Table of contents

```
                                        .TITLE  TSEM5   TSX-Plus EMT Overlay
                                        .ENABL  LC
                                        .ENABL  AMA
                                        .DSABL  GBL
                                ;
                                ;  Copyright (C) 1986.
                                ;
                                ;  S&H Computer Systems, Inc.
                                ;  Nashville, Tennessee
                                ;
                                ;  This software is furnished under a license for use only
                                ;  on a single computer system and may be copied only with
                                ;  the inclusion of the above copyright notice.  This
                                ;  software, or any other copies thereof, may not be provided
                                ;  or otherwise made available to any other person except
                                ;  for use on such system and to one who agrees to these
                                ;  license terms.  Title to and ownership of the software
                                ;  shall at all times remain with S&H Computer Systems, Inc.
                                ;
000000                                  .CSECT  TSEM5
000000  020553                  TSEM2:  .RAD50  /EM5/            ;Overlay id
        177776                  PS      =       177776          ;Processor Status Word
                                ;
                                ;  Macro calls
                                ;
                                        .MCALL  .FPROT,.LOOKUP,.PURGE
                                ;
                                ;  Global definitions
                                ;
                                        .GLOBL  LDMEMT,LDDEMT,LDIEMT,LDAEMT
                                ;
                                ;  Global references
                                ;
                                        .GLOBL  CXTRMN,R$CHN,R$XCHN,CHNSIZ,NLCHN
                                        .GLOBL  LDSIZE,C.DEVQ,C.SBLK,CS$RON,LDBASE,LDDEVX
                                        .GLOBL  OVRHC,CHNADR,CS$OPN,C.CSW,SETERR,EMTBLK,MAXLD
                                        .GLOBL  VLDSYS,VALADW,CSIFIL,CSIARE,LDNAME,LDFLAG
                                        .GLOBL  LD$RON,SERFLG,EMTXIT,LDPDEV,MAXASN,ASNTBL,AT$LOG
                                        .GLOBL  AT$$SZ,AT$DEV,AT$FIL,AT$EXT,AT$SIZ,SYNAME
                                        .GLOBL  RESDEV,OKFILE,OKFAND,OKFNND,OF$FIL,OF$$SZ,OF$DEV
                                        .GLOBL  OF$UNT,OF$FLG,OT$RON,WLDNAM
                                ;
                                ;  Data areas
                                ;
000002  045676                  R50LD0: .RAD50  /LD0/
000004  045705                  R50LD7: .RAD50  /LD7/
000006  045640                  R50LD:  .RAD50  /LD /
000010  015270                  R50DK:  .RAD50  /DK /
000012  075250                  R50SY:  .RAD50  /SY /
                                ;
                                ;  EMT arg block to enable caching for a device
                                ;
000014     000      134         MNTARG: .BYTE   0,134
000016  000000G                         .WORD   CSIARE
                                ;
                                ;  EMT arg block to dismount a device
                                ;
```

```
    58 000020      000      135          DMTARG: .BYTE   0,135
    59 000022  000000G                          .WORD   CSIARE
```

```
                        ;+-----------------------------------------------------------------
                        ;  Macros to enable and disable interrupts.
                        ;
                              .MACRO  DISABL          ;DISABLE INTERRUPTS
                              BIS     #340,@#PS
                              .ENDM   DISABL
                        ;
                              .MACRO  ENABL           ;ENABLE INTERRUPTS
                              BIC     INTPRI,@#PS
                              .ENDM   ENABL
                        ;+-----------------------------------------------------------------
                        ;  Macro to print an error message when a system crash occurs.
                        ;
                        ;  Arguments:
                        ;   MSG = Name of error message to print.
                        ;   ARG = (Optional) argument value to display with error message.
                        ;
                              .GLOBL  DIEMSG,DIEARG,SYSHLT
                              .MACRO  DIE     MSG,ARG
                              MOV     MSG,@#DIEMSG
                              .IF     NB,ARG
                              MOV     ARG,@#DIEARG
                              .ENDC
                              CALL    @#SYSHLT
                              .ENDM   DIE

                        ;+-----------------------------------------------------------------------------
                        ;  Macro definition for calling global routines residing in mapped
                        ;  system regions.
                        ;
                              .MACRO  OCALL   ENTADD
                              .IF     B,ENTADD
                                      .ERROR  ;OCALL SPECIFIED WITH NO ENTRY ADDRESS
                                      .MEXIT
                              .ENDC
                              CALL    OVRHC           ;CALL THE OVERLAY HANDLER
                              .WORD   ENTADD          ;SPECIFY THE ENTRY POINT
                              .ENDM
```

```
                                        .SBTTL  LDMEMT -- EMT to mount a logical disk
                                ;-------------------------------------------------------------------
                                ;  EMT to mount a logical disk:
                                ;  Argument block format:
                                ;
                                ;       .BYTE   chan,163
                                ;       .BYTE   ld_number,flags
                                ;       .WORD   file_spec_pointer
                                ;
000024                          LDMEMT:
                                ;
                                ;  Make sure the specified channel is closed
                                ;
000024  013703  0000000         1$:     MOV     CHNADR,R3       ;Get address of channel block
000030  032763  0000000 0000000         BIT     #CS$OPN,C.CSW(R3);Is the channel open or closed?
000036  001403                          BEQ     1$              ;Br if closed
000040  005000                          CLR     R0              ;Error 0 -- Channel is open
000042  000137  0000000                 JMP     SETERR
                                ;
                                ;  Make sure the logical disk number is valid
                                ;
000046  113702  0000020         1$:     MOVB    EMTBLK+2,R2     ;Get specified LD number
000052  020227  0000000                 CMP     R2,#MAXLD       ;Must be in range 0 - MAXLD
000056  103404                          BLO     2$              ;Br if ok
000060  012700  000001                  MOV     #1,R0           ;Error 1 -- Invalid LD number
000064  000137  0000000                 JMP     SETERR
                                ;
                                ;  Make sure LD support included in system
                                ;
000070  105737  0000000         2$:     TSTB    VLDSYS          ;Is LD support included in system?
000074  001004                          BNE     3$              ;Br if yes
000076  012700  000002                  MOV     #2,R0           ;Error 2 -- LD support not included in system
000102  000137  0000000                 JMP     SETERR
                                ;
                                ;  Error if this LD is already mounted
                                ;
000106  006302                  3$:     ASL     R2              ;Get LD index
000110  010200                          MOV     R2,R0           ;Get LD index
000112  072027  000002                  ASH     #2,R0           ;Convert to index into LDNAME table
000116  005760  0000000                 TST     LDNAME(R0)      ;Is this LD unit already mounted?
000122  001404                          BEQ     4$              ;Br if not
000124  012700  000003                  MOV     #3.,R0          ;Error 3 -- LD unit already mounted
000130  000137  0000000                 JMP     SETERR
                                ;
                                ;  Move file spec to internal buffer
                                ;
000134  013700  0000040         4$:     MOV     EMTBLK+4,R0     ;Get address of user's file spec
000140  001423                          BEQ     6$              ;Br if no file spec specified
000142  004737  0000000                 CALL    VALADW          ;Make sure it is valid
000146  012704  0000000                 MOV     #CSIFIL,R4      ;Point to area where we will store file spec
000152  012703  000004                  MOV     #4,R3           ;Get # words to move
000156  106520                  5$:     MFPD    (R0)+           ;Get word from user's area
000160  012624                          MOV     (SP)+,(R4)+     ;Move to internal area
000162  077303                          SOB     R3,5$           ;Loop till all of file spec moved
                                ;
                                ;  Translate logical file device name to physical device
                                ;  and make sure we have a non-null file name.
```

```
                        ;
000164  012705  000000G          MOV     #CSIFIL,R5      ;Point to file spec to be translated
000170  004737  001466'          CALL    LOGASN          ;Do the assign
000174  005737  000000G          TST     CSIFIL          ;Is device name null?
000200  001403                   BEQ     6$              ;Br if yes
000202  005737  000002G          TST     CSIFIL+2        ;Is file name null?
000206  001004                   BNE     7$              ;Br if not
000210  012700  000004     6$:   MOV     #4.,R0          ;Error 4 -- Invalid file spec
000214  000137  000000G          JMP     SETERR
                        ;
                        ;  Make sure file is not on same LD as is being mounted
                        ;
000220  013700  000000G     7$:  MOV     CSIFIL,R0       ;Get name of device file is on
000224  004737  001640'          CALL    LDDEVN          ;See if file is on a LD device
000230  103406                   BCS     8$              ;Br if not
000232  020002                   CMP     R0,R2           ;Is it on a lower # LD?
000234  103404                   BLO     8$              ;Br if yes
000236  012700  000005           MOV     #5.,R0          ;Error 5 -- File is on an invalid LD
000242  000137  000000G          JMP     SETERR
                        ;
                        ;  Move file name into logical disk name table
                        ;
000246  010205              8$:  MOV     R2,R5           ;Get LD index number
000250  072527  000002           ASH     #2,R5           ;Cvt to index into name table
000254  062705  000000G          ADD     #LDNAME,R5      ;Point to name entry for this LD
000260  012704  000000G          MOV     #CSIFIL,R4      ;Point to cell with file spec
000264  012700  000004           MOV     #4,R0           ;Get # words to move
000270  012425              9$:  MOV     (R4)+,(R5)+     ;Move file spec into LD name table
000272  077002                   SOB     R0,9$           ;Loop till all of spec moved
                        ;
                        ;  Set up flags for logical disk
                        ;
000274  005062  000000G          CLR     LDFLAG(R2)      ;Clear all flags
000300  105737  000003G          TSTB    EMTBLK+3        ;Was NOWRITE flag specified?
000304  001403                   BEQ     10$             ;Br if not
000306  052762  000000G 000000G  BIS     #LD$RON,LDFLAG(R2);Set NOWRITE flag to the LD
                        ;
                        ;  Lookup file and set up information about position of logical
                        ;  disk on physical disk.
                        ;
000314  113703  000000G    10$: MOVB    EMTBLK,R3       ;Get channel # for lookup
000320  004737  001174'          CALL    LDMNT           ;Set up info about file for logical disk
000324  103004                   BCC     11$             ;Br if ok
000326  012700  000006           MOV     #6.,R0          ;Error 6 -- Cannot lookup file
000332  000137  000000G          JMP     SETERR          ;Error on lookup
                        ;
                        ;  Protect the file
                        ;
000336  113746  000000G    11$: MOVB    SERFLG,-(SP)    ;Save hard-error flag
000342  112737  000001  000000G  MOVB    #1,SERFLG       ;Don't abort on hard errors
000350  113703  000000G          MOVB    EMTBLK,R3       ;Get channel number
000354                           .FPROT  #CSIARE,R3,#CSIFIL,#1 ;Protect the file
000404  112637  000000G          MOVB    (SP)+,SERFLG    ;Restore error control
                        ;
                        ;  Initiate directory caching
                        ;
000410  113700  000002G          MOVB    EMTBLK+2,R0     ;Get LD #
```

```
000414  063700  000002'          ADD     R50LDO,R0       ;Add "LDO" to form rad50 device name
000420  010037  000000G          MOV     R0,CSIARE       ;Store in cell pointed to by arg block
000424  012700  000014'          MOV     #MNTARG,R0      ;Point to EMT argument block
000430  104375                   EMT     375             ;Enable caching for the device
                                ;
                                ;  Now do a SET LD CLEAN operation to make sure all logical disks
                                ;  are set up correctly.
                                ;
000432  113703  000000G          MOVB    EMTBLK,R3       ;Get channel number for lookups
000436  004737  001050'          CALL    LDCLEN          ;Do SET LD CLEAN
                                ;
                                ;  Finished mounting the EMT
                                ;
000442  000137  000000G          JMP     EMTXIT
```

```
                                        .SBTTL  LDDEMT -- EMT to dismount a logical disk
                                ;-----------------------------------------------------------------------
                                ;  The form of the EMT used to dismount a logical disk is:
                                ;
                                ;       .BYTE   3,135
                                ;       .BYTE   ld_unit,0
                                ;
000446                          LDDEMT:
                                ;
                                ;  Make sure the LD unit number is valid
                                ;
000446  113702  0000020                 MOVB    EMTBLK+2,R2     ;Get LD unit #
000452  020227  0000000                 CMP     R2,#MAXLD       ;See if valid
000456  103404                         BLO     1$              ;Br if ok
000460  012700  000001                  MOV     #1,R0           ;Error 1 -- Invalid LD #
000464  000137  0000000                 JMP     SETERR
                                ;
                                ;  See if the LD is active
                                ;
000470  006302                  1$:     ASL     R2              ;Convert to unit index
000472  010200                          MOV     R2,R0           ;Get unit index
000474  072027  000002                  ASH     #2,R0           ;Convert to index into LDNAME table
000500  005760  0000000                 TST     LDNAME(R0)      ;Is this LD mounted?
000504  001003                         BNE     2$              ;Br if yes
000506  005000                         CLR     R0              ;Error 0 -- Specified LD is not mounted
000510  000137  0000000                 JMP     SETERR
                                ;
                                ;  Try to do the dismount
                                ;
000514  004737  000760'         2$:     CALL    LDDMT           ;Try to dismount the LD
000520  103004                         BCC     3$              ;Br if ok
000522  012700  000003                  MOV     #3.,R0          ;Error 3 -- Open file on LD being dismounted
000526  000137  0000000                 JMP     SETERR
                                ;
                                ;  Finished
                                ;
000532  000137  0000000         3$:     JMP     EMTXIT          ;Finished with dismount EMT
```

```
                        .SBTTL  LDIEMT -- EMT to get information about a logical disk
                ;-----------------------------------------------------------------------
                ;  The LDIEMT is used to obtain information about a logical disk structure.
                ;
                ;  The form of the argument block is:
                ;
                ;       .BYTE   4,135
                ;       .BYTE   ld_number,0
                ;       .WORD   buffer_pointer
                ;
000536          LDIEMT:
                ;
                ;  Make sure the LD unit number is valid
                ;
000536  113702  000002G         MOVB    EMTBLK+2,R2     ;Get LD unit #
000542  020227  000000G         CMP     R2,#MAXLD       ;See if valid
000546  103404                  BLO     7$              ;Br if ok
000550  012700  000001          MOV     #1,R0           ;Error 1 -- Invalid LD #
000554  000137  000000G         JMP     SETERR
                ;
                ;  Make sure the buffer address is valid
                ;
000560  013700  000004G 7$:     MOV     EMTBLK+4,R0     ;Get address of buffer
000564  004737  000000G         CALL    VALADW          ;Make sure it is valid
                ;
                ;  Transfer the LD file name to the buffer
                ;
000570  006302                  ASL     R2              ;Get LD unit index
000572  010203                  MOV     R2,R3           ;Get LD unit index
000574  072327  000002          ASH     #2,R3           ;Cvt to index into LDNAME table
000600  062703  000000G         ADD     #LDNAME,R3      ;Point to LDNAME entry
000604  005713                  TST     (R3)            ;Is there a specified file name?
000606  001423                  BEQ     2$              ;Br if not
000610  012704  000004          MOV     #4,R4           ;Move 4 words
000614  012346          1$:     MOV     (R3)+,-(SP)     ;Move a word of the file name
000616  106620                  MTPD    (R0)+           ;to user's buffer
000620  077403                  SOB     R4,1$
                ;
                ;  Pass flag word
                ;
000622  005046                  CLR     -(SP)           ;Start with all flags off
000624  032762  000000G 000000G BIT     #LD$RON,LDFLAG(R2);Is read-only flag set?
000632  001402                  BEQ     5$              ;Br if not
000634  052716  000001          BIS     #1,(SP)         ;Set read-only flag for return
000640  005762  000000G 5$:     TST     LDPDEV(R2)      ;Is LD currently accessible?
000644  001002                  BNE     6$              ;Br if yes
000646  052716  000002          BIS     #2,(SP)         ;Set not-accessible flag
000652  106620          6$:     MTPD    (R0)+           ;Move flag word to user's buffer
000654  000405                  BR      9$
                ;
                ;  This LD is not active.
                ;  Return all zeroes to user's buffer
                ;
000656  012704  000005  2$:     MOV     #5,R4           ;We will move 5 words of zeroes
000662  005046  4$:     CLR     -(SP)           ;Move 5 words of zeroes
000664  106620                  MTPD    (R0)+
000666  077403                  SOB     R4,4$
```

```
                                        ;
                                        ;  Finished
                                        ;
000670  000137  000000G         9$:     JMP     EMTXIT          ;Finished with EMT
```

```
                                        .SBTTL  LDAEMT -- Dismount all logical disks
                                ;-------------------------------------------------------------------
                                ;
                                ;  The form of the argument block is:
                                ;    .BYTE   5,135
                                ;    .WORD   0
                                ;
000674                          LDAEMT:
                                ;
                                ;  Set up for loop to dismount all logical disks
                                ;
000674  005001                          CLR     R1              ;Set up error flag
000676  005002                          CLR     R2              ;First LD to be dismounted
                                ;
                                ;  Loop to dismount all logical disks
                                ;
000700                          1$:
                                ;
                                ;  See if the LD is active
                                ;
000700  010200                          MOV     R2,R0           ;Get unit index
000702  072027  000002                  ASH     #2,R0           ;Convert to index into LDNAME table
000706  005760  000000G                 TST     LDNAME(R0)      ;Is this LD mounted?
000712  001405                          BEQ     2$              ;Br if no
                                ;
                                ;  Dismount logical disks
                                ;
000714  004737  000760'                 CALL    LDDMT           ;Attempt to dismount LD
000720  103002                          BCC     2$              ;Br if OK
000722  012701  000001                  MOV     #1,R1           ;Set error flag
000726                          2$:
000726  062702  000002                  ADD     #2,R2           ;Increment logical disk
000732  020227  000020                  CMP     R2,#16.         ;All done?
000736  002760                          BLT     1$              ;Loop if not
000740  005701                          TST     R1              ;Was an error encountered?
000742  001404                          BEQ     3$              ;Br if not
000744  012700  000003                  MOV     #3,R0           ;Error #3 ==> LD has open files
000750  000137  000000G                 JMP     SETERR
                                ;
                                ;  Finished
                                ;
000754  000137  000000G         3$:     JMP     EMTXIT
```

```
                        .SBTTL  LDDMT  -- Dismount a logical disk
                ;-----------------------------------------------------------------------
                ;  Dismount a logical disk
                ;
                ;  Inputs:
                ;   R2 = LD device index
                ;
                ;  Outputs:
                ;   C-flag cleared ==> No error on dismount
                ;   C-flag set ==> Error on dismount.  Open file on LD being dismounted.
                ;
                ;
000760          LDDMT:
                ;
                ;  Error if this job has any open files on LD being dismounted
                ;
000760 004737 002206'           CALL    LDOFCK          ;Any open files on this LD?
000764 103430                   BCS     9$              ;Br if open file on LD being dismounted
                ;
                ;  Stop doing directory caching for the device
                ;
000766 010200                   MOV     R2,R0           ;Get the LD unit index
000770 006200                   ASR     R0              ;Convert to unit number
000772 063700 000002'           ADD     R50LD0,R0       ;Add "LD0" to form device name
000776 010037 000000G           MOV     R0,CSIARE       ;Store in work cell
001002 113746 000000G           MOVB    SERFLG,-(SP)    ;Save error control flag
001006 112737 000001 000000G    MOVB    #1,SERFLG       ;Suppress aborts
001014 012700 000020'           MOV     #DMTARG,R0      ;Stop caching on device
001020 104375                   EMT     375
001022 112637 000000G           MOVB    (SP)+,SERFLG    ;Reset error control
                ;
                ;  Clear logical disk data tables
                ;
001026 005062 000000G           CLR     LDPDEV(R2)      ;No physical device assignment
001032 010200                   MOV     R2,R0           ;Get unit index
001034 072027 000002            ASH     #2,R0           ;Convert to index into LDNAME table
001040 005060 000000G           CLR     LDNAME(R0)      ;No file associated with LD
                ;
                ;  Deassign any logical names assigned to this LD
                ;
                ;       MOV     CSIARE,R0       ;Get LD device name
                ;       CALL    DEADEV          ;Deassign any logical names
                ;
                ;  Successfully finished
                ;
001044 000241                   CLC                     ;Signal success on return
                ;
                ;  Finished
                ;
001046 000207   9$:             RETURN
```

```
                                        .SBTTL  LDCLEN -- Perform SET LD CLEAN operation
                                ;-------------------------------------------------------------
                                ;  LDCLEN is called to perform the SET LD CLEAN function.
                                ;  It also causes the file directory cache to be cleaned out.
                                ;
                                ;  Inputs:
                                ;   R3 = Number of free channel to use for lookup
                                ;   CHNADR = Pointer to channel block of free channel
                                ;
001050  010246                  LDCLEN: MOV     R2,-(SP)
001052  010546                          MOV     R5,-(SP)
                                ;
                                ;  Perform LD CLEAN operation (reset logical disk information)
                                ;
001054  013705  000002'                 MOV     R50LDO,R5       ;Get name of first logical disk (LDO)
001060  005002                          CLR     R2              ;Get index to 1st logical disk
                                ;
                                ;  See if this logical disk is in use
                                ;
001062  010200                  1$:     MOV     R2,R0           ;Get LD index number
001064  072027  000002                  ASH     #2,R0           ;Convert to index into LDNAME table
001070  005760  000000G                 TST     LDNAME(R0)      ;Is this logical disk in use?
001074  001426                          BEQ     2$              ;Br if not
                                ;
                                ;  Dismount the logical disk
                                ;
001076  010537  000000G                 MOV     R5,CSIARE       ;Set device name to be dismounted
001102  113746  000000G                 MOVB    SERFLG,-(SP)    ;Save error control
001106  112737  000001  000000G         MOVB    #1,SERFLG       ;Do .SERR
001114  012700  000020'                 MOV     #DMTARG,R0      ;Point to EMT argument block
001120  104375                          EMT     375             ;Dismount the device
001122  112637  000000G                 MOVB    (SP)+,SERFLG    ;Restore error control
                                ;
                                ;  Now reinitialize information about the logical disk
                                ;
001126  004737  001174'                 CALL    LDMNT           ;Reset information about the LD
                                ;
                                ;  Now remount the logical disk
                                ;
001132  005762  000000G                 TST     LDPDEV(R2)      ;Is the logical disk there?
001136  001405                          BEQ     2$              ;Br if not
001140  010537  000000G                 MOV     R5,CSIARE       ;Set device name for the mount
001144  012700  000014'                 MOV     #MNTARG,R0      ;Mount the device
001150  104375                          EMT     375
                                ;
                                ;  Check the next logical disk
                                ;
001152  005205                  2$:     INC     R5              ;Advance LDn number
001154  062702  000002                  ADD     #2,R2           ;Advance logical disk index number
001160  020227  000000C                 CMP     R2,#<2*MAXLD>   ;More LD's to check?
001164  103736                          BLO     1$              ;Br if yes
                                ;
                                ;  Finished
                                ;
001166  012605                          MOV     (SP)+,R5
001170  012602                          MOV     (SP)+,R2
001172  000207                          RETURN
```

```
                                        .SBTTL  LDMNT  -- Set up information about LD being mounted
                                ;-------------------------------------------------------------------
                                ;  LDMNT is called to set up information about a logical disk.
                                ;
                                ;  Inputs:
                                ;   R2 = LD unit index
                                ;   R3 = Channel number to use for lookup
                                ;   CHNADR = Pointer to channel block for channel used for lookup
                                ;   EMTBLK (byte 0) = Free channel to use for looking up file.
                                ;   LDNAME(unit) = Name of file associated with logical disk.
                                ;
                                ;  Outputs:
                                ;   LDPDEV(unit) = Physical device index # and unit #
                                ;   LDSIZE(unit) = Size of file
                                ;   LDBASE(unit) = Base block on physical disk of start of logical disk.
                                ;   Carry-flag is set on return if file cannot be found.
                                ;
001174  010446                 LDMNT:  MOV     R4,-(SP)
001176  010546                         MOV     R5,-(SP)
                                ;
                                ;  Remove any entry for this logical disk from access control table
                                ;
001200  004737  002060'                 CALL    DLLDAC          ;Remove LD entry from access control table
                                ;
                                ;  Do lookup on file
                                ;
001204  010205                          MOV     R2,R5           ;Get LD unit index
001206  072527  000002                   ASH     #2,R5           ;4 words per entry
001212  062705  000000G                 ADD     #LDNAME,R5      ;Point to file spec in LDNAME table
001216  005715                          TST     (R5)            ;Is there a file spec for this LD?
001220  001474                          BEQ     9$              ;Br if not
001222  113746  000000G                 MOVB    SERFLG,-(SP)    ;Save error control byte
001226  112737  000001  000000G         MOVB    #1,SERFLG       ;Suppress error aborts
001234                                  .LOOKUP #CSIARE,R3,R5   ;Lookup the file
001254  112637  000000G                 MOVB    (SP)+,SERFLG    ;Restore error control byte
001260  103454                          BCS     9$              ;Br if could not find the file
001262  010062  000000G                 MOV     R0,LDSIZE(R2)   ;Save the size of the file
                                ;
                                ;  Set up information about the file
                                ;
001266  013705  000000G                 MOV     CHNADR,R5       ;Get pointer to channel block
001272  016500  000000G                 MOV     C.CSW(R5),R0    ;Get channel status word
001276  042700  177701                  BIC     #^C<76>,R0      ;Extract device index number
001302  110062  000000G                 MOVB    R0,LDPDEV(R2)   ;Save physical device index number
001306  116504  000000G                 MOVB    C.DEVQ(R5),R4   ;Get physical unit number
001312  042704  177770                  BIC     #^C<7>,R4       ;Extract unit number
001316  110462  000001G                 MOVB    R4,LDPDEV+1(R2) ;Save unit #
001322  016562  000000G 000000G         MOV     C.SBLK(R5),LDBASE(R2) ;Get base block # of file on disk
                                ;
                                ;  If we have read-only access to the file associated with the
                                ;  logical disk, set no-write flag for this LD entry.
                                ;
001330  032765  000000G 000000G         BIT     #CS$RON,C.CSW(R5);Do we have read-only access to the file?
001336  001403                          BEQ     2$              ;Br if not
001340  052762  000000G 000000G         BIS     #LD$RON,LDFLAG(R2);Set no-write flag for this LD
                                ;
                                ;  See if the logical disk file is itself within a logical disk
```

```
                                        ;  (i.e., this is a nested logical disk)
                                        ;
001346  020037  0000006         2$:     CMP     R0,LDDEVX       ;Is file on a logical disk?
001352  001007                        BNE     1$              ;Br if not
001354  006304                        ASL     R4              ;Cvt unit # to logical disk table index
001356  016462  0000006 0000006       MOV     LDPDEV(R4),LDPDEV(R2) ;Get real physical device & unit #
001364  066462  0000006 0000006       ADD     LDBASE(R4),LDBASE(R2) ;Bias base block # by log disk base
                                        ;
                                        ;  Purge the channel we used to open the file
                                        ;
001372                          1$:     .PURGE  R3              ;Purge channel we opened to the file
                                        ;
                                        ;  Make entry for the LD in the access control table
                                        ;
001402  004737  001704'                 CALL    ADLDAC          ;Make entry in access control table
                                        ;
                                        ;  Success
                                        ;
001406  000241                          CLC                     ;Signal success on return
001410  000403                          BR      10$
                                        ;
                                        ;  Error -- Could not find the file
                                        ;
001412  005062  0000006         9$:     CLR     LDPDEV(R2)      ;Say logical disk is not active
001416  000261                          SEC                     ;Signal error on return
001420  012605                  10$:    MOV     (SP)+,R5
001422  012604                          MOV     (SP)+,R4
001424  000207                          RETURN
```

```
                                        .SBTTL  ASNSRC -- Search assign table for logical name
                                ;-------------------------------------------------------------
                                ;  ASNSRC is called to search the assign table for an entry
                                ;  with a specified logical name.
                                ;
                                ;  Inputs:
                                ;   R0 = Logical name to search for.
                                ;
                                ;  Outputs:
                                ;   C-flag set on return if no assign block found with matching name.
                                ;   R2 = Address of assign block if one found.
                                ;
001426  010146                  ASNSRC: MOV     R1,-(SP)
001430  012701  000000G                 MOV     #MAXASN,R1      ;Get # assign entries
001434  012702  000000G                 MOV     #ASNTBL,R2      ;Point to assign table
001440  020062  000000G         1$:     CMP     R0,AT$LOG(R2)   ;Compare logical names
001444  001405                         BEQ     2$              ;Br if we found entry we are looking for
001446  062702  000000G                 ADD     #AT$$SZ,R2      ;Point to next assign block
001452  077106                         SOB     R1,1$           ;Loop if more blocks to check
001454  000261                         SEC                     ;Signal failure
001456  000401                         BR      3$
001460  000241                 2$:     CLC                     ;Signal success
001462  012601                 3$:     MOV     (SP)+,R1
001464  000207                         RETURN
```

```
                                        .SBTTL  LOGASN -- Perform full logical device assignment
                                ;-----------------------------------------------------------------
                                ;  LOGASN is called to perform a full logical device name assignment.
                                ;  The logical name associated with a file specification is translated
                                ;  into the corresponding physical device.  The file name, extension,
                                ;  and size may also be translated if a file spec was specified with
                                ;  the assignment of the logical name.
                                ;
                                ;  Inputs:
                                ;   R5 = Pointer to 5 word block containing file spec (dev,file,file,ext,size)
                                ;
                                ;  Outputs:
                                ;   File spec is updated to have physical device name and possibly
                                ;   altered file name.
                                ;
001466  010246                  LOGASN: MOV     R2,-(SP)
001470  010446                          MOV     R4,-(SP)
                                ;
                                ;  See if device name is in our assign table
                                ;
001472  011500                          MOV     (R5),R0         ;Get logical device name
001474  004737  001426'                 CALL    ASNSRC          ;See if name is in assign table
001500  103421                          BCS     1$              ;Br if name is not in assign table
                                ;
                                ;  Found logical device name in the assign table.
                                ;  Translate to physical device.
                                ;
001502  010504                          MOV     R5,R4           ;Get pointer to file spec buffer
001504  016224  000000G                 MOV     AT$DEV(R2),(R4)+;Put in physical device name
001510  016200  000000G                 MOV     AT$FIL(R2),R0   ;Was file name assigned?
001514  001413                          BEQ     1$              ;Br if not
001516  010024                          MOV     R0,(R4)+        ;Translate file name
001520  016224  000002G                 MOV     AT$FIL+2(R2),(R4)+
001524  016200  000000G                 MOV     AT$EXT(R2),R0   ;Was file extension specified?
001530  001405                          BEQ     1$              ;Br if not
001532  010024                          MOV     R0,(R4)+        ;Translate file extension
001534  016200  000000G                 MOV     AT$SIZ(R2),R0   ;Was file size specified?
001540  001401                          BEQ     1$              ;Br if not
001542  010014                          MOV     R0,(R4)         ;Translate file size
                                ;
                                ;  Translate "DK" and "SY" to physical device names
                                ;
001544  021537  000012'         1$:     CMP     (R5),R50SY      ;Is device name "SY"?
001550  001403                          BEQ     2$              ;Br if yes
001552  021537  000010'                 CMP     (R5),R50DK      ;Is device name "DK"?
001556  001002                          BNE     3$              ;Br if not
001560  013715  000000G         2$:     MOV     SYNAME,(R5)     ;Translate to physical device
                                ;
                                ;  Finished
                                ;
001564  012604                  3$:     MOV     (SP)+,R4
001566  012602                          MOV     (SP)+,R2
001570  000207                          RETURN
```

```
                                        .SBTTL  DEADEV -- Deassign physical device
                                ;-----------------------------------------------------------------
                                ;  DEADEV is called to remove from the assign table all entries
                                ;  for logical device names that are assigned to a specified
                                ;  physical device.
                                ;
                                ;  Inputs:
                                ;   R0 = Name of physical device.
                                ;
001572  010246                  DEADEV: MOV     R2,-(SP)
001574  010346                          MOV     R3,-(SP)
001576  012702  000000G                 MOV     #ASNTBL,R2      ;Point to assign table
001602  012703  000000G                 MOV     #MAXASN,R3      ;Get # assign table entries
001606  020062  000000G          1$:    CMP     R0,AT$DEV(R2)   ;Is this entry for specified phys device?
001612  001004                          BNE     2$              ;Br if not
001614  005062  000000G                 CLR     AT$LOG(R2)      ;Clear logical device name
001620  005062  000000G                 CLR     AT$DEV(R2)      ;Clear physical device name
001624  062702  000000G          2$:    ADD     #AT$$SZ,R2      ;Point to next assign entry
001630  077312                          SOB     R3,1$           ;Loop if more to check
001632  012603                          MOV     (SP)+,R3
001634  012602                          MOV     (SP)+,R2
001636  000207                          RETURN
```

```
                                        .SBTTL  LDDEVN -- See if device name is a LD unit
                                ;-----------------------------------------------------------------------
                                ;  LDDEVN is called to determine if a device name is of the form LDn and
                                ;  if so, to determine the unit number.
                                ;
                                ;  Inputs:
                                ;   R0 = Device name (rad50)
                                ;
                                ;  Outputs:
                                ;   C-flag cleared ==> Device is LDn
                                ;   C-flag set     ==> Device is not LDn
                                ;   R0 = LD device index number (2*n)
                                ;
001640  020037  000006'         LDDEVN: CMP     R0,R50LD        ;Is name "LD"?
001644  001002                          BNE     1$              ;Br if not
001646  013700  000002'                 MOV     R50LD0,R0       ;Replace "LD" by "LD0"
001652  020037  000002'         1$:     CMP     R0,R50LD0       ;Is name in the range LD0 to LD7?
001656  103410                          BLO     2$              ;Br if not
001660  020037  000004'                 CMP     R0,R50LD7
001664  101005                          BHI     2$
001666  163700  000002'                 SUB     R50LD0,R0       ;Get unit number only
001672  006300                          ASL     R0              ;Convert to device index number
001674  000241                          CLC                     ;Signal success on return
001676  000401                          BR      3$
001700  000261                  2$:     SEC                     ;Signal failure on return
001702  000207                  3$:     RETURN
```

```
                        .SBTTL  ADLDAC -- Add LD entry to access control table
                ;-----------------------------------------------------------------------
                ;  ADLDAC is called to add a logical disk entry to the device/file
                ;  access control table.  If there are no protected devices or files
                ;  no entry is made.
                ;
                ;  Inputs:
                ;   R2 = Logical disk index number (2 * unit number)
                ;
                ;  Outputs:
                ;   C-flag cleared ==> all ok
                ;   C-flag set     ==> Access table overflow
                ;
001704  010246          ADLDAC: MOV     R2,-(SP)
001706  010546                  MOV     R5,-(SP)
                ;
                ;  See if there are any entries in the access table
                ;
001710  005737  000000G         TST     RESDEV          ;Any entries in access table?
001714  001446                  BEQ     4$              ;Br if not
                ;
                ;  There are entries in the access control table
                ;
001716  006202                  ASR     R2              ;Convert ld index # to unit #
                ;
                ;  Find a free entry in the access table
                ;
001720  012705  000000G         MOV     #OKFILE,R5      ;Point to start of access table
001724  020537  000000G 1$:     CMP     R5,OKFNND       ;Reached end of table?
001730  103036                  BHIS    3$              ;Br if yes -- table overflow
001732  005765  000000G         TST     OF$FIL(R5)      ;Is this entry free?
001736  001403                  BEQ     2$              ;Br if free
001740  062705  000000G         ADD     #OF$$SZ,R5      ;Point to next entry
001744  000767                  BR      1$              ;Go check it
                ;
                ;  We found a free entry.  Add entry for LD.
                ;
001746  113765  000000G 000000G 2$:     MOVB    LDDEVX,OF$DEV(R5);Set logical disk device index number
001754  110265  000000G         MOVB    R2,OF$UNT(R5)   ;Set logical disk unit #
001760  012700  000000G         MOV     #WLDNAM,R0      ;Set file name to wildcards
001764  010065  000000G         MOV     R0,OF$FIL(R5)
001770  010065  000002G         MOV     R0,OF$FIL+2(R5)
001774  010065  000004G         MOV     R0,OF$FIL+4(R5)
002000  105065  000000G         CLRB    OF$FLG(R5)      ;Initially clear all control flags
002004  006302                  ASL     R2              ;Cvt unit # to LD index #
002006  032762  000000G 000000G BIT     #LD$RON,LDFLAG(R2);Is logical disk write protected?
002014  001406                  BEQ     4$              ;Br if not
002016  152765  000000G 000000G BISB    #OT$RON,OF$FLG(R5);Set read-only flag in access table
002024  000402                  BR      4$
                ;
                ;  Error -- Access table overflow
                ;
002026  000261          3$:     SEC                     ;Signal error on return
002030  000410                  BR      10$
                ;
                ;  Success! If this entry extends the ACCESS table, bump up the
                ;  table end pointer to include it.
```

```
     58                                     ;
     59 002032  062705  0000006            4$:     ADD     #OF$$SZ,R5      ;Point past this LD ACCESS entry
     60 002036  020537  0000006                    CMP     R5,OKFAND       ;Do we need to increase last ACCESS ptr?
     61 002042  101402                          BLOS    41$             ;Br if not
     62 002044  010537  0000006                    MOV     R5,OKFAND       ;Extend ACCESS table to contain entry
     63 002050  000241                      41$:    CLC                     ;Signal success on return
     64                                     ;
     65                                     ;  Finished
     66                                     ;
     67 002052  012605                      10$:    MOV     (SP)+,R5
     68 002054  012602                          MOV     (SP)+,R2
     69 002056  000207                          RETURN
```

```
                                        .SBTTL  DLLDAC -- Delete LD entry from access control table
                                ;-----------------------------------------------------------------------
                                ;  DLLDAC is called to delete any entry in the access control table
                                ;  for a specified logical disk.
                                ;
                                ;  Inputs:
                                ;   R2 = Logical disk index number (2 * unit number)
                                ;
002060  004737  002124'         DLLDAC: CALL    CKLDAC          ;Is there an entry for this LD in table?
002064  103416                          BCS     1$              ;Br if not
                                ;
                                ;  We have found an entry in the access table for this LD.
                                ;  R0 = Pointer to the entry.
                                ;
002066  105060  000000G                 CLRB    OF$DEV(R0)      ;Mark entry as free
002072  105060  000000G                 CLRB    OF$UNT(R0)
002076  005060  000000G                 CLR     OF$FIL(R0)
                                ;
                                ;  If the ACCESS table was extended to include this entry,
                                ;  reduce the table end ptr below it.
                                ;
002102  062700  000000G                 ADD     #OF$$SZ,R0      ;Point past this entry
002106  020037  000000G                 CMP     R0,OKFAND       ;Was it the last entry?
002112  103403                          BLO     1$              ;Br if not
002114  162737  000000G 000000G        SUB     #OF$$SZ,OKFAND  ;It was, lower the end below it
                                ;
                                ;  Finished
                                ;
002122  000207                 1$:     RETURN
```

```
                                        .SBTTL  CKLDAC -- Check if LD is in access control table
                                ;-------------------------------------------------------------------------
                                ;  CKLDAC is called to determine if a certain logical disk is in the
                                ;  device/file access control table.
                                ;
                                ;  Inputs:
                                ;   R2 = Logical disk index number (2 * unit #)
                                ;
                                ;  Outputs:
                                ;   C-flag cleared ==> Found logical disk entry in access control table.
                                ;   C-flag set     ==> Logical disk entry is not in access table.
                                ;   R0 = Pointer to access control entry.
                                ;
002124  010246                  CKLDAC: MOV     R2,-(SP)        ;save original LD index number
002126  006202                          ASR     R2              ;Convert index # to unit #
002130  005737  000000G                 TST     RESDEV          ;Are there any entries in access table?
002134  001417                          BEQ     4$              ;Br if not
                                ;
                                ;  There are entries in the access control table.
                                ;  Search for entry matching our logical disk.
                                ;
002136  012700  000000G                 MOV     #OKFILE,R0      ;Point to start of access table
002142  020037  000000G         1$:     CMP     R0,OKFAND       ;Checked all entries?
002146  103012                          BHIS    4$              ;Br if so
002150  126037  000000G 000000G         CMPB    OF$DEV(R0),LDDEVX ;Is this entry for a logical disk?
002156  001003                          BNE     2$              ;Br if not
002160  120260  000000G                 CMPB    R2,OF$UNT(R0)   ;Is entry for specified unit?
002164  001405                          BEQ     3$              ;Br if yes -- found entry
002166  062700  000000G         2$:     ADD     #OF$$SZ,R0      ;Point to next access entry
002172  000763                          BR      1$              ;Check all entries
                                ;
                                ;  LD entry is not in access table
                                ;
002174  000261                  4$:     SEC                     ;Signal failure on return
002176  000401                          BR      9$
                                ;
                                ;  Found LD entry in table
                                ;
002200  000241                  3$:     CLC                     ;Signal success on return
002202  012602                  9$:     MOV     (SP)+,R2        ;Recover LD index number
002204  000207                          RETURN
```

```
                                        .SBTTL  LDOFCK -- See if any files are open on a logical disk
                                ;-------------------------------------------------------------------
                                ;  LDOFCK is called to determine if the current job has any files
                                ;  open on a specified logical disk.
                                ;
                                ;  Inputs:
                                ;   R2 = LD unit index
                                ;
                                ;  Outputs:
                                ;   C-flag set if any files are open on the LD.
                                ;
002206  010146                  LDOFCK: MOV     R1,-(SP)
002210  010346                          MOV     R3,-(SP)
002212  010446                          MOV     R4,-(SP)
002214  010546                          MOV     R5,-(SP)
                                ;
                                ;  Begin loop to check each channel
                                ;
002216  016204  000000G                 MOV     LDBASE(R2),R4   ;Get base block # of LD
002222  066204  000000G                 ADD     LDSIZE(R2),R4   ;Get block # above top of LD
002226  005003                          CLR     R3              ;Start with channel 0
                                ;
                                ;  Compute address of channel block for this channel
                                ;
002230  010305                  1$:     MOV     R3,R5           ;Get channel number
002232  013700  000000G                 MOV     CXTRMN,R0       ;Get address of simulated rmon
002236  062700  000000G                 ADD     #R$CHN,R0       ;Point to 1st channel area
002242  020527  000021                  CMP     R5,#17.         ;Is this channel in extended channel area?
002246  103404                          BLO     2$              ;Br if not
002250  162705  000021                  SUB     #17.,R5         ;Get channel # in extended channel area
002254  062700  000000C                 ADD     #R$XCHN-R$CHN,R0;Point to extended channel area
002260  070527  000000G          2$:    MUL     #CHNSIZ,R5      ;Multiply by # bytes per channel
002264  060005                          ADD     R0,R5           ;Get absolute address of channel block
                                ;
                                ;  See if this channel is open
                                ;
002266  016500  000000G                 MOV     C.CSW(R5),R0    ;Get channel status word
002272  032700  000000G                 BIT     #CS$OPN,R0      ;Is this channel open?
002276  001427                          BEQ     3$              ;Br if not
                                ;
                                ;  See if file is within LD
                                ;
002300  042700  177701                  BIC     #^C<76>,R0      ;Extract device index number
002304  020037  000000G                 CMP     R0,LDDEVX       ;Is file on a logical disk?
002310  001022                          BNE     3$              ;Br if not
002312  116501  000000G                 MOVB    C.DEVQ(R5),R1   ;Get unit number
002316  042701  177770                  BIC     #^C<7>,R1       ;Extract unit #
002322  006301                          ASL     R1              ;Convert unit # to index
002324  026162  000000G 000000G         CMP     LDPDEV(R1),LDPDEV(R2);Are LD's on same physical drive?
002332  001011                          BNE     3$              ;Br if not
002334  016500  000000G                 MOV     C.SBLK(R5),R0   ;Get base block of file
002340  066100  000000G                 ADD     LDBASE(R1),R0   ;Add base block of LD file is on
002344  020062  000000G                 CMP     R0,LDBASE(R2)   ;Is it below base of LD?
002350  103402                          BLO     3$              ;Br if yes
002352  020004                          CMP     R0,R4           ;Is it above top of LD?
002354  103406                          BLO     4$              ;Br if it is within LD
                                ;
```

```
     58                                 ;  Check next channel
     59                                 ;
     60 002356  005203                  3$:     INC     R3                  ;Advance channel #
     61 002360  020327  000000G                 CMP     R3,#NLCHN           ;Checked all channels?
     62 002364  103721                          BLO     1$                  ;Br if more to check
     63                                 ;
     64                                 ;  No file is open on the LD
     65                                 ;
     66 002366  000241                          CLC                         ;Signal OK on return
     67 002370  000401                          BR      9$
     68                                 ;
     69                                 ;  There is a file open on the LD
     70                                 ;
     71 002372  000261                  4$:     SEC                         ;Signal error on return
     72                                 ;
     73                                 ;  Finished
     74                                 ;
     75 002374  012605                  9$:     MOV     (SP)+,R5
     76 002376  012604                          MOV     (SP)+,R4
     77 002400  012603                          MOV     (SP)+,R3
     78 002402  012601                          MOV     (SP)+,R1
     79 002404  000207                          RETURN
     80
     81         000001                          .END
Errors detected:  0

*** Assembler statistics

Work  file  reads:  0
Work  file writes:  0
Size of work file:  9621 Words  ( 38 Pages)
Size of core pool:  18176 Words  ( 71 Pages)
Operating  system:  RT-11

Elapsed time:  00:00:23.01
,LP:TSEM5=DK:TSEM5/C/N:SYM
```

Cross reference table (CREF V05.05)

```
...V1     3-109    9-34     9-34     9-34     9-68
...V2     3-109    3-109    3-109#   3-109#   9-34     9-34     9-34#    9-34#    9-68     9-68#
ADLDAC    9-72    14-14#
ASNSRC   10-13#   11-22
ASNTBL    1-38    10-15    12-12
AT$$SZ    1-39    10-18    12-18
AT$DEV    1-39    11-29    12-14    12-17*
AT$EXT    1-39    11-34
AT$FIL    1-39    11-30    11-33
AT$LOG    1-38    10-16    12-16*
AT$SIZ    1-39    11-37
C.CSW     1-36     3-15     9-42     9-53    17-37
C.DEVQ    1-35     9-45    17-46
C.SBLK    1-35     9-48    17-51
CHNADR    1-36     3-14     9-41
CHNSIZ    1-34    17-32
CKLDAC    15-9    16-14#
CS$OPN    1-36     3-15    17-38
CS$RON    1-35     9-53
CSIARE    1-37     1-54     1-59     3-109    3-116*   7-25*    8-27*    8-42*    9-34
CSIFIL    1-37     3-50     3-59     3-61     3-63     3-70     3-83     3-109
CXTRMN    1-34    17-26
DEADEV   12-10#
DIEARG    2-18
DIEMSG    2-18
DLLDAC    9-23    15-9#
DMTARG    1-58#    7-28     8-30
EMTBLK    1-36     3-22     3-47     3-91     3-98     3-108    3-114    3-123    4-12     5-15     5-23
EMTXIT    1-38    3-128     4-37     5-61     6-42
LD$RON    1-38     3-93     5-42     9-55    14-46
LDAEMT    1-30     6-8#
LDBASE    1-35    9-48*     9-64     9-64*    17-19    17-52    17-53
LDCLEN    3-124    8-10#
LDDEMT    1-30     4-8#
LDDEVN    3-71    13-14#
LDDEVX    1-35     9-60    14-38    16-25    17-44
LDDMT     4-30     6-28     7-13#
LDFLAG    1-37    3-90*    3-93*     5-42     9-55*    14-46
LDIEMT    1-30     5-11#
LDMEMT    1-30     3-10#
LDMNT     3-99     8-36     9-18#
LDNAME    1-37     3-40     3-82     4-23     5-31     6-23     7-37*    8-22     9-29
LDOFCK    7-17    17-12#
LDPDEV    1-38     5-45     7-34*    8-40     9-44*    9-47*    9-63     9-63*    9-81*    17-49    17-49
LDSIZE    1-35    9-37*    17-20
LOGASN    3-60    11-16#
MAXASN    1-38    10-14    12-13
MAXLD     1-36     3-23     4-13     5-16     8-50
MNTARG    1-53#    3-117    8-43
NLCHN     1-34    17-61
OF$$SZ    1-40    14-33    14-59    15-22    15-25    16-29
OF$DEV    1-40    14-38*   15-15*   16-25
OF$FIL    1-40    14-31    14-41*   14-42*   14-43*   15-17*
OF$FLG    1-41    14-44*   14-48*
OF$UNT    1-41    14-39*   15-16*   16-27
OKFAND    1-40    14-60    14-62*   15-23    15-25*   16-23
```

Cross reference table (CREF V05.05)

```
OKFILE   1-40     14-28     16-22
OKFNND   1-40     14-29
OT$RON   1-41     14-48
OVRHC    1-36
PS       1-22#
R$CHN    1-34     17-27     17-31
R$XCHN   1-34     17-31
R50DK    1-48#    11-45
R50LD    1-47#    13-14
R50LDO   1-45#     3-115     7-24      8-15      13-16     13-17     13-21
R50LD7   1-46#    13-19
R50SY    1-49#    11-43
RESDEV   1-40     14-19     16-16
SERFLG   1-38      3-106     3-107*    3-110*    7-26      7-27*     7-30*     8-28      8-29*     8-32*     9-32      9-33*
         9-35*
SETERR   1-36      3-18      3-26      3-33      3-43      3-66      3-76      3-102     4-16      4-26      4-33      5-19
         6-38
SYNAME   1-39     11-47
SYSHLT   2-18
TSEM2    1-21#
VALADW   1-37      3-49      5-24
VLDSYS   1-37      3-30
WLDNAM   1-41     14-40
```

Cross reference table (CREF V05.05)

```
...CM1    3-109      9-34
...CM2    3-109      3-109      9-34       9-34
...CM3    9-68
...CM5    3-109      9-34
.FPROT    1-26#      3-109
.LOOKU    1-26#      9-34
.PURGE    1-26#      9-68
DIE       2-19#
DISABL    2-4#
ENABL     2-8#
OCALL     2-31#
```