```
DDDDDDDDDDDD        IIIIIIIII      SSSSSSSSSSS  MMM         MMM    OOOOOOOOO     UUU         UUU
DDDDDDDDDDDD        IIIIIIIII      SSSSSSSSSSS  MMM         MMM    OOOOOOOOO     UUU         UUU
DDDDDDDDDDDD        IIIIIIIII      SSSSSSSSSSS  MMM         MMM    OOOOOOOOO     UUU         UUU
DDD          DDD       III       SSS           MMMMMM   MMMMMM  OOO       OOO   UUU         UUU
DDD          DDD       III       SSS           MMMMMM   MMMMMM  OOO       OOO   UUU         UUU
DDD          DDD       III       SSS           MMMMMM   MMMMMM  OOO       OOO   UUU         UUU
DDD          DDD       III       SSS           MMM  MMM  MMM  OOO       OOO   UUU         UUU
DDD          DDD       III       SSS           MMM  MMM  MMM  OOO       OOO   UUU         UUU
DDD          DDD       III       SSS           MMM  MMM  MMM  OOO       OOO   UUU         UUU
DDD          DDD       III        SSSSSSSSS    MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III        SSSSSSSSS    MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III        SSSSSSSSS    MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III                SSS   MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III                SSS   MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III                SSS   MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III                SSS   MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III                SSS   MMM       MMM  OOO       OOO   UUU         UUU
DDD          DDD       III                SSS   MMM       MMM  OOO       OOO   UUU         UUU
DDDDDDDDDDDD        IIIIIIIII      SSSSSSSSSSS  MMM       MMM     OOOOOOOOO     UUUUUUUUUUUUUUU
DDDDDDDDDDDD        IIIIIIIII      SSSSSSSSSSS  MMM       MMM     OOOOOOOOO     UUUUUUUUUUUUUUU
DDDDDDDDDDDD        IIIIIIIII      SSSSSSSSSSS  MMM       MMM     OOOOOOOOO     UUUUUUUUUUUUUUU
```

FILEID**DISMOU

```
DDDDDDDD      IIIIII      SSSSSSSS  MM      MM    OOOOOO    UU      UU
DDDDDDDD      IIIIII      SSSSSSSS  MM      MM    OOOOOO    UU      UU
DD      DD      II      SS          MMMM  MMMM  OO      OO  UU      UU
DD      DD      II      SS          MMMM  MMMM  OO      OO  UU      UU
DD      DD      II      SS          MM  MM  MM  OO      OO  UU      UU
DD      DD      II      SS          MM  MM  MM  OO      OO  UU      UU
DD      DD      II        SSSSSS    MM      MM  OO      OO  UU      UU
DD      DD      II        SSSSSS    MM      MM  OO      OO  UU      UU
DD      DD      II              SS  MM      MM  OO      OO  UU      UU
DD      DD      II              SS  MM      MM  OO      OO  UU      UU
DD      DD      II              SS  MM      MM  OO      OO  UU      UU
DD      DD      II              SS  MM      MM  OO      OO  UU      UU
DDDDDDDD      IIIIII    SSSSSSSS    MM      MM    OOOOOO    UUUUUUUUUU
DDDDDDDD      IIIIII    SSSSSSSS    MM      MM    OOOOOO    UUUUUUUUUU

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
```

```
;  1     0001  0 MODULE DISMOU (
;  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:  DISMOUNT Utility Structure Level 1
; 34     0034  1 !
; 35     0035  1 ! ABSTRACT:
; 36     0036  1 !
; 37     0037  1 !       This is the main routine of the DISMOUNT command.
; 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:  28-Oct-1977  14:12
; 48     0048  1 !
; 49     0049  1 ! MODIFIED BY:
; 50     0050  1 !
; 51     0051  1 !       V03-016 HH0035          Hai Huang               10-Jul-1984
; 52     0052  1 !               Fix truncation errors.
; 53     0053  1 !
; 54     0054  1 !       V03-015 HH0027          Hai Huang               26-Jun-1984
; 55     0055  1 !               Prevent race condition between two simultaneous dismounts
; 56     0056  1 !               on the same volume.
; 57     0057  1 !
```

```
;    58          0058  1 ;        V03-014 MHB0154         Mark Bramhall           27-Apr-1984
;    59          0059  1 ;                Correct NSA$B_ARG_FLAG setting for multiple audits enabled.
;    60          0060  1 ;
;    61          0061  1 ;        V03-013 LMP0229         L. Mark Pilant,         12-Apr-1984  12:24
;    62          0062  1 ;                Remove references to the CHIP block.
;    63          0063  1 ;
;    64          0064  1 ;        V03-012 HH0014          Hai Huang               10-Apr-1984
;    65          0065  1 ;                Synchronize $GETDVIW on MOUNT_EFN.
;    66          0066  1 ;
;    67          0067  1 ;        V03-011 HH0013          Hai Huang               09-Apr-1984
;    68          0068  1 ;                Use LNM$C_MAXDEPTH to represent maximum number of times to
;    69          0069  1 ;                recursively translate a logical name.
;    70          0070  1 ;
;    71          0071  1 ;        V03-010 HH0009          Hai Huang               28-Mar-1984
;    72          0072  1 ;                Add security auditing support.
;    73          0073  1 ;
;    74          0074  1 ;        V03-009 LMP0221         L. Mark Pilant,         26-Mar-1984  16:27
;    75          0075  1 ;                Change the device owner location to the ORB from the UCB.
;    76          0076  1 ;
;    77          0077  1 ;        V03-008 HH0007          Hai Huang               22-Mar-1984
;    78          0078  1 ;                Add cluster-wide group volume support.
;    79          0079  1 ;
;    80          0080  1 ;        V03-007 HH0004          Hai Huang               28-Feb-1984
;    81          0081  1 ;                Add cluster-wide mount support.
;    82          0082  1 ;
;    83          0083  1 ;        V03-006 HH0006          Hai Huang               05-Mar-1984
;    84          0084  1 ;                Fix bug introduced by HH0003 when dismounting a
;    85          0085  1 ;                foreign magtape.
;    86          0086  1 ;
;    87          0087  1 ;        V03-005 HH0003          Hai Huang               07-Feb-1984
;    88          0088  1 ;                Add forced dismount support.
;    89          0089  1 ;
;    90          0090  1 ;        V03-004 HH0002          Hai Huang               23-Jan-1984
;    91          0091  1 ;                Add job-wide mount support.
;    92          0092  1 ;
;    93          0093  1 ;        V03-003 RAS0168         Ron Schaefer            12-Jul-1983
;    94          0094  1 ;                Interlock the logical name mutex when interogating
;    95          0095  1 ;                MTL$L_LOGNAME.
;    96          0096  1 ;
;    97          0097  1 ;        V03-002 DMW4051         DMWalp                  20-Jun-1983
;    98          0098  1 ;                Intergration of new logical name structures
;    99          0099  1 ;
;   100          0100  1 ;        V03-001 STJ0240         Steven T. Jeffreys,     23-Mar-1982
;   101          0101  1 ;                Use system routines to check descriptors.
;   102          0102  1 ;
;   103          0103  1 ;        V02-010 STJ0231         Steven T. Jeffreys,     02-Mar-1982
;   104          0104  1 ;                Copy buffer descriptor to internal storage before probing.
;   105          0105  1 ;
;   106          0106  1 ;        V02-009 STJ0227         Steven T. Jeffreys,     17-Feb-1982
;   107          0107  1 ;                Fix incorrect probe of the user-specified device name.
;   108          0108  1 ;                Also fix typos in update packet.
;   109          0109  1 ;
;   110          0110  1 ;        V02-008 STJ0176         Steven T. Jeffreys,     07-Jan-1981
;   111          0111  1 ;                Set BUGCHECK and EXQUOTA privileges for the user, and
;   112          0112  1 ;                clear them when we are done with them.
;   113          0113  1 ;
;   114          0114  1 ;        V02-007 ACG0248         Andrew C. Goldstein,    31-Dec-1981  13:14
```

```
;   115          0115  1 !               Interlock mounted volume list with I/O database mutex
;   116          0116  1 !
;   117          0117  1 !       V02-006 STJ0075         Steven T. Jeffreys      24-Jul-1981
;   118          0118  1 !               Liberal rewrite to convert the existing dismount code
;   119          0119  1 !               to a system service.
;   120          0120  1 !
;   121          0121  1 !       V02-005 PCG0001         Peter C. George         03-Feb-1981 10:00
;   122          0122  1 !               Change MOUNTMSG require to DISMOUMSG.
;   123          0123  1 !
;   124          0124  1 !       V02-004 ACG0181         Andrew C. Goldstein,    9-Oct-1980  15:59
;   125          0125  1 !               Fix cross facility source reference
;   126          0126  1 !
;   127          0127  1 !       X0103   ACG0072         Andrew C. Goldstein,    15-Oct-1979  16:21
;   128          0128  1 !               Check primary and secondary device characteristics
;   129          0129  1 !
;   130          0130  1 !       X0102   ACG0025         Andrew C. Goldstein,  6-Mar-1979  21:03
;   131          0131  1 !       Fix magtape testing code
;   132          0132  1 !
;   133          0133  1 !       X0101   ACG0003         Andrew C. Goldstein,  10-Jan-1979  20:02
;   134          0134  1 !       Add multi-volume disk support
;   135          0135  1 !
;   136          0136  1 !       X0100   ACG0001         Andrew C. Goldstein,  24-Oct-1978  13:47
;   137          0137  1 !       Previous revision history moved to [DISMOU.SRC]DISMOUNT.REV
;   138          0138  1 !**
;   139          0139  1
;   140          0140  1
;   141          0141  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;   142          0142  1 REQUIRE 'LIB$:MOUDEF.B32';
;   143          0674  1 REQUIRE 'LIBD$:[VMSLIB.OBJ]DISMOUMSG.B32';
;   144          0751  1
;   145          0752  1
;   146          0753  1 FORWARD ROUTINE
;   147          0754  1         SYS$DISMOU,                             ! main program
;   148          0755  1         MAKE_DISMOUNT,                          ! kernel mode routine
;   149          0756  1         TRAN_LOGNAME,                           ! recursive logical name translator
;   150          0757  1         SEARCH_MOUNT,                           ! find MTL entry
;   151          0758  1         SETUP_MTL,                              ! set up local MTL database
;   152          0759  1         MOVE_MTL,                               ! set up MTL database for a volume set
;   153          0760  1         FIND_MTL,                               ! set up MTL database for a volume
;   154          0761  1         CHECK_PRIV,                             ! privilege check routine
;   155          0762  1         DISMOUNT_CLUSTER,                       ! cluster-wide dismount routine
;   156          0763  1         DISMOUNT_ENCIPHER,                      ! create a cluster-dismount packet
;   157          0764  1         DISMOUNT_AUDIT  : NOVALUE,              ! security auditing
;   158          0765  1         LABEL_LENGTH;                           ! return the length of a label
;   159          0766  1
;   160          0767  1 GLOBAL
;   161          0768  1         CLUSTER_DEVICE;                         ! global area to hold cluster device
;   162          0769  1                                                 ! characterstic bit
;   163          0770  1
;   164          0771  1
;   165          0772  1 ! Define the CODE psect so that the generated code has PIC and SHR attributes.
;   166          0773  1 !
;   167          0774  1
;   168          0775  1 PSECT   CODE = Z$DISMOUNT (PIC,SHARE);
```

```
;  170     0776  1 GLOBAL ROUTINE SYS$DISMOU (DEVNAM, FLAGS) =
;  171     0777  1
;  172     0778  1 !++
;  173     0779  1 !
;  174     0780  1 ! FUNCTIONAL DESCRIPTION:
;  175     0781  1 !
;  176     0782  1 !       This is the main routine of the DISMOUNT command.
;  177     0783  1 !
;  178     0784  1 ! INPUT PARAMETERS:
;  179     0785  1 !       DEVNAM  : Address of a device name descriptor.
;  180     0786  1 !       FLAGS   : A longword bit mask.
;  181     0787  1 !
;  182     0788  1 ! IMPLICIT INPUTS:
;  183     0789  1 !       NONE
;  184     0790  1 !
;  185     0791  1 ! OUTPUT PARAMETERS:
;  186     0792  1 !       NONE
;  187     0793  1 !
;  188     0794  1 ! IMPLICIT OUTPUTS:
;  189     0795  1 !       NONE
;  190     0796  1 !
;  191     0797  1 ! ROUTINE VALUE:
;  192     0798  1 !       assorted status values
;  193     0799  1 !
;  194     0800  1 ! SIDE EFFECTS:
;  195     0801  1 !       volume(s) dismounted, device data base updated
;  196     0802  1 !
;  197     0803  1 !--
;  198     0804  1
;  199     0805  2 BEGIN
;  200     0806  2
;  201     0807  2 !
;  202     0808  2 ! Allocate plits in the Z$DISMOUNT psect to avoid truncation error when
;  203     0809  2 ! linking mountshr.
;  204     0810  2 !
;  205     0811  2 PSECT
;  206     0812  2         PLIT = Z$DISMOUNT;
;  207     0813  2
;  208     0814  2 LINKAGE
;  209     0815  2         L_PDESC         = JSB (REGISTER=1; REGISTER=1, REGISTER=2):
;  210     0816  2                                 NOPRESERVE (3)
;  211     0817  2                                 NOTUSED (4,5,6,7,8,9,10,11);
;  212     0818  2
;  213     0819  2 EXTERNAL ROUTINE
;  214     0820  2         EXE$PROBER_DSC  : L_PDESC ADDRESSING_MODE (GENERAL);
;  215     0821  2
;  216     0822  2 LOCAL
;  217     0823  2         LENGTH          : LONG,         ! Output from EXE$PROBER_DSC
;  218     0824  2         ADDRESS         : LONG,         ! Output from EXE$PROBER_DSC
;  219     0825  2         DEV_NAME        : BBLOCK [DSC$K_S_BLN],
;  220     0826  2         USER_PRIVS      : BBLOCK [8],   ! storge for initial user privs
;  221     0827  2         DISMOUNT_PRIVS  : BBLOCK [8],   ! privileges needed for $DISMOU
;  222     0828  2         CHANNEL         : LONG,         ! channel number for I/O
;  223     0829  2         PHYS_NAME       : BBLOCK [DSC$K_S_BLN],
;  224     0830  2                                         ! descriptor of physical device name
;  225     0831  2         NAME_BUFFER     : VECTOR [NAMEBUF_LEN, BYTE],
;  226     0832  2                                         ! string buffer for physical device name
```

```
;  227     0833  2          STATUS,                             ! system service status
;  228     0834  2          LOCK_STATUS     : VECTOR [2, LONG],
;  229     0835  2                                              ! lock status block
;  230     0836  2          DMTLCKNAM_BUF   : VECTOR [NAMEBUF_LEN, BYTE]
;  231     0837  2                            INITIAL (BYTE ('DMT$', REP NAMEBUF_LEN-4 OF (' '))),
;  232     0838  2                                              ! DMT resource name buffer
;  233     0839  2          DMTLCKNAM_DSC   : VECTOR [2, LONG]
;  234     0840  2                            INITIAL (0, DMTLCKNAM_BUF),
;  235     0841  2                                              ! DMT resource name descriptor
;  236     0842  2          ITMLST          : BBLOCK [(1+12) + 4] INITIAL
;  237     0843  2                            (
;  238     0844  2                            ! Item: Allocation device name
;  239     0845  2                            !
;  240     0846  2                            (WORD (NAMEBUF_LEN-4),
;  241     0847  2                             WORD (DVI$_ALLDEVNAM),
;  242     0848  2                             LONG (DMTLCKNAM_BUF+4),
;  243     0849  2                             LONG (DMTLCKNAM_DSC),
;  244     0850  2                            !
;  245     0851  2                            ! Item list stopper
;  246     0852  2                            !
;  247     0853  2                             LONG (0));
;  248     0854  2
;  249     0855  2
;  250     0856  2  ! Probe the device descriptor and the string it describes for read access.
;  251     0857  2  ! The string descriptor is copied to DEV_NAME for future reference.
;  252     0858  2  !
;  253     0859  3  IF NOT (STATUS = EXE$PROBER_DSC (.DEVNAM; LENGTH, ADDRESS))
;  254     0860  2  THEN
;  255     0861  2      RETURN (.STATUS);
;  256     0862  2  DEV_NAME [DSC$W_LENGTH] = .LENGTH;
;  257     0863  2  DEV_NAME [DSC$B_DTYPE]  = 0;
;  258     0864  2  DEV_NAME [DSC$B_CLASS]  = 0;
;  259     0865  2  DEV_NAME [DSC$A_POINTER]= .ADDRESS;
;  260     0866  2
;  261     0867  2  ! Set up the physical device name descriptor.
;  262     0868  2  !
;  263     0869  2
;  264     0870  2  PHYS_NAME[DSC$B_CLASS]          = 0;    ! set up physical device name descriptor
;  265     0871  2  PHYS_NAME[DSC$B_DTYPE]          = 0;
;  266     0872  2  PHYS_NAME[DSC$W_LENGTH]         = NAMEBUF_LEN;
;  267     0873  2  PHYS_NAME[DSC$A_POINTER]        = NAME_BUFFER;
;  268     0874  2
;  269     0875  2  ! Translate the logical name and then assign a channel to the device.
;  270     0876  2  ! The channel is needed for two reasons; first, the device UCB address
;  271     0877  2  ! is needed, and it can easily be gotten once a channel has been assigned
;  272     0878  2  ! to the device, and second, having a channel assigned to the device will
;  273     0879  2  ! act as an interlock, and will prevent premature deallocation of the VCB.
;  274     0880  2  !
;  275     0881  2
;  276     0882  2  CHANNEL = 0;
;  277     0883  3  IF NOT (STATUS = TRAN_LOGNAME (DEV_NAME, PHYS_NAME[DSC$W_LENGTH]))
;  278     0884  2  THEN
;  279     0885  2      RETURN .STATUS;
;  280     0886  3  IF NOT (STATUS = $ASSIGN (CHAN = CHANNEL, DEVNAM = PHYS_NAME[DSC$W_LENGTH]))
;  281     0887  2  THEN
;  282     0888  2      RETURN .STATUS;
;  283     0889  2
```

```
;  284          0890  2 ! Give the user the necessary privileges and dismount the volume.
;  285          0891  2 !
;  286          0892  2
;  287          0893  2 DISMOUNT_PRIVS = 0;
;  288          0894  2 DISMOUNT_PRIVS+4 = 0;
;  289          0895  2 DISMOUNT_PRIVS [PRV$V_BUGCHK]  = 1;              ! Grant BUGCHECK privilege
;  290          0896  2 DISMOUNT_PRIVS [PRV$V_EXQUOTA] = 1;              ! Grant EXQUOTA privilege
;  291          0897  2 $SETPRV (ENBFLG=1, PRVADR=DISMOUNT_PRIVS, PRVPRV=USER_PRIVS);
;  292          0898  2
;  293          0899  2 !
;  294          0900  2 ! Take out the DMT$ interlock on this device to prevent race condition
;  295          0901  2 ! between simultaneous dismounts on the same volume.
;  296          0902  2 !
;  297          0903  2
;  298        P 0904  2 $GETDVIW (    CHAN = .CHANNEL,                    ! Get the full device name
;  299        P 0905  2               ITMLST = ITMLST,
;  300          0906  2                  EFN = MOUNT_EFN );
;  301          0907  2
;  302          0908  2 DMTLCKNAM_DSC [0] = .DMTLCKNAM_DSC [0] + 4;      ! Add in "DMT$" prefix
;  303          0909  2
;  304        P 0910  2 $ENQW    (  LKMODE = LCK$K_EXMODE,                ! Take out the DMT$ interlock
;  305        P 0911  2                LKSB = LOCK_STATUS,
;  306        P 0912  2               FLAGS = LCK$M_SYSTEM,
;  307        P 0913  2              RESNAM = DMTLCKNAM_DSC,
;  308          0914  2                  EFN = MOUNT_EFN  );
;  309          0915  2
;  310          0916  2 !
;  311          0917  2 ! Go dismount the volume.
;  312          0918  2 !
;  313          0919  2
;  314          0920  2 STATUS = MAKE_DISMOUNT (.FLAGS, .CHANNEL);
;  315          0921  2
;  316          0922  2 !
;  317          0923  2 ! Dequeue the DMT interlock.
;  318          0924  2 !
;  319          0925  2
;  320          0926  3 IF ( .LOCK_STATUS [1] NEQ 0 )                    ! Release the DMT$ interlock
;  321          0927  2 THEN
;  322          0928  2     $DEQ ( LKID = .LOCK_STATUS [1] );
;  323          0929  2
;  324          0930  2 !
;  325          0931  2 ! If the dismount was successful, send this dismount request cluster-wide
;  326          0932  2 ! when appropriate.
;  327          0933  2 !
;  328          0934  2
;  329          0935  2 IF .STATUS THEN
;  330          0936  2     STATUS = DISMOUNT_CLUSTER (PHYS_NAME, .FLAGS); ! Do cluster-wide dismount
;  331          0937  2                                                  ! with the physical device name
;  332          0938  2
;  333          0939  2 ! Revoke whatever special privileges were
;  334          0940  2 ! granted, deassign the channel, and exit.
;  335          0941  2 !
;  336          0942  2
;  337          0943  2 $SETPRV (ENBFLG=0, PRVADR=DISMOUNT_PRIVS);       ! Revoke granted privileges
;  338          0944  2 $SETPRV (ENBFLG=1, PRVADR=USER_PRIVS);           ! Restore old privileges
;  339          0945  2 $DASSGN (CHAN = .CHANNEL);
;  340          0946  2 RETURN .STATUS;
```

```
;  341          0947  2
;  342          0948  1 END;                                ! end of routine DISMNT_COMMAND

                                                           .TITLE  DISMOU
                                                           .IDENT  \V04-000\

                                                           .PSECT  Z$DISMOUNT,NOWRT,  SHR,  PIC,2

                                        24  54  4D  44  00000 P.AAA:  .ASCII  \DMT$\                                  ;
                                                    20  00004        .ASCII  \ \                                    ;
                                                    20  00005        .ASCII  \ \                                    ;
                                                    20  00006        .ASCII  \ \                                    ;
                                                    20  00007        .ASCII  \ \                                    ;
                                                    20  00008        .ASCII  \ \                                    ;
                                                    20  00009        .ASCII  \ \                                    ;
                                                    20  0000A        .ASCII  \ \                                    ;
                                                    20  0000B        .ASCII  \ \                                    ;
                                                    20  0000C        .ASCII  \ \                                    ;
                                                    20  0000D        .ASCII  \ \                                    ;
                                                    20  0000E        .ASCII  \ \                                    ;
                                                    20  0000F        .ASCII  \ \                                    ;
                                                    20  00010        .ASCII  \ \                                    ;
                                                    20  00011        .ASCII  \ \                                    ;
                                                    20  00012        .ASCII  \ \                                    ;
                                                    20  00013        .ASCII  \ \                                    ;
                                                    20  00014        .ASCII  \ \                                    ;
                                                    20  00015        .ASCII  \ \                                    ;
                                                    20  00016        .ASCII  \ \                                    ;
                                                    20  00017        .ASCII  \ \                                    ;
                                                    20  00018        .ASCII  \ \                                    ;
                                                    20  00019        .ASCII  \ \                                    ;
                                                    20  0001A        .ASCII  \ \                                    ;
                                                    20  0001B        .ASCII  \ \                                    ;
                                                    20  0001C        .ASCII  \ \                                    ;
                                                    20  0001D        .ASCII  \ \                                    ;
                                                    20  0001E        .ASCII  \ \                                    ;
                                                    20  0001F        .ASCII  \ \                                    ;
                                                001C  00020 P.AAB:  .WORD   28                                     ;
                                                00EC  00022        .WORD   236                                     ;
                                            00000000  00024        .LONG   0                                       ;
                                            00000000  00028        .LONG   0                                       ;
                                            00000000  0002C        .LONG   0                                       ;

                                                           .PSECT  $GLOBAL$,NOEXE,2

                                                00000 CLUSTER_DEVICE::
                                                           .BLKB   4

                                                           .EXTRN  EXE$PROBER_DSC, SYS$ASSIGN
                                                           .EXTRN  SYS$SETPRV, SYS$GETDVIW
                                                           .EXTRN  SYS$ENQW, SYS$DEQ
                                                           .EXTRN  SYS$DASSGN

                                                           .PSECT  Z$DISMOUNT,NOWRT,  SHR,  PIC,2

                                                007C 00000        .ENTRY  SYS$DISMOU, Save R2,R3,R4,R5,R6          ; 0776
```

```
                        56 00000000G   00  9E 00002          MOVAB   SYS$SETPRV, R6                                    ;
                        5E     FF7C    CE  9E 00009          MOVAB   -132(SP), SP                                      ;
                1C  AE          BE      AF  20 28 0000E       MOVC3   #32, P.AAA, DMTLCKNAM_BUF                         ; 0837
                                    14  AE  D4 00014          CLRL    DMTLCKNAM_DSC                                     ;
                            18  AE      1C  AE 9E 00017       MOVAB   DMTLCKNAM_BUF, DMTLCKNAM_DSC+4                    ;
                04  AE          D0      AF  10 28 0001C       MOVC3   #16, P.AAB, ITMLST                                ; 0853
                            08  AE      20  AE 9E 00022       MOVAB   DMTLCKNAM_BUF+4, ITMLST+4                         ; 0848
                            0C  AE      14  AE 9E 00027       MOVAB   DMTLCKNAM_DSC, ITMLST+8                           ; 0837
                                51          04  AC D0 0002C   MOVL    DEVNAM, R1                                        ; 0859
                                    00000000G   00  16 00030  JSB     EXE$PROBER_DSC                                    ;
                                53              50  D0 00036  MOVL    R0, STATUS                                        ;
                                36              53  E9 00039  BLBC    STATUS, 1$                                        ;
                            7C  AE              51  3C 0003C  MOVZWL  LENGTH, DEV_NAME                                  ; 0862
                            FC  AD              52  D0 00040  MOVL    ADDRESS, DEV_NAME+4                               ; 0865
                            64  AE              20  D0 00044  MOVL    #32, PHYS_NAME                                    ; 0872
                            68  AE          44  AE  9E 00048  MOVAB   NAME_BUFFER, PHYS_NAME+4                          ; 0873
                                                6E  D4 0004D  CLRL    CHANNEL                                           ; 0882
                                            64  AE  9F 0004F  PUSHAB  PHYS_NAME                                         ; 0883
                                            F8  AD  9F 00052  PUSHAB  DEV_NAME                                         ;
                        0000V   CF              02  FB 00055  CALLS   #2, TRAN_LOGNAME                                 ;
                                53              50  D0 0005A  MOVL    R0, STATUS                                        ;
                                12              53  E9 0005D  BLBC    STATUS, 1$                                        ;
                                                7E  7C 00060  CLRQ    -(SP)                                             ; 0886
                                            08  AE  9F 00062  PUSHAB  CHANNEL                                           ;
                                            70  AE  9F 00065  PUSHAB  PHYS_NAME                                         ;
                    00000000G   00              04  FB 00068  CALLS   #4, SYS$ASSIGN                                   ;
                                53              50  D0 0006F  MOVL    R0, STATUS                                        ;
                                03              53  E8 00072  1$:    BLBS    STATUS, 2$                                ;
                                            0095    31 00075  BRW     5$                                                ;
                                        6C  AE      7C 00078  2$:    CLRQ    DISMOUNT_PRIVS                            ; 0893
                            6E  AE          88  8F  88 0007B  BISB2   #136, DISMOUNT_PRIVS+2                           ; 0896
                                            74  AE  9F 00080  PUSHAB  USER_PRIVS                                       ; 0897
                                                7E  D4 00083  CLRL    -(SP)                                             ;
                                            74  AE  9F 00085  PUSHAB  DISMOUNT_PRIVS                                   ;
                                                01  DD 00088  PUSHL   #1                                               ;
                                66                  04  FB 0008A  CALLS   #4, SYS$SETPRV                               ;
                                                    7E  7C 0008D  CLRQ    -(SP)                                         ; 0906
                                                    7E  7C 0008F  CLRQ    -(SP)                                         ;
                                            14  AE  9F 00091  PUSHAB  ITMLST                                           ;
                                                7E  D4 00094  CLRL    -(SP)                                             ;
                                            18  AE  DD 00096  PUSHL   CHANNEL                                         ;
                                                1A  DD 00099  PUSHL   #26                                               ;
                    00000000G   00                  08  FB 0009B  CALLS   #8, SYS$GETDVIW                               ;
                            14  AE                  04  C0 000A2  ADDL2   #4, DMTLCKNAM_DSC                            ; 0908
                                                    7E  7C 000A6  CLRQ    -(SP)                                         ; 0914
                                                    7E  7C 000A8  CLRQ    -(SP)                                         ;
                                                    7E  7C 000AA  CLRQ    -(SP)                                         ;
                                            2C  AE  9F 000AC  PUSHAB  DMTLCKNAM_DSC                                    ;
                                                10  DD 000AF  PUSHL   #16                                             ;
                                            5C  AE  9F 000B1  PUSHAB  LOCK_STATUS                                      ;
                                                05  DD 000B4  PUSHL   #5                                               ;
                                                1A  DD 000B6  PUSHL   #26                                               ;
                    00000000G   00                  0B  FB 000B8  CALLS   #11, SYS$ENQW                              ;
                                                6E  DD 000BF  PUSHL   CHANNEL                                          ; 0920
                                            08  AC  DD 000C1  PUSHL   FLAGS                                           ;
                        0000V   CF              02  FB 000C4  CALLS   #2, MAKE_DISMOUNT                              ;
                                53              50  D0 000C9  MOVL    R0, STATUS                                        ;
```

```
                                40 AE  D5 000CC          TSTL    LOCK_STATUS+4                   ; 0926
                                0E 13  000CF             BEQL    3$                              ;
                                7E 7C  000D1             CLRQ    -(SP)                           ; 0928
                                7E D4  000D3             CLRL    -(SP)                           ;
                                4C AE  DD 000D5          PUSHL   LOCK_STATUS+4                   ;
                00000000G       00     04 FB 000D8       CALLS   #4, SYS$DEQ                     ;
                                0E     53 E9 000DF 3$:   BLBC    STATUS, 4$                      ; 0935
                                   08  AC DD 000E2       PUSHL   FLAGS                           ; 0936
                                   68  AE 9F 000E5       PUSHAB  PHYS_NAME                       ;
                     0000V      CF     02 FB 000E8       CALLS   #2, DISMOUNT_CLUSTER            ;
                                53     50 D0 000ED       MOVL    R0, STATUS                      ;
                                7E 7C  000F0 4$:         CLRQ    -(SP)                           ; 0943
                                   74  AE 9F 000F2       PUSHAB  DISMOUNT_PRIVS                  ;
                                7E D4  000F5             CLRL    -(SP)                           ;
                                66     04 FB 000F7       CALLS   #4, SYS$SETPRV                  ;
                                7E 7C  000FA             CLRQ    -(SP)                           ; 0944
                                   7C  AE 9F 000FC       PUSHAB  USER_PRIVS                      ;
                                01 DD  000FF             PUSHL   #1                              ;
                                66     04 FB 00101       CALLS   #4, SYS$SETPRV                  ;
                                6E DD  00104             PUSHL   CHANNEL                         ; 0945
                00000000G       00     01 FB 00106       CALLS   #1, SYS$DASSGN                  ;
                                50     53 D0 0010D 5$:   MOVL    STATUS, R0                      ; 0946
                                          04 00110       RET                                     ; 0948

; Routine Size:  273 bytes,    Routine Base:  Z$DISMOUNT + 0030
```

```
;   344     0949  1 ROUTINE MAKE_DISMOUNT (FLAGS, CHANNEL) =
;   345     0950  1
;   346     0951  1 !++
;   347     0952  1 !
;   348     0953  1 ! FUNCTIONAL DESCRIPTION:
;   349     0954  1 !
;   350     0955  1 !       This routine does the kernel mode validation and initial setup
;   351     0956  1 !       of the dismount operation.
;   352     0957  1 !
;   353     0958  1 !
;   354     0959  1 ! INPUT PARAMETERS:
;   355     0960  1 !       FLAGS   : A longword bit mask.
;   356     0961  1 !       CHANNEL : The channel number of the channel assigned to the device.
;   357     0962  1 !
;   358     0963  1 ! IMPLICIT INPUTS:
;   359     0964  1 !       NONE.
;   360     0965  1 !
;   361     0966  1 ! OUTPUT PARAMETERS:
;   362     0967  1 !       NONE
;   363     0968  1 !
;   364     0969  1 ! IMPLICIT OUTPUTS:
;   365     0970  1 !       NONE
;   366     0971  1 !
;   367     0972  1 ! ROUTINE VALUE:
;   368     0973  1 !       1 if successful, various statuses if not
;   369     0974  1 !
;   370     0975  1 ! SIDE EFFECTS:
;   371     0976  1 !       Volume dismounted, logical name and MTL entry deleted.
;   372     0977  1 !       The cluster device characteristic bit is save in global
;   373     0978  1 !       area.
;   374     0979  1 !
;   375     0980  1 !--
;   376     0981  1
;   377     0982  2 BEGIN
;   378     0983  2
;   379     0984  2 MAP
;   380     0985  2         FLAGS           : BBLOCK;       ! flag bits for dismount options
;   381     0986  2
;   382     0987  2 BUILTIN
;   383     0988  2         REMQUE;                         ! remove an item from a queue
;   384     0989  2
;   385     0990  2 LINKAGE
;   386     0991  2         IOC_DISMOUNT    = JSB (REGISTER = 6, REGISTER = 3, REGISTER = 4) :
;   387     0992  2                         NOPRESERVE (2);
;   388     0993  2
;   389     0994  2 LITERAL
;   390     0995  2         DEVCHAR_SIZE    = 4;            ! Length (in bytes) of device characteristics
;   391     0996  2
;   392     0997  2 LOCAL
;   393     0998  2         DEVICE_CHAR     : BBLOCK [DEVCHAR_SIZE],
;   394     0999  2                                         ! buffer for device characteristics
;   395     1000  2         DEVICE_CHAR2    : BBLOCK [DEVCHAR_SIZE],
;   396     1001  2                                         ! buffer for sec. device characteristics
;   397     1002  2         DEVCHAR_DESC    : BBLOCK [DSC$K_S_BLN],
;   398     1003  2                                         ! descriptor for device characteristics
;   399     1004  2         DEVCHAR_DESC2   : BBLOCK [DSC$K_S_BLN],
;   400     1005  2                                         ! descriptor for sec. device characteristics
```

```
;  401          1006  2          MAGTAPE,                        ! flag indicating that device is magtape
;  402          1007  2          J,                              ! RVT loop index
;  403          1008  2          PRIVATE,                        ! flag indicating private mount found
;  404          1009  2          RVT_LENGTH,                     ! number of entries in RVT
;  405          1010  2          LIST_HEAD       : REF VECTOR,   ! pointer to current mount list head
;  406          1011  2          CCB             : REF BBLOCK,   ! CCB of channel (points to UCB)
;  407          1012  2          UCB             : REF BBLOCK,   ! UCB of device
;  408          1013  2          VCB             : REF BBLOCK,   ! VCB of device
;  409          1014  2          RVT             : REF BBLOCK,   ! address of relative volume table
;  410          1015  2          MTL             : REF BBLOCK,   ! address of found MTL entry
;  411          1016  2          STATUS;                         ! status of various routines
;  412          1017  2
;  413          1018  2  EXTERNAL
;  414          1019  2          CTL$GL_CCBBASE  : ADDRESSING_MODE (GENERAL),
;  415          1020  2                                          ! base address of CCB table
;  416          1021  2          SCH$GL_CURPCB   : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  417          1022  2                                          ! address of our PCB
;  418          1023  2          CTL$GL_PHD      : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  419          1024  2                                          ! address of our process header
;  420          1025  2          EXE$GL_SYSUCB   : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  421          1026  2                                          ! address of system device UCB
;  422          1027  2          CTL$GQ_MOUNTLST : VECTOR ADDRESSING_MODE (GENERAL),
;  423          1028  2                                          ! temporary mount list head
;  424          1029  2          IOC$GQ_MOUNTLST : VECTOR ADDRESSING_MODE (GENERAL);
;  425          1030  2                                          ! system mounted volume listhead
;  426          1031  2
;  427          1032  2  EXTERNAL ROUTINE
;  428          1033  2          LOCK_IODB,                      ! lock I/O database
;  429          1034  2          UNLOCK_IODB,                    ! unlock I/O database
;  430          1035  2          IOC$DISMOUNT    : IOC_DISMOUNT ADDRESSING_MODE (GENERAL);
;  431          1036  2                                          ! system dismount routine
;  432          1037  2
;  433          1038  2
;  434          1039  2  ! Get the device characteristics and make sure it can be dismounted at all.
;  435          1040  2  ! i.e., that it is file oriented, etc. A mismatch between primary and
;  436          1041  2  ! secondary device characteristics indicates a spooled device or something
;  437          1042  2  ! else strange - reject it if so.
;  438          1043  2  !
;  439          1044  2
;  440          1045  2  DEVCHAR_DESC[DSC$B_CLASS]       = 0;    ! set up primary characteristics buffer descriptor
;  441          1046  2  DEVCHAR_DESC[DSC$B_DTYPE]       = 0;
;  442          1047  2  DEVCHAR_DESC[DSC$W_LENGTH]      = DEVCHAR_SIZE;
;  443          1048  2  DEVCHAR_DESC[DSC$A_POINTER]     = DEVICE_CHAR;
;  444          1049  2
;  445          1050  2  DEVCHAR_DESC2[DSC$B_CLASS]      = 0;    ! set up secondary characteristics buffer descriptor
;  446          1051  2  DEVCHAR_DESC2[DSC$B_DTYPE]      = 0;
;  447          1052  2  DEVCHAR_DESC2[DSC$W_LENGTH]     = DEVCHAR_SIZE;
;  448          1053  2  DEVCHAR_DESC2[DSC$A_POINTER]    = DEVICE_CHAR2;
;  449          1054  2
;  450          1055  2  $GETCHN (CHAN = .CHANNEL, PRIBUF = DEVCHAR_DESC, SCDBUF = DEVCHAR_DESC2);
;  451          1056  2
;  452          1057  2  IF CH$NEQ (DEVCHAR_SIZE, DEVICE_CHAR, DEVCHAR_SIZE, DEVICE_CHAR2, 0)
;  453          1058  2  OR NOT .DEVICE_CHAR[DEV$V_FOD]
;  454          1059  2  THEN RETURN (SS$_NOTFILEDEV);
;  455          1060  2
;  456          1061  2  IF NOT .DEVICE_CHAR[DEV$V_AVL]
;  457          1062  2  THEN RETURN (SS$_DEVOFFLINE);
```

```
;  458          1063  2
;  459          1064  2 IF NOT .DEVICE_CHAR[DEV$V_MNT] OR .DEVICE_CHAR[DEV$V_DMT]
;  460          1065  2 THEN RETURN (SS$_DEVNOTMOUNT);
;  461          1066  2
;  462          1067  2
;  463          1068  2 ! Get the UCB and VCB addresses for the channel. If this is a volume set also
;  464          1069  2 ! get the RVT address; for a disk volume set we will iterate for all volumes
;  465          1070  2 ! (provided /UNIT was not specified). First we search the process mounted volume
;  466          1071  2 ! list for entries of the volume; if found, we remove them and proceed with the
;  467          1072  2 ! dismount. If none were found, we try the system mounted volume list for
;  468          1073  2 ! volumes mounted /GROUP or /SYSTEM. Dismounting these requires the appropriate
;  469          1074  2 ! privilege.
;  470          1075  2 !
;  471          1076  2 CCB = .CTL$GL_CCBBASE - .CHANNEL;
;  472          1077  2 UCB = .CCB[CCB$L_UCB];
;  473          1078  2 VCB = .UCB[UCB$L_VCB];
;  474          1079  2 PRIVATE = 0;
;  475          1080  2 RVT = 0;
;  476          1081  2 RVT_LENGTH = 0;
;  477          1082  2 MAGTAPE = .BBLOCK [UCB[UCB$L_DEVCHAR], DEV$V_SQD];
;  478          1083  2 CLUSTER_DEVICE = .BBLOCK [UCB [UCB$L_DEVCHAR2], DEV$V_CLU]; ! Save cluster device characteristic bit
;  479          1084  2
;  480          1085  2 IF NOT .BBLOCK [UCB[UCB$L_DEVCHAR], DEV$V_FOR]
;  481          1086  4 AND ((.VCB[VCB$W_RVN] NEQ 0 AND NOT .FLAGS [DMT$V_UNIT])
;  482          1087  3     OR .MAGTAPE
;  483          1088  3     )
;  484          1089  2 THEN
;  485          1090  3     BEGIN
;  486          1091  3     RVT = .VCB[VCB$L_RVT];
;  487          1092  3     RVT_LENGTH = .RVT[RVT$B_NVOLS];
;  488          1093  3     END;
;  489          1094  2
;  490          1095  2 STATUS = CHECK_PRIV (.UCB, .FLAGS);      ! check privilege
;  491          1096  2 IF NOT .STATUS THEN RETURN (.STATUS);    ! if failed, return immediately
;  492          1097  2
;  493          1098  2 SETUP_MTL (.UCB,.FLAGS);                 ! set up local mounted volume database
;  494          1099  2
;  495          1100  2 LIST_HEAD = CTL$GQ_MOUNTLST[0];          ! point to local mounted volume database
;  496          1101  2
;  497          1102  2
;  498          1103  2
;  499          1104  2 WHILE 1 DO                              ! loop forever
;  500          1105  2
;  501          1106  3 BEGIN
;  502          1107  3
;  503          1108  3 DECR K FROM 2 TO 1 DO                  ! loop for process, then system mount list
;  504          1109  4     BEGIN
;  505          1110  4     J = 0;
;  506          1111  4
;  507          1112  4     DO                                  ! loop for all entries in RVT
;  508          1113  5         BEGIN
;  509          1114  5         IF .RVT NEQ 0
;  510          1115  5         THEN UCB = .VECTOR [RVT[RVT$L_UCBLST], .J];
;  511          1116  5
;  512          1117  5         IF .UCB NEQ 0
;  513          1118  5         THEN
;  514          1119  6             BEGIN
```

```
;   515          1120  6              VCB = .UCB[UCB$L_VCB];
;   516          1121  6              !
;   517          1122  6              ! Note: With job-wide mount support, the check below is no longer
;   518          1123  6              ! appropriate. With job-wide mount, any process in the process
;   519          1124  6              ! tree can mount/dismount the volume. In a private mount, the
;   520          1125  6              ! device is allocated to the parent process, and a subprocess
;   521          1126  6              ! should be able to dismount the volume.
;   522          1127  6              !
;   523          1128  6              ! IF .BBLOCK[UCB[UCB$L_DEVCHAR], DEV$V_ALL]
;   524          1129  6              ! AND .UCB[UCB$L_PID] NEQ .SCH$GL_CURPCB[PCB$L_PID]
;   525          1130  6              ! THEN RETURN (SS$_DEVALLOC);
;   526          1131  6              !
;   527          1132  6              LOCK_IODB ();
;   528          1133  6              MTL = SEARCH_MOUNT (.LIST_HEAD, .UCB);
;   529          1134  6              IF .MTL NEQ 0
;   530          1135  6              THEN
;   531          1136  7                  BEGIN
;   532          1137  7
;   533          1138  7                  IF NOT .K                   ! if first pass (private list)
;   534          1139  7                  THEN
;   535          1140  8                      BEGIN
;   536          1141  8                      PRIVATE = 1;
;   537          1142  8                      END
;   538          1143  8
;   539          1144  7                  ELSE                        ! if second pass (system list)
;   540          1145  8                      BEGIN
;   541          1146  8              !
;   542          1147  8              ! Clear the /SYSTEM or /GROUP bits to correctly show residual /SHARE mounts.
;   543          1148  8              !
;   544          1149  8
;   545          1150  8                      VCB[VCB$V_GROUP] = 0;
;   546          1151  8                      VCB[VCB$V_SYSTEM] = 0;
;   547          1152  7                      END;
;   548          1153  7
;   549          1154  7              ! Having passed all the checks, take the MTL entry out of the list and call
;   550          1155  7              ! the system dismount routine with it.
;   551          1156  7              !
;   552          1157  7
;   553          1158  7                  REMQUE (.MTL, MTL);
;   554          1159  7                  UNLOCK_IODB ();
;   555          1160  7                  DISMOUNT_AUDIT (.FLAGS, .CHANNEL, .UCB, .MTL);
;   556          1161  7                  IOC$DISMOUNT (.MTL, .FLAGS[DMT$V_NOUNLOAD], .SCH$GL_CURPCB);
;   557          1162  7
;   558          1163  7              ! On RVN 1 of a disk volume set, there are two MTL entries. Find and process
;   559          1164  7              ! the second.
;   560          1165  7              !
;   561          1166  7
;   562          1167  7                  IF .J EQL 0
;   563          1168  8                  AND (.RVT NEQ 0 OR .FLAGS[DMT$V_UNIT])
;   564          1169  7                  AND NOT .MAGTAPE
;   565          1170  7                  THEN
;   566          1171  8                      BEGIN
;   567          1172  8                      LOCK_IODB ();
;   568          1173  8                      MTL = SEARCH_MOUNT (.LIST_HEAD, .UCB);
;   569          1174  8                      IF .MTL NEQ 0
;   570          1175  8                      THEN
;   571          1176  9                          BEGIN
```

```
;  572          1177  9                    REMQUE (.MTL, MTL);
;  573          1178  9                    UNLOCK_IODB ();
;  574          1179  9                    IOC$DISMOUNT (.MTL, .FLAGS[DMT$V_NOUNLOAD], .SCH$GL_CURPCB);
;  575          1180  9                    END
;  576          1181  8                ELSE
;  577          1182  8                    UNLOCK_IODB ();
;  578          1183  7                END;
;  579          1184  7            END
;  580          1185  7
;  581          1186  7 ! If normal dismount, failure to find an MTL entry on the second pass is an error.
;  582          1187  7 ! If a forced dismount, keep looping for more MTL entries.
;  583          1188  7
;  584          1189  6            ELSE
;  585          1190  7                BEGIN
;  586          1191  7                UNLOCK_IODB ();
;  587          1192  7                IF .K AND NOT .FLAGS [DMT$V_ABORT]
;  588          1193  7                THEN RETURN (SS$_DEVNOTMOUNT);
;  589          1194  6                END;
;  590          1195  6
;  591          1196  6 ! We exit the RVT scan loop if this is magtape or a single disk volume.
;  592          1197  6 !
;  593          1198  6
;  594          1199  6            IF .MAGTAPE
;  595          1200  6            OR .RVT EQL 0
;  596          1201  6            THEN EXITLOOP;
;  597          1202  6
;  598          1203  5            END;                            ! end of UCB NEQ 0 condition
;  599          1204  5
;  600          1205  5        J = .J + 1;
;  601          1206  5        END                                 ! end of RVT scan loop
;  602          1207  4    UNTIL .J GEQU .RVT_LENGTH;
;  603          1208  4
;  604          1209  4 ! If any entries were found in the process mounted volume list, we are now
;  605          1210  4 ! done. If not, go back to try the whole volume set against the system list.
;  606          1211  4 !
;  607          1212  4
;  608          1213  4    IF NOT .PRIVATE
;  609          1214  4    THEN
;  610          1215  5        BEGIN
;  611          1216  5        IF .MAGTAPE
;  612          1217  5        THEN RETURN (SS$_DEVNOTMOUNT);
;  613          1218  5        END
;  614          1219  4    ELSE
;  615          1220  4        IF NOT .FLAGS [DMT$V_ABORT] THEN EXITLOOP; ! If normal dismount, get out
;  616          1221  4                                            ! for forced dismount, keep looping
;  617          1222  4
;  618          1223  4
;  619          1224  4    LIST_HEAD = IOC$GQ_MOUNTLST[0];         ! switch to system-wide mount list
;  620          1225  4
;  621          1226  3    END;                                    ! end of private/system scan loop
;  622          1227  3
;  623          1228  3
;  624          1229  3 IF NOT .FLAGS [DMT$V_ABORT]               ! If normal dismount, get out
;  625          1230  3 THEN EXITLOOP
;  626          1231  3 ELSE
;  627          1232  3    IF .CTL$GQ_MOUNTLST[1] EQL CTL$GQ_MOUNTLST[0]
;  628          1233  3    THEN EXITLOOP;                          ! for forced dismount, if mount list
```

```
;  629     1234  3                                    ! is empty, exit while loop
;  630     1235  3
;  631     1236  3 !
;  632     1237  3 ! Reinitialize critical variables for another iteration for forced dismount
;  633     1238  3 !
;  634     1239  3 UCB = .CCB[CCB$L_UCB];               ! reinitialize UCB address
;  635     1240  3 PRIVATE = 0;                        ! and privately mounted flag
;  636     1241  3
;  637     1242  3 LIST_HEAD = CTL$GQ_MOUNTLST[0];     ! point to local mounted volume database
;  638     1243  3
;  639     1244  3
;  640     1245  2 END;                                ! forever loop
;  641     1246  2
;  642     1247  2
;  643     1248  2 RETURN 1;
;  644     1249  2
;  645     1250  1 END;                                ! end of routine MAKE_DISMOUNT


                                                        .EXTRN  CTL$GL_CCBBASE, SCH$GL_CURPCB
                                                        .EXTRN  CTL$GL_PHD, EXE$GL_SYSUCB
                                                        .EXTRN  CTL$GQ_MOUNTLST
                                                        .EXTRN  IOC$GQ_MOUNTLST
                                                        .EXTRN  LOCK_IODB, UNLOCK_IODB
                                                        .EXTRN  IOC$DISMOUNT, SYS$GETCHN

                                    0FFC 00000 MAKE_DISMOUNT:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0949
                                         5E             28  C2 00002          SUBL2   #40, SP                     ;
                                   20    AE             04  D0 00005          MOVL    #4, DEVCHAR_DESC            ; 1047
                                   24    AE         10  AE  9E 00009          MOVAB   DEVICE_CHAR, DEVCHAR_DESC+4 ; 1048
                                   18    AE             04  D0 0000E          MOVL    #4, DEVCHAR_DESC2           ; 1052
                                   1C    AE         14  AE  9E 00012          MOVAB   DEVICE_CHAR2, DEVCHAR_DESC2+4 ; 1053
                                                    18  AE  9F 00017          PUSHAB  DEVCHAR_DESC2               ; 1055
                                                        7E  D4 0001A          CLRL    -(SP)                       ;
                                                    28  AE  9F 0001C          PUSHAB  DEVCHAR_DESC                ;
                                                        7E  D4 0001F          CLRL    -(SP)                       ;
                                                    08  AC  DD 00021          PUSHL   CHANNEL                     ;
                                   00000000G    00         05  FB 00024      CALLS   #5, SYS$GETCHN              ;
                                         14    AE         10  AE  D1 0002B      CMPL    DEVICE_CHAR, DEVICE_CHAR2   ; 1057
                                                            05  12 00030      BNEQ    1$                          ;
                                   06        11    AE             06  E0 00032  BBS     #6, DEVICE_CHAR+1, 2$       ; 1058
                                                  50        01CC  8F  3C 00037  1$:     MOVZWL  #460, R0            ; 1059
                                                                  04 0003C      RET                                 ;
                                   05        12    AE             02  E0 0003D  2$:     BBS     #2, DEVICE_CHAR+2, 3$       ; 1061
                                                  50            84  8F  9A 00042      MOVZBL  #132, R0            ; 1062
                                                                  04 00046      RET                                 ;
                                   03        12    AE             03  E0 00047  3$:     BBS     #3, DEVICE_CHAR+2, 5$       ; 1064
                                                                0134  31 0004C  4$:     BRW     23$                 ;
                                   F8        12    AE             05  E0 0004F  5$:     BBS     #5, DEVICE_CHAR+2, 4$       ;
                              08   AE 00000000G    00         08  AC  C3 00054      SUBL3   CHANNEL, CTL$GL_CCBBASE, CCB ; 1076
                                                  55         08  BE  D0 0005E      MOVL    @CCB, UCB               ; 1077
                                                  57         34  A5  D0 00062      MOVL    52(UCB), VCB            ; 1078
                                                             04  AE  D4 00066      CLRL    PRIVATE                 ; 1079
                                                                 58  D4 00069      CLRL    RVT                     ; 1080
                                                             0C  AE  D4 0006B      CLRL    RVT_LENGTH              ; 1081
```

```
      6E      38   A5        01           05  EF 0006E          EXTZV   #5, #1, 56(UCB), MAGTAPE                    ; 1082
0000'  CF     3C   A5        01           00  EF 00074          EXTZV   #0, #1, 60(UCB), CLUSTER_DEVICE             ; 1083
                             16      3B   A5  E8 0007C          BLBS    59(UCB), 8$                                 ; 1085
                                     0E   A7  B5 00080          TSTW    14(VCB)                                     ; 1086
                                          05  13 00083          BEQL    6$                                          ;
              03        04   AC           01  E1 00085          BBC     #1, FLAGS, 7$                               ;
                             09           6E  E9 0008A  6$:     BLBC    MAGTAPE, 8$                                 ; 1087
                             58      20   A7  D0 0008D  7$:     MOVL    32(VCB), RVT                                ; 1091
                        0C   AE      0B   A8  9A 00091          MOVZBL  11(RVT), RVT_LENGTH                         ; 1092
                                     04   AC  DD 00096  8$:     PUSHL   FLAGS                                       ; 1095
                                          55  DD 00099          PUSHL   UCB                                         ;
                     0000V   CF           02  FB 0009B          CALLS   #2, CHECK_PRIV                              ;
                             01           50  E8 000A0          BLBS    STATUS, 9$                                  ; 1096
                                          04     000A3          RET                                                 ;
                                     04   AC  DD 000A4  9$:     PUSHL   FLAGS                                       ; 1098
                                          55  DD 000A7          PUSHL   UCB                                         ;
                     0000V   CF           02  FB 000A9          CALLS   #2, SETUP_MTL                               ;
                             5B 00000000G 00  9E 000AE  10$:    MOVAB   CTL$GQ_MOUNTLST, LIST_HEAD                  ; 1100
                             5A           02  D0 000B5          MOVL    #2, K                                       ; 1108
                                          59  D4 000B8  11$:    CLRL    J                                           ; 1110
                                          58  D5 000BA  12$:    TSTL    RVT                                         ; 1114
                                          05  13 000BC          BEQL    13$                                         ;
                             55      44 A849  D0 000BE          MOVL    68(RVT)[J], UCB                             ; 1115
                                          55  D5 000C3  13$:    TSTL    UCB                                         ; 1117
                                          03  12 000C5          BNEQ    14$                                         ;
                                        00A7  31 000C7          BRW     21$                                         ;
                             57      34   A5  D0 000CA  14$:    MOVL    52(UCB), VCB                                ; 1120
                     0000G   CF           00  FB 000CE          CALLS   #0, LOCK_IODB                               ; 1132
                                          55  DD 000D3          PUSHL   UCB                                         ; 1133
                                          5B  DD 000D5          PUSHL   LIST_HEAD                                   ;
                     0000V   CF           02  FB 000D7          CALLS   #2, SEARCH_MOUNT                            ;
                             56           50  D0 000DC          MOVL    R0, MTL                                     ;
                                          7C  13 000DF          BEQL    19$                                         ; 1134
                             06           5A  E8 000E1          BLBS    K, 15$                                      ; 1138
                        04   AE           01  D0 000E4          MOVL    #1, PRIVATE                                 ; 1141
                                          05  11 000E8          BRB     16$                                         ; 1138
                        0B   A7      C0   8F  8A 000EA  15$:    BICB2   #192, 11(VCB)                               ; 1151
                             56           66  0F 000EF  16$:    REMQUE  (MTL), MTL                                  ; 1158
                     0000G   CF           00  FB 000F2          CALLS   #0, UNLOCK_IODB                             ; 1159
                             7E           55  7D 000F7          MOVQ    UCB, -(SP)                                  ; 1160
                             7E      04   AC  7D 000FA          MOVQ    FLAGS, -(SP)                                ;
                     0000V   CF           04  FB 000FE          CALLS   #4, DISMOUNT_AUDIT                          ;
                             54 00000000G 00  D0 00103          MOVL    SCH$GL_CURPCB, R4                           ; 1161
      53      04   AC        01           00  EF 0010A          EXTZV   #0, #1, FLAGS, R3                           ;
                                00000000G 00  16 00110          JSB     IOC$DISMOUNT                                ;
                                          59  D5 00116          TSTL    J                                           ; 1167
                                          50  12 00118          BNEQ    20$                                         ;
                                          58  D5 0011A          TSTL    RVT                                         ; 1168
                                          05  12 0011C          BNEQ    17$                                         ;
              47        04   AC           01  E1 0011E          BBC     #1, FLAGS, 20$                              ;
                             56           6E  E8 00123  17$:    BLBS    MAGTAPE, 22$                                ; 1169
                     0000G   CF           00  FB 00126          CALLS   #0, LOCK_IODB                               ; 1172
                                          55  DD 0012B          PUSHL   UCB                                         ; 1173
                                          5B  DD 0012D          PUSHL   LIST_HEAD                                   ;
                     0000V   CF           02  FB 0012F          CALLS   #2, SEARCH_MOUNT                            ;
                             56           50  D0 00134          MOVL    R0, MTL                                     ;
                                          1D  13 00137          BEQL    18$                                         ; 1174
```

```
                          56          66   0F 00139         REMQUE  (MTL), MTL                                ; 1177
                   0000G  CF          00   FB 0013C         CALLS   #0, UNLOCK_IODB                           ; 1178
                          54 00000000G 00  D0 00141         MOVL    SCH$GL_CURPCB, R4                         ; 1179
      53      04    AC    01          00   EF 00148         EXTZV   #0, #1, FLAGS, R3                         ;
                             00000000G 00  16 0014E         JSB     IOC$DISMOUNT                              ;
                                      14   11 00154         BRB     20$                                       ; 1174
                   0000G  CF          00   FB 00156  18$:   CALLS   #0, UNLOCK_IODB                           ; 1182
                                      0D   11 0015B         BRB     20$                                       ; 1167
                   0000G  CF          00   FB 0015D  19$:   CALLS   #0, UNLOCK_IODB                           ; 1191
                          05          5A   E9 00162         BLBC    K, 20$                                    ; 1192
            19      04    AC          02   E1 00165         BBC     #2, FLAGS, 23$                            ;
                          0F          6E   E8 0016A  20$:   BLBS    MAGTAPE, 22$                              ; 1199
                                      58   D5 0016D         TSTL    RVT                                       ; 1200
                                      0B   13 0016F         BEQL    22$                                       ;
                                      59   D6 00171  21$:   INCL    J                                         ; 1205
                    0C    AE          59   D1 00173         CMPL    J, RVT_LENGTH                             ; 1207
                                      03   1E 00177         BGEQU   22$                                       ;
                                    FF3E   31 00179         BRW     12$                                       ;
                          08     04   AE   E8 0017C  22$:   BLBS    PRIVATE, 24$                              ; 1213
                          0A          6E   E9 00180         BLBC    MAGTAPE, 25$                              ; 1216
                          50     7C   8F   9A 00183  23$:   MOVZBL  #124, R0                                  ; 1217
                                           04 00187         RET                                               ;
            2E      04    AC          02   E1 00188  24$:   BBC     #2, FLAGS, 28$                            ; 1220
                          5B 00000000G 00  9E 0018D  25$:   MOVAB   IOC$GQ_MOUNTLST, LIST_HEAD                ; 1224
                          02          5A   F5 00194         SOBGTR  K, 26$                                    ; 1108
                                      03   11 00197         BRB     27$                                       ;
                                    FF1C   31 00199  26$:   BRW     11$                                       ;
            1A      04    AC          02   E1 0019C  27$:   BBC     #2, FLAGS, 28$                            ; 1229
                          50 00000000G 00  9E 001A1         MOVAB   CTL$GQ_MOUNTLST, R0                       ; 1232
                          50 00000000G 00  D1 001A8         CMPL    CTL$GQ_MOUNTLST+4, R0                     ;
                                      0A   13 001AF         BEQL    28$                                       ;
                          55     08   BE   D0 001B1         MOVL    @CCB, UCB                                 ; 1239
                                 04   AE   D4 001B5         CLRL    PRIVATE                                   ; 1240
                                    FEF3   31 001B8         BRW     10$                                       ; 1242
                          50          01   D0 001BB  28$:   MOVL    #1, R0                                    ; 1248
                                           04 001BE         RET                                               ; 1250

; Routine Size:  447 bytes,     Routine Base:  Z$DISMOUNT + 0141
```

```
;  647     1251  1 ROUTINE TRAN_LOGNAME (LOG_NAME, RESULT)  =
;  648     1252  1
;  649     1253  1 !++
;  650     1254  1 !
;  651     1255  1 ! FUNCTIONAL DESCRIPTION:
;  652     1256  1 !
;  653     1257  1 !       This routine performs simple recursive logical name translation.
;  654     1258  1 !
;  655     1259  1 !
;  656     1260  1 ! CALLING SEQUENCE:
;  657     1261  1 !       TRAN_LOGNAME (ARG1, ARG2)
;  658     1262  1 !
;  659     1263  1 ! INPUT PARAMETERS:
;  660     1264  1 !       ARG1: descriptor of logical name to translate
;  661     1265  1 !
;  662     1266  1 ! IMPLICIT INPUTS:
;  663     1267  1 !       NONE
;  664     1268  1 !
;  665     1269  1 ! OUTPUT PARAMETERS:
;  666     1270  1 !       ARG2: descriptor of result string buffer
;  667     1271  1 !               (first word receives length of result)
;  668     1272  1 !
;  669     1273  1 ! IMPLICIT OUTPUTS:
;  670     1274  1 !       NONE
;  671     1275  1 !
;  672     1276  1 ! ROUTINE VALUE:
;  673     1277  1 !       SS$_NORMAL      : The translation was a success
;  674     1278  1 !       SS$_NONLOCAL    : The device is not local to the host machine
;  675     1279  1 !       SS$_NOTRAN      : The logical name did not translate
;  676     1280  1 !
;  677     1281  1 ! SIDE EFFECTS:
;  678     1282  1 !       NONE
;  679     1283  1 !
;  680     1284  1 !--
;  681     1285  1
;  682     1286  2 BEGIN
;  683     1287  2
;  684     1288  2 MAP
;  685     1289  2         LOG_NAME        : REF BBLOCK,   ! logical name descriptor
;  686     1290  2         RESULT          : REF BBLOCK;   ! result string descriptor
;  687     1291  2
;  688     1292  2 LOCAL
;  689     1293  2         NAME_DESC       : BBLOCK [DSC$K_S_BLN], ! descriptor of current logical name string
;  690     1294  2         STATUS,                         ! system service status
;  691     1295  2         P;                              ! string search pointer
;  692     1296  2
;  693     1297  2 ! We iterate on logical name translation until the service returns SS$_NOTRAN.
;  694     1298  2 ! Perform device name extraction by using only the part of the logical name to
;  695     1299  2 ! the left of the colon (if any), also checking for node names.
;  696     1300  2 !
;  697     1301  2
;  698     1302  2 NAME_DESC[DSC$W_LENGTH] = .LOG_NAME[DSC$W_LENGTH];       ! get initial logical name
;  699     1303  2 NAME_DESC[DSC$A_POINTER] = .RESULT[DSC$A_POINTER];
;  700     1304  2 CH$COPY (.LOG_NAME[DSC$W_LENGTH],                       ! copy input to output
;  701     1305  2           .LOG_NAME[DSC$A_POINTER],
;  702     1306  2           0,
;  703     1307  2           .RESULT[DSC$W_LENGTH],
```

```
;   704          1308  2          .RESULT[DSC$A_POINTER]
;   705          1309  2         );
;   706          1310  2
;   707          1311  3 IF BEGIN
;   708          1312  3 DECR N FROM LNM$C_MAXDEPTH TO 1 DO
;   709          1313  4     BEGIN
;   710          1314  4     P = CH$FIND_CH (.NAME_DESC[DSC$W_LENGTH], .NAME_DESC[DSC$A_POINTER], ':');
;   711          1315  4     IF NOT CH$FAIL (.P)
;   712          1316  4     THEN
;   713          1317  5         BEGIN
;   714          1318  5         IF .P - .NAME_DESC[DSC$A_POINTER] LSSU .NAME_DESC[DSC$W_LENGTH] - 1
;   715          1319  5         AND .(.P)<0,16> EQL '::'
;   716          1320  5         THEN RETURN (SS$_NONLOCAL);
;   717          1321  5         NAME_DESC[DSC$W_LENGTH] = .P - .NAME_DESC[DSC$A_POINTER];
;   718          1322  4         END;
;   719          1323  4
;   720          1324  4     IF CH$RCHAR (.NAME_DESC[DSC$A_POINTER]) EQL '_'
;   721          1325  4     THEN EXITLOOP 0;
;   722          1326  4
;   723        P 1327  4     STATUS = $TRNLOG (LOGNAM = NAME_DESC[DSC$W_LENGTH],
;   724        P 1328  4                       RSLLEN = NAME_DESC[DSC$W_LENGTH],
;   725          1329  4                       RSLBUF = RESULT[DSC$W_LENGTH]);
;   726          1330  4     IF .STATUS EQL SS$_NOTRAN THEN EXITLOOP 0;
;   727          1331  4     IF NOT .STATUS THEN RETURN (.STATUS);
;   728          1332  4     END
;   729          1333  3 END
;   730          1334  2 THEN RETURN (SS$_NOTRAN);
;   731          1335  2
;   732          1336  2 ! Return the result length.
;   733          1337  2 ! The high-order word in the first longword of the result descriptor
;   734          1338  2 ! is zeroed to allow a more relaxed interpretation of the descriptor.
;   735          1339  2 !
;   736          1340  2
;   737          1341  2 RESULT[DSC$B_DTYPE]  = 0;
;   738          1342  2 RESULT[DSC$B_CLASS]  = 0;
;   739          1343  2 RESULT[DSC$W_LENGTH] = .NAME_DESC[DSC$W_LENGTH];
;   740          1344  2
;   741          1345  2 RETURN SS$_NORMAL;
;   742          1346  1 END;                                   ! end of routine TRAN_LOGNAME


                                                        .EXTRN  SYS$TRNLOG

                                              007C 00000 TRAN_LOGNAME:
                                                        .WORD   Save R2,R3,R4,R5,R6                     ; 1251
                                  5E                08  C2 00002           SUBL2   #8, SP                ;
                                  50          04    AC  D0 00005           MOVL    LOG_NAME, R0          ; 1302
                                  6E                60  B0 00009           MOVW    (R0), NAME_DESC       ;
                                  56          08    AC  D0 0000C           MOVL    RESULT, R6            ; 1303
                            04    AE          04    A6  D0 00010           MOVL    4(R6), NAME_DESC+4    ;
                66                  00        04    B0  60 2C 00015        MOVC5   (R0), @4(R0), #0, (R6), @4(R6)  ; 1308
                                                    04  B6   0001B
                                  52                0A  D0 0001D           MOVL    #10, N                ; 1312
                      04    BE          6E            3A 3A 00020 1$:     LOCC    #58, NAME_DESC, @NAME_DESC+4  ; 1314
                                                    02  12 00025           BNEQ    2$                    ;
                                                    51  D4 00027           CLRL    R1                    ;
```

```
                    53              51  D0 00029 2$:    MOVL    R1, P                           ;
                                    1F  13 0002C        BEQL    4$                              ; 1315
                51  53        04    AE  C3 0002E        SUBL3   NAME_DESC+4, P, R1              ; 1318
                    50              6E  3C 00033        MOVZWL  NAME_DESC, R0                   ;
                                    50  D7 00036        DECL    R0                              ;
                    50              51  D1 00038        CMPL    R1, R0                          ;
                                    0D  1E 0003B        BGEQU   3$                              ;
                3A3A    8F          63  B1 0003D        CMPW    (P), #14906                     ; 1319
                                    06  12 00042        BNEQ    3$                              ;
                    50      08F0    8F  3C 00044        MOVZWL  #2288, R0                       ; 1320
                                        04 00049        RET                                     ;
                    6E              51  B0 0004A 3$:    MOVW    R1, NAME_DESC                   ; 1321
                  5F    8F        04  BE  91 0004D 4$:    CMPB    @NAME_DESC+4, #95               ; 1324
                                    2F  13 00052        BEQL    6$                              ;
                                    7E  7C 00054        CLRQ    -(SP)                           ; 1329
                                    7E  D4 00056        CLRL    -(SP)                           ;
                                    56  DD 00058        PUSHL   R6                              ;
                                10  AE  9F 0005A        PUSHAB  NAME_DESC                       ;
                                14  AE  9F 0005D        PUSHAB  NAME_DESC                       ;
            00000000G   00          06  FB 00060        CALLS   #6, SYS$TRNLOG                  ;
                    54              50  D0 00067        MOVL    R0, STATUS                      ;
            00000629    8F          54  D1 0006A        CMPL    STATUS, #1577                   ; 1330
                                    10  13 00071        BEQL    6$                              ;
                    04              54  E8 00073        BLBS    STATUS, 5$                      ; 1331
                    50              54  D0 00076        MOVL    STATUS, R0                      ;
                                        04 00079        RET                                     ;
                    A3              52  F5 0007A 5$:    SOBGTR  N, 1$                           ; 1312
                    50      0629    8F  3C 0007D        MOVZWL  #1577, R0                       ; 1334
                                        04 00082        RET                                     ;
                    66              6E  3C 00083 6$:    MOVZWL  NAME_DESC, (R6)                 ; 1343
                    50              01  D0 00086        MOVL    #1, R0                          ; 1345
                                        04 00089        RET                                     ; 1346

; Routine Size:  138 bytes,    Routine Base:  Z$DISMOUNT + 0300
```

```
;  744     1347  1 ROUTINE SEARCH_MOUNT (MTL_HEAD, UCB) =
;  745     1348  1
;  746     1349  1 !++
;  747     1350  1 !
;  748     1351  1 ! FUNCTIONAL DESCRIPTION:
;  749     1352  1 !
;  750     1353  1 !       This routine searches the given mounted volume list for the entry
;  751     1354  1 !       representing the indicated UCB.
;  752     1355  1 !
;  753     1356  1 !
;  754     1357  1 ! CALLING SEQUENCE:
;  755     1358  1 !       SEARCH_MOUNT (ARG1, ARG2)
;  756     1359  1 !
;  757     1360  1 ! INPUT PARAMETERS:
;  758     1361  1 !       ARG1: address of mounted volume list head
;  759     1362  1 !       ARG2: address of desired UCB
;  760     1363  1 !
;  761     1364  1 ! IMPLICIT INPUTS:
;  762     1365  1 !       NONE
;  763     1366  1 !
;  764     1367  1 ! OUTPUT PARAMETERS:
;  765     1368  1 !       NONE
;  766     1369  1 !
;  767     1370  1 ! IMPLICIT OUTPUTS:
;  768     1371  1 !       NONE
;  769     1372  1 !
;  770     1373  1 ! ROUTINE VALUE:
;  771     1374  1 !       address of entry or 0
;  772     1375  1 !
;  773     1376  1 ! SIDE EFFECTS:
;  774     1377  1 !       NONE
;  775     1378  1 !
;  776     1379  1 !--
;  777     1380  1
;  778     1381  2 BEGIN
;  779     1382  2
;  780     1383  2 MAP
;  781     1384  2         MTL_HEAD        : REF VECTOR,   ! mounted volume list head
;  782     1385  2         UCB             : REF BBLOCK;   ! desired UCB
;  783     1386  2
;  784     1387  2 LOCAL
;  785     1388  2         MTL             : REF BBLOCK;   ! list entry in question
;  786     1389  2
;  787     1390  2
;  788     1391  2 ! Simply scan through the doubly linked list, checking consistency as we go.
;  789     1392  2 !
;  790     1393  2
;  791     1394  2 MTL = .MTL_HEAD[0];
;  792     1395  2
;  793     1396  2 UNTIL .MTL EQL MTL_HEAD[0] DO
;  794     1397  3     BEGIN
;  795     1398  3     IF .MTL[MTL$B_TYPE] NEQ DYN$C_MTL
;  796     1399  3     THEN BUG_CHECK (NOTMTLMTL, FATAL, 'Corrupted mounted volume list');
;  797     1400  3     IF .MTL[MTL$L_UCB] EQL .UCB THEN RETURN .MTL;
;  798     1401  3     MTL = .MTL[MTL$L_MTLFL];
;  799     1402  2     END;
;  800     1403  2
```

```
;  801          1404  2 RETURN 0;
;  802          1405  2
;  803          1406  1 END;                                    ! end of routine SEARCH_MOUNT


                                                                .EXTRN  BUG$_NOTMTLMTL

                                                    0000 00000 SEARCH_MOUNT:
                                                                .WORD   Save nothing                    ; 1347
                                50          04      BC  D0 00002            MOVL    @MTL_HEAD, MTL  ; 1394
                          04    AC                    50  D1 00006 1$:       CMPL    MTL, MTL_HEAD   ; 1396
                                                      16  13 0000A           BEQL    3$              ;
                                19          0A      A0  91 0000C           CMPB    10(MTL), #25    ; 1398
                                                      04  13 00010           BEQL    2$              ;
                                                        FEFF 00012           BUGW                    ; 1399
                                                       0000* 00014           .WORD   <BUG$_NOTMTLMTL!4>
                          08    AC          0C      A0  D1 00016 2$:       CMPL    12(MTL), UCB    ; 1400
                                                      07  13 0001B           BEQL    4$              ;
                                50                    60  D0 0001D           MOVL    (MTL), MTL      ; 1401
                                                      E4  11 00020           BRB     1$              ; 1396
                                                      50  D4 00022 3$:       CLRL    R0              ; 1404
                                                          04 00024 4$:       RET                     ; 1406

; Routine Size:  37 bytes,    Routine Base:  Z$DISMOUNT + 038A

;  804          1407  1
;  805          1408  1
```

```
;  807     1409  1
;  808     1410  1 ROUTINE SETUP_MTL ( UCB, FLAGS ) =
;  809     1411  1
;  810     1412  1 !+
;  811     1413  1 !
;  812     1414  1 ! FUNCTIONAL DESCRIPTION:
;  813     1415  1 !
;  814     1416  1 !       This routine sets up a local mounted volume database by collecting
;  815     1417  1 !       the appropriate mount list entries (MTLs) from the system's mounted
;  816     1418  1 !       database.
;  817     1419  1 !
;  818     1420  1 ! CALLING SEQUENCE:
;  819     1421  1 !       SETUP_MTL (ARG1, ARG2)
;  820     1422  1 !
;  821     1423  1 ! INPUT PARAMETERS:
;  822     1424  1 !       ARG1    : Address of the desired UCB
;  823     1425  1 !       ARG2    : A longword bit mask
;  824     1426  1 !
;  825     1427  1 ! IMPLICIT INPUTS:
;  826     1428  1 !       NONE
;  827     1429  1 !
;  828     1430  1 ! OUTPUT PARAMETERS:
;  829     1431  1 !       NONE
;  830     1432  1 !
;  831     1433  1 ! ROUTINE VALUE:
;  832     1434  1 !       1
;  833     1435  1 !
;  834     1436  1 ! SIDE EFFECTS:
;  835     1437  1 !       Appropriate MTLs in the system are removed from the system's
;  836     1438  1 !       mount database and inserted into the local mounted volume
;  837     1439  1 !       database.
;  838     1440  1 !
;  839     1441  1 !-
;  840     1442  1
;  841     1443  2 BEGIN
;  842     1444  2
;  843     1445  2 MAP
;  844     1446  2     UCB         : REF BBLOCK,
;  845     1447  2     FLAGS       : BBLOCK;
;  846     1448  2
;  847     1449  2 LOCAL
;  848     1450  2     PIX,                                    ! process index counter
;  849     1451  2     LIST_HEAD   : REF BBLOCK,               ! local mount listhead
;  850     1452  2     MTL         : REF BBLOCK,               ! variable for MTL
;  851     1453  2     NULL        : REF BBLOCK,               ! PCB of the null process
;  852     1454  2     PCB         : REF BBLOCK,               ! variable for PCB
;  853     1455  2     JIB         : REF BBLOCK;               ! variable for JIB
;  854     1456  2
;  855     1457  2 EXTERNAL
;  856     1458  2     SCH$GL_PCBVEC : REF VECTOR ADDRESSING_MODE (GENERAL),
;  857     1459  2                                             ! PCB vector
;  858     1460  2     SCH$GL_CURPCB : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  859     1461  2                                             ! address of current PCB
;  860     1462  2     SCH$GL_MAXPIX : ADDRESSING_MODE (GENERAL);
;  861     1463  2                                             ! max number of processes
;  862     1464  2
;  863     1465  2 EXTERNAL ROUTINE
```

```
;  864          1466  2     LOCK_IODB,
;  865          1467  2     UNLOCK_IODB;
;  866          1468  2
;  867          1469  2
;  868          1470  2 IF .FLAGS [DMT$V_ABORT]
;  869          1471  2 THEN                                    ! for dismount /abort
;  870          1472  3     BEGIN
;  871          1473  3     !
;  872          1474  3     ! Set up the local MTL database
;  873          1475  3     !
;  874          1476  3     NULL = .SCH$GL_PCBVEC [0];          ! remeber pcb of the null process
;  875          1477  3     LOCK_IODB ();                       ! lock I/O database
;  876          1478  3     INCR PIX FROM 1 TO .SCH$GL_MAXPIX   ! look thru each process in system
;  877          1479  3     DO
;  878          1480  4         BEGIN
;  879          1481  4         SET_IPL (IPL$_SYNCH);           ! raise IPL
;  880          1482  5         IF (( PCB = .SCH$GL_PCBVEC [.PIX] ) NEQ .NULL ) ! non-null process
;  881          1483  5         AND ( .PCB [PCB$L_OWNER] EQL 0 )                ! master process
;  882          1484  5         AND (( JIB = .PCB [PCB$L_JIB] ) NEQ 0 )         ! forget the swapper
;  883          1485  5         AND (.JIB [JIB$L_MTLBL] NEQ JIB [JIB$L_MTLFL] ) ! something in mountlist
;  884          1486  4         THEN
;  885          1487  5             BEGIN
;  886          1488  5             !
;  887          1489  5             ! Note that at this point, we have a JIB with at least one volume
;  888          1490  5             ! mounted. Lower the IPL to ASTDEL since MTLs are located in
;  889          1491  5             ! paged-pool. We can safely do this because the existence of
;  890          1492  5             ! an MTL entry means that this process will not be deleted
;  891          1493  5             ! until we give up the I/O database mutex.
;  892          1494  5             !
;  893          1495  5             SET_IPL (IPL$_ASTDEL);      ! lower IPL to ASTDEL since we
;  894          1496  5                                         ! still have the I/O database mutex
;  895          1497  5             LIST_HEAD = JIB [JIB$L_MTLFL];
;  896          1498  5             DO                          ! now loop until we have all MTLs for
;  897          1499  5                 MTL = MOVE_MTL ( .LIST_HEAD, .UCB, .FLAGS ) ! this process
;  898          1500  5             UNTIL ( .MTL EQL 0 );
;  899          1501  4             END;
;  900          1502  3         END;                            ! end for loop
;  901          1503  3     UNLOCK_IODB ();                     ! unlock I/O databse
;  902          1504  3     END                                 ! of dismount abort setup
;  903          1505  2 ELSE
;  904          1506  3     BEGIN                               ! normal dismount path
;  905          1507  3     JIB = .SCH$GL_CURPCB [PCB$L_JIB];   ! get our JIB address
;  906          1508  3     LIST_HEAD = JIB [JIB$L_MTLF[];      ! get job-wide mount listhead
;  907          1509  3     LOCK_IODB ();                       ! lock I/O database
;  908          1510  3     MTL = MOVE_MTL ( .LIST_HEAD, .UCB, .FLAGS ); ! set up local MTL database
;  909          1511  3     UNLOCK_IODB ();                     ! unlock I/O database
;  910          1512  2     END;                                ! of normal dismount setup
;  911          1513  2
;  912          1514  2 RETURN 1;
;  913          1515  2
;  914          1516  1 END;                                    ! of routine SETUP_MTL


                                                    .EXTRN  SCH$GL_PCBVEC, SCH$GL_MAXPIX

                                        01FC 00000 SETUP_MTL:
```

```
                                                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8                       ; 1410
                        58 00000000G   00   9E 00002                           MOVAB   SCH$GL_PCBVEC, R8                               ;
              52          08    AC       02   E1 00009                           BBC     #2, FLAGS, 4$                                   ; 1470
                        50          68   D0 0000E                           MOVL    SCH$GL_PCBVEC, R0                               ; 1476
                        57          60   D0 00011                           MOVL    (R0), NULL                                      ;
                  0000G CF          00   FB 00014                           CALLS   #0, LOCK_IODB                                   ; 1477
                        56 00000000G   00   D0 00019                           MOVL    SCH$GL_MAXPIX, R6                               ; 1478
                        54          D4   00020                           CLRL    PIX                                             ; 1482
                                    36   11 00022                           BRB     3$                                              ;
                        12          08   DA 00024 1$:                       MTPR    #8, #18                                         ; 1481
                        51          68   D0 00027                           MOVL    SCH$GL_PCBVEC, R1                               ; 1482
                        53        6144   D0 0002A                           MOVL    (R1)[PIX], PCB                                  ;
                        57          53   D1 0002E                           CMPL    PCB, NULL                                       ;
                                    27   13 00031                           BEQL    3$                                              ;
                              1C    A3   D5 00033                           TSTL    28(PCB)                                         ; 1483
                                    22   12 00036                           BNEQ    3$                                              ;
                        52      0080    C3   D0 00038                           MOVL    128(PCB), JIB                                   ; 1484
                                    1B   13 0003D                           BEQL    3$                                              ;
                        52        04    A2   D1 0003F                           CMPL    4(JIB), JIB                                     ; 1485
                                    15   13 00043                           BEQL    3$                                              ;
                        12          02   DA 00045                           MTPR    #2, #18                                         ; 1495
                        55          52   D0 00048                           MOVL    JIB, LIST_HEAD                                  ; 1497
                        7E        04    AC   7D 0004B 2$:                       MOVQ    UCB, -(SP)                                      ; 1499
                                    55   DD 0004F                           PUSHL   LIST_HEAD                                       ;
                  0000V CF          03   FB 00051                           CALLS   #3, MOVE_MTL                                    ;
                                    50   D5 00056                           TSTL    MTL                                             ; 1500
                                    F1   12 00058                           BNEQ    2$                                              ;
              C6          54          56   F3 0005A 3$:                       AOBLEQ  R6, PIX, 1$                                     ; 1478
                                    1F   11 0005E                           BRB     5$                                              ; 1503
                        50 00000000G   00   D0 00060 4$:                       MOVL    SCH$GL_CURPCB, R0                               ; 1507
                        52      0080    C0   D0 00067                           MOVL    128(R0), JIB                                    ;
                        55          52   D0 0006C                           MOVL    JIB, LIST_HEAD                                  ; 1508
                  0000G CF          00   FB 0006F                           CALLS   #0, LOCK_IODB                                   ; 1509
                        7E        04    AC   7D 00074                           MOVQ    UCB, -(SP)                                      ; 1510
                                    55   DD 00078                           PUSHL   LIST_HEAD                                       ;
                  0000V CF          03   FB 0007A                           CALLS   #3, MOVE_MTL                                    ;
                  0000G CF          00   FB 0007F 5$:                       CALLS   #0, UNLOCK_IODB                                 ; 1511
                        50          01   D0 00084                           MOVL    #1, R0                                          ; 1514
                                    04   00087                           RET                                                     ; 1516

; Routine Size:  136 bytes,    Routine Base:  Z$DISMOUNT + 03AF

;  915          1517  1
;  916          1518  1
;  917          1519  1
;  918          1520  1
```

```
;  920          1521  1
;  921          1522  1 ROUTINE MOVE_MTL ( LIST_HEAD, UCB, FLAGS ) =
;  922          1523  1
;  923          1524  1 !+
;  924          1525  1 !
;  925          1526  1 ! FUNCTIONAL DESCRIPTION:
;  926          1527  1 !
;  927          1528  1 !       This routine is an envolope to set up the local mounted
;  928          1529  1 !       volume database by calling a routine to move each MTL entry.
;  929          1530  1 !       If the requested UCB is a member of the volume set,
;  930          1531  1 !       then this routine iterates over the entire volume set,
;  931          1532  1 !       unless the DMT$V_UNIT flag is specified.
;  932          1533  1 !
;  933          1534  1 ! CALLING SEQUENCE:
;  934          1535  1 !       MOVE_MTL (ARG1, ARG2,ARG3)
;  935          1536  1 !
;  936          1537  1 ! INPUT PARAMETERS:
;  937          1538  1 !       ARG1    : Address of a mount listhead
;  938          1539  1 !       ARG2    : Address of the desired UCB
;  939          1540  1 !       ARG3    : A longwork bit mask
;  940          1541  1 !
;  941          1542  1 ! IMPLICIT INPUTS:
;  942          1543  1 !       NONE
;  943          1544  1 !
;  944          1545  1 ! OUTPUT PARAMETERS:
;  945          1546  1 !       NONE
;  946          1547  1 !
;  947          1548  1 ! ROUTINE VALUE:
;  948          1549  1 !       0  : If no MTL entry is found for the desired UCB
;  949          1550  1 !       1  : If an MTL entry is successfully set up in the local database
;  950          1551  1 !
;  951          1552  1 !
;  952          1553  1 ! SIDE EFFECTS:
;  953          1554  1 !       Appropriate MTLs in the system are removed from the mount database
;  954          1555  1 !       and inserted into the local mounted volume database.
;  955          1556  1 !
;  956          1557  1 !-
;  957          1558  1
;  958          1559  2 BEGIN
;  959          1560  2
;  960          1561  2 MAP
;  961          1562  2     LIST_HEAD   : REF BBLOCK,
;  962          1563  2     UCB         : REF BBLOCK,
;  963          1564  2     FLAGS       : BBLOCK;
;  964          1565  2
;  965          1566  2 LOCAL
;  966          1567  2     MAGTAPE,                          ! magtape indicator
;  967          1568  2     J,                                ! loop counter
;  968          1569  2     VCB         : REF BBLOCK,         ! address of VCB
;  969          1570  2     RVT         : REF BBLOCK,         ! address of RVT
;  970          1571  2     RVT_LENGTH  : REF BBLOCK,         ! length of RVT
;  971          1572  2     MTL         : REF BBLOCK,         ! local variable for MTL
;  972          1573  2     LUCB        : REF BBLOCK,         ! local variable for UCB
;  973          1574  2     VAL         : REF BBLOCK;         ! local variable for MTL
;  974          1575  2
;  975          1576  2
;  976          1577  2 MAGTAPE = .BBLOCK [UCB [UCB$L_DEVCHAR], DEV$V_SQD]; ! magtape flag
```

```
;  977          1578  2 VCB = .UCB [UCB$L_VCB];                       ! get VCB address
;  978          1579  2
;  979          1580  2 IF NOT .BBLOCK [UCB[UCB$L_DEVCHAR], DEV$V_FOR]
;  980          1581  4 AND ((.VCB[VCB$W_RVN] NEQ 0 AND NOT .FLAGS [DMT$V_UNIT])
;  981          1582  3     OR .MAGTAPE
;  982          1583  3     )
;  983          1584  2 THEN
;  984          1585  2
;  985          1586  3     BEGIN                                       ! process a volume set
;  986          1587  3     RVT = .VCB [VCB$L_RVT];                     ! get RVT address
;  987          1588  3     RVT_LENGTH = .RVT [RVT$B_NVOLS];            ! get number of volumes
;  988          1589  3     MTL = 0;
;  989          1590  3     J = 0;
;  990          1591  3     DO
;  991          1592  4         BEGIN                                   ! loop for each volume in RVT
;  992          1593  4         LUCB = .VECTOR [RVT [RVT$L_UCBLST], .J]; ! get UCB address
;  993          1594  4         IF .LUCB NEQ 0                          ! if UCB still mounted
;  994          1595  4         THEN
;  995          1596  5             BEGIN
;  996          1597  6             IF ( VAL = FIND_MTL ( .LIST_HEAD, .LUCB ) NEQ 0 )
;  997          1598  5             THEN MTL = .VAL;
;  998          1599  5             IF .J EQL 0                         ! RVN 1 of a volume set, there
;  999          1600  5             THEN                                ! are two MTL entries
; 1000          1601  6                 IF ( VAL = FIND_MTL ( .LIST_HEAD, .LUCB ) NEQ 0 )
; 1001          1602  5                 THEN MTL = .VAL;
; 1002          1603  4             END;                                ! end of UCB eql 0 condition
; 1003          1604  4         J = .J + 1;                             ! bump index
; 1004          1605  4         END
; 1005          1606  3     UNTIL  .J GEQU .RVT_LENGTH;                 !
; 1006          1607  3     END                                         ! of volume set processing
; 1007          1608  3
; 1008          1609  2 ELSE
; 1009          1610  2
; 1010          1611  2     MTL = FIND_MTL ( .LIST_HEAD, .UCB );        ! single volume, find one MTL
; 1011          1612  2
; 1012          1613  2 RETURN .MTL;
; 1013          1614  2
; 1014          1615  1 END;                                           ! routine MOVE_MTL


                                01FC 00000 MOVE_MTL:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8                    ; 1522
                            58  0000V CF 9E 00002      MOVAB   FIND_MTL, R8                      ;
                            50    08 AC D0 00007      MOVL    UCB, R0                           ; 1577
        52      38  A0          01    05 EF 0000B      EXTZV   #5, #1, 56(R0), MAGTAPE           ;
                            51    34 A0 D0 00011      MOVL    52(R0), VCB                       ; 1578
                            62    3B A0 E8 00015      BLBS    59(R0), 8$                        ; 1580
                                  0E A1 B5 00019      TSTW    14(VCB)                           ; 1581
                                     05 13 0001C      BEQL    1$                                ;
                      03    0C  AC      01 E1 0001E   BBC     #1, FLAGS, 2$                     ;
                            55          52 E9 00023 1$: BLBC    MAGTAPE, 8$                       ; 1582
                            50    20 A1 D0 00026 2$:  MOVL    32(VCB), RVT                      ; 1587
                            57    0B A0 9A 0002A      MOVZBL  11(RVT), RVT_LENGTH               ; 1588
                                     55 D4 0002E      CLRL    MTL                               ; 1589
```

```
                52  D4 00030          CLRL    J                                   ; 1590
           56   44  A0 9E 00032          MOVAB   68(RVT), R6                         ; 1593
           54     6642  D0 00036 3$:     MOVL    (R6)[J], LUCB
                  36  13 0003A          BEQL    7$                                  ; 1594
                  54  DD 0003C          PUSHL   LUCB                                ; 1597
                04    AC  DD 0003E          PUSHL   LIST_HEAD
           68         02  FB 00041          CALLS   #2, FIND_MTL
                      51  D4 00044          CLRL    R1
                      50  D5 00046          TSTL    R0
                      02  13 00048          BEQL    4$
                      51  D6 0004A          INCL    R1
           53         51  D0 0004C 4$:      MOVL    R1, VAL
           03         51  E9 0004F          BLBC    R1, 5$
           55         53  D0 00052          MOVL    VAL, MTL                        ; 1598
                      52  D5 00055 5$:      TSTL    J                               ; 1599
                      19  12 00057          BNEQ    7$
                      54  DD 00059          PUSHL   LUCB                            ; 1601
                04    AC  DD 0005B          PUSHL   LIST_HEAD
           68         02  FB 0005E          CALLS   #2, FIND_MTL
                      51  D4 00061          CLRL    R1
                      50  D5 00063          TSTL    R0
                      02  13 00065          BEQL    6$
                      51  D6 00067          INCL    R1
           53         51  D0 00069 6$:      MOVL    R1, VAL
           03         51  E9 0006C          BLBC    R1, 7$
           55         53  D0 0006F          MOVL    VAL, MTL                        ; 1602
                      52  D6 00072 7$:      INCL    J                               ; 1604
           57         52  D1 00074          CMPL    J, RVT_LENGTH                   ; 1606
                      BD  1F 00077          BLSSU   3$
                      0B  11 00079          BRB     9$                              ; 1580
                      50  DD 0007B 8$:      PUSHL   R0                              ; 1611
                04    AC  DD 0007D          PUSHL   LIST_HEAD
                   68     02  FB 00080          CALLS   #2, FIND_MTL
                   55     50  D0 00083          MOVL    R0, MTL
                   50     55  D0 00086 9$:      MOVL    MTL, R0                     ; 1613
                          04  00089          RET                                 ; 1615

; Routine Size:  138 bytes,    Routine Base:  Z$DISMOUNT + 0437

; 1015          1616  1
; 1016          1617  1
```

```
;  1018          1618  1
;  1019          1619  1 ROUTINE FIND_MTL ( LIST_HEAD, UCB ) =
;  1020          1620  1
;  1021          1621  1 !+
;  1022          1622  1 !
;  1023          1623  1 ! FUNCTIONAL DESCRIPTION:
;  1024          1624  1 !
;  1025          1625  1 !       This routine searches the mount list entries (MTLs) in the
;  1026          1626  1 !       given listhead for a given UCB. If an entry is found, it
;  1027          1627  1 !       is removed from the list and inserted into the local
;  1028          1628  1 !       MTL database.
;  1029          1629  1 !
;  1030          1630  1 !       Note: This routine must be called with the I/O database locked,
;  1031          1631  1 !       i.e. it assumes the calling routine has locked the I/O database
;  1032          1632  1 !       mutex.
;  1033          1633  1 !
;  1034          1634  1 ! CALLING SEQUENCE:
;  1035          1635  1 !       FIND_MTL (ARG1, ARG2)
;  1036          1636  1 !
;  1037          1637  1 ! INPUT PARAMETERS:
;  1038          1638  1 !       ARG1    : Address of the mounted volume listhead
;  1039          1639  1 !       ARG2    : Address of the desired UCB
;  1040          1640  1 !
;  1041          1641  1 ! IMPLICIT INPUTS:
;  1042          1642  1 !       NONE
;  1043          1643  1 !
;  1044          1644  1 ! OUTPUT PARAMETERS:
;  1045          1645  1 !       NONE
;  1046          1646  1 !
;  1047          1647  1 ! ROUTINE VALUE:
;  1048          1648  1 !       0       : If no MTL entry is found
;  1049          1649  1 !       1       : Address of the MTL entry
;  1050          1650  1 !
;  1051          1651  1 ! SIDE EFFECTS:
;  1052          1652  1 !       One MTL entry for the desired UCB is removed from the mount
;  1053          1653  1 !       database and inserted into the local mounted volume database.
;  1054          1654  1 !
;  1055          1655  1 !-
;  1056          1656  1
;  1057          1657  2 BEGIN
;  1058          1658  2
;  1059          1659  2 MAP
;  1060          1660  2     LIST_HEAD   : REF BBLOCK,           ! mount listhead
;  1061          1661  2     UCB         : REF BBLOCK;           ! UCB address
;  1062          1662  2
;  1063          1663  2 LOCAL
;  1064          1664  2     LOCAL_MOUNTLST  : REF BBLOCK,       ! local mount listhead
;  1065          1665  2     MTL             : REF BBLOCK;       ! local variable for MTL
;  1066          1666  2
;  1067          1667  2 BUILTIN
;  1068          1668  2     INSQUE,
;  1069          1669  2     REMQUE;
;  1070          1670  2
;  1071          1671  2 EXTERNAL
;  1072          1672  2     CTL$GQ_MOUNTLST : VECTOR ADDRESSING_MODE (GENERAL);
;  1073          1673  2
;  1074          1674  2
```

```
;  1075          1675  2 MTL = SEARCH_MOUNT ( .LIST_HEAD, .UCB );  ! search for MTL
;  1076          1676  2 IF .MTL NEQ 0                             ! found one
;  1077          1677  2 THEN
;  1078          1678  3     BEGIN
;  1079          1679  3     LOCAL_MOUNTLST = CTL$GQ_MOUNTLST [1]; ! set up local MTL listhead
;  1080          1680  3     REMQUE ( .MTL, MTL );                 ! remove from old mountlist
;  1081          1681  3     INSQUE ( .MTL, ..LOCAL_MOUNTLST);     ! insert into local mountlist
;  1082          1682  2     END;                                  ! done for this MTL
;  1083          1683  2
;  1084          1684  2 RETURN .MTL;
;  1085          1685  2
;  1086          1686  1 END;                                      ! of routine FIND_MTL



                                        0000 00000 FIND_MTL:
                                                           .WORD   Save nothing                                    ; 1619
                          7E        04  AC  7D 00002             MOVQ    LIST_HEAD, -(SP)                          ; 1675
                    FEBE  CF            02  FB 00006             CALLS   #2, SEARCH_MOUNT                          ;
                                        50  D5 00009             TSTL    MTL                                       ; 1676
                                        0E  13 0000B             BEQL    1$                                        ;
                          51 00000000G  00  9E 0000D             MOVAB   CTL$GQ_MOUNTLST+4, LOCAL_MOUNTLST         ; 1679
                          50            60  0F 00014             REMQUE  (MTL), MTL                                ; 1680
                       00 B1            60  0E 00017             INSQUE  (MTL), @0(LOCAL_MOUNTLST)                 ; 1681
                                        04  0001A 1$:            RET                                               ; 1686

; Routine Size:  27 bytes,    Routine Base:  Z$DISMOUNT + 04C1

;  1087          1687  1
;  1088          1688  1
```

```
;  1090     1689  1
;  1091     1690  1 ROUTINE CHECK_PRIV ( UCB, FLAGS ) =
;  1092     1691  1
;  1093     1692  1 !+
;  1094     1693  1 !
;  1095     1694  1 ! FUNCTIONAL DESCRIPTION:
;  1096     1695  1 !
;  1097     1696  1 !       This routine performs the privilege checks for the attempted
;  1098     1697  1 !       dismount operation.
;  1099     1698  1 !
;  1100     1699  1 ! CALLING SEQUENCE:
;  1101     1700  1 !       PRIV_CHECK (ARG1,ARG2)
;  1102     1701  1 !
;  1103     1702  1 ! INPUT PARAMETER:
;  1104     1703  1 !       ARG1: Address of the desired UCB
;  1105     1704  1 !       ARG2: A longword bit mask
;  1106     1705  1 !
;  1107     1706  1 ! IMPLICIT INPUTS:
;  1108     1707  1 !       NONE
;  1109     1708  1 !
;  1110     1709  1 ! IMPLICIT OUTPUTS:
;  1111     1710  1 !       NONE
;  1112     1711  1 !
;  1113     1712  1 ! ROUTINE CALUES:
;  1114     1713  1 !       SS$_NORMAL      : Success
;  1115     1714  1 !       SS$_NOPRIV      : No privilege for attempted operation
;  1116     1715  1 !       SS$_NOGRPNAM    : Operation requires GRPNAM privilege
;  1117     1716  1 !       SS$_NOSYSNAM    : Operation requires SYSNAM privilege
;  1118     1717  1 !       DIS$_SYSDEV     : Attempt to dismount the system disk
;  1119     1718  1 !
;  1120     1719  1 ! SIDE EFFECTS:
;  1121     1720  1 !       NONE
;  1122     1721  1 !
;  1123     1722  1 !-
;  1124     1723  1
;  1125     1724  2 BEGIN
;  1126     1725  2
;  1127     1726  2 MAP
;  1128     1727  2     UCB         : REF BBLOCK,
;  1129     1728  2     FLAGS       : BBLOCK;
;  1130     1729  2
;  1131     1730  2 LOCAL
;  1132     1731  2     LIST_HEAD   : REF BBLOCK,           ! local mount listhead
;  1133     1732  2     MTL         : REF BBLOCK,           ! variable for MTL
;  1134     1733  2     VCB         : REF BBLOCK,           ! VCB
;  1135     1734  2     ORB         : REF BBLOCK,           ! ORB
;  1136     1735  2     JIB         : REF BBLOCK,           ! address of the JIB
;  1137     1736  2     PRIVILEGE_MASK : REF BBLOCK,        ! process privilege mask
;  1138     1737  2     UIC;                                ! process UIC
;  1139     1738  2
;  1140     1739  2 EXTERNAL
;  1141     1740  2     IOC$GQ_MOUNTLST     : VECTOR ADDRESSING_MODE (GENERAL),
;  1142     1741  2                                         ! system-wide mount list
;  1143     1742  2     CTL$GL_PHD          : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  1144     1743  2                                         ! address of process header
;  1145     1744  2     EXE$GL_SYSUCB       : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  1146     1745  2                                         ! address of system device UCB
```

```
;  1147          1746  2     SCH$GL_CURPCB       : REF BBLOCK ADDRESSING_MODE (GENERAL);
;  1148          1747  2                                 ! address of current PCB
;  1149          1748  2
;  1150          1749  2 EXTERNAL ROUTINE
;  1151          1750  2     LOCK_IODB,
;  1152          1751  2     UNLOCK_IODB,
;  1153          1752  2     LOCK_LNM,
;  1154          1753  2     UNLOCK_LNM;
;  1155          1754  2
;  1156          1755  2
;  1157          1756  2 !
;  1158          1757  2 ! If this UCB is mounted by the current process and this is a normal
;  1159          1758  2 ! dismount request, we immediately return without further privilege
;  1160          1759  2 ! checks.
;  1161          1760  2 !
;  1162          1761  2 ! If this is a dismount /abort or /cluster request, then there are three
;  1163          1762  2 ! seperate checks:
;  1164          1763  2 !
;  1165          1764  2 ! 1. If the volume is mounted /system, the dismounter must have SYSNAM
;  1166          1765  2 !    privilege.
;  1167          1766  2 !
;  1168          1767  2 ! 2. If the volume is mounted /group, the dismounter must:
;  1169          1768  2 !       a. have SYSNAM privilege, or
;  1170          1769  2 !       b. be in the same group with GRPNAM privilege.
;  1171          1770  2 !
;  1172          1771  2 ! 3. If neither, then the dismounter must have the same owner UIC as the
;  1173          1772  2 !    device, or have VOLPRO privilege.
;  1174          1773  2 !
;  1175          1774  2
;  1176          1775  2
;  1177          1776  3 IF NOT (  .FLAGS [DMT$V_ABORT]
;  1178          1777  3       OR .FLAGS [DMT$V_CLUSTER] )       ! if normal dismount
;  1179          1778  2 THEN
;  1180          1779  3     BEGIN
;  1181          1780  3     JIB = .SCH$GL_CURPCB [PCB$L_JIB];   ! get the JIB of current process
;  1182          1781  3     LIST_HEAD = JIB [JIB$L_MTLF[];      ! point to our job-wide mount list head
;  1183          1782  3     LOCK_IODB ();
;  1184          1783  3     MTL = SEARCH_MOUNT ( .LIST_HEAD, .UCB ); ! see if volume is privately mounted
;  1185          1784  3     UNLOCK_IODB ();
;  1186          1785  4     IF ( .MTL NEQ 0 )
;  1187          1786  3     THEN                                ! normal dismount of a privately mounted volume
;  1188          1787  3         RETURN 1;                       ! return immediately
;  1189          1788  2     END;
;  1190          1789  2
;  1191          1790  2 PRIVILEGE_MASK = CTL$GL_PHD [PHD$Q_PRIVMSK]; ! Get process privilege mask
;  1192          1791  2 VCB = .UCB [UCB$L_VCB];                 ! get VCB
;  1193          1792  2 LIST_HEAD = IOC$GQ_MOUNTLST[0];         ! search system wide list
;  1194          1793  2
;  1195          1794  2 LOCK_IODB ();                           ! lock I/O database
;  1196          1795  2 MTL = SEARCH_MOUNT ( .LIST_HEAD, .UCB );
;  1197          1796  2
;  1198          1797  2 IF .MTL EQL 0                          ! if not mounted system or group
;  1199          1798  2 THEN                                   ! check proper privilege
;  1200          1799  2
;  1201          1800  3     BEGIN
;  1202          1801  3     ORB = .UCB[UCB$L_ORB];              ! get ORB address
;  1203          1802  3     UNLOCK_IODB ();                     ! unlock I/O database
```

```
;  1204          1803  3     UIC = .SCH$GL_CURPCB [PCB$L_UIC];    ! get process UIC
;  1205          1804  4     IF   ( .FLAGS [DMT$V_ABORT] )        ! check privilege
;  1206          1805  4     AND  ( .UIC NEQ .ORB [ORB$L_OWNER] )
;  1207          1806  4     AND  ( NOT .PRIVILEGE_MASK [PRV$V_VOLPRO] )
;  1208          1807  3         THEN
;  1209          1808  3         RETURN SS$_NOPRIV;               ! no privilege to dismount /abort
;  1210          1809  3     END
;  1211          1810  3
;  1212          1811  2 ELSE
;  1213          1812  2     !
;  1214          1813  2     ! If this is a disk mounted /GROUP, the dismounter must be in the group
;  1215          1814  2     ! that mounted the disk, or have SYSNAM privilege.
;  1216          1815  2     !
;  1217          1816  3     BEGIN
;  1218          1817  3
;  1219          1818  3     IF .VCB[VCB$V_GROUP]                 ! volume mounted /group
;  1220          1819  3     THEN
;  1221          1820  4         BEGIN
;  1222          1821  5         IF NOT (
;  1223          1822  5             .PRIVILEGE_MASK[PRV$V_SYSNAM]
;  1224          1823  6             OR (.PRIVILEGE_MASK[PRV$V_GRPNAM]
;  1225          1824  7                 AND (IF .MTL[MTL$L_LOGNAME] NEQ 0
;  1226          1825  7                     THEN
;  1227          1826  8                         (LOCAL
;  1228          1827  8                             LNMB     : REF BBLOCK,
;  1229          1828  8                             LNMTH    : REF BBLOCK,
;  1230          1829  8                             ORB      : REF BBLOCK,
;  1231          1830  8                             FULL_UIC : LONG;
;  1232          1831  8                         LOCK_LNM();
;  1233          1832  8                         LNMB     = .MTL[MTL$L_LOGNAME];
;  1234          1833  8                         LNMTH    = .LNMB[LNMB$L_TABLE];
;  1235          1834  8                         ORB      = .LNMTH[LNMTH$L_ORB];
;  1236          1835  8                         FULL_UIC = .ORB[ORB$L_OWNER];
;  1237          1836  8                         UNLOCK_LNM();
;  1238          1837  8                         .FULL_UIC <16,16>
;  1239          1838  8                                 EQL .SCH$GL_CURPCB[PCB$W_GRP]
;  1240          1839  8                         )
;  1241          1840  7                     ELSE 1)
;  1242          1841  6                 )
;  1243          1842  5             )
;  1244          1843  4         THEN
;  1245          1844  5             BEGIN
;  1246          1845  5             UNLOCK_IODB ();
;  1247          1846  5             RETURN (SS$_NOGRPNAM);
;  1248          1847  4             END;
;  1249          1848  4         END
;  1250          1849  3     ELSE
;  1251          1850  3         IF .VCB[VCB$V_SYSTEM]            ! volume mounted /system
;  1252          1851  3         THEN
;  1253          1852  3             IF NOT .PRIVILEGE_MASK[PRV$V_SYSNAM]
;  1254          1853  3             THEN
;  1255          1854  4                 BEGIN
;  1256          1855  4                 UNLOCK_IODB ();
;  1257          1856  4                 RETURN (SS$_NOSYSNAM);
;  1258          1857  3                 END;
;  1259          1858  3
;  1260          1859  3     UNLOCK_IODB ();                      ! unlock I/O database
```

```
; 1261        1860 3      IF .UCB EQL .EXE$GL_SYSUCB              ! check for dismount of system device
; 1262        1861 3      THEN
; 1263        1862 3          RETURN (DISM$_SYSDEV);
; 1264        1863 3
; 1265        1864 2      END;                                    ! mounted /GROUP or /SYSTEM checks
; 1266        1865 2
; 1267        1866 2
; 1268        1867 2  RETURN 1;
; 1269        1868 2
; 1270        1869 1  END;                                        ! routine CHECK_PRIV


                                                            .EXTRN  LOCK_LNM, UNLOCK_LNM

                                     01FC 00000 CHECK_PRIV:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7,R8                    ; 1690
                  58      0000G  CF  9E 00002               MOVAB   UNLOCK_IODB, R8                              ;
                  57 00000000G   00  9E 00007               MOVAB   SCH$GL_CURPCB, R7                            ;
         2C       08         AC  02  E0 0000E               BBS     #2, FLAGS, 1$                                ; 1776
         27       08         AC  03  E0 00013               BBS     #3, FLAGS, 1$                                ; 1777
                  50             67  D0 00018               MOVL    SCH$GL_CURPCB, R0                            ; 1780
                  50      0080   C0  D0 0001B               MOVL    128(R0), JIB                                 ;
                  55             50  D0 00020               MOVL    JIB, LIST_HEAD                               ; 1781
            0000G CF             00  FB 00023               CALLS   #0, LOCK_IODB                                ; 1782
                          04     AC  DD 00028               PUSHL   UCB                                          ; 1783
                                 55  DD 0002B               PUSHL   LIST_HEAD                                    ;
            FE79  CF             02  FB 0002D               CALLS   #2, SEARCH_MOUNT                             ;
                  53             50  D0 00032               MOVL    R0, MTL                                      ;
                  68             00  FB 00035               CALLS   #0, UNLOCK_IODB                              ; 1784
                                 53  D5 00038               TSTL    MTL                                          ; 1785
                                 03  13 0003A               BEQL    1$                                           ;
                               00B2  31 0003C               BRW     6$                                           ;
                  54 00000000G   00  D0 0003F  1$:          MOVL    CTL$GL_PHD, PRIVILEGE_MASK                   ; 1790
                  56      04     AC  D0 00046               MOVL    UCB, R6                                      ; 1791
                  52      34     A6  D0 0004A               MOVL    52(R6), VCB                                  ;
                  55 00000000G   00  9E 0004E               MOVAB   IOC$GQ_MOUNTLST, LIST_HEAD                   ; 1792
            0000G CF             00  FB 00055               CALLS   #0, LOCK_IODB                                ; 1794
                  7E             55  7D 0005A               MOVQ    LIST_HEAD, -(SP)                             ; 1795
            FE49  CF             02  FB 0005D               CALLS   #2, SEARCH_MOUNT                             ;
                  53             50  D0 00062               MOVL    R0, MTL                                      ;
                                 21  12 00065               BNEQ    2$                                           ; 1797
                  55      1C     A6  D0 00067               MOVL    28(R6), ORB                                  ; 1801
                  68             00  FB 0006B               CALLS   #0, UNLOCK_IODB                              ; 1802
                  50             67  D0 0006E               MOVL    SCH$GL_CURPCB, R0                            ; 1803
                  50      00BC   C0  D0 00071               MOVL    188(R0), UIC                                 ;
         76       08         AC  02  E1 00076               BBC     #2, FLAGS, 6$                                ; 1804
                  65             50  D1 0007B               CMPL    UIC, (ORB)                                   ; 1805
                                 71  13 0007E               BEQL    6$                                           ;
         6D       64             15  E0 00080               BBS     #21, (PRIVILEGE_MASK), 6$                    ; 1806
                  50             24  D0 00084               MOVL    #36, R0                                      ; 1808
                                     04 00087               RET                                                  ;
         3E       0B         A2  06  E1 00088  2$:          BBC     #6, 11(VCB), 4$                              ; 1818
         4C       64             02  E0 0008D               BBS     #2, (PRIVILEGE_MASK), 5$                     ; 1822
         2D       64             03  E1 00091               BBC     #3, (PRIVILEGE_MASK), 3$                     ; 1823
                          10     A3  D5 00095               TSTL    16(MTL)                                      ; 1824
                                 43  13 00098               BEQL    5$                                           ;
```

```
                        0000G  CF          00  FB 0009A          CALLS   #0, LOCK_LNM                          ; 1831
                               50      10   A3  D0 0009F          MOVL    16(MTL), LNMB                        ; 1832
                               50      0C   A0  D0 000A3          MOVL    12(LNMB), LNMTH                      ; 1833
                               50      05   A0  D0 000A7          MOVL    5(LNMTH), ORB                        ; 1834
                               52           60  D0 000AB          MOVL    (ORB), FULL_UIC                      ; 1835
                        0000G  CF          00  FB 000AE          CALLS   #0, UNLOCK_LNM                       ; 1836
                               50           67  D0 000B3          MOVL    SCH$GL_CURPCB, R0                    ; 1838
                               51      00BE C0  3C 000B6          MOVZWL  190(R0), R1                          ;
          51        52         10           10  ED 000BB          CMPZV   #16, #16, FULL_UIC, R1               ;
                                          1B  13 000C0          BEQL    5$                                   ;
                               68           00  FB 000C2 3$:      CALLS   #0, UNLOCK_IODB                      ; 1845
                               50      281C 8F  3C 000C5          MOVZWL  #10268, R0                           ; 1846
                                               04 000CA          RET                                          ;
                                       0B  A2  95 000CB 4$:      TSTB    11(VCB)                              ; 1850
                                           0D  18 000CE          BGEQ    5$                                   ;
                09                     64       02  E0 000D0          BBS     #2, (PRIVILEGE_MASK), 5$     ; 1852
                                       68       00  FB 000D4          CALLS   #0, UNLOCK_IODB              ; 1855
                                       50  2814 8F  3C 000D7          MOVZWL  #10260, R0                   ; 1856
                                               04 000DC          RET                                  ;
                                       68       00  FB 000DD 5$:      CALLS   #0, UNLOCK_IODB              ; 1859
                    00000000G          00       56  D1 000E0          CMPL    R6, EXE$GL_SYSUCB            ; 1860
                                               08  12 000E7          BNEQ    6$                           ;
                                       50 00738014 8F  D0 000E9          MOVL    #7569428, R0                 ; 1862
                                               04 000F0          RET                                  ;
                                       50           01  D0 000F1 6$:      MOVL    #1, R0                       ; 1867
                                                   04 000F4          RET                                  ; 1869

; Routine Size:  245 bytes,    Routine Base:  Z$DISMOUNT + 04DF

; 1271            1870  1
; 1272            1871  1
```

```
;  1274     1872  1
;  1275     1873  1 ROUTINE DISMOUNT_CLUSTER (DEV_NAME, FLAGS) =
;  1276     1874  1
;  1277     1875  1 !+
;  1278     1876  1 !
;  1279     1877  1 ! FUNCTIONAL DESCRIPTION:
;  1280     1878  1 !
;  1281     1879  1 !       This routine performs the cluster-wide dismount operation.
;  1282     1880  1 !       It calls another routine to create a cluster-dismount packet
;  1283     1881  1 !       and then sends this dismount request to other nodes in the
;  1284     1882  1 !       cluster.
;  1285     1883  1 !
;  1286     1884  1 ! CALLING SEQUENCE:
;  1287     1885  1 !
;  1288     1886  1 !       DISMOUNT_CLUSTER (ARG1,ARG2)
;  1289     1887  1 !
;  1290     1888  1 ! INPUTS:
;  1291     1889  1 !
;  1292     1890  1 !       ARG1    : Address of the device descriptor
;  1293     1891  1 !       ARG2    : A longword bit mask
;  1294     1892  1 !
;  1295     1893  1 ! OUTPUTS:
;  1296     1894  1 !
;  1297     1895  1 !       None.
;  1298     1896  1 !
;  1299     1897  1 ! IMPLICIT INPUTS:
;  1300     1898  1 !
;  1301     1899  1 !       None.
;  1302     1900  1 !
;  1303     1901  1 ! OUTPUT PARAMETERS:
;  1304     1902  1 !
;  1305     1903  1 !       1         : If success
;  1306     1904  1 !       Otherwise : Status from comm primitive.
;  1307     1905  1 !
;  1308     1906  1 ! IMPLICIT OUTPUTS:
;  1309     1907  1 !
;  1310     1908  1 !       None.
;  1311     1909  1 !
;  1312     1910  1 ! ROUTINE VALUE:
;  1313     1911  1 !
;  1314     1912  1 !       None.
;  1315     1913  1 !
;  1316     1914  1 ! SIDE EFFECTS:
;  1317     1915  1 !
;  1318     1916  1 !       The dismount request is sent to other nodes in the cluster.
;  1319     1917  1 !
;  1320     1918  1 !-
;  1321     1919  1
;  1322     1920  1
;  1323     1921  2 BEGIN                                           ! Start of DISMOUNT_CLUSTER
;  1324     1922  2
;  1325     1923  2 MACRO   ITEM_LENG =  0,0,16,0%;                 ! Define item list offsets
;  1326     1924  2 MACRO   ITEM_CODE =  2,0,16,0%;
;  1327     1925  2 MACRO   ITEM_ADDR =  4,0,32,0%;
;  1328     1926  2 MACRO   ITEM_LADR =  8,0,32,0%;
;  1329     1927  2 MACRO   ITEM_STOP = 12,0,32,0%;                 ! Item list stopper
;  1330     1928  2 LITERAL ITEM_SIZE = 16;
```

```
;  1331          1929  2
;  1332          1930  2 LITERAL BUF_SIZE  = DSC$K_S_BLN + NAMEBUF_LEN + 4; ! Define cluster-dismount buffer size
;  1333          1931  2
;  1334          1932  2
;  1335          1933  2 MAP
;  1336          1934  2         DEV_NAME          : REF BBLOCK,
;  1337          1935  2         FLAGS             : BBLOCK;
;  1338          1936  2
;  1339          1937  2 EXTERNAL ROUTINE
;  1340          1938  2         IN_CLUSTER,
;  1341          1939  2         SEND_CLUSTER;
;  1342          1940  2
;  1343          1941  2 LOCAL
;  1344          1942  2         STATUS,
;  1345          1943  2         LENGTH,
;  1346          1944  2         BUFFER            : BBLOCK [BUF_SIZE],    ! Buffer area
;  1347          1945  2
;  1348          1946  2         ITEM              : BBLOCK [ITEM_SIZE],   ! Item list for $GETDVI
;  1349          1947  2         FUL_DEV_DSC       : BBLOCK[DSC$K_S_BLN],  ! Descriptor for full device name
;  1350          1948  2         FUL_DEV_STR       : VECTOR[NAMEBUF_LEN,BYTE]; ! Full device name
;  1351          1949  2
;  1352          1950  2
;  1353          1951  3 IF ( NOT (.FLAGS [DMT$V_CLUSTER] ))                 ! If not /cluster
;  1354          1952  3 OR ( NOT  .CLUSTER_DEVICE )                        ! or not a cluster device
;  1355          1953  3 OR NOT ( STATUS = IN_CLUSTER() )                   ! or not in a cluster environment
;  1356          1954  2 THEN                                               ! return immediately
;  1357          1955  2     RETURN 1;
;  1358          1956  2
;  1359          1957  2 FLAGS [DMT$V_CLUSTER] = 0;                         ! Clear cluster-wide flag
;  1360          1958  2 LENGTH = 0;                                        ! Initialize work area
;  1361          1959  2 CH$FILL (0, BUF_SIZE, BUFFER);                     ! Zero buffer area
;  1362          1960  2
;  1363          1961  2 ITEM [ITEM_LENG] = NAMEBUF_LEN;                    ! Set up item descriptor to
;  1364          1962  2 ITEM [ITEM_CODE] = DVI$_FULLDEVNAM;                ! get full device name
;  1365          1963  2 ITEM [ITEM_ADDR] = FUL_DEV_STR;                    ! ...
;  1366          1964  2 ITEM [ITEM_LADR] = LENGTH;
;  1367          1965  2 ITEM [ITEM_STOP] = 0;                             ! End of item list
;  1368          1966  2
;  1369          1967  2 !
;  1370          1968  2 ! Since the dismount request will be sent to other nodes in the cluster, we
;  1371          1969  2 ! must use the full device name. Obtain the full device name.
;  1372          1970  2 !
;  1373        P 1971  2 STATUS = $GETDVIW ( EFN     = MOUNT_EFN,
;  1374        P 1972  2                     DEVNAM  = .DEV_NAME,
;  1375          1973  2                     ITMLST  = ITEM );                ! Get full device name
;  1376          1974  2 IF NOT .STATUS                                      ! If error, return
;  1377          1975  2 THEN
;  1378          1976  2     RETURN .STATUS;
;  1379          1977  2
;  1380          1978  2 FUL_DEV_DSC [DSC$W_LENGTH] = .LENGTH;               ! Create a descriptor for the
;  1381          1979  2 FUL_DEV_DSC [DSC$A_POINTER] = FUL_DEV_STR;         ! full device name
;  1382          1980  2
;  1383          1981  2 STATUS = DISMOUNT_ENCIPHER (FUL_DEV_DSC, .FLAGS, BUFFER, LENGTH); ! Encipher the dismount request
;  1384          1982  2 IF NOT .STATUS                                      ! If error, return
;  1385          1983  2 THEN
;  1386          1984  2     RETURN .STATUS;
;  1387          1985  2
```

```
;  1388          1986  2 STATUS = SEND_CLUSTER (BUFFER, .LENGTH, 0);      ! Broadcast the request
;  1389          1987  2                                                  ! Arg3=0 means a cluster-dismount
;  1390          1988  2
;  1391          1989  2 RETURN .STATUS;
;  1392          1990  2
;  1393          1991  1 END;                                              ! End of DISMOUNT_CLUSTER


                                                                          .EXTRN  IN_CLUSTER, SEND_CLUSTER

                                                 007C 00000 DISMOUNT_CLUSTER:
                                                                          .WORD   Save R2,R3,R4,R5,R6                             ; 1873
                                          5E      98   AE  9E 00002          MOVAB   -104(SP), SP                                    ;
                                    10      08   AC      03  E1 00006          BBC     #3, FLAGS, 1$                           ; 1951
                                                 0B 0000' CF  E9 0000B          BLBC    CLUSTER_DEVICE, 1$                      ; 1952
                                           0000G CF          00  FB 00010          CALLS   #0, IN_CLUSTER                          ; 1953
                                                 56          50  D0 00015          MOVL    R0, STATUS                              ;
                                                 04          56  E8 00018          BLBS    STATUS, 2$                              ;
                                                 50          01  D0 0001B 1$:      MOVL    #1, R0                                  ; 1955
                                                             04  0001E            RET                                             ;
                                                 08   AC      08  8A 0001F 2$:      BICB2   #8, FLAGS                               ; 1957
                                                             6E  D4 00023          CLRL    LENGTH                                  ; 1958
                                    2C      00           6E      00  2C 00025          MOVC5   #0, (SP), #0, #44, BUFFER       ; 1959
                                                      3C     AE      0002A                                                          ;
                                                 2C   AE 00E80020 8F  D0 0002C          MOVL    #15204384, ITEM                 ; 1961
                                                 30   AE      04  AE  9E 00034          MOVAB   FUL_DEV_STR, ITEM+4             ; 1963
                                                 34   AE          6E  9E 00039          MOVAB   LENGTH, ITEM+8                  ; 1964
                                                      38     AE  D4  0003D          CLRL    ITEM+12                         ; 1965
                                                             7E  7C 00040          CLRQ    -(SP)                           ; 1973
                                                             7E  7C 00042          CLRQ    -(SP)                           ;
                                                      3C     AE  9F 00044          PUSHAB  ITEM                            ;
                                                      04     AC  DD 00047          PUSHL   DEV_NAME                        ;
                                                 7E          1A  7D 0004A          MOVQ    #26, -(SP)                      ;
                                           00000000G 00          08  FB 0004D          CALLS   #8, SYS$GETDVIW                 ;
                                                 56          50  D0 00054          MOVL    R0, STATUS                      ;
                                                 2F          56  E9 00057          BLBC    STATUS, 3$                      ; 1974
                                              24 AE          6E  B0 0005A          MOVW    LENGTH, FUL_DEV_DSC             ; 1978
                                              28 AE       04 AE  9E 0005E          MOVAB   FUL_DEV_STR, FUL_DEV_DSC+4      ; 1979
                                                          5E  DD 00063          PUSHL   SP                              ; 1981
                                                      40     AE  9F 00065          PUSHAB  BUFFER                          ;
                                                      08     AC  DD 00068          PUSHL   FLAGS                           ;
                                                      30     AE  9F 0006B          PUSHAB  FUL_DEV_DSC                     ;
                                           0000V CF          04  FB 0006E          CALLS   #4, DISMOUNT_ENCIPHER           ;
                                                 56          50  D0 00073          MOVL    R0, STATUS                      ;
                                                 10          56  E9 00076          BLBC    STATUS, 3$                      ; 1982
                                                             7E  D4 00079          CLRL    -(SP)                           ; 1986
                                                      04     AE  DD 0007B          PUSHL   LENGTH                          ;
                                                      44     AE  9F 0007E          PUSHAB  BUFFER                          ;
                                           0000G CF          03  FB 00081          CALLS   #3, SEND_CLUSTER                ;
                                                 56          50  D0 00086          MOVL    R0, STATUS                      ;
                                                 50          56  D0 00089 3$:      MOVL    STATUS, R0                      ; 1989
                                                             04  0008C            RET                                             ; 1991

; Routine Size:  141 bytes,    Routine Base:  Z$DISMOUNT + 05D4
```

```
; 1394          1992  1
; 1395          1993  1
```

```
;  1397     1994  1
;  1398     1995  1 ROUTINE DISMOUNT_ENCIPHER (DEV_DSC, FLAGS, BUFFER, LENGTH) =
;  1399     1996  1
;  1400     1997  1 !+
;  1401     1998  1 !
;  1402     1999  1 ! FUNCTIONAL DESCRIPTION:
;  1403     2000  1 !
;  1404     2001  1 !       This routine takes the parameters of the dismount request
;  1405     2002  1 !       and enciphers the parameters into a cluster-dismount packet.
;  1406     2003  1 !
;  1407     2004  1 ! CALLING SEQUENCE:
;  1408     2005  1 !
;  1409     2006  1 !       DISMOUNT_ENCIPHER (ARG1,ARG2,ARG3,ARG4)
;  1410     2007  1 !
;  1411     2008  1 ! INPUTS:
;  1412     2009  1 !
;  1413     2010  1 !       ARG1    : Address of the device descriptor
;  1414     2011  1 !       ARG2    : A longword bit mask
;  1415     2012  1 !
;  1416     2013  1 ! OUTPUTS:
;  1417     2014  1 !
;  1418     2015  1 !       ARG3    : Address of the output buffer to receive the
;  1419     2016  1 !                   cluster-dismount packet
;  1420     2017  1 !       ARG4    : Address of a longword to receive the length of
;  1421     2018  1 !                   the output buffer
;  1422     2019  1 !
;  1423     2020  1 ! IMPLICIT INPUTS:
;  1424     2021  1 !
;  1425     2022  1 !       None.
;  1426     2023  1 !
;  1427     2024  1 ! OUTPUT PARAMETERS:
;  1428     2025  1 !
;  1429     2026  1 !       None.
;  1430     2027  1 !
;  1431     2028  1 ! IMPLICIT OUTPUTS:
;  1432     2029  1 !
;  1433     2030  1 !       None.
;  1434     2031  1 !
;  1435     2032  1 ! ROUTINE VALUES:
;  1436     2033  1 !
;  1437     2034  1 !       1               : If successful
;  1438     2035  1 !       SS$_BUFFEROVF   : Insufficient internal buffer space
;  1439     2036  1 !
;  1440     2037  1 ! SIDE EFFECTS:
;  1441     2038  1 !
;  1442     2039  1 !       None.
;  1443     2040  1 !
;  1444     2041  1 !
;  1445     2042  1 ! NOTES:
;  1446     2043  1 !
;  1447     2044  1 ! This encipher routine takes the given device descriptor and turns it
;  1448     2045  1 ! into a cluster-dismount packet of the form:
;  1449     2046  1 !
;  1450     2047  1 !                                   Offset
;  1451     2048  1 !       +----------------+
;  1452     2049  1 !       |     flags      |  0  BUF_FLAGS
;  1453     2050  1 !       +----------------+
```

```
1 !          ! dev descrptor !  4  BUF_DSC
1 !          +----------------+
1 !          !                !  8
1 !          +----------------+
1 !          ! device string  ! 12  BUF_STR
1 !          +----------------+
1 !          !      ...       !
1 !          +----------------+
1 !
1 !          1. This cluster-dismount packet is to be sent to other nodes in
1 !             the cluster and processed by CSP (the Cluster Server Process).
1 !
1 !          2. The address in the device descriptor is "relocated" to be the
1 !             offset from the beginning of the packet (i.e. self-relative).
1 !             This is done for consistency's sake with the cluster-mount packets.
1 !
1 !-
1
1
2 BEGIN                                                ! Start of DISMOUNT_ENCIPHER
2
2 MAP
2          DEV_DSC : REF BBLOCK,
2          BUFFER  : REF BBLOCK;
2
2 LOCAL
2          LOC_DSC : REF BBLOCK;
2
2
2 MACRO BUF_FLAG    =  0,0,32,0%;                     ! Define buffer offsets
2 MACRO BUF_DSC     =  4,0,32,0%;
2 MACRO BUF_STR     = 12,0,32,0%;
2 LITERAL BUF_HDR_LEN = 12;
2
2
3 IF (.DEV_DSC [DSC$W_LENGTH]  GTRU NAMEBUF_LEN)     ! Check if internal buffer large
2 THEN                                                 ! enough
2     RETURN SS$_BUFFEROVF;                            ! If not, return error
2
2 .LENGTH = BUF_HDR_LEN + .DEV_DSC[DSC$W_LENGTH];     ! Compute length of output,
2                                                      ! including parameters
2 BUFFER[BUF_FLAG] = .FLAGS;                           ! Set flags in buffer
2 LOC_DSC = BUFFER[BUF_DSC];                           !
2
2 !
2 ! Copy the device descriptor into the output buffer
2 !
2 CH$COPY (DSC$K_S_BLN,
2          .DEV_DSC,
2          0,
2          DSC$K_S_BLN,
2          BUFFER[BUF_DSC]);
2
2 LOC_DSC[DSC$A_POINTER] = BUF_HDR_LEN;               ! "Relocate" the string pointer
2
2 !
2 ! Copy the device string into the output buffer
```

```
;  1511          2108  2 !
;  1512          2109  2 CH$COPY (.DEV_DSC[DSC$W_LENGTH],
;  1513          2110  2           .DEV_DSC[DSC$A_POINTER],
;  1514          2111  2           0,
;  1515          2112  2           .DEV_DSC[DSC$W_LENGTH],
;  1516          2113  2           BUFFER[BUF_STR]);
;  1517          2114  2
;  1518          2115  2 RETURN 1;
;  1519          2116  1 END;                                    ! End of DISMOUNT_ENCIPHER


                                        01FC 00000 DISMOUNT_ENCIPHER:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8                ; 1995
                          58       04   AC  D0 00002         MOVL    DEV_DSC, R8                     ; 2086
                          20            68  B1 00006         CMPW    (R8), #32                       ;
                                        06  1B 00009         BLEQU   1$                              ;
                          50       0601 8F  3C 0000B         MOVZWL  #1537, R0                       ; 2088
                                        04  00010           RET                                     ;
                      10  BC            68  3C 00011 1$:    MOVZWL  (R8), @LENGTH                   ; 2090
                      10  BC            0C  C0 00015         ADDL2   #12, @LENGTH                    ;
                          57       0C   AC  D0 00019         MOVL    BUFFER, R7                      ; 2092
                          67       08   AC  D0 0001D         MOVL    FLAGS, (R7)                     ;
                          56       04   A7  9E 00021         MOVAB   4(R7), LOC_DSC                  ; 2093
                04  A7            68       08 28 00025      MOVC3   #8, (R8), 4(R7)                 ; 2102
                          04  A6            0C D0 0002A      MOVL    #12, 4(LOC_DSC)                 ; 2104
                0C  A7    04  B8            68 28 0002E      MOVC3   (R8), @4(R8), 12(R7)            ; 2113
                          50                01 D0 00034      MOVL    #1, R0                          ; 2115
                                            04 00037         RET                                     ; 2116

; Routine Size:  56 bytes,    Routine Base:  Z$DISMOUNT + 0661

;  1520          2117  1
```

```
;  1522          2118  1
;  1523          2119  1 ROUTINE DISMOUNT_AUDIT (FLAGS, CHANNEL, UCB, MTL): NOVALUE =
;  1524          2120  1
;  1525          2121  1 !+
;  1526          2122  1 !
;  1527          2123  1 ! FUNCTIONAL DESCRIPTION:
;  1528          2124  1 !
;  1529          2125  1 !       This routine determines if a security auditing packet should
;  1530          2126  1 !       be logged. If so, it creates the security auditing packet and
;  1531          2127  1 !       logs the event.
;  1532          2128  1 !
;  1533          2129  1 ! CALLING SEQUENCE:
;  1534          2130  1 !
;  1535          2131  1 !       DISMOUNT_AUDIT (ARG1,ARG2,ARG3)
;  1536          2132  1 !
;  1537          2133  1 ! INPUTS:
;  1538          2134  1 !
;  1539          2135  1 !       ARG1    : A longword bit mask
;  1540          2136  1 !       ARG2    : The channel number of the channel assigned to the device
;  1541          2137  1 !       ARG3    : Address of the desired UCB
;  1542          2138  1 !       ARG4    : Address of the mount list entry
;  1543          2139  1 !
;  1544          2140  1 ! OUTPUTS:
;  1545          2141  1 !
;  1546          2142  1 !       None.
;  1547          2143  1 !
;  1548          2144  1 ! IMPLICIT INPUTS:
;  1549          2145  1 !
;  1550          2146  1 !       None.
;  1551          2147  1 !
;  1552          2148  1 ! OUTPUT PARAMETERS:
;  1553          2149  1 !
;  1554          2150  1 !       None.
;  1555          2151  1 !
;  1556          2152  1 ! IMPLICIT OUTPUTS:
;  1557          2153  1 !
;  1558          2154  1 !       None.
;  1559          2155  1 !
;  1560          2156  1 ! ROUTINE VALUES:
;  1561          2157  1 !
;  1562          2158  1 !       None.
;  1563          2159  1 !
;  1564          2160  1 ! SIDE EFFECTS:
;  1565          2161  1 !
;  1566          2162  1 !       If security auditing is enabled, then create a security auditing
;  1567          2163  1 !       packet and log this event.
;  1568          2164  1 !
;  1569          2165  1 !-
;  1570          2166  1
;  1571          2167  1
;  1572          2168  2 BEGIN                                           ! Start of DISMOUNT_AUDIT
;  1573          2169  2
;  1574          2170  2 MAP
;  1575          2171  2         FLAGS   : BBLOCK,
;  1576          2172  2         UCB     : REF BBLOCK,
;  1577          2173  2         MTL     : REF BBLOCK;
;  1578          2174  2
```

```
;  1579     2175  2 BUILTIN
;  1580     2176  2         CALLG;
;  1581     2177  2
;  1582     2178  2 EXTERNAL
;  1583     2179  2         SCH$GL_CURPCB    : REF BBLOCK ADDRESSING_MODE (GENERAL),
;  1584     2180  2                                                    ! address of current PCB
;  1585     2181  2         NSA$GR_ALARMVEC : BBLOCK ADDRESSING_MODE (GENERAL),
;  1586     2182  2                                                    ! Alarm enable bit vector
;  1587     2183  2         NSA$GR_JOURNVEC : BBLOCK ADDRESSING_MODE (GENERAL);
;  1588     2184  2                                                    ! Journal enable bit vector
;  1589     2185  2
;  1590     2186  2 LINKAGE
;  1591     2187  2         ARGLST_IMGNAM    = JSB (REGISTER = 2;) :
;  1592     2188  2                            NOPRESERVE (0,1)
;  1593     2189  2                            NOTUSED (3,4,5,6,7,8,9,10,11);
;  1594     2190  2
;  1595     2191  2 EXTERNAL ROUTINE
;  1596     2192  2         NSA$EVENT_AUDIT : ADDRESSING_MODE (GENERAL),
;  1597     2193  2                                                    ! Security auditing routine
;  1598     2194  2         NSA$ARGLST_IMGNAM : ARGLST_IMGNAM ADDRESSING_MODE (GENERAL);
;  1599     2195  2                                                    ! Insert IMGNAM into ARGLST
;  1600     2196  2
;  1601     2197  2 PSECT
;  1602     2198  2         PLIT    = Z$DISMOUNT;                      ! Define PLITS in Z$DISMOUNT psect to
;  1603     2199  2                                                    ! avoid truncation errors
;  1604     2200  2 LOCAL
;  1605     2201  2         VCB     : REF BBLOCK,                      ! Address of the VCB
;  1606     2202  2         RVT     : REF BBLOCK,                      ! Address of the RVT
;  1607     2203  2         LNMB    : REF BBLOCK,                      ! Address of the LNMB
;  1608     2204  2         ARGLIST : BBLOCK[NSA$K_ARG3_LENGTH],
;  1609     2205  2                                                    ! Security auditing argument list
;  1610     2206  2         DEV_LEN : INITIAL (0),                     ! Length of full device name
;  1611     2207  2         DEV_STR : VECTOR [LOG$C_NAMLENGTH],        ! Full device name buffer
;  1612     2208  2         ITEM_LIST : BBLOCK [12+4]
;  1613     2209  2                     INITIAL                        ! Item list to get full device name
;  1614     2210  2                     ( WORD (LOG$C_NAMLENGTH),      ! Length of buffer
;  1615     2211  2                       WORD (DVI$_FULLDEVNAM),      ! Get full device name
;  1616     2212  2                       LONG (DEV_STR),              ! Full device name buffer address
;  1617     2213  2                       LONG (DEV_LEN),              ! Length of full device name
;  1618     2214  2                       LONG (0) );                  ! Item list stopper
;  1619     2215  2
;  1620     2216  2
;  1621     2217  3 IF (.SCH$GL_CURPCB [PCB$V_SECAUDIT]
;  1622     2218  3 OR  .NSA$GR_ALARMVEC [NSA$V_EVT_MOUNT]
;  1623     2219  3 OR  .NSA$GR_JOURNVEC [NSA$V_EVT_MOUNT])
;  1624     2220  2 THEN
;  1625     2221  3     BEGIN
;  1626     2222  3
;  1627     2223  3     CH$FILL (0, NSA$K_ARG3_LENGTH, ARGLIST);       ! Zero argument list
;  1628     2224  3
;  1629     2225  3     !
;  1630     2226  3     ! Set up the security auditing argument list header
;  1631     2227  3     !
;  1632     2228  3
;  1633     2229  3     ARGLIST [NSA$L_ARG_COUNT] = ( NSA$K_ARG3_LENGTH/4 ) - 4;
;  1634     2230  3                                                    ! Initialize length of argument list
;  1635     2231  3                                                    ! Less vol-set pkt and arg count
```

```
;  1636          2232  3      ARGLIST [NSA$L_ARG_ID] = NSA$K_RECID_VOL_DMOU;
;  1637          2233                                                  ! Initialize record id as dismount
;  1638          2234         IF .SCH$GL_CURPCB [PCB$V_SECAUDIT]       ! Set up proper flags
;  1639          2235         THEN
;  1640          2236             ARGLIST [NSA$V_ARG_FLAG_MANDY] = 1;  ! Mandatory auditing
;  1641          2237         IF .NSA$GR_ALARMVEC [NSA$V_EVT_MOUNT]
;  1642          2238         THEN
;  1643          2239             ARGLIST [NSA$V_ARG_FLAG_ALARM] = 1;  ! Generate alarm for this record
;  1644          2240         IF .NSA$GR_JOURNVEC [NSA$V_EVT_MOUNT]
;  1645          2241         THEN
;  1646          2242             ARGLIST [NSA$V_ARG_FLAG_JOURN] = 1;  ! Journal this record
;  1647          2243
;  1648          2244         ARGLIST [NSA$B_ARG_PKTNUM] = 5;          ! Initialize number of items
;  1649          2245                                                  ! less vol-set pkt
;  1650          2246
;  1651          2247         !
;  1652          2248         ! Set up the security auditing argument list for dismount
;  1653          2249         !
;  1654          2250
;  1655          2251         ARGLIST [NSA$L_ARG3_DMOUFLG_TM] = NSA$K_ARG_MECH_WORD^16 + NSA$K_PKTTYP_DMOUFLG;
;  1656          2252                                                  ! Note: set mech to word, OPCOM expects it
;  1657          2253         ARGLIST [NSA$L_ARG3_DMOUFLG ] = .FLAGS;  ! Set dismount flags
;  1658          2254
;  1659          2255         NSA$ARGLST_IMGNAM (ARGLIST [NSA$L_ARG3_IMGNAM_TM]); ! Set image name
;  1660          2256
;  1661          2257         ARGLIST [NSA$L_ARG3_DEVNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_DEVNAM;
;  1662    P     2258         $GETDVIW  ( EFN    = MOUNT_EFN,
;  1663    P     2259                     CHAN   = .CHANNEL,
;  1664          2260                     ITMLST = ITEM_LIST );        ! Obtain full device name
;  1665          2261         ARGLIST [NSA$L_ARG3_DEVNAM_SIZ] = .DEV_LEN; ! Set size of full device name
;  1666          2262         ARGLIST [NSA$L_ARG3_DEVNAM_PTR] = DEV_STR;  ! Set full device name buffer address
;  1667          2263
;  1668          2264         ARGLIST [NSA$L_ARG3_LOGNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_LOGNAM;
;  1669          2265         LNMB = .MTL [MTL$L_LOGNAME];             ! Get address of LNM block
;  1670          2266         IF .LNMB NEQ 0                           ! If the LNM block exists
;  1671          2267         THEN
;  1672          2268  4          BEGIN
;  1673          2269  4          ARGLIST [NSA$L_ARG3_LOGNAM_SIZ] = .LNMB [LNMB$T_NAME];  ! Set size of logical name
;  1674          2270  4          ARGLIST [NSA$L_ARG3_LOGNAM_PTR] = LNMB [LNMB$T_NAME]+1; ! Set logical name buffer address
;  1675          2271  4          END
;  1676          2272         ELSE
;  1677          2273  4          BEGIN
;  1678          2274  4          ARGLIST [NSA$L_ARG3_LOGNAM_SIZ] = 0;    ! Set size of logical name as null
;  1679          2275  4          ARGLIST [NSA$L_ARG3_LOGNAM_PTR] = 0;    ! Set logical name buffer address as null
;  1680          2276  4          END;
;  1681          2277
;  1682          2278         ARGLIST [NSA$L_ARG3_VOLNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_VOLNAM;
;  1683          2279         VCB = .UCB [UCB$L_VCB];                  ! Get address of VCB
;  1684          2280         ARGLIST [NSA$L_ARG3_VOLNAM_SIZ] =
;  1685          2281             LABEL_LENGTH (VCB$S_VOLNAME, VCB [VCB$T_VOLNAME]); ! Set size of volume name
;  1686          2282         ARGLIST [NSA$L_ARG3_VOLNAM_PTR] = VCB [VCB$T_VOLNAME]; ! Set volume name buffer address
;  1687          2283
;  1688          2284         !
;  1689          2285         ! If the volume is a member of a volume set, then
;  1690          2286         !    a. increment argument count
;  1691          2287         !    b. increment number of packets
;  1692          2288  3      !    c. set up volume set descriptor
```

```
;  1693          2289  3     !
;  1694          2290  3
;  1695          2291  4     IF (NOT .BBLOCK [UCB [UCB$L_DEVCHAR], DEV$V_FOR])
;  1696          2292  4     AND  (  .VCB [VCB$W_RVN] NEQ 0 )
;  1697          2293  3     THEN
;  1698          2294  4         BEGIN
;  1699          2295  4         ARGLIST [NSA$L_ARG_COUNT] = .ARGLIST [NSA$L_ARG_COUNT] + 3; ! Count vol-set pkt
;  1700          2296  4         ARGLIST [NSA$B_ARG_PKTNUM] = .ARGLIST [NSA$B_ARG_PKTNUM] + 1;
;  1701          2297  4         ARGLIST [NSA$L_ARG3_VOLSNAM_TM] = NSA$K_ARG_MECH_DESCR^16 + NSA$K_PKTTYP_VOLSNAM;
;  1702          2298  4         RVT = .VCB [VCB$L_RVT];
;  1703          2299  4         ARGLIST [NSA$L_ARG3_VOLSNAM_SIZ] =
;  1704          2300  4             LABEL_LENGTH (RVT$S_STRUCNAME, RVT [RVT$T_STRUCNAME]); ! Set size of vol-set name
;  1705          2301  4         ARGLIST [NSA$L_ARG3_VOLSNAM_PTR] = RVT [RVT$T_STRUCNAME];  ! Set vol-set name buffer address
;  1706          2302  3         END;
;  1707          2303  3
;  1708          2304  3     CALLG (ARGLIST, NSA$EVENT_AUDIT);           ! Call event audit routine
;  1709          2305  3
;  1710          2306  2     END;                                        ! End of security auditing block
;  1711          2307  2
;  1712          2308  2 RETURN;                                        ! Back to caller
;  1713          2309  1 END;                                           ! End of DISMOUNT_AUDIT


                                                00699                   .BLKB   3
                                            0040 0069C P.AAC:           .WORD   64                                      ;
                                            00E8 0069E                  .WORD   232                                     ;
                                        00000000 006A0                  .LONG   0                                       ;
                                        00000000 006A4                  .LONG   0                                       ;
                                        00000000 006A8                  .LONG   0                                       ;

                                                                        .EXTRN  NSA$GR_ALARMVEC
                                                                        .EXTRN  NSA$GR_JOURNVEC
                                                                        .EXTRN  NSA$EVENT_AUDIT
                                                                        .EXTRN  NSA$ARGLST_IMGNAM

                                                01FC 00000 DISMOUNT_AUDIT:
                                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8               ; 2119
                                                    58 00000000G    00  9E 00002                MOVAB   NSA$GR_JOURNVEC, R8     ;
                                                    57 00000000G    00  9E 00009                MOVAB   NSA$GR_ALARMVEC, R7     ;
                                                    5E      FEA0    CE  9E 00010                MOVAB   -352(SP), SP            ;
                                                                    7E  D4 00015                CLRL    DEV_LEN                 ; 2168
                                        04    AE        D5  AF      10  28 00017                MOVC3   #16, P.AAC, ITEM_LIST   ; 2214
                                                    08  AE      14  AE  9E 0001D                MOVAB   DEV_STR, ITEM_LIST+4    ; 2168
                                                    0C  AE          6E  9E 00022                MOVAB   DEV_LEN, ITEM_LIST+8    ;
                                                    56 00000000G    00  D0 00026                MOVL    SCH$GL_CURPCB, R6       ; 2217
                                                09      27  A6      03  E0 0002D                BBS     #3, 39(R6), 1$          ;
                                                05          67      01  E0 00032                BBS     #1, NSA$GR_ALARMVEC, 1$ ; 2218
                                                01          68      01  E0 00036                BBS     #1, NSA$GR_JOURNVEC, 1$ ; 2219
                                                                    04  0003A                RET                             ;
                                        0050  8F        00      6E      00  2C 0003B 1$:        MOVC5   #0, (SP), #0, #80, ARGLIST ; 2223
                                                                B0  AD      00042               ;                               ;
                                                    B0  AD          10  D0 00044                MOVL    #16, ARGLIST            ; 2229
                                                    B4  AD  00020008    8F  D0 00048            MOVL    #131080, ARGLIST+4      ; 2232
                                                04      27  A6          03  E1 00050            BBC     #3, 39(R6), 2$          ; 2234
                                                    B8  AD              04  88 00055            BISB2   #4, ARGLIST+8           ; 2236
                                                04          67          01  E1 00059 2$:        BBC     #1, NSA$GR_ALARMVEC, 3$ ; 2237
```

```
                B8   AD                    01  88 0005D                BISB2   #1, ARGLIST+8                                   ; 2239
          04    68                         01  E1 00061 3$:           BBC     #1, NSA$GR_JOURNVEC, 4$                         ; 2240
                B8   AD                    02  88 00065                BISB2   #2, ARGLIST+8                                   ; 2242
                B9   AD                    05  90 00069 4$:           MOVB    #5, ARGLIST+9                                   ; 2244
                BC   AD 0001000F           8F  D0 0006D                MOVL    #65551, ARGLIST+12                              ; 2251
                C0   AD          04        AC  D0 00075                MOVL    FLAGS, ARGLIST+16                               ; 2253
                     52          C4        AD  9E 0007A                MOVAB   ARGLIST+20, R2                                  ; 2255
                           00000000G        00  16 0007E                JSB     NSA$ARGLST_IMGNAM                               ;
                D0   AD 00040005           8F  D0 00084                MOVL    #262149, ARGLIST+32                             ; 2257
                                           7E  7C 0008C                CLRQ    -(SP)                                           ; 2260
                                           7E  7C 0008E                CLRQ    -(SP)                                           ;
                                    14     AE  9F 00090                PUSHAB  ITEM_LIST                                       ;
                                           7E  D4 00093                CLRL    -(SP)                                           ;
                                    08     AC  DD 00095                PUSHL   CHANNEL                                         ;
                                           1A  DD 00098                PUSHL   #26                                             ;
          00000000G 00                     08  FB 0009A                CALLS   #8, SYS$GETDVIW                                 ;
                D4   AD                    6E  D0 000A1                MOVL    DEV_LEN, ARGLIST+36                             ; 2261
                D8   AD          14        AE  9E 000A5                MOVAB   DEV_STR, ARGLIST+40                             ; 2262
                DC   AD 00040006           8F  D0 000AA                MOVL    #262150, ARGLIST+44                             ; 2264
                     50          10        AC  D0 000B2                MOVL    MTL, R0                                         ; 2265
                     50          10        A0  D0 000B6                MOVL    16(R0), LNMB                                    ;
                                           0C  13 000BA                BEQL    5$                                              ; 2266
                E0   AD          11        A0  9A 000BC                MOVZBL  17(LNMB), ARGLIST+48                            ; 2269
                E4   AD          12        A0  9E 000C1                MOVAB   18(R0), ARGLIST+52                              ; 2270
                                           03  11 000C6                BRB     6$                                              ; 2266
                                    E0     AD  7C 000C8 5$:           CLRQ    ARGLIST+48                                      ; 2274
                E8   AD 00040007           8F  D0 000CB 6$:           MOVL    #262151, ARGLIST+56                             ; 2278
                     52          0C        AC  D0 000D3                MOVL    UCB, R2                                         ; 2279
                     53          34        A2  D0 000D7                MOVL    52(R2), VCB                                     ;
                                    14     A3  9F 000DB                PUSHAB  20(VCB)                                         ; 2281
                                           0C  DD 000DE                PUSHL   #12                                             ;
                0000V CF                   02  FB 000E0                CALLS   #2, LABEL_LENGTH                                ;
                  EC  AD                   50  D0 000E5                MOVL    R0, ARGLIST+60                                  ;
                  F0  AD          14       A3  9E 000E9                MOVAB   20(VCB), ARGLIST+64                             ; 2282
                      2B          3B       A2  E8 000EE                BLBS    59(R2), 7$                                      ; 2291
                                  0E       A3  B5 000F2                TSTW    14(VCB)                                         ; 2292
                                           26  13 000F5                BEQL    7$                                              ;
                B0   AD                    03  C0 000F7                ADDL2   #3, ARGLIST                                     ; 2295
                                    B9     AD  96 000FB                INCB    ARGLIST+9                                       ; 2296
                F4   AD 00040008           8F  D0 000FE                MOVL    #262152, ARGLIST+68                             ; 2297
                     52          20        A3  D0 00106                MOVL    32(VCB), RVT                                    ; 2298
                                 0C        A2  9F 0010A                PUSHAB  12(RVT)                                         ; 2300
                                           0C  DD 0010D                PUSHL   #12                                             ;
                0000V CF                   02  FB 0010F                CALLS   #2, LABEL_LENGTH                                ;
                  F8  AD                   50  D0 00114                MOVL    R0, ARGLIST+72                                  ;
                  FC  AD          0C       A2  9E 00118                MOVAB   12(RVT), ARGLIST+76                             ; 2301
          00000000G 00          B0   AD    FA  0011D 7$:              CALLG   ARGLIST, NSA$EVENT_AUDIT                        ; 2304
                                     04    00125                       RET                                                     ; 2309

; Routine Size:  294 bytes,    Routine Base:  Z$DISMOUNT + 06AC

; 1714          2310  1
```

```
;  1716          2311  1
;  1717          2312  1 ROUTINE LABEL_LENGTH (STR_LENGTH, STR_TEXT) =
;  1718          2313  1
;  1719          2314  1 !++
;  1720          2315  1 !
;  1721          2316  1 ! FUNCTIONAL DESCRIPTION:
;  1722          2317  1 !
;  1723          2318  1 !       This routine will return the length of a given string.
;  1724          2319  1 !       Trailing blanks at the end of the string are not counted
;  1725          2320  1 !       as part of the string.
;  1726          2321  1 !
;  1727          2322  1 ! CALLING SEQUENCE:
;  1728          2323  1 !
;  1729          2324  1 !       LABEL_LENGTH (ARG1, ARG2)
;  1730          2325  1 !
;  1731          2326  1 ! INPUT PARAMETERS:
;  1732          2327  1 !
;  1733          2328  1 !       ARG1    : Input string length
;  1734          2329  1 !       ARG2    : Input string address
;  1735          2330  1 !
;  1736          2331  1 ! IMPLICIT INPUTS:
;  1737          2332  1 !
;  1738          2333  1 !       None.
;  1739          2334  1 !
;  1740          2335  1 ! OUTPUT PARAMETERS:
;  1741          2336  1 !
;  1742          2337  1 !       None.
;  1743          2338  1 !
;  1744          2339  1 ! IMPLICIT OUTPUTS:
;  1745          2340  1 !
;  1746          2341  1 !       None.
;  1747          2342  1 !
;  1748          2343  1 ! ROUTINE VALUE:
;  1749          2344  1 !
;  1750          2345  1 !       None.
;  1751          2346  1 !
;  1752          2347  1 ! SIDE EFFECTS:
;  1753          2348  1 !
;  1754          2349  1 !       None.
;  1755          2350  1 !
;  1756          2351  1 !--
;  1757          2352  1
;  1758          2353  2 BEGIN
;  1759          2354  2
;  1760          2355  2 MAP
;  1761          2356  2         STR_TEXT        : REF VECTOR [,BYTE];   ! Input string
;  1762          2357  2
;  1763          2358  2 LOCAL
;  1764          2359  2         PTR             : LONG;                 ! Pointer to current char.
;  1765          2360  2
;  1766          2361  2 ! Starting at the end of the string, decrement the string length
;  1767          2362  2 ! until a nonblank character is found, or the beginning of the string
;  1768          2363  2 ! is encountered.
;  1769          2364  2 !
;  1770          2365  2
;  1771          2366  2 PTR = .STR_LENGTH;
;  1772          2367  2 WHILE (.PTR GTR 0) AND (.STR_TEXT [.PTR-1] EQL %ASCII' ') DO
```

```
;  1773          2368  2      PTR = .PTR - 1;
;  1774          2369  2
;  1775          2370  3 RETURN (.PTR)
;  1776          2371  1 END;


                                        0000 00000 LABEL_LENGTH:
                                                   .WORD   Save nothing                          ; 2312
                              51        04   AC  D0 00002       MOVL    STR_LENGTH, PTR         ; 2366
                                             0F  15 00006 1$:    BLEQ    2$                      ; 2367
                        50    51        08   AC  C1 00008       ADDL3   STR_TEXT, PTR, R0       ;
                              20        FF   A0  91 0000D       CMPB    -1(R0), #32             ;
                                             04  12 00011       BNEQ    2$                      ;
                                             51  D7 00013       DECL    PTR                     ; 2368
                                             EF  11 00015       BRB     1$                      ;
                              50             51  D0 00017 2$:    MOVL    PTR, R0                 ; 2370
                                                 04 0001A       RET                             ; 2371

; Routine Size:  27 bytes,    Routine Base:  Z$DISMOUNT + 07D2

;  1777          2372  1
;  1778          2373  1 END
;  1779          2374  0 ELUDOM



;                                PSECT SUMMARY

;      Name                      Bytes                         Attributes

;  $GLOBAL$                         4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  Z$DISMOUNT                    2029  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                              Library Statistics

;                                      -------- Symbols --------      Pages       Processing
;      File                            Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]LIB.L32;1        18619       121         0        1000        00:01.8


;                                COMMAND QUALIFIERS
```

```
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:DISMOU/OBJ=OBJ$:DISMOU MSRC$:DISMOU/UPDATE=(ENH$:DISMOU)

; Size:          1962 code + 71 data bytes
; Run Time:        00:47.9
; Elapsed Time:    01:52.1
; Lines/CPU Min:    2974
; Lexemes/CPU-Min: 23902
; Memory Used:  226 pages
; Compilation Complete
```