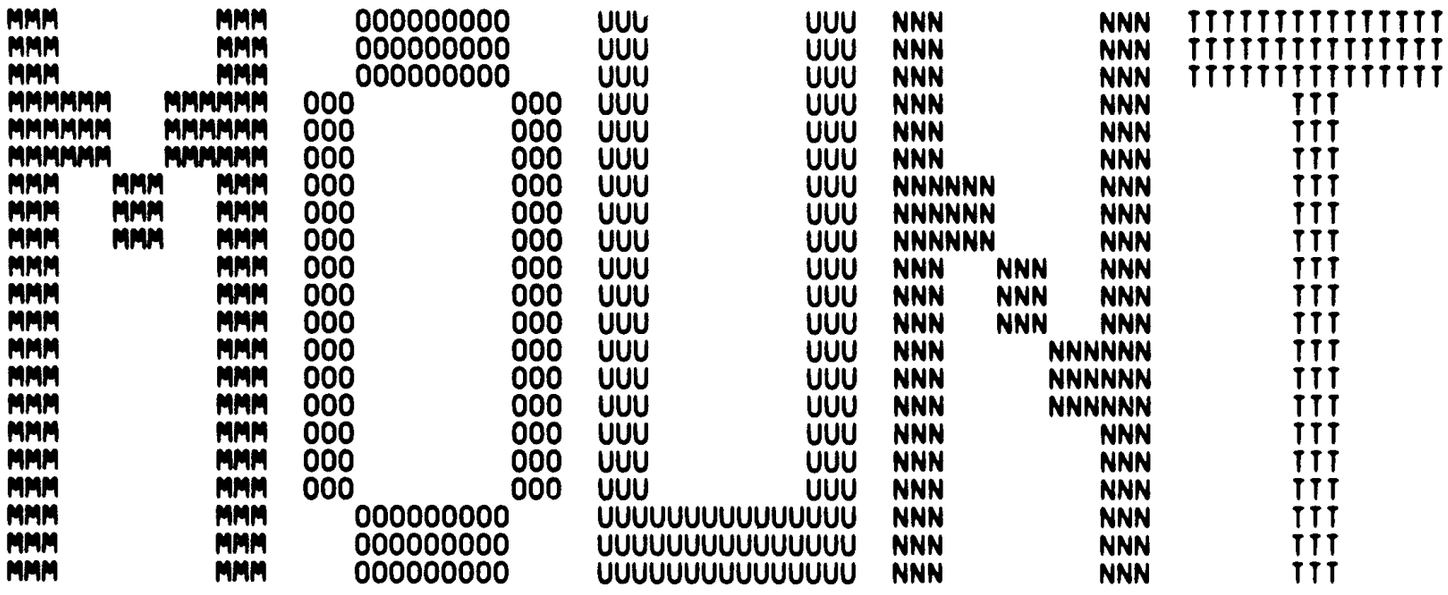
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMM MMM MMM 000 000 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**ALLOCM

```
  AAAAAA    LL          LL            000000      CCCCCCCC  MM      MM
  AAAAAA    LL          LL            000000      CCCCCCCC  MM      MM
AA      AA  LL          LL          00      00  CC          MMMM  MMMM
AA      AA  LL          LL          00      00  CC          MMMM  MMMM
AA      AA  LL          LL          00      00  CC          MM  MM  MM
AA      AA  LL          LL          00      00  CC          MM  MM  MM
AA      AA  LL          LL          00      00  CC          MM      MM
AA      AA  LL          LL          00      00  CC          MM      MM
AAAAAAAAAA  LL          LL          00      00  CC          MM      MM
AAAAAAAAAA  LL          LL          00      00  CC          MM      MM
AA      AA  LL          LL          00      00  CC          MM      MM      ....
AA      AA  LL          LL          00      00  CC          MM      MM      ....
AA      AA  LLLLLLLLLL  LLLLLLLLLL    000000      CCCCCCCC  MM      MM      ....
AA      AA  LLLLLLLLLL  LLLLLLLLLL    000000      CCCCCCCC  MM      MM      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
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
```

ASS
VOL

```
;     1        0001  0 MODULE ALLOCM (
;     2        0002  0                 LANGUAGE (BLISS32),
;     3        0003  0                 IDENT = 'V04-000'
;     4        0004  0                 ) =
;     5        0005  1 BEGIN
;     6        0006  1
;     7        0007  1 !
;     8        0008  1 !**************************************************************************
;     9        0009  1 !*                                                                        *
;    10        0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;    11        0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;    12        0012  1 !*  ALL RIGHTS RESERVED.                                                   *
;    13        0013  1 !*                                                                        *
;    14        0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    15        0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    16        0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17        0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18        0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19        0019  1 !*  TRANSFERRED.                                                           *
;    20        0020  1 !*                                                                        *
;    21        0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22        0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23        0023  1 !*  CORPORATION.                                                           *
;    24        0024  1 !*                                                                        *
;    25        0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26        0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;    27        0027  1 !*                                                                        *
;    28        0028  1 !*                                                                        *
;    29        0029  1 !**************************************************************************
;    30        0030  1
;    31        0031  1 !++
;    32        0032  1 !
;    33        0033  1 ! FACILITY:  MOUNT Utility Structure Levels 1 & 2
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !
;    37        0037  1 !       This routine allocates dynamic memory from the selected allocation pool.
;    38        0038  1 !
;    39        0039  1 ! ENVIRONMENT:
;    40        0040  1 !
;    41        0041  1 !       STARLET operating system, including privileged system services
;    42        0042  1 !       and internal exec routines.
;    43        0043  1 !
;    44        0044  1 !--
;    45        0045  1 !
;    46        0046  1 !
;    47        0047  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  19-Oct-1977 8:49
;    48        0048  1 !
;    49        0049  1 ! MODIFIED BY:
;    50        0050  1 !
;    51        0051  1 !       V03-003 HH0041          Hai Huang                   24-Jul-1984
;    52        0052  1 !               Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.B32'.
;    53        0053  1 !
;    54        0054  1 !       V03-002 CDS0001         Christian D. Saether     6-Feb-1984
;    55        0055  1 !               Have ALLOCATE_MEM store returned size as a longword
;    56        0056  1 !               so that large buffer caches may be allocated from
;    57        0057  1 !               paged pool.
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-001 RAS0180         Ron Schaefer            4-Aug-1983
;    60          0060  1 !               Fix broken calls caused by changes in DISMNTSHR.
;    61          0061  1 !
;    62          0062  1 !       V02-001 ACG0175         Andrew C. Goldstein,    4-Jun-1980  20:16
;    63          0063  1 !               Release I/O database lock before error exit
;    64          0064  1 !
;    65          0065  1 !       V02-000 ACG0167         Andrew C. Goldstein,    18-Apr-1980  13:37
;    66          0066  1 !               Previous revision history moved to MOUNT.REV
;    67          0067  1 !**
;    68          0068  1
;    69          0069  1
;    70          0070  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    71          0071  1 REQUIRE 'SRC$:MOUDEF.B32';
```

```
;    73          0603  1 GLOBAL ROUTINE ALLOCATE_MEM (SIZE_NEEDED, AREA) =
;    74          0604  1
;    75          0605  1 !++
;    76          0606  1 !
;    77          0607  1 ! FUNCTIONAL DESCRIPTION:
;    78          0608  1 !
;    79          0609  1 !       This routine allocates dynamic memory from the selected allocation pool.
;    80          0610  1 !
;    81          0611  1 !
;    82          0612  1 ! CALLING SEQUENCE:
;    83          0613  1 !       ALLOCATE_MEM (ARG1, ARG2)
;    84          0614  1 !
;    85          0615  1 ! INPUT PARAMETERS:
;    86          0616  1 !       ARG1: size needed in bytes
;    87          0617  1 !       ARG2: code of area to allocate from:
;    88          0618  1 !               0 = non-paged system pool
;    89          0619  1 !               1 = paged system pool
;    90          0620  1 !               2 = process allocation region
;    91          0621  1 !
;    92          0622  1 ! IMPLICIT INPUTS:
;    93          0623  1 !       NONE
;    94          0624  1 !
;    95          0625  1 ! OUTPUT PARAMETERS:
;    96          0626  1 !       NONE
;    97          0627  1 !
;    98          0628  1 ! IMPLICIT OUTPUTS:
;    99          0629  1 !       NONE
;   100          0630  1 !
;   101          0631  1 ! ROUTINE VALUE:
;   102          0632  1 !       address of block allocated
;   103          0633  1 !
;   104          0634  1 ! SIDE EFFECTS:
;   105          0635  1 !       memory allocated, zeroed, and full longword size is stored
;   106          0636  1 !       at size word location.
;   107          0637  1 !
;   108          0638  1 !--
;   109          0639  1
;   110          0640  2 BEGIN
;   111          0641  2
;   112          0642  2 BUILTIN
;   113          0643  2         TESTBITSC;
;   114          0644  2
;   115          0645  2 LINKAGE
;   116          0646  2         EXE_ALLOC0      = JSB :
;   117          0647  2                         NOPRESERVE (3, 4, 5)
;   118          0648  2                         GLOBAL (SIZE = 1, ADDRESS = 2),
;   119          0649  2
;   120          0650  2         EXE_ALLOC1      = JSB (REGISTER = 3) :
;   121          0651  2                         NOPRESERVE (4, 5)
;   122          0652  2                         GLOBAL (SIZE = 1, ADDRESS = 2);
;   123          0653  2
;   124          0654  2 LOCAL
;   125          0655  2         STATUS,                         ! status return of allocator
;   126          0656  2         BLOCK_SIZE,                     ! local copy of size allocated
;   127          0657  2         BLOCK_ADDRESS;                  ! local copy of address
;   128          0658  2
;   129          0659  2 EXTERNAL
```

```
;  130          0660  2          CLEANUP_FLAGS   : BITVECTOR,     ! cleanup action flags
;  131          0661  2          CTL$GQ_ALLOCREG : ADDRESSING_MODE (ABSOLUTE);
;  132          0662  2                                          ! process allocation region listhead
;  133          0663  2
;  134          0664  2  EXTERNAL ROUTINE
;  135          0665  2          UNLOCK_IODB     : ADDRESSING_MODE (LONG_RELATIVE);
;  136          0666  2                                          ! unlock I/O data base
;  137          0667  2
;  138          0668  3  BEGIN                                   ! nested block to avoid scope conflicts
;  139          0669  3
;  140          0670  3  GLOBAL REGISTER
;  141          0671  3          SIZE            = 1,            ! rounded up allocation size
;  142          0672  3          ADDRESS         = 2 : REF VECTOR [,WORD]; ! address returned by exec routines
;  143          0673  3
;  144          0674  3  EXTERNAL ROUTINE
;  145          0675  3          EXE$ALONONPAGED : EXE_ALLOC0 ADDRESSING_MODE (ABSOLUTE),
;  146          0676  3          EXE$ALOPAGED    : EXE_ALLOC0 ADDRESSING_MODE (ABSOLUTE),
;  147          0677  3          EXE$ALLOCATE    : EXE_ALLOC1 ADDRESSING_MODE (ABSOLUTE);
;  148          0678  3
;  149          0679  3
;  150          0680  3  ! Simply compute the size needed rounded up to the next quadword and call the
;  151          0681  3  ! appropriate exec allocation routine.
;  152          0682  3  !
;  153          0683  3
;  154          0684  3  SIZE = .SIZE_NEEDED;
;  155          0685  3
;  156          0686  4  STATUS = (
;  157          0687  4  CASE .AREA FROM 0 TO 2 OF
;  158          0688  4  SET
;  159          0689  4      [0]:        EXE$ALONONPAGED ();
;  160          0690  4      [1]:        EXE$ALOPAGED ();
;  161          0691  4      [2]:        EXE$ALLOCATE (CTL$GQ_ALLOCREG);
;  162          0692  4  TES
;  163          0693  3  );
;  164          0694  3
;  165          0695  3  ! Copy the block size and address into locals to dodge the MOVC5.
;  166          0696  3  !
;  167          0697  3
;  168          0698  3  BLOCK_SIZE = .SIZE;
;  169          0699  3  BLOCK_ADDRESS = .ADDRESS;
;  170          0700  2  END;
;  171          0701  2
;  172          0702  2  IF NOT .STATUS
;  173          0703  2  THEN
;  174          0704  3      BEGIN
;  175          0705  3      IF TESTBITSC (CLEANUP_FLAGS[CLF_UNLOCKDB])
;  176          0706  3      THEN UNLOCK_IODB ();
;  177          0707  3      ERR_EXIT (SS$_INSFMEM);
;  178          0708  2      END;
;  179          0709  2
;  180          0710  2  CH$FILL (0, .BLOCK_SIZE, .BLOCK_ADDRESS);
;  181          0711  2  (.BLOCK_ADDRESS + 8) = .BLOCK_SIZE;
;  182          0712  2  RETURN .BLOCK_ADDRESS;
;  183          0713  2
;  184          0714  1  END;                                   ! end of routine ALLOCATE_MEM
```

```
                                                        .TITLE  ALLOCM
                                                        .IDENT  \V04-000\

                                                        .EXTRN  CLEANUP_FLAGS, CTL$GQ_ALLOCREG
                                                        .EXTRN  UNLOCK_IODB, EXE$ALONONPAGED
                                                        .EXTRN  EXE$ALOPAGED, EXE$ALLOCATE

                                                        .PSECT  $CODE$,NOWRT,2

                                        0FFC 00000      .ENTRY  ALLOCATE_MEM, Save R2,R3,R4,R5,R6,R7,R8,R9,-      ; 0603
                                                                R10,R11
                                  51    04   AC  D0 00002      MOVL    SIZE_NEEDED, SIZE                          ; 0684
                            02    00    08   AC  CF 00006      CASEL   AREA, #0, #2                               ; 0687
                          0016          000E       0006       .WORD   2$-1$,-
                                                      0008 1$:          3$-1$,-
                                                                        4$-1$
                                00000000G 9F  16 00011 2$:      JSB     @#EXE$ALONONPAGED                          ; 0689
                                          15  11 00017          BRB     5$
                                00000000G 9F  16 00019 3$:      JSB     @#EXE$ALOPAGED                             ; 0690
                                          0D  11 0001F          BRB     5$
                                  53 00000000G 8F  D0 00021 4$: MOVL    #CTL$GQ_ALLOCREG, R3                       ; 0691
                                00000000G 9F  16 00028          JSB     @#EXE$ALLOCATE
                                  57          51  D0 0002E 5$: MOVL    SIZE, BLOCK_SIZE                           ; 0698
                                  56          52  D0 00031     MOVL    ADDRESS, BLOCK_ADDRESS                     ; 0699
                                  19          50  E8 00034     BLBS    STATUS, 7$                                 ; 0702
                            07      0000G     CF  0B E5 00037 BBCC    #11, CLEANUP_FLAGS, 6$                     ; 0705
                                00000000G     EF  00 FB 0003D CALLS   #0, UNLOCK_IODB                            ; 0706
                                  7E      0124  8F  3C 00044 6$: MOVZWL #292, -(SP)                               ; 0707
                                00000000G     00  01 FB 00049 CALLS   #1, LIB$STOP
                    57          00            6E  00 2C 00050 7$: MOVC5   #0, (SP), #0, BLOCK_SIZE, (BLOCK_ADDRESS) ; 0710
                                                  66     00055
                                        08    A6  57 D0 00056     MOVL    BLOCK_SIZE, 8(BLOCK_ADDRESS)           ; 0711
                                        50        56 D0 0005A     MOVL    BLOCK_ADDRESS, R0                      ; 0712
                                                  04 0005D        RET                                            ; 0714

; Routine Size:  94 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  186     0715  1 GLOBAL ROUTINE DEALLOCATE_MEM (ADDRESS, AREA) : NOVALUE =
;  187     0716  1
;  188     0717  1 !++
;  189     0718  1 !
;  190     0719  1 ! FUNCTIONAL DESCRIPTION:
;  191     0720  1 !
;  192     0721  1 !       This routine deallocates dynamic memory to the selected allocation pool.
;  193     0722  1 !
;  194     0723  1 !
;  195     0724  1 ! CALLING SEQUENCE:
;  196     0725  1 !       DEALLOCATE_MEM (ARG1, ARG2)
;  197     0726  1 !
;  198     0727  1 ! INPUT PARAMETERS:
;  199     0728  1 !       ARG1: address of block to deallocate
;  200     0729  1 !       ARG2: code of area to deallocate to:
;  201     0730  1 !               0 = non-paged system pool
;  202     0731  1 !               1 = paged system pool
;  203     0732  1 !               2 = process allocation region
;  204     0733  1 !
;  205     0734  1 ! IMPLICIT INPUTS:
;  206     0735  1 !       NONE
;  207     0736  1 !
;  208     0737  1 ! OUTPUT PARAMETERS:
;  209     0738  1 !       NONE
;  210     0739  1 !
;  211     0740  1 ! IMPLICIT OUTPUTS:
;  212     0741  1 !       NONE
;  213     0742  1 !
;  214     0743  1 ! ROUTINE VALUE:
;  215     0744  1 !       NONE
;  216     0745  1 !
;  217     0746  1 ! SIDE EFFECTS:
;  218     0747  1 !       memory deallocated
;  219     0748  1 !
;  220     0749  1 !--
;  221     0750  1
;  222     0751  2 BEGIN
;  223     0752  2
;  224     0753  2 MAP
;  225     0754  2         ADDRESS         : REF VECTOR [,WORD];
;  226     0755  2
;  227     0756  2 LINKAGE
;  228     0757  2         EXE_DEALLOC0    = JSB (REGISTER = 0) :
;  229     0758  2                         NOPRESERVE (2, 3, 4, 5),
;  230     0759  2
;  231     0760  2         EXE_DEALLOC1    = JSB (REGISTER = 0, REGISTER = 1, REGISTER = 3) :
;  232     0761  2                         NOPRESERVE (2, 4, 5);
;  233     0762  2
;  234     0763  2 EXTERNAL
;  235     0764  2         CTL$GQ_ALLOCREG : ADDRESSING_MODE (ABSOLUTE);
;  236     0765  2                                           ! process allocation region listhead
;  237     0766  2
;  238     0767  2 EXTERNAL ROUTINE
;  239     0768  2         EXE$DEANONPAGED : EXE_DEALLOC0 ADDRESSING_MODE (ABSOLUTE),
;  240     0769  2         EXE$DEAPAGED    : EXE_DEALLOC0 ADDRESSING_MODE (ABSOLUTE),
;  241     0770  2         EXE$DEALLOCATE  : EXE_DEALLOC1 ADDRESSING_MODE (ABSOLUTE);
;  242     0771  2
```

```
;  243          0772  2
;  244          0773  2 ! Just return the block to the appropriate area.
;  245          0774  2 !
;  246          0775  2
;  247          0776  2 CASE .AREA FROM 0 TO 2 OF
;  248          0777  2 SET
;  249          0778  2     [0]:        EXE$DEANONPAGED (ADDRESS[0]);
;  250          0779  2     [1]:        EXE$DEAPAGED (ADDRESS[0]);
;  251          0780  2     [2]:        EXE$DEALLOCATE (ADDRESS[0], .ADDRESS[4], CTL$GQ_ALLOCREG);
;  252          0781  2 TES;
;  253          0782  1 END;                                   ! end of routine DEALLOCATE_MEM


                                                        .EXTRN  EXE$DEANONPAGED
                                                        .EXTRN  EXE$DEAPAGED, EXE$DEALLOCATE

                                    0FFC 00000          .ENTRY  DEALLOCATE_MEM, Save R2,R3,R4,R5,R6,R7,R8,- ; 0715
                                                                R9,R10,R11
                          56    04   AC  D0 00002        MOVL    ADDRESS, R6                                 ; 0778
                    02    00    08   AC  CF 00006        CASEL   AREA, #0, #2                                ; 0776
                  001A          0010      0006    0000B 1$:     .WORD   2$-1$,-
                                                                        3$-1$,-
                                                                        4$-1$
                          50              56  D0 00011 2$:      MOVL    R6, R0                              ; 0778
                             00000000G    9F  16 00014         JSB     @#EXE$DEANONPAGED
                                              04 0001A         RET
                          50              56  D0 0001B 3$:      MOVL    R6, R0                              ; 0779
                             00000000G    9F  16 0001E         JSB     @#EXE$DEAPAGED
                                              04 00024         RET
                          53 00000000G    8F  D0 00025 4$:      MOVL    #CTL$GQ_ALLOCREG, R3                ; 0780
                          51        08    A6  3C 0002C         MOVZWL  8(R6), R1
                          50              56  D0 00030         MOVL    R6, R0
                             00000000G    9F  16 00033         JSB     @#EXE$DEALLOCATE
                                              04 00039         RET                                         ; 0782

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 005E

;  254          0783  1
;  255          0784  1 END
;  256          0785  0 ELUDOM


                                                        .EXTRN  LIB$STOP

;                                  PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  $CODE$                        152  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;                               Library Statistics
;
;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1         18619         9         0      1000          00:01.9
;
;
;
;
;                                       COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:ALLOCM/OBJ=OBJ$:ALLOCM MSRC$:ALLOCM/UPDATE=(ENH$:ALLOCM)
;
; Size:         152 code + 0 data bytes
; Run Time:       00:11.6
; Elapsed Time:   00:24.4
; Lines/CPU Min:   4067
; Lexemes/CPU-Min: 33528
; Memory Used:  91 pages
; Compilation Complete
```

BINDVL
LIS

MOUNT

SYSFAO
LIS

MOUNTSHR
MAP

TEMPLATE
LIS

VMOUNT
MAP

ASSIST
LIS

ALLOCM
LIS

MOUDEF
B32