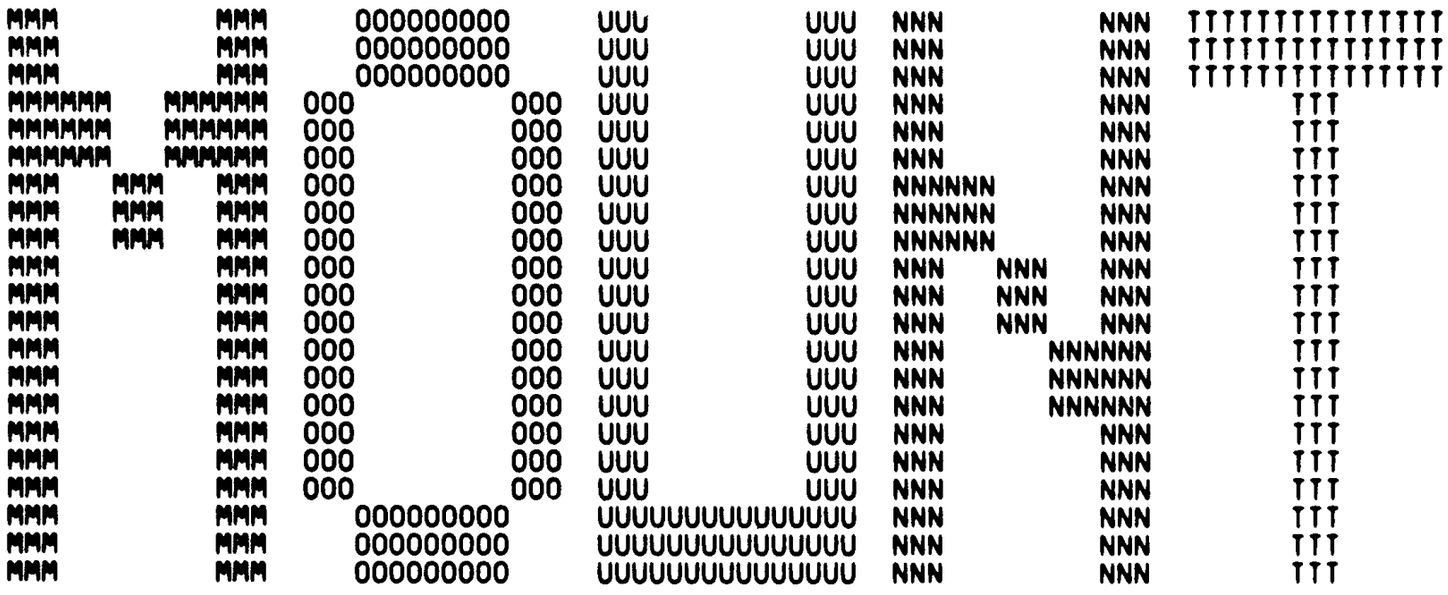
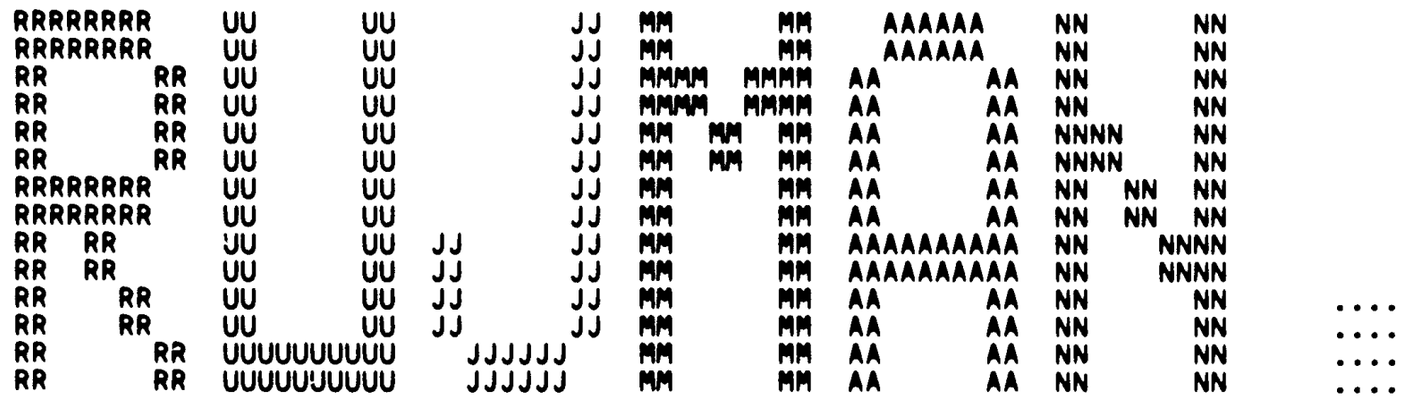

MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMM MMM 000000000 UUU UUU NNN NNN TTTTTTTTTTTTTTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMMMMMM MMMMMMM 000 000 UUU UUU NNN NNN TTT
MMM MMM MMM 000 000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


RRRRRRRR UU UU JJ MM MM AAAAAA NN NN
RRRRRRRR UU UU JJ MM MM AAAAAA NN NN
RR RR UU UU JJ MMMM MMMM AA AA NN NN
RR RR UU UU JJ MMMM MMMM AA AA NN NN
RR RR UU UU JJ MM MM MM AA AA NNNN NN
RR RR UU UU JJ MM MM MM AA AA NNNN NN
RRRRRRRR UU UU JJ MM MM AA AA NN NN NN
RRRRRRRR UU UU JJ MM MM AA AA NN NN NN
RR RR UU UU JJ JJ MM MM AAAAAAAAAA NN NNNN
RR RR UU UU JJ JJ MM MM AAAAAAAAAA NN NNNN
RR RR UU UU JJ JJ MM MM AA AA NN NN
RR RR UU UU JJ JJ MM MM AA AA NN NN
RR RR UUUUUUUUUU JJJJJJ MM MM AA AA NN NN
RR RR UUUUUUUUUU JJJJJJ MM MM AA AA NN NN
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 RUJMAN (
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 ADDRESSING_MODE (EXTERNAL=GENERAL),
;     4          0004  0                 IDENT = 'V04-000'
;     5          0005  0                 ) =
;     6          0006  1 BEGIN
;     7          0007  1
;     8          0008  1 !
;     9          0009  1 !*****************************************************************************
;    10          0010  1 !*                                                                           *
;    11          0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    12          0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    13          0013  1 !*  ALL RIGHTS RESERVED.                                                      *
;    14          0014  1 !*                                                                           *
;    15          0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    16          0016  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    17          0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    18          0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    19          0019  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    20          0020  1 !*  TRANSFERRED.                                                              *
;    21          0021  1 !*                                                                           *
;    22          0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    23          0023  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    24          0024  1 !*  CORPORATION.                                                              *
;    25          0025  1 !*                                                                           *
;    26          0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    27          0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*                                                                           *
;    30          0030  1 !*****************************************************************************
;    31          0031  1
;    32          0032  1 !++
;    33          0033  1 !
;    34          0034  1 ! FACILITY:  MOUNT Utility Structure Level 2
;    35          0035  1 !
;    36          0036  1 ! ABSTRACT:
;    37          0037  1 !
;    38          0038  1 !       This module contains those routine necessary to handle the creation
;    39          0039  1 !       and manipulation of Recovery Unit Journals (RUJ) on Files-11 ODS-2
;    40          0040  1 !       disk volumes.
;    41          0041  1 !
;    42          0042  1 ! ENVIRONMENT:
;    43          0043  1 !
;    44          0044  1 !       STARLET operating system, including privileged system services
;    45          0045  1 !       and internal exec routines.
;    46          0046  1 !
;    47          0047  1 !--
;    48          0048  1 !
;    49          0049  1 !
;    50          0050  1 ! AUTHOR:  Steven T. Jeffreys    CREATION DATE:  18-Jul-1983
;    51          0051  1 !
;    52          0052  1 ! MODIFIED BY:
;    53          0053  1 !
;    54          0054  1 !       V03-008 HH0041          Hai Huang                       24-Jul-1984
;    55          0055  1 !               Remove REQUIRE 'LIBD$:[VMSLIB.OBJ]MOUNTMSG.B32'.
;    56          0056  1 !
;    57          0057  1 !       V03-007 HH0019          Hai Huang                       08-May-1984
```

```
;    58          0058  1 !                   Another round in fixing up truncation errors.
;    59          0059  1 !
;    60          0060  1 !           V03-006 HH0007          Hai Huang                       22-Mar-1984
;    61          0061  1 !                   Fix truncation error introduced by cluster-mount support.
;    62          0062  1 !
;    63          0063  1 !           V03-004 WMC0002         Wayne Cardoza                   16-Jan-1984
;    64          0064  1 !                   Immediately return from main routine to disable RU journals
;    65          0065  1 !
;    66          0066  1 !           V03-004 DAS0001         David Solomon                   29-Nov-1983
;    67          0067  1 !                   Add support for specifying maximum journal record size
;    68          0068  1 !                   with a new keyword, /JOURNAL=(RECORD_SIZE=n).
;    69          0069  1 !
;    70          0070  1 !           V03-003 WMC0001         Wayne Cardoza                   20-Sep-1983
;    71          0071  1 !                   DEVEXI is legal return status for CREJNL.
;    72          0072  1 !
;    73          0073  1 !           V03-002 CDS0001         Christian D. Saether            30-Aug-1983
;    74          0074  1 !                   Change name of default cjf directory from [journal] to
;    75          0075  1 !                   [sysjnl].
;    76          0076  1 !
;    77          0077  1 !           V03-001 STJ3116         Steven T. Jeffreys,             02-Aug-1983
;    78          0078  1 !                   Created local definition of $DISMOU macro.
;    79          0079  1 !
;    80          0080  1 !**
;    81          0081  1
;    82          0082  1
;    83          0083  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    84          0084  1 REQUIRE 'SRC$:MOUDEF.B32';
;    85          0616  1
;    86          0617  1
;    87          0618  1 FORWARD ROUTINE
;    88          0619  1         ACCESS_JOURNAL,                                 ! Look up RUJ file on volume
;    89          0620  1         ACTIVATE_JOURNAL,                               ! Main routine
;    90          0621  1         ANALYZE_DEVLIST : NOVALUE,                      ! Build volume data base from device list
;    91          0622  1         CREATE_RUJ,                                     ! Call $CREJNL to create an RUJ
;    92          0623  1         DISMOUNT_VOLSET : NOVALUE,                      ! Dismount all or part of a volume set
;    93          0624  1         GET_VOLUME_NAME,                                ! Return volume name
;    94          0625  1         INTERCEPT_SIGNAL,                               ! Condition handler
;    95          0626  1         RCP_RESTART;                                    ! Restart "frozen' RUJ
```

```
;    97          0627  1 !******************************************************************************
;    98          0628  1 !******************************************************************************
;    99          0629  1 !**                                                                          **
;   100          0630  1 !**                                  N O T E                                 **
;   101          0631  1 !**                                                                          **
;   102          0632  1 !** The following macro is a local redefinition of the $DISMOU system service **
;   103          0633  1 !** macro, and is necessary in order to make use of the disount system        **
;   104          0634  1 !** within the $MOUNT system service.  Since both $DISMOU and $MOUNT are      **
;   105          0635  1 !** protected shareable images, and cannot make outbound references, the      **
;   106          0636  1 !** $DISMOU code must be included into the $MOUNT code at link time.  The     **
;   107          0637  1 !** $MOUNT service then calls $DISMOU as if it were a normal routine, instead **
;   108          0638  1 !** of going through the $DISMOU service dispatcher.                         **
;   109          0639  1 !**                                                                          **
;   110          0640  1 !******************************************************************************
;   111          0641  1 !******************************************************************************
;   112          0642  1 !
;   113          0643  1 !   $DISMOU
;   114          0644  1 !
;   115          0645  1 !     Dismount Volume
;   116          0646  1 !
;   117          0647  1 !      $DISMOU devnam ,[flags]
;   118          0648  1 !
;   119          0649  1 !      devnam = address of device name string descriptor
;   120          0650  1 !
;   121          0651  1 !      flags  = 32-bit status mask  selecting options for  the  dismount
;   122          0652  1 !               The  symbols  for  the flags are  defined by the $DMTDEF
;   123          0653  1 !               macro.
;   124          0654  1 !
;   125          0655  1 !                 Flag      Meaning
;   126          0656  1 !
;   127          0657  1 !            DMT$M_NOUNLOAD   Do not unload the volume.
;   128          0658  1 !
;   129          0659  1 !            DMT$M_UNIT       Dismount the specified device, rather
;   130          0660  1 !                             than the entire volume set.
;   131          0661  1 !
;   132          0662  1 UNDECLARE %QUOTE $DISMOU;
;   133        M 0663  1 KEYWORDMACRO $DISMOU (DEVNAM=0,FLAGS=0) =
;   134        M 0664  1         (
;   135        M 0665  1         EXTERNAL ROUTINE SYS$DISMOU : BLISS ADDRESSING_MODE (GENERAL);
;   136        M 0666  1         KERNEL_CALL (SYS$DISMOU, DEVNAM, FLAGS)
;   137          0667  1         ) %;
;   138          0668  1
;   139          0669  1
;   140          0670  1 !+
;   141          0671  1 !
;   142          0672  1 ! Own storage for general use in this module.
;   143          0673  1 !
;   144          0674  1 !-
;   145          0675  1
;   146          0676  1 LITERAL
;   147          0677  1         ROOTVOL_NAMLEN  = 64;                                   ! Maximum length of a device name
;   148          0678  1
;   149          0679  1 EXTERNAL LITERAL
;   150          0680  1         CJF$_DEVEXI;
;   151          0681  1
;   152          0682  1 OWN
;   153          0683  1         AA$_DATA_BEGIN  : VECTOR [0],                           ! Mark start of data area
```

```
;  154     0684  1          DISMOUNT_COUNT   : LONG,                                  ! Count of volumes that were dismounted
;  155     0685  1          ODS2_VOLUME      : BITVECTOR [DEVMAX],                    ! Indicates volume supports ODS-2 file struc
;  156     0686  1          ROOT_PRESENT     : BITVECTOR [DEVMAX],                    ! Indicates presence of root volume for a gi
;  157     0687  1          RUJRNL_PRESENT   : BITVECTOR [DEVMAX],                    ! Indicates presence of RUJ file on a given
;  158     0688  1          ROOTVOL_DSC      : BBLOCK [DSC$K_S_BLN],                  ! Root volume device name descriptor
;  159     0689  1          ROOTVOL_BUF      : BBLOCK [ROOTVOL_NAMLEN],               ! Buffer for root volume device name
;  160     0690  1          ROOTVOL_INDEX    : VECTOR [DEVMAX, BYTE, UNSIGNED],       ! Each entry contians index+1 of root volume
;  161     0691  1          VOLSET_RVN       : VECTOR [DEVMAX, BYTE, UNSIGNED],       ! Vector of Relative Volume Numbers
;  162     0692  1          ZZ$_DATA_END     : VECTOR [0];                            ! Mark end of data area
```

```
;  164     0693  1 ROUTINE ACCESS_JOURNAL (DEVICE_NAME) =
;  165     0694  1
;  166     0695  1 !++
;  167     0696  1 !
;  168     0697  1 ! FUNCTIONAL DESCRIPTION:
;  169     0698  1 !
;  170     0699  1 !       This routine will attempt to access the Recovery Unit (RU) journal
;  171     0700  1 !       file on a given volume.
;  172     0701  1 !
;  173     0702  1 ! CALLING SEQUENCE:
;  174     0703  1 !       ACCESS_JOURNAL (ARG1)
;  175     0704  1 !
;  176     0705  1 ! INPUT PARAMETERS:
;  177     0706  1 !       ARG1:   address of a device name descriptor
;  178     0707  1 !
;  179     0708  1 ! IMPLICIT INPUTS:
;  180     0709  1 !       NONE
;  181     0710  1 !
;  182     0711  1 ! OUTPUT PARAMETERS:
;  183     0712  1 !       NONE
;  184     0713  1 !
;  185     0714  1 ! IMPLICIT OUTPUTS:
;  186     0715  1 !       NONE
;  187     0716  1 !
;  188     0717  1 ! ROUTINE VALUE:
;  189     0718  1 !       Status code passed back by $OPEN
;  190     0719  1 !
;  191     0720  1 ! SIDE EFFECTS:
;  192     0721  1 !       None.
;  193     0722  1 !
;  194     0723  1 !--
;  195     0724  1
;  196     0725  2 BEGIN
;  197     0726  2
;  198     0727  2 !
;  199     0728  2 ! Allocate plits in the $CODE$ psect to avoid truncation error when
;  200     0729  2 ! linking mountshr.
;  201     0730  2 !
;  202     0731  2 PSECT
;  203     0732  2         PLIT = $CODE$;
;  204     0733  2
;  205     0734  2 BIND
;  206     0735  2         RUJRNL_FILE     = DESCRIPTOR ('[SYSJNL]RUJNL.RUJ;1');
;  207     0736  2
;  208     0737  2 LOCAL
;  209     0738  2         STATUS;
;  210     0739  2
;  211     0740  2 MAP
;  212     0741  2         DEVICE_NAME     : REF BBLOCK,
;  213     0742  2         RUJRNL_FILE     : BBLOCK;
;  214     0743  2
;  215     0744  2 OWN
;  216     0745  2         RUJRNL_FAB      : $FAB (FOP=UFO);
;  217     0746  2
;  218     0747  2 !
;  219     0748  2 ! Build a File Attributes Block (FAB) and determine
;  220     0749  2 ! if there is a journal file on volume by attempting
```

```
;  221          0750  2 ! to open it.
;  222          0751  2 !
;  223          0752  2
;  224          0753  2 RUJRNL_FAB[FAB$B_DNS] = .DEVICE_NAME[DSC$W_LENGTH];
;  225          0754  2 RUJRNL_FAB[FAB$L_DNA] = .DEVICE_NAME[DSC$A_POINTER];
;  226          0755  2 RUJRNL_FAB[FAB$B_FNS] = .RUJRNL_FILE[DSC$W_LENGTH];
;  227          0756  2 RUJRNL_FAB[FAB$L_FNA] = .RUJRNL_FILE[DSC$A_POINTER];
;  228          0757  2
;  229          0758  3 IF (STATUS = $OPEN (FAB = RUJRNL_FAB))                    ! Attempt to open the file
;  230          0759  2 THEN
;  231          0760  2     $DASSGN (CHAN = .RUJRNL_FAB[FAB$L_STV]);               ! Deassign channel to close file
;  232          0761  2
;  233          0762  2 !
;  234          0763  2 ! If the status was RMS$_DNR, it indicates that the journal file
;  235          0764  2 ! was present, but could not be accessed, as it or the directory
;  236          0765  2 ! describing it were spread across one or more disks and at least
;  237          0766  2 ! one of those disks is not mounted.  Convert the RMS$_DNR to success
;  238          0767  2 ! to indicate that the journal file is present.
;  239          0768  2 !
;  240          0769  2 IF .STATUS EQL RMS$_DNR
;  241          0770  2 THEN
;  242          0771  2     STATUS = 1;
;  243          0772  2
;  244          0773  2 RETURN .STATUS
;  245          0774  2
;  246          0775  1 END;


                                                .TITLE  RUJMAN
                                                .IDENT  \V04-000\

                                                .PSECT  $CODE$,NOWRT,2

52 2E 4C 4E 4A 55 52 5D 4C 4E 4A 53 59 53 5B 00000 P.AAB:  .ASCII  \[SYSJNL]RUJNL.RUJ;1\                ;
                                        31 3B 4A 55 0000F                                               ;
                                              00013         .BLKB   1
                                   00000013   00014 P.AAA:  .LONG   19                                   ;
                                   00000000'  00018         .ADDRESS P.AAB                              ;

                                                .PSECT  $OWN$,NOEXE,2

                                              00000 AA$_DATA_BEGIN:
                                                    .BLKB   0
                                              00000 DISMOUNT_COUNT:
                                                    .BLKB   4
                                              00004 ODS2_VOLUME:
                                                    .BLKB   2
                                              00006 .BLKB   2
                                              00008 ROOT_PRESENT:
                                                    .BLKB   2
                                              0000A .BLKB   2
                                              0000C RUJRNL_PRESENT:
                                                    .BLKB   2
                                              0000E .BLKB   2
                                              00010 ROOTVOL_DSC:
                                                    .BLKB   8
                                              00018 ROOTVOL_BUF:
```

```
                                .BLKB   64
                00058 ROOTVOL_INDEX:
                                .BLKB   16
                00068 VOLSET_RVN:
                                .BLKB   16
                00078 ZZ$_DATA_END:
                                .BLKB   0
          03    00078 RUJRNL_FAB:
                                .BYTE   3
          50    00079           .BYTE   80
        0000    0007A           .WORD   0
    00020000    0007C           .LONG   131072
    00000000    00080           .LONG   0
    00000000    00084           .LONG   0
    00000000    00088           .LONG   0
        0000    0008C           .WORD   0
          02    0008E           .BYTE   2
          00    0008F           .BYTE   0
    00000000    00090           .LONG   0
          00    00094           .BYTE   0
          00    00095           .BYTE   0
          00    00096           .BYTE   0
          02    00097           .BYTE   2
    00000000    00098           .LONG   0
    00000000    0009C           .LONG   0
    00000000    000A0           .LONG   0
    00000000    000A4           .LONG   0
    00000000    000A8           .LONG   0
          00    000AC           .BYTE   0
          00    000AD           .BYTE   0
        0000    000AE           .WORD   0
    00000000    000B0           .LONG   0
        0000    000B4           .WORD   0
          00    000B6           .BYTE   0
          00    000B7           .BYTE   0
    00000000    000B8           .LONG   0
    00000000    000BC           .LONG   0
        0000    000C0           .WORD   0
          00    000C2           .BYTE   0
          00    000C3           .BYTE   0
    00000000    000C4           .LONG   0

                                RUJRNL_FILE=        P.AAA
                                .EXTRN  CJF$_DEVEXI, SYS$OPEN
                                .EXTRN  SYS$DASSGN

                                .PSECT  $CODE$,NOWRT,2

                        000C 00000 ACCESS_JOURNAL:
                                .WORD   Save R2,R3                              ; 0493
          53    0000'   CF  9E 00002          MOVAB   RUJRNL_FAB+53, R3
          50      04    AC  D0 00007          MOVL    DEVICE_NAME, R0           ; 0753
          63            60  90 0000B          MOVB    (R0), RUJRNL_FAB+53
    FB    A3      04    A0  D0 0000E          MOVL    4(R0), RUJRNL_FAB+48      ; 0754
    FF    A3      E2    AF  90 00013          MOVB    RUJRNL_FILE, RUJRNL_FAB+52 ; 0755
    F7    A3      E1    AF  D0 00018          MOVL    RUJRNL_FILE+4, RUJRNL_FAB+44 ; 0756
                  CB    A3  9F 0001D          PUSHAB  RUJRNL_FAB                ; 0758
```

```
                00000000G  00              01  FB 00020          CALLS   #1, SYS$OPEN                                    ;
                           52              50  D0 00027          MOVL    R0, STATUS                                      ;
                           0A              52  E9 0002A          BLBC    STATUS, 1$                                      ;
                                       D7  A3  DD 0002D          PUSHL   RUJRNL_FAB+12                                   ; 0760
                00000000G  00              01  FB 00030          CALLS   #1, SYS$DASSGN                                  ;
                00018272   8F              52  D1 00037 1$:      CMPL    STATUS, #98930                                  ; 0769
                                           03  12 0003E          BNEQ    2$                                              ;
                           52              01  D0 00040          MOVL    #1, STATUS                                      ; 0771
                           50              52  D0 00043 2$:      MOVL    STATUS, R0                                      ; 0773
                                               04 00046          RET                                                     ; 0775

; Routine Size:  71 bytes,    Routine Base:  $CODE$ + 001C
```

```
;  248     0776  1 GLOBAL ROUTINE ACTIVATE_JOURNAL =
;  249     0777  1
;  250     0778  1 !++
;  251     0779  1 !
;  252     0780  1 ! FUNCTIONAL DESCRIPTION:
;  253     0781  1 !
;  254     0782  1 !       This routine will do the appropriate RUJ operation for
;  255     0783  1 !       each device listed in the PHYS_NAME descriptor vector.
;  256     0784  1 !
;  257     0785  1 !
;  258     0786  1 ! CALLING SEQUENCE:
;  259     0787  1 !       ACTIVATE_JOURNAL ()
;  260     0788  1 !
;  261     0789  1 ! INPUT PARAMETERS:
;  262     0790  1 !       NONE
;  263     0791  1 !
;  264     0792  1 ! IMPLICIT INPUTS:
;  265     0793  1 !       <see the EXTERNAL declaration>
;  266     0794  1 !
;  267     0795  1 ! OUTPUT PARAMETERS:
;  268     0796  1 !       NONE
;  269     0797  1 !
;  270     0798  1 ! IMPLICIT OUTPUTS:
;  271     0799  1 !       NONE
;  272     0800  1 !
;  273     0801  1 ! ROUTINE VALUE:
;  274     0802  1 !       Status code passed back by RUJ routines.
;  275     0803  1 !
;  276     0804  1 ! SIDE EFFECTS:
;  277     0805  1 !       The state of the volume may be changed depending on the action of
;  278     0806  1 !       the journalling ACP.  A serious error returned by the journalling
;  279     0807  1 !       service will mandate dismounting pieces (or all) of the volume sets.
;  280     0808  1 !
;  281     0809  1 !--
;  282     0810  1
;  283     0811  2 BEGIN
;  284     0812  2
;  285     0813  2 !EXTERNAL LITERAL
;  286     0814  2 !       ROOTVOL_NAMLEN  = 64;                                   ! Maximum length of a device name
;  287     0815  2
;  288     0816  2 !EXTERNAL
;  289     0817  2 !       AA$_DATA_BEGIN  : VECTOR [0],                           ! Mark start of data area
;  290     0818  2 !       DISMOUNT_COUNT  : LONG,                                 ! Count of volumes that were dismounted
;  291     0819  2 !       ODS2_VOLUME     : BITVECTOR [DEVMAX],                   ! Indicates volume supports ODS-2 file struc
;  292     0820  2 !       ROOT_PRESENT    : BITVECTOR [DEVMAX],                   ! Indicates presence of root volume for a gi
;  293     0821  2 !       RUJRNL_PRESENT  : BITVECTOR [DEVMAX],                   ! Indicates presence of RUJ file on a given
;  294     0822  2 !       ROOTVOL_DSC     : BBLOCK [DSC$K_S_BLN],                 ! Root volume device name descriptor
;  295     0823  2 !       ROOTVOL_BUF     : BBLOCK [ROOTVOL_NAMLEN],              ! Buffer for root volume device name
;  296     0824  2 !       ROOTVOL_INDEX   : BBLOCK [DEVMAX, BYTE, UNSIGNED],      ! Each entry contians index+1 of root volume
;  297     0825  2 !       VOLSET_RVN      : VECTOR [DEVMAX, BYTE, UNSIGNED],      ! Vector of Relative Volume Numbers
;  298     0826  2 !       ZZ$_DATA_END    : VECTOR [0];                           ! Mark end of data area
;  299     0827  2
;  300     0828  2 !EXTERNAL ROUTINE
;  301     0829  2 !       ANALYZE_DEVLIST : NOVALUE ADDRESSING_MODE (GENERAL),    ! Inspect the list of devices
;  302     0830  2 !       DISMOUNT_VOLSET : NOVALUE ADDRESSING_MODE (GENERAL);    ! Dismount volumes if journalling doesn't wo
;  303     0831  2
;  304     0832  2 EXTERNAL
```

```
;  305     0833  2          MOUNT_OPTIONS   : BITVECTOR,                                  ! Mount option flag bits
;  306     0834  2          PHYS_COUNT      : LONG,                                      ! # of descriptors in PHYS_NAME
;  307     0835  2          PHYS_NAME       : BBLOCKVECTOR [DEVMAX, DSC$K_S_BLN],        ! Descriptor of root volume name
;  308     0836  2          STORED_CONTEXT  : BITVECTOR;                                 ! Determines if tape or disk mount
;  309     0837  2
;  310     0838  2 LITERAL
;  311     0839  2          ALL_VOLUMES     = 1,                                         ! Various and sundry mnemonics
;  312     0840  2          ONE_VOLUME      = 0,
;  313     0841  2          PERMANENT_RUJ   = 1,
;  314     0842  2          TEMPORARY_RUJ   = 0;
;  315     0843  2
;  316     0844  2 LOCAL
;  317     0845  2          RUJRNL_EXISTS,                                               ! Boolean
;  318     0846  2          STATUS;
;  319     0847  2
;  320     0848  2
;  321     0849  2 !
;  322     0850  2 ! If the user explicitly requested that Recovery Unit Journalling (RUJ)
;  323     0851  2 ! not be activated, return a success status immediately.
;  324     0852  2 !
;  325     0853  2 !***JNL** turn off RU journals
;  326     0854  2
;  327     0855  2 MOUNT_OPTIONS[OPT_NOJRNL] = 1; !***JNL**
;  328     0856  2 IF .MOUNT_OPTIONS[OPT_NOJRNL]
;  329     0857  2 THEN
;  330     0858  2     RETURN 1;
;  331     0859  2
;  332     0860  2 !
;  333     0861  2 ! RUJ is only appropriate for disk volumes.  If we are mounting a tape
;  334     0862  2 ! or a foreign disk, return a success status immediately.
;  335     0863  2 !
;  336     0864  2 IF .STORED_CONTEXT[TAPE_MOUNT]
;  337     0865  2 OR .MOUNT_OPTIONS[OPT_FOREIGN]
;  338     0866  2 THEN
;  339     0867  2     RETURN 1;
;  340     0868  2
;  341     0869  2 !
;  342     0870  2 ! Inspect the list of devices that have been mounted, and record some
;  343     0871  2 ! useful information.  This is necessary because $MOUNT allows a great
;  344     0872  2 ! deal of flexibility in the amount and type of disks volumes that may
;  345     0873  2 ! be mounted.
;  346     0874  2 !
;  347     0875  2 ANALYZE_DEVLIST (PHYS_NAME[0, DSC$W_LENGTH], .PHYS_COUNT);
;  348     0876  2
;  349     0877  2 !+
;  350     0878  2 ! Now that the volume/device data base has been created, process each
;  351     0879  2 ! volume in the device list.  Three passes are made through the list,
;  352     0880  2 ! and the ordering of the processing is important.  The actions performed are:
;  353     0881  2 !
;  354     0882  2 !       1. For each ODS-2 volume (mounted on a device in the device list) that
;  355     0883  2 !          is a member of a volume set, but not the root volume of a volume set,
;  356     0884  2 !          and contains a RUJ file, create a temporary RUJ to process the
;  357     0885  2 !          existing RUJ file, and delete the file when done.
;  358     0886  2 !
;  359     0887  2 !          This pass is meant to process new members to a volume set that
;  360     0888  2 !          already contain user files, and might have an active RUJ file
;  361     0889  2 !          present on the volume.  The volume must be in a consistent state
```

```
;  362     0890  2 !           before it can be a full-fledged member of the volume set.  A
;  363     0891  2 !           journalling failure at this point will cause the volume, but not
;  364     0892  2 !           the volume set, to be dismounted.
;  365     0893  2 !
;  366     0894  2 !       2. For each ODS-2 volume (mounted on a device in the device list) that
;  367     0895  2 !           is a root volume of a volume set, or not part of any volume set,
;  368     0896  2 !           attempt to create a permanent RUJ on the volume if a journal file
;  369     0897  2 !           already exists, or if the user requested one be created.
;  370     0898  2 !
;  371     0899  2 !           Volumes of this type are the most prevalent.  A journalling failure
;  372     0900  2 !           at this point will cause the entire volume set to be dismounted.
;  373     0901  2 !
;  374     0902  2 !       3. For each ODS-2 volume (mounted on a device in the device list) that
;  375     0903  2 !           is a member of a volume set but not a root volume, attempt to restart
;  376     0904  2 !           a "frozen" recovery unit.
;  377     0905  2 !
;  378     0906  2 !           This pass handles the case of a volume set that is mounted in stages.
;  379     0907  2 !           An active RUJ on an imcomplete volume set might become "frozen" if
;  380     0908  2 !           data on missing volumes is needed to complete the journalling
;  381     0909  2 !           operation.  Note that if more pieces of the volume set are still
;  382     0910  2 !           not mounted, the RUJ may remain frozen.  A journalling failure at
;  383     0911  2 !           this point will cause the entire volume set to be dismounted.  Note
;  384     0912  2 !           that if the RUJ remains frozen, the volume set is not dismounted.
;  385     0913  2 !
;  386     0914  2 ! Note that DISMOUNT_VOLSET will update the local device list database to reflect
;  387     0915  2 ! the removal (dismounting) of volumes.
;  388     0916  2 !
;  389     0917  2 ! This level of checking is necessary because $MOUNT will allow many types
;  390     0918  2 ! of volumes to be mounted with a single invocation.  For example, more than
;  391     0919  2 ! one volume set can be mounted; ODS1 and ODS2 disks can be mounted together
;  392     0920  2 ! also.
;  393     0921  2 !-
;  394     0922  2
;  395     0923  2 !
;  396     0924  2 ! Handle case 1
;  397     0925  2 !
;  398     0926  2 INCR J FROM 0 TO .PHYS_COUNT-1 DO
;  399     0927  2     IF  .ODS2_VOLUME[.J]
;  400     0928  2     AND .VOLSET_RVN[.J] GTR 1
;  401     0929  2     AND .RUJRNL_PRESENT[.J]
;  402     0930  2     THEN
;  403     0931  3         IF NOT (STATUS = EXEC_CALL (CREATE_RUJ, PHYS_NAME[.J, DSC$W_LENGTH], TEMPORARY_RUJ))
;  404     0932  2         THEN
;  405     0933  2             DISMOUNT_VOLSET (PHYS_NAME, .PHYS_COUNT, .J,  ONE_VOLUME, .STATUS);
;  406     0934  2
;  407     0935  2 !
;  408     0936  2 ! Handle case 2.
;  409     0937  2 !
;  410     0938  2 INCR J FROM 0 TO .PHYS_COUNT-1 DO
;  411     0939  2     IF  .ODS2_VOLUME[.J]
;  412     0940  2     AND .VOLSET_RVN[.J] LEQ 1
;  413     0941  3     AND (.RUJRNL_PRESENT[.J] OR .MOUNT_OPTIONS[OPT_NEWJRNL])
;  414     0942  2     THEN
;  415     0943  3         IF NOT (STATUS = EXEC_CALL (CREATE_RUJ, PHYS_NAME[.J, DSC$W_LENGTH], PERMANENT_RUJ))
;  416     0944  2         THEN
;  417     0945  2             DISMOUNT_VOLSET (PHYS_NAME, .PHYS_COUNT, .J,  ALL_VOLUMES, .STATUS);
;  418     0946  2
```

```
;  419          0947  2 !
;  420          0948  2 ! Handle case 3.
;  421          0949  2 !
;  422          0950  2 INCR J FROM 0 TO .PHYS_COUNT-1 DO
;  423          0951  2     IF  .ODS2_VOLUME[.J]
;  424          0952  2     AND .VOLSET_RVN[.J] GTR 1
;  425          0953  2     AND NOT .ROOT_PRESENT[.J]
;  426          0954  2     THEN
;  427          0955  3         IF NOT (STATUS = EXEC_CALL (RCP_RESTART, PHYS_NAME[.J, DSC$W_LENGTH]))
;  428          0956  2         THEN
;  429          0957  2             DISMOUNT_VOLSET (PHYS_NAME, .PHYS_COUNT, .J,  ALL_VOLUMES, .STATUS);
;  430          0958  2
;  431          0959  2 !
;  432          0960  2 ! Compare the number of volumes dismounted against the original number
;  433          0961  2 ! of volumes in the device list.  There are three posibilities:
;  434          0962  2 !       1. No volumes have been dismounted - return SS$_NORMAL
;  435          0963  2 !       2. Some but not all volumes have been dismounted - return MOUN$_DISMPART
;  436          0964  2 !       3. All volumes have been dismounted - return MOUN$_DISMAL
;  437          0965  2 !
;  438          0966  2 IF .DISMOUNT_COUNT LEQ 0
;  439          0967  2 THEN
;  440          0968  2     RETURN SS$_NORMAL
;  441          0969  2 ELSE
;  442          0970  2     IF .DISMOUNT_COUNT LSS .PHYS_COUNT
;  443          0971  2     THEN
;  444          0972  2         RETURN MOUN$_DISMPART
;  445          0973  2     ELSE
;  446          0974  2         RETURN MOUN$_DISMAL
;  447          0975  2
;  448          0976  1 END;

                                                .EXTRN  MOUNT_OPTIONS, PHYS_COUNT
                                                .EXTRN  PHYS_NAME, STORED_CONTEXT
                                                .EXTRN  SYS$CMEXEC

                                07FC 00000      .ENTRY  ACTIVATE_JOURNAL, Save R2,R3,R4,R5,R6,R7,-      ; 0776
                                                        R8,R9,R10
                    5A      0000V CF   9E 00002         MOVAB   DISMOUNT_VOLSET, R10                    ;
                    59 00000000G  00   9E 00007         MOVAB   MOUNT_OPTIONS+6, R9                     ;
                    58 00000000G  9F   9E 0000E         MOVAB   @#SYS$CMEXEC, R8                        ;
                    57      0000' CF   9E 00015         MOVAB   ODS2_VOLUME, R7                         ;
                    56 00000000G  00   9E 0001A         MOVAB   PHYS_NAME, R6                           ;
                    55 00000000G  00   9E 00021         MOVAB   PHYS_COUNT, R5                          ;
                    69        80  8F   88 00028         BISB2   #128, MOUNT_OPTIONS+6                   ; 0855
                                  03   18 0002C         BGEQ    2$                                      ; 0856
                                00DA   31 0002E 1$:     BRW     10$                                     ;
                    F6 00000000G  00   E8 00031 2$:     BLBS    STORED_CONTEXT, 1$                      ; 0864
F1        FB        A9            03   E0 00038         BBS     #3, MOUNT_OPTIONS+1, 1$                 ; 0865
                                  65   DD 0003D         PUSHL   PHYS_COUNT                              ; 0875
                                  56   DD 0003F         PUSHL   R6                                      ;
          0000V     CF            02   FB 00041         CALLS   #2, ANALYZE_DEVLIST                     ;
                    53            65   D0 00046         MOVL    PHYS_COUNT, R3                          ; 0926
                    52            01   CE 00049         MNEGL   #1, J                                   ;
                                  33   11 0004C         BRB     4$                                      ;
2F                  67            52   E1 0004E 3$:     BBC     J, ODS2_VOLUME, 4$                      ; 0927
```

```
                01      64 A742  91 00052          CMPB    VOLSET_RVN[J], #1                       ; 0928
                              28  1B 00057          BLEQU   4$
23      08      A7            52  E1 00059          BBC     J, RUJRNL_PRESENT, 4$                   ; 0929
                              7E  D4 0005E          CLRL    -(SP)                                   ; 0931
                            6642  7F 00060          PUSHAQ  PHYS_NAME[J]
                              02  DD 00063          PUSHL   #2
                              5E  DD 00065          PUSHL   SP
                        0000V CF  9F 00067          PUSHAB  CREATE_RUJ
                      68        05  FB 0006B          CALLS   #5, SYS$CMEXEC
                      54        50  D0 0006E          MOVL    R0, STATUS
                      0D        54  E8 00071          BLBS    STATUS, 4$
                                54  DD 00074          PUSHL   STATUS                                  ; 0933
                                7E  D4 00076          CLRL    -(SP)
                                52  DD 00078          PUSHL   J
                                65  DD 0007A          PUSHL   PHYS_COUNT
                                56  DD 0007C          PUSHL   R6
                      6A        05  FB 0007E          CALLS   #5, DISMOUNT_VOLSET
C9                    52        53  F2 00081 4$:      AOBLSS  R3, J, 3$                       ; 0927
                      53        65  D0 00085          MOVL    PHYS_COUNT, R3                  ; 0938
                      52        01  CE 00088          MNEGL   #1, J
                                37  11 0008B          BRB     7$
33                    67        52  E1 0008D 5$:      BBC     J, ODS2_VOLUME, 7$              ; 0939
                01      64 A742  91 00091          CMPB    VOLSET_RVN[J], #1               ; 0940
                              2C  1A 00096          BGTRU   7$
04      08      A7            52  E0 00098          BBS     J, RUJRNL_PRESENT, 6$           ; 0941
                23      01    A9  E9 0009D          BLBC    MOUNT_OPTIONS+7, 7$
                                01  DD 000A1 6$:      PUSHL   #1                                      ; 0943
                            6642  7F 000A3          PUSHAQ  PHYS_NAME[J]
                              02  DD 000A6          PUSHL   #2
                              5E  DD 000A8          PUSHL   SP
                        0000V CF  9F 000AA          PUSHAB  CREATE_RUJ
                      68        05  FB 000AE          CALLS   #5, SYS$CMEXEC
                      54        50  D0 000B1          MOVL    R0, STATUS
                      0D        54  E8 000B4          BLBS    STATUS, 7$
                                54  DD 000B7          PUSHL   STATUS                                  ; 0945
                                01  DD 000B9          PUSHL   #1
                                52  DD 000BB          PUSHL   J
                                65  DD 000BD          PUSHL   PHYS_COUNT
                                56  DD 000BF          PUSHL   R6
                      6A        05  FB 000C1          CALLS   #5, DISMOUNT_VOLSET
C5                    52        53  F2 000C4 7$:      AOBLSS  R3, J, 5$                       ; 0939
                      53        65  D0 000C8          MOVL    PHYS_COUNT, R3                  ; 0950
                      52        01  CE 000CB          MNEGL   #1, J
                                31  11 000CE          BRB     9$
2D                    67        52  E1 000D0 8$:      BBC     J, ODS2_VOLUME, 9$              ; 0951
                01      64 A742  91 000D4          CMPB    VOLSET_RVN[J], #1               ; 0952
                              26  1B 000D9          BLEQU   9$
21      04      A7            52  E0 000DB          BBS     J, ROOT_PRESENT, 9$             ; 0953
                            6642  7F 000E0          PUSHAQ  PHYS_NAME[J]                    ; 0955
                              01  DD 000E3          PUSHL   #1
                              5E  DD 000E5          PUSHL   SP
                        0000V CF  9F 000E7          PUSHAB  RCP_RESTART
                      68        04  FB 000EB          CALLS   #4, SYS$CMEXEC
                      54        50  D0 000EE          MOVL    R0, STATUS
                      0D        54  E8 000F1          BLBS    STATUS, 9$
                                54  DD 000F4          PUSHL   STATUS                                  ; 0957
                                01  DD 000F6          PUSHL   #1
```

```
                                        52  DD 000F8          PUSHL   J                                                       ;
                                        65  DD 000FA          PUSHL   PHYS_COUNT                                              ;
                                        56  DD 000FC          PUSHL   R6                                                      ;
                                6A      05  FB 000FE          CALLS   #5, DISMOUNT_VOLSET                                     ;
                             CB 52      53  F2 00101 9$:      AOBLSS  R3, J, 8$                                               ; 0951
                                50      FC  A7 DO 00105       MOVL    DISMOUNT_COUNT, RO                                      ; 0966
                                        04  14 00109          BGTR    11$
                                50          01  DO 0010B 10$: MOVL    #1, RO                                                  ; 0970
                                            04  0010E         RET
                                65          50  D1 0010F 11$: CMPL    RO, PHYS_COUNT                                          ;
                                            08  18 00112      BGEQ    12$
                                50 00729050 8F  DO 00114      MOVL    #7508048, RO                                            ; 0972
                                            04  0011B         RET
                                50 0072822C 8F  DO 0011C 12$: MOVL    #7504428, RO                                            ; 0974
                                            04  00123         RET                                                             ; 0976

; Routine Size:  292 bytes,    Routine Base:  $CODE$ + 0063
```

```
;  450     0977  1 ROUTINE ANALYZE_DEVLIST (DEVNAM_LIST, LIST_LENGTH) : NOVALUE =
;  451     0978  1
;  452     0979  1 !++
;  453     0980  1 !
;  454     0981  1 ! FUNCTIONAL DESCRIPTION:
;  455     0982  1 !
;  456     0983  1 !       For each device in the physical device name descriptor vector, determine
;  457     0984  1 !               o the file structure level (ODS-1 or ODS-2)
;  458     0985  1 !               o the name of the root volume in the volume set
;  459     0986  1 !                 (if present in the device list, record the index)
;  460     0987  1 !               o the Relative Volume Number (RVN)
;  461     0988  1 !               o whether or not the root volume is in the list
;  462     0989  1 !               o whether of not a volume has a RUJ file
;  463     0990  1 !
;  464     0991  1 ! CALLING SEQUENCE:
;  465     0992  1 !       ANALYZE_DEVLIST (ARG1, ARG2)
;  466     0993  1 !
;  467     0994  1 ! INPUT PARAMETERS:
;  468     0995  1 !       ARG1    : address of a vector of device name descriptors
;  469     0996  1 !       ARG2    : number of device name descriptors in the list
;  470     0997  1 !
;  471     0998  1 ! IMPLICIT INPUTS:
;  472     0999  1 !       <see the EXTERNAL declaration>
;  473     1000  1 !
;  474     1001  1 ! OUTPUT PARAMETERS:
;  475     1002  1 !       NONE
;  476     1003  1 !
;  477     1004  1 ! IMPLICIT OUTPUTS:
;  478     1005  1 !       NONE
;  479     1006  1 !
;  480     1007  1 ! ROUTINE VALUE:
;  481     1008  1 !       NONE
;  482     1009  1 !
;  483     1010  1 ! SIDE EFFECTS:
;  484     1011  1 !       NONE
;  485     1012  1 !
;  486     1013  1 !--
;  487     1014  1
;  488     1015  2 BEGIN                                                   ! Start of ANALYZE_DEVLIST
;  489     1016  2
;  490     1017  2 !EXTERNAL LITERAL
;  491     1018  2 !       ROOTVOL_NAMLEN  = 64;                           ! Maximum length of a device name
;  492     1019  2
;  493     1020  2 !EXTERNAL
;  494     1021  2 !       AA$_DATA_BEGIN  : VECTOR [0],                   ! Mark start of data area
;  495     1022  2 !       DISMOUNT_COUNT  : LONG,                         ! Count of volumes that were dismounted
;  496     1023  2 !       ODS2_VOLUME     : BITVECTOR [DEVMAX],           ! Indicates volume supports ODS-2 file struc
;  497     1024  2 !       ROOT_PRESENT    : BITVECTOR [DEVMAX],           ! Indicates presence of root volume for a gi
;  498     1025  2 !       RUJRNL_PRESENT  : BITVECTOR [DEVMAX],           ! Indicates presence of RUJ file on a given
;  499     1026  2 !       ROOTVOL_DSC     : BBLOCK [DSC$K_S_BLN],         ! Root volume device name descriptor
;  500     1027  2 !       ROOTVOL_BUF     : BBLOCK [ROOTVOL_NAMLEN],      ! Buffer for root volume device name
;  501     1028  2 !       ROOTVOL_INDEX   : BBLOCK [DEVMAX, BYTE, UNSIGNED],  ! Each entry contians index+1 of root volume
;  502     1029  2 !       VOLSET_RVN      : VECTOR [DEVMAX, BYTE, UNSIGNED],  ! Vector of Relative Volume Numbers
;  503     1030  2 !       ZZ$_DATA_END    : VECTOR [0];                   ! Mark end of data area
;  504     1031  2
;  505     1032  2 !EXTERNAL ROUTINE
;  506     1033  2 !       ACCESS_JOURNAL;                                 ! Determine if RUJ file present on a volume
```

```
;  507     1034  2
;  508     1035  2 LOCAL
;  509     1036  2         I;                                              ! Loop counter
;  510     1037  2
;  511     1038  2 MAP
;  512     1039  2         DEVNAM_LIST     : REF BBLOCKVECTOR [DEVMAX, DSC$K_S_BLN];
;  513     1040  2
;  514     1041  2 OWN
;  515     1042  2         ACP_TYPE        : LONG,                         ! Volume ACP type
;  516     1043  2         ROOTNAME_BUF    : BBLOCK [ROOTVOL_NAMLEN],      ! Root volume device name buffer
;  517     1044  2         VOLUME_NUMBER   : LONG,                         ! Relative volume number
;  518     1045  2
;  519     1046  2         DVI_LIST        : BBLOCK [10*4]                 ! $GETDVI item list
;  520     1047  2                           INITIAL (
;  521     1048  2                                    WORD (64),         !    ITEM - Root volume device name
;  522     1049  2                                    WORD (DVI$_ROOTDEVNAM),
;  523     1050  2                                    LONG (0),          !    Filled in at run time
;  524     1051  2                                    LONG (0),          !    Filled in at run time
;  525     1052  2                                    WORD (4),          !    ITEM - Relative Volume Number
;  526     1053  2                                    WORD (DVI$_VOLNUMBER),
;  527     1054  2                                    LONG (VOLUME_NUMBER),
;  528     1055  2                                    LONG (0),
;  529     1056  2                                    WORD (4),          !    ITEM - ACP type
;  530     1057  2                                    WORD (DVI$_ACPTYPE),
;  531     1058  2                                    LONG (ACP_TYPE),
;  532     1059  2                                    LONG (0),
;  533     1060  2                                    LONG (0)           !    List terminator
;  534     1061  2                                   );
;  535     1062  2
;  536     1063  2 !
;  537     1064  2 ! Initialize the data area.
;  538     1065  2 !
;  539     1066  2 CH$FILL (0, (ZZ$_DATA_END-AA$_DATA_BEGIN), AA$_DATA_BEGIN);
;  540     1067  2
;  541     1068  2 !
;  542     1069  2 ! For each device listed, determine
;  543     1070  2 !       o the file structure level (ODS-1 or ODS-2)
;  544     1071  2 !       o its Relative Volume Number (RVN)
;  545     1072  2 !       o if it has a journal file
;  546     1073  2 !       o the device name the root volume is mounted on
;  547     1074  2 !       o if the root volume was mounted by this invocation of $MOUNT
;  548     1075  2 !
;  549     1076  2 INCR J FROM 0 TO .LIST_LENGTH-1 DO
;  550     1077  3     BEGIN
;  551     1078  3     !
;  552     1079  3     ! Get the RVN and root volume device name.
;  553     1080  3     !
;  554     1081  3     ROOTVOL_DSC[DSC$A_POINTER] = ROOTVOL_BUF;
;  555     1082  3     DVI_LIST[4, 0, 32, 0] = ROOTVOL_BUF;
;  556     1083  3     DVI_LIST[8, 0, 32, 0] = ROOTVOL_DSC[DSC$W_LENGTH];
;  557     1084  3     $GETDVI (DEVNAM = DEVNAM_LIST[.J, DSC$W_LENGTH], ITMLST=DVI_LIST);
;  558     1085  3     VOLSET_RVN[.J] = .VOLUME_NUMBER;
;  559     1086  3
;  560     1087  3     !
;  561     1088  3     ! If the volume supports the ODS-2 file structure, note it, and
;  562     1089  3     ! determine if the root volume for this volume set was mounted in
;  563     1090  3     ! this invocation of $MOUNT.  Note that if the RVN is LEQ 1, then
```

```
;  564          1091  3      ! this volume is the root volume.
;  565          1092  3      !
;  566          1093  3      IF .ACP_TYPE EQL DVI$C_ACP_F11V2
;  567          1094  3      THEN
;  568          1095  4          BEGIN
;  569          1096  4          ODS2_VOLUME[.J] = 1;                                    ! Note that this volume supports ODS-2.
;  570          1097  4          RUJRNL_PRESENT[.J] = ACCESS_JOURNAL (DEVNAM_LIST[.J, DSC$W_LENGTH]);
;  571          1098  4          IF .VOLSET_RVN[.J] LEQ 1
;  572          1099  4          THEN
;  573          1100  5              BEGIN                                               ! This is the volume is its own root
;  574          1101  5              ROOT_PRESENT[.J] = 1;                               ! Note the the root volume is present in the
;  575          1102  5              ROOTVOL_INDEX[.J] = .J + 1;                         ! Note the index of the root volume (biased
;  576          1103  4              END;
;  577          1104  4          !
;  578          1105  4          ! Check the root volume device name for this device against those
;  579          1106  4          ! in the supplied list.  If no match, it means that this volume is
;  580          1107  4          ! an addition to an already mounted volume set.
;  581          1108  4          !
;  582          1109  4          I = 0;
;  583          1110  4          WHILE NOT .ROOT_PRESENT[.J] AND (.I LEQ .LIST_LENGTH-1) DO
;  584          1111  4              IF CH$EQL   (.DEVNAM_LIST[.I, DSC$W_LENGTH],
;  585          1112  4                           .DEVNAM_LIST[.I, DSC$A_POINTER],
;  586          1113  4                           .ROOTVOL_DSC[DSC$W_LENGTH],
;  587          1114  4                           .ROOTVOL_DSC[DSC$A_POINTER]
;  588          1115  4                          )
;  589          1116  4              THEN
;  590          1117  5                  BEGIN
;  591          1118  5                  ROOT_PRESENT[.J] = 1;                       ! Note the the root volume is present in the
;  592          1119  5                  ROOTVOL_INDEX[.J] = .I + 1;                 ! Note the index of the root volume (biased
;  593          1120  5                  END
;  594          1121  4              ELSE
;  595          1122  4                  I = .I + 1;
;  596          1123  3          END;
;  597          1124  2      END;
;  598          1125  2
;  599          1126  1 END;                                                           ! End of ANALYZE_DEVLIST


                                                .PSECT  $OWN$,NOEXE,2

                                        000C8 ACP_TYPE:
                                                .BLKB   4
                                        000CC ROOTNAME_BUF:
                                                .BLKB   64
                                        0010C VOLUME_NUMBER:
                                                .BLKB   4
                                    0040  00110 DVI_LIST:
                                                .WORD   64                              ;
                                    0032  00112         .WORD   50                      ;
                                00000000  00114         .LONG   0                       ;
                                00000000  00118         .LONG   0                       ;
                                    0004  0011C         .WORD   4                       ;
                                    002E  0011E         .WORD   46                      ;
                                00000000' 00120         .ADDRESS VOLUME_NUMBER          ;
                                00000000  00124         .LONG   0                       ;
                                    0004  00128         .WORD   4                       ;
```

```
                                  0042  0012A         .WORD    66                                   ;
                             00000000'  0012C         .ADDRESS ACP_TYPE                             ;
                              00000000  00130         .LONG    0                                    ;
                              00000000  00134         .LONG    0                                    ;

                                                      .EXTRN   SYS$GETDVI

                                                      .PSECT   $CODE$,NOWRT,2

                                  007C 00000 ANALYZE_DEVLIST:
                                                      .WORD    Save R2,R3,R4,R5,R6                  ; 0977
                      56   0000'  CF  9E 00002        MOVAB    ROOT_PRESENT, R6                     ;
 0078  8F        00   6E          00  2C 00007        MOVC5    #0, (SP), #0, #120, AA$_DATA_BEGIN   ; 1066
                            F8    A6     0000E                                                      ;
                      54          01  CE 00010        MNEGL    #1, J                                ; 1084
                                0089  31 00013        BRW      8$                                   ;
            0C   A6       10      A6  9E 00016 1$:    MOVAB    ROOTVOL_BUF, ROOTVOL_DSC+4           ; 1081
          010C   C6       10      A6  9E 0001B        MOVAB    ROOTVOL_BUF, DVI_LIST+4              ; 1082
          0110   C6       08      A6  9E 00021        MOVAB    ROOTVOL_DSC, DVI_LIST+8              ; 1083
                                  7E  7C 00027        CLRQ     -(SP)                                ; 1084
                                  7E  7C 00029        CLRQ     -(SP)                                ;
                        0108      C6  9F 0002B        PUSHAB   DVI_LIST                             ;
                           04   BC44  7F 0002F        PUSHAQ   @DEVNAM_LIST[J]                      ;
                                  7E  7C 00033        CLRQ     -(SP)                                ;
      00000000G  00               08  FB 00035        CALLS    #8, SYS$GETDVI                       ;
          60 A644         0104    C6  90 0003C        MOVB     VOLUME_NUMBER, VOLSET_RVN[J]         ; 1085
                 02       00C0    C6  D1 00043        CMPL     ACP_TYPE, #2                         ; 1093
                                  55  12 00048        BNEQ     8$                                   ;
          00        FC    A6      54  E2 0004A        BBSS     J, ODS2_VOLUME, 2$                   ; 1096
                           04   BC44  7F 0004F 2$:    PUSHAQ   @DEVNAM_LIST[J]                      ; 1097
                  FE3D    CF      01  FB 00053        CALLS    #1, ACCESS_JOURNAL                   ;
  04   A6        01       54      50  F0 00058        INSV     R0, J, #1, RUJRNL_PRESENT            ;
                          01  60 A644  91 0005E        CMPB     VOLSET_RVN[J], #1                    ; 1098
                                  0A  1A 00063        BGTRU    4$                                   ;
          00              66      54  E2 00065        BBSS     J, ROOT_PRESENT, 3$                  ; 1101
      50 A644             54      01  81 00069 3$:    ADDB3    #1, J, ROOTVOL_INDEX[J]              ; 1102
                                  55  D4 0006F 4$:    CLRL     I                                    ; 1109
          2A              66      54  E0 00071 5$:    BBS      J, ROOT_PRESENT, 8$                  ; 1110
          50       08     AC      01  C3 00075        SUBL3    #1, LIST_LENGTH, R0                  ;
                          50      55  D1 0007A        CMPL     I, R0                                ;
                                  20  14 0007D        BGTR     8$                                   ;
                      50   04   BC45  7E 0007F        MOVAQ    @DEVNAM_LIST[I], R0                  ; 1111
  08   A6        00  04   B0      60  2D 00084        CMPC5    (R0), @4(R0), #0, ROOTVOL_DSC, -     ;
                            0C    B6     0008B                 @ROOTVOL_DSC+4                       ;
                                  0C  12 0008D        BNEQ     7$                                   ;
          00              66      54  E2 0008F        BBSS     J, ROOT_PRESENT, 6$                  ; 1118
      50 A644             55      01  81 00093 6$:    ADDB3    #1, I, ROOTVOL_INDEX[J]              ; 1119
                                  D6  11 00099        BRB      5$                                   ; 1111
                                  55  D6 0009B 7$:    INCL     I                                    ; 1122
                                  D2  11 0009D        BRB      5$                                   ; 1111
          01              54  08  AC  F2 0009F 8$:    AOBLSS   LIST_LENGTH, J, 9$                   ; 1076
                                  04     000A4        RET                                           ; 1126
                                FF6E  31 000A5 9$:    BRW      1$                                   ; 1076

; Routine Size:  168 bytes,    Routine Base:  $CODE$ + 0187
```

```
;  601     1127  1 ROUTINE CREATE_RUJ (DEVICE_NAME, PERMANENT_RUJ) =
;  602     1128  1
;  603     1129  1 !++
;  604     1130  1 !
;  605     1131  1 ! FUNCTIONAL DESCRIPTION:
;  606     1132  1 !
;  607     1133  1 !       Create a JSB and call the $CREJNL system service to create a
;  608     1134  1 !       RUJ on a specified device.  The longevity of the RUJ is also
;  609     1135  1 !       specified.
;  610     1136  1 !
;  611     1137  1 ! CALLING SEQUENCE:
;  612     1138  1 !       CREATE_RUJ (ARG1, ARG2)
;  613     1139  1 !
;  614     1140  1 ! INPUT PARAMETERS:
;  615     1141  1 !       ARG1    : address of a device name descriptor
;  616     1142  1 !       ARG2    : if 0 then create a temporary RUJ, otherwise create a permanent RUJ.
;  617     1143  1 !
;  618     1144  1 ! IMPLICIT INPUTS:
;  619     1145  1 !       NONE
;  620     1146  1 !
;  621     1147  1 ! OUTPUT PARAMETERS:
;  622     1148  1 !       NONE
;  623     1149  1 !
;  624     1150  1 ! IMPLICIT OUTPUTS:
;  625     1151  1 !       NONE
;  626     1152  1 !
;  627     1153  1 ! ROUTINE VALUE:
;  628     1154  1 !       Status code passed back by RUJ routines.
;  629     1155  1 !
;  630     1156  1 ! SIDE EFFECTS:
;  631     1157  1 !       The state of the volume may be changed depending on the action of
;  632     1158  1 !       the journalling ACP.
;  633     1159  1 !       Success messages are SIGNALED, errors are handled outside of this
;  634     1160  1 !       routine.
;  635     1161  1 !
;  636     1162  1 !--
;  637     1163  1
;  638     1164  2 BEGIN                                                   ! Start of CREATE_JOURNAL
;  639     1165  2
;  640     1166  2 EXTERNAL
;  641     1167  2         JRNL_EXTEND,                                    ! RUJ default file extension size
;  642     1168  2         JRNL_QUOTA,                                     ! RUJ byte quota per recovery unit
;  643     1169  2         JRNL_SIZE,                                      ! Recovery Unit Journal (RUJ)  initial size
;  644     1170  2         JRNL_RECORD_SIZE,                                       ! RUJ maximum record size
;  645     1171  2         MOUNT_OPTIONS   : BITVECTOR;                    ! Option flags
;  646     1172  2
;  647     1173  2 !EXTERNAL ROUTINE
;  648     1174  2 !       GET_VOLUME_NAME : ADDRESSING_MODE (GENERAL),    ! Return volume name for a given device
;  649     1175  2 !       INTERCEPT_SIGNAL: ADDRESSING_MODE (GENERAL);    ! Condition handler
;  650     1176  2
;  651     1177  2 LOCAL
;  652     1178  2         JRNL_CHANNEL,                                   ! Channel to volume
;  653     1179  2         STATUS          : BBLOCK [4],
;  654     1180  2         VOLUME_NAME     : REF BBLOCK;                   ! Volume name descriptor
;  655     1181  2
;  656     1182  2 OWN
;  657     1183  2         JSB             : BBLOCK [JSB$C_LENGTH];        ! Journal State Block
```

```
;  658          1184  2
;  659          1185  2
;  660          1186  2 !
;  661          1187  2 ! Enable a condition handler to field and print signalled messages.
;  662          1188  2 !
;  663          1189  2 ENABLE INTERCEPT_SIGNAL;
;  664          1190  2
;  665          1191  2 !
;  666          1192  2 ! Build the JSB in OWN storage (stack space is at a premium).
;  667          1193  2 !
;  668          1194  2 JRNL_CHANNEL = 0;
;  669          1195  2 CH$FILL (0, JSB$C_LENGTH, JSB);                          ! Initialize the JSB
;  670          1196  2 JSB[JSB$B_JNLTYP] = DT$_RUJNL;                          ! Set the journal type
;  671          1197  2 JSB[JSB$L_FILSIZ] = .JRNL_SIZE;                        ! Set RUJ initial size
;  672          1198  2 JSB[JSB$W_FILEXT] = .JRNL_EXTEND;                      ! Set RUJ extend size
;  673          1199  2 JSB[JSB$W_MAXSIZ] = .JRNL_RECORD_SIZE;                 ! Set RUJ max record size
;  674          1200  2 JSB[JSB$L_QUOTA] =  .JRNL_QUOTA;                      ! Set RUJ process byte quota
;  675          1201  2 JSB[JSB$W_BUFSIZ] = 1;                                ! Use 1 block buffers in driver
;  676          1202  2 JSB[JSB$B_COPIES] = 1;                                ! Only one RUJ on volume
;  677          1203  2 JSB[JSB$B_JNLDEV] = JSB$C_DISK;                       ! Specify device class
;  678          1204  2 JSB[JSB$B_ACMODE] = PSL$C_USER;                       ! Specify journal access mode
;  679          1205  2 JSB[JSB$L_PRINAMDES] = .DEVICE_NAME;                  ! devnam descriptor address
;  680          1206  2 CH$FILL (-1, 8, JSB[JSB$Q_EXPDAT]);                    ! default exparation date
;  681          1207  2                                                        !  Ignorant Bliss STILL does not handle quad
;  682          1208  2 !
;  683          1209  2 ! Set RUJ creation flags
;  684          1210  2 !
;  685          1211  2 IF .PERMANENT_RUJ
;  686          1212  2 THEN
;  687          1213  2     JSB[JSB$V_CIF] = .MOUNT_OPTIONS[OPT_NEWJRNL]
;  688          1214  2 ELSE
;  689          1215  3     BEGIN
;  690          1216  3     JSB[JSB$V_TMPJNL] = 1;
;  691          1217  3     JSB[JSB$V_TMPFIL] = 1;
;  692          1218  2     END;
;  693          1219  2
;  694          1220  2 !
;  695          1221  2 ! Activate the journal (this is a synchronous operation).
;  696          1222  2 ! Parameters that are defaulted are flaged with (*).
;  697          1223  2 !
;  698          1224  2 ! The $CREJNL code is loadable, and not necessarily present in the system.
;  699          1225  2 !
;  700        P 1226  2 STATUS = $CREJNL (CHAN   = JRNL_CHANNEL,                 ! Channel returned by the service
;  701        P 1227  2                   JSB    = JSB,                        ! JSB address
;  702        P 1228  2                   ACMODE = 0,                          ! Access mode for channel
;  703        P 1229  2                   PROT   = 0,                          ! Protection for channel    (*)
;  704        P 1230  2                   FACCOD = 0,                          ! Addr of facility code     (*)
;  705        P 1231  2                   FLAGS  = 0,                          ! Option flags              (*)
;  706        P 1232  2                   OBJUIC = 0,                          ! Object UIC                (*)
;  707        P 1233  2                   SESSID = 0,                          ! Session ID                (*)
;  708        P 1234  2                   IOSB   = 0                           ! Internal IOSB             (*)
;  709          1235  2                  );
;  710          1236  2
;  711          1237  2 !
;  712          1238  2 ! Device already exists should be changed to a success code
;  713          1239  2 !
;  714          1240  2 IF .STATUS EQL CJF$_DEVEXI
```

```
;  715          1241  2 THEN
;  716          1242  2     STATUS[STS$V_SEVERITY] = STS$K_SUCCESS;
;  717          1243  2 !
;  718          1244  2 ! Deassign the journal channe..  The return status is not interesting.
;  719          1245  2 ! If the RUJ that was created was a temporary journal, deassigning the
;  720          1246  2 ! channel will get rid of the journal and the RUJ file.
;  721          1247  2 !
;  722          1248  2 $DEASJNL (CHAN = .JRNL_CHANNEL);
;  723          1249  2
;  724          1250  2 !
;  725          1251  2 ! If the status code returned by $CREJNL indicates an "informational" message,
;  726          1252  2 ! attempt to inform the user of the event.  All other status values are handled
;  727          1253  2 ! outside of this routine.  The status code is signalled, and a condition
;  728          1254  2 ! handler prints the message.
;  729          1255  2 !
;  730          1256  2 IF .STATUS[STS$V_SEVERITY] EQL STS$K_INFO
;  731          1257  2 THEN
;  732          1258  3     BEGIN
;  733          1259  3     VOLUME_NAME = GET_VOLUME_NAME (.DEVICE_NAME);
;  734          1260  3     ERR_MESSAGE (MOUN$_VOLSTATUS, 1, .VOLUME_NAME, .STATUS);
;  735          1261  2     END;
;  736          1262  2
;  737          1263  2 RETURN .STATUS
;  738          1264  2
;  739          1265  1 END;                                                    ! End of CREATE_JOURNAL


                                                .PSECT  $OWN$,NOEXE,2

                                        00138 JSB:      .BLKB   80

                                                .EXTRN  JRNL_EXTEND, JRNL_QUOTA
                                                .EXTRN  JRNL_SIZE, JRNL_RECORD_SIZE
                                                .EXTRN  CJF$CREJNL, CJF$DEASJNL

                                                .PSECT  $CODE$,NOWRT,2

                                        007C 00000 CREATE_RUJ:
                                                .WORD   Save R2,R3,R4,R5,R6                     ; 1127
                                56      0000'   CF  9E 00002    MOVAB   JSB, R6
                                6D      00B3    CF  DE 00007    MOVAL   5$, (FP)                ; 1164
                                                7E  D4 0000C    CLRL    JRNL_CHANNEL            ; 1194
0050    8F              00              6E      00  2C 0000E    MOVC5   #0, (SP), #0, #80, JSB  ; 1195
                                                66      00015
                        18      A6 00000000G    00  D0 00016    MOVL    JRNL_SIZE, JSB+24       ; 1197
                        1C      A6 00000000G    00  B0 0001E    MOVW    JRNL_EXTEND, JSB+28     ; 1198
                        14      A6 00000000G    00  B0 00026    MOVW    JRNL_RECORD_SIZE, JSB+20 ; 1199
                        20      A6 00000000G    00  D0 0002E    MOVL    JRNL_QUOTA, JSB+32      ; 1200
                        1E      A6              01  B0 00036    MOVW    #1, JSB+30              ; 1201
                        3A      A6              01  90 0003A    MOVB    #1, JSB+58              ; 1202
                        0A      A6      0101    8F  B0 0003E    MOVW    #257, JSB+10            ; 1196
                        24      A6              03  90 00044    MOVB    #3, JSB+36              ; 1204
                        44      A6      04      AC  D0 00048    MOVL    DEVICE_NAME, JSB+68     ; 1205
        08      FF      8F              6E      00  2C 0004D    MOVC5   #0, (SP), #-1, #8, JSB+60 ; 1206
                                        3C      A6      00053
                                0C      08      AC  E9 00055    BLBC    PERMANENT_RUJ, 1$       ; 1211
```

```
      2C    A6              01              03 00000000G   00   F0 00059          INSV    MOUNT_OPTIONS+7, #3, #1, JSB+44          ; 1213
                                                           04   11 00063          BRB     2$                                       ;
                                        2C    A6                11   88 00065 1$:      BISB2   #17, JSB+44                      ; 1217
                                                                 7E   7C 00069 2$:      CLRQ    -(SP)                            ; 1235
                                                                 7E   7C 0006B          CLRQ    -(SP)                            ;
                                                                 7E   7C 0006D          CLRQ    -(SP)                            ;
                                                                 7E   D4 0006F          CLRL    -(SP)                            ;
                                                                 56   DD 00071          PUSHL   R6                               ;
                                                    20    AE    9F   00073          PUSHAB  JRNL_CHANNEL                     ;
                                    00000000G       00          09   FB 00076          CALLS   #9, CJF$CREJNL                   ;
                                                    52          50   D0 0007D          MOVL    R0, STATUS                       ;
                                    00000000G       8F          52   D1 00080          CMPL    STATUS, #CJF$_DEVEXI             ; 1240
                                                                05   12 00087          BNEQ    3$                               ;
      52                03                          00          01   F0 00089          INSV    #1, #0, #3, STATUS               ; 1242
                                                                 7E   D4 0008E 3$:      CLRL    -(SP)                            ; 1248
                                                    04    AE    DD   00090          PUSHL   JRNL_CHANNEL                     ;
                                    00000000G       00          02   FB 00093          CALLS   #2, CJF$DEASJNL                  ;
      03                52                          03          00   ED 0009A          CMPZV   #0, #3, STATUS, #3               ; 1256
                                                                19   12 0009F          BNEQ    4$                               ;
                                                    04    AC    DD   000A1          PUSHL   DEVICE_NAME                      ; 1259
                                        0000V       CF          01   FB 000A4          CALLS   #1, GET_VOLUME_NAME              ;
                                                                05   BB 000A9          PUSHR   #^M<R0,R2>                       ; 1260
                                                                01   DD 000AB          PUSHL   #1                               ;
                                                  0072A07B      8F   DD 000AD          PUSHL   #7512187                         ;
                                    00000000G       00          04   FB 000B3          CALLS   #4, LIB$SIGNAL                   ;
                                                    50          52   D0 000BA 4$:      MOVL    STATUS, R0                       ; 1263
                                                                04   000BD          RET                                      ; 1265
                                                              0000   000BE 5$:      .WORD   Save nothing                     ; 1164
                                                                 7E   D4 000C0          CLRL    -(SP)                            ;
                                                                 5E   DD 000C2          PUSHL   SP                               ;
                                                    7E          04   AC   7D 000C4          MOVQ    4(AP), -(SP)                     ;
                                        0000V       CF          03   FB 000C8          CALLS   #3, INTERCEPT_SIGNAL             ;
                                                                04   000CD          RET                                      ;

; Routine Size:  206 bytes,    Routine Base:  $CODE$ + 022F
```

```
;  741          1266  1 ROUTINE DISMOUNT_VOLSET (DEVLIST, LIST_SIZE, INDEX, ENTIRE_VOLSET, ERROR_STATUS) : NOVALUE =
;  742          1267  1
;  743          1268  1 !++
;  744          1269  1 !
;  745          1270  1 ! FUNCTIONAL DESCRIPTION:
;  746          1271  1 !
;  747          1272  1 !       Dismount all or part of a volume set, and update the local device/volume
;  748          1273  1 !       data base to reflect the loss of one or more volumes.
;  749          1274  1 !
;  750          1275  1 ! CALLING SEQUENCE:
;  751          1276  1 !       DISMOUNT_VOLSET (ARG1, ARG2)
;  752          1277  1 !
;  753          1278  1 ! INPUT PARAMETERS:
;  754          1279  1 !       ARG1    : address of the list of all volumes mounted in this invocation of $MOUNT
;  755          1280  1 !       ARG2    : number of volumes mounted in this invocation of $MOUNT
;  756          1281  1 !       ARG3    : index into ARG1 (the current volume)
;  757          1282  1 !       ARG4    : if 0 dismount a single volume, otherwise dismount the entire volume set
;  758          1283  1 !       ARG5    : the journalling error status code
;  759          1284  1 !
;  760          1285  1 ! IMPLICIT INPUTS:
;  761          1286  1 !       <see the EXTERNAL declarations>
;  762          1287  1 !
;  763          1288  1 ! OUTPUT PARAMETERS:
;  764          1289  1 !       NONE
;  765          1290  1 !
;  766          1291  1 ! IMPLICIT OUTPUTS:
;  767          1292  1 !       NONE
;  768          1293  1 !
;  769          1294  1 ! ROUTINE VALUE:
;  770          1295  1 !       NONE.
;  771          1296  1 !
;  772          1297  1 ! SIDE EFFECTS:
;  773          1298  1 !       The internal device data base is updated to reflect the dismounting of the volume(s).
;  774          1299  1 !       The user is informed of the progress.
;  775          1300  1 !
;  776          1301  1 !--
;  777          1302  1
;  778          1303  2 BEGIN                                                   ! Start of CREATE_JOURNAL
;  779          1304  2
;  780          1305  2 !EXTERNAL LITERAL
;  781          1306  2 !       ROOTVOL_NAMLEN  = 64;                           ! Maximum length of a device name
;  782          1307  2
;  783          1308  2 !EXTERNAL
;  784          1309  2 !       AA$_DATA_BEGIN  : VECTOR [0],                   ! Mark start of data area
;  785          1310  2 !       DISMOUNT_COUNT  : BBLOCK [4],                   ! Count of volumes that were dismounted
;  786          1311  2 !       ODS2_VOLUME     : BITVECTOR [DEVMAX],           ! Indicates volume supports ODS-2 file struc
;  787          1312  2 !       ROOT_PRESENT    : BITVECTOR [DEVMAX],           ! Indicates presence of root volume for a gi
;  788          1313  2 !       RUJRNL_PRESENT  : BITVECTOR [DEVMAX],           ! Indicates presence of RUJ file on a given
;  789          1314  2 !       ROOTVOL_DSC     : BBLOCK [DSC$K_S_BLN],         ! Root volume device name descriptor
;  790          1315  2 !       ROOTVOL_BUF     : BBLOCK [ROOTVOL_NAMLEN],      ! Buffer for root volume device name
;  791          1316  2 !       ROOTVOL_INDEX   : BBLOCK [DEVMAX, BYTE, UNSIGNED],  ! Each entry contians index+1 of root volume
;  792          1317  2 !       VOLSET_RVN      : VECTOR [DEVMAX, BYTE, UNSIGNED],  ! Vector of Relative Volume Numbers
;  793          1318  2 !       ZZ$_DATA_END    : VECTOR [0];                   ! Mark end of data area
;  794          1319  2 !
;  795          1320  2 !EXTERNAL ROUTINE
;  796          1321  2 !       GET_VOLUME_NAME : ADDRESSING_MODE (GENERAL),    ! Return volume name for a given device
;  797          1322  2 !       INTERCEPT_SIGNAL: ADDRESSING_MODE (GENERAL);    ! Condition handler
```

```
;  798          1323  2
;  799          1324  2 LOCAL
;  800          1325  2         MULTI_DISMOUNT  : LONG,                                  ! Boolean, indicates more than one volume di
;  801          1326  2         PRIMARY_STATUS  : LONG,                                  ! Primary status message code
;  802          1327  2         ROOT_INDEX      : LONG,                                  ! Index into device list of root volume for
;  803          1328  2         VOLUME_NAME     : REF BBLOCK;                            ! Volume name descriptor
;  804          1329  2
;  805          1330  2 MAP
;  806          1331  2         DEVLIST         : REF BBLOCKVECTOR [DEVMAX, DSC$K_S_BLN];
;  807          1332  2
;  808          1333  2 OWN
;  809          1334  2         DISMOUNT_FLAGS  : BBLOCK [4],                            ! Flags for $DISMOU call
;  810          1335  2         ROOTDEV1_BUF    : BBLOCK [64],                           ! Buffer for primary root device name
;  811          1336  2         ROOTDEV1_DSC    : BBLOCK [DSC$K_S_BLN]                   ! Descriptor for primary root device name
;  812          1337  2                           PRESET ([DSC$A_POINTER] = ROOTDEV1_BUF),
;  813          1338  2         ROOTDEV2_BUF    : BBLOCK [64],                           ! Buffer for secondary root device name
;  814          1339  2         ROOTDEV2_DSC    : BBLOCK [DSC$K_S_BLN]                   ! Descriptor for secondary root device name
;  815          1340  2                           PRESET ([DSC$A_POINTER] = ROOTDEV2_BUF),
;  816          1341  2         VOLNAMLIST_BUF  : BBLOCK [DEVMAX*12],                    ! Buffer for volume names
;  817          1342  2         VOLNAMLIST_DSC  : BBLOCKVECTOR [DEVMAX, DSC$K_S_BLN],    ! Vector of volume name descriptors
;  818          1343  2         VOLSET_SIZE     : LONG,                                  ! Number of volumes in the volume set
;  819          1344  2         VOLSETNAM_BUF   : BBLOCK [12],                           ! Volume set name buffer
;  820          1345  2         VOLSETNAM_DSC   : BBLOCK [DSC$K_S_BLN],                  ! Volume set name descriptor
;  821          1346  2         VOLUME_REMOVED  : BITVECTOR [DEVMAX],                    ! Mark volume as being removed
;  822          1347  2
;  823          1348  2         PRIMARY_DVI     : BBLOCK [7*4]                           ! $GETDVI item list for volume set info
;  824          1349  2                           INITIAL (
;  825          1350  2                                    WORD (4),                   !    ITEM - Volume set size
;  826          1351  2                                    WORD (DVI$_VOLCOUNT),
;  827          1352  2                                    LONG (VOLSET_SIZE),
;  828          1353  2                                    LONG (0),
;  829          1354  2                                    WORD (64),                  !    ITEM - Volume set root device name
;  830          1355  2                                    WORD (DVI$_ROOTDEVNAM),
;  831          1356  2                                    LONG (ROOTDEV1_BUF),
;  832          1357  2                                    LONG (ROOTDEV1_DSC),
;  833          1358  2                                    LONG (0)                    ! List terminator
;  834          1359  2                                   ),
;  835          1360  2
;  836          1361  2         SECONDARY_DVI   : BBLOCK [4*4]                           ! $GETDVI item list for volume root device n
;  837          1362  2                           INITIAL (
;  838          1363  2                                    WORD (64),                  !    ITEM - Volume set root device name
;  839          1364  2                                    WORD (DVI$_ROOTDEVNAM),
;  840          1365  2                                    LONG (ROOTDEV2_BUF),
;  841          1366  2                                    LONG (ROOTDEV2_DSC),
;  842          1367  2                                    LONG (0)                    ! List terminator
;  843          1368  2                                   );
;  844          1369  2
;  845          1370  2
;  846          1371  2 !
;  847          1372  2 ! Enable a condition handler to field and print signalled messages.
;  848          1373  2 !
;  849          1374  2 ENABLE INTERCEPT_SIGNAL;
;  850          1375  2
;  851          1376  2 !
;  852          1377  2 ! For the specified device, fetch the root device name and the number
;  853          1378  2 ! of volumes in the volume set.
;  854          1379  2 !
```

```
;   855          1380  2 $GETDVI (DEVNAM=DEVLIST[.INDEX, DSC$W_LENGTH], ITMLST=PRIMARY_DVI);
;   856          1381  2
;   857          1382  2 !
;   858          1383  2 ! For each volume in the device list that is being dismounted,
;   859          1384  2 ! update the internal device data base to reflect this change of status.
;   860          1385  2 ! If only volume need be dismounted, this is very simple, otherwise...
;   861          1386  2 !
;   862          1387  2 IF NOT .ENTIRE_VOLSET
;   863          1388  2 OR .VOLSET_SIZE LEQ 1
;   864          1389  2 THEN
;   865          1390  3     BEGIN
;   866          1391  3     !
;   867          1392  3     ! Remove this volume from the data base, and increment the dismount count.
;   868          1393  3     !
;   869          1394  3     ODS2_VOLUME[.INDEX] = 0;
;   870          1395  3     ROOT_PRESENT[.INDEX] = 0;
;   871          1396  3     RUJRNL_PRESENT[.INDEX] = 0;
;   872          1397  3     VOLSET_RVN[.INDEX] = 0;
;   873          1398  3     ROOTVOL_INDEX[.INDEX] = 0;
;   874          1399  3     DISMOUNT_COUNT = .DISMOUNT_COUNT + 1;
;   875          1400  3     END
;   876          1401  2 ELSE
;   877          1402  3     BEGIN
;   878          1403  3     !
;   879          1404  3     ! ... otherwise, if the volume being dismounted is part of a volume
;   880          1405  3     ! set, each volume in the device list that is associated with the
;   881          1406  3     ! volume set must be removed from the device data base and dismounted.
;   882          1407  3     ! The process is complicated if the root volume is not present in the
;   883          1408  3     ! device list.  A ROOTVOL_INDEX of 0 implies that the root volume is
;   884          1409  3     ! not present in the list, and a string comparison is necessary.
;   885          1410  3     !
;   886          1411  3     VOLUME_REMOVED = 0;
;   887          1412  3     ROOT_INDEX = .ROOTVOL_INDEX[.INDEX];
;   888          1413  3     INCR J FROM 0 TO .LIST_SIZE-1 DO
;   889          1414  4         BEGIN
;   890          1415  4         !
;   891          1416  4         ! See if the root volume indexes match.  If so, further checks are mandated.
;   892          1417  4         !
;   893          1418  5         IF  (.ROOT_INDEX EQL .ROOTVOL_INDEX[.J])
;   894          1419  4         AND .ODS2_VOLUME[.J]
;   895          1420  4         THEN
;   896          1421  5             BEGIN
;   897          1422  5             !
;   898          1423  5             ! The indexes match.  If the number is nonzero, then the root volume has been mounted
;   899          1424  5             ! by this invocation of $MOUNT.  Otherwise, the root device name of the current device
;   900          1425  5             ! must be fetched and compared against the root device name supplied by the caller.
;   901          1426  5             !
;   902          1427  5             IF .ROOT_INDEX NEQ 0
;   903          1428  6             OR ($GETDVI (DEVNAM=DEVLIST[.J, DSC$W_LENGTH], ITMLST=SECONDARY_DVI);
;   904          1429  6                 IF CH$EQL (.ROOTDEV1_DSC[DSC$W_LENGTH],
;   905          1430  6                            .ROOTDEV1_DSC[DSC$A_POINTER],
;   906          1431  6                            .ROOTDEV2_DSC[DSC$W_LENGTH],
;   907          1432  6                            .ROOTDEV2_DSC[DSC$A_POINTER]
;   908          1433  6                           )
;   909          1434  6                 THEN
;   910          1435  6                     1
;   911          1436  6                 ELSE
```

```
;  912     1437  6                    0
;  913     1438  6                )
;  914     1439  5            THEN
;  915     1440  6                BEGIN
;  916     1441  6                !
;  917     1442  6                ! Remove this volume from the data base, and increment the dismount count.
;  918     1443  6                !
;  919     1444  6                ODS2_VOLUME[.J] = 0;
;  920     1445  6                ROOT_PRESENT[.J] = 0;
;  921     1446  6                RUJRNL_PRESENT[.J] = 0;
;  922     1447  6                VOLSET_RVN[.J] = 0;
;  923     1448  6                ROOTVOL_INDEX[.J] = 0;
;  924     1449  6                VOLUME_REMOVED[.J] = 1;
;  925     1450  6                DISMOUNT_COUNT = .DISMOUNT_COUNT + 1;
;  926     1451  6                !
;  927     1452  6                ! Get and save the volume name.  This must be done before dismounting the volume.
;  928     1453  6                !
;  929     1454  6                VOLUME_NAME = GET_VOLUME_NAME (DEVLIST[.J, DSC$W_LENGTH]);
;  930     1455  6                VOLNAMLIST_DSC[.J, DSC$W_LENGTH] = .VOLUME_NAME[DSC$W_LENGTH];
;  931     1456  6                VOLNAMLIST_DSC[.J, DSC$A_POINTER] = VOLNAMLIST_BUF + (12 * .J);
;  932     1457  6                CH$MOVE (.VOLUME_NAME[DSC$W_LENGTH], .VOLUME_NAME[DSC$A_POINTER], .VOLNAMLIST_DSC[.J, DSC$A_
;  933     1458  5                END;
;  934     1459  4            END;
;  935     1460  3        END;
;  936     1461  2    END;
;  937     1462  2
;  938     1463  2 !
;  939     1464  2 ! The informational message will vary depending on whether or not this is
;  940     1465  2 ! a volume set.  Determine the primary message code and the volume name.
;  941     1466  2 ! The volume name must be fetched before the volume is dismounted.
;  942     1467  2 ! A local copy of the volume set name is made for future reference.
;  943     1468  2 !
;  944     1469  2 IF (.VOLSET_SIZE GTR 1) AND .ENTIRE_VOLSET
;  945     1470  2 THEN
;  946     1471  3     BEGIN
;  947     1472  3     MULTI_DISMOUNT = 1;
;  948     1473  3     VOLUME_NAME = GET_VOLUME_NAME (DEVLIST[.INDEX, DSC$W_LENGTH], 1);
;  949     1474  3     VOLSETNAM_DSC[DSC$W_LENGTH] = .VOLUME_NAME[DSC$W_LENGTH];
;  950     1475  3     VOLSETNAM_DSC[DSC$A_POINTER] = VOLSETNAM_BUF;
;  951     1476  3     CH$MOVE (.VOLUME_NAME[DSC$W_LENGTH], .VOLUME_NAME[DSC$A_POINTER], .VOLSETNAM_DSC[DSC$A_POINTER]);
;  952     1477  3     PRIMARY_STATUS = MOUN$_VOLSETSTS;
;  953     1478  3     END
;  954     1479  2 ELSE
;  955     1480  3     BEGIN
;  956     1481  3     MULTI_DISMOUNT = 0;
;  957     1482  3     VOLUME_NAME = GET_VOLUME_NAME (DEVLIST[.INDEX, DSC$W_LENGTH]);
;  958     1483  3     PRIMARY_STATUS = MOUN$_VOLSTATUS;
;  959     1484  2     END;
;  960     1485  2
;  961     1486  2 !
;  962     1487  2 ! Dismount all or part of the volume set, as specified by the caller.
;  963     1488  2 !
;  964     1489  2 DISMOUNT_FLAGS = 0;
;  965     1490  2 IF NOT .ENTIRE_VOLSET
;  966     1491  2 THEN
;  967     1492  2     DISMOUNT_FLAGS[DMT$V_UNIT] = 1;
;  968     1493  2 $DISMOU (DEVNAM=DEVLIST[.INDEX, DSC$W_LENGTH], FLAGS=.DISMOUNT_FLAGS);
```

```
;  969          1494  2
;  970          1495  2 !
;  971          1496  2 ! Inform the user that at least one volume has been dismounted.
;  972          1497  2 ! Include the status code specified by the caller.
;  973          1498  2 !
;  974          1499  2 ERR_MESSAGE (.PRIMARY_STATUS, 1, .VOLUME_NAME, MOUN$_CJFERR, 0, .ERROR_STATUS);
;  975          1500  2
;  976          1501  2 !
;  977          1502  2 ! If more than one volume was dismounted, inform the user of each volume
;  978          1503  2 ! that was dismounted.
;  979          1504  2 !
;  980          1505  2 IF .MULTI_DISMOUNT
;  981          1506  2 THEN
;  982          1507  2     INCR J FROM 0 TO .LIST_SIZE-1 DO
;  983          1508  2         IF .VOLUME_REMOVED[.J]
;  984          1509  2         THEN
;  985          1510  3             BEGIN
;  986          1511  3             VOLUME_NAME = GET_VOLUME_NAME (DEVLIST[.J, DSC$W_LENGTH]);
;  987          1512  3             ERR_MESSAGE (MOUN$_DISMOUNTD, 2, VOLNAMLIST_DSC[.J, DSC$W_LENGTH], VOLSETNAM_DSC);
;  988          1513  2             END;
;  989          1514  2
;  990          1515  1 END;                                                    ! End of DISMOUNT_VOLSET


                                                .PSECT  $OWN$,NOEXE,2

                                        00188 DISMOUNT_FLAGS:
                                                .BLKB   4
                                        0018C ROOTDEV1_BUF:
                                                .BLKB   64
                                      00# 001CC ROOTDEV1_DSC:
                                                .BYTE   0[4]                    ;
                                00000000' 001D0         .ADDRESS ROOTDEV1_BUF   ;
                                        001D4 ROOTDEV2_BUF:
                                                .BLKB   64
                                      00# 00214 ROOTDEV2_DSC:
                                                .BYTE   0[4]                    ;
                                00000000' 00218         .ADDRESS ROOTDEV2_BUF   ;
                                        0021C VOLNAMLIST_BUF:
                                                .BLKB   192                     ;
                                        002DC VOLNAMLIST_DSC:
                                                .BLKB   128                     ;
                                        0035C VOLSET_SIZE:
                                                .BLKB   4                       ;
                                        00360 VOLSETNAM_BUF:
                                                .BLKB   12                      ;
                                        0036C VOLSETNAM_DSC:
                                                .BLKB   8                       ;
                                        00374 VOLUME_REMOVED:
                                                .BLKB   2                       ;
                                        00376           .BLKB   2               ;
                                    0004  00378 PRIMARY_DVI:
                                                .WORD   4                       ;
                                    0030  0037A         .WORD   48              ;
                                00000000' 0037C         .ADDRESS VOLSET_SIZE    ;
                                00000000  00380         .LONG   0               ;
```

```
                    0040  00384          .WORD   64                                                      ;
                    0032  00386          .WORD   50                                                      ;
                00000000' 00388          .ADDRESS ROOTDEV1_BUF                                           ;
                00000000' 0038C          .ADDRESS ROOTDEV1_DSC                                           ;
                00000000  00390          .LONG   0                                                       ;
                    0040  00394 SECONDARY_DVI:
                                         .WORD   64                                                      ;
                    0032  00396          .WORD   50                                                      ;
                00000000' 00398          .ADDRESS ROOTDEV2_BUF                                           ;
                00000000' 0039C          .ADDRESS ROOTDEV2_DSC                                           ;
                00000000  003A0          .LONG   0                                                       ;

                                         .EXTRN  SYS$DISMOU, SYS$CMKRNL

                                         .PSECT  $CODE$,NOWRT,2

                    0FFC 00000 DISMOUNT_VOLSET:
                                         .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11                    ; 1266
5B 00000000G   00    9E 00002            MOVAB   SYS$GETDVI, R11                                         ;
5A      0000'  CF    9E 00009            MOVAB   ROOTVOL_INDEX, R10                                      ;
6D      0196   CF    DE 0000E            MOVAL   21$, (FP)                                               ; 1340
               7E    7C 00013            CLRQ    -(SP)                                                   ; 1380
               7E    7C 00015            CLRQ    -(SP)                                                   ;
          0320 CA    9F 00017            PUSHAB  PRIMARY_DVI                                             ;
52          0C AC    D0 0001B            MOVL    INDEX, R2                                               ;
59        04 BC42    7E 0001F            MOVAQ   @DEVLIST[R2], R9                                        ;
               59    DD 00024            PUSHL   R9                                                      ;
               7E    7C 00026            CLRQ    -(SP)                                                   ;
6B                   08 FB 00028         CALLS   #8, SYS$GETDVI                                          ;
07          10   AC    E9 0002B         BLBC    ENTIRE_VOLSET, 1$                                       ; 1387
01        0304   CA    D1 0002F         CMPL    VOLSET_SIZE, #1                                         ; 1388
               1C    14 00034            BGTR    5$                                                      ;
      00    AC    AA       52 E5 00036 1$: BBCC    R2, ODS2_VOLUME, 2$                                     ; 1394
      00    B0    AA       52 E5 0003B 2$: BBCC    R2, ROOT_PRESENT, 3$                                    ; 1395
      00    B4    AA       52 E5 00040 3$: BBCC    R2, RUJRNL_PRESENT, 4$                                  ; 1396
            10 AA42    94 00045 4$:        CLRB    VOLSET_RVN[R2]                                          ; 1397
               6A42    94 00049            CLRB    ROOTVOL_INDEX[R2]                                       ; 1398
            A8   AA    D6 0004C            INCL    DISMOUNT_COUNT                                          ; 1399
               0099    31 0004F            BRW     14$                                                     ; 1387
          031C CA    B4 00052 5$:        CLRW    VOLUME_REMOVED                                          ; 1411
58                 6A42    9A 00056         MOVZBL  ROOTVOL_INDEX[R2], ROOT_INDEX                           ; 1412
56                   01    CE 0005A         MNEGL   #1, J                                                   ; 1418
               0081    31 0005D            BRW     12$                                                     ;
   58        6A46          08 00060 6$:        CMPZV   #0, #8, ROOTVOL_INDEX[J], ROOT_INDEX                ;
               79    12 00066            BNEQ    12$                                                     ;
      74    AC    AA       56 E1 00068         BBC     J, ODS2_VOLUME, 12$                             ; 1419
               58    D5 0006D            TSTL    ROOT_INDEX                                              ; 1427
               21    12 0006F            BNEQ    7$                                                      ;
               7E    7C 00071            CLRQ    -(SP)                                                   ; 1428
               7E    7C 00073            CLRQ    -(SP)                                                   ;
          033C CA    9F 00075            PUSHAB  SECONDARY_DVI                                           ;
            04 BC46    7F 00079         PUSHAQ  @DEVLIST[J]                                             ;
               7E    7C 0007D            CLRQ    -(SP)                                                   ;
6B                   08 FB 0007F         CALLS   #8, SYS$GETDVI                                          ;
01BC CA    00    0178 DA    0174 CA    2D 00082 CMPC5   ROOTDEV1_DSC, @ROOTDEV1_DSC+4, #0, -           ; 1429
                        01C0 DA          0008D                ROOTDEV2_DSC, @ROOTDEV2_DSC+4              ;
               4F    12 00090            BNEQ    12$                                                     ;
```

```
                00      AC      AA              56  E5 00092 7$:     BBCC    J, ODS2_VOLUME, 8$                              ; 1444
                00      B0      AA              56  E5 00097 8$:     BBCC    J, ROOT_PRESENT, 9$                             ; 1445
                00      B4      AA              56  E5 0009C 9$:     BBCC    J, RUJRNL_PRESENT, 10$                          ; 1446
                                        10 AA46 94  00A1 10$:     CLRB    VOLSET_RVN[J]                                   ; 1447
                                           6A46 94  00A5          CLRB    ROOTVOL_INDEX[J]                                ; 1448
                00      031C    CA              56  E2 00A8          BBSS    J, VOLUME_REMOVED, 11$                          ; 1449
                                        A8      AA  D6 00AE 11$:     INCL    DISMOUNT_COUNT                                  ; 1450
                                        04 BC46 7F  00CB1          PUSHAQ  @DEVLIST[J]                                     ; 1454
                        0000V   CF              01  FB 00CB5          CALLS   #1, GET_VOLUME_NAME                             ;
                        57              50  D0 00CBA          MOVL    R0, VOLUME_NAME                                 ;
                                    0284 CA46 7F  00CBD          PUSHAQ  VOLNAMLIST_DSC[J]                               ; 1455
                        9E              67  B0 00CC2          MOVW    (VOLUME_NAME), @(SP)+                           ;
                50              56          0C  C5 00CC5          MULL3   #12, J, R0                                      ; 1456
                                    0288 CA46 7F  00CC9          PUSHAQ  VOLNAMLIST_DSC+4[J]                             ;
                        9E      01C4 CA40 9E  00CCE          MOVAB   VOLNAMLIST_BUF[R0], @(SP)+                      ;
                                    0288 CA46 7F  00CD4          PUSHAQ  VOLNAMLIST_DSC+4[J]                             ; 1457
                        50              9E  D0 00CD9          MOVL    @(SP)+, R0                                      ;
                60      04      B7          67  28 00CDC          MOVC3   (VOLUME_NAME), @4(VOLUME_NAME), (R0)            ;
                02              56      08  AC  F2 00CE1 12$:     AOBLSS  LIST_SIZE, J, 13$                               ; 1413
                                            03  11 00CE6          BRB     14$                                             ;
                                        FF75    31 00CE8 13$:     BRW     6$                                              ;
                        01      0304    CA  D1 00CEB 14$:     CMPL    VOLSET_SIZE, #1                                 ; 1469
                                        2F  15 00CF0          BLEQ    15$                                             ;
                        2B      10      AC  E9 00CF2          BLBC    ENTIRE_VOLSET, 15$                              ;
                        58              01  D0 00CF6          MOVL    #1, MULTI_DISMOUNT                              ; 1472
                                        01  DD 00CF9          PUSHL   #1                                              ; 1473
                                        59  DD 00CFB          PUSHL   R9                                              ;
                        0000V   CF              02  FB 00CFD          CALLS   #2, GET_VOLUME_NAME                             ;
                        57              50  D0 00102          MOVL    R0, VOLUME_NAME                                 ;
                0314    CA              67  B0 00105          MOVW    (VOLUME_NAME), VOLSETNAM_DSC                    ; 1474
                0318    CA      0308    CA  9E 0010A          MOVAB   VOLSETNAM_BUF, VOLSETNAM_DSC+4                  ; 1475
        0318    DA      04      B7          67  28 00111          MOVC3   (VOLUME_NAME), @4(VOLUME_NAME), -               ; 1476
                                                                        @VOLSETNAM_DSC+4                                ;
                        52 0072A083     8F  D0 00118          MOVL    #7512195, PRIMARY_STATUS                        ; 1477
                                        13  11 0011F          BRB     16$                                             ; 1469
                                        58  D4 00121 15$:     CLRL    MULTI_DISMOUNT                                  ; 1481
                                        59  DD 00123          PUSHL   R9                                              ; 1482
                        0000V   CF              01  FB 00125          CALLS   #1, GET_VOLUME_NAME                             ;
                        57              50  D0 0012A          MOVL    R0, VOLUME_NAME                                 ;
                        52 0072A07B     8F  D0 0012D          MOVL    #7512187, PRIMARY_STATUS                        ; 1483
                                0130    CA  D4 00134 16$:     CLRL    DISMOUNT_FLAGS                                  ; 1489
                        05      10      AC  E8 00138          BLBS    ENTIRE_VOLSET, 17$                              ; 1490
                0130    CA              02  88 0013C          BISB2   #2, DISMOUNT_FLAGS                              ; 1492
                                0130    CA  DD 00141 17$:     PUSHL   DISMOUNT_FLAGS                                  ; 1493
                                        59  DD 00145          PUSHL   R9                                              ;
                                        02  DD 00147          PUSHL   #2                                              ;
                                        5E  DD 00149          PUSHL   SP                                              ;
                        00000000G       00  9F 0014B          PUSHAB  SYS$DISMOU                                      ;
        00000000G       9F              05  FB 00151          CALLS   #5, @#SYS$CMKRNL                                ;
                                    14      AC  DD 00158          PUSHL   ERROR_STATUS                                    ; 1499
                                        7E  D4 0015B          CLRL    -(SP)                                           ;
                        00729058        8F  DD 0015D          PUSHL   #7508056                                        ;
                                        57  DD 00163          PUSHL   VOLUME_NAME                                     ;
                                        01  DD 00165          PUSHL   #1                                              ;
                                        52  DD 00167          PUSHL   PRIMARY_STATUS                                  ;
        00000000G       00              06  FB 00169          CALLS   #6, LIB$SIGNAL                                  ;
                        34              58  E9 00170          BLBC    MULTI_DISMOUNT, 20$                             ; 1505
```

```
                                52              01  CE 00173          MNEGL   #1, J                                  ; 1507
                                                2A  11 00176          BRB     19$                                     ;
                        24      031C    CA      52  E1 00178 18$:     BBC     J, VOLUME_REMOVED, 19$                  ; 1508
                                            04  BC42  7F 0017E          PUSHAQ  @DEVLIST[J]                             ; 1511
                                0000V   CF          01  FB 00182          CALLS   #1, GET_VOLUME_NAME                     ;
                                57              50  D0 00187          MOVL    R0, VOLUME_NAME                         ;
                                            0314    CA  9F 0018A          PUSHAB  VOLSETNAM_DSC                           ; 1512
                                            0284 CA42  7F 0018E          PUSHAQ  VOLNAMLIST_DSC[J]                       ;
                                                02  DD 00193          PUSHL   #2                                      ;
                                        00729060    8F  DD 00195          PUSHL   #7508064                            ;
                            00000000G   00          04  FB 0019B          CALLS   #4, LIB$SIGNAL                      ;
                        D1              52      08  AC  F2 001A2 19$:     AOBLSS  LIST_SIZE, J, 18$                   ; 1508
                                                    04 001A7 20$:         RET                                         ; 1515
                                                  0000 001A8 21$:         .WORD   Save nothing                        ; 1340
                                                7E  D4 001AA          CLRL    -(SP)                                   ;
                                                5E  DD 001AC          PUSHL   SP                                      ;
                                7E          04  AC  7D 001AE          MOVQ    4(AP), -(SP)                            ;
                                0000V   CF          03  FB 001B2          CALLS   #3, INTERCEPT_SIGNAL                ;
                                                    04 001B7          RET                                             ;

; Routine Size:  440 bytes,    Routine Base:  $CODE$ + 02FD
```

```
;  992     1516  1 ROUTINE GET_VOLUME_NAME (DEVICE_NAME, FLAG) =
;  993     1517  1
;  994     1518  1 !++
;  995     1519  1 !
;  996     1520  1 ! FUNCTIONAL DESCRIPTION:
;  997     1521  1 !
;  998     1522  1 !       Given the address of a device name descriptor, return the address of
;  999     1523  1 !       a string descriptor for the volume mounted in that device.  The volume
; 1000     1524  1 !       name is from 1 to 12 characters.  Since the address returned points to
; 1001     1525  1 !       routine-local storage, the string must be copied by the caller to prevent
; 1002     1526  1 !       the next invocation of this routine from overwriting the result.
; 1003     1527  1 !
; 1004     1528  1 ! CALLING SEQUENCE:
; 1005     1529  1 !       GET_VOLUME_NAME (ARG1)
; 1006     1530  1 !
; 1007     1531  1 ! INPUT PARAMETERS:
; 1008     1532  1 !       ARG1    : address of device name descriptor
; 1009     1533  1 !       ARG2    : if present, indicates that the volume set name should be returned.
; 1010     1534  1 !
; 1011     1535  1 ! IMPLICIT INPUTS:
; 1012     1536  1 !       <see the EXTERNAL declarations>
; 1013     1537  1 !
; 1014     1538  1 ! OUTPUT PARAMETERS:
; 1015     1539  1 !       NONE
; 1016     1540  1 !
; 1017     1541  1 ! IMPLICIT OUTPUTS:
; 1018     1542  1 !       NONE
; 1019     1543  1 !
; 1020     1544  1 ! ROUTINE VALUE:
; 1021     1545  1 !       Address of the volume name descriptor.
; 1022     1546  1 !       Return 0 if the device does not contain a volume.
; 1023     1547  1 !
; 1024     1548  1 ! SIDE EFFECTS:
; 1025     1549  1 !       NONE.
; 1026     1550  1 !
; 1027     1551  1 !--
; 1028     1552  1
; 1029     1553  2 BEGIN                                                   ! Start of GET_VOLUME_NAME
; 1030     1554  2
; 1031     1555  2
; 1032     1556  2 !
; 1033     1557  2 ! Allocate plits in the $CODE$ psect to avoid truncation error when
; 1034     1558  2 ! linking mountshr.
; 1035     1559  2 !
; 1036     1560  2 PSECT
; 1037     1561  2         PLIT = $CODE$;
; 1038     1562  2
; 1039     1563  2 BIND
; 1040     1564  2         NO_VOLSET_NAME  = DESCRIPTOR ('<name n/a>'),
; 1041     1565  2         NAME_PREFIX     = DESCRIPTOR ('DISK$');
; 1042     1566  2
; 1043     1567  2 BUILTIN
; 1044     1568  2         ACTUALCOUNT;
; 1045     1569  2
; 1046     1570  2 MAP
; 1047     1571  2         NAME_PREFIX     : BBLOCK;
; 1048     1572  2
```

```
;  1049          1573  2 OWN
;  1050          1574  2         LOGNAM_BUF      : BBLOCK [64],                          ! Buffer for volume logical name
;  1051          1575  2         LOGNAM_DSC      : BBLOCK [DSC$K_S_BLN]                  ! Descriptor for logical name string
;  1052          1576  2                           PRESET ([DSC$A_POINTER] = LOGNAM_BUF),
;  1053          1577  2         VOLNAM_BUF      : BBLOCK [12],                          ! Buffer for volume name string
;  1054          1578  2         VOLNAM_DSC      : BBLOCK [DSC$K_S_BLN]                  ! Descriptor for volume name string
;  1055          1579  2                           PRESET ([DSC$A_POINTER] = VOLNAM_BUF),
;  1056          1580  2         VOLSETNAM_DSC   : BBLOCK [DSC$K_S_BLN],                 ! Decriptor for isolated volume set name
;  1057          1581  2
;  1058          1582  2         VOLUME_DVI      : BBLOCK [7*4]                          ! $GETDVI item list for volume info
;  1059          1583  2                           INITIAL (
;  1060          1584  2                                    WORD (12),                 !    ITEM - Volume name
;  1061          1585  2                                    WORD (DVI$_LOGVOLNAM),
;  1062          1586  2                                    LONG (LOGNAM_BUF),
;  1063          1587  2                                    LONG (LOGNAM_DSC),
;  1064          1588  2                                    WORD (12),                 !    ITEM - Volume name
;  1065          1589  2                                    WORD (DVI$_VOLNAM),
;  1066          1590  2                                    LONG (VOLNAM_BUF),
;  1067          1591  2                                    LONG (VOLNAM_DSC),
;  1068          1592  2                                    LONG (0)                   ! List terminator
;  1069          1593  2                                   );
;  1070          1594  2
;  1071          1595  2
;  1072          1596  2 !
;  1073          1597  2 ! Get the requested information.
;  1074          1598  2 ! If $GETDVI failied, return 0.
;  1075          1599  2 !
;  1076          1600  3 IF NOT $GETDVI (DEVNAM=.DEVICE_NAME, ITMLST=VOLUME_DVI)
;  1077          1601  2 THEN
;  1078          1602  2     RETURN 0;
;  1079          1603  2
;  1080          1604  2 !
;  1081          1605  2 ! If the volume set name was requested, strip off the system-provided
;  1082          1606  2 ! prefix before returning it to the user.  If no prefix exists, then
;  1083          1607  2 ! this is not a volume set, and a special message indicating that the
;  1084          1608  2 ! volume set name is not available is returned instead.
;  1085          1609  2 !
;  1086          1610  2 IF ACTUALCOUNT() GTR 1
;  1087          1611  2 THEN
;  1088          1612  2     IF CH$EQL  (.NAME_PREFIX[DSC$W_LENGTH],
;  1089          1613  2                 .NAME_PREFIX[DSC$A_POINTER],
;  1090          1614  2                 .NAME_PREFIX[DSC$W_LENGTH],
;  1091          1615  2                 .LOGNAM_DSC[DSC$A_POINTER]
;  1092          1616  2                )
;  1093          1617  2     THEN
;  1094          1618  3         BEGIN
;  1095          1619  3         VOLSETNAM_DSC[DSC$W_LENGTH] = .LOGNAM_DSC[DSC$W_LENGTH] - .NAME_PREFIX[DSC$W_LENGTH];
;  1096          1620  3         VOLSETNAM_DSC[DSC$A_POINTER] = .LOGNAM_DSC[DSC$A_POINTER] + .NAME_PREFIX[DSC$W_LENGTH];
;  1097          1621  3         RETURN VOLSETNAM_DSC;
;  1098          1622  3         END
;  1099          1623  2     ELSE
;  1100          1624  2         RETURN NO_VOLSET_NAME;
;  1101          1625  2
;  1102          1626  2 RETURN VOLNAM_DSC
;  1103          1627  2
;  1104          1628  1 END;                                                           ! End of GET_VOLUME_NAME
```

```
                                                004B5 P.AAD:    .ASCII  \<name n/a>\                     ;
                                                004BF           .BLKB   1
                                        0000000A 004C0 P.AAC:   .LONG   10                              ;
                                        00000000' 004C4         .ADDRESS P.AAD                          ;
                                    24  4B 53 49 44 004C8 P.AAF: .ASCII  \DISK$\                        ;
                                                004CD           .BLKB   3
                                        00000005 004D0 P.AAE:   .LONG   5                               ;
                                        00000000' 004D4         .ADDRESS P.AAF                          ;

                                                                .PSECT  $OWN$,NOEXE,2

                                                003A4 LOGNAM_BUF:
                                                                .BLKB   64
                                              00# 003E4 LOGNAM_DSC:
                                                                .BYTE   0[4]
                                        00000000' 003E8         .ADDRESS LOGNAM_BUF                     ;
                                                003EC VOLNAM_BUF:
                                                                .BLKB   12
                                              00# 003F8 VOLNAM_DSC:
                                                                .BYTE   0[4]
                                        00000000' 003FC         .ADDRESS VOLNAM_BUF                     ;
                                                00400 VOLSETNAM_DSC:
                                                                .BLKB   8
                                            000C  00408 VOLUME_DVI:
                                                                .WORD   12                              ;
                                            002C  0040A         .WORD   44                              ;
                                        00000000' 0040C         .ADDRESS LOGNAM_BUF                     ;
                                        00000000' 00410         .ADDRESS LOGNAM_DSC                     ;
                                            000C  00414         .WORD   12                              ;
                                            0022  00416         .WORD   34                              ;
                                        00000000' 00418         .ADDRESS VOLNAM_BUF                     ;
                                        00000000' 0041C         .ADDRESS VOLNAM_DSC                     ;
                                        00000000  00420         .LONG   0                               ;

                                                        NO_VOLSET_NAME=     P.AAC
                                                        NAME_PREFIX=        P.AAE


                                                                .PSECT  $CODE$,NOWRT,2

                                                  003C 00000 GET_VOLUME_NAME:
                                                                .WORD   Save R2,R3,R4,R5                ; 1516
                                    55      0000' CF  9E 00002  MOVAB   LOGNAM_DSC+4, R5                ; 1600
                                                  7E  7C 00007  CLRQ    -(SP)                           ;
                                                  7E  7C 00009  CLRQ    -(SP)                           ;
                                              20  A5  9F 0000B  PUSHAB  VOLUME_DVI                      ;
                                              04  AC  DD 0000E  PUSHL   DEVICE_NAME                     ;
                                                  7E  7C 00011  CLRQ    -(SP)                           ;
                            00000000G     00        08  FB 00013  CALLS   #8, SYS$GETDVI               ;
                                          2B        50  E9 0001A  BLBC    R0, 3$                        ;
                                          01        6C  91 0001D  CMPB    (AP), #1                      ; 1610
                                                    21  1B 00020  BLEQU   2$                            ;
                                          54    D3  AF  3C 00022  MOVZWL  NAME_PREFIX, R4               ; 1612
                00    B5        D2      BF        54  29 00026  CMPC3   R4, @NAME_PREFIX+4, @LOGNAM_DSC+4 ;
                                                    10  12 0002C  BNEQ    1$                            ;
```

```
                18   A5        FC   A5              54   A3 0002E          SUBW3   R4, LOGNAM_DSC, VOLSETNAM_DSC                   ; 1619
                1C   A5             65              54   C1 00034          ADDL3   R4, LOGNAM_DSC+4, VOLSETNAM_DSC+4           ; 1620
                                     50        18    A5   9E 00039          MOVAB   VOLSETNAM_DSC, R0                       ; 1621
                                                      04   0003D          RET                                             ; 1624
                                     50        A7    AF   9E 0003E 1$:     MOVAB   NO_VOLSET_NAME, R0                      ;
                                                      04   00042          RET                                             ;
                                     50        10    A5   9E 00043 2$:     MOVAB   VOLNAM_DSC, R0                          ; 1626
                                                      04   00047          RET                                             ;
                                               50    D4   00048 3$:     CLRL    R0                                      ; 1628
                                                      04   0004A          RET                                             ;

; Routine Size:  75 bytes,    Routine Base:  $CODE$ + 04D8
```

```
;  1106          1629  1 ROUTINE INTERCEPT_SIGNAL (SIGNAL, MECHANISM) =
;  1107          1630  1
;  1108          1631  1 !++
;  1109          1632  1 ! Functional Description:
;  1110          1633  1 !
;  1111          1634  1 !       This routine is a conditon handler whose sole
;  1112          1635  1 !       reason for existence is to force the primary
;  1113          1636  1 !       conditon code's facility-code to that of the
;  1114          1637  1 !       MOUNT facility.
;  1115          1638  1 !
;  1116          1639  1 ! Input:
;  1117          1640  1 !
;  1118          1641  1 !       SIGNAL    = Address of the signal array
;  1119          1642  1 !       MECHANISM = Address of the mechanism array
;  1120          1643  1 !
;  1121          1644  1 ! Output:
;  1122          1645  1 !
;  1123          1646  1 !       The condition facility code is equal to MOUN$_FACILITY
;  1124          1647  1 !--
;  1125          1648  1
;  1126          1649  2 BEGIN                                           ! Start of INTERCEPT_SIGNAL
;  1127          1650  2
;  1128          1651  2 MAP
;  1129          1652  2
;  1130          1653  2         SIGNAL          : REF BBLOCK,          ! Signal array
;  1131          1654  2         MECHANISM       : REF BBLOCK;          ! Mechanism array
;  1132          1655  2
;  1133          1656  2 EXTERNAL
;  1134          1657  2
;  1135          1658  2         MOUNT_OPTIONS   : BITVECTOR VOLATILE,  ! parser option flags
;  1136          1659  2         USER_STATUS     : VECTOR;              ! Status return of some routines
;  1137          1660  2
;  1138          1661  2
;  1139          1662  2 IF .SIGNAL[CHF$L_SIG_NAME] NEQ SS$_UNWIND
;  1140          1663  2 THEN
;  1141          1664  3     BEGIN
;  1142          1665  3     !
;  1143          1666  3     ! Make the facility code MOUN$_FCILITY.
;  1144          1667  3     !
;  1145          1668  3     IF .BBLOCK [SIGNAL[CHF$L_SIG_NAME], STS$V_FAC_NO] EQL 0
;  1146          1669  3     THEN
;  1147          1670  3         BBLOCK [SIGNAL[CHF$L_SIG_NAME], STS$V_FAC_NO] = MOUN$_FACILITY;
;  1148          1671  3
;  1149          1672  3     IF .BBLOCK [SIGNAL[CHF$L_SIG_NAME], STS$V_MSG_NO] EQL 0
;  1150          1673  3     THEN
;  1151          1674  3         BBLOCK [SIGNAL[CHF$L_SIG_NAME], STS$V_MSG_NO] = .USER_STATUS [0] ^ (-$BITPOSITION (STS$V_MSG_NO));
;  1152          1675  3
;  1153          1676  3     !
;  1154          1677  3     ! If the caller requested it, print the message text associated with the message.
;  1155          1678  3     !
;  1156          1679  3     IF .MOUNT_OPTIONS [OPT_MESSAGE]
;  1157          1680  3     THEN
;  1158          1681  4         BEGIN
;  1159          1682  4         SIGNAL [CHF$L_SIG_ARGS] = .SIGNAL [CHF$L_SIG_ARGS] - 2;
;  1160          1683  4         $PUTMSG (MSGVEC = SIGNAL [CHF$L_SIG_ARGS], ACTRTN=0, FACNAM=0);
;  1161          1684  4         SIGNAL [CHF$L_SIG_ARGS] = .SIGNAL [CHF$L_SIG_ARGS] + 2;
;  1162          1685  4         BBLOCK [SIGNAL [CHF$L_SIG_NAME], STS$V_INHIB_MSG] = 1;
```

```
; 1163      1686  3             END;
; 1164      1687  3
; 1165      1688  3         !
; 1166      1689  3         ! If the condition severity code is SEVERE or ERROR, then unwind the
; 1167      1690  3         ! stack back to the caller of the frame that established this handler.
; 1168      1691  3         ! Return the condition code in R0.
; 1169      1692  3         !
; 1170      1693  3         IF .BBLOCK [SIGNAL [CHF$L_SIG_NAME], STS$V_SEVERITY] EQL STS$K_SEVERE
; 1171      1694  3         OR .BBLOCK [SIGNAL [CHF$L_SIG_NAME], STS$V_SEVERITY] EQL STS$K_ERROR
; 1172      1695  3         THEN
; 1173      1696  4             BEGIN
; 1174      1697  4             MECHANISM [CHF$L_MCH_SAVR0] = .SIGNAL [CHF$L_SIG_NAME];
; 1175      1698  4             $UNWIND ();
; 1176      1699  3             END;
; 1177      1700  2         END;
; 1178      1701  2
; 1179      1702  2     !
; 1180      1703  2     ! Attempt to continue the operation.
; 1181      1704  2     !
; 1182      1705  2     RETURN (SS$_CONTINUE);
; 1183      1706  2
; 1184      1707  1     END;                                    ! End of INTERCEPT_SIGNAL


                                                        .EXTRN  USER_STATUS, SYS$PUTMSG
                                                        .EXTRN  SYS$UNWIND

                                   000C 00000 INTERCEPT_SIGNAL:
                                                        .WORD   Save R2,R3                        ; 1629
                         52     04  AC  D0 00002        MOVL    SIGNAL, R2                        ; 1662
                         53     04  A2  9E 00006        MOVAB   4(R2), R3
              00000920   8F         63  D1 0000A        CMPL    (R3), #2336                       ;
                                    65  13 00011        BEQL    5$                                ;
                  0FFF   8F     02  A3  B3 00013        BITW    2(R3), #4095                      ; 1668
                                    0A  12 00019        BNEQ    1$                                ;
  02   A3       0C       00 00000072 8F F0 0001B        INSV    #114, #0, #12, 2(R3)              ; 1670
                  FFF8   8F         63  B3 00025 1$:    BITW    (R3), #65528                      ; 1672
                                    0E  12 0002A        BNEQ    2$                                ;
            50 00000000G 00     FD  8F  78 0002C        ASHL    #-3, USER_STATUS, R0              ; 1674
       63       0D       03         50  F0 00035        INSV    R0, #3, #13, (R3)                 ;
            17 00000000G 00         03  E1 0003A 2$:    BBC     #3, MOUNT_OPTIONS+6, 3$           ; 1679
                         62         02  C2 00042        SUBL2   #2, (R2)                          ; 1682
                                    7E  7C 00045        CLRQ    -(SP)                             ; 1683
                                    7E  D4 00047        CLRL    -(SP)                             ;
                                    52  DD 00049        PUSHL   R2                                ;
               00000000G 00         04  FB 0004B        CALLS   #4, SYS$PUTMSG                    ;
                         62         02  C0 00052        ADDL2   #2, (R2)                          ; 1684
                    03   A3         10  88 00055        BISB2   #16, 3(R3)                        ; 1685
       04       63       03         00  ED 00059 3$:    CMPZV   #0, #3, (R3), #4                  ; 1693
                                    07  13 0005E        BEQL    4$                                ;
       02       63       03         00  ED 00060        CMPZV   #0, #3, (R3), #2                  ; 1694
                                    11  12 00065        BNEQ    5$                                ;
                         50     08  AC  D0 00067 4$:    MOVL    MECHANISM, R0                     ; 1697
                    0C   A0         63  D0 0006B        MOVL    (R3), 12(R0)                      ;
                                    7E  7C 0006F        CLRQ    -(SP)                             ; 1698
               00000000G 00         02  FB 00071        CALLS   #2, SYS$UNWIND                    ;
```

```
                          50          01  D0 00078 5$:     MOVL    #1, R0                          ; 1705
                                          04 0007B          RET                                     ; 1707

; Routine Size:  124 bytes,    Routine Base:  $CODE$ + 0523
```

```
;  1186          1708  1 ROUTINE RCP_RESTART ( DEVNAM: REF BBLOCK) =
;  1187          1709  1
;  1188          1710  1 !++
;  1189          1711  1 ! RCP_RESTART
;  1190          1712  1 !
;  1191          1713  1 ! FUNCTIONAL DESCRIPTION:
;  1192          1714  1 !
;  1193          1715  1 !       Issue RESTART command to the RCP for the specified volume. This
;  1194          1716  1 !       causes the RCP to restart any recovery operations which were frozen
;  1195          1717  1 !       due to inaccessibility of objects.
;  1196          1718  1 !
;  1197          1719  1 ! CALLING SEQUENCE:
;  1198          1720  1 !
;  1199          1721  1 !       STATUS  = RCP_RESTART ( DEVNAM );
;  1200          1722  1 !
;  1201          1723  1 ! FORMAL PARAMETERS:
;  1202          1724  1 !
;  1203          1725  1 !       DEVNAM  = address of descriptor of device name for which frozen
;  1204          1726  1 !                 Recovery Unit operations are to be restarted.
;  1205          1727  1 !
;  1206          1728  1 ! COMPLETION CODES:
;  1207          1729  1 !
;  1208          1730  1 !       SS$_NORMAL      = successful completion
;  1209          1731  1 !       ...
;  1210          1732  1 !--
;  1211          1733  1
;  1212          1734  2 BEGIN
;  1213          1735  2
;  1214          1736  2 EXTERNAL ROUTINE
;  1215          1737  2         CJF$RECOVERW    : ADDRESSING_MODE (GENERAL);            ! interface to RCP
;  1216          1738  2 !       GET_VOLUME_NAME : ADDRESSING_MODE (GENERAL),            ! Return volume name for a given device
;  1217          1739  2 !       INTERCEPT_SIGNAL: ADDRESSING_MODE (GENERAL);            ! Condition handler
;  1218          1740  2
;  1219          1741  2 EXTERNAL LITERAL
;  1220          1742  2         EXE$C_SYSEFN;                                         ! system event flag
;  1221          1743  2
;  1222          1744  2 LOCAL
;  1223          1745  2         VOLUME_NAME     : REF BBLOCK;                         ! Volume name descriptor
;  1224          1746  2
;  1225          1747  2 OWN
;  1226          1748  2     !
;  1227          1749  2     ! Allocate space for RODB and RODBA structures.  OWN storage is used
;  1228          1750  2     ! since stack space is at a premium.
;  1229          1751  2     !
;  1230          1752  2         RODB            : BLOCK [RODB$K_LENGTH  + 4, BYTE],   ! RODB plus term. zero
;  1231          1753  2         RODBA           : BLOCK [RODBA$K_LENGTH + 4, BYTE],   ! RODBA + 0
;  1232          1754  2         IOSB            : VECTOR [2,LONG],                    ! I/O status block
;  1233          1755  2         STATUS          : BBLOCK [4];                         ! completion status
;  1234          1756  2
;  1235          1757  2
;  1236          1758  2 !
;  1237          1759  2 ! Enable a condition handler to field and print signalled messages.
;  1238          1760  2 !
;  1239          1761  2 ENABLE INTERCEPT_SIGNAL;
;  1240          1762  2
;  1241          1763  2 !
;  1242          1764  2 ! Fill the fixed portions of the RODB and RODBA.
```

```
;  1243          1765  2 !
;  1244          1766  2 RODB [RODB$B_TYPE] = RODB$K_RUJNL;                          ! object is RU journal
;  1245          1767  2 RODB [RODB$B_COUNT] = 1;                                   ! only one attribute
;  1246          1768  2 RODB [RODB$W_SIZE] = RODB$K_LENGTH + 4;                    ! actually, this isn't used
;  1247          1769  2 RODB [RODB$A_POINTER] = RODBA;                            ! point to attribute block
;  1248          1770  2 RODB + RODB$K_LENGTH = 0;                                 ! termination zero
;  1249          1771  2 RODBA [RODBA$B_TYPE] = RODBA$K_RUJDEVNAM;                  ! attribute is device name
;  1250          1772  2 RODBA [RODBA$W_SIZE] = .DEVNAM[DSC$W_LENGTH];             ! length of attribute
;  1251          1773  2 RODBA [RODBA$A_POINTER] = .DEVNAM[DSC$A_POINTER];         ! addr of attribute
;  1252          1774  2 RODBA + RODBA$K_LENGTH = 0;                              ! termination zero
;  1253          1775  2
;  1254          1776  2 !
;  1255          1777  2 ! Send it to the RCP.
;  1256          1778  2 !
;  1257          1779  2 STATUS = CJF$RECOVERW ( CJF$M_RESTART,                    ! function = RESTART
;  1258          1780  2                        RODB,                             ! OBJECT data block
;  1259          1781  2                        0,                                ! filter list (none)
;  1260          1782  2                        %REF(PSL$C_EXEC),                 ! access mode
;  1261          1783  2                        EXE$C_SYSEFN,                     ! event flag
;  1262          1784  2                        IOSB,                             ! I/O status block
;  1263          1785  2                        0,                                ! no AST address
;  1264          1786  2                        0 );                              ! no AST parameter
;  1265          1787  2 IF .STATUS THEN STATUS = .IOSB [ 0 ];
;  1266          1788  2
;  1267          1789  2 !
;  1268          1790  2 ! If the status code returned by $CREJNL indicates an "informational" message,
;  1269          1791  2 ! attempt to inform the user of the event.  All other status values are handled
;  1270          1792  2 ! outside of this routine.  The status code is signalled, and a condition
;  1271          1793  2 ! handler prints the message.
;  1272          1794  2 !
;  1273          1795  2 IF .STATUS[STS$V_SEVERITY] EQL STS$K_INFO
;  1274          1796  2 THEN
;  1275          1797  3     BEGIN
;  1276          1798  3     VOLUME_NAME = GET_VOLUME_NAME (.DEVNAM);
;  1277          1799  3     ERR_MESSAGE (MOUN$_VOLSTATUS, 1, .VOLUME_NAME, .STATUS);
;  1278          1800  2     END;
;  1279          1801  2
;  1280          1802  2 .STATUS                                                    ! Return status to caller
;  1281          1803  2
;  1282          1804  1 END;


                                                .PSECT  $OWN$,NOEXE,2

                                    00424 RODB:     .BLKB   12
                                    00430 RODBA:    .BLKB   12
                                    0043C IOSB:     .BLKB   8
                                    00444 STATUS:   .BLKB   4

                                                .EXTRN  CJF$RECOVERW, EXE$C_SYSEFN

                                                .PSECT  $CODE$,NOWRT,2

                                000C 00000 RCP_RESTART:
                                                .WORD   Save R2,R3                  ; 1708
                        53      0000'   CF  9E 00002    MOVAB   STATUS, R3          ;
```

```
                        5E              04  C2 00007           SUBL2   #4, SP                                  ;
                        6D         0078 CF  DE 0000A           MOVAL   3$, (FP)                                ; 1734
                    E0  A3 000C0104 8F  D0 0000F           MOVL    #786692, RODB                           ; 1766
                    E4  A3          EC  A3 9E 00017        MOVAB   RODBA, RODB+4                           ; 1769
                        E8          A3  D4 0001C           CLRL    RODB+8                                  ; 1770
                    EC  A3          06  90 0001F           MOVB    #6, RODBA                               ; 1771
                        52      04  AC  D0 00023           MOVL    DEVNAM, R2                              ; 1772
                    EE  A3          62  B0 00027           MOVW    (R2), RODBA+2                           ;
                    F0  A3      04  A2  D0 0002B           MOVL    4(R2), RODBA+4                          ; 1773
                        F4      A3  D4 00030               CLRL    RODBA+8                                 ; 1774
                                7E  7C 00033               CLRQ    -(SP)                                   ; 1779
                        F8      A3  9F 00035               PUSHAB  IOSB                                    ;
                 00000000G      8F  DD 00038               PUSHL   #EXE$C_SYSEFN                           ;
                    10  AE          01  D0 0003E           MOVL    #1, 16(SP)                              ; 1782
                        10      AE  9F 00042               PUSHAB  16(SP)                                  ;
                                7E  D4 00045               CLRL    -(SP)                                   ; 1779
                        E0      A3  9F 00047               PUSHAB  RODB                                    ;
                 02000000       8F  DD 0004A               PUSHL   #33554432                               ;
          00000000G  00             08  FB 00050           CALLS   #8, CJF$RECOVERW                        ;
                     63             50  D0 00057           MOVL    R0, STATUS                              ;
                     04             63  E9 0005A           BLBC    STATUS, 1$                              ; 1787
                     63         F8  A3  D0 0005D           MOVL    IOSB, STATUS                            ;
03          63           03             00  ED C0061 1$:   CMPZV   #0, #3, STATUS, #3                      ; 1795
                                       1A  12 00066        BNEQ    2$                                      ;
                                       52  DD 00068        PUSHL   R2                                      ; 1798
                FECA    CF             01  FB 0006A        CALLS   #1, GET_VOLUME_NAME                     ;
                                       63  DD 0006F        PUSHL   STATUS                                  ; 1799
                                       50  DD 00071        PUSHL   VOLUME_NAME                             ;
                                       01  DD 00073        PUSHL   #1                                      ;
                 0072A07B              8F  DD 00075        PUSHL   #7512187                                ;
          00000000G  00                04  FB 0007B        CALLS   #4, L!B$SIGNAL                          ;
                     50             63  D0 00082 2$:    MOVL    STATUS, R0                              ; 1804
                                    04  00085             RET                                             ;
                                  0000  00086 3$:    .WORD   Save nothing                            ; 1734
                                7E  D4  00088          CLRL    -(SP)                                   ;
                                5E  DD  0008A          PUSHL   SP                                      ;
                            7E      04  AC  7D 0008C       MOVQ    4(AP), -(SP)                            ;
                        FEEF    CF      03  FB 00090       CALLS   #3, INTERCEPT_SIGNAL                    ;
                                        04  00095       RET                                             ;

; Routine Size:  150 bytes,    Routine Base:  $CODE$ + 059F

; 1283          1805  1
; 1284          1806  1 END
; 1285          1807  0 ELUDOM


                                                                .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY                                                        ;

;       Name                    Bytes                           Attributes                             ;

;  $OWN$                         1096  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)  ;
```

```
;  $CODE$                       1589  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                                Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1            18619       105         0      1000          00:02.0
;
;
;
;
;
;                                       COMMAND QUALIFIERS

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

; Size:         1526 code + 1159 data bytes
; Run Time:       00:39.0
; Elapsed Time:   01:17.8
; Lines/CPU Min:   2782
; Lexemes/CPU-Min: 25956
; Memory Used:  212 pages
; Compilation Complete
```

0246 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

RDHOME
LIS

MWTUR2
LIS

RUJMAN
LIS

MWTUR1
LIS

SRCVOL
LIS

STACP
LIS

MRDBLK
LIS

REBUILD
LIS