000012G                  MOV     #CINDAT+12,R4   ;Point to where chain data for 512 starts
000410  020427  000270G          7$:     CMP     R4,#CINDAT+270  ;Have we hit end of chain data area?
000414  103002                          BHIS    8$              ;Don't go past end of chain data area
000416  112224                          MOVB    (R2)+,(R4)+     ;Copy argument to chain data area
000420  001373                          BNE     7$              ;Loop till end reached
000422  162704  000012G          8$:     SUB     #CINDAT+12,R4   ;Get # chars in string (including null)
000426  010467  000010G                  MOV     R4,CINDAT+10    ;It will be passed in 510
                                        ;
                                        ;  Set argument up to be read like it was part of a command file.
                                        ;  If we have pending commands in the command file buffer, compress
                                        ;  them to make room for new commands.
                                        ;  If input is coming from a real command file, we do not need to
                                        ;  compress since we will reread buffer when we hit end of new command.
                                        ;
000432  032761  000000G 000000G          BIT     #$CFOPN,LSW4(R1);Is a command file open?
000440  001403                          BEQ     11$             ;Br if not
000442  012705  001000G                  MOV     #CFBUF+512.,R5  ;Say entire command file buffer is free
000446  000403                          BR      12$
000450  004767  000000G          11$:    CALL    CFSQEZ          ;Compress info in current command file buffer
000454  010005                          MOV     R0,R5           ;Save address of end of free space in buffer
                                        ;
                                        ;  Push current command file and start new one
                                        ;
000456  004767  000000G          12$:    CALL    PUSHCF          ;START NEW COMMAND FILE
```

RUN command

```
     58 000462  012704  000000G                 MOV     #CFBUF,R4       ;POINT TO COMMAND FILE BUFFER
     59                                 ;
     60                                 ;  See if arg string contains a space which separates input file
     61                                 ;  specs from output file specs.
     62                                 ;
     63 000466  010302                  2$:     MOV     R3,R2           ;POINT TO START OF ARG STRING
     64 000470  112200                          MOVB    (R2)+,R0        ;GET CHAR FROM STRING
     65 000472  001416                          BEQ     3$              ;NO SPACE IN STRING
     66 000474  120027  000040                  CMPB    R0,#' '         ;SEARCH FOR SPACE
     67 000500  001373                          BNE     2$
     68                                 ;
     69                                 ;  String contains a space. Move part of string following space
     70                                 ;  to front of command file buffer and insert trailing '='.
     71                                 ;
     72 000502  032761  000000G 000000G         BIT     #$INDRN,LSW5(R1);Is IND being started?
     73 000510  001007                          BNE     3$              ;Br if yes -- Don't reverse parameters
     74 000512  105062  177777                  CLRB    -1(R2)          ;MAKE 1ST PART BE ASCIZ
     75 000516  112224                  4$:     MOVB    (R2)+,(R4)+     ;MOVE STRING TO COMMAND FILE BUFFER
     76 000520  001376                          BNE     4$              ;LOOP TILL DONE
     77 000522  112764  000075  177777         MOVB    #'=,-1(R4)      ;PUT IN SEPARATING =
     78                                 ;
     79                                 ;  Now move 1st part of string to command file buffer
     80                                 ;
     81 000530  112324                  3$:     MOVB    (R3)+,(R4)+
     82 000532  001376                          BNE     3$
     83                                 ;
     84                                 ;  Put in CR-LF-^C to terminate the line
     85                                 ;
     86 000534  012703  000000G                 MOV     #TRMSTR,R3      ;POINT TO STRING OF CONTROL CHARS
     87 000540  005304                          DEC     R4              ;STORE OVER TRAILING NULL
     88 000542  112324                  5$:     MOVB    (R3)+,(R4)+     ;PUT IN TERMINATING CHARS
     89 000544  001376                          BNE     5$
     90                                 ;
     91                                 ;  Pad rest of buffer with nulls
     92                                 ;
     93 000546  020405                  6$:     CMP     R4,R5           ;Reached end of area to fill?
     94 000550  001407                          BEQ     13$             ;Br if yes
     95 000552  103002                          BHIS    14$             ;Br if buffer overflow
     96 000554  105024                          CLRB    (R4)+           ;Null fill the buffer
     97 000556  000773                          BR      6$
     98 000560                          14$:    FABORT  #TOOLNG         ;Buffer overflow
     99                                 ;
    100                                 ;  Set flags to suppress listing of argument string being read.
    101                                 ;
    102 000570  052761  000000G 000000G 13$:     BIS     #$QUIET,LSW4(R1);SUPPRESS LISTING
    103                                 ;
    104                                 ;  Program file is open.
    105                                 ;
    106 000576  012704  000000G         RNOARG: MOV     #FILNAM,R4      ;POINT TO CELL WITH FILE NAME
    107 000602  016700  177172                  MOV     RUNOPS,R0       ;Get run option flags for PLOAD
    108 000606  000400                          BR      PLOAD           ;Load and start execution of program
```

```
      1                                 ;-----------------------------------------------------------------
      2                                 ;  Load a SAV file into memory and start its execution.
      3                                 ;
      4                                 ;  Inputs:
      5                                 ;   RUNCHN channel is now open to program file.
      6                                 ;   R0 = AF$xxx run option flags.
      7                                 ;   R1 = Job index number.
      8                                 ;   R4 = Address of buffer with RAD50 program name.
      9                                 ;
     10 000610  010067  177164          PLOAD:  MOV     R0,RUNOPS       ;Save initial run option flags
     11                                 ;
     12                                 ;  Set name of running program
     13                                 ;
     14 000614  012705  000000G                 MOV     #RUNDEV,R5      ;Save program name here
     15 000620  012700  000004                  MOV     #4,R0           ;Save 4 words
     16 000624  012425                  1$:     MOV     (R4)+,(R5)+     ;Copy program file spec
     17 000626  077002                          SOB     R0,1$
     18 000630  012705  000000G                 MOV     #RUNDEV,R5      ;Point to file spec
     19 000634  004767  000000G                 CALL    LOGASN          ;Do any logical assignment
     20 000640  012704  000000G                 MOV     #RUNDEV,R4      ;Point to file spec
```

```
                                        ;  See if we need to invoke some automatic run switches for this program.
                                        ;
                                        ;  See if we should default to single character activation
                                        ;
000644  032761  000000G 000000G RUNASW: BIT     #$STSNG,LSW6(R1);Is single char activation the default?
000652  001403                          BEQ     7$              ;Br if not
000654  052767  000000G 177116          BIS     #AF$SCA,RUNOPS  ;Remember to enable single char activation
                                        ;
                                        ;  See if the program has been installed
                                        ;
000662  010346                  7$:     MOV     R3,-(SP)
000664  012703  000000G                 MOV     #RUNDEV,R3      ;Point to file spec for program
000670  004767  000000G                 CALL    FORCEO          ;Make sure unit number is included
000674  010300                          MOV     R3,R0           ;Copy file spec ptr
000676  012603                          MOV     (SP)+,R3
000700  004767  000000G                 CALL    INSSRC          ;See if this program has been installed
000704  103420                          BCS     6$              ;Br if not
                                        ;
                                        ;  Apply privilege flags specified for installed program
                                        ;
000706  012702  000000C                 MOV     #<PVNPW-1>*2,R2 ;Get index to last privilege word
000712  056262  000000C 000000G 4$:     BIS     II$PRV+IIBUF(R2),PRIVCO(R2);Set flags specified for program
000720  046262  000000C 000000G         BIC     II$NPV+IIBUF(R2),PRIVCO(R2);Clear flags specified for program
000726  162702  000002                  SUB     #2,R2           ;Get index to next word
000732  002367                          BGE     4$              ;Loop if more to do
000734  004767  000000G                 CALL    FIXPRV          ;Do privilege setup
000740  056767  000001C 177032          BIS     II$FLG+IIBUF,RUNOPS ;Combine run option switches for program
                                        ;
                                        ;  Or in attribute flags for enclosing command file
                                        ;
000746  056767  000000G 177024  6$:     BIS     AFCF,RUNOPS     ;Combine attributes from command file
                                        ;
                                        ;  Set some automatic switches for this program
                                        ;
000754  016702  177020                  MOV     RUNOPS,R2       ;Get attribute flags for program
000760  032702  000000G                 BIT     #AF$SCA,R2      ;Single character activation wanted?
000764  001403                          BEQ     3$              ;Br if not
000766  052761  000000G 000000G         BIS     #$CHACT,LSW5(R1);Enable single character activation
000774  032702  000000G          3$:    BIT     #AF$NOW,R2      ;Non wait TT input wanted?
001000  001403                          BEQ     15$             ;Br if not
001002  052761  000000G 000000G         BIS     #$NOWTT,LSW5(R1);Enable non-wait TT input
001010  032702  000000G         15$:    BIT     #AF$HIE,R2      ;Is high efficiency mode wanted?
001014  001403                          BEQ     16$             ;Br if not
001016  052761  000000G 000000G         BIS     #$HITTY,LSW4(R1);Enable high-efficiency mode
001024  032702  000000G         16$:    BIT     #AF$TPO,R2      ;Transparent output wanted?
001030  001403                          BEQ     23$             ;Br if not
001032  052761  000000G 000000G         BIS     #$TRNSP,LSW3(R1);Enable transparent output
001040  032702  000000G         23$:    BIT     #AF$NOI,R2      ;Is non-interactive execution wanted?
001044  001406                          BEQ     1$              ;Br if not
001046  052761  000000G 000000G         BIS     #$NOINT,LSW7(R1);Set non-interactive execution flag
001054  012761  000002  000000G         MOV     #2,LITIME(R1)   ;Set very short interactive time
001062  032702  000000G          1$:    BIT     #AF$IOP,R2      ;Map to I/O page?
001066  001417                          BEQ     17$             ;Br if not
001070  032767  000000G 000000G         BIT     #PO$MEM,PRIVCO  ;Are we authorized to access I/O page?
001076  001010                          BNE     20$             ;Br if yes
001100                                  FERR    #EM$MPV         ;Not authorized to run this program
001114  000167  000404                  JMP     CLSEXT
```

RUN command

```
     58 001120  052761  000000G 000000G 20$:    BIS     #$RNIOP,LSW9(R1);Set flag to cause mapping to I/O page
     59 001126  032702  000000G         17$:    BIT     #AF$MEM,R2       ;Lock program in memory?
     60 001132  001417                         BEQ     18$              ;Br if not
     61 001134  032767  000000G 000000G         BIT     #PO$LOK,PRIVCO   ;Are we authorized to do this?
     62 001142  001010                         BNE     21$              ;Br if yes
     63 001144                                 FERR    #EM$NAL          ;Not authorized to lock program in memory
     64 001160  000167  000340                 JMP     CLSEXT
     65 001164  052761  000000G 000000G 21$:    BIS     #$RNMLK,LSW9(R1);Set flag to cause program to be locked
     66 001172  032702  000000G         18$:    BIT     #AF$PLK,R2       ;Want to lock program to line?
     67 001176  001407                         BEQ     22$              ;Br if not
     68 001200  052761  000000G 000000G         BIS     #$PRGLK,LSW5(R1);Remember program is locked to line
     69 001206  004767  000000G                 CALL    CCSPRV           ;Copy current to set privileges
     70 001212  004767  000000G                 CALL    ABRTCF           ;Close any open command files
     71 001216  032702  000000G         22$:    BIT     #AF$DBG,R2       ;Want to run program with debugger?
     72 001222  001417                         BEQ     24$              ;Br if not
     73 001224  032767  000000G 000000G         BIT     #PO$DBG,PRIVCO   ;Are we privileged to use the debugger?
     74 001232  001010                         BNE     19$              ;Br if yes
     75 001234                                 FERR    #EM$NAD          ;Can't use debugger
     76 001250  000167  000250                 JMP     CLSEXT
     77 001254  052761  000000G 000000G 19$:    BIS     #$DEBUG,LSW9(R1);Enable the debugger
     78 001262  032702  000000G         24$:    BIT     #AF$CCA,R2       ;Disable control-C abort?
     79 001266  001403                         BEQ     32$              ;Br if not
     80 001270  052761  000000G 000000G         BIS     #$SCCA,LSW5(R1) ;Disable control-C abort
     81 001276  032702  000000G         32$:    BIT     #AF$NPW,R2       ;Suspend process windowing?
     82 001302  001403                         BEQ     30$              ;Br if not
     83 001304  052761  000000G 000000G         BIS     #$NOWIN,LSW11(R1);Suspend process windowing
     84                                        ;
     85                                        ;  Check for startup of special programs
     86                                        ;
     87 001312  032702  000000G         30$:    BIT     #AF$DUP,R2       ;Is DUP being started?
     88 001316  001403                         BEQ     25$              ;Br if not
     89 001320  052761  000000G 000000G         BIS     #$DUPRN,LSW6(R1);Remember DUP is running
     90 001326  032702  000000G         25$:    BIT     #AF$IND,R2       ;Is IND being started?
     91 001332  001423                         BEQ     26$              ;Br if not
     92 001334  132767  000000G 000000G         BITB    #IN$ACT,INDSTA   ;Is IND already active?
     93 001342  001414                         BEQ     27$              ;Br if not
     94 001344  132767  000000G 000000G         BITB    #IN$CNT,INDSTA   ;Are we continuing its execution?
     95 001352  001010                         BNE     27$              ;Br if yes
     96 001354                                 FERR    #INDACT          ;Error -- IND is already active
     97 001370  000167  000130                 JMP     CLSEXT
     98 001374  052761  000000G 000000G 27$:    BIS     #$INDRN,LSW5(R1);Remember IND is running
     99 001402  032702  000000G         26$:    BIT     #AF$UCL,R2       ;Is TSXUCL being started?
    100 001406  001412                         BEQ     28$              ;Br if not
    101 001410  005767  000000G                 TST     UCLBLK           ;Are we supporting user-defined commands?
    102 001414  001004                         BNE     31$              ;Br if yes
    103 001416                                 FABORT  #EM$NUC          ;Not supporting user-defined commands
    104 001426  052761  000000G 000000G 31$:    BIS     #$UCLRN,LSW7(R1);Remember TSXUCL is running
    105 001434  032702  000000G         28$:    BIT     #AF$SET,R2       ;Is SETUP being started?
    106 001440  001403                         BEQ     2$               ;Br if not
    107 001442  052761  000000G 000000G         BIS     #$SETRN,LSW9(R1);Remember SETUP is running
    108 001450                         2$:
```

```
 1                                          ;
 2                                          ;  Read in block zero of file being started.
 3                                          ;
 4 001450                                   RNREAD: .READW  #XAREA,#RUNCHN,#BLKO,#BLKWDS,#0
 5 001506  103045                                   BCC     SVINBG          ;BRANCH IF READ OK
 6 001510                                           FERR    #RDERM
 7 001524                                   CLSEXT: .CLOSE  #RUNCHN         ;CLOSE SAV FILE
 8 001532  105067  000000G                        CLRB    CINFLG          ;RESET CHAIN FLAG
 9 001536  105067  000000G                        CLRB    RUNARG          ;No argument string with RUN command
10 001542  116701  000000G                        MOVB    CORUSR,R1       ;GET CURRENT JOB INDEX NUMBER
11 001546  042761  000000G 000000G              BIC     #$TRNSP,LSW3(R1)
12 001554  042761  000000G 000000G              BIC     #$HITTY,LSW4(R1);CLEAN OUT MORE RUN FLAGS
13 001562  042761  000000C 000000G              BIC     #<$INDRN!$CHACT!$NOWTT>,LSW5(R1);CLEAN OUT FLAGS
14 001570  042761  000000G 000000G              BIC     #$DUPRN,LSW6(R1)
15 001576  042761  000000C 000000G              BIC     #$UCLRN!$NOINT,LSW7(R1)
16 001604  042761  000000C 000000G              BIC     #$DEBUG!$RNIOP!$RNMLK!$SETRN,LSW9(R1)
17 001612  004767  000000G                        CALL    RSTPRV          ;Reset job privilege flags
18 001616  000167  000000G                        JMP     RDCMD
19                                          ;
20                                          ;  Examine information in block 0 of sav file.
21                                          ;
22 001622  116701  000000G                 SVINBG: MOVB    CORUSR,R1       ;GET JOB INDEX NUMBER
23 001626  012702  000000G                        MOV     #BLKO,R2        ;POINT TO BLOCK 0 DATA WE JUST READ IN
24                                          ;  Set up new JSW for job being started.
25 001632  016203  000044                         MOV     44(R2),R3       ;GET JSW SPECIFIED IN SAV FILE IMAGE
26 001636  042703  100007                         BIC     #^C<077770>,R3  ;CLEAR SOME FLAGS
27 001642  016104  000000G                        MOV     LJSW(R1),R4     ;GET CURRENT JSW
28 001646  042704  075570                         BIC     #075570,R4      ;CLEAR SOME FLAGS IN IT
29 001652  050304                                 BIS     R3,R4           ;MERGE FLAGS
30 001654  010467  000000G                        MOV     R4,NEWJSW       ;THIS WILL BE JSW FOR JOB WE ARE STARTING
31 001660  026267  000000  000000G               CMP     0(R2),R50VIR    ;WAS JOB LINKED WITH /V SWITCH?
32 001666  001410                                 BEQ     23$             ;IF YES THEN SET VIRTUAL-MODE FLAG
33 001670  026227  000056  000034                CMP     56(R2),#28.     ;DOES JOB NEED MORE THAN 28KW?
34 001676  101004                                 BHI     23$             ;BR IF YES
35 001700  026227  000050  160000                CMP     50(R2),#160000  ;CHECK ADDRESS ABOVE TOP OF PROGRAM ROOT
36 001706  101403                                 BLOS    27$             ;BR IF DOES NOT NEED MORE THAN 28KW
37 001710  052767  000000G 000000G        23$:    BIS     #VIMAGE,NEWJSW  ;FORCE VIRTUAL-JOB MODE IF > 28KW NEEDED
38                                          ;  Set up information about the max amount of memory job will be allowed
39                                          ;  to use.  This is constrained by three things:
40                                          ;   1. The amount of memory specified by the MEMORY command.
41                                          ;   2. 56Kb if this is not a virtual job, (64Kb if it is a virtual job).
42 001716  016705  000000G                 27$:    MOV     MAXMEM,R5       ;Get amt of memory specified by MEMORY command
43 001722  032767  000000G 000000G              BIT     #VIMAGE,NEWJSW  ;Is this a virtual job?
44 001730  001005                                 BNE     17$             ;Br if this is a virtual job
45 001732  020527  160000                         CMP     R5,#160000      ;Constrain to 56Kb if not virtual job
46 001736  101402                                 BLOS    17$
47 001740  012705  160000                         MOV     #160000,R5
48 001744  010567  000000G                 17$:    MOV     R5,ODTBAS       ;This is base address of debugger
49                                          ;  If stack is allocated above top of program then say top of program
50                                          ;  is equal to top of stack.
51 001750  026262  000042  000050                CMP     42(R2),50(R2)   ;IS STACK ALLOCATED ABOVE PROGRAM TOP?
52 001756  101403                                 BLOS    16$             ;BR IF NOT
53 001760  016262  000042  000050                MOV     42(R2),50(R2)   ;SAY TOP OF PROGRAM = STACK TOP
54                                          ;  See if amount of memory to allocate for program was specified
55                                          ;  in SAV file.
56 001766  016703  000000G                 16$:    MOV     MAXMEM,R3       ;GET DEFAULT TOP OF MEMORY ADDRESS
57 001772  105767  000000G                        TSTB    VSWPFL          ;IS THIS A NON-SWAPPING SYSTEM?
```

```
     58 001776  001442                          BEQ     13$             ;BR IF YES
     59 002000  032767  000000G 000000G         BIT     #VIMAGE,NEWJSW  ;IS THIS A VIRTUAL IMAGE?
     60 002006  001005                          BNE     26$             ;BR IF YES
     61 002010  020327  160000                  CMP     R3,#160000      ;CONSTRAIN TO 56KB IF NOT VIRTUAL
     62 002014  101402                          BLOS    26$
     63 002016  012703  160000                  MOV     #160000,R3
     64 002022  016200  000056          26$:    MOV     56(R2),R0       ;WAS MEMORY ALLOCATION FOR PROGRAM SPECIFIED?
     65 002026  001406                          BEQ     7$              ;BR IF NOT
     66 002030  072027  000013                  ASH     #11.,R0         ;CONVERT # KW TO BYTE ADDRESS
     67 002034  001002                          BNE     21$             ;BR IF NOT 64KB
     68 002036  012700  177774                  MOV     #177774,R0      ;SET 64KB ADDRESS TOP
     69 002042  010003                  21$:    MOV     R0,R3           ;SET AS ADDRESS OF TOP OF PROGRAM SPACE
     70 002044  016200  000050          7$:     MOV     50(R2),R0       ;GET ADDRESS OF TOP OF STATIC PROGRAM AREA
     71 002050  062700  000002                  ADD     #2,R0
     72 002054  001002                          BNE     22$             ;BR IF DIDN'T OVERFLOW 64KB
     73 002056  012700  177774                  MOV     #177774,R0      ;SET SIZE FOR 64KB
     74 002062  020300                  22$:    CMP     R3,R0           ;IS ALLOCATED SIZE AT LEAST THIS BIG?
     75 002064  103001                          BHIS    8$              ;BR IF YES
     76 002066  010003                          MOV     R0,R3           ;USE ENOUGH SPACE FOR STATIC PROGRAM SIZE
     77 002070  016705  000000G         8$:     MOV     MXJADR,R5       ;GET MAX ADDRESS JOB CAN POSSIBLY USE
     78 002074  010367  000000G                 MOV     R3,ODTBAS       ;START ODT ABOVE TOP OF PROGRAM AREA
     79 002100  020305                          CMP     R3,R5           ;ROOM FOR PROGRAM TO RUN?
     80 002102  101123                          BHI     TOOBIG          ;BR IF NOT ENOUGH MEMORY
     81 002104  032767  000000G 000000G 13$:    BIT     #VIMAGE,NEWJSW  ;IS THIS A VIRTUAL JOB?
     82 002112  001005                          BNE     25$             ;BR IF YES
     83 002114  020327  160000                  CMP     R3,#160000      ;IS NOT, THEN CONSTRAIN SIZE TO 56KB
     84 002120  101402                          BLOS    25$
     85 002122  012703  160000                  MOV     #160000,R3
     86 002126  010367  000000G         25$:    MOV     R3,PRGSIZ       ;ADDRESS ABOVE TOP OF PROGRAM AREA
     87                             ;  Check top of program address.
     88 002132  016203  000050                  MOV     50(R2),R3       ;GET TOP-OF-PROGRAM ADDRESS FOR SAV FILE
     89 002136  020327  001000                  CMP     R3,#1000        ;MAKE SURE IT'S NOT TOO SMALL
     90 002142  103512                          BLO     ILLSAV          ;BR IF INVALID
     91 002144  010367  000000G                 MOV     R3,PRGTOP       ;SAVE TOP ADDRESS OF PROGRAM
     92 002150  016700  000000G                 MOV     ODTBAS,R0       ;GET TOP ADDRESS WE MAY GO UP TO
     93 002154  020300                          CMP     R3,R0           ;IS PROGRAM TOO BIG?
     94 002156  103075                          BHIS    TOOBIG          ;BR IF YES
     95                             ;  Check program's stack address.
     96 002160  016203  000042                  MOV     42(R2),R3       ;GET SPECIFIED STARTING STACK ADDRESS
     97 002164  001407                          BEQ     2$              ;BR IF NO INITIAL STACK ADDRESS SPECIFIED
     98 002166  020367  000000G                 CMP     R3,PRGTOP       ;CAN'T BE ABOVE TOP OF PROGRAM
     99 002172  101076                          BHI     ILLSAV
    100 002174  032703  000001                  BIT     #1,R3           ;CAN'T BE ODD
    101 002200  001073                          BNE     ILLSAV
    102 002202  000402                          BR      3$
    103 002204  012703  001000          2$:     MOV     #1000,R3        ;DEFAULT TO 1000 FOR STACK IF NONE SPECIFIED
    104 002210  010367  000000G         3$:     MOV     R3,USRSTK       ;SET AS STARTUP STACK ADDRESS
    105                            ;  Check program starting address.
    106 002214  016203  000040                  MOV     40(R2),R3       ;GET PROGRAM START ADDRESS
    107 002220  020327  000400                  CMP     R3,#400         ;MAKE SURE IT'S VALID
    108 002224  103406                          BLO     4$              ;BR IF TOO LOW
    109 002226  032703  000001                  BIT     #1,R3           ;MAKE SURE IT'S NOT ODD
    110 002232  001003                          BNE     4$
    111 002234  020367  000000G                 CMP     R3,PRGTOP       ;MAKE SURE IT IS IN MEMORY RANGE FOR PROGRAM
    112 002240  101410                          BLOS    5$
    113 002242                          4$:     FERR    #NOSTRT         ;INVALID START ADDRESS
    114 002256  000167  177242                  JMP     CLSEXT
```

```
    115 002262  010367  000000G          5$:     MOV     R3,USTART       ;SAVE PROGRAM STARTING ADDRESS
    116                                  ;
    117                                  ;  Set up name of program that is being started
    118                                  ;
    119 002266  012704  000000G                  MOV     #RUNDEV,R4      ;POINT TO PROGRAM NAME STORAGE CELLS
    120 002272  016461  000002  000000G          MOV     2(R4),LPRG1(R1) ;SAVE NAME OF RUNNING PROGRAM
    121 002300  016461  000004  000000G          MOV     4(R4),LPRG2(R1)
    122                                  ;
    123                                  ;  Set up status flags for program being started
    124                                  ;
    125 002306  016767  175466  000000G          MOV     RUNOPS,RUNFLG   ;Set status flags for running program
    126                                  ;
    127                                  ;  Broadcast status message to any monitoring jobs telling them that
    128                                  ;  we are starting execution of a job.
    129                                  ;
    130 002314  005761  000000G                  TST     LMONHD(R1)      ;Is our job being monitored?
    131 002320  001003                           BNE     29$             ;Br if it is
    132 002322  005767  000000G                  TST     SMONHD          ;Anyone monitoring all jobs?
    133 002326  001406                           BEQ     28$             ;Br if not
    134 002330  012700  000000G          29$:    MOV     #GENMON,R0      ;Point to EMT argument block
    135 002334  012760  000000G 000002          MOV     #JS$RUN,2(R0)   ;Set status code
    136 002342  104375                           EMT     375             ;Broadcast status message
    137                                  ;
    138                                  ;  Enter TSX to read in SAV file and transfer control to it.
    139                                  ;
    140 002344  012700  000000G          28$:    MOV     #RUNEMT,R0      ;EMT TO START PROGRAM EXECUTION
    141 002350  104375                           EMT     375             ;START SAV FILE
    142                                  ;
    143                                  ;  PROGRAM IS TOO BIG TO FIT IN MEMORY.
    144                                  ;
    145 002352                           TOOBIG: FERR    #OVRCOR
    146 002366  000406                           BR      CLEJMP
    147                                  ;
    148                                  ;  BAD SAV FILE FORMAT
    149                                  ;
    150 002370                           ILLSAV: FERR    #BADSAV
    151 002404  000167  177114           CLEJMP: JMP     CLSEXT
```

```
                                ;-----------------------------------------------
                                ;  PROCESS A .CHAIN REQUEST
                                ;
002410  012704  000000G  KDOCIN: MOV     #CINDAT,R4      ;POINT TO CELL WITH PROGRAM NAME
                                ;
                                ;  If chain is being done to SY:LD.SYS, simply do a SET LD CLEAN
                                ;
002414  010405                  MOV     R4,R5           ;POINT TO PROGRAM NAME
002416  012703  000000G          MOV     #LDNAM,R3       ;POINT TO "SY:LD.SYS"
002422  012700  000004          MOV     #4,R0           ;COMPARE 4 WORDS
002426  022325          3$:     CMP     (R3)+,(R5)+     ;COMPARE PROGRAM NAMES
002430  001004                  BNE     2$              ;BR IF NOT CHAINING TO LD HANDLER
002432  077003                  SOB     R0,3$
002434  004767  000000G          CALL    LDCLEN          ;DO "SET LD CLEAN" OPERATION
002440  000421                  BR      4$              ;DON'T DO THE CHAIN
                                ;
                                ;  Look up SAV file we are chaining to
                                ;
002442          2$:     .LOOKUP #XAREA,#RUNCHN,R4
002460  103403                  BCS     1$              ;BRANCH IF ERROR
002462  005000                  CLR     R0              ;Say no RUN options
002464  000167  176120          JMP     PLOAD           ;LOAD AND RUN PROGRAM
002470          1$:     FERR    #NOPRG
002504  105067  000000G  4$:     CLRB    CINFLG
002510  000167  000000G          JMP     NOCIN
                                ;
```

```
      1                                                .SBTTL  CHKIND -- Force IND to run from boot device
      2                                       ;-------------------------------------------------------------
      3                                       ;
      4                                       ;  Inputs: FILNAM has dev:filspc.ext of program about to be run
      5                                       ;
      6                                       ;  Outputs: If program is IND, dev: in FILNAM changed to boot device
      7                                       ;                              and .ext in FILNAM changed to .SAV
      8                                       ;
      9 002514  026727  000002G 035164  CHKIND: CMP     FILNAM+2,#^RIND ;IS PROGRAM IND?
     10 002522  001021                         BNE     9$              ;RETURN IF NOT
     11 002524  005767  000004G                TST     FILNAM+4        ;AND ONLY 3 CHARS?
     12 002530  001016                         BNE     9$              ;RETURN IF NOT
     13                                       ;  Program about to be run is IND
     14 002532  005767  000000G                TST     INDSAV          ;WAS IND AVAILABLE DURING START-UP?
     15 002536  001005                         BNE     1$              ;BR IF SO
     16 002540  005726                         TST     (SP)+           ;THROW AWAY RETURN POINTER IF NOT
     17 002542                                 FABORT  #NOIND          ;AND ABORT COMMAND
     18                                       ;  Ensure that IND is always run from "real" SY
     19 002552  016767  000000G 000000G 1$:    MOV     SYNAME,FILNAM   ;ALWAYS RUN FROM BOOT DEVICE
     20 002560  012767  073376  000006G        MOV     #^RSAV,FILNAM+6 ;USE IND.SAV
     21 002566  000207                  9$:    RETURN
     22                                       ;
     23         000001                         .END
Errors detected:  0

*** Assembler statistics

    Work  file  reads: 0
    Work  file writes: 0
    Size of work file: 9840 Words  ( 39 Pages)
    Size of core pool: 18176 Words  ( 71 Pages)
    Operating  system: RT-11

    Elapsed time: 00:00:21.06
    ,LP:TSKM2C=DK:TSKM2C/C/N:SYM
```

Cross reference table (CREF V05.05)

```
$CFOPN    1-30      5-48
$CHACT    1-48      7-38      8-13
$DEBUG    1-25      7-77      8-16
$DUPRN    1-39      7-89      8-14
$HITTY    1-25      7-44      8-12
$INDRN    1-25      5-72      7-98      8-13
$NOINT    1-25      7-50      8-15
$NOWIN    1-39      7-83
$NOWTT    1-26      7-41      8-13
$PRGLK    1-26      7-68
$QUIET    1-31      5-102
$RNIOP    1-26      7-58      8-16
$RNMLK    1-26      7-65      8-16
$SCCA     1-47      7-80
$SETRN    1-41      7-107     8-16
$STSNG    1-33      7-5
$TRNSP    1-26      7-47      8-11
$UCLRN    1-26      7-104     8-15
. . . V1  5-8       5-8       5-8       8-4       8-4       8-7       8-7       9-19      9-19      9-19
. . . V2  5-8       5-8       5-8#      5-8#      8-4       8-4       8-4#      8-4#      8-7       8-7#      9-19      9-19
          9-19#     9-19#
ABRTCF    1-38      7-70
ACRFN     1-30      5-4
AF$BYA    1-27      4-54
AF$CCA    1-27      4-82      7-78
AF$DBG    1-27      4-77      7-71
AF$DUP    1-27      7-87
AF$HIE    1-27      4-59      7-42
AF$IND    1-28      7-90
AF$IOP    1-28      4-64      7-52
AF$MEM    1-28      4-39      7-59
AF$NOI    1-28      4-69      7-48
AF$NOW    1-28      4-44      7-39
AF$NPW    1-32      4-87      7-81
AF$PLK    1-29      4-34      5-10      7-66
AF$SCA    1-29      4-44      7-7       7-36
AF$SET    1-29      7-105
AF$TPO    1-29      4-49      7-45
AF$UCL    1-29      7-99
AFCF      1-48      7-31
BADSAV    1-45      8-150
BLKO      1-47      8-4       8-23
BLKWDS    1-47      8-4
CCSPRV    1-38      7-69
CFBUF     1-30      5-50      5-58
CFSQEZ    1-30      5-52
CHKIND    5-7       10-9#
CINDAT    1-33      5-21*     5-22*     5-34      5-35      5-39      5-40*     9-4
CINFLG    1-41      8-8*      9-24*
CLEJMP    8-146     8-151#
CLSEXT    5-6       5-14      7-57      7-64      7-76      7-97      8-7#      8-114     8-151
CMDOFF    1-24      5-12
CMDRSY    1-20      4-12#
CMDRUN    1-20      4-7#
CORUSR    1-32      4-29      5-16      8-10      8-22
CVTTAB    1-41      4-16
```

Cross reference table (CREF V05.05)

```
DKSAV    1-32    4-7
DORUN    1-20    5-15#
DORUNX   5-9     5-16#
EM$MPV   1-37    7-56
EM$NAD   1-38    7-75
EM$NAL   1-37    7-63
EM$NPD   1-32    4-76
EM$NUC   1-40    7-103
FILNAM   1-47    5-8     5-106   10-9    10-11   10-19*  10-20*
FIXPRV   1-48    7-26
FKILL    1-46    4-28    4-76    5-98    7-103   10-17
FORCEO   1-33    7-13
FPRINT   1-24    5-13    7-56    7-63    7-75    7-96    8-6     8-113   8-145   8-150   9-23
GENMON   1-45    8-134
II$FLG   1-31    7-27
II$NPV   1-31    7-23
II$PRV   1-31    7-22
IIBUF    1-35    7-22    7-23    7-27
ILLSAV   8-90    8-99    8-101   8-150#
IN$ACT   1-39    7-92
IN$CNT   1-40    7-94
INDACT   1-40    7-96
INDSAV   1-24   10-14
INDSTA   1-39    7-92    7-94
INSSRC   1-34    7-16
INVOPT   1-46    4-28
JS$RUN   1-35    8-135
KDOCIN   1-20    9-4#
L        3-6     3-6#    3-7     3-7#    3-8     3-8#    3-9     3-9#    3-10    3-10#   3-11    3-11#
         3-12    3-12#   3-13    3-13#   3-14    3-14#   3-15    3-15#   3-16    3-16#   3-17    3-17#
LDCLEN   1-45    9-14
LDNAM    1-45    9-9
LITIME   1-36    7-51*
LJSW     1-42    8-27
LMONHD   1-44    8-130
LOGASN   1-33    6-19
LPRG1    1-44    8-120*
LPRG2    1-44    8-121*
LSW11    1-39    7-83*
LSW2     1-36    5-17
LSW2S    1-36    5-17*
LSW3     1-48    7-47*   8-11*
LSW4     1-28    5-48    5-102*  7-44*   8-12*
LSW5     1-47    5-72    7-38*   7-41*   7-68*   7-80*   7-98*   8-13*
LSW6     1-33    7-5     7-89*   8-14*
LSW7     1-36    7-50*   7-104*  8-15*
LSW9     1-41    7-58*   7-65*   7-77*   7-107*  8-16*
MAXMEM   1-42    8-42    8-56
MXJADR   1-48    8-77
NARGS    3-5#    3-6     3-6     3-7     3-7     3-8     3-8     3-9     3-9     3-10    3-10    3-11
         3-11    3-12    3-12    3-13    3-13    3-14    3-14    3-15    3-15    3-16    3-16    3-17
         3-17
NEWJSW   1-42    8-30*   8-37*   8-43    8-59    8-81
NOCIN    1-46    9-25
NOIND    1-24   10-17
NOPRG    1-24    5-13    9-23
```

Cross reference table (CREF V05.05)

```
NOSTRT   1-44    8-113
ODTBAS   1-43    8-48*   8-78*   8-92
OVRCOR   1-45    8-145
PO$DBG   1-25    1-38    7-73
PO$LOK   1-37    7-61
PO$MEM   1-36    7-54
PLOAD    1-20    5-108   6-10#   9-22
PRGSIZ   1-43    8-86*
PRGTOP   1-43    8-91*   8-98    8-111
PRIVCO   1-37    7-22*   7-23*   7-54    7-61    7-73
PUSHCF   1-30    5-57
PVNPW    1-34    7-21
R50VIR   1-42    8-31
RDCMD    1-48    8-18
RDERM    1-41    8-6
RNBYA    3-6     4-54#
RNDB     3-7     3-14    4-74#
RNNPW    3-13    4-87#
RNOARG   5-25    5-106#
RNREAD   8-4#
RNSCCA   3-16    4-82#
RNSCHR   3-15    4-44#
RNSHI    3-8     4-59#
RNSIOP   3-9     4-64#
RNSLK    3-10    4-34#
RNSMLK   3-11    4-39#
RNSNOI   3-12    4-69#
RNSTPO   3-17    4-49#
RSTPRV   1-42    8-17
RUNARG   1-34    5-23*   8-9*
RUNASW   7-5#
RUNCHN   1-34    5-8     5-8     8-4     8-4     8-7     8-7     9-19    9-19
RUNDEV   1-34    6-14    6-18    6-20    7-12    8-119
RUNEMT   1-35    8-140
RUNFLG   1-34    8-125*
RUNHD    1-35    3-5#    4-25
RUNNAM   1-20    4-23    5-4#
RUNOPS   1-50#   4-17*   4-34*   4-39*   4-44*   4-49*   4-54*   4-59*   4-64*   4-69*   4-77*   4-82*
         4-87*   5-10    5-15*   5-107   6-10*   7-7*    7-27*   7-31*   7-35    8-125
RUNPRS   4-8     4-16#
RUNSW    4-21#   4-35    4-40    4-45    4-50    4-55    4-60    4-65    4-70    4-78    4-83    4-88
SEARCH   1-46    4-26
SKPSPC   1-46    4-21
SMONHD   1-45    8-132
SVINBG   8-5     8-22#
SYNAME   1-24   10-19
SYSAV    1-32    4-12
TOOBIG   8-80    8-94    8-145#
TOOLNG   1-30    5-98
TRMSTR   1-27    1-29    5-86
TSKM2C   1-5#
UCLBLK   1-40    7-101
USRSTK   1-44    8-104*
USTART   1-44    8-115*
VDBFLG   1-33    4-74
VIMAGE   1-42    8-37    8-43    8-59    8-81
```

Cross reference table (CREF V05.05)

```
VSWPFL    1-43       8-57
XAREA     1-47       5-8        8-4        9-19
```

Cross reference table (CREF V05.05)

```
...CM1    5-8       8-4       9-19
...CM2    5-8       5-8       8-4       8-4       8-4       8-4       9-19      9-19
...CM3    8-7
...CM5    5-8       8-4       9-19
...CM7    8-4
.CLOSE    1-17#     8-7
.LOOKU    1-17#     5-8       9-19
.READW    1-17#     8-4
CMDDEF    2-47#     3-6       3-7       3-8       3-9       3-10      3-11      3-12      3-13      3-14      3-15      3-16
          3-17
FABORT    2-15#     4-28      4-76      5-98      7-103     10-17
FERR      2-4#      5-13      7-56      7-63      7-75      7-96      8-6       8-113     8-145     8-150     9-23
FWARN     2-23#
TBLDEF    2-35#     3-5
TBLEND    2-61#     3-18
```



```
TTTTT  SSS  K   K M   M  222  DDD
  T   S   S K  K  MM MM 2   2 D  D
  T   S     K K   M M M    2  D   D
  T    SSS  KK    M   M   2   D   D
  T       S K K   M   M  2    D   D
  T   S   S K  K  M   M 2     D  D
  T    SSS  K   K M   M 22222 DDD

TTTTTTTTTT    SSSSSS    KK      KK  MM      MM    222222    DDDDDD
TTTTTTTTTT    SSSSSS    KK      KK  MM      MM    222222    DDDDDD
    TT       SS       SS  KK    KK    MMMM  MMMM  22      22  DD    DD
    TT       SS       SS  KK    KK    MMMM  MMMM  22      22  DD    DD
    TT       SS           KK  KK      MM  MM  MM        22    DD      DD
    TT       SS           KK  KK      MM  MM  MM        22    DD      DD
    TT         SSSSSS     KKKK        MM      MM      22      DD      DD
    TT         SSSSSS     KKKK        MM      MM      22      DD      DD
    TT               SS   KK  KK      MM      MM    22        DD      DD
    TT               SS   KK  KK      MM      MM    22        DD      DD
    TT       SS      SS   KK    KK    MM      MM  22          DD    DD
    TT       SS      SS   KK    KK    MM      MM  22          DD    DD
    TT         SSSSSS     KK      KK  MM      MM  2222222222  DDDDDD
    TT         SSSSSS     KK      KK  MM      MM  2222222222  DDDDDD

[[[    000  55555        000   333    ]]]
[     0   0 5          0   0 3   3     ]
[     0  00 5          0  00     3     ]
[     0 0 0 5555       0 0 0    33     ]
[     00  0     5  ,,  00  0     3     ]
[     0   0 5   5  ,,  0   0 3   3     ]
[[[    000   555  ,,    000   333    ]]]
```