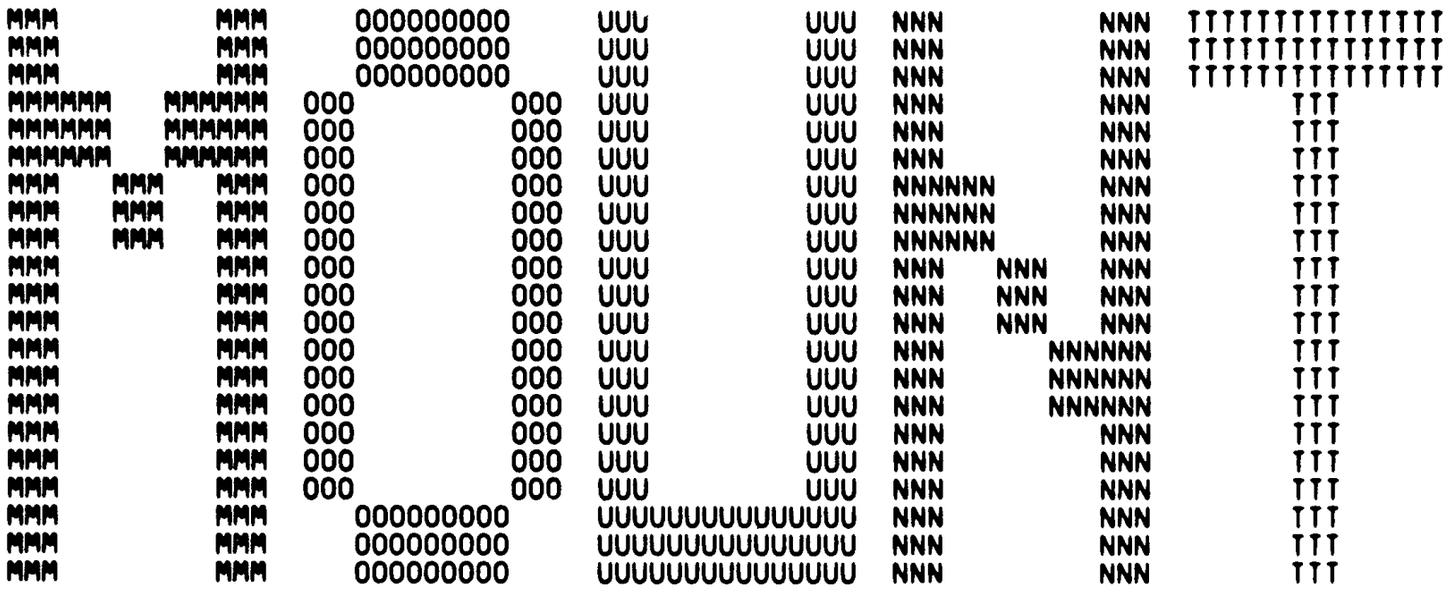
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMM MMM MMM 000 000 UUU UUU NNNNNN NNN TTT
MMM MMM MMM 000 000 UUU UUU NNNNNN NNN TTT
MMM MMM MMM 000 000 UUU UUU NNNNNN NNN TTT
MMM MMM 000 000 UUU UUU NNN NNN NNN TTT
MMM MMM 000 000 UUU UUU NNN NNN NNN TTT
MMM MMM 000 000 UUU UUU NNN NNN NNN TTT
MMM MMM 000 000 UUU UUU NNN NNNNNN TTT
MMM MMM 000 000 UUU UUU NNN NNNNNN TTT
MMM MMM 000 000 UUU UUU NNN NNNNNN TTT
MMM MMM 000 000 UUU UUU NNN NNN TTT
MMM MMM 000 000 UUU UUU NNN NNN TTT
MMM MMM 000 000 UUU UUU NNN NNN TTT
MMM MMM 000000000 UUUUUUUUUUUUUUUU NNN NNN TTT
MMM MMM 000000000 UUUUUUUUUUUUUUUU NNN NNN TTT
MMM MMM 000000000 UUUUUUUUUUUUUUUU NNN NNN TTT
MMM MMM 000000000 UUUUUUUUUUUUUUUU NNN NNN TTT

FILEID**MOUNTDSP

```
MM      MM    OOOOOO    UU      UU  NN      NN  TTTTTTTTTT  DDDDDDDD     SSSSSSSS  PPPPPPPP
MM      MM    OOOOOO    UU      UU  NN      NN  TTTTTTTTTT  DDDDDDDD     SSSSSSSS  PPPPPPPP
MMMM  MMMM  OO      OO  UU      UU  NN      NN      TT      DD      DD  SS        PP      PP
MMMM  MMMM  OO      OO  UU      UU  NN      NN      TT      DD      DD  SS        PP      PP
MM  MM  MM  OO      OO  UU      UU  NNNN    NN      TT      DD      DD  SS        PP      PP
MM  MM  MM  OO      OO  UU      UU  NNNN    NN      TT      DD      DD  SS        PP      PP
MM      MM  OO      OO  UU      UU  NN  NN  NN      TT      DD      DD    SSSSSS  PPPPPPPP
MM      MM  OO      OO  UU      UU  NN  NN  NN      TT      DD      DD    SSSSSS  PPPPPPPP
MM      MM  OO      OO  UU      UU  NN    NNNN      TT      DD      DD        SS  PP
MM      MM  OO      OO  UU      UU  NN    NNNN      TT      DD      DD        SS  PP
MM      MM  OO      OO  UU      UU  NN      NN      TT      DD      DD        SS  PP
MM      MM  OO      OO  UU      UU  NN      NN      TT      DD      DD        SS  PP
MM      MM    OOOOOO    UUUUUUUUUU  NN      NN      TT      DDDDDDDD  SSSSSSSS  PP
MM      MM    OOOOOO    UUUUUUUUUU  NN      NN      TT      DDDDDDDD  SSSSSSSS  PP

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

Table of contents

```
0000     1          .TITLE  MOUNTDSP - MOUNT system service dispatcher
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;****************************************************************************
0000     5 ;*                                                                          *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000     8 ;*  ALL RIGHTS RESERVED.                                                    *
0000     9 ;*                                                                          *
0000    10 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    11 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    12 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    13 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    14 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    15 ;*  TRANSFERRED.                                                            *
0000    16 ;*                                                                          *
0000    17 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    18 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    19 ;*  CORPORATION.                                                            *
0000    20 ;*                                                                          *
0000    21 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    22 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    23 ;*                                                                          *
0000    24 ;*                                                                          *
0000    25 ;****************************************************************************
0000    26 ;
0000    27 ;
0000    28 ; Facility: $MOUNT System Service
0000    29 ;++
0000    30 ; Abstract:
0000    31 ;       This module contains the dispatcher for the $MOUNT system service,
0000    32 ;       which is implemented as a privileged shareable image.
0000    33 ;
0000    34 ; Overview:
0000    35 ;       The $MOUNT system service is contained as a privileged
0000    36 ;       shareable image that is linked into user program images in exactly
0000    37 ;       the same fashion as any shareable image.  The creation and installation
0000    38 ;       of a privileged, shareable image is slightly different from that
0000    39 ;       of an ordinary shareable image.  These differences are:
0000    40 ;
0000    41 ;               1. A vector defining the entry points and providing other
0000    42 ;                  control information to the image activator.  This vector
0000    43 ;                  is a the lowest address in an image section with the VEC
0000    44 ;                  attribute.
0000    45 ;
0000    46 ;               2. The shareable image is linked with the /PROTECT option
0000    47 ;                  that marks all of the image sections so that they will
0000    48 ;                  protected and given EXEC mode ownership by the image
0000    49 ;                  activator.
0000    50 ;
0000    51 ;               3. The shareable image MUST be installed /SHARE /PROTECT
0000    52 ;                  with the INSTALL utility in order for the image activator
0000    53 ;                  to connect the privileged shareable image to the change mode
0000    54 ;                  dispatchers.
0000    55 ;
0000    56 ;       A privileged shareable image implementing system services is comprised
0000    57 ;       of the following major components:
```

```
                                                M 3
0000     58 ;
0000     59 ;               1. A transfer vector containing all of the entry points and
0000     60 ;                  collecting them at the lowest virtual address in the shareable
0000     61 ;                  image.  This formalism enables revision of the shareable
0000     62 ;                  image without necessitating the relinking of images that
0000     63 ;                  use it.
0000     64 ;
0000     65 ;               2. A Privileged Library Vector in a PSECT with the VEC attribute
0000     66 ;                  that describes the entry points for dispatching EXEC and
0000     67 ;                  KERNEL mode services along with validation information.
0000     68 ;
0000     69 ;               3. A dispatcher for kernel mode services.  This code will
0000     70 ;                  be called by the VMS change mode dispatcher when it
0000     71 ;                  fails to recognize a kernel mode service request.
0000     72 ;
0000     73 ;               4. A dispatcher for executive mode services.  This code will
0000     74 ;                  be called by the VMS change mode dispatcher when it fails
0000     75 ;                  to recognize an executive mode service request.
0000     76 ;
0000     77 ;               5. Service routines to perform the various services.
0000     78 ;
0000     79 ;
0000     80 ; Environment:
0000     81 ;
0000     82 ;       VAX/VMS operating system, installed as a privliged shareable image.
0000     83 ;
0000     84 ;--
0000     85 ;
0000     86 ; Author:
0000     87 ;
0000     88 ;       Steven T. Jeffreys
0000     89 ;
0000     90 ; Modified by:
0000     91 ;
0000     92 ;       V03-001 STJ50311        Steven T. Jeffreys,             24-Feb-1983
0000     93 ;               - Move FP to SP before dispatching to called routine.  This
0000     94 ;                 is necessary because high level langauges (like BLISS)
0000     95 ;                 assume that FP=SP at routine entry.
0000     96 ;               - Add kernel dipatch vector for $DALLOC_DEV'.
0000     97 ;
0000     98 ;       V02-002 STJ0204         Steven T. Jeffreys,             07-Feb-1982
0000     99 ;               Added R4 to the register save mask.
0000    100 ;
0000    101 ;       V01-001 STJ0158         Steven T. Jeffreys,             04-Jan-1981
0000    102 ;               Removed .LIBRARY reference to SYS$LIBRARY:LIB.MLB
0000    103 ;
0000    104 ;--
```

```
0000   106          .SBTTL  Declarations and Equates
0000   107 ;
0000   108 ;        Include Files
0000   109 ;
0000   110
0000   111 ;
0000   112 ;        Macro Definitions
0000   113 ;
0000   114 ;        DEFINE_SERVICE - A macro to make the appropriate entries in several
0000   115 ;                         different PSECTs required to define an EXEC or KERNEL
0000   116 ;                         mode service.  These include the transfer vector,
0000   117 ;                         the case table for dispatching, and a table containing
0000   118 ;                         the number of required arguments.
0000   119 ;
0000   120 ;                         Note that the case table that is created is used to
0000   121 ;                         branch to another table which will in turn JMP to the
0000   122 ;                         actual routine entry point.  This is done because the
0000   123 ;                         routine entry point may be further away than the
0000   124 ;                         signed-word branch displacements used by the CASE
0000   125 ;                         instruction can reach.
0000   126 ;
0000   127 ;                         In addition, a special provision has been made to
0000   128 ;                         allow for system services that execute in the access
0000   129 ;                         mode of the caller.  In effect, the routines are
0000   130 ;                         called directly by the user.
0000   131 ;
0000   132 ;        DEFINE_SERVICE Name,Number_of_Arguments,Mode
0000   133 ;
0000   134          .MACRO  DEFINE_SERVICE,NAME,NARG=0,MODE=KERNEL
0000   135          .PSECT  $$$TRANSFER_VECTOR,PAGE,NOWRT,EXE,PIC
0000   136          .ALIGN  QUAD                            ; Align entry points for speed and style
0000   137 NAME'$U::                                       ; Define a convenient label that can be
0000   138                                                  ; referenced at link time by internal
0000   139                                                  ; modules.
0000   140          .TRANSFER       NAME                    ; Define name as universal symbol for entry
0000   141          .MASK   NAME,^M<R4>                     ; Use entry mask defined in main routine
0000   142
0000   143          ;
0000   144          ; Define KERNEL transfer vector
0000   145          ;
0000   146          .IF     IDN MODE,KERNEL
0000   147          CHMK    #<KCODE_BASE+KERNEL_COUNTER> ; Change to kernel mode and execute
0000   148          RET                                     ; Return
0000   149          KERNEL_COUNTER=KERNEL_COUNTER+1 ; Advance counter
0000   150
0000   151          .PSECT  KERNEL_NARG,BYTE,NOWRT,EXE,PIC
0000   152          .BYTE   NARG                            ; Define number of required arguments
0000   153
0000   154          .PSECT  MOUNTDSP_KERNEL_DISP1,BYTE,NOWRT,EXE,PIC
0000   155          .WORD   NAME'$K-KCASE_BASE              ; Make entry in kernel mode CASE table
0000   156
0000   157          .PSECT  MOUNTDSP_KERNEL_DISP3,BYTE,NOWRT,EXE,PIC
0000   158 NAME'$K:JMP     G^NAME+2                        ; Make entry in kernel mode JMP table
0000   159          .ENDC
0000   160
0000   161          ;
0000   162          ; Define EXEC transfer vector.
```

```
                                                B 4
0C00   163          ;
0000   164          .IF     IDN MODE,EXEC
0000   165          CHME    #<ECODE_BASE+EXEC_COUNTER> ; Change to executive mode and execute
0000   166          RET                         ; Return
0000   167          EXEC_COUNTER=EXEC_COUNTER+1 ; Advance counter
0000   168
0000   169          .PSECT  EXEC_NARG,BYTE,NOWRT,EXE,PIC
0000   170          .BYTE   NARG                ; Define number of required arguments
0000   171
0000   172          .PSECT  MOUNTDSP_EXEC_DISP1,BYTE,NOWRT,EXE,PIC
0000   173          .WORD   NAME'$E-ECASE_BASE  ; Make entry in exec mode CASE table
0000   174
0000   175          .PSECT  MOUNTDSP_EXEC_DISP3,BYTE,NOWRT,EXE,PIC
0000   176 NAME'$E:JMP      G^NAME+2           ; Make entry in exec mode JMP table
0000   177          .ENDC                       ;
0000   178
0000   179          ;
0000   180          ; Define a transfer vector that will allow execution
0000   181          ; to occur in the arcess mode of the caller.
0000   182          ;
0000   183          .IF     IDN MODE,ALL
0000   184          JMP     G^NAME+2
0000   185          .ENDC
0000   186
0000   187
0000   188          .ENDM   DEFINE_SERVICE      ;
0000   189
0000   190 ;
0000   191 ; Define a macro that will perform a check on the number of KERNEL and
0000   192 ; EXEC vectors defined.  If too many are defined, issue a compile-time
0000   193 ; error message.
0000   194 ;
0000   195          .MACRO  CHECK_VECTORS
0000   196          .IF     GREATER KERNEL_COUNTER-<KCODE_VECTORS-1>
0000   197          .ERROR  1                   ; Too many KERNEL vectors
0000   198          .ENDC
0000   199          .IF     GREATER EXEC_COUNTER-<ECODE_VECTORS-1>
0000   200          .ERROR 2                    ; Too many EXEC vectors
0000   201          .ENDC
0000   202          .ENDM   CHECK_VECTORS
0000   203
0000   204 ;
0000   205 ;
0000   206 ;        Equated Symbols
0000   207 ;
0000   208
0000   209          $PLVDEF                     ; Define PLV offsets and values
0000   210          $SSDEF                      ; Define system status codes
0000   211 ;
0000   212 ;        Initialize counters for change mode dispatching codes
0000   213 ;
00000000  0000   214 EXEC_COUNTER=0              ; Exec code counter
00000000  0000   215 KERNEL_COUNTER=0            ; Kernel code counter
0000   216 ;
0000   217 ;        Own Storage
0000   218 ;
00000000   219          .PSECT  KERNEL_NARG,BYTE,NOWRT,EXE,PIC
```

```
                                                C 4
     0000    220 KERNEL_NARG:                                   ; Base of byte table containing the
     0000    221                                                ;  number of required arguments.
 00000000    222                .PSECT  EXEC_NARG,BYTE,NOWRT,EXE,PIC
     0000    223 EXEC_NARG:                                     ; Base of byte table containing the
     0000    224                                                ;  number of required arguments.
```

```
                        226          .SBTTL  Transfer Vector and Service Definitions
                        227 ;++
                        228 ; The use of transfer vectors to effect entry to the system services
                        229 ; enables some updating of the shareable image containing them without necessitating
                        230 ; a re-link of all programs that call them.  The PSECT containinng the transfer
                        231 ; vector will be positioned at the lowest virtual address in the shareable image
                        232 ; and so long as the transfer vector is not re-ordered, programs linked with
                        233 ; one version of the shareable image will continue to work with the next.
                        234 ;
                        235 ; Thus as additional services are added to a privileged shareable image, their
                        236 ; definitions should be added to the end of the following list to ensure that
                        237 ; programs using previous versions of it will not need to be re-linked.
                        238 ; To completely avoid relinking existing programs the size of the privileged
                        239 ; shareable image must not change so some padding will be required to provide
                        240 ; the opportunity for future growth.
                        241 ;--
                        242
                        243          DEFINE_SERVICE  SYS$MOUNT,1,ALL          ; $MOUNT main entry point
                        244          DEFINE_SERVICE  SYS$VMOUNT,1,EXEC        ; Internal MOUNT routine
                        245          DEFINE_SERVICE  $COPY_INFO,2,EXEC        ; Internal MOUNT routine
                        246          DEFINE_SERVICE  $CHANGE_PROT,0,EXEC      ; Internal MOUNT routine
                        247          DEFINE_SERVICE  $DALLOC_DEVS,1,KERNEL    ; Internal MOUNT routine
                        248 ;
                        249 ; The base values used to generate the dispatching codes should be negative for
                        250 ; user services and must be chosen to avoid overlap with any other privileged
                        251 ; shareable images that will be used concurrently.  Their definition is
                        252 ; deferred to this point in the assembly to cause their use in the preceding
                        253 ; macro calls to be forward references that guarantee the size of the change
                        254 ; mode instructions to be four bytes.  This satisfies an assumption that is
                        255 ; made by for services that have to wait and be retried.  The PC for retrying
                        256 ; the change mode instruction that invokes the service is assumed to be 4 bytes
                        257 ; less than that saved in the change mode exception frame.  Of course, the particula
                        258 ; service routine determines whether this is possible.
                        259 ;
                        260 ; The change-mode codes available to $MOUNT are the decimal values
                        261 ; from 16522 to 16527.  Allow for 4 EXEC and 2 KERNEL change mode vectors.
                        262 ;
0000408A                263 ECODE_BASE=16522                              ; Base CHME code value for these services
00000004                264 ECODE_VECTORS=4                              ; Number of EXEC change mode vectors
0000408E                265 KCODE_BASE=<ECODE_BASE+ECODE_VECTORS>        ; Base CHMK code value for these services
00000002                266 KCODE_VECTORS=2                              ; Number of KERNEL change mode vectors
```

```
                        .SBTTL  Change Mode Dispatcher Vector Block
                ;++
                ; This vector is used by the image activator to connect the privileged shareable
                ; image to the VMS change mode dispatcher.  The offsets in the vector are self-
                ; relative to enable the construction of position independent images.  The system
                ; version number will be used by the image activator to verify that this shareable
                ; image was linked with the symbol table for the current system.
                ;
                ;                               Change Mode Vector Format
                ;
                ;       +-----------------------------------------+
                ;       |           Vector Type Code              |   PLV$L_TYPE
                ;       |        (PLV$C_TYP_^MOD)                 |
                ;       +-----------------------------------------+
                ;       |        System Version Number            |   PLV$L_VERSION
                ;       |           (SYS$K_VERSION)               |
                ;       +-----------------------------------------+
                ;       |      Kernel Mode Dispatcher Offset      |   PLV$L_KERNEL
                ;       |                                         |
                ;       +-----------------------------------------+
                ;       |         Exec Mode Entry Offset          |   PLV$L_EXEC
                ;       |                                         |
                ;       +-----------------------------------------+
                ;       |                Reserved                 |
                ;       |                                         |
                ;       +-----------------------------------------+
                ;       |                Reserved                 |
                ;       |                                         |
                ;       +-----------------------------------------+
                ;       |         RMS Dispatcher Offset           |   PLV$L_RMS
                ;       |                                         |
                ;       +-----------------------------------------+
                ;       |              Address Check              |   PLV$L_CHECK
                ;       |                                         |
                ;       +-----------------------------------------+
                ;
00000000        .PSECT  MOUNTDSP,PAGE,VEC,PIC,NOWRT,EXE
                
00000001        .LONG   PLV$C_TYP_CMOD          ; Set type of vector to change mode dispatch
00000000'       .LONG   SYS$K_VERSION           ; Identify system version
00000003'       .LONG   KERNEL_DISPATCH-.       ; Offset to kernel mode dispatcher
FFFFFFFF'       .LONG   EXEC_DISPATCH-.         ; Offset to executive mode dispatcher
00000000        .LONG   0                       ; Reserved.
00000000        .LONG   0                       ; Reserved.
00000000        .LONG   0                       ; No RMS dispatcher
00000000        .LONG   0                       ; Address check - PIC image
```

```
                     0020   316          .SBTTL  Kernel Mode Dispatcher
                     0020   317 ;++
                     0020   318 ; Input Parameters:
                     0020   319 ;
                     0020   320 ;       (SP) - Return address if bad change mode value
                     0020   321 ;
                     0020   322 ;        R0  - Change mode argument value.
                     0020   323 ;
                     0020   324 ;        R4  - Current PCB Address. (Therefore R4 must be specified in all
                     0020   325 ;                register save masks for kernel routines.)
                     0020   326 ;
                     0020   327 ;        AP  - Argument pointer existing when the change
                     0020   328 ;              mode instruction was executed.
                     0020   329 ;
                     0020   330 ;        FP  - Address of minimal call frame to exit
                     0020   331 ;              the change mode dispatcher and return to
                     0020   332 ;              the original mode.
                     0020   333 ;--
                00000000    334          .PSECT  MOUNTDSP_KERNEL_DISP0,BYTE,NOWRT,EXE,PIC
                     0000   335 KACCVIO:                                ; Kernel access violation
     50    0C      3C  0000   336          MOVZWL  #SS$_ACCVIO,R0       ; Set access violation status code
                   04  0003   337          RET                          ;  and return
                     0004   338 KINSFARG:                               ; Kernel insufficient arguments.
50   0114  8F      3C  0004   339          MOVZWL  #SS$_INSFARG,R0      ; Set status code and
                   04  0009   340          RET                          ;  return
                   05  000A   341 KNOTME: RSB                          ; RSB to forward request
                     000B   342
                     000B   343 KERNEL_DISPATCH::                       ; Entry to dispatcher
51   BF72  C0      9E  000B   344          MOVAB   W^-KCODE_BASE(R0),R1 ; Normalize dispatch code value
         F8      19  0010   345          BLSS    KNOTME               ; Branch if code value too low
     01  51      B1  0012   346          CMPW    R1,#KERNEL_COUNTER   ; Check high limit
         F3      1E  0015   347          BGEQU   KNOTME               ; Branch if out of range
                     0017   348 ;
                     0017   349 ; The dispatch code has now been verified as being handled by this dispatcher,
                     0017   350 ; now the argument list will be probed and the required number of arguments
                     0017   351 ; verified.
                     0017   352 ;
     51  0000'CF41      9A  0017   353          MOVZBL  W^KERNEL_NARG[R1],R1 ; Get required argument count
51    00000004 9F41      DE  001D   354          MOVAL   @#4[R1],R1           ; Compute byte count including arg count
                         25   355          IFNORD  R1,(AP),KACCVIO      ; Branch if arglist not readable
     BF72'CF40      6C  91  002B   356          CMPB    (AP),W^<KERNEL_NARG-KCODE_BASE>[R0] ; Check for required number
                       D1  1F  0031   357          BLSSU   KINSFARG             ;  of arguments
            5E      5D  D0  0033   358          MOVL    FP,SP                ; Copy FP to SP
                    50  AF  0036   359          CASEW   R0,-                 ; Case on change mode
                         0038   360                  -                    ; argument value
                         0038   361                  #KCODE_BASE,-        ; Base value
      00    408E  8F          0038   362                  #<KERNEL_COUNTER-1>  ; Limit value (number of entries)
                         003C   363 KCASE_BASE:                           ; Case table base address for DEFINE_SERVICE
                         003C   364 ;
                         003C   365 ;       Case table entries are made in the PSECT MOUNTDSP_KERNEL_DISP1 by
                         003C   366 ;       invocations of the DEFINE_SERVICE macro.  The four PSECTS,
                         003C   367 ;       MOUNTDSP_KERNEL_DISP0,1,2,3 will be abutted in lexical order at link-time.
                         003C   368 ;
                00000000    369          .PSECT  MOUNTDSP_KERNEL_DISP2,BYTE,NOWRT,EXE,PIC
                     0000   370          BUG_CHECK IVSSRVRQST,FATAL   ; Invalid system service request
                   05  0004   371          RSB                          ; Return to reject out of
                     0005   372                                       ; range value
```

```
                                0005   374          .SBTTL  Executive Mode Dispatcher
                                0005   375 ;++
                                0005   376 ; Input Parameters:
                                0005   377 ;
                                0005   378 ;       (SP) - Return address if bad change mode value
                                0005   379 ;
                                0005   380 ;        R0  - Change mode argument value.
                                0005   381 ;
                                0005   382 ;        AP  - Argument pointer existing when the change
                                0005   383 ;              mode instruction was executed.
                                0005   384 ;
                                0005   385 ;        FP  - Address of minimal call frame to exit
                                0005   386 ;              the change mode dispatcner and return to
                                0005   387 ;              the original mode.
                                0005   388 ;--
                           00000000    389          .PSECT  MOUNTDSP_EXEC_DISP0,BYTE,NOWRT,EXE,PIC
                                0000   390 EACCVIO:                                ; Exec access violation
                     50    0C   3C     0000   391          MOVZWL  #SS$_ACCVIO,R0          ; Set access violation status code
                           04   0003   392          RET                             ;  and return
                                0004   393 EINSFARG:                               ; Exec insufficient arguments.
                50   0114 8F   3C     0004   394          MOVZWL  #SS$_INSFARG,R0         ; Set status code and
                           04   0009   395          RET                             ;  return
                           05   000A   396 ENOTME: RSB                             ; RSB to forward request
                                000B   397
                                000B   398 EXEC_DISPATCH::                          ; Entry to dispatcher
                51   BF76 C0   9E     000B   399          MOVAB   W^-ECODE_BASE(R0),R1    ; Normalize dispatch code value
                          F8   19     0010   400          BLSS    ENOTME                  ; Branch if code value too low
                     03   51   B1     0012   401          CMPW    R1,#EXEC_COUNTER        ; Check high limit
                          F3   1E     0015   402          BGEQU   ENOTME                  ; Branch if out of range
                                0017   403 ;
                                0017   404 ; The dispatch code has now been verified as being handled by this dispatcher,
                                0017   405 ; now the argument list will be probed and the required number of arguments
                                0017   406 ; verified.
                                0017   407 ;
                51    0000'CF41   9A   0017   408          MOVZBL  W^EXEC_NARG[R1],R1      ; Get required argument count
           51    00000004 9F41   DE     001D   409          MOVAL   @#4[R1],R1              ; Compute byte count including arg count
                                0025   410          IFNORD  R1,(AP),EACCVIO         ; Branch if arglist not readable
                BF76'CF40   6C   91     002B   411          CMPB    (AP),W^<EXEC_NARG-ECODE_BASE>[R0] ; Check for required number
                          D1   1F     0031   412          BLSSU   EINSFARG                ;  of arguments
                     5E   5D   D0     0033   413          MOVL    FP,SP                   ; Copy FP to SP
                          50   AF     0036   414          CASEW   R0,-                    ; Case on change mode
                                0038   415                  -                           ; argument value
                                0038   416                  #ECODE_BASE,-               ; Base value
                  02    408A 8F         0038   417                  #<EXEC_COUNTER-1>           ; Limit value (number of entries)
                                003C   418 ECASE_BASE:                               ; Case table base address for DEFINE_SERVICE
                                003C   419 ;
                                003C   420 ;       Case table entries are made in the PSECT MOUNTDSP_EXEC_DISP1 by
                                003C   421 ;       invocations of the DEFINE_SERVICE macro.  The four PSECTS,
                                003C   422 ;       MOUNTDSP_EXEC_DISP0,1,2,3 will be abutted in lexical order at link-time.
                                003C   423 ;
                           00000000    424          .PSECT  MOUNTDSP_EXEC_DISP2,BYTE,NOWRT,EXE,PIC
                                0000   425          BUG_CHECK IVSSRVRQST,FATAL      ; Invalid system service request
                           05   0004   426          RSB                             ; Return to reject out of
                                0005   427                                          ; range value
                                0005   428
                                0005   429 ;
                                0005   430 ; By this time, all EXEC and KERNEL vectors have been defined.  Perform
```

```
                                                                H 4
0005    431 ; a assembly-time check to make sure the number of vectors is reasonable.
0005    432 ;
0005    433         CHECK_VECTORS
0005    434
0005    435         .END
```

```
$CHANGE_PROT                    ********   X    04
$CHANGE_PROT$E                  0000000C R       06
$CHANGE_PROT$U                  00000018 RG      04
$COPY_INFO                      ********     X   04
$COPY_INFO$E                    00000006 R       06
$COPY_INFO$U                    00000010 RG      04
$DALLOC_DEVS                    ********     X   04
$DALLOC_DEVS$K                  00000000 R       08
$DALLOC_DEVS$U                  00000020 RG      04
BUG$_IVSSRVRQST                 ********     X   0B
EACCVIO                         00000000 R       0C
ECASE_BASE                      0000003C R       0C
ECODE_BASE                  =   0000408A
ECODE_VECTORS               =   00000004
EINSFARG                        00000004 R       0C
ENOTME                          0000000A R       0C
EXEC_COUNTER                =   00000003
EXEC_DISPATCH                   0000000B RG      0C
EXEC_NARG                       00000000 R       03
KACCVIO                         00000000 R       0A
KCASE_BASE                      0000003C R       0A
KCODE_BASE                  =   0000408E
KCODE_VECTORS               =   00000002
KERNEL_COUNTER              =   00000001
KERNEL_DISPATCH                 0000000B RG      0A
KERNEL_NARG                     00000000 R       02
KINSFARG                        00000004 R       0A
KNOTME                          0000000A R       0A
PLV$C_TYP_CMOD              =   00000001
SS$_ACCVIO                  =   0000000C
SS$_INSFARG                 =   00000114
SYS$K_VERSION                   ********     X   09
SYS$MOUNT                       ********     X   04
SYS$MOUNT$U                     00000000 RG      04
SYS$VMOUNT                      ********     X   04
SYS$VMOUNT$E                    00000000 R       06
SYS$VMOUNT$U                    00000008 RG      04
```

+----------------+
! Psect synopsis !
+----------------+

PSECT name	Allocation		PSECT No.		Attributes									
. ABS .	00000000	(0.)	00	(0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC BYTE
ABS	00000000	(0.)	01	(1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC BYTE
KERNEL_NARG	00000001	(1.)	02	(2.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
EXEC_NARG	00000003	(3.)	03	(3.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
$$$TRANSFER_VECTOR	00000027	(39.)	04	(4.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC PAGE
MOUNTDSP_EXEC_DISP1	00000006	(6.)	05	(5.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
MOUNTDSP_EXEC_DISP3	00000012	(18.)	06	(6.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
MOUNTDSP_KERNEL_DISP1	00000002	(2.)	07	(7.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
MOUNTDSP_KERNEL_DISP3	00000006	(6.)	08	(8.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
MOUNTDSP	00000020	(32.)	09	(9.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	VEC PAGE
MOUNTDSP_KERNEL_DISP0	0000003C	(60.)	0A	(10.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
MOUNTDSP_KERNEL_DISP2	00000005	(5.)	0B	(11.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE
MOUNTDSP_EXEC_DISP0	0000003C	(60.)	0C	(12.)	PIC	USR	CON	REL	LCL	NOSHR	EXE	RD	NOWRT	NOVEC BYTE

```
MOUNTDSP_EXEC_DISP2             00000005  (    5.)  0D ( 13.)    PIC   USR   CON   REL   LCL NOSHR   EXE   RD  NOWRT NOVEC BYTE
```

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	47	00:00:00.12	00:00:01.21
Command processing	158	00:00:00.74	00:00:04.56
Pass 1	227	00:00:05.07	00:00:17.27
Symbol table sort	0	00:00:00.69	00:00:01.95
Pass 2	98	00:00:01.33	00:00:03.68
Symbol table output	5	00:00:00.10	00:00:00.62
Psect synopsis output	5	00:00:00.06	00:00:00.44
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	543	00:00:08.11	00:00:29.77

```
The working set limit was 1350 pages.
26370 bytes (52 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 452 non-local and 0 local symbols.
435 source lines were read in Pass 1, producing 35 object records in Pass 2.
16 pages of virtual memory were used to define 12 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	7

```
493 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:MOUNTDSP/OBJ=OBJ$:MOUNTDSP MSRC$:MOUNTDSP/UPDATE=(ENH$:MOUNTDSP)+EXECML$/LIB
```

0245 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY