```
   CCCCCCCCCCCC  LLL              IIIIIIIII    UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL              IIIIIIIII    UUU         UUU  TTTTTTTTTTTTTTT  LLL
   CCCCCCCCCCCC  LLL              IIIIIIIII    UUU         UUU  TTTTTTTTTTTTTTT  LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
CCC              LLL                 III       UUU         UUU        TTT        LLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL  IIIIIIIII    UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL  IIIIIIIII    UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
   CCCCCCCCCCCC  LLLLLLLLLLLLLLL  IIIIIIIII    UUUUUUUUUUUUUUU        TTT        LLLLLLLLLLLLLLL
```

```
  SSSSSSSS  HH      HH    OOOOOO    DDDDDDDD   EEEEEEEEEE  VV      VV  PPPPPPPP    RRRRRRRR    TTTTTTTTTT
  SSSSSSSS  HH      HH    OOOOOO    DDDDDDDD   EEEEEEEEEE  VV      VV  PPPPPPPP    RRRRRRRR    TTTTTTTTTT
SS          HH      HH  OO      OO  DD      DD EE          VV      VV  PP      PP  RR      RR      TT
SS          HH      HH  OO      OO  DD      DD EE          VV      VV  PP      PP  RR      RR      TT
SS          HH      HH  OO      OO  DD      DD EE          VV      VV  PP      PP  RR      RR      TT
SS          HH      HH  OO      OO  DD      DD EE          VV      VV  PP      PP  RR      RR      TT
  SSSSSS    HHHHHHHHHH  OO      OO  DD      DD EEEEEEEE    VV      VV  PPPPPPPP    RRRRRRRR        TT
  SSSSSS    HHHHHHHHHH  OO      OO  DD      DD EEEEEEEE    VV      VV  PPPPPPPP    RRRRRRRR        TT
        SS  HH      HH  OO      OO  DD      DD EE          VV      VV  PP          RR  RR          TT
        SS  HH      HH  OO      OO  DD      DD EE          VV      VV  PP          RR  RR          TT
        SS  HH      HH  OO      OO  DD      DD EE            VV  VV    PP          RR    RR        TT          ....
        SS  HH      HH  OO      OO  DD      DD EE            VV  VV    PP          RR    RR        TT          ....
SSSSSSSS    HH      HH    OOOOOO    DDDDDDDD   EEEEEEEEEE      VV      PP          RR      RR      TT          ....
SSSSSSSS    HH      HH    OOOOOO    DDDDDDDD   EEEEEEEEEE      VV      PP          RR      RR      TT          ....

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 shodevprt (IDENT = 'V04-000',
;     2          0002  0                    ADDRESSING_MODE (EXTERNAL = GENERAL)) =
;     3          0003  0
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
;    30          0030  1 !
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  SHOW utility
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !       This module contains the display routines for the SHOW commands which
;    37          0037  1 !       deal with devices, i.e. SHOW DEVICES, SHOW MAGTAPE,
;    38          0038  1 !       and SHOW PRINTER.
;    39          0039  1 !
;    40          0040  1 ! ENVIRONMENT:
;    41          0041  1 !       VAX native, user mode.
;    42          0042  1 !
;    43          0043  1 ! AUTHOR:  Gerry Smith           CREATION DATE:  28-Jul-1982
;    44          0044  1 !
;    45          0045  1 ! MODIFIED BY:
;    46          0046  1 !
;    47          0047  1 !       V03-023 CWH3023         CW Hobbs                    8-Aug-1984
;    48          0048  1 !               If extent caches are disabled, display zeroes in all three
;    49          0049  1 !               fields, rather than displaying max blocks as the number it
;    50          0050  1 !               would be if caches were enabled.
;    51          0051  1 !
;    52          0052  1 !       V03-022 CWH3022         CW Hobbs                    24-Jul-1984
;    53          0053  1 !               Add ACL support, a few more fields, expand bit explanations
;    54          0054  1 !               to be more sentence-like.  Change ADD_TO_LIST to use a
;    55          0055  1 !               routine which performs better breaking.
;    56          0056  1 !
;    57          0057  1 !       V03-021 CWH3021         CW Hobbs                    14-Apr-1984
```

```
;    58          0058  1 !                 Add a couple of display items for printers.
;    59          0059  1 !
;    60          0060  1 !       V03-020 CWH3020         CW Hobbs                14-Apr-1984
;    61          0061  1 !               Finish up the cluster display, and rework all full displays
;    62          0062  1 !               to improve their appearance and readability (thanks to
;    63          0063  1 !               Greg Robert for the suggestion for new style).
;    64          0064  1 !
;    65          0065  1 !       V03-019 CWH3019         CW Hobbs                13-Mar-1984
;    66          0066  1 !               When displaying status, look at the MNTVERIP before
;    67          0067  1 !               looking at the VALID bit.  The VALID bit can be in
;    68          0068  1 !               either state during mount verification.  Add feature
;    69          0069  1 !               to scan cluster lock database (INCOMPLETE!).
;    70          0070  1 !
;    71          0071  1 !       V03-018 CWH3018         CW Hobbs                3-Mar-1984
;    72          0072  1 !               Some changes to print allocation class device names.
;    73          0073  1 !               Also simplified brief display by removing two-thirds
;    74          0074  1 !               of the FAO control strings.
;    75          0075  1 !
;    76          0076  1 !       V03-017 CWH3017         CW Hobbs                28-FEB-1984
;    77          0077  1 !               Add more info to dual-path displays, and reduce the
;    78          0078  1 !               amount of noise in the brief display.  Add device
;    79          0079  1 !               types for uVAX disks and other new devices.
;    80          0080  1 !
;    81          0081  1 !       V03-016 EMD0049         Ellen M. Dusseault      27-FEB-1984
;    82          0082  1 !               Add ability to display fallback characteristic of
;    83          0083  1 !               a printer in routine, display_printer.
;    84          0084  1 !
;    85          0085  1 !       V03-015 JLV0330         Jake VanNoy             27-FEB-1984
;    86          0086  1 !               Add 'disconnected' to virtual terminal display.
;    87          0087  1 !
;    88          0088  1 !       V03-014 GJA0060         Greg Awdziewicz,        4-Jan-1984  8:24
;    89          0089  1 !               - Add journal specific flags to the full journal device
;    90          0090  1 !               display.
;    91          0091  1 !               - Add the DeaDmo (Deallocate upon dismount) bit to both the
;    92          0092  1 !               full and the brief journal device displays.
;    93          0093  1 !
;    94          0094  1 !       V03-013 LMP0165         L. Mark Pilant,         11-Oct-1983  9:00
;    95          0095  1 !               Make the brief dual port display more readable (and less
;    96          0096  1 !               confusing).
;    97          0097  1 !
;    98          0098  1 !       V03-012 LMP0162         L. Mark Pilant,         6-Oct-1983  15:18
;    99          0099  1 !               Add support for displaying dual ported device names.
;   100          0100  1 !
;   101          0101  1 !       V03-011 GAS0186         Gerry Smith             17-Sep-1983
;   102          0102  1 !               Fix /FULL display of journal name, by extending the
;   103          0103  1 !               length to ucb$s_jnl_nam.
;   104          0104  1 !
;   105          0105  1 !       V03-010 LMP0140         L. Mark Pilant,         24-Aug-1983  0:35
;   106          0106  1 !               Add support for alphanumeric UICs.
;   107          0107  1 !
;   108          0108  1 !       V03-009 GAS0167         Gerry Smith             22-Aug-1983
;   109          0109  1 !               Fix the processing of the journal devices for journals.
;   110          0110  1 !
;   111          0111  1 !       V03-008 GAS0149         Gerry Smith             28-Jun-1983
;   112          0112  1 !               Use IOC$CVT_DEVNAM for the device name.
;   113          0113  1 !
;   114          0114  1 !       V03-007 GAS0133         Gerry Smith             16-May-1983
```

```
;  115          0115  1 !               Add default extend quantity and file protection, as well
;  116          0116  1 !               as retention periods, for disks.
;  117          0117  1 !
;  118          0118  1 !       V03-006 GAS0120         Gerry Smith             14-Apr-1983
;  119          0119  1 !               Modify the displays slightly for a more tabular format.
;  120          0120  1 !
;  121          0121  1 !       V03-005 GAS0114         Gerry Smith              1-Apr-1983
;  122          0122  1 !               Fix display to handle long device names.
;  123          0123  1 !
;  124          0124  1 !       V03-004 CWH1002         CW Hobbs                1-Mar-1983
;  125          0125  1 !               Display the extended pid for the owner.
;  126          0126  1 !
;  127          0127  1 !       V03-003 GAS0110         Gerry Smith             28-Feb-1983
;  128          0128  1 !               Add support for cluster devices.
;  129          0129  1 !
;  130          0130  1 !       V03-002 GAS0107         Gerry Smith             11-Feb-1983
;  131          0131  1 !               Add support for journals.
;  132          0132  1 !
;  133          0133  1 !       V03-001 GAS0105         Gerry Smith             21-Jan-1983
;  134          0134  1 !               Fix the length of the device name field.  Remove the
;  135          0135  1 !               quotation remarks from volume labels.
;  136          0136  1 !
;  137          0137  1 !--
```

```
;  139          0138  1 !
;  140          0139  1 ! Include files
;  141          0140  1 !
;  142          0141  1 LIBRARY 'SYS$LIBRARY:LIB';                  ! VAX/VMS system definitions
;  143          0142  1 REQUIRE 'SRC$:SHOWDEF';                     ! SHOW common definitions
;  144          0241  1 REQUIRE 'SRC$:SHODEVDEF';                   ! SHOW DEVICES common definitions
;  145          0532  1
;  146          0533  1 !
;  147          0534  1 ! Define a shared message code for the read error
;  148          0535  1 !
;  149          0536  1 $SHR_MSGDEF     (SHOW, 120, LOCAL,   (READERR, SEVERE) );
;  150          0537  1
;  151          0538  1 !
;  152          0539  1 ! Macro to setup a table of keys and counted ascii strings
;  153          0540  1 !
;  154          0541  1 MACRO
;  155          0542  1     table_keys  [tag,string] = tag%,
;  156          0543  1
;  157          0544  1     table_entries  [tag,string] = cstring(%STRING(string))%,
;  158          0545  1
;  159        M 0546  1     make_table (name) =
;  160        M 0547  1         LITERAL %NAME(%STRING(name,'_number')) = (%LENGTH-1)/2;
;  161        M 0548  1         OWN
;  162        M 0549  1             %NAME(%STRING(name),'_type') : VECTOR[%NAME(name, '_number'),BYTE]
;  163        M 0550  1             INITIAL (BYTE(table_keys(%REMAINING))),
;  164        M 0551  1             %NAME(%STRING(name),'_label') : VECTOR[%NAME(name, '_number')]
;  165          0552  1             INITIAL (table_entries(%REMAINING));%;
;  166          0553  1
;  167          0554  1 MACRO
;  168          0555  1     unknown = cstring('unknown')%;
;  169          0556  1
;  170          0557  1 !
;  171          0558  1 ! A macro to add successive ASCII items to a string buffer, and update the
;  172          0559  1 ! descriptor for the string.  Saves lots of typing of %ASCIDs.
;  173          0560  1 !
;  174          0561  1 MACRO
;  175          0562  1     add_to_list (ptr, string, length) =         add_to_list_rtn ((ptr), %ASCID string, (length))  %;
```

```
;  177          0563  1 !
;  178          0564  1 ! Set up tables of specific device types, and the counted ascii string
;  179          0565  1 ! describing each type.
;  180          0566  1 !
;  181          0567  1
;  182        P 0568  1 make_table (printer,
;  183        P 0569  1                 dt$_lp11,          LP11,
;  184        P 0570  1                 dt$_la11,          LA11,
;  185          0571  1                 dt$_la180,         LA180);
;  186          0572  1
;  187        P 0573  1 make_table (tape,
;  188        P 0574  1                 dt$_te16,          TE16,          dt$_tu45,          TU45,
;  189        P 0575  1                 dt$_tu77,          TU77,          dt$_ts11,          TS11,
;  190        P 0576  1                 dt$_tu78,          TU78,          dt$_ta78,          TA78,
;  191        P 0577  1                 dt$_tu80,          TU80,          dt$_tu81,          TU81,
;  192          0578  1                 dt$_ta81,          TA81,          dt$_tk50,          TK50);
;  193          0579  1
;  194        P 0580  1 make_table (format,
;  195        P 0581  1                 mt$k_normal11,     'Normal-11',
;  196        P 0582  1                 mt$k_cordmp11,     'Coredump-11',
;  197          0583  1                 mt$k_normal15,     'Normal-15');
;  198          0584  1
;  199        P 0585  1 make_table (disk,
;  200        P 0586  1                 dt$_rk06,          RK06,          dt$_rk07,          RK07,
;  201        P 0587  1                 dt$_rp04,          RP04,          dt$_rp05,          RP05,
;  202        P 0588  1                 dt$_rp06,          RP06,          dt$_rm03,          RM03,
;  203        P 0589  1                 dt$_rp07,          RP07,          dt$_rp07ht,        RP07,
;  204        P 0590  1                 dt$_rl01,          RL01,          dt$_rl02,          RL02,
;  205        P 0591  1                 dt$_rx02,          RX02,          dt$_rx04,          RX04,
;  206        P 0592  1                 dt$_rm80,          RM80,          dt$_tu58,          TU58,
;  207        P 0593  1                 dt$_rm05,          RM05,          dt$_rx01,          RX01,
;  208        P 0594  1                 dt$_ml11,          ML11,          dt$_rb02,          RB02,
;  209        P 0595  1                 dt$_rb80,          RB80,          dt$_ra80,          RA80,
;  210        P 0596  1                 dt$_ra81,          RA81,          dt$_rc25,          'RC25 Removable',
;  211        P 0597  1                 dt$_ra60,          RA60,          dt$_rcf25,         'RC25 Fixed',
;  212        P 0598  1                 dt$_rd51,          RD51,          dt$_rc26,          'RC26 Removable',
;  213        P 0599  1                 dt$_rx50,          RX50,          dt$_rcf26,         'RC26 Fixed',
;  214        P 0600  1                 dt$_rd52,          RD52,          dt$_rd53,          RD53,
;  215        P 0601  1                 dt$_rd26,          RD26,          dt$_ra82,          RA82,
;  216          0602  1                 dt$_crx50,         'Console RX50', dt$_cdr50,        CDR50);
;  217          0603  1
;  218        P 0604  1 make_table (terminal,
;  219        P 0605  1                 tt$_vt05,          vt05,          tt$_ft1,           ft1,
;  220        P 0606  1                 tt$_ft2,           ft2,           tt$_ft3,           ft3,
;  221        P 0607  1                 tt$_ft4,           ft4,           tt$_ft5,           ft5,
;  222        P 0608  1                 tt$_ft6,           ft6,           tt$_ft7,           ft7,
;  223        P 0609  1                 tt$_ft8,           ft8,           tt$_lax,           'LAx',
;  224        P 0610  1                 tt$_la36,          la36,          tt$_la120,         la120,
;  225        P 0611  1                 tt$_vt5x,          'VT5x',        tt$_vt52,          vt52,
;  226        P 0612  1                 tt$_vt55,          vt55,          tt$_vt100,         vt100,
;  227        P 0613  1                 tt$_vk100,         vk100,         tt$_vt173,         vt173,
;  228        P 0614  1                 tt$_la34,          la34,          tt$_la38,          la38,
;  229        P 0615  1                 tt$_la12,          la12,          tt$_la24,          la24,
;  230        P 0616  1                 tt$_la100,         la100,         tt$_lqp02,         lqp02,
;  231        P 0617  1                 tt$_vt101,         vt101,         tt$_vt102,         vt102,
;  232        P 0618  1                 tt$_vt105,         vt105,         tt$_vt125,         vt125,
;  233        P 0619  1                 tt$_vt131,         vt131,         tt$_vt132,         vt132,
```

```
;  234        P 0620  1              tt$_vt200_series,   'VT200 Series', tt$_pro_series,         'PRO series',
;  235        P 0621  1              tt$_tq_bts,         bts,            tt$_tek401x,            'TEK401x',
;  236          0622  1              tt$_la84,           la84);
;  237          0623  1
;  238        P 0624  1 make_table (journal,
;  239        P 0625  1              dt$_rujnl,          'recovery-unit journal',
;  240        P 0626  1              dt$_bijnl,          'before-image journal',
;  241        P 0627  1              dt$_aijnl,          'after-image journal',
;  242        P 0628  1              dt$_atjnl,          'audit-trail journal',
;  243        P 0629  1              dt$_cljnl,          'control journal',
;  244          0630  1              dt$_unknjnl,        'unknown journal');
;  245          0631  1
;  246          0632  1 !
;  247          0633  1 ! Make a table of access modes
;  248          0634  1 !
;  249          0635  1 OWN
;  250          0636  1     change_acl_addr : INITIAL (0),          ! Address of change_acl routine for call by name
;  251          0637  1     format_acl_addr : INITIAL (0),          ! Address of format_acl routine for call by name
;  252          0638  1     leading_blanks : INITIAL (0),           ! Leading blanks for continuation lines
;  253          0639  1     mode_table : VECTOR[4]
;  254          0640  1         INITIAL(cstring('Kernel'),
;  255          0641  1                 cstring('Exec'),
;  256          0642  1                 cstring('Super'),
;  257          0643  1                 cstring('User'));
;  258          0644  1
;  259          0645  1 BIND
;  260          0646  1     ascid_null = %ASCID '',
;  261          0647  1     ascid_secureshr = %ASCID 'SECURESHR';
```

```
;  263          0648  1 !
;  264          0649  1 ! Table of contents
;  265          0650  1 !
;  266          0651  1 FORWARD ROUTINE
;  267          0652  1     add_to_list_rtn     : NOVALUE,
;  268          0653  1     check_mnt_all,
;  269          0654  1     cluster_nodes       : NOVALUE,
;  270          0655  1     show_device_acl     : NOVALUE,
;  271          0656  1     sys$change_acl,                         ! Jacket routines for call-by-name
;  272          0657  1     sys$format_acl,
;  273          0658  1     display_brief       : NOVALUE,
;  274          0659  1     display_disk        : NOVALUE,
;  275          0660  1     display_terminal    : NOVALUE,
;  276          0661  1     display_journal     : NOVALUE,
;  277          0662  1     display_printer     : NOVALUE,
;  278          0663  1     display_magtape     : NOVALUE,
;  279          0664  1     display_general     : NOVALUE,
;  280          0665  1     common_display      : NOVALUE;
;  281          0666  1
;  282          0667  1 EXTERNAL ROUTINE
;  283          0668  1     expand_prot         : NOVALUE,
;  284          0669  1     lib$find_image_symbol,
;  285          0670  1     scan_cluster_locks  : NOVALUE,
;  286          0671  1     show$write_line     : NOVALUE;
;  287          0672  1
;  288          0673  1
;  289          0674  1
;  290          0675  1
;  291          0676  1
;  292          0677  1
;  293          0678  1
;  294          0679  1
;  295          0680  1
;  296          0681  1
;  297          0682  1
;  298          0683  1
;  299          0684  1
;  300          0685  1 !---
;  301          0686  1 !
;  302          0687  1 !           ************************************************
;  303          0688  1 !           **                                            **
;  304          0689  1 !           **     A couple of words about formats:       **
;  305          0690  1 !           **     The current format for /FULL           **
;  306          0691  1 !           **     displays is that there are 4           **
;  307          0692  1 !           **     spaces, followed by 32 characters,     **
;  308          0693  1 !           **     followed by 4 blanks, followed by      **
;  309          0694  1 !           **     39 characters of text.  Please         **
;  310          0695  1 !           **     don't f#ck this up, unless you're      **
;  311          0696  1 !           **     willing to fix ALL the displays        **
;  312          0697  1 !           **                                            **
;  313          0698  1 !           ************************************************
;  314          0699  1 !
;  315          0700  1 !+++
```

```
;   317          0701  1 GLOBAL ROUTINE add_to_list_rtn (ptr, string : REF $BBLOCK, length) : NOVALUE =
;   318          0702  2 BEGIN
;   319          0703  2
;   320          0704  2 !---
;   321          0705  2 !
;   322          0706  2 ! Add a string to the string buffer.  If the string would exceed the length
;   323          0707  2 ! desired, then break the string and form a new string.  To find the place to
;   324          0708  2 ! break, scan from the back of the string looking for spaces.  If a space is
;   325          0709  2 ! found, attempt to split at the space.
;   326          0710  2 !
;   327          0711  2 ! Inputs
;   328          0712  2 !       ptr     - the address of a pointer to an array of string descriptors
;   329          0713  2 !       string  - pointer to a string descriptor for the input string
;   330          0714  2 !       length  - a maximum length for a fragment
;   331          0715  2 !
;   332          0716  2 ! Outputs
;   333          0717  2 !       ptr     - might be adjusted to point to a new descriptor
;   334          0718  2 !         ptr[0] - length of output fragment
;   335          0719  2 !         ptr[1] - pointer to start of current output fragment
;   336          0720  2 !
;   337          0721  2 !---
;   338          0722  2
;   339          0723  2 REGISTER
;   340          0724  2     dptr : REF VECTOR,
;   341          0725  2     slen,
;   342          0726  2     sptr : REF VECTOR [, BYTE];
;   343          0727  2
;   344          0728  2 !
;   345          0729  2 ! Load some registers with a few common values
;   346          0730  2 !
;   347          0731  2 slen = .string[dsc$w_length];                   ! Grab the length into a register
;   348          0732  2 sptr = .string[dsc$a_pointer];                  ! Grab the pointer into a register
;   349          0733  2 dptr = ..ptr;                                   ! Look at the first descriptor itself
;   350          0734  2 !
;   351          0735  2 ! If the new string fits, append the new string to the buffer and adjust the length
;   352          0736  2 !
;   353          0737  2 IF .dptr[0] + .slen LEQ .length
;   354          0738  2 THEN
;   355          0739  3     BEGIN
;   356          0740  3     CH$MOVE(.slen,                              ! Add the string to the buffer
;   357          0741  3             .sptr,
;   358          0742  3             .dptr[1] + .dptr[0]);
;   359          0743  3     dptr[0] = .dptr[0] + .slen;                 ! Adjust the length for the appended segment
;   360          0744  3     RETURN;                                     ! All done, go back
;   361          0745  3     END
;   362          0746  2 !
;   363          0747  2 ! The new string would make the output string too long, if we can stick part of the
;   364          0748  2 ! new string on then do so.  Move to a new descriptor and move the rest to the new
;   365          0749  2 ! output string.
;   366          0750  2 !
;   367          0751  2 ELSE
;   368          0752  3     BEGIN
;   369          0753  3     !
;   370          0754  3     ! Scan the segment, looking for a word break (a space).
;   371          0755  3     !
;   372          0756  3     DECR si FROM .slen-1 TO 1                   ! Scan backwards through the new segment
;   373          0757  4     DO
```

```
;  374     0758  4          BEGIN
;  375     0759  4          IF .sptr[.si] EQL %C' '                 ! If we find a space
;  376     0760  4            AND                                   !  and
;  377     0761  4              .dptr[0] + .si LEQ .length          !   the string before the space fits
;  378     0762  4          THEN
;  379     0763  5              BEGIN
;  380     0764  5              CH$MOVE(.si,                        ! Add the string before ' ' to the old buffer
;  381     0765  5                      .sptr,
;  382     0766  5                      .dptr[1] + .dptr[0]);
;  383     0767  5              dptr[0] = .dptr[0] + .si;           ! Adjust the length for the appended segment
;  384     0768  5              slen = .slen - .si;                 ! Remove the first piece from the segment,
;  385     0769  5              sptr = .sptr + .si;                 !  leaving a leading space
;  386     0770  5              EXITLOOP;                           ! And finally, leave the loop
;  387     0771  4              END;
;  388     0772  3          END;
;  389     0773  3      !
;  390     0774  3      ! Another place to break the line would be at a hyphenated word, look for one
;  391     0775  3      ! of those.
;  392     0776  3      !
;  393     0777  3      DECR si FROM .slen-1 TO 1                   ! Scan backwards through the new segment
;  394     0778  3      DO
;  395     0779  4          BEGIN
;  396     0780  4          IF .sptr[.si] EQL %C'-'                 ! If we find a hyphen
;  397     0781  4            AND                                   !  and
;  398     0782  4              .dptr[0] + .si LSS .length          !   the string including the hyphen fits
;  399     0783  4          THEN
;  400     0784  5              BEGIN
;  401     0785  5              CH$MOVE(.si + 1,                    ! Add the string with the '-' to the old buffer
;  402     0786  5                      .sptr,
;  403     0787  5                      .dptr[1] + .dptr[0]);
;  404     0788  5              dptr[0] = .dptr[0] + .si + 1;       ! Adjust the length for the appended segment
;  405     0789  5              slen = .slen - .si;                 ! Remove the first piece from the segment,
;  406     0790  5              sptr = .sptr + .si;                 !  leaving a leading hyphen (stripped shortly)
;  407     0791  5              EXITLOOP;                           ! And finally, leave the loop
;  408     0792  4              END;
;  409     0793  3          END;
;  410     0794  3      dptr[3] = .dptr[1] + .dptr[0];              ! Point the next descriptor at the buffer
;  411     0795  3      dptr = dptr[2];                             ! Move to the next descriptor
;  412     0796  3      .ptr = .dptr;                               ! Shuffle the passed pointer also
;  413     0797  3      slen = .slen - 1;                           ! Adjust the length and pointer,
;  414     0798  3      sptr = .sptr + 1;                           !  in order to strip the leading space (or hyphen)
;  415     0799  3      CH$FILL(%C' ', .leading_blanks, .dptr[1]);  ! Add any leading blanks which are desired
;  416     0800  3      CH$MOVE(.slen,                              ! Copy the string to the buffer,
;  417     0801  3              .sptr,                              !  but remove the leading space (or hyphen)
;  418     0802  3              .dptr[1] + .leading_blanks);        ! Place after the leading blanks, if any
;  419     0803  3      dptr[0] = .slen + .leading_blanks;          ! Save the new length
;  420     0804  3      RETURN.                                     ! All done, go back
;  421     0805  2      END;
;  422     0806  2
;  423     0807  1 END;


                                                                  .TITLE  SHODEVPRT
                                                                  .IDENT  \V04-000\

                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2
```

```
                                                                    31 31 50 4C 04 00000 P.AAA:  .ASCII  <4>\LP11\
                                                                    31 31 41 4C 04 00005 P.AAB:  .ASCII  <4>\LA11\
                                                                 30 38 31 41 4C 05 0000A P.AAC:  .ASCII  <5>\LA180\
                                                                    36 31 45 54 04 00010 P.AAD:  .ASCII  <4>\TE16\
                                                                    35 34 55 54 04 00015 P.AAE:  .ASCII  <4>\TU45\
                                                                    37 37 55 54 04 0001A P.AAF:  .ASCII  <4>\TU77\
                                                                    31 31 53 54 04 0001F P.AAG:  .ASCII  <4>\TS11\
                                                                    38 37 55 54 04 00024 P.AAH:  .ASCII  <4>\TU78\
                                                                    38 37 41 54 04 00029 P.AAI:  .ASCII  <4>\TA78\
                                                                    30 38 55 54 04 0002E P.AAJ:  .ASCII  <4>\TU80\
                                                                    31 38 55 54 04 00033 P.AAK:  .ASCII  <4>\TU81\
                                                                    31 38 41 54 04 00038 P.AAL:  .ASCII  <4>\TA81\
                                                                    30 35 4B 54 04 0003D P.AAM:  .ASCII  <4>\TK50\
                                  31 31 2D 6C 61 6D 72 6F 4E 09 00042 P.AAN:  .ASCII  <9>\Normal-11\
                              31 31 2D 70 6D 75 64 65 72 6F 43 0B 0004C P.AAO:  .ASCII  <11>\Coredump-11\
                                  35 31 2D 6C 61 6D 72 6F 4E 09 00058 P.AAP:  .ASCII  <9>\Normal-15\
                                                                    36 30 4B 52 04 00062 P.AAQ:  .ASCII  <4>\RK06\
                                                                    37 30 4B 52 04 00067 P.AAR:  .ASCII  <4>\RK07\
                                                                    34 30 50 52 04 0006C P.AAS:  .ASCII  <4>\RP04\
                                                                    35 30 50 52 04 00071 P.AAT:  .ASCII  <4>\RP05\
                                                                    36 30 50 52 04 00076 P.AAU:  .ASCII  <4>\RP06\
                                                                    33 30 4D 52 04 0007B P.AAV:  .ASCII  <4>\RM03\
                                                                    37 30 50 52 04 00080 P.AAW:  .ASCII  <4>\RP07\
                                                                    37 30 50 52 04 00085 P.AAX:  .ASCII  <4>\RP07\
                                                                    31 30 4C 52 04 0008A P.AAY:  .ASCII  <4>\RL01\
                                                                    32 30 4C 52 04 0008F P.AAZ:  .ASCII  <4>\RL02\
                                                                    32 30 58 52 04 00094 P.ABA:  .ASCII  <4>\RX02\
                                                                    34 30 58 52 04 00099 P.ABB:  .ASCII  <4>\RX04\
                                                                    30 38 4D 52 04 0009E P.ABC:  .ASCII  <4>\RM80\
                                                                    38 35 55 54 04 000A3 P.ABD:  .ASCII  <4>\TU58\
                                                                    35 30 4D 52 04 000A8 P.ABE:  .ASCII  <4>\RM05\
                                                                    31 30 58 52 04 000AD P.ABF:  .ASCII  <4>\RX01\
                                                                    31 31 4C 4D 04 000B2 P.ABG:  .ASCII  <4>\ML11\
                                                                    32 30 42 52 04 000B7 P.ABH:  .ASCII  <4>\RB02\
                                                                    30 38 42 52 04 000BC P.ABI:  .ASCII  <4>\RB80\
                                                                    30 38 41 52 04 000C1 P.ABJ:  .ASCII  <4>\RA80\
                                                                    31 38 41 52 04 000C6 P.ABK:  .ASCII  <4>\RA81\
65 6C 62 61 76 6F 6D 65 52 20                                       35 32 43 52 0E 000CB P.ABL:  .ASCII  <14>\RC25 Removable\
                                                                    30 36 41 52 04 000DA P.ABM:  .ASCII  <4>\RA60\
                        64 65 78 69 46 20                           35 32 43 52 0A 000DF P.ABN:  .ASCII  <10>\RC25 Fixed\
                                                                    31 35 44 52 04 000EA P.ABO:  .ASCII  <4>\RD51\
65 6C 62 61 76 6F 6D 65 52 20                                       36 32 43 52 0E 000EF P.ABP:  .ASCII  <14>\RC26 Removable\
                                                                    30 35 58 52 04 000FE P.ABQ:  .ASCII  <4>\RX50\
                        64 65 78 69 46 20                           36 32 43 52 0A 00103 P.ABR:  .ASCII  <10>\RC26 Fixed\
                                                                    32 35 44 52 04 0010E P.ABS:  .ASCII  <4>\RD52\
                                                                    33 35 44 52 04 00113 P.ABT:  .ASCII  <4>\RD53\
                                                                    36 32 44 52 04 00118 P.ABU:  .ASCII  <4>\RD26\
                                                                    32 38 41 52 04 0011D P.ABV:  .ASCII  <4>\RA82\
                  30 35 58 52 20 65 6C 6F                           73 6E 6F 43 0C 00122 P.ABW:  .ASCII  <12>\Console RX50\
                                       30                           35 52 44 43 05 0012F P.ABX:  .ASCII  <5>\CDR50\
                                                                    35 30 54 56 04 00135 P.ABY:  .ASCII  <4>\VT05\
                                                                    31 54 46 03 0013A P.ABZ:  .ASCII  <3>\FT1\
                                                                    32 54 46 03 0013E P.ACA:  .ASCII  <3>\FT2\
                                                                    33 54 46 03 00142 P.ACB:  .ASCII  <3>\FT3\
                                                                    34 54 46 03 00146 P.ACC:  .ASCII  <3>\FT4\
                                                                    35 54 46 03 0014A P.ACD:  .ASCII  <3>\FT5\
                                                                    36 54 46 03 0014E P.ACE:  .ASCII  <3>\FT6\
```

```
                                    37 54 46 03  00152 P.ACF:  .ASCII  <3>\FT7\                                  ;
                                    38 54 46 03  00156 P.ACG:  .ASCII  <3>\FT8\                                  ;
                                    78 41 4C 03  0015A P.ACH:  .ASCII  <3>\LAx\                                  ;
                                 36 33 41 4C 04  0015E P.ACI:  .ASCII  <4>\LA36\                                 ;
                              30 32 31 41 4C 05  00163 P.ACJ:  .ASCII  <5>\LA120\                                ;
                                 78 35 54 56 04  00169 P.ACK:  .ASCII  <4>\VT5x\                                 ;
                                 32 35 54 56 04  0016E P.ACL:  .ASCII  <4>\VT52\                                 ;
                                 35 35 54 56 04  00173 P.ACM:  .ASCII  <4>\VT55\                                 ;
                              30 30 31 54 56 05  00178 P.ACN:  .ASCII  <5>\VT100\                                ;
                              30 30 31 4B 56 05  0017E P.ACO:  .ASCII  <5>\VK100\                                ;
                              33 37 31 54 56 05  00184 P.ACP:  .ASCII  <5>\VT173\                                ;
                                 34 33 41 4C 04  0018A P.ACQ:  .ASCII  <4>\LA34\                                 ;
                                 38 33 41 4C 04  0018F P.ACR:  .ASCII  <4>\LA38\                                 ;
                                 32 31 41 4C 04  00194 P.ACS:  .ASCII  <4>\LA12\                                 ;
                                 34 32 41 4C 04  00199 P.ACT:  .ASCII  <4>\LA24\                                 ;
                              30 30 31 41 4C 05  0019E P.ACU:  .ASCII  <5>\LA100\                                ;
                              32 30 50 51 4C 05  001A4 P.ACV:  .ASCII  <5>\LQP02\                                ;
                              31 30 31 54 56 05  001AA P.ACW:  .ASCII  <5>\VT101\                                ;
                              32 30 31 54 56 05  001B0 P.ACX:  .ASCII  <5>\VT102\                                ;
                              35 30 31 54 56 05  001B6 P.ACY:  .ASCII  <5>\VT105\                                ;
                              35 32 31 54 56 05  001BC P.ACZ:  .ASCII  <5>\VT125\                                ;
                              31 33 31 54 56 05  001C2 P.ADA:  .ASCII  <5>\VT131\                                ;
                              32 33 31 54 56 05  001C8 P.ADB:  .ASCII  <5>\VT132\                                ;
      73 65 69 72 65 53 20 30 30 32 54 56 0C  001CE P.ADC:  .ASCII  <12>\VT200 Series\                          ;
               73 65 69 72 65 73 5F 4F 52 50 0A  001DB P.ADD:  .ASCII  <10>\PRO_series\                          ;
                                    53 54 42 03  001E6 P.ADE:  .ASCII  <3>\BTS\                                  ;
                        78 31 30 34 4B 45 54 07  001EA P.ADF:  .ASCII  <7>\TEK401x\                              ;
                                 34 38 41 4C 04  001F2 P.ADG:  .ASCII  <4>\LA84\                                 ;
20 74 69 6E 75 2D 79 72 65 76 6F 63 65 72 15  001F7 P.ADH:  .ASCII  <21>\recovery-unit journal\              ;
                           6C 61 6E 72 75 6F 6A  00206                                                       ;
6A 20 65 67 61 6D 69 2D 65 72 6F 66 65 62 14  0020D P.ADI:  .ASCII  <20>\before-image journal\               ;
                              6C 61 6E 72 75 6F  0021C                                                       ;
6F 6A 20 65 67 61 6D 69 2D 72 65 74 66 61 13  00222 P.ADJ:  .ASCII  <19>\after-image journal\                ;
                                 6C 61 6E 72 75  00231                                                       ;
6F 6A 20 6C 69 61 72 74 2D 74 69 64 75 61 13  00236 P.ADK:  .ASCII  <19>\audit-trail journal\                ;
                                 6C 61 6E 72 75  00245                                                       ;
61 6E 72 75 6F 6A 20 6C 6F 72 74 6E 6F 63 0F  0024A P.ADL:  .ASCII  <15>\control journal\                    ;
                                             6C  00259                                                       ;
61 6E 72 75 6F 6A 20 6E 77 6F 6E 6B 6E 75 0F  0025A P.ADM:  .ASCII  <15>\unknown journal\                    ;
                                             6C  00269                                                       ;
                        6C 65 6E 72 65 4B 06  0026A P.ADN:  .ASCII  <6>\Kernel\                               ;
                                 63 65 78 45 04  00271 P.ADO:  .ASCII  <4>\Exec\                                 ;
                              72 65 70 75 53 05  00276 P.ADP:  .ASCII  <5>\Super\                                ;
                                 72 65 73 55 04  0027C P.ADQ:  .ASCII  <4>\User\                                 ;
                                                 00281         .BLKB   3
                                                 00284 P.ADS:  .BLKB   0                                         ;
                                       010E0000  00284 P.ADR:  .LONG   17694720                                  ;
                                      00000000'  00288         .ADDRESS P.ADS                                    ;
         00 00 00 52 48 53 45 52 55 43 45 53  0028C P.ADU:  .ASCII  \SECURESHR\<0><0><0>                      ;
                                       010E0009  00298 P.ADT:  .LONG   17694729                                  ;
                                      00000000'  0029C         .ADDRESS P.ADU                                    ;

                                                               .PSECT  $OWN$,NOEXE,2

                                       03 02 01  00000 PRINTER_TYPE:
                                                               .BYTE   1, 2, 3                                   ;
                                                 00003         .BLKB   1
```

```
                                00000000' 00000000' 00000000' 00004 PRINTER_LABEL:
                                                                     .ADDRESS P.AAA, P.AAB, P.AAC                          ;
                        0A  09  08  07  06  05  04  03  02  01  00010 TAPE_TYPE:
                                                                     .BYTE   1, 2, 3, 4, 5, 6, 7, 8, 9, 10                 ;
                                                                0001A .BLKB   2
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0001C TAPE_LABEL:
                                                                     .ADDRESS P.AAD, P.AAE, P.AAF, P.AAG, P.AAH, -         ;
                        00000000' 00000000' 00000000' 00000000' 00034        P.AAI, P.AAJ, P.AAK, P.AAL, P.AAM             ;
                                                  0E  0D  0C  00044 FORMAT_TYPE:
                                                                     .BYTE   12, 13, 14                                    ;
                                                                00047 .BLKB   1
                                00000000' 00000000' 00000000' 00048 FORMAT_LABEL:
                                                                     .ADDRESS P.AAN, P.AAO, P.AAP                          ;
0F  0E  0D  0C  0B  0A  09  08  07  06  05  04  03  02  01  00054 DISK_TYPE:
                                                                     .BYTE   1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, -      ;
1C  1B  20  1A  1F  19  18  16  17  15  14  13  12  11  10  00063        13, 14, 15, 16, 17, 18, 19, 20, 21, 23, -      ;
                                        22  21  1E  1D  00072        22, 24, 25, 31, 26, 32, 27, 28, 29, 30, -      ;
                                                                     33, 34                                                ;
                                                                00076 .BLKB   2
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00078 DISK_LABEL:
                                                                     .ADDRESS P.AAQ, P.AAR, P.AAS, P.AAT, P.AAU, -         ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00090        P.AAV, P.AAW, P.AAX, P.AAY, P.AAZ, P.ABA, -    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 000A8        P.ABB, P.ABC, P.ABD, P.ABE, P.ABF, P.ABG, -    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 000C0        P.ABH, P.ABI, P.ABJ, P.ABK, P.ABL, P.ABM, -    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 000D8        P.ABN, P.ABO, P.ABP, P.ABQ, P.ABR, P.ABS, -    ;
                    00000000' 00000000' 00000000' 00000000' 000F0        P.ABT, P.ABU, P.ABV, P.ABW, P.ABX              ;
41  40  40  21  20  20  17  16  15  14  13  12  11  10  01  00100 TERMINAL_TYPE:
                                                                     .BYTE   1, 16, 17, 18, 19, 20, 21, 22, 23, 32, -      ;
66  65  64  63  62  61  26  25  25  24  23  22  03  02  60  0010F        32, 33, 64, 64, 65, 96, 2, 3, 34, 35, 36, -    ;
                                        27  0A  04  6F  6E  0011E        37, 37, 38, 97, 98, 99, 100, 101, 102, -      ;
                                                                     110, 111, 4, 10, 39                                   ;
                                                                00123 .BLKB   1
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00124 TERMINAL_LABEL:
                                                                     .ADDRESS P.ABY, P.ABZ, P.ACA, P.ACB, P.ACC, -         ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0013C        P.ACD, P.ACE, P.ACF, P.ACG, P.ACH, P.ACI, -    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00154        P.ACJ, P.ACK, P.ACL, P.ACM, P.ACN, P.ACO, -    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 0016C        P.ACP, P.ACQ, P.ACR, P.ACS, P.ACT, P.ACU, -    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00184        P.ACV, P.ACW, P.ACX, P.ACY, P.ACZ, P.ADA, -    ;
          00000000' 00000000' 00000000' 00000000' 00000000' 0019C        P.ADB, P.ADC, P.ADD, P.ADE, P.ADF, P.ADG       ;
                                    00  05  04  03  02  01  001B0 JOURNAL_TYPE:
                                                                     .BYTE   1, 2, 3, 4, 5, 0                              ;
                                                                001B6 .BLKB   2
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 001B8 JOURNAL_LABEL:
                                                                     .ADDRESS P.ADH, P.ADI, P.ADJ, P.ADK, P.ADL, P.ADM     ;
                                        00000000  001D0 CHANGE_ACL_ADDR:
                                                                     .LONG   0                                             ;
                                        00000000  001D4 FORMAT_ACL_ADDR:
                                                                     .LONG   0                                             ;
                                        00000000  001D8 LEADING_BLANKS:
                                                                     .LONG   0                                             ;
                    00000000' 00000000' 00000000' 00000000' 001DC MODE_TABLE:
                                                                     .ADDRESS P.ADN, P.ADO, P.ADP, P.ADQ                   ;

                                                           ASCID_NULL=          P.ADR
                                                           ASCID_SECURESHR=     P.ADT
                                                                     .EXTRN  EXPAND_PROT, LIB$FIND_IMAGE_SYMBOL
                                                                     .EXTRN  SCAN_CLUSTER_LOCKS
```

```
                                                .EXTRN  SHOW$WRITE_LINE

                                                .PSECT  $CODE$,NOWRT,2

                                03FC 00000      .ENTRY  ADD_TO_LIST_RTN, Save R2,R3,R4,R5,R6,R7,R8,-; 0701
                                                        R9                                          ;
                        50      08   AC  D0 00002      MOVL    STRING, R0                          ; 0731
                        57           60  3C 00006      MOVZWL  (R0), SLEN                          ;
                        58      04   A0  D0 00009      MOVL    4(R0), SPTR                         ; 0732
                        56      04   BC  D0 0000D      MOVL    @PTR, DPTR                          ; 0733
                50              66   57  C1 00011      ADDL3   SLEN, (DPTR), R0                    ; 0737
                    0C          AC   50  D1 00015      CMPL    R0, LENGTH                          ;
                                     0D  14 00019      BGTR    1$                                  ;
                50      04      A6   66  C1 0001B      ADDL3   (DPTR), 4(DPTR), R0                 ; 0742
                60              68   57  28 00020      MOVC3   SLEN, (SPTR), (R0)                  ;
                                66   57  C0 00024      ADDL2   SLEN, (DPTR)                        ; 0743
                                     04  00027      RET                                         ; 0752
                                59   57  D0 00028 1$:  MOVL    SLEN, SI                            ; 0756
                                     24  11 0002B      BRB     3$                                  ;
                                20       6948 91 0002D 2$:  CMPB    (SI)[SPTR], #32             ; 0759
                                         1E  12 00031      BNEQ    3$                          ;
                50              66       59 C1 00033      ADDL3   SI, (DPTR), R0                  ; 0761
                    0C          AC       50 D1 00037      CMPL    R0, LENGTH                      ;
                                         14 14 0003B      BGTR    3$                              ;
                50      04      A6       66 C1 0003D      ADDL3   (DPTR), 4(DPTR), R0             ; 0766
                60              68       59 28 00042      MOVC3   SI, (SPTR), (R0)                ;
                                66       59 C0 00046      ADDL2   SI, (DPTR)                      ; 0767
                                57       59 C2 00049      SUBL2   SI, SLEN                        ; 0768
                                58       59 C0 0004C      ADDL2   SI, SPTR                        ; 0769
                                         03 11 0004F      BRB     4$                              ; 0763
                                D9       59 F5 00051 3$:  SOBGTR  SI, 2$                          ; 0756
                                59       57 D0 00054 4$:  MOVL    SLEN, SI                        ; 0777
                                         2D 11 00057      BRB     6$                              ;
                                2D           6948 91 00059 5$:  CMPB    (SI)[SPTR], #45         ; 0780
                                             27 12 0005D      BNEQ    6$                          ;
                    50                   66   59 C1 0005F      ADDL3   SI, (DPTR), R0              ; 0782
                        0C               AC   50 D1 00063      CMPL    R0, LENGTH                  ;
                                              1D 18 00067      BGEQ    6$                          ;
                                51       0'  A9 9E 00069      MOVAB   1(R9), R1                   ; 0785
                    50          04       A6  66 C1 0006D      ADDL3   (DPTR), 4(DPTR), R0         ; 0787
                    60                   68  51 28 00072      MOVC3   R1, (SPTR), (R0)            ;
                    50                   66  59 C1 00076      ADDL3   SI, (DPTR), R0              ; 0788
                                         66  01  A0 9E 0007A      MOVAB   1(R0), (DPTR)           ;
                                         57  59 C2 0007E      SUBL2   SI, SLEN                    ; 0789
                                         58  59 C0 00081      ADDL2   SI, SPTR                    ; 0790
                                             03 11 00084      BRB     7$                          ; 0784
                                         D0  59 F5 00086 6$:  SOBGTR  SI, 5$                      ; 0777
                        04      A6       86  86 C1 00089 7$:  ADDL3   (DPTR)+, (DPTR)+, 4(DPTR)  ; 0794
                                04       BC  56 D0 0008E      MOVL    DPTR, @PTR                  ; 0796
                                             57 D7 00092      DECL    SLEN                        ; 0797
                                             58 D6 00094      INCL    SPTR                        ; 0798
                                         59  0000' CF D0 00096      MOVL    LEADING_BLANKS, R9      ; 0799
            59                   20      6E  00 2C 0009B      MOVC5   #0, (SP), #32, R9, @4(DPTR) ;
                                             04 B6 000A0      ;
                        04 B649              68  57 28 000A2      MOVC3   SLEN, (SPTR), @4(DPTR)[R9]  ; 0802
                               66            57  59 C1 000A8      ADDL3   R9, SLEN, (DPTR)            ; 0803
                                                04 000AC      RET                                 ; 0807
```

```
; Routine Size:  173 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  425     0808  1 GLOBAL ROUTINE check_mnt_all (scratch, flags) =
;  426     0809  2 BEGIN
;  427     0810  2
;  428     0811  2 !---
;  429     0812  2 !
;  430     0813  2 ! This checks to see that we pass any SHOW DEVICE /ALLOCATED or /MOUNTED
;  431     0814  2 ! restrictions.  Note that we must wait until after we check clusterness
;  432     0815  2 ! before we can decide whether a device is really mounted or not.
;  433     0816  2 !
;  434     0817  2 ! Inputs
;  435     0818  2 !       SCRATCH - contains data about the device
;  436     0819  2 !       FLAGS   - contains info on what to print
;  437     0820  2 !
;  438     0821  2 ! Outputs
;  439     0822  2 !       FLAGS[DEVI$V_DISPLAYED] set if we passed
;  440     0823  2 !       Status - true if we pass, false if we didn't
;  441     0824  2 !
;  442     0825  2 !---
;  443     0826  2
;  444     0827  2 MAP
;  445     0828  2     scratch : REF $BBLOCK,
;  446     0829  2     flags : REF $BBLOCK;
;  447     0830  2
;  448     0831  2 !
;  449     0832  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;  450     0833  2 ! then check to see if the device is mounted/allocated.  If not, then
;  451     0834  2 ! don't display information on the device.
;  452     0835  2 !
;  453     0836  2 IF .flags[devi$v_mounted]
;  454     0837  2 THEN IF NOT .$BBLOCK[scratch[d_l_devchar], dev$v_mnt]
;  455     0838  2 THEN RETURN false;
;  456     0839  2
;  457     0840  2 IF .flags[devi$v_allocated]
;  458     0841  2 THEN IF NOT .$BBLOCK[scratch[d_l_devchar], dev$v_all]
;  459     0842  2 THEN RETURN false;
;  460     0843  2
;  461     0844  2 !
;  462     0845  2 ! Set the bit and return true
;  463     0846  2 !
;  464     0847  2 flags[devi$v_displayed] = 1;
;  465     0848  2
;  466     0849  2 RETURN true;
;  467     0850  1 END;


                                                0000 00000          .ENTRY  CHECK_MNT_ALL, Save nothing                     ; 0808
                                51      08  AC  D0 00002          MOVL    FLAGS, R1                                       ; 0836
                                61          02  E1 00006          BBC     #2, (R1), 1$                                    ;
                                50      04  AC  D0 0000A          MOVL    SCRATCH, R0                                     ; 0837
                            72  A0          03  E1 0000E          BBC     #3, 114(R0), 3$                                 ;
                                09          61  E9 00013 1$:      BLBC    (R1), 2$                                        ; 0840
                                50      04  AC  D0 00016          MOVL    SCRATCH, R0                                     ; 0841
                                        72  A0  95 0001A          TSTB    114(R0)                                         ;
                                            08  18 0001D          BGEQ    3$                                              ;
```

```
                    01    A1              20  88 0001F 2$:     BISB2   #32, 1(R1)                          ; 0847
                          50              01  D0 00023         MOVL    #1, R0                              ; 0849
                                          04 00026             RET                                         ;
                                          50  D4 00027 3$:     CLRL    R0                                  ; 0850
                                          04 00029             RET                                         ;

; Routine Size:  42 bytes,    Routine Base:  $CODE$ + 00AD
```

```
;  469          0851  1 GLOBAL ROUTINE cluster_nodes (scratch, clubuf) : NOVALUE =
;  470          0852  2 BEGIN
;  471          0853  2
;  472          0854  2 !---
;  473          0855  2 !
;  474          0856  2 ! This routine displays a line telling which remote nodes have
;  475          0857  2 ! talons embedded in the device.
;  476          0858  2 !
;  477          0859  2 ! Inputs
;  478          0860  2 !       SCRATCH - contains data about the device
;  479          0861  2 !       CLUBUF  - longword vector, first longword is CSID count,
;  480          0862  2 !                 followed by that many CSIDs
;  481          0863  2 !
;  482          0864  2 ! Outputs
;  483          0865  2 !       None
;  484          0866  2 !
;  485          0867  2 !---
;  486          0868  2
;  487          0869  2 MAP
;  488          0870  2     scratch : REF $BBLOCK,
;  489          0871  2     clubuf : REF VECTOR [, LONG];
;  490          0872  2
;  491          0873  2 LOCAL
;  492          0874  2     arglist : VECTOR[4, LONG],
;  493          0875  2     desc : VECTOR[2, LONG],
;  494          0876  2     head : REF $BBLOCK,
;  495          0877  2     buffer : VECTOR[84, BYTE],
;  496          0878  2     nodename : VECTOR[16, BYTE];
;  497          0879  2
;  498          0880  2 !
;  499          0881  2 ! Return if nothing to note
;  500          0882  2 !
;  501          0883  3 IF 0 EQL (.scratch[d_w_bits] AND (1^$BITPOSITION(d_v_remote_mounts) OR 1^$BITPOSITION(d_v_remote_all)))
;  502          0884  2 THEN RETURN;
;  503          0885  2
;  504          0886  2 !
;  505          0887  2 ! If the extra files-11 line hasn't been printed, then add a null line
;  506          0888  2 !
;  507          0889  2 IF .scratch[d_b_aqbtype] NEQ aqb$k_f11v2
;  508          0890  2 THEN
;  509          0891  2     show$write_line (ascid_null, arglist);
;  510          0892  2
;  511          0893  2 !
;  512          0894  2 ! Determine the appropriate label for the line and initialize the output buffer and descriptor
;  513          0895  2 !
;  514          0896  2 IF .scratch[d_v_local_mount]
;  515          0897  2 THEN
;  516          0898  2     head = %ASCID '  Volume is also mounted on '
;  517          0899  2 ELSE IF .scratch[d_v_remote_mounts]
;  518          0900  2 THEN
;  519          0901  2     head = %ASCID '  Volume is mounted on '
;  520          0902  2 ELSE IF .scratch[d_v_remote_all]
;  521          0903  2 THEN
;  522          0904  2     head = %ASCID '  Device is allocated on node ';
;  523          0905  2 desc[0] = .head[dsc$w_length];
;  524          0906  2 desc[1] = buffer;
;  525          0907  2 CH$MOVE(.desc[0], .head[dsc$a_pointer], buffer);
```

```
;  526          0908  2
;  527          0909  2 !
;  528          0910  2 ! Add the system names to the buffer, wrapping if we hit 80 characters
;  529          0911  2 !
;  530          0912  2 INCR idx FROM 1 TO .clubuf[0]
;  531          0913  2 DO
;  532          0914  3     BEGIN
;  533          0915  3     arglist[0] = (syi$_nodename^16 OR 16);
;  534          0916  3     arglist[1] = nodename;
;  535          0917  3     arglist[2] = arglist[3];            ! Returned length word goes here
;  536          0918  3     arglist[3] = 0;                     ! Note: this clears the high word of the length
;  537          0919  4     IF $getsyi (csidadr=clubuf[.idx], itmlst=arglist)
;  538          0920  3     THEN
;  539          0921  4         BEGIN
;  540          0922  4         IF (.desc[0] + .arglist[3] + 1) GTR 80  ! Will this one make us wrap?
;  541          0923  4         THEN
;  542          0924  5             BEGIN
;  543          0925  5             desc[0] = .desc[0] - 1;             ! Remove the extra " "
;  544          0926  5             show$write_line(desc,arglist);
;  545          0927  5             CH$FILL(' ', 10, buffer);
;  546          0928  5             desc[0] = 10;
;  547          0929  4             END;
;  548          0930  4         CH$MOVE (.arglist[3], nodename, buffer[.desc[0]]);
;  549          0931  4         desc[0] = .desc[0] + .arglist[3];
;  550          0932  4         (buffer[.desc[0]])<0,16,0> = ', ';
;  551          0933  4         desc[0] = .desc[0] + 2;
;  552          0934  3         END;
;  553          0935  2     END;
;  554          0936  2
;  555          0937  2 !
;  556          0938  2 ! Write the remainder of the buffer if we have modified it (it is possible that
;  557          0939  2 ! a node will have crashed since we found the CSID, and we might not have anything
;  558          0940  2 ! to say).
;  559          0941  2 !
;  560          0942  2 IF .desc[0] NEQ .head[dsc$w_length]
;  561          0943  2 THEN
;  562          0944  3     BEGIN
;  563          0945  3     desc[0] = .desc[0] - 1;                 ! Remove the extra " "
;  564          0946  3     buffer[.desc[0]-1] = '.';               ! Change the final "," to a "."
;  565          0947  3     show$write_line(desc,desc);
;  566          0948  2     END;
;  567          0949  2
;  568          0950  2 RETURN;
;  569          0951  1 END;


                                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

73 6C 61 20 73 69 20 65 6D 75 6C 6F 56 20 20  002A0 P.ADW:  .ASCII  \  Volume is also mounted on \                ;
      20 6E 6F 20 64 65 74 6E 75 6F 6D 20 6F  002AF                                                              ;
                                              002BC P.ADV:  .LONG   17694748                                     ;
                                              002C0         .ADDRESS P.ADW                                       ;
75 6F 6D 20 73 69 20 65 6D 75 6C 6F 56 20 20  002C4 P.ADY:  .ASCII  \  Volume is mounted on \<0>                 ;
                  00 20 6E 6F 20 64 65 74 6E  002D3                                                              ;
                                              002DC P.ADX:  .LONG   17694743                                     ;
                                              002E0         .ADDRESS P.ADY                                       ;
```

```
6C 6C 61 20 73 69 20 65 63 69 76 65 44 20 20 002E4 P.AEA:  .ASCII  \ Device is allocated on node \<0><0>           ;
20 65 64 6F 6E 20 6E 6F 20 64 65 74 61 63 6F 002F3                                                               ;
                                       00 00 00302                                                               ;
                                   010E001E  00304 P.ADZ:  .LONG   17694750                                      ;
                                   00000000' 00308         .ADDRESS P.AEA                                        ;

                                                           .EXTRN  SYS$GETSYI

                                                           .PSECT  $CODE$,NOWRT,2

                                       01FC 00000          .ENTRY  CLUSTER_NODES, Save R2,R3,R4,R5,R6,R7,R8      ; 0851
                  58 00000000G   00    9E 00002            MOVAB   SHOW$WRITE_LINE, R8                           ;
                  5E          84 AE    9E 00009            MOVAB   -124(SP), SP                                  ;
                  52          04 AC    D0 0000D            MOVL    SCRATCH, R2                                   ; 0883
          0180    8F          04 A2    B3 00011            BITW    4(R2), #384                                   ;
                                 01    12 00017            BNEQ    1$                                            ;
                                       04 00019            RET                                                   ;
                  02        009D C2    91 0001A 1$:        CMPB    157(R2), #2                                   ; 0889
                                 0A    13 0001F            BEQL    2$                                            ;
                              6C AE    9F 00021            PUSHAB  ARGLIST                                       ; 0891
                           0000' CF    9F 00024            PUSHAB  ASCID_NULL                                    ;
                  68             02    FB 00028            CALLS   #2, SHOW$WRITE_LINE                           ;
      07       04 A2             06    E1 0002B 2$:        BBC     #6, 4(R2), 3$                                 ; 0896
                  57       0000' CF    9E 00030            MOVAB   P.ADV, HEAD                                   ; 0898
                                 15    11 00035            BRB     5$                                            ;
                              04 A2    95 00037 3$:        TSTB    4(R2)                                         ; 0899
                                 07    18 0003A            BGEQ    4$                                            ;
                  57       0000' CF    9E 0003C            MOVAB   P.ADX, HEAD                                   ; 0901
                                 09    11 00041            BRB     5$                                            ;
                  05          05 A2    E9 00043 4$:        BLBC    5(R2), 5$                                     ; 0902
                  57       0000' CF    9E 00047            MOVAB   P.ADZ, HEAD                                   ; 0904
               64 AE             67    3C 0004C 5$:        MOVZWL  (HEAD), DESC                                  ; 0905
               68 AE          10 AE    9E 00050            MOVAB   BUFFER, DESC+4                                ; 0906
   10 AE       04 B7          64 AE    28 00055            MOVC3   DESC, @4(HEAD), BUFFER                        ; 0907
                                 56    D4 0005C            CLRL    IDX                                           ; 0912
                                 76    11 0005E            BRB     8$                                            ;
               6C AE 10D90010    8F    D0 00060 6$:        MOVL    #282656784, ARGLIST                           ; 0915
               70 AE             6E    9E 00068            MOVAB   NODENAME, ARGLIST+4                           ; 0916
               74 AE          78 AE    9E 0006C            MOVAB   ARGLIST+12, ARGLIST+8                         ; 0917
                              78 AE    D4 00071            CLRL    ARGLIST+12                                    ; 0918
                                 7E    7C 00074            CLRQ    -(SP)                                         ; 0919
                                 7E    D4 00076            CLRL    -(SP)                                         ;
                              78 AE    9F 00078            PUSHAB  ARGLIST                                       ;
                                 7E    D4 0007B            CLRL    -(SP)                                         ;
                           08 BC46     DF 0007D            PUSHAL  @CLUBUF[IDX]                                  ;
                                 7E    D4 00081            CLRL    -(SP)                                         ;
           00000000G   00        07    FB 00083            CALLS   #7, SYS$GETSYI                                ;
                       49        50    E9 0008A            BLBC    R0, 8$                                        ;
      50       64 AE          78 AE    C1 0008D            ADDL3   ARGLIST+12, DESC, R0                          ; 0922
                                 50    D6 00093            INCL    R0                                            ;
           00000050    8F        50    D1 00095            CMPL    R0, #80                                       ;
                                 17    15 0009C            BLEQ    7$                                            ;
                              64 AE    D7 0009E            DECL    DESC                                          ; 0925
                              6C AE    9F 000A1            PUSHAB  ARGLIST                                       ; 0926
                              68 AE    9F 000A4            PUSHAB  DESC                                          ;
                  68             02    FB 000A7            CALLS   #2, SHOW$WRITE_LINE                           ;
   0A      20     6E             00    2C 000AA            MOVC5   #0, (SP), #32, #10, BUFFER                    ; 0927
```

```
                              10   AE       000AF                                                   ;
                    64   AE        0A  D0   000B1        MOVL     #10, DESC                         ; 0928
                         50   10   AE  9E   000B5  7$:   MOVAB    BUFFER, R0                        ; 0930
           64 BE40       6E   78   AE  28   000B9        MOVC3    ARGLIST+12, NODENAME, @DESC[R0]   ;
                    64   AE   78   AE  C0   000C0        ADDL2    ARGLIST+12, DESC                  ; 0931
                         50   10   AE  9E   000C5        MOVAB    BUFFER, R0                        ; 0932
                         50   64   AE  C0   000C9        ADDL2    DESC, R0                          ;
                         60 202C   8F  B0   000CD        MOVW     #8236, (R0)                       ;
                    64   AE        02  C0   000D2        ADDL2    #2, DESC                          ; 0933
                85       56   08   BC  F3   000D6  8$:   AOBLEQ   @CLUBUF, IDX, 6$                  ; 0912
   64   AE      67       10        00  ED   000DB        CMPZV    #0, #16, (HEAD), DESC             ; 0942
                                   18  13   000E1        BEQL     9$                                ;
                              64   AE  D7   000E3        DECL     DESC                              ; 0945
                         50   10   AE  9E   000E6        MOVAB    BUFFER, R0                        ; 0946
                         50   64   AE  C0   000EA        ADDL2    DESC, R0                          ;
                    FF   A0        2E  90   000EE        MOVB     #46, -1(R0)                       ;
                              64   AE  9F   000F2        PUSHAB   DESC                              ; 0947
                              68   AE  9F   000F5        PUSHAB   DESC                              ;
                         68        02  FB   000F8        CALLS    #2, SHOW$WRITE_LINE               ;
                                       04   000FB  9$:   RET                                        ; 0951

; Routine Size:  252 bytes,    Routine Base:  $CODE$ + 00D7
```

```
;  571          0952  1 GLOBAL ROUTINE show_device_acl (scratch : REF $BBLOCK) : NOVALUE =
;  572          0953  2 BEGIN
;  573          0954  2
;  574          0955  2 !---
;  575          0956  2 !
;  576          0957  2 ! This routine displays the access control list if it exists.  Thanks to
;  577          0958  2 ! Mark Pilant for [CLIUTL.SRC]SHOWACL.B32, from which it was lifted.
;  578          0959  2 !
;  579          0960  2 ! Inputs
;  580          0961  2 !       SCRATCH - contains data about the device
;  581          0962  2 !
;  582          0963  2 !--
;  583          0964  2
;  584          0965  2 LOCAL
;  585          0966  2     acl_lockid          : $BBLOCK [acl$s_rlock_acl],        ! Lock-id for ACL lock
;  586          0967  2     object_chan,                                            ! Channel for object
;  587          0968  2     object_name         : VECTOR [2, LONG],                 ! Descriptor for device name
;  588          0969  2     object_type,                                            ! Object's type code
;  589          0970  2     display_width,                                          ! Output device width
;  590          0971  2     ace_bin_desc        : VECTOR [2, LONG],                 ! Binary ACE descr
;  591          0972  2     ace_storage         : $BBLOCK [acl$s_readace] VOLATILE, ! Binary ACE storage
;  592          0973  2     ace_txt_desc        : VECTOR [2, LONG],                 ! Text ACE descr
;  593          0974  2     ace_text            : VECTOR [512, BYTE],               ! Text ACE storage
;  594          0975  2     atr_arglist         : BLOCKVECTOR [2, itm$s_item, BYTE], ! $CHANGE_ACL item list
;  595          0976  2     acl_context,                                            ! $CHANGE_ACL context
;  596          0977  2     arglist             : VECTOR [4, LONG],                 ! Output line storage
;  597          0978  2     status;                                                 ! Routine return status
;  598          0979  2
;  599          0980  2 EXTERNAL LITERAL
;  600          0981  2         show$_objlocked;                                    ! object locked error message
;  601          0982  2
;  602          0983  2 !
;  603          0984  2 ! If we don't need to do this, then exit now, before wasting time initing variables
;  604          0985  2 ! which we won't use.
;  605          0986  2 !
;  606          0987  2 IF NOT .scratch[d_v_acl_present]
;  607          0988  2 THEN
;  608          0989  2     RETURN;
;  609          0990  2
;  610          0991  2 !
;  611          0992  2 ! Initialize all necessary storage.
;  612          0993  2 !
;  613          0994  2 CH$FILL (0, 2*itm$s_item, atr_arglist);
;  614          0995  2 ace_bin_desc[1] = ace_storage;
;  615          0996  2 ace_txt_desc[1] = ace_text;
;  616          0997  2 ace_txt_desc[0] = 512;
;  617          0998  2 arglist[0] = 0;
;  618          0999  2 arglist[1] = ace_text;
;  619          1000  2 object_chan = 0;
;  620          1001  2 object_name[0] = .scratch[d_b_devlen];
;  621          1002  2 object_name[1] = scratch[d_t_device];
;  622          1003  2 object_type = acl$c_device;
;  623          1004  2 display_width = 80;
;  624          1005  2 acl_context = 0;
;  625          1006  2
;  626          1007  2 !
;  627          1008  2 ! Print the heading
```

```
;  628          1009  2 !
;  629          1010  2 show$write_line (ascid_null, arglist);
;  630          1011  2 show$write_line (%ASCID '  Device access control list:', arglist);
;  631          1012  2
;  632          1013  2 !
;  633          1014  2 ! If this is the first time, then map the acl routines from SECURESHR.  This
;  634          1015  2 ! avoids extra image-activation time for mapping SECURESHR, something not
;  635          1016  2 ! frequently needed.
;  636          1017  2 !
;  637          1018  2 IF .format_acl_addr EQL 0
;  638          1019  2 THEN
;  639          1020  3     BEGIN
;  640          1021  3     status = lib$find_image_symbol (ascid_secureshr, %ASCID 'SYS$FORMAT_ACL', format_acl_addr);
;  641          1022  3     IF NOT .status
;  642          1023  3     THEN
;  643          1024  4         BEGIN
;  644          1025  4         SIGNAL (.status);
;  645          1026  4         RETURN;
;  646          1027  3         END;
;  647          1028  3     status = lib$find_image_symbol (ascid_secureshr, %ASCID 'SYS$CHANGE_ACL', change_acl_addr);
;  648          1029  3     IF NOT .status
;  649          1030  3     THEN
;  650          1031  4         BEGIN
;  651          1032  4         SIGNAL (.status);
;  652          1033  4         RETURN;
;  653          1034  3         END;
;  654          1035  2     END;
;  655          1036  2
;  656          1037  2 !
;  657          1038  2 ! Attempt to obtain a read lock for the object.
;  658          1039  2 !
;  659          1040  2 atr_arglist[0, itm$w_itmcod] = acl$c_rlock_acl;
;  660          1041  2 atr_arglist[0, itm$w_bufsiz] = acl$s_rlock_acl;
;  661          1042  2 atr_arglist[0, itm$l_bufadr] = acl_lockid;
;  662        P 1043  2 status = $CHANGE_ACL (CHAN = .object_chan,
;  663        P 1044  2                       OBJTYP = object_type,
;  664        P 1045  2                       OBJNAM = object_name,
;  665          1046  2                       ITMLST = atr_arglist);
;  666          1047  2 IF NOT .status
;  667          1048  2 THEN
;  668          1049  3     BEGIN
;  669          1050  3     IF .status EQL ss$_notqueued THEN status = show$_objlocked;
;  670          1051  3     SIGNAL (.status);
;  671          1052  3     RETURN;
;  672          1053  2     END;
;  673          1054  2
;  674          1055  2 ! Read, format, and display all the ACEs in the object's ACL until the end
;  675          1056  2 ! of the ACL is seen.  Or until an error occurs.
;  676          1057  2
;  677          1058  2 WHILE 1
;  678          1059  2 DO
;  679          1060  3     BEGIN
;  680          1061  3     atr_arglist[0, itm$w_itmcod] = acl$c_readace;
;  681          1062  3     atr_arglist[0, itm$w_bufsiz] = acl$s_readace;
;  682          1063  3     atr_arglist[0, itm$l_bufadr] = ace_storage;
;  683        P 1064  3     status = $CHANGE_ACL (CHAN = .object_chan,
;  684        P 1065  3                           OBJTYP = object_type,
```

```
;  685        P 1066  3                        OBJNAM = object_name,
;  686        P 1067  3                        ITMLST = atr_arglist,
;  687          1068  3                        CONTXT = acl_context);
;  688          1069  3      IF NOT .status
;  689          1070  3      THEN
;  690          1071  4          BEGIN
;  691          1072  4          IF .status EQL ss$_aclempty OR .status EQL ss$_nomoreace THEN EXITLOOP;
;  692          1073  4          SIGNAL (show$_readerr, 1, object_name, .status);
;  693          1074  4          RETURN;
;  694          1075  3          END;
;  695          1076  3      !
;  696          1077  3      ! If the ACE is tagged as being hidden, don't display it.
;  697          1078  3      !
;  698          1079  3      IF NOT .ace_storage[ace$v_hidden]
;  699          1080  3      THEN
;  700          1081  4          BEGIN
;  701          1082  4          !
;  702          1083  4          ! Format and display the ACE.
;  703          1084  4          !
;  704          1085  4          ace_bin_desc[0] = .ace_storage[ace$b_size];
;  705        P 1086  4          status = $FORMAT_ACL (ACLENT = ace_bin_desc,
;  706        P 1087  4                                ACLLEN = arglist[0],
;  707        P 1088  4                                ACLSTR = ace_txt_desc,
;  708        P 1089  4                                WIDTH = display_width,
;  709        P 1090  4                                TRMDSC = $DESCRIPTOR (%CHAR (13), %CHAR (10)),
;  710          1091  4                                INDENT = %REF (4));
;  711          1092  4          IF NOT .status
;  712          1093  4          THEN
;  713          1094  5              BEGIN
;  714          1095  5              SIGNAL (.status);
;  715          1096  5              RETURN;
;  716          1097  4              END;
;  717          1098  4          !
;  718          1099  4          ! Show the entry
;  719          1100  4          !
;  720          1101  4          show$write_line (%ASCID '!AD', arglist);
;  721          1102  4
;  722          1103  3          END;
;  723          1104  2      END;
;  724          1105  2
;  725          1106  2 RETURN;
;  726          1107  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

73  73  65  63  63  61  20  65  63  69  76  65  44  20  20  0030C P.AEC:  .ASCII  \  Device access control list:\<0><0>      ;
00  3A  74  73  69  6C  20  6C  6F  72  74  6E  6F  63  20  0031B                                                          ;
                                                        00  0032A                                                          ;
                                                        00  0032B          .ASCII  <0>                                     ;
                                                  010E001D  0032C P.AEB:   .LONG   17694749                                ;
                                                  00000000' 00330          .ADDRESS P.AEC                                  ;
00  4C  43  41  5F  54  41  4D  52  4F  46  24  53  59  53  00334 P.AEE:   .ASCII  \SYS$FORMAT_ACL\<0><0>                  ;
                                                        00  00343                                                          ;
                                                  010E000E  00344 P.AED:   .LONG   17694734                                ;
                                                  00000000' 00348          .ADDRESS P.AEE                                  ;
```

```
00 4C 43 41 5F 45 47 4E 41 48 43 24 53 59 53 0034C P.AEG:   .ASCII  \SYS$CHANGE_ACL\<0><0>          ;
                                          00 0035B                                                ;
                                   010E000E 0035C P.AEF:   .LONG   17694734                        ;
                                  00000000' 00360          .ADDRESS P.AEG                          ;
                                         0D 00364 P.AEI:   .ASCII  <13>                            ;
                                         0A 00365          .ASCII  <10>                            ;
                                            00366          .BLKB   2
                                   00000002 00368 P.AEH:   .LONG   2                               ;
                                  00000000' 0036C          .ADDRESS P.AEI                          ;
                              00 44 41 21 00370 P.AEK:   .ASCII  \!AD\<0>                        ;
                                   010E0003 00374 P.AEJ:   .LONG   17694723                        ;
                                  00000000' 00378          .ADDRESS P.AEK                          ;

                                                           .EXTRN  SHOW$_OBJLOCKED
                                                           .EXTRN  SYS$CHANGE_ACL, SYS$FORMAT_ACL

                                                           .PSECT  $CODE$,NOWRT,2

                                     07FC 00000            .ENTRY  SHOW_DEVICE_ACL, Save R2,R3,R4,R5,R6,R7,R8,-; 0952
                                                                   R9,R10
                   5A 00000000G  00  9E 00002            MOVAB   LIB$SIGNAL, R10                 ;
                   59 00000000G  00  9E 00009            MOVAB   SYS$CHANGE_ACL, R9              ;
                   58 00000000G  00  9E 00010            MOVAB   LIB$FIND_IMAGE_SYMBOL, R8       ;
                   57 00000000G  00  9E 00017            MOVAB   SHOW$WRITE_LINE, R7             ;
                   5E      FCAC  CE  9E 0001E            MOVAB   -852(SP), SP                    ;
                   56        04  AC  D0 00023            MOVL    SCRATCH, R6                     ; 0987
         01       05  A6         01  E0 00027            BBS     #1, 5(R6), 1$                   ;
                                 04 0002C                RET                                     ;
   18    00           6E         00  2C 0002D 1$:        MOVC5   #0, (SP), #0, #24, ATR_ARGLIST  ; 0994
                             24  AE     00032                                                    ;
             F4   AD       FEF0  CD  9E 00034            MOVAB   ACE_STORAGE, ACE_BIN_DESC+4     ; 0995
           FEEC   CD         3C  AE  9E 0003A            MOVAB   ACE_TEXT, ACE_TXT_DESC+4        ; 0996
           FEE8   CD       0200  8F  3C 00040            MOVZWL  #512, ACE_TXT_DESC              ; 0997
                             14  AE  D4 00047            CLRL    ARGLIST                         ; 0998
             18   AE         3C  AE  9E 0004A            MOVAB   ACE_TEXT, ARGLIST+4             ; 0999
                                 53  D4 0004F            CLRL    OBJECT_CHAN                     ; 1000
             F8   AD         06  A6  9A 00051            MOVZBL  6(R6), OBJECT_NAME              ; 1001
             FC   AD         08  A6  9E 00056            MOVAB   8(R6), OBJECT_NAME+4            ; 1002
             0C   AE             02  D0 0005B            MOVL    #2, OBJECT_TYPE                 ; 1003
             10   AE         50  8F  9A 0005F            MOVZBL  #80, DISPLAY_WIDTH              ; 1004
                             08  AE  D4 00064            CLRL    ACL_CONTEXT                     ; 1005
                             14  AE  9F 00067            PUSHAB  ARGLIST                         ; 1010
                          0000'  CF  9F 0006A            PUSHAB  ASCID_NULL                      ;
                  67             02  FB 0006E            CALLS   #2, SHOW$WRITE_LINE             ;
                             14  AE  9F 00071            PUSHAB  ARGLIST                         ; 1011
                          0000'  CF  9F 00074            PUSHAB  P.AEB                           ;
                  67             02  FB 00078            CALLS   #2, SHOW$WRITE_LINE             ;
                          0000'  CF  D5 0007B            TSTL    FORMAT_ACL_ADDR                 ; 1018
                                 2A  12 0007F            BNEQ    2$                              ;
                          0000'  CF  9F 00081            PUSHAB  FORMAT_ACL_ADDR                 ; 1021
                          0000'  CF  9F 00085            PUSHAB  P.AED                           ;
                          0000'  CF  9F 00089            PUSHAB  ASCID_SECURESHR                 ;
                  68             03  FB 0008D            CALLS   #3, LIB$FIND_IMAGE_SYMBOL       ;
                  52             50  D0 00090            MOVL    R0, STATUS                      ;
                  4A             52  E9 00093            BLBC    STATUS, 3$                      ; 1022
                          0000'  CF  9F 00096            PUSHAB  CHANGE_ACL_ADDR                 ; 1028
                          0000'  CF  9F 0009A            PUSHAB  P.AEF                           ;
```

```
                        0000'  CF  9F 0009E          PUSHAB  ASCID_SECURESHR                 ;
                    68          03  FB 000A2          CALLS   #3, LIB$FIND_IMAGE_SYMBOL       ;
                    52          50  D0 000A5          MOVL    R0, STATUS                      ;
                    35          52  E9 000A8          BLBC    STATUS, 3$                      ; 1029
                24  AE 000A0004 8F  D0 000AB 2$:      MOVL    #655364, ATR_ARGLIST            ; 1041
                28  AE      04 AE  9E 000B3          MOVAB   ACL_LOCKID, ATR_ARGLIST+4       ; 1042
                                7E  7C 000B8          CLRQ    -(SP)                           ; 1046
                                7E  D4 000BA          CLRL    -(SP)                           ;
                            30  AE  9F 000BC          PUSHAB  ATR_ARGLIST                     ;
                            F8  AD  9F 000BF          PUSHAB  OBJECT_NAME                     ;
                            20  AE  9F 000C2          PUSHAB  OBJECT_TYPE                     ;
                                53  DD 000C5          PUSHL   OBJECT_CHAN                     ;
                    69          07  FB 000C7          CALLS   #7, SYS$CHANGE_ACL              ;
                    52          50  D0 000CA          MOVL    R0, STATUS                      ;
                    12          52  E8 000CD          BLBS    STATUS, 4$                      ; 1047
            000009B8 8F         52  D1 000D0          CMPL    STATUS, #2488                   ; 1050
                                07  12 000D7          BNEQ    3$                              ;
                    52 00000000G 8F D0 000D9          MOVL    #SHOW$_OBJLOCKED, STATUS        ;
                                7D  11 000E0 3$:      BRB     6$                              ; 1051
                24  AE 000900FF  8F  D0 000E2 4$:      MOVL    #590079, ATR_ARGLIST            ; 1062
                28  AE     FEF0 CD  9E 000EA          MOVAB   ACE_STORAGE, ATR_ARGLIST+4      ; 1063
                            08  AE  9F 000F0          PUSHAB  ACL_CONTEXT                     ; 1068
                                7E  7C 000F3          CLRQ    -(SP)                           ;
                            30  AE  9F 000F5          PUSHAB  ATR_ARGLIST                     ;
                            F8  AD  9F 000F8          PUSHAB  OBJECT_NAME                     ;
                            20  AE  9F 000FB          PUSHAB  OBJECT_TYPE                     ;
                                53  DD 000FE          PUSHL   OBJECT_CHAN                     ;
                    69          07  FB 00100          CALLS   #7, SYS$CHANGE_ACL              ;
                    52          50  D0 00103          MOVL    R0, STATUS                      ;
                    23          52  E8 00106          BLBS    STATUS, 5$                      ; 1069
            000009D0 8F         52  D1 00109          CMPL    STATUS, #2512                   ; 1072
                                60  13 00110          BEQL    8$                              ;
            000009E0 8F         52  D1 00112          CMPL    STATUS, #2528                   ;
                                57  13 00119          BEQL    8$                              ;
                                52  DD 0011B          PUSHL   STATUS                          ; 1073
                            F8  AD  9F 0011D          PUSHAB  OBJECT_NAME                     ;
                                01  DD 00120          PUSHL   #1                              ;
                        007810B4 8F  DD 00122          PUSHL   #7868596                        ;
                    6A          04  FB 00128          CALLS   #4, LIB$SIGNAL                  ;
                                04  00 0012B          RET                                     ; 1071
            B0      FEF3 CD         02  E0 0012C 5$:      BBS     #2, ACE_STORAGE+3, 4$       ; 1079
                    F0  AD      FEF0 CD  9A 00132          MOVZBL  ACE_STORAGE, ACE_BIN_DESC   ; 1085
                                7E  D4 00138          CLRL    -(SP)                           ; 1091
                    04  AE          04  D0 0013A          MOVL    #4, 4(SP)                       ;
                            04  AE  9F 0013E          PUSHAB  4(SP)                           ;
                            0000' CF  9F 00141          PUSHAB  P.AEH                           ;
                            1C  AE  9F 00145          PUSHAB  DISPLAY_WIDTH                   ;
                            FEE8 CD  9F 00148          PUSHAB  ACE_TXT_DESC                    ;
                            28  AE  9F 0014C          PUSHAB  ARGLIST                         ;
                            F0  AD  9F 0014F          PUSHAB  ACE_BIN_DESC                    ;
            00000000G       00      07  FB 00152          CALLS   #7, SYS$FORMAT_ACL              ;
                            52      50  D0 00159          MOVL    R0, STATUS                      ;
                            06      52  E8 0015C          BLBS    STATUS, 7$                      ; 1092
                                    52  DD 0015F 6$:      PUSHL   STATUS                          ; 1095
                            6A      01  FB 00161          CALLS   #1, LIB$SIGNAL                  ;
                                    04  00 00164          RET                                     ; 1094
                            14  AE  9F 00165 7$:      PUSHAB  ARGLIST                         ; 1101
```

```
                0000'  CF  9F 00168          PUSHAB  P.AEJ                                   ;
                    67     02  FB 00016C     CALLS   #2, SHOW$WRITE_LINE                     ;
                         FF70  31 00016F     BRW     4$                                      ; 1058
                               04 00172 8$:  RET                                             ; 1107

; Routine Size:  371 bytes,    Routine Base:  $CODE$ + 01D3
```

```
;  728          1108  1 GLOBAL ROUTINE sys$change_acl =
;  729          1109  2 BEGIN
;  730          1110  2
;  731          1111  2 !++
;  732          1112  2 !
;  733          1113  2 ! FUNCTIONAL DESCRIPTION:
;  734          1114  2 !
;  735          1115  2 !       This is a dummy routine to satisfy the global reference of
;  736          1116  2 !       the $CHANGE_ACL macro.  It simply calls the real service,
;  737          1117  2 !       after dynamically loading the routine.
;  738          1118  2 !
;  739          1119  2 ! CALLING SEQUENCE:
;  740          1120  2 !       via $CHANGE_ACL macro
;  741          1121  2 !
;  742          1122  2 ! ROUTINE VALUE:
;  743          1123  2 !       status returned from sys$change_acl service
;  744          1124  2 !
;  745          1125  2 !--
;  746          1126  2
;  747          1127  2 BUILTIN
;  748          1128  2     CALLG,AP;
;  749          1129  2
;  750          1130  2 LOCAL
;  751          1131  2     status;
;  752          1132  2
;  753          1133  2 RETURN CALLG (.AP, .change_acl_addr);
;  754          1134  1 END;


                                0000 00000          .ENTRY  SYS$CHANGE_ACL, Save nothing                    ; 1108
                      0000'  DF      6C  FA 00002          CALLG   (AP), @CHANGE_ACL_ADDR                  ; 1133
                                      04 00007          RET                                             ; 1134

; Routine Size:  8 bytes,    Routine Base:  $CODE$ + 0346
```

```
;   756          1135  1 GLOBAL ROUTINE sys$format_acl =
;   757          1136  2 BEGIN
;   758          1137  2
;   759          1138  2 !++
;   760          1139  2 !
;   761          1140  2 ! FUNCTIONAL DESCRIPTION:
;   762          1141  2 !
;   763          1142  2 !       This is a dummy routine to satisfy the global reference of
;   764          1143  2 !       the $FORMAT_ACL macro.  It simply calls the real service,
;   765          1144  2 !       after dynamically loading the routine.
;   766          1145  2 !
;   767          1146  2 ! CALLING SEQUENCE:
;   768          1147  2 !       via $FORMAT_ACL macro
;   769          1148  2 !
;   770          1149  2 ! ROUTINE VALUE:
;   771          1150  2 !       status returned from sys$format_acl service
;   772          1151  2 !
;   773          1152  2 !--
;   774          1153  2
;   775          1154  2 BUILTIN
;   776          1155  2     CALLG,AP;
;   777          1156  2
;   778          1157  2 LOCAL
;   779          1158  2     status;
;   780          1159  2
;   781          1160  2 RETURN CALLG (.AP, .format_acl_addr);
;   782          1161  1 END;


                                0000 00000          .ENTRY  SYS$FORMAT_ACL, Save nothing                    ; 1135
                      0000'  DF         6C  FA 00002          CALLG   (AP), @FORMAT_ACL_ADDR                  ; 1160
                                           04 00007          RET                                             ; 1161

; Routine Size:  8 bytes,    Routine Base:  $CODE$ + 034E
```

```
;  784          1162  1 GLOBAL ROUTINE display_brief (scratch, flags) : NOVALUE =
;  785          1163  2 BEGIN
;  786          1164  2
;  787          1165  2 !---
;  788          1166  2 !
;  789          1167  2 ! This is the brief display routine for all devices
;  790          1168  2 !
;  791          1169  2 ! Inputs
;  792          1170  2 !       SCRATCH - contains data about the device
;  793          1171  2 !       FLAGS   - contains info on what to print
;  794          1172  2 !
;  795          1173  2 ! Outputs
;  796          1174  2 !       None
;  797          1175  2 !
;  798          1176  2 !---
;  799          1177  2
;  800          1178  2 MAP
;  801          1179  2     scratch : REF $BBLOCK,
;  802          1180  2     flags : REF $BBLOCK;
;  803          1181  2
;  804          1182  2 LOCAL
;  805          1183  2     show_mnt,                                  ! Show the string "mnt" in the status line
;  806          1184  2     ptr : REF VECTOR,                          ! Pointer to char. descriptor
;  807          1185  2     desc_vector : VECTOR[16]                   ! and the descriptors themselves
;  808          1186  2                   INITIAL(REP 16 OF (0)),
;  809          1187  2     desc : VECTOR[2],                          ! Characteristics descriptor
;  810          1188  2     buffer : VECTOR[80, BYTE],                 ! Characteristics buffer
;  811          1189  2     nambuf : VECTOR[64, BYTE],                 ! Buffer for device name
;  812          1190  2     clubuf : VECTOR[1024, BYTE],               ! Buffer for lock information
;  813          1191  2     arglist : VECTOR[15];                      ! SHOW$WRITE_LINE argument list
;  814          1192  2
;  815          1193  2 !
;  816          1194  2 ! First thing, fix up the scratch area with any available clusterwide info.  Note
;  817          1195  2 ! that we might set the dev$v_mnt or dev$v_all bits in this routine
;  818          1196  2 !
;  819          1197  2 IF .$BBLOCK[scratch[d_l_devchar2], dev$v_clu]   ! If available clusterwide
;  820          1198  2 THEN
;  821          1199  2     scan_cluster_locks (.scratch, clubuf);
;  822          1200  2
;  823          1201  2 !
;  824          1202  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;  825          1203  2 ! then check to see if the device is mounted/allocated.  If not, then
;  826          1204  2 ! don't display information on the device.
;  827          1205  2 !
;  828          1206  2 IF NOT check_mnt_all(.scratch, .flags)
;  829          1207  2 THEN RETURN;
;  830          1208  2
;  831          1209  2 !
;  832          1210  2 ! Write a header if it hasn't been done already.
;  833          1211  2 !
;  834          1212  2 IF .flags[devi$v_header]
;  835          1213  2 THEN
;  836          1214  3     BEGIN
;  837          1215  3     IF .scratch[d_b_devclass] EQLU dc$_disk
;  838          1216  3     OR .scratch[d_b_devclass] EQLU dc$_tape
;  839          1217  3     THEN show$write_line(ascid_null, arglist[0],
;  840          1218  3                          %ASCID 'Device                  Device           Error    Volume         Free  Tran
```

```
;  841     1219  3                          arglist[0],
;  842     1220  3                          %ASCID ' Name                    Status          Count     Label        Blocks Coun
;  843     1221  3                          arglist[0])
;  844     1222  3      ELSE IF .scratch[d_b_devclass] EQLU dc$_journal
;  845     1223  3      THEN show$write_line(ascid_null, arglist[0],
;  846     1224  3                          %ASCID 'Device                   Device             Journal        Buff  Cur. Seque
;  847     1225  3                          arglist[0],
;  848     1226  3                          %ASCID ' Name                    Status              Name          Size      Number
;  849     1227  3                          arglist[0])
;  850     1228  3      ELSE show$write_line(ascid_null, arglist[0],
;  851     1229  3                          %ASCID 'Device                   Device          Error',
;  852     1230  3                          arglist[0],
;  853     1231  3                          %ASCID ' Name                    Status          Count',
;  854     1232  3                          arglist[0]);
;  855     1233  3      flags[devi$v_header] = false;
;  856     1234  2      END;
;  857     1235  2
;  858     1236  2  !
;  859     1237  2  ! Get the device name.  The actual string is of the form "_DDCU", so shuffle
;  860     1238  2  ! to get rid of the underscore.
;  861     1239  2  !
;  862     1240  2  arglist[0] = .scratch[d_b_devlen] - 1;
;  863     1241  2  arglist[1] = scratch[d_t_device] + 1;
;  864     1242  2
;  865     1243  2  !
;  866     1244  2  ! If there is an allocation class device name, put the name of the host in
;  867     1245  2  ! parentheses after the device name, right justified.  For example, we want to see
;  868     1246  2  !       $255$DUA17:  (PRIDE)
;  869     1247  2  !       $255$DUA19:   (LUST)
;  870     1248  2  !
;  871     1249  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_fod]
;  872     1250  2   AND
;  873     1251  2     .scratch[d_l_allocls] NEQ 0
;  874     1252  2  THEN
;  875     1253  3      BEGIN
;  876     1254  3      LOCAL
;  877     1255  3          blen,                                   ! Length of beginning, devname plus blanks to fill
;  878     1256  3          hlen,                                   ! Length of host name
;  879     1257  3          tlen;                                   ! Temporary len
;  880     1258  3      hlen = .scratch[d_t_host_name];             ! Get the host name length into a temporary
;  881     1259  3      tlen = .arglist[0] + .hlen + 2;             ! Device name + host name + parentheses
;  882     1260  3      blen = MAX (1, 22 - .tlen) + .arglist[0];   ! Count of blanks between device and host (at least one), +
;  883     1261  3      tlen = .blen + .hlen + 2;                   ! Total length
;  884     1262  3      CH$COPY (.arglist[0], .arglist[1], %C' ',   ! Move the device name to the scratch buffer
;  885     1263  3               .blen, nambuf);                    !  adding spaces for right justification
;  886     1264  3      nambuf[.blen] = %C'(';                      ! Add the left parens
;  887     1265  3      CH$MOVE (.hlen, scratch[d_t_host_name]+1,   ! Move the primary (or only) host name
;  888     1266  3               nambuf[.blen+1]);                  !  after the left parens
;  889     1267  3      nambuf[.tlen-1] = %C')';                    ! Add the right parens
;  890     1268  3      arglist[0] = .tlen;                         ! Pass the new length
;  891     1269  3      arglist[1] = nambuf;                        ! And the buffer
;  892     1270  3      IF .tlen GTR 23                             ! If it is too big for our area
;  893     1271  3      THEN
;  894     1272  4          BEGIN
;  895     1273  4          show$write_line(%ASCID '!AF', arglist); ! Print the names on their own line
;  896     1274  4          arglist[0]= 0;                          ! Zero the length so that it will not print again
;  897     1275  3          END;
```

```
;  898          1276  2     END;
;  899          1277  2
;  900          1278  2 !
;  901          1279  2 ! Fill in the status and characteristics buffer
;  902          1280  2 !
;  903          1281  2 ptr = desc_vector[0];                          ! Initialize the descriptor pointer
;  904          1282  2 desc_vector[1] = buffer;                       !  and the first descriptor and
;  905          1283  2 leading_blanks = 0;                            !  and say no leading blanks
;  906          1284  2 arglist[2] = desc_vector;                      ! Put descriptor in arg list
;  907          1285  2 show_mnt = .$BBLOCK[scratch[d_l_devchar], dev$v_mnt];   ! Make a temp for the "mnt" bit
;  908          1286  2 !
;  909          1287  2 ! If the device is online, see if we can expand a more meaningful string
;  910          1288  2 !
;  911          1289  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_online), 1, 0]
;  912          1290  2 THEN
;  913          1291  2     BEGIN
;  914          1292  3     !
;  915          1293  3     ! If the primary host is not AVAIL, then call it "HostUnavailable"
;  916          1294  3     !
;  917          1295  3     IF NOT .scratch[d_v_host_avail]
;  918          1296  3     THEN add_to_list(ptr, 'HostUnavailable', 16)
;  919          1297  3     !
;  920          1298  3     ! Disconnected terminals should say so, rather than "Online det"
;  921          1299  3     !
;  922          1300  3     ELSE IF .$BBLOCK[scratch[d_l_devchar2], dev$v_det]  ! detached from physical
;  923          1301  3     THEN add_to_list(ptr, 'Disconnected', 16)
;  924          1302  3     !
;  925          1303  3     ! If SET DEVICE /NOAVAIL was done, then call it "Unavailable"
;  926          1304  3     !
;  927          1305  3     ELSE IF NOT .$BBLOCK[scratch[d_l_devchar], dev$v_avl]
;  928          1306  3     THEN add_to_list(ptr, 'Unavailable  ', 16)
;  929          1307  3     !
;  930          1308  3     ! If mount verification found the wrong volume, mark it "WrongVolume"
;  931          1309  3     !
;  932          1310  3     ELSE IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_wrongvol), 1, 0]
;  933          1311  3     THEN add_to_list(ptr, 'WrongVolume  ', 16)
;  934          1312  3     !
;  935          1313  3     ! If the volume is undergoing mount verification, mark it "MntVerify"
;  936          1314  3     !
;  937          1315  3     ELSE IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_mntverip), 1, 0]
;  938          1316  3     THEN add_to_list(ptr, 'MountVerify  ', 16)
;  939          1317  3     !
;  940          1318  3     ! If the volume is not valid AND mounted, then it has failed mount
;  941          1319  3     ! verification and it should be marked "MntVerfyTimeout"
;  942          1320  3     !
;  943          1321  3     ELSE IF ((NOT .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_valid), 1, 0])
;  944          1322  4           AND .$BBLOCK[scratch[d_l_devchar], dev$v_mnt])       ! Mounted
;  945          1323  5           AND ((.scratch[d_b_aqbtype] EQL aqb$k_f11v1)
;  946          1324  4                OR
;  947          1325  4                (.scratch[d_b_aqbtype] EQL aqb$k_f11v2))
;  948          1326  3     THEN add_to_list(ptr, 'MntVerifyTimeout', 16)
;  949          1327  3     !
;  950          1328  3     ! If the volume is mounted, then call it "Mounted" instead of "Online mnt".
;  951          1329  3     !
;  952          1330  3     ELSE IF .$BBLOCK[scratch[d_l_devchar], dev$v_mnt]
;  953          1331  3     THEN
;  954          1332  4         BEGIN
```

```
;  955          1333  4          show_mnt = false;                       ! Don't show the mnt bit agair
;  956          1334  4          add_to_list(ptr, 'Mounted', 16)
;  957          1335  4          END
;  958          1336  4      !
;  959          1337  4      ! Otherwise just call it "Online"
;  960          1338  4      !
;  961          1339  3      ELSE add_to_list(ptr, 'Online', 16)         ! Say whether online or
;  962          1340  3      END
;  963          1341  3  !
;  964          1342  3  ! Nothing else matched, call it offline
;  965          1343  3  !
;  966          1344  2  ELSE
;  967          1345  2      add_to_list(ptr, 'Offline', 16);            ! offline.
;  968          1346  2  !
;  969          1347  2  ! Now that the major status is set, explode any other interesting bits
;  970          1348  2  !
;  971          1349  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_spl]    ! Spooled
;  972          1350  2  THEN add_to_list(ptr, ' spooled', 16);
;  973          1351  2  IF .show_mnt                                    ! Mounted
;  974          1352  2  THEN add_to_list(ptr, ' mounted', 16);
;  975          1353  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_dmt]    ! Marked for dismount
;  976          1354  2  THEN add_to_list(ptr, ' dismount', 16);
;  977          1355  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_all]    ! Allocated explicitly
;  978          1356  2  THEN add_to_list(ptr, ' alloc', 16);
;  979          1357  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_for]    ! Foreign
;  980          1358  2  THEN add_to_list(ptr, ' foreign', 16);
;  981          1359  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_swl]    ! Software-writelocked
;  982          1360  2  THEN add_to_list(ptr, ' wrtlck', 16);
;  983          1361  2  IF .scratch[d_b_devclass] EQLU dc$_journal
;  984          1362  2  THEN
;  985          1363  3      BEGIN
;  986          1364  3      IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_deadmo), 1, 0]
;  987          1365  3      THEN add_to_list(ptr, ' del', 16);  ! Delete pending (for RU journals).
;  988          1366  3      IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_jnl_cls), 1, 0]
;  989          1367  3      THEN add_to_list(ptr, ' cls', 16);
;  990          1368  3      IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_jnl_slv), 1, 0]
;  991          1369  3      THEN add_to_list(ptr, ' slv', 16);
;  992          1370  3      IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_known_jnl), 1, 0]
;  993          1371  3      THEN add_to_list(ptr, ' knw', 16)
;  994          1372  3      ELSE
;  995          1373  4          BEGIN
;  996          1374  4          IF .$BBLOCK[scratch[d_l_jnl_char], 0, $BITPOSITION(vcb$v_jnl_tmpfi), 1, 0]
;  997          1375  4          THEN add_to_list(ptr, ' tf', 16)
;  998          1376  4          ELSE add_to_list(ptr, ' pf', 16);
;  999          1377  3          END;
; 1000          1378  3      IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_perm_jnl), 1, 0]
; 1001          1379  3      THEN add_to_list(ptr, ' pj', 16)
; 1002          1380  3      ELSE add_to_list(ptr, ' tj', 16);
; 1003          1381  2      END;
; 1004          1382  2
; 1005          1383  2  !
; 1006          1384  2  ! Miscellaneous data
; 1007          1385  2  !
; 1008          1386  2  arglist[3] = .scratch[d_w_errcnt];                    ! Error count
; 1009          1387  2
; 1010          1388  2  !
; 1011          1389  2  ! Check to see if a long (not FULL) display
```

```
;1012     1390  2 !
;1013     1391  4 IF ((.scratch[d_b_devclass] EQLU dc$_disk                    ! If mounted disk,
;1014     1392  4 OR .scratch[d_b_devclass] EQLU dc$_tape)                     ! tape,
;1015     1393  3 AND .$BBLOCK[scratch[d_l_devchar], dev$v_mnt])
;1016     1394  2 OR .scratch[d_b_devclass] EQLU dc$_journal                   ! journal, or ...
;1017     1395  2 THEN
;1018     1396  2     BEGIN
;1019     1397  2
;1020     1398  2 !
;1021     1399  2 ! Print out the volume label and other crud.
;1022     1400  2 !
;1023     1401  2     arglist[5] = scratch[d_t_volnam];
;1024     1402  2     IF .scratch[d_b_devclass] EQLU dc$_disk
;1025     1403  2     OR .scratch[d_b_devclass] EQLU dc$_tape
;1026     1404  3     THEN                                                    ! Put in disk/tape data
;1027     1405  4         BEGIN
;1028     1406  4         arglist[6] = .scratch[d_l_free];                    ! Free blocks on volume
;1029     1407  4         arglist[7] = .scratch[d_w_trans];                   ! Transaction count
;1030     1408  4         arglist[8] = .scratch[d_w_mcount];                  ! Mount count
;1031     1409  4         END
;1032     1410  3     ELSE                                                    ! Put in journal stuff
;1033     1411  4         BEGIN
;1034     1412  4         arglist[6] = .scratch[d_w_devbufsiz]/512;           ! Buffer size in blocks
;1035     1413  4         arglist[7] = .scratch[d_l_jnl_seqno];
;1036     1414  3         END;
;1037     1415  3
;1038     1416  3 !
;1039     1417  3 ! For disk and journal, the volume label is 12 characters long, while magtapes have
;1040     1418  3 ! volume names that are only 6 bytes long.  Oh well...
;1041     1419  3 !
;1042     1420  3     IF .scratch[d_b_devclass] EQLU dc$_disk
;1043     1421  3     THEN
;1044     1422  4         BEGIN
;1045     1423  4         IF .scratch[d_b_cont] OR .scratch[d_v_remote_mounts]
;1046     1424  4         THEN arglist[4] = vcb$s_volname                     ! Volume label
;1047     1425  4         ELSE arglist[4] = 0;
;1048     1426  4         show$write_line(%ASCID '!23<!AF!> !16AS !5UW  !12AF !9UL!6UW!4UW', arglist);
;1049     1427  4         END
;1050     1428  3     ELSE IF .scratch[d_b_devclass] EQLU dc$_journal
;1051     1429  3     THEN
;1052     1430  4         BEGIN
;1053     1431  4         IF .scratch[d_b_cont]
;1054     1432  4         THEN
;1055     1433  5             BEGIN
;1056     1434  5             arglist[4] = ucb$s_jnl_nam;                     ! Journal name
;1057     1435  5             show$write_line(%ASCID '!23<!AF!> !16AS!+ !18AF !4UL     !10UL', arglist);
;1058     1436  5             END
;1059     1437  4         ELSE show$write_line(%ASCID '!23<!AF!> !16AS', arglist);
;1060     1438  4         END
;1061     1439  3     ELSE
;1062     1440  4         BEGIN
;1063     1441  4         IF .scratch[d_b_cont]
;1064     1442  4         THEN arglist[4] = mvl$s_vollbl                          ! Volume label
;1065     1443  4         ELSE arglist[4] = 0;
;1066     1444  4         show$write_line(%ASCID '!23<!AF!> !16AS !5UW     !6AF    !9UL!6UW!4UW',arglist);
;1067     1445  3         END;
;1068     1446  3     END
```

```
;  1069          1447  2 ELSE
;  1070          1448  2     show$write_line(%ASCID '!23<!AF!> !16AS !5UW', arglist);
;  1071          1449  2
;  1072          1450  2 !
;  1073          1451  2 ! Print out any additional information required.
;  1074          1452  2 !
;  1075          1453  2 INCR i FROM 1 TO 7 DO
;  1076          1454  3     BEGIN
;  1077          1455  3     BIND string = desc_vector[2*.i] : VECTOR;
;  1078          1456  3     IF .string[0] NEQ 0
;  1079          1457  3     THEN show$write_line(%ASCID '!23< !> !16AS', %REF(string))
;  1080          1458  3     ELSE EXITLOOP
;  1081          1459  2     END;
;  1082          1460  2
;  1083          1461  2 RETURN;
;  1084          1462  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                    00000000#  0037C P.AEL:  .LONG   0[16]
20 20 20 20 20 20 20 20 20 65 63 69 76 65 44  003BC P.AEN:  .ASCII  \Device                    Device          \
65 63 69 76 65 44 20 20 20 20 20 20 20 20 20  003CB
                  20 20 20 20 20 20 20 20 20  003DA
6D 75 6C 6F 56 20 20 20 20 72 6F 72 72 45 20  003E4          .ASCII  \ Error    Volume          Free  Trans Mnt\
20 65 65 72 46 20 20 20 20 20 20 20 20 65 20  003F3
                  74 6E 4D 20 73 6E 61 72 54  00402
                                              010E0050  0040C P.AEM:  .LONG   17694800
                                              00000000'  00410          .ADDRESS P.AEN
20 20 20 20 20 20 20 20 20 20 65 6D 61 4E 20  00414 P.AEP:  .ASCII  \ Name                      Status          \
73 75 74 61 74 53 20 20 20 20 20 20 20 20 20  00423
                  20 20 20 20 20 20 20 20 20  00432
65 62 61 4C 20 20 20 20 20 74 6E 75 6F 43 20  0043C          .ASCII  \ Count     Label         Blocks Count Cnt\
73 6B 63 6F 6C 42 20 20 20 20 20 20 20 6C 20  0044B
                  74 6E 43 20 74 6E 75 6F 43  0045A
                                              010E0050  00464 P.AEO:  .LONG   17694800
                                              00000000'  00468          .ADDRESS P.AEP
20 20 20 20 20 20 20 20 20 65 63 69 76 65 44  0046C P.AER:  .ASCII  \Device                    Device          \
65 63 69 76 65 44 20 20 20 20 20 20 20 20 20  0047B
                  20 20 20 20 20 20 20 20 20  0048A
20 20 20 20 6C 61 6E 72 75 6F 4A 20 20 20 20  00494          .ASCII  \    Journal        Buff  Cur. Sequence-
20 2E 72 75 43 20 20 66 66 75 42 20 20 20 20  004A3                  \<0>
                  00 65 63 6E 65 75 71 65 53  004B2
                                        00    004BB          .ASCII  <0>
                                              010E004E  004BC P.AEQ:  .LONG   17694798
                                              00000000'  004C0          .ADDRESS P.AER
20 20 20 20 20 20 20 20 20 20 65 6D 61 4E 20  004C4 P.AET:  .ASCII  \ Name                      Status          \
73 75 74 61 74 53 20 20 20 20 20 20 20 20 20  004D3
                  20 20 20 20 20 20 20 20 20  004E2
20 20 20 20 20 20 65 6D 61 4E 20 20 20 20 20  004EC          .ASCII  \     Name          Size      Number\<0>
4E 20 20 20 20 20 20 65 7A 69 53 20 20 20 20  004FB
                        00 72 65 62 6D 75 4E  0050A
                                              010E004B  00510 P.AES:  .LONG   17694795
                                              00000000'  00514          .ADDRESS P.AET
20 20 20 20 20 20 20 20 20 65 63 69 76 65 44  00518 P.AEV:  .ASCII  \Device                    Device          \
65 63 69 76 65 44 20 20 20 20 20 20 20 20 20  00527
```

```
                                20 20 20 20 20 20 20 20 20 20 20 20  00536
                                00 00 72 6F 72 72 45 20              00540          .ASCII  \ Error\<0><0>
                                   010E002E                          00548 P.AEU:   .LONG   17694766
                                   00000000'                         0054C          .ADDRESS P.AEV
20 20 20 20 20 20 20 20 20 20 65 6D 61 4E 20                         00550 P.AEX:   .ASCII  \ Name                          Status          \
73 75 74 61 74 53 20 20 20 20 20 20 20 20 20                         0055F
                20 20 20 20 20 20 20 20 20 20 20                     0056E
                00 00 74 6E 75 6F 43 20                              00578          .ASCII  \ Count\<0><0>
                   010E002E                                          00580 P.AEW:   .LONG   17694766
                   00000000'                                         00584          .ADDRESS P.AEX
                                00 46 41 21                          00588 P.AEZ:   .ASCII  \!AF\<0>
                   010E0003                                          0058C P.AEY:   .LONG   17694723
                   00000000'                                         00590          .ADDRESS P.AEZ
65 6C 62 61 6C 69 61 76 61 6E 55 74 73 6F 48                         00594 P.AFB:   .ASCII  \HostUnavailable\<0>
                                                  00                 005A3
                   010E000F                                          005A4 P.AFA:   .LONG   17694735
                   00000000'                                         005A8          .ADDRESS P.AFB
                   64 65 74 63 65 6E 6E 6F 63 73 69 44               005AC P.AFD:   .ASCII  \Disconnected\
                   010E000C                                          005B8 P.AFC:   .LONG   17694732
                   00000000'                                         005BC          .ADDRESS P.AFD
00 00 20 20 65 6C 62 61 6C 69 61 76 61 6E 55                         005C0 P.AFF:   .ASCII  \Unavailable  \<0><0><0>
                                                  00                 005CF
                   010E000D                                          005D0 P.AFE:   .LONG   17694733
                   00000000'                                         005D4          .ADDRESS P.AFF
00 00 20 20 65 6D 75 6C 6F 56 67 6E 6F 72 57                         005D8 P.AFH:   .ASCII  \WrongVolume  \<0><0><0>
                                                  00                 005E7
                   010E000D                                          005E8 P.AFG:   .LONG   17694733
                   00000000'                                         005EC          .ADDRESS P.AFH
00 00 20 20 79 66 69 72 65 56 74 6E 75 6F 4D                         005F0 P.AFJ:   .ASCII  \MountVerify  \<0><0><0>
                                                  00                 005FF
                   010E000D                                          00600 P.AFI:   .LONG   17694733
                   00000000'                                         00604          .ADDRESS P.AFJ
75 6F 65 6D 69 54 79 66 69 72 65 56 74 6E 4D                         00608 P.AFL:   .ASCII  \MntVerifyTimeout\
                                                  74                 00617
                   010E0010                                          00618 P.AFK:   .LONG   17694736
                   00000000'                                         0061C          .ADDRESS P.AFL
                      00 64 65 74 6E 75 6F 4D                        00620 P.AFN:   .ASCII  \Mounted\<0>
                   010E0007                                          00628 P.AFM:   .LONG   17694727
                   00000000'                                         0062C          .ADDRESS P.AFN
                      00 00 65 6E 69 6C 6E 4F                        00630 P.AFP:   .ASCII  \Online\<0><0>
                   010E0006                                          00638 P.AFO:   .LONG   17694726
                   00000000'                                         0063C          .ADDRESS P.AFP
                      00 65 6E 69 6C 66 66 4F                        00640 P.AFR:   .ASCII  \Offline\<0>
                   010E0007                                          00648 P.AFQ:   .LONG   17694727
                   00000000'                                         0064C          .ADDRESS P.AFR
                      64 65 6C 6F 6F 70 73 20                        00650 P.AFT:   .ASCII  \ spooled\
                   010E0008                                          00658 P.AFS:   .LONG   17694728
                   00000000'                                         0065C          .ADDRESS P.AFT
                      64 65 74 6E 75 6F 6D 20                        00660 P.AFV:   .ASCII  \ mounted\
                   010E0008                                          00668 P.AFU:   .LONG   17694728
                   00000000'                                         0066C          .ADDRESS P.AFV
                00 00 00 74 6E 75 6F 6D 73 69 64 20                  00670 P.AFX:   .ASCII  \ dismount\<0><0><0>
                   010E0009                                          0067C P.AFW:   .LONG   17694729
                   00000000'                                         00680          .ADDRESS P.AFX
                      00 00 63 6F 6C 6C 61 20                        00684 P.AFZ:   .ASCII  \ alloc\<0><0>
                   010E0006                                          0068C P.AFY:   .LONG   17694726
                   00000000'                                         00690          .ADDRESS P.AFZ
```

```
                                                    00694 P.AGB:   .ASCII  \ foreign\
                                                    0069C P.AGA:   .LONG   17694728
                                                    006A0          .ADDRESS P.AGB
                                                    006A4 P.AGD:   .ASCII  \ wrtlck\<0>
                                                    006AC P.AGC:   .LONG   17694727
                                                    006B0          .ADDRESS P.AGD
                                                    006B4 P.AGF:   .ASCII  \ del\
                                                    006B8 P.AGE:   .LONG   17694724
                                                    006BC          .ADDRESS P.AGF
                                                    006C0 P.AGH:   .ASCII  \ cls\
                                                    006C4 P.AGG:   .LONG   17694724
                                                    006C8          .ADDRESS P.AGH
                                                    006CC P.AGJ:   .ASCII  \ slv\
                                                    006D0 P.AGI:   .LONG   17694724
                                                    006D4          .ADDRESS P.AGJ
                                                    006D8 P.AGL:   .ASCII  \ knw\
                                                    006DC P.AGK:   .LONG   17694724
                                                    006E0          .ADDRESS P.AGL
                                                    006E4 P.AGN:   .ASCII  \ tf\<0>
                                                    006E8 P.AGM:   .LONG   17694723
                                                    006EC          .ADDRESS P.AGN
                                                    006F0 P.AGP:   .ASCII  \ pf\<0>
                                                    006F4 P.AGO:   .LONG   17694723
                                                    006F8          .ADDRESS P.AGP
                                                    006FC P.AGR:   .ASCII  \ pj\<0>
                                                    00700 P.AGQ:   .LONG   17694723
                                                    00704          .ADDRESS P.AGR
                                                    00708 P.AGT:   .ASCII  \ tj\<0>
                                                    0070C P.AGS:   .LONG   17694723
                                                    00710          .ADDRESS P.AGT
53 41 36 31 21 20 3E 21 46 41 21 3C 33 32 21        00714 P.AGV:   .ASCII  \!23<!AF!> !16AS !5UW  !12AF !9UL!6UW!4UW\
39 21 20 46 41 32 31 21 20 20 57 55 35 21 20        00723
                  57 55 34 21 57 55 36 21 4C 55     00732
                                                    0073C P.AGU:   .LONG   17694760
                                                    00740          .ADDRESS P.AGV
53 41 36 31 21 20 3E 21 46 41 21 3C 33 32 21        00744 P.AGX:   .ASCII  \!23<!AF!> !16AS!+ !18AF !4UL      !10UL-
20 20 4C 55 34 21 20 46 41 38 31 21 20 2B 21        00753                  \<0>
                     00 4C 55 30 31 21 20 20 20     00762
                                       00           0076B          .ASCII  <0>
                                                    0076C P.AGW:   .LONG   17694758
                                                    00770          .ADDRESS P.AGX
53 41 36 31 21 20 3E 21 46 41 21 3C 33 32 21        00774 P.AGZ:   .ASCII  \!23<!AF!> !16AS\<0>
                                       00           00783
                                                    00784 P.AGY:   .LONG   17694735
                                                    00788          .ADDRESS P.AGZ
53 41 36 31 21 20 3E 21 46 41 21 3C 33 32 21        0078C P.AHB:   .ASCII  \!23<!AF!> !16AS !5UW     !6AF     !9UL!6U\
20 46 41 36 21 20 20 20 20 20 57 55 35 21 20        0079B
                  55 36 21 4C 55 39 21 20 20 20     007AA
                        00 00 00 57 55 34 21 57     007B4          .ASCII  \W!4UW\<0><0><0>
                                                    007BC P.AHA:   .LONG   17694765
                                                    007C0          .ADDRESS P.AHB
53 41 36 31 21 20 3E 21 46 41 21 3C 33 32 21        007C4 P.AHD:   .ASCII  \!23<!AF!> !16AS !5UW\
                                    57 55 35 21 20  007D3
                                                    007D8 P.AHC:   .LONG   17694740
                                                    007DC          .ADDRESS P.AHD
00 00 53 41 36 31 21 20 3E 21 20 3C 33 32 21        007E0 P.AHF:   .ASCII  \!23< !> !16AS\<0><0><0>
                                       00           007EF
```

```
                                010E000D  (07F0 P.AHE:  .LONG   17694733                                          ;
                                00000000' (07F4         .ADDRESS P.AHF                                          ;


                                                        .PSECT  $CODE$,NOWRT,2

                                        0FFC 00000      .ENTRY  DISPLAY_BRIEF, Save R2,R3,R4,R5,R6,R7,R8,-      ; 1162
                                                                R9,R10,R11                                      ;
                                    5B  FCA4   CF  9E 00002     MOVAB   ADD_TO_LIST_RTN, R11                    ;
                                    5E  FAE4   CE  9E 00007     MOVAB   -1308(SP), SP                           ;
                C0  AD      0000'   CF  0040   8F  28 0000C     MOVC3   #64, P.AEL, DESC_VECTOR                 ; 1186
                                    58    04   AC  D0 00015     MOVL    SCRATCH, R8                             ; 1197
                                    0C    74   A8  E9 00019     BLBC    116(R8), 1$                             ;
                                          44   AE  9F 0001D     PUSHAB  CLUBUF                                  ; 1199
                                               58  DD 00020     PUSHL   R8                                      ;
                        00000000G   00         02  FB 00022     CALLS   #2, SCAN_CLUSTER_LOCKS                  ;
                                    52    08   AC  D0 00029 1$:  MOVL    FLAGS, R2                               ; 1206
                                               52  DD 0002D     PUSHL   R2                                      ;
                                               58  DD 0002F     PUSHL   R8                                      ;
                            00AD    CB         02  FB 00031     CALLS   #2, CHECK_MNT_ALL                       ;
                                    01         50  E8 00036     BLBS    R0, 2$                                  ;
                                               04  00039     RET                                             ;
                        53    01    A2         03  E1 0003A 2$:  BBC     #3, 1(R2), 7$                           ; 1212
                                    01    78   A8  91 0003F     CMPB    120(R8), #1                             ; 1215
                                               06  13 00043     BEQL    3$                                      ;
                                    02    78   A8  91 00045     CMPB    120(R8), #2                             ; 1216
                                               10  12 00049     BNEQ    4$                                      ;
                                          08   AE  9F 0004B 3$:  PUSHAB  ARGLIST                                ; 1221
                                    0000'   CF  9F 0004E        PUSHAB  P.AEO                                   ; 1219
                                          10   AE  9F 00052     PUSHAB  ARGLIST                                 ;
                                    0000'   CF  9F 00055        PUSHAB  P.AEM                                   ; 1217
                                               25  11 00059     BRB     6$                                      ;
                            A1      8F    78   A8  91 0005B 4$:  CMPB    120(R8), #161                           ; 1222
                                               10  12 00060     BNEQ    5$                                      ;
                                          08   AE  9F 00062     PUSHAB  ARGLIST                                 ; 1227
                                    0000'   CF  9F 00065        PUSHAB  P.AES                                   ; 1225
                                          10   AE  9F 00069     PUSHAB  ARGLIST                                 ;
                                    0000'   CF  9F 0006C        PUSHAB  P.AEQ                                   ; 1223
                                               0E  11 00070     BRB     6$                                      ;
                                          08   AE  9F 00072 5$:  PUSHAB  ARGLIST                                ; 1232
                                    0000'   CF  9F 00075        PUSHAB  P.AEW                                   ; 1230
                                          10   AE  9F 00079     PUSHAB  ARGLIST                                 ;
                                    0000'   CF  9F 0007C        PUSHAB  P.AEU                                   ; 1228
                                          18   AE  9F 00080 6$:  PUSHAB  ARGLIST                                ;
                                    0000'   CF  9F 00083        PUSHAB  ASCID_NULL                              ;
                    00000000G   00         06  FB 00087     CALLS   #6, SHOW$WRITE_LINE                     ;
                            01  A2         08  8A 0008E     BICB2   #8, 1(R2)                               ; 1233
                            08  AE    06   A8  9A 00092 7$:  MOVZBL  6(R8), ARGLIST                          ; 1240
                                          08   AE  D7 00097     DECL    ARGLIST                                 ;
                            0C  AE    09   A8  9E 0009A     MOVAB   9(R8), ARGLIST+4                        ; 1241
                                5A    70   A8  9E 0009F     MOVAB   112(R8), R10                            ; 1249
                    63          6A         0E  E1 000A3     BBC     #14, (R10), 9$                          ;
                                          54   A8  D5 000A7     TSTL    84(R8)                                  ; 1251
                                               5E  13 000AA     BEQL    9$                                      ;
                                59    1C   A8  9A 000AC     MOVZBL  28(R8), HLEN                            ; 1258
                    50          59    08   AE  C1 000B0     ADDL3   ARGLIST, HLEN, R0                       ; 1259
```

```
                       56    02  A0  9E 000B5        MOVAB   2(R0), TLEN                              ;
              50       16        56  C3 000B9        SUBL3   TLEN, #22, R0                            ; 1260
                                 03  14 000BD        BGTR    8$                                       ;
                       50        01  D0 000BF        MOVL    #1, R0                                   ;
              57       50    08  AE  C1 000C2 8$:    ADDL3   ARGLIST, R0, BLEN                        ;
                       56    02 A947 9E 000C7        MOVAB   2(HLEN)[BLEN], TLEN                      ; 1261
     57       20  0C   BE    08  AE  2C 000CC        MOVC5   ARGLIST, @ARGLIST+4, #32, BLEN, NAMBUF   ; 1262
                           FF28  CD     000D3                                                         ;
              FF28 CD47          28  90 000D6        MOVB    #40, NAMBUF[BLEN]                        ; 1264
     FF29 CD47      1D   A8      59  28 000DC        MOVC3   HLEN, 29(R8), NAMBUF+1[BLEN]             ; 1266
              FF27 CD46          29  90 000E4        MOVB    #41, NAMBUF-1[TLEN]                      ; 1267
                    08   AE      56  D0 000EA        MOVL    TLEN, ARGLIST                            ; 1268
                    0C   AE FF28 CD  9E 000EE        MOVAB   NAMBUF, ARGLIST+4                        ; 1269
                         17      56  D1 000F4        CMPL    TLEN, #23                                ; 1270
                                 11  15 000F7        BLEQ    9$                                       ;
                             08  AE  9F 000F9        PUSHAB  ARGLIST                                  ; 1273
                          0000'  CF  9F 000FC        PUSHAB  P.AEY                                    ;
              00000000G  00      02  FB 00100        CALLS   #2, SHOW$WRITE_LINE                      ;
                             08  AE  D4 00107        CLRL    ARGLIST                                  ; 1274
                    04   AE  C0  AD  9E 0010A 9$:    MOVAB   DESC_VECTOR, PTR                         ; 1281
                    C4   AD FF68 CD  9E 0010F        MOVAB   BUFFER, DESC_VECTOR+4                    ; 1282
                          0000'  CF  D4 00115        CLRL    LEADING_BLANKS                           ; 1283
                    10   AE  C0  AD  9E 00119        MOVAB   DESC_VECTOR, ARGLIST+8                   ; 1284
     53       6A                 01  13  EF 0011E    EXTZV   #19, #1, (R10), SHOW_MNT                 ; 1285
                         52 0088 C8  9E 00123        MOVAB   136(R8), R2                              ; 1289
              72         62      04  E1 00128        BBC     #4, (R2), 18$                            ;
              08    04   A8      01  E0 0012C        BBS     #1, 4(R8), 10$                           ; 1295
                                 10  DD 00131        PUSHL   #16                                      ; 1296
                          0000'  CF  9F 00133        PUSHAB  P.AFA                                    ;
                                 6B  11 00137        BRB     19$                                      ;
              08    74   A8      01  E1 00139 10$:   BBC     #1, 116(R8), 11$                         ; 1300
                                 10  DD 0013E        PUSHL   #16                                      ; 1301
                          0000'  CF  9F 00140        PUSHAB  P.AFC                                    ;
                                 5E  11 00144        BRB     19$                                      ;
              08         6A      12  E0 00146 11$:   BBS     #18, (R10), 12$                          ; 1305
                                 10  DD 0014A        PUSHL   #16                                      ; 1306
                          0000'  CF  9F 0014C        PUSHAB  P.AFE                                    ;
                                 52  11 00150        BRB     19$                                      ;
                                 62  B5 00152 12$:   TSTW    (R2)                                     ; 1310
                                 08  18 00154        BGEQ    13$                                      ;
                                 10  DD 00156        PUSHL   #16                                      ; 1311
                          0000'  CF  9F 00158        PUSHAB  P.AFG                                    ;
                                 46  11 0015C        BRB     19$                                      ;
              08         62      0E  E1 0015E 13$:   BBC     #14, (R2), 14$                           ; 1315
                                 10  DD 00162        PUSHL   #16                                      ; 1316
                          0000'  CF  9F 00164        PUSHAB  P.AFI                                    ;
                                 3A  11 00168        BRB     19$                                      ;
              1A         62      0B  E0 0016A 14$:   BBS     #11, (R2), 16$                           ; 1321
              24         6A      13  E1 0016E        BBC     #19, (R10), 17$                          ; 1322
                         01 009D C8  91 00172        CMPB    157(R8), #1                              ; 1323
                                 07  13 00177        BEQL    15$                                      ;
                         02 009D C8  91 00179        CMPB    157(R8), #2                              ; 1325
                                 08  12 0017E        BNEQ    16$                                      ;
                                 10  DD 00180 15$:   PUSHL   #16                                      ; 1326
                          0000'  CF  9F 00182        PUSHAB  P.AFK                                    ;
                                 1C  11 00186        BRB     19$                                      ;
              0A         6A      13  E1 00188 16$:   BBC     #19, (R10), 17$                          ; 1330
```

```
                        53  D4 0018C          CLRL    SHOW_MNT                        ; 1333
                        10  DD 0018E          PUSHL   #16                             ; 1334
                0000'   CF  9F 00190          PUSHAB  P.AFM
                        02  11 00194          BRB     19$
                        10  DD 00196 17$:     PUSHL   #16                             ; 1339
                0000'   CF  9F 00198          PUSHAB  P.AFO
                        06  11 0019C          BRB     19$
                        10  DD 0019E 18$:     PUSHL   #16                             ; 1345
                0000'   CF  9F 001A0          PUSHAB  P.AFQ
                  0C    AE  9F 001A4 19$:     PUSHAB  PTR
            6B          03  FB 001A7          CALLS   #3, ADD_TO_LIST_RTN
    0C      6A          06  E1 001AA          BBC     #6, (R10), 20$                  ; 1349
                        10  DD 001AE          PUSHL   #16                             ; 1350
                0000'   CF  9F 001B0          PUSHAB  P.AFS
                  0C    AE  9F 001B4          PUSHAB  PTR
            6B          03  FB 001B7          CALLS   #3, ADD_TO_LIST_RTN
    0C                  53  E9 001BA 20$:     BLBC    SHOW_MNT, 21$                   ; 1351
                        10  DD 001BD          PUSHL   #16                             ; 1352
                0000'   CF  9F 001BF          PUSHAB  P.AFU
                  0C    AE  9F 001C3          PUSHAB  PTR
            6B          03  FB 001C6          CALLS   #3, ADD_TO_LIST_RTN
    0C      6A          15  E1 001C9 21$:     BBC     #21, (R10), 22$                 ; 1353
                        10  DD 001CD          PUSHL   #16                             ; 1354
                0000'   CF  9F 001CF          PUSHAB  P.AFW
                  0C    AE  9F 001D3          PUSHAB  PTR
            6B          03  FB 001D6          CALLS   #3, ADD_TO_LIST_RTN
    0C      6A          17  E1 001D9 22$:     BBC     #23, (R10), 23$                 ; 1355
                        10  DD 001DD          PUSHL   #16                             ; 1356
                0000'   CF  9F 001DF          PUSHAB  P.AFY
                  0C    AE  9F 001E3          PUSHAB  PTR
            6B          03  FB 001E6          CALLS   #3, ADD_TO_LIST_RTN
    0C        03        AA  E9 001E9 23$:     BLBC    3(R10), 24$                     ; 1357
                        10  DD 001ED          PUSHL   #16                             ; 1358
                0000'   CF  9F 001EF          PUSHAB  P.AGA
                  0C    AE  9F 001F3          PUSHAB  PTR
            6B          03  FB 001F6          CALLS   #3, ADD_TO_LIST_RTN
    0C      6A          19  E1 001F9 24$:     BBC     #25, (R10), 25$                 ; 1359
                        10  DD 001FD          PUSHL   #16                             ; 1360
                0000'   CF  9F 001FF          PUSHAB  P.AGC
                  0C    AE  9F 00203          PUSHAB  PTR
            6B          03  FB 00206          CALLS   #3, ADD_TO_LIST_RTN
                        53  D4 00209 25$:     CLRL    R3                              ; 1361
        A1  8F      78  A8  91 0020B          CMPB    120(R8), #161
                        75  12 00210          BNEQ    34$
                        53  D6 00212          INCL    R3
    0C      62          0A  E1 00214          BBC     #10, (R2), 26$                  ; 1364
                        10  DD 00218          PUSHL   #16                             ; 1365
                0000'   CF  9F 0021A          PUSHAB  P.AGE
                  0C    AE  9F 0021E          PUSHAB  PTR
            6B          03  FB 00221          CALLS   #3, ADD_TO_LIST_RTN
            52  0090    C8  9E 00224 26$:     MOVAB   144(R8), R2                     ; 1366
    0C      62          06  E1 00229          BBC     #6, (R2), 27$
                        10  DD 0022D          PUSHL   #16                             ; 1367
                0000'   CF  9F 0022F          PUSHAB  P.AGG
                  0C    AE  9F 00233          PUSHAB  PTR
            6B          03  FB 00236          CALLS   #3, ADD_TO_LIST_RTN
                        62  95 00239 27$:     TSTB    (R2)                            ; 1368
```

```
                                    0C  18 0023B          BGEQ    28$                                    ;
                                    10  DD 0023D          PUSHL   #16                                    ; 1369
                            0000'   CF  9F 0023F          PUSHAB  P.AGI                                  ;
                              0C    AE  9F 00243          PUSHAB  PTR                                    ;
                          6B        03  FB 00246          CALLS   #3, ADD_TO_LIST_RTN                    ;
                    08    62        05  E1 00249 28$:     BBC     #5, (R2), 29$                          ; 1370
                                    10  DD 0024D          PUSHL   #16                                    ; 1371
                            0000'   CF  9F 0024F          PUSHAB  P.AGK                                  ;
                                    14  11 00253          BRB     31$                                    ;
                    08  00E0  C8        02  E1 00255 29$:     BBC     #2, 224(R8), 30$                       ; 1374
                                    10  DD 0025B          PUSHL   #16                                    ; 1375
                            0000'   CF  9F 0025D          PUSHAB  P.AGM                                  ;
                                    06  11 00261          BRB     31$                                    ;
                                    10  DD 00263 30$:     PUSHL   #16                                    ; 1376
                            0000'   CF  9F 00265          PUSHAB  P.AGO                                  ;
                              0C    AE  9F 00269 31$:     PUSHAB  PTR                                    ;
                          6B        03  FB 0026C          CALLS   #3, ADD_TO_LIST_RTN                    ;
                    08    62        04  E1 0026F          BBC     #4, (R2), 32$                          ; 1378
                                    10  DD 00273          PUSHL   #16                                    ; 1379
                            0000'   CF  9F 00275          PUSHAB  P.AGQ                                  ;
                                    06  11 00279          BRB     33$                                    ;
                                    10  DD 0027B 32$:     PUSHL   #16                                    ; 1380
                            0000'   CF  9F 0027D          PUSHAB  P.AGS                                  ;
                              0C    AE  9F 00281 33$:     PUSHAB  PTR                                    ;
                          6B        03  FB 00284          CALLS   #3, ADD_TO_LIST_RTN                    ;
                       14 AE  0092    C8  3C 00287 34$:     MOVZWL  146(R8), ARGLIST+12                    ; 1386
                                    51  D4 0028D          CLRL    R1                                     ; 1391
                          01    78    A8  91 0028F          CMPB    120(R8), #1                            ;
                                    04  12 00293          BNEQ    35$                                    ;
                                    51  D6 00295          INCL    R1                                     ;
                                    06  11 00297          BRB     36$                                    ;
                          02    78    A8  91 00299 35$:     CMPB    120(R8), #2                            ; 1392
                                    04  12 0029D          BNEQ    37$                                    ;
                    06    6A          13  E0 0029F 36$:     BBS     #19, (R10), 38$                        ; 1393
                          03          53  E8 002A3 37$:     BLBS    R3, 38$                                ; 1394
                                0089  31  002A6          BRW     50$                                    ;
                       1C AE  00B8    C8  9E 002A9 38$:     MOVAB   184(R8), ARGLIST+20                    ; 1401
                          06          51  E8 002AF          BLBS    R1, 39$                                ; 1402
                          02    78    A8  91 002B2          CMPB    120(R8), #2                            ; 1403
                                    14  12 002B6          BNEQ    40$                                    ;
                       20 AE  00D2    C8  D0 002B8 39$:     MOVL    210(R8), ARGLIST+24                    ; 1406
                       24 AE  009B    C8  3C 002BE          MOVZWL  155(R8), ARGLIST+28                    ; 1407
                       28 AE  00CC    C8  3C 002C4          MOVZWL  204(R8), ARGLIST+32                    ; 1408
                                    13  11 002CA          BRB     41$                                    ; 1402
                          50    7A    A8  3C 002CC 40$:     MOVZWL  122(R8), R0                            ; 1412
                 20 AE    50 00000200    8F  C7 002D0          DIVL3   #512, R0, ARGLIST+24                   ;
                       24 AE  00E8    C8  D0 002D9          MOVL    232(R8), ARGLIST+28                    ; 1413
                          50    0099    C8  9E 002DF 41$:     MOVAB   153(R8), R0                            ; 1423
                          1A          51  E9 002E4          BLBC    R1, 45$                                ;
                          05          60  E8 002E7          BLBS    (R0), 42$                              ;
                                04    A8  95 002EA          TSTB    4(R8)                                  ;
                                    06  18 002ED          BGEQ    43$                                    ;
                       18 AE          0C  D0 002EF 42$:     MOVL    #12, ARGLIST+16                        ; 1424
                                    03  11 002F3          BRB     44$                                    ;
                                18  AE  D4 002F5 43$:     CLRL    ARGLIST+16                             ; 1425
                                08  AE  9F 002F8 44$:     PUSHAB  ARGLIST                                ; 1426
                            0000'   CF  9F 002FB          PUSHAB  P.AGU                                  ;
```

```
                                38  11 002FF            BRB     51$                                     ;
                             19     53  E9 00301 45$:    BLBC    R3, 47$                                 ; 1428
                             0D     60  E9 00304        BLBC    (R0), 46$                               ; 1431
                          18 AE     12  D0 00307        MOVL    #18, ARGLIST+16                         ; 1434
                       08 AE        9F 0030B            PUSHAB  ARGLIST                                 ; 1435
                    0000' CF        9F 0030E            PUSHAB  P.AGW                                   ;
                                25  11 00312            BRB     51$                                     ;
                       08 AE        9F 00314 46$:       PUSHAB  ARGLIST                                 ; 1437
                    0000' CF        9F 00317            PUSHAB  P.AGY                                   ;
                                1C  11 0031B            BRB     51$                                     ;
                             06     60  E9 0031D 47$:    BLBC    (R0), 48$                               ; 1441
                          18 AE     06  D0 00320        MOVL    #6, ARGLIST+16                          ; 1442
                                03  11 00324            BRB     49$                                     ;
                       18 AE        D4 00326 48$:       CLRL    ARGLIST+16                              ; 1443
                       08 AE        9F 00329 49$:       PUSHAB  ARGLIST                                 ; 1444
                    0000' CF        9F 0032C            PUSHAB  P.AHA                                   ;
                                07  11 00330            BRB     51$                                     ;
                       08 AE        9F 00332 50$:       PUSHAB  ARGLIST                                 ; 1448
                    0000' CF        9F 00335            PUSHAB  P.AHC                                   ;
                00000000G   00      02  FB 00339 51$:    CALLS   #2, SHOW$WRITE_LINE                     ;
                            52      01  D0 00340        MOVL    #1, I                                   ; 1453
                 50         52      01  78 00343 52$:    ASHL    #1, I, R0                               ; 1455
                         C0 AD40     D5 00347            TSTL    DESC_VECTOR[R0]                         ; 1456
                                16  13 0034B            BEQL    53$                                     ;
                            6E   C0 AD40     DE 0034D    MOVAL   DESC_VECTOR[R0], (SP)                   ; 1457
                                 5E  DD 00352            PUSHL   SP                                      ;
                    0000' CF        9F 00354            PUSHAB  P.AHE                                   ;
                00000000G   00      02  FB 00358        CALLS   #2, SHOW$WRITE_LINE                     ;
                 E0         52      07  F3 0035F        AOBLEQ  #7, I, 52$                              ; 1456
                                        04 00363 53$:    RET                                             ; 1462

; Routine Size:  868 bytes,    Routine Base:  $CODE$ + 0356
```

```
;1086     1463  1 GLOBAL ROUTINE display_disk (scratch, flags) : NOVALUE =
;1087     1464  2 BEGIN
;1088     1465  2
;1089     1466  2 !---
;1090     1467  2 !
;1091     1468  2 ! This is the display routine for disks
;1092     1469  2 !
;1093     1470  2 ! Inputs
;1094     1471  2 !       SCRATCH - contains data about the disk
;1095     1472  2 !       FLAGS   - contain info on what to print
;1096     1473  2 !
;1097     1474  2 ! Outputs
;1098     1475  2 !       None
;1099     1476  2 !
;1100     1477  2 !---
;1101     1478  2
;1102     1479  2 BUILTIN
;1103     1480  2     emul,ediv,cmpm;
;1104     1481  2
;1105     1482  2 MAP
;1106     1483  2     scratch : REF $BBLOCK,
;1107     1484  2     flags : REF $BBLOCK;
;1108     1485  2
;1109     1486  2 LOCAL
;1110     1487  2     desc_vector : VECTOR[9]
;1111     1488  2                   INITIAL(REP 9 OF (0)),
;1112     1489  2     ptr : REF VECTOR,
;1113     1490  2     fao_ctr : REF $BBLOCK,
;1114     1491  2     buffer : VECTOR[160],       !8 80 byte lines
;1115     1492  2     prot_addr : VECTOR[4],
;1116     1493  2     prot_desc : VECTOR[2],
;1117     1494  2     prot_buff : VECTOR[20],
;1118     1495  2     clubuf : VECTOR[256, LONG],         ! Buffer for lock information
;1119     1496  2     arglist : VECTOR[20];
;1120     1497  2
;1121     1498  2 !
;1122     1499  2 ! First thing, fix up the scratch area with any available clusterwide info
;1123     1500  2 !
;1124     1501  2 IF .$BBLOCK[scratch[d_l_devchar2], dev$v_clu]   ! If available clusterwide
;1125     1502  2 THEN
;1126     1503  2     scan_cluster_locks (.scratch, clubuf);
;1127     1504  2
;1128     1505  2 !
;1129     1506  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;1130     1507  2 ! then check to see if the device is mounted/allocated.  If not, then
;1131     1508  2 ! don't display information on the device.
;1132     1509  2 !
;1133     1510  2 IF NOT check_mnt_all(.scratch, .flags)
;1134     1511  2 THEN RETURN;
;1135     1512  2
;1136     1513  2 arglist[0] = cstring('Disk ');
;1137     1514  2
;1138     1515  2 !
;1139     1516  2 ! Determine which type of disk it is.  If not in the list, then
;1140     1517  2 ! simply say that it is unknown.
;1141     1518  2 !
;1142     1519  2 arglist[6] = unknown;                    ! Assume unknown
```

```
;1143     1520  2 INCR i FROM 0 TO disk_number - 1 DO     ! Loop thru known disk types
;1144     1521  3     (IF .scratch[d_b_devtype] EQLU .disk_type[.i]
;1145     1522  3     THEN (arglist[6] = .disk_label[.i]; EXITLOOP));
;1146     1523  2
;1147     1524  2 common_display(arglist, .scratch);              ! Display stuff common to all devices
;1148     1525  2
;1149     1526  2 !
;1150     1527  2 ! Display some geometry items
;1151     1528  2 !
;1152     1529  2 IF (arglist[0] = .scratch[d_l_maxblock]) NEQ 0
;1153     1530  2 THEN
;1154     1531  3     BEGIN
;1155     1532  3     arglist[1] = .scratch[d_b_sectors];
;1156     1533  3     arglist[2] = .scratch[d_w_cylinders];
;1157     1534  3     arglist[3] = .scratch[d_b_tracks];
;1158     1535  3     show$write_line(%ASCID '    Total blocks!20UL    Sectors per track!22UW',
;1159     1536  3                     arglist[0],
;1160     1537  3                     %ASCID '    Total cylinders!17UW    Tracks per cylinder!20UW',
;1161     1538  3                     arglist[2]);
;1162     1539  2     END;
;1163     1540  2
;1164     1541  2 !
;1165     1542  2 ! If remote or dual-pathed, then print some more info
;1166     1543  2 !
;1167     1544  2 IF .scratch[d_v_remote_device] OR .$BBLOCK[scratch[d_l_devchar2], dev$v_2p]
;1168     1545  2 THEN
;1169     1546  3     BEGIN
;1170     1547  3     BIND
;1171     1548  3         t_yes = UPLIT BYTE ('yes'),
;1172     1549  3         t_no  = UPLIT BYTE (' no');
;1173     1550  3     arglist[0] = 15 - .scratch[d_t_host_name];
;1174     1551  3     arglist[1] = .scratch[d_t_host_name];
;1175     1552  3     arglist[2] = scratch[d_t_host_name] + 1;
;1176     1553  3     arglist[3] = 4;
;1177     1554  3     arglist[4] = scratch[d_l_host_type];
;1178     1555  3     arglist[5] = 3;
;1179     1556  4     arglist[6] = (IF .scratch[d_v_host_avail]
;1180     1557  4                   THEN t_yes
;1181     1558  3                   ELSE t_no);
;1182     1559  3     show$write_line(%ASCID '    Host name      !#< !>"!AF"    Host type, available          !AF, !AF',
;1183     1560  3                                                 arglist[0]);
;1184     1561  3     IF .$BBLOCK[scratch[d_l_devchar2], dev$v_2p]
;1185     1562  3     THEN
;1186     1563  4         BEGIN
;1187     1564  4         arglist[0] = 8 - .scratch[d_t_host2_name];
;1188     1565  4         arglist[1] = .scratch[d_t_host2_name];
;1189     1566  4         arglist[2] = scratch[d_t_host2_name] + 1;
;1190     1567  4         arglist[3] = 4;
;1191     1568  4         arglist[4] = scratch[d_l_host2_type];
;1192     1569  4         arglist[5] = 3;
;1193     1570  5         arglist[6] = (IF .scratch[d_v_host2_avail]
;1194     1571  5                       THEN t_yes
;1195     1572  4                       ELSE t_no);
;1196     1573  4         show$write_line(%ASCID '    Alternate host name    !#< !>"!AF"    Alternate host type, avail    !AF,
;1197     1574  4                                                    arglist[0]);
;1198     1575  3         END;
;1199     1576  2     END;
```

```
;  1200          1577  2
;  1201          1578  2     !
;  1202          1579  2     ! If an allocation class is present, display it
;  1203          1580  2     !
;  1204          1581  2     IF .scratch[d_l_allocls] NEQ 0
;  1205          1582  2     THEN
;  1206          1583  2         show$write_line(%ASCID '    Allocation class!16UL', scratch[d_l_allocls]);
;  1207          1584  2
;  1208          1585  2     !
;  1209          1586  2     ! Now for the ACP-dependent things
;  1210          1587  2     !
;  1211          1588  2     IF .scratch[d_b_cont]
;  1212          1589  2     THEN
;  1213          1590  2         BEGIN
;  1214          1591  3         arglist[1] = vcb$s_volname;                          ! Start with full volume name
;  1215          1592  3         DECR idx FROM vcb$s_volname-1 TO 0                   ! Reduce length
;  1216          1593  4         DO (IF .(scratch[d_t_volnam]+.idx)<0,8,0> EQL %C' ' ! by the number of
;  1217          1594  4             THEN arglist[1] = .arglist[1]-1                  ! trailing spaces
;  1218          1595  4             ELSE exitloop);
;  1219          1596  3         arglist[0] = 6+vcb$s_volname-.arglist[1];            ! Adjust field width
;  1220          1597  3         arglist[2] = scratch[d_t_volnam];                    ! Address of volume name
;  1221          1598  3         arglist[3] = .scratch[d_w_rvn];                      ! Relative volume number
;  1222          1599  3         arglist[4] = .scratch[d_w_cluster];                  ! Cluster factor
;  1223          1600  3         arglist[5] = .scratch[d_w_trans];                    ! Transaction count
;  1224          1601  3         arglist[6] = .scratch[d_l_free];                     ! Free blocks on volume
;  1225          1602  3         arglist[7] = .scratch[d_l_maxfiles];                 ! Max number of files
;  1226          1603  3         arglist[8] = .scratch[d_w_extend];                   ! Default extend quantity
;  1227          1604  3         arglist[9] = .scratch[d_w_mcount];                   ! Count of sharers
;  1228          1605  3
;  1229          1606  3         !
;  1230          1607  3         ! Is it mounted /GROUP or /SYSTEM ?
;  1231          1608  3         !
;  1232          1609  3         IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_group),1,0]
;  1233          1610  3         THEN arglist[11] = cstring('Group')
;  1234          1611  3         ELSE IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_system),1,0]
;  1235          1612  3         THEN arglist[11] = cstring('System')
;  1236          1613  3         ELSE arglist[11] = cstring('Process');
;  1237          1614  3         arglist[10] = 20 - .(.arglist[11])<0,8>;
;  1238          1615  3         IF .scratch[d_v_cachename]
;  1239          1616  3         THEN
;  1240          1617  4             BEGIN
;  1241          1618  4             fao_ctr = %ASCID '    Mount status!#< !>!AC    Cache name!#< !>''!AF''';
;  1242          1619  4             arglist[12] = 27 - .scratch[d_t_acpnam];
;  1243          1620  4             END
;  1244          1621  3         ELSE
;  1245          1622  4             BEGIN
;  1246          1623  4             fao_ctr = %ASCID '    Mount status!#< !>!AC    ACP process name!#< !>''!AF''';
;  1247          1624  4             arglist[12] = 21 - .scratch[d_t_acpnam];
;  1248          1625  4             END;
;  1249          1626  3         arglist[13] = .scratch[d_t_acpnam];          ! Process name of the ACP, or XQP cache name
;  1250          1627  3         arglist[14] = scratch[d_t_acpnam] + 1;
;  1251          1628  3
;  1252          1629  3         show$write_line(ascid_null, arglist[0],
;  1253          1630  3                         %ASCID '    Volume label!#< !>''!AF''    Relative volume number!17UW',
;  1254          1631  3                         arglist[0],
;  1255          1632  3                         %ASCID '    Cluster size!20UW    Transaction count!22UW',
;  1256          1633  3                         arglist[4],
```

```
;1257     1634  3                    %ASCID '    Free blocks!21UL    Maximum files allowed!18UL',
;1258     1635                       arglist[6],
;1259     1636                       %ASCID '    Extend quantity!17UW    Mount count!28UW',
;1260     1637                       arglist[8],
;1261     1638                       .fao_ctr,
;1262     1639                       arglist[10]);
;1263     1640
;1264     1641          !
;1265     1642          ! Now for special bits.
;1266     1643          !
;1267     1644          ptr = desc_vector[0];
;1268     1645          desc_vector[1] = buffer;
;1269     1646          leading_blanks = 6;
;1270     1647          add_to_list(ptr, '  Volume status: ', 80);  !Note - string length is checked later
;1271     1648          IF .$BBLOCK[scratch[d_b_status2], 0,$BITPOSITION(vcb$v_mountver),1,0]
;1272     1649          THEN add_to_list(ptr, ' subject to mount verification,', 80);
;1273     1650          IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_homblkbad),1,0]
;1274     1651          THEN add_to_list(ptr, ' primary home block is bad,', 80);
;1275     1652          IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_idxhdrbad),1,0]
;1276     1653          THEN add_to_list(ptr, ' primary index file header is bad,', 80);
;1277     1654          IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_noalloc),1,0]
;1278     1655          THEN add_to_list(ptr, ' allocation inhibited because of error on bitmap,', 80);
;1279     1656          IF NOT .$BBLOCK[scratch[d_l_sts], 0,$BITPOSITION(ucb$v_unload),1,0]
;1280     1657          THEN add_to_list(ptr, ' do not unload on dismount,', 80);
;1281     1658          IF .scratch[d_b_aqbtype] EQL aqb$k_fl1v2            ! If ODS-2, find some more
;1282     1659          THEN
;1283     1660  4           BEGIN
;1284     1661  4           !
;1285     1662  4           ! Get ODS-2 specific information: caching, mount verification, and
;1286     1663  4           ! expiration dates, erasing, high-water
;1287     1664  4           !
;1288     1665  4           IF .$BBLOCK[scratch[d_b_status2], 0,$BITPOSITION(vcb$v_erase),1,0]
;1289     1666  4           THEN add_to_list(ptr, ' erase on delete,', 80);
;1290     1667  4           IF NOT .$BBLOCK[scratch[d_b_status2], 0,$BITPOSITION(vcb$v_nohighwater),1,0]
;1291     1668  4           THEN add_to_list(ptr, ' file high-water marking,', 80);
;1292     1669  4           IF .$BBLOCK[scratch[d_b_status2], 0,$BITPOSITION(vcb$v_nocache),1,0]
;1293     1670  4           THEN add_to_list(ptr, ' caching is disabled,', 80)
;1294     1671  4           ELSE
;1295     1672  5               BEGIN
;1296     1673  5               LOCAL
;1297     1674  5                   temp : VECTOR [2, LONG], dummy, limit;
;1298     1675  5               IF (arglist[0] = .scratch[d_w_extsize]) EQL 0       ! Extent cache size
;1299     1676  5               THEN
;1300     1677  5                   arglist[1] = arglist[3] = 0                     ! No extent cache, zero for all fields
;1301     1678  5               ELSE
;1302     1679  6                   BEGIN
;1303     1680  6                   arglist[1] = .scratch[d_w_extlimit];            ! Extend to longword (limit in percent_free/
;1304     1681  6                   EMUL (arglist[1], scratch[d_l_free], %REF(0), temp);
;1305     1682  6                   EDIV (%REF(1000), temp, arglist[1], dummy);     ! Store limit in arglist
;1306     1683  6                   arglist[3] = .scratch[d_l_exttotal];            ! Blocks currently in extent cache
;1307     1684  5                   END;
;1308     1685  5               arglist[2] = .scratch[d_w_fidsize];          ! File ID cache size
;1309     1686  5               arglist[4] = .scratch[d_w_quosize];          ! Quota cache size
;1310     1687  5               arglist[5] = .scratch[d_w_bfrcnt];           ! Cache buffer count
;1311     1688  5
;1312     1689  5               show$write_line(%ASCID '    Extent cache size!15UW    Maximum blocks in extent cache!9UL',
;1313     1690  5                               arglist[0],
```

```
;  1314          1691  5                          %ASCID '    File ID cache size!14UW    Blocks currently in extent cache!7UL',
;  1315          1692  5                          arglist[2],
;  1316          1693  5                          %ASCID '    Quota cache size!16UW    Maximum buffers in FCP cache!11UW',
;  1317          1694  5                          arglist[4]);
;  1318          1695  5                  IF .$BBLOCK[scratch[d_b_status2], 0,$BITPOSITION(vcb$v_writethru),1,0]
;  1319          1696  5                  THEN add_to_list(ptr, ' write-through caching enabled,', 80)
;  1320          1697  5                  ELSE add_to_list(ptr, ' write-back caching enabled,', 80);
;  1321          1698  4                  END;
;  1322          1699  4
;  1323          1700  4                  !
;  1324          1701  4                  ! If a retention min/max are specified, then output them.
;  1325          1702  4                  !
;  1326          1703  4                  IF CMPM(2, scratch[d_q_retainmin], UPLIT(0,0)) NEQ 0
;  1327          1704  4                  THEN
;  1328          1705  5                      BEGIN
;  1329          1706  5                      LOCAL
;  1330          1707  5                          min_tim : VECTOR[7,WORD],
;  1331          1708  5                          max_tim : VECTOR[7,WORD];
;  1332        P 1709  5                      $NUMTIM(TIMBUF = min_tim,
;  1333          1710  5                              TIMADR = scratch[d_q_retainmin]);
;  1334        P 1711  5                      $NUMTIM(TIMBUF = max_tim,
;  1335          1712  5                              TIMADR = scratch[d_q_retainmax]);
;  1336          1713  5                      arglist[0] = .min_tim[2];
;  1337          1714  5                      arglist[1] = .max_tim[2];
;  1338          1715  5                      show$write_line(%ASCID '   Min ret. period (days)!10UW    Max ret. period (days)!17UW',
;  1339          1716  5                              arglist[0]);
;  1340          1717  4                      END;
;  1341          1718  3                  END;
;  1342          1719  3
;  1343          1720  3              !
;  1344          1721  3              ! Separate the special and node information by a blank line
;  1345          1722  3              !
;  1346          1723  3              show$write_line(ascid_null, arglist[0]);
;  1347          1724  3
;  1348          1725  3              !
;  1349          1726  3              ! Output the special bits, if there are any.
;  1350          1727  3              !
;  1351          1728  3              IF .desc_vector[0] NEQ %CHARCOUNT('  Volume status: ')
;  1352          1729  3              THEN INCR i FROM 0 TO 3 DO
;  1353          1730  4                  BEGIN
;  1354          1731  4                  IF .desc_vector[2*(.i+1)] EQL 0          ! On the last string change the final
;  1355          1732  4                  THEN                                     !  comma to a period.
;  1356          1733  5                      BEGIN
;  1357          1734  5                      BIND
;  1358          1735  5                          str = .desc_vector[(2*.i)+1],
;  1359          1736  5                          dot = (str+.desc_vector[2*.i]-1) : BYTE;
;  1360          1737  5                      dot = '.';
;  1361          1738  4                      END;
;  1362          1739  4                  arglist[0] = desc_vector[2*.i];
;  1363          1740  4                  show$write_line(%ASCID '!AS', arglist);
;  1364          1741  4                  IF .desc_vector[2*(.i+1)] EQL 0
;  1365          1742  4                  THEN EXITLOOP;
;  1366          1743  3                  END;
;  1367          1744  2              END;
;  1368          1745  2
;  1369          1746  2 !
;  1370          1747  2 ! Let them know if it is accessed elsewhere
```

```
;  1371          1748  2 !
;  1372          1749  2 cluster_nodes (.scratch, clubuf);
;  1373          1750  2
;  1374          1751  2 !
;  1375          1752  2 ! Show any access control list which might be present
;  1376          1753  2 !
;  1377          1754  2 show_device_acl (.scratch);
;  1378          1755  2
;  1379          1756  2 RETURN;
;  1380          1757  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                    00000000#   007F8 P.AHG:  .LONG   0[9]                                                  ;
                                                20  6B  73  69   44  05    0081C P.AHH:  .ASCII  <5>\Disk \                                    ;
                                            6E  77  6F  6E  6B   6E  75  07   00822 P.AHI:  .ASCII  <7>\unknown\                               ;
                                                                             0082A         .BLKB   2                                            ;
6B  63  6F  6C  62  20  6C  61  74  6F  54  20  20   20  20   0082C P.AHK:  .ASCII  \    Total blocks!20UL    Sectors per tra\  ;
6F  74  63  65  53  20  20  20  20  4C  55  30  32   21  73   0083B                                                                        ;
                    61  72  74  20  72  65  70  20   73  72   0084A                                                                        ;
                        00  57  55  32  32  21  6B   63        00854         .ASCII  \ck!22UW\<0>                                          ;
                                                    010E002F    0085C P.AHJ:  .LONG   17694767                                              ;
                                                    00000000'   00860         .ADDRESS P.AHK                                                ;
6E  69  6C  79  63  20  6C  61  74  6F  54  20  20   20  20   00864 P.AHM:  .ASCII  \    Total cylinders!17UW    Tracks per c\  ;
72  54  20  20  20  20  57  55  37  31  21  73  72   65  64   00873                                                                        ;
                    63  20  72  65  70  20  73  6B   63  61   00882                                                                        ;
            57  55  30  32  21  72  65  64  6E  69   6C  79   0088C         .ASCII  \ylinder!20UW\                                        ;
                                                    010E0034    00898 P.AHL:  .LONG   17694772                                              ;
                                                    00000000'   0089C         .ADDRESS P.AHM                                                ;
                                                73  65  79      008A0 P.AHN:  .ASCII  \yes\                                                 ;
                                                6F  6E  20      008A3 P.AHO:  .ASCII  \ no\                                                 ;
                                                                008A6         .BLKB   2                                                     ;
20  20  65  6D  61  6E  20  74  73  6F  48  20  20   20  20   008A8 P.AHQ:  .ASCII  \    Host name      !#< !>"'!AF"    Host t\  ;
22  46  41  21  22  3E  21  20  3C  23  21  20  20   20  20   008B7                                                                        ;
                    74  20  74  73  6F  48  20  20   20  20   008C6                                                                        ;
20  65  6C  62  61  6C  69  61  76  61  20  2C  65   70  79   008D0         .ASCII  \ype, available          !AF, !AF\                    ;
21  20  2C  46  41  21  20  20  20  20  20  20  20   20  20   008DF                                                                        ;
                                                    46  41      008EE                                                                     ;
                                                    010E0048    008F0 P.AHP:  .LONG   17694792                                              ;
                                                    00000000'   008F4         .ADDRESS P.AHQ                                                ;
68  20  65  74  61  6E  72  65  74  6C  41  20  20   20  20   008F8 P.AHS:  .ASCII  \    Alternate host name   !#< !>"'!AF"   \  ;
20  3C  23  21  20  20  20  65  6D  61  6E  20  74   73  6F   00907                                                                        ;
                    20  20  20  22  46  41  21  22   3E  21   00916                                                                        ;
74  73  6F  68  20  65  74  61  6E  72  65  74  6C   41  20   00920         .ASCII  \ Alternate host type, avail    !AF, !AF-\  ;
20  20  20  6C  69  61  76  61  20  2C  65  70  79   74  20   0092F                 \<0>                                                  ;
                    00  46  41  21  20  2C  46  41   21  20   0093E                                                                        ;
                                                    010E004F    00948 P.AHR:  .LONG   17694799                                              ;
                                                    00000000'   0094C         .ADDRESS P.AHS                                                ;
20  6E  6F  69  74  61  63  6F  6C  6C  41  20  20   20  20   00950 P.AHU:  .ASCII  \    Allocation class!16UL\<0><0><0>               ;
        00  00  00  4C  55  36  31  21  73  73  61   6C  63   0095F                                                                        ;
                                                    010E0019    0096C P.AHT:  .LONG   17694745                                              ;
                                                    00000000'   00970         .ADDRESS P.AHU                                                ;
                                            70  75  6F  72   47  05    00974 P.AHV:  .ASCII  <5>\Group\                                    ;
                                        6D  65  74  73  79   53  06    0097A P.AHW:  .ASCII  <6>\System\                                   ;
                                    73  73  65  63  6F  72   50  07    00981 P.AHX:  .ASCII  <7>\Process\                                  ;
```

```
                                                                00989                   .BLKB   3
75 74 61 74 73 20 74 6E 75 6F 4D 20 20 20 20                    0098C P.AHZ:    .ASCII  \    Mount status!#< !>!AC    Cache name!\
43 20 20 20 20 43 41 21 3E 21 20 3C 23 21 73                    0099B
                  21 65 6D 61 6E 20 65 68 63 61                 009AA
                  00 00 22 46 41 21 22 3E 21 20 3C 23           009B4                   .ASCII  \#< !>"!AF"\<0><0>
                                                  010E0032      009C0 P.AHY:    .LONG   17694770
                                                  00000000'     009C4                   .ADDRESS P.AHZ
75 74 61 74 73 20 74 6E 75 6F 4D 20 20 20 20                    009C8 P.AIB:    .ASCII  \    Mount status!#< !>!AC    ACP process\
41 20 20 20 20 43 41 21 3E 21 20 3C 23 21 73                    009D7
                  73 73 65 63 6F 72 70 20 50 43                 009E6
46 41 21 22 3E 21 20 3C 23 21 65 6D 61 6E 20                    009F0                   .ASCII  \ name!#< !>"!AF"\
                                                  22            009FF
                                                  010E0038      00A00 P.AIA:    .LONG   17694776
                                                  00000000'     00A04                   .ADDRESS P.AIB
65 62 61 6C 20 65 6D 75 6C 6F 56 20 20 20 20                    00A08 P.AID:    .ASCII  \    Volume label!#< !>"!AF"    Relative \
20 20 20 22 46 41 21 22 3E 21 20 3C 23 21 6C                    00A17
                  20 65 76 69 74 61 6C 65 52 20                 00A26
31 21 72 65 62 6D 75 6E 20 65 6D 75 6C 6F 76                    00A30                   .ASCII  \volume number!17UW\<0><0>
                              00 00 57 55 37                    00A3F
                                                  010E003A      00A44 P.AIC:    .LONG   17694778
                                                  00000000'     00A48                   .ADDRESS P.AID
7A 69 73 20 72 65 74 73 75 6C 43 20 20 20 20                    00A4C P.AIF:    .ASCII  \    Cluster size!20UW    Transaction cou\
73 6E 61 72 54 20 20 20 20 57 55 30 32 21 65                    00A5B
                  75 6F 63 20 6E 6F 69 74 63 61                 00A6A
                        00 57 55 32 32 21 74 6E                 00A74                   .ASCII  \nt!22UW\<0>
                                                  010E002F      00A7C P.AIE:    .LONG   17694767
                                                  00000000'     00A80                   .ADDRESS P.AIF
73 6B 63 6F 6C 62 20 65 65 72 46 20 20 20 20                    00A84 P.AIH:    .ASCII  \    Free blocks!21UL    Maximum files al\
75 6D 69 78 61 4D 20 20 20 20 4C 55 31 32 21                    00A93
                  6C 61 20 73 65 6C 69 66 20 6D                 00AA2
                  00 00 4C 55 38 31 21 64 65 77 6F 6C           00AAC                   .ASCII  \lowed!18UL\<0><0>
                                                  010E0032      00AB8 P.AIG:    .LONG   17694770
                                                  00000000'     00ABC                   .ADDRESS P.AIH
6E 61 75 71 20 64 6E 65 74 78 45 20 20 20 20                    00AC0 P.AIJ:    .ASCII  \    Extend quantity!17UW    Mount count!\
6F 4D 20 20 20 20 57 55 37 31 21 79 74 69 74                    00ACF
                  21 74 6E 75 6F 63 20 74 6E 75                 00ADE
                                    57 55 38 32                 00AE8                   .ASCII  \28UW\
                                                  010E002C      00AEC P.AII:    .LONG   17694764
                                                  00000000'     00AF0                   .ADDRESS P.AIJ
73 75 74 61 74 73 20 65 6D 75 6C 6F 56 20 20                    00AF4 P.AIL:    .ASCII  \  Volume status: \<0><0><0>
                                    00 00 00 20 3A              00B03
                                                  010E0011      00B08 P.AIK:    .LONG   17694737
                                                  00000000'     00B0C                   .ADDRESS P.AIL
75 6F 6D 20 6F 74 20 74 63 65 6A 62 75 73 20                    00B10 P.AIN:    .ASCII  \ subject to mount verification,\<0>
6E 6F 69 74 61 63 69 66 69 72 65 76 20 74 6E                    00B1F
                                       00 2C                    00B2E
                                                  010E001F      00B30 P.AIM:    .LONG   17694751
                                                  00000000'     00B34                   .ADDRESS P.AIN
62 20 65 6D 6F 68 20 79 72 61 6D 69 72 70 20                    00B38 P.AIP:    .ASCII  \ primary home block is bad,\<0>
      00 2C 64 61 62 20 73 69 20 68 63 6F 6C                    00B47
                                                  010E001B      00B54 P.AIO:    .LONG   17694747
                                                  00000000'     00B58                   .ADDRESS P.AIP
20 78 65 64 6E 69 20 79 72 61 6D 69 72 70 20                    00B5C P.AIR:    .ASCII  \ primary index file header is bad,\<0>
20 73 69 20 72 65 64 61 65 68 20 65 6C 69 66                    00B6B
                                 00 2C 64 61 62                 00B7A
                                          00                    00B7F                   .ASCII  <0>
                                                  010E0022      00B80 P.AIQ:    .LONG   17694754
```

```
                                                00000000'  00B84                .ADDRESS P.AIR
68 6E 69 20 6E 6F 69 74 61 63 6F 6C 6C 61 20    00B88 P.AIT:  .ASCII  \ allocation inhibited because of error o\
20 65 73 75 61 63 65 62 20 64 65 74 69 62 69    00B97
                  6F 20 72 6F 72 72 65 20 66 6F 00BA6
          00 00 00 2C 70 61 6D 74 69 62 20 6E   00BB0                .ASCII  \n bitmap,\<0><0><0>
                                                010E0031   00BBC P.AIS:  .LONG   17694769
                                                00000000'  00BC0        .ADDRESS P.AIT
20 64 61 6F 6C 6E 75 20 74 6F 6E 20 6F 64 20    00BC4 P.AIV:  .ASCII  \ do not unload on dismount,\<0>
   00 2C 74 6E 75 6F 6D 73 69 64 20 6E 6F       00BD3
                                                010E001B   00BE0 P.AIU:  .LONG   17694747
                                                00000000'  00BE4        .ADDRESS P.AIV
74 65 6C 65 64 20 6E 6F 20 65 73 61 72 65 20    00BE8 P.AIX:  .ASCII  \ erase on delete,\<0><0><0>
                              00 00 00 2C 65    00BF7
                                                010E0011   00BFC P.AIW:  .LONG   17694737
                                                00000000'  00C00        .ADDRESS P.AIX
65 74 61 77 2D 68 67 69 68 20 65 6C 69 66 20    00C04 P.AIZ:  .ASCII  \ file high-water marking,\<0><0><0>
   00 00 00 2C 67 6E 69 6B 72 61 6D 20 72       00C13
                                                010E0019   00C20 P.AIY:  .LONG   17694745
                                                00000000'  00C24        .ADDRESS P.AIZ
73 69 64 20 73 69 20 67 6E 69 68 63 61 63 20    00C28 P.AJB:  .ASCII  \ caching is disabled,\<0><0><0>
                  00 00 00 2C 64 65 6C 62 61    00C37
                                                010E0015   00C40 P.AJA:  .LONG   17694741
                                                00000000'  00C44        .ADDRESS P.AJB
68 63 61 63 20 74 6E 65 74 78 45 20 20 20 20    00C48 P.AJD:  .ASCII  \    Extent cache size!15UW    Maximum bl\
20 20 20 20 57 55 35 31 21 65 7A 69 73 20 65    00C57
                  6C 62 20 6D 75 6D 69 78 61 4D 00C66
20 74 6E 65 74 78 65 20 6E 69 20 73 6B 63 6F    00C70                .ASCII  \ocks in extent cache!9UL\
                        4C 55 39 21 65 68 63 61 63 00C7F
                                                010E0040   00C88 P.AJC:  .LONG   17694784
                                                00000000'  00C8C        .ADDRESS P.AJD
63 61 63 20 44 49 20 65 6C 69 46 20 20 20 20    00C90 P.AJF:  .ASCII  \    File ID cache size!14UW    Blocks cu\
20 20 20 57 55 34 31 21 65 7A 69 73 20 65 68    00C9F
                  75 63 20 73 6B 63 6F 6C 42 20 00CAE
65 74 78 65 20 6E 69 20 79 6C 74 6E 65 72 72    00CB8                .ASCII  \rrently in extent cache!7UL\<0>
   00 4C 55 37 21 65 68 63 61 63 20 74 6E       00CC7
                                                010E0043   00CD4 P.AJE:  .LONG   17694787
                                                00000000'  00CD8        .ADDRESS P.AJF
65 68 63 61 63 20 61 74 6F 75 51 20 20 20 20    00CDC P.AJH:  .ASCII  \    Quota cache size!16UW    Maximum buf\
4D 20 20 20 20 57 55 36 31 21 65 7A 69 73 20    00CEB
                  66 75 62 20 6D 75 6D 69 78 61 00CFA
63 61 63 20 50 43 46 20 6E 69 20 73 72 65 66    00D04                .ASCII  \fers in FCP cache!11UW\<0><0>
                        00 00 57 55 31 31 21 65 68 00D13
                                                010E003E   00D1C P.AJG:  .LONG   17694782
                                                00000000'  00D20        .ADDRESS P.AJH
20 68 67 75 6F 72 68 74 2D 65 74 69 72 77 20    00D24 P.AJJ:  .ASCII  \ write-through caching enabled,\<0>
64 65 6C 62 61 6E 65 20 67 6E 69 68 63 61 63    00D33
                                    00 2C       00D42
                                                010E001F   00D44 P.AJI:  .LONG   17694751
                                                00000000'  00D48        .ADDRESS P.AJJ
63 61 63 20 6B 63 61 62 2D 65 74 69 72 77 20    00D4C P.AJL:  .ASCII  \ write-back caching enabled,\
   2C 64 65 6C 62 61 6E 65 20 67 6E 69 68       00D5B
                                                010E001C   00D68 P.AJK:  .LONG   17694748
                                                00000000'  00D6C        .ADDRESS P.AJL
                                    00000000    00000000   00D70 P.AJM:  .LONG   0, 0
72 65 70 20 2E 74 65 72 20 6E 69 4D 20 20 20    00D78 P.AJO:  .ASCII  \   Min ret. period (days)!10UW    Max re\
57 55 30 31 21 29 73 79 61 64 28 20 64 6F 69    00D87
                  65 72 20 78 61 4D 20 20 20 20 00D96
```

```
73 79 61 64 28 20 64 6F 69 72 65 70 20 2E 74 00DA0           .ASCII  \t. period (days)!17UW\<0><0><0>              ;
                  00 00 00 57 55 37 31 21 29 00DAF                                                                  ;
                                  010E003D 00DB8 P.AJN:     .LONG   17694781                                        ;
                                  00000000' 00DBC           .ADDRESS P.AJO                                          ;
                           00 53 41 21 00DC0 P.AJQ:         .ASCII  \!AS\<0>                                        ;
                                  010E0003 00DC4 P.AJP:     .LONG   17694723                                        ;
                                  00000000' 00DC8           .ADDRESS P.AJQ                                          ;

                                           T_YES=                   P.AHN
                                           T_NO=                    P.AHO
                                                     .EXTRN  SYS$NUM,IM

                                                     .PSECT  $CODE$,NOWRT,2

                                   03FC 00000        .ENTRY  DISPLAY_DISK, Save R2,R3,R4,R5,R6,R7,R8,R9    ; 1463
                 59 00000000G   00   9E 00002        MOVAB   SYS$NUMTIM, R9                                ;
                 58      F939   CF   9E 00009        MOVAB   ADD_TO_LIST_RTN, R8                           ;
                 57 000C0000G   00   9E 0000E        MOVAB   SHOQ$WRITE_LINE, R7                           ;
                 56      0000'  CF   9E 00015        MOVAB   T_YES, R6                                     ;
                 5E      F880   CE   9E 0001A        MOVAB   -1920(SP), SP                                 ;
      DC   AD   FF58   C6            24 28 0001F     MOVC3   #36, P.AHG, DESC_VECTOR                       ; 1488
                 53        04   AC   D0 00026        MOVL    SCRATCH, R3                                   ; 1501
                 0C        74   A3   E9 0002A        BLBC    116(R3), 1$                                   ;
                           74   AE   9F 0002E        PUSHAB  CLUBUF                                        ; 1503
                                53   DD 00031        PUSHL   R3                                            ;
           00000000G  00        02   FB 00033        CALLS   #2, SCAN_CLUSTER_LOCKS                        ;
                           08   AC   DD 0003A 1$:    PUSHL   FLAGS                                         ; 1510
                                53   DD 0003D        PUSHL   R3                                            ;
              00AD   C8         02   FB 0003F        CALLS   #2, CHECK_MNT_ALL                             ;
                     01         50   E8 00044        BLBS    R0, 2$                                        ;
                                04      00047        RET                                                   ;
              24     AE   FF7C   C6   9E 00048 2$:   MOVAB   P.AHH, ARGLIST                                ; 1513
              3C     AE     82   A6   9E 0004E       MOVAB   P.AHI, ARGLIST+24                             ; 1519
                                50   D4 00053        CLRL    I                                             ; 1521
              0000'CF40     79   A3   91 00055 3$:   CMPB    121(R3), DISK_TYPE[I]                         ;
                                09   12 0005C        BNEQ    4$                                            ;
              3C     AE  0000'CF40   D0 0005E        MOVL    DISK_LABEL[I], ARGLIST+24                     ; 1522
                                04   11 00065        BRB     5$                                            ;
      EA             50         21   F3 00067 4$:    AOBLEQ  #33, I, 3$                                    ; 1521
                                53   DD 0006B 5$:    PUSHL   R3                                            ; 1524
                           28   AE   9F 0006D        PUSHAB  ARGLIST                                       ;
              0000V  CF         02   FB 00070        CALLS   #2, COMMON_DISPLAY                            ;
              24     AE   0094  C3   D0 00075        MOVL    148(R3), ARGLIST                              , 1529
                                1E   13 0007B        BEQL    6$                                            ;
              28     AE     7C  A3   9A 0007D        MOVZBL  124(R3), ARGLIST+4                            ; 1532
              2C     AE     7E  A3   3C 00082        MOVZWL  126(R3), ARGLIST+8                            ; 1533
              30     AE     7D  A3   9A 00087        MOVZBL  125(R3), ARGLIST+12                           ; 1534
                            2C  AE   9F 0008C        PUSHAB  ARGLIST+8                                     ; 1538
                            F8  A6   9F 0008F        PUSHAB  P.AHL                                         ; 1536
                            2C  AE   9F 00092        PUSHAB  ARGLIST                                       ;
                            BC  A6   9F 00095        PUSHAB  P.AHJ                                         ; 1535
                     67         04   FB 00098        CALLS   #4, SHOW$WRITE_LINE                           ;
      05      04     A3         03   E0 0009B 6$:    BBS     #3, 4(R3), 7$                                 ; 1544
      3D      74     A3         04   E1 000A0        BBC     #4, 116(R3), 10$                              ;
              24     AE     1C  A3   9A 000A5 7$:    MOVZBL  28(R3), ARGLIST                               ; 1550
      24   AE        0F     24  AE   C3 000AA        SUBL3   ARGLIST, #15, ARGLIST                         ;
              28     AE     1C  A3   9A 000B0        MOVZBL  28(R3), ARGLIST+4                             ; 1551
```

```
                        2C   AE        1D   A3   9E 000B5          MOVAB   29(R3), ARGLIST+8                       ; 1552
                        30   AE             04   D0 000BA          MOVL    #4, ARGLIST+12                          ; 1553
                        34   AE        2C   A3   9E 000BE          MOVAB   44(R3), ARGLIST+16                      ; 1554
                        38   AE             03   D0 000C3          MOVL    #3, ARGLIST+20                          ; 1555
                   05   04   A3             01   E1 000C7          BBC     #1, 4(R3), 8$                           ; 1556
                             50             66   9E 000CC          MOVAB   T_YES, R0                               ;
                                            04   11 000CF          BRB     9$                                      ;
                             50        03   A6   9E 000D1 8$:      MOVAB   T_NO, R0                                ;
                        3C   AE             50   D0 000D5 9$:      MOVL    R0, ARGLIST+24                          ;
                                       24   AE   9F 000D9          PUSHAB  ARGLIST                                 ; 1560
                                       50   A6   9F 000DC          PUSHAB  P.AHP                                   ; 1559
                                  67        02   FB 000DF          CALLS   #2, SHOW$WRITE_LINE                     ;
                   3E   74   A3             04   E1 000E2 10$:     BBC     #4, 116(R3), 13$                        ; 1561
                        24   AE        30   A3   9A 000E7          MOVZBL  48(R3), ARGLIST                         ; 1564
              24   AE             08   24   AE   C3 000EC          SUBL3   ARGLIST, #8, ARGLIST                    ;
                        28   AE        30   A3   9A 000F2          MOVZBL  48(R3), ARGLIST+4                       ; 1565
                        2C   AE        31   A3   9E 000F7          MOVAB   49(R3), ARGLIST+8                       ; 1566
                        30   AE             04   D0 000FC          MOVL    #4, ARGLIST+12                          ; 1567
                        34   AE        40   A3   9E 00100          MOVAB   64(R3), ARGLIST+16                      ; 1568
                        38   AE             03   D0 00105          MOVL    #3, ARGLIST+20                          ; 1569
                   05   04   A3             02   E1 00109          BBC     #2, 4(R3), 11$                          ; 1570
                             50             66   9E 0010E          MOVAB   T_YES, R0                               ;
                                            04   11 00111          BRB     12$                                     ;
                             50        03   A6   9E 00113 11$:     MOVAB   T_NO, R0                                ;
                        3C   AE             50   D0 00117 12$:     MOVL    R0, ARGLIST+24                          ;
                                       24   AE   9F 0011B          PUSHAB  ARGLIST                                 ; 1574
                                  00A8 C6   9F 0011E              PUSHAB  P.AHR                                   ; 1573
                                  67        02   FB 00122          CALLS   #2, SHOW$WRITE_LINE                     ;
                                       54   A3   D5 00125 13$:     TSTL    84(R3)                                  ; 1581
                                            0A   13 00128          BEQL    14$                                     ;
                                       54   A3   9F 0012A          PUSHAB  84(R3)                                  ; 1583
                                  00CC C6   9F 0012D              PUSHAB  P.AHT                                   ;
                                  67        02   FB 00131          CALLS   #2, SHOW$WRITE_LINE                     ;
                                  03   0099 C3   E8 00134 14$:     BLBS    153(R3), 15$                            ; 1588
                                       0295 31   00139            BRW     42$                                     ;
                        28   AE             0C   D0 0013C 15$:     MOVL    #12, ARGLIST+4                          ; 1591
                             50             0B   D0 00140          MOVL    #11, IDX                                ; 1592
                                  20   00B8 C043 91 00143 16$:     CMPB    184(IDX)[R3], #32                       ; 1593
                                            06   12 00149          BNEQ    17$                                     ;
                                       28   AE   D7 0014B          DECL    ARGLIST+4                               ; 1594
                                  F2        50   F4 0014E          SOBGEQ  IDX, 16$                                ; 1593
              24   AE             12        28   AE   C3 00151 17$:     SUBL3   ARGLIST+4, #18, ARGLIST             ; 1596
                        2C   AE        00B8 C3   9E 00157          MOVAB   184(R3), ARGLIST+8                      ; 1597
                        30   AE        00B6 C3   3C 0015D          MOVZWL  182(R3), ARGLIST+12                     ; 1598
                        34   AE        00CE C3   3C 00163          MOVZWL  206(R3), ARGLIST+16                     ; 1599
                        38   AE        009B C3   3C 00169          MOVZWL  155(R3), ARGLIST+20                     ; 1600
                        3C   AE        00D2 C3   7D 0016F          MOVQ    210(R3), ARGLIST+24                     ; 1601
                        44   AE        00D0 C3   3C 00175          MOVZWL  208(R3), ARGLIST+32                     ; 1603
                        48   AE        00CC C3   3C 0017B          MOVZWL  204(R3), ARGLIST+36                     ; 1604
                             54   009A C3   9E 00181              MOVAB   154(R3), R4                             ; 1609
                   08        64        06   E1 00186              BBC     #6, (R4), 18$                           ;
                        50   AE   00D4 C6   9E 0018A              MOVAB   P.AHV, ARGLIST+44                       ; 1610
                                            12   11 00190          BRB     20$                                     ;
                                            64   95 00192 18$:     TSTB    (R4)                                    ; 1611
                                            08   18 00194          BGEQ    19$                                     ;
                        50   AE   00DA C6   9E 00196              MOVAB   P.AHW, ARGLIST+44                       ; 1612
                                            06   11 0019C          BRB     20$                                     ;
```

```
                50   AE      00E1   C6   9E 0019E 19$:    MOVAB   P.AHX, ARGLIST+44                        ; 1613
                4C   AE        50   BE   9A 001A4 20$:    MOVZBL  @ARGLIST+44, ARGLIST+40                  ; 1614
          4C    AE              14     4C   AE   C3 001A9        SUBL3   ARGLIST+40, #20, ARGLIST+40
                                50   009E   C3   9E 001AF        MOVAB   158(R3), R0                      ; 1619
          0F          04        A3          05   E1 001B4        BBC     #5, 4(R3), 21$                   ; 1615
                                51   0120   C6   9E 001B9        MOVAB   P.AHY, FAO_CTR                   ; 1618
                                50          60   9A 001BE        MOVZBL  (R0), R0                         ; 1619
          54    AE              1B          50   C3 001C1        SUBL3   R0, #27, ARGLIST+48
                                             0D   11 001C6        BRB     22$                              ; 1615
                                51   0160   C6   9E 001C8 21$:    MOVAB   P.AIA, FAO_CTR                   ; 1623
                                50          60   9A 001CD        MOVZBL  (R0), R0                         ; 1624
          54    AE              15          50   C3 001D0        SUBL3   R0, #21, ARGLIST+48
                58    AE              50   D0 001D5 22$:    MOVL    R0, ARGLIST+52                   ; 1626
                5C    AE        009F   C3   9E 001D9        MOVAB   159(R3), ARGLIST+56              ; 1627
                                  4C   AE   9F 001DF        PUSHAB  ARGLIST+40                       ; 1639
                                  51   DD 001E2              PUSHL   FAO_CTR                          ; 1638
                                  4C   AE   9F 001E4        PUSHAB  ARGLIST+32                       ; 1637
                                024C   C6   9F 001E7        PUSHAB  P.AII                            ; 1635
                                  4C   AE   9F 001EB        PUSHAB  ARGLIST+24
                                0218   C6   9F 001EE        PUSHAB  P.AIG                            ; 1633
                                  4C   AE   9F 001F2        PUSHAB  ARGLIST+16
                                01DC   C6   9F 001F5        PUSHAB  P.AIE                            ; 1631
                                  44   AE   9F 001F9        PUSHAB  ARGLIST
                                01A4   C6   9F 001FC        PUSHAB  P.AIC                            ; 1629
                                  4C   AE   9F 00200        PUSHAB  ARGLIST
                                F9E4   C6   9F 00203        PUSHAB  ASCID_NULL
                          67           0C   FB 00207        CALLS   #12, SHOWSWRITE_LINE
                          6E            DC   AD   9E 0020A        MOVAB   DESC_VECTOR, PTR                 ; 1644
                    E0    AD      FD5C   CD   9E 0020E        MOVAB   BUFFER, DESC_VECTOR+4            ; 1645
              0000'       CF          06   D0 00214        MOVL    #6, LEADING_BLANKS               ; 1646
                          7E     50   8F   9A 00219        MOVZBL  #80, -(SP)                       ; 1647
                                0268   C6   9F 0021D        PUSHAB  P.AIK
                                  08   AE   9F 00221        PUSHAB  PTR
                          68           03   FB 00224        CALLS   #3, ADD_TO_LIST_RTN
                          52      00DC   C3   9E 00227        MOVAB   220(R3), R2                      ; 1648
          0E              62          02   E1 0022C        BBC     #2, (R2), 23$                    ; 1649
                          7E     50   8F   9A 00230        MOVZBL  #80, -(SP)
                                0290   C6   9F 00234        PUSHAB  P.AIM
                                  08   AE   9F 00238        PUSHAB  PTR
                          68           03   FB 0023B        CALLS   #3, ADD_TO_LIST_RTN
          0E              64          02   E1 0023E 23$:    BBC     #2, (R4), 24$                    ; 1650
                          7E     50   8F   9A 00242        MOVZBL  #80, -(SP)                       ; 1651
                                02B4   C6   9F 00246        PUSHAB  P.AIO
                                  08   AE   9F 0024A        PUSHAB  PTR
                          68           03   FB 0024D        CALLS   #3, ADD_TO_LIST_RTN
          0E              64          03   E1 00250 24$:    BBC     #3, (R4), 25$                    ; 1652
                          7E     50   8F   9A 00254        MOVZBL  #80, -(SP)                       ; 1653
                                02E0   C6   9F 00258        PUSHAB  P.AIQ
                                  08   AE   9F 0025C        PUSHAB  PTR
                          68           03   FB 0025F        CALLS   #3, ADD_TO_LIST_RTN
          0E              64          04   E1 00262 25$:    BBC     #4, (R4), 26$                    ; 1654
                          7E     50   8F   9A 00266        MOVZBL  #80, -(SP)                       ; 1655
                                031C   C6   9F 0026A        PUSHAB  P.AIS
                                  08   AE   9F 0026E        PUSHAB  PTR
                          68           03   FB 00271        CALLS   #3, ADD_TO_LIST_RTN
          0E      0089    C3          04   E0 00274 26$:    BBS     #4, 137(R3), 27$                 ; 1656
                          7E     50   8F   9A 0027A        MOVZBL  #80, -(SP)                       ; 1657
```

```
                              0340  C6  9F 0027E        PUSHAB  P.AIU                              ;
                                08  AE  9F 00282        PUSHAB  PTR                                ;
                        68          03  FB 00285        CALLS   #3, ADD_TO_LIST_RTN                ;
                        02    009D  C3  91 00288 27$:   CMPB    157(R3), #2                        ; 1658
                                    03  13 0028D        BEQL    28$                                ;
                                  00F0  31 0028F        BRW     39$                                ;
               0E       62          03  E1 00292 28$:   BBC     #3, (R2), 29$                      ; 1665
                        7E      50  8F  9A 00296        MOVZBL  #80, -(SP)                         ; 1666
                              035C  C6  9F 0029A        PUSHAB  P.AIW                              ;
                                08  AE  9F 0029E        PUSHAB  PTR                                ;
                        68          03  FB 002A1        CALLS   #3, ADD_TO_LIST_RTN                ;
               0E       62          04  E0 002A4 29$:   BBS     #4, (R2), 30$                      ; 1667
                        7E      50  8F  9A 002A8        MOVZBL  #80, -(SP)                         ; 1668
                              0380  C6  9F 002AC        PUSHAB  P.AIY                              ;
                                08  AE  9F 002B0        PUSHAB  PTR                                ;
                        68          03  FB 002B3        CALLS   #3, ADD_TO_LIST_RTN                ;
               0A       62          01  E1 002B6 30$:   BBC     #1, (R2), 31$                      ; 1669
                        7E      50  8F  9A 002BA        MOVZBL  #80, -(SP)                         ; 1670
                              03A0  C6  9F 002BE        PUSHAB  P.AJA                              ;
                                    6F  11 002C2        BRB     35$                                ;
                  24    AE    00EF  C3  3C 002C4 31$:   MOVZWL  239(R3), ARGLIST                   ; 1675
                                    08  12 002CA        BNEQ    32$                                ;
                                30  AE  D4 002CC        CLRL    ARGLIST+12                         ; 1677
                                28  AE  D4 002CF        CLRL    ARGLIST+4                          ;
                                    20  11 002D2        BRB     33$                                ;
                          28    AE    00F1  C3  3C 002D4 32$:   MOVZWL  241(R3), ARGLIST+4         ; 1680
  1C   AE          00   00D2    C3      28  AE  7A 002DA        EMUL    ARGLIST+4, 210(R3), #0, TEMP   ; 1681
  50         28    AE     1C    AE 000003E8  8F  7B 002E3        EDIV    #1000, TEMP, ARGLIST+4, DUMMY ; 1682
                          30    AE    00F3  C3  D0 002EE        MOVL    243(R3), ARGLIST+12        ; 1683
                          2C    AE    00ED  C3  3C 002F4 33$:   MOVZWL  237(R3), ARGLIST+8         ; 1685
                          34    AE    00F7  C3  3C 002FA        MOVZWL  247(R3), ARGLIST+16        ; 1686
                          38    AE    00F9  C3  3C 00300        MOVZWL  249(R3), ARGLIST+20        ; 1687
                                34  AE  9F 00306        PUSHAB  ARGLIST+16                         ; 1694
                              047C  C6  9F 00309        PUSHAB  P.AJG                              ; 1692
                                34  AE  9F 0030D        PUSHAB  ARGLIST+8                          ;
                              0434  C6  9F 00310        PUSHAB  P.AJE                              ; 1690
                                34  AE  9F 00314        PUSHAB  ARGLIST                            ;
                              03E8  C6  9F 00317        PUSHAB  P.AJC                              ; 1689
                        67          06  FB 0031B        CALLS   #6, SHOW$WRITE_LINE                ;
                        0A          62  E9 0031E        BLBC    (R2), 34$                          ; 1695
                        7E      50  8F  9A 00321        MOVZBL  #80, -(SP)                         ; 1696
                              04A4  C6  9F 00325        PUSHAB  P.AJI                              ;
                                    08  11 00329        BRB     35$                                ;
                        7E      50  8F  9A 0032B 34$:   MOVZBL  #80, -(SP)                         ; 1697
                              04C8  C6  9F 0032F        PUSHAB  P.AJK                              ;
                                08  AE  9F 00333 35$:   PUSHAB  PTR                                ;
                        68          03  FB 00336        CALLS   #3, ADD_TO_LIST_RTN                ;
                        50          01  CE 00339        MNEGL   #1, R0                             ; 1703
                  04D4  C6    00E1  C3  D1 0033C        CMPL    225(R3), P.AJM+4                   ;
                                    11  19 00343        BLSS    38$                                ;
                                    0B  14 00345        BGTR    36$                                ;
                  04D0  C6    00DD  C3  D1 00347        CMPL    221(R3), P.AJM                     ;
                                    04  13 0034E        BEQL    37$                                ;
                                    04  1F 00350        BLSSU   38$                                ;
                                    50  D6 00352 36$:   INCL    R0                                 ;
                                    50  D6 00354 37$:   INCL    R0                                 ;
                                    50  D5 00356 38$:   TSTL    R0                                 ;
```

```
                                    28  13 00358          BEQL    39$                                                     ;
                              00DD  C3  9F 0035A          PUSHAB  221(R3)                                                 ; 1710
                                18  AE  9F 0035E          PUSHAB  MIN_TIM                                                 ;
                                69      02  FB 00361      CALLS   #2, SYS$NUMTIM                                          ;
                              00E5  C3  9F 00364          PUSHAB  229(R3)                                                 ; 1712
                                08  AE  9F 00368          PUSHAB  MAX_TIM                                                 ;
                                69      02  FB 0036B      CALLS   #2, SYS$NUMTIM                                          ;
                          24    AE    18  AE  3C 0036E    MOVZWL  MIN_TIM+4, ARGLIST                                      ; 1713
                          28    AE    08  AE  3C 00373    MOVZWL  MAX_TIM+4, ARGLIST+4                                    ; 1714
                                24  AE  9F 00378          PUSHAB  ARGLIST                                                 ; 1716
                              0518  C6  9F 0037B          PUSHAB  P.AJN                                                   ; 1715
                                67      02  FB 0037F      CALLS   #2, SHOW$WRITE_LINE                                     ;
                                24  AE  9F 00382 39$:     PUSHAB  ARGLIST                                                 ; 1723
                              F9E4  C6  9F 00385          PUSHAB  ASCID_NULL                                              ;
                                67      02  FB 00389      CALLS   #2, SHOW$WRITE_LINE                                     ;
                                11    DC  AD  D1 0038C    CMPL    DESC_VECTOR, #17                                        ; 1728
                                    3F  13 00390          BEQL    42$                                                     ;
                                    52  D4 00392          CLRL    I                                                       ; 1729
                      50            52      01  78 00394 40$:     ASHL    #1, I, R0                                       ; 1731
                                    54  D4 00398          CLRL    R4                                                      ;
                                  E4 AD40  D5 0039A       TSTL    DESC_VECTOR+8[R0]                                       ;
                                    16  12 0039E          BNEQ    41$                                                     ;
                                    54  D6 003A0          INCL    R4                                                      ;
                      51            52      01  78 003A2  ASHL    #1, I, R1                                               ; 1735
                      50            52      01  78 003A6  ASHL    #1, I, R0                                               ; 1736
                      50      E0 AD41    DC AD40  C1 003AA  ADDL3   DESC_VECTOR[R0], DESC_VECTOR+4[R1], R0                  ;
                              FF    A0      2E  90 003B2    MOVB    #46, -1(R0)                                             ; 1737
                      50            52      01  78 003B6 41$:     ASHL    #1, I, R0                                       ; 1739
                              24    AE    DC AD40  DE 003BA   MOVAL   DESC_VECTOR[R0], ARGLIST                            ;
                                24  AE  9F 003C0          PUSHAB  ARGLIST                                                 ; 1740
                              0524  C6  9F 003C3          PUSHAB  P.AJP                                                   ;
                                67      02  FB 003C7      CALLS   #2, SHOW$WRITE_LINE                                     ;
                                04      54  E8 003CA      BLBS    R4, 42$                                                 ; 1741
                        C3            52      03  F3 003CD  AOBLEQ  #3, I, 40$                                           ; 1729
                                74  AE  9F 003D1 42$:     PUSHAB  CLUBUF                                                  ; 1749
                                    53  DD 003D4          PUSHL   R3                                                      ;
                            00D7  C8      02  FB 003D6    CALLS   #2, CLUSTER_NODES                                       ;
                                    53  DD 003DB          PUSHL   R3                                                      ; 1754
                            01D3  C8      01  FB 003DD    CALLS   #1, SHOW_DEVICE_ACL                                     ;
                                    04 003E2              RET                                                             ; 1757

; Routine Size:  995 bytes,    Routine Base:  $CODE$ + 06BA
```

```
;1382     1758  1 GLOBAL ROUTINE display_magtape (scratch, flags) : NOVALUE =
;1383     1759  2 BEGIN
;1384     1760  2
;1385     1761  2 !---
;1386     1762  2 !
;1387     1763  2 ! This is the display routine for magtapes.
;1388     1764  2 !
;1389     1765  2 ! Inputs
;1390     1766  2 !       SCRATCH - contains data about the magtape.
;1391     1767  2 !       FLAGS   - contains info on what to print
;1392     1768  2 !
;1393     1769  2 ! Outputs
;1394     1770  2 !       None
;1395     1771  2 !
;1396     1772  2 !---
;1397     1773  2
;1398     1774  2 MAP
;1399     1775  2     scratch : REF $BBLOCK,
;1400     1776  2     flags : REF $BBLOCK;
;1401     1777  2
;1402     1778  2 LOCAL
;1403     1779  2     clubuf : VECTOR[1024, BYTE],                ! Buffer for lock information
;1404     1780  2     arglist : VECTOR[20];
;1405     1781  2
;1406     1782  2 !
;1407     1783  2 ! First thing, fix up the scratch area with any available clusterwide info
;1408     1784  2 !
;1409     1785  2 IF .$BBLOCK[scratch[d_l_devchar2], dev$v_clu]   ! If available clusterwide
;1410     1786  2 THEN
;1411     1787  2     scan_cluster_locks (.scratch, clubuf);
;1412     1788  2
;1413     1789  2 !
;1414     1790  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;1415     1791  2 ! then check to see if the device is mounted/allocated.  If not, then
;1416     1792  2 ! don't display information on the device.
;1417     1793  2 !
;1418     1794  2 IF NOT check_mnt_all(.scratch, .flags)
;1419     1795  2 THEN RETURN;
;1420     1796  2
;1421     1797  2 arglist[0] = cstring('Magtape ');
;1422     1798  2
;1423     1799  2 !
;1424     1800  2 ! Determine which type of tape drive it is.  If not in the list, then
;1425     1801  2 ! simply say that it is unknown.
;1426     1802  2 !
;1427     1803  2 arglist[6] = unknown;                          ! Assume unknown
;1428     1804  2 INCR i FROM 0 TO tape_number - 1 DO            ! Loop thru known magtape types
;1429     1805  3     (IF .scratch[d_b_devtype] EQLU .tape_type[.i]
;1430     1806  2     THEN (arglist[6] = .tape_label[.i]; EXITLOOP));
;1431     1807  2
;1432     1808  2 common_display(arglist, .scratch);           ! Display stuff common to all devices
;1433     1809  2
;1434     1810  2 !
;1435     1811  2 ! Display other data
;1436     1812  2 !
;1437     1813  2 IF .scratch[d_b_cont]
;1438     1814  2 THEN
```

```
;  1439     1815  3     BEGIN
;  1440     1816  3     arglist[0] = mvl$s_vollbl;
;  1441     1817  3     arglist[1] = scratch[d_t_volnam];                    ! Volume label
;  1442     1818  3     arglist[2] = .scratch[d_w_rvn];                      ! Relative volume number
;  1443     1819  3     arglist[3] = .scratch[d_w_recordsz];                 ! size in bytes
;  1444     1820  3     arglist[4] = .scratch[d_w_trans];                    ! Transaction count
;  1445     1821  3     IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_group),1,0]
;  1446     1822  3     THEN arglist[6] = cstring('Group')
;  1447     1823  3     ELSE IF .$BBLOCK[scratch[d_b_status], 0,$BITPOSITION(vcb$v_system),1,0]
;  1448     1824  3     THEN arglist[6] = cstring('System')
;  1449     1825  3     ELSE arglist[6] = cstring('Process');
;  1450     1826  3     arglist[5] = 20 - .(.arglist[6])<0,8>;
;  1451     1827  3     arglist[7] = .scratch[d_w_mcount];                   ! Count of sharers
;  1452     1828  3     arglist[8] = 14 - .scratch[d_t_acpnam];
;  1453     1829  3     arglist[9] = .scratch[d_t_acpnam];
;  1454     1830  3     arglist[10] = scratch[d_t_acpnam] + 1;
;  1455     1831  2     END;
;  1456     1832  2 !
;  1457     1833  2 ! Check for the specific tape characteristics
;  1458     1834  2 !
;  1459     1835  3 arglist[12] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_density] EQL 0
;  1460     1836  3                OR .$BBLOCK[scratch[d_l_devdepend], mt$v_density] EQLU mt$k_normal11
;  1461     1837  3                THEN cstring('Normal-11')
;  1462     1838  3                ELSE IF .$BBLOCK[scratch[d_l_devdepend], mt$v_density] EQLU mt$k_nrzi_800
;  1463     1839  3                THEN cstring('800')
;  1464     1840  3                ELSE IF .$BBLOCK[scratch[d_l_devdepend], mt$v_density] EQLU mt$k_pe_1600
;  1465     1841  3                THEN cstring('1600')
;  1466     1842  3                ELSE IF .$BBLOCK[scratch[d_l_devdepend], mt$v_density] EQLU mt$k_gcr_6250
;  1467     1843  3                THEN cstring('6250')
;  1468     1844  2                ELSE unknown);
;  1469     1845  2 arglist[11] = 25 - .(.arglist[12])<0,8>;
;  1470     1846  2 IF .$BBLOCK[scratch[d_l_devdepend], mt$v_format] EQL 0
;  1471     1847  2 THEN arglist[14] = .format_label[0]
;  1472     1848  2 ELSE
;  1473     1849  3     BEGIN
;  1474     1850  3     arglist[14] = unknown;
;  1475     1851  3     INCR i FROM 0 TO format_number - 1
;  1476     1852  4     DO (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_format] EQLU .format_type[.i]
;  1477     1853  3     THEN (arglist[14] = .format_label[.i]; EXITLOOP));
;  1478     1854  2     END;
;  1479     1855  2 arglist[13] = 33 - .(.arglist[14])<0,8>;
;  1480     1856  2
;  1481     1857  2 IF .scratch[d_b_cont]
;  1482     1858  2 THEN show$write_line(ascid_null, arglist[0],
;  1483     1859  2                      %ASCID '    Volume label            "!AF"    Relative volume no.!20UW',
;  1484     1860  2                      arglist[0],
;  1485     1861  2                      %ASCID '    Record size!21UW    Transaction count!22UW',
;  1486     1862  2                      arglist[3],
;  1487     1863  2                      %ASCID '    Mount status!#< !>!AC    Mount count!28UW',
;  1488     1864  2                      arglist[5],
;  1489     1865  2                      %ASCID '    ACP process name!#< !>"!AF"',
;  1490     1866  2                      arglist[8]);
;  1491     1867  2
;  1492     1868  2 show$write_line(%ASCID '    Density!#< !>!AC    Format!#< !>!AC',
;  1493     1869  2                  arglist[11]);
;  1494     1870  2
;  1495     1871  3 arglist[0] = (IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_unload), 1, 0]
```

```
;  1496          1872  3                THEN UPLIT BYTE(0)
;  1497          1873  2                ELSE cstring('no-unload on dismount, '));
;  1498          1874  2
;  1499          1875  3 arglist[1] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_bot]
;  1500          1876  3                THEN cstring('beginning-of-tape, ')
;  1501          1877  2                ELSE UPLIT BYTE(0));
;  1502          1878  2
;  1503          1879  3 arglist[2] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_eof]
;  1504          1880  3                THEN cstring('end-of-file, ')
;  1505          1881  2                ELSE UPLIT BYTE(0));
;  1506          1882  2
;  1507          1883  3 arglist[3] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_eot]
;  1508          1884  3                THEN cstring('end-of-tape, ')
;  1509          1885  2                ELSE UPLIT BYTE(0));
;  1510          1886  2
;  1511          1887  3 arglist[4] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_hwl]
;  1512          1888  3                THEN cstring('write-locked, ')
;  1513          1889  2                ELSE UPLIT BYTE(0));
;  1514          1890  2
;  1515          1891  3 arglist[5] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_lost]
;  1516          1892  3                THEN cstring('position lost, ')
;  1517          1893  2                ELSE UPLIT BYTE(0));
;  1518          1894  2
;  1519          1895  3 arglist[6] = (IF .$BBLOCK[scratch[d_l_devdepend], mt$v_parity]
;  1520          1896  3                THEN cstring('even')
;  1521          1897  2                ELSE cstring('odd'));
;  1522          1898  2
;  1523          1899  2
;  1524          1900  2 !
;  1525          1901  2 ! Format the data for output and print it.
;  1526          1902  2 !
;  1527          1903  2 show$write_line(ascid_null, arglist);
;  1528          1904  2 show$write_line(%ASCID '  Volume status:  !AC!AC!AC!AC!AC!AC!AC parity.', arglist);
;  1529          1905  2
;  1530          1906  2 !
;  1531          1907  2 ! Let them know if it is accessed elsewhere
;  1532          1908  2 !
;  1533          1909  2 cluster_nodes (.scratch, clubuf);
;  1534          1910  2
;  1535          1911  2 !
;  1536          1912  2 ! Show any access control list which might be present
;  1537          1913  2 !
;  1538          1914  2 show_device_acl (.scratch);
;  1539          1915  2
;  1540          1916  2 RETURN;
;  1541          1917  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                    20  65  70  61  74  67  61  4D  08  00DCC P.AJR:  .ASCII  <8>\Magtape \                ;
                        6E  77  6F  6E  6B  6E  75  07  00DD5 P.AJS:  .ASCII  <7>\unknown\                 ;
                                70  75  6F  72  47  05  00DDD P.AJT:  .ASCII  <5>\Group\                   ;
                            6D  65  74  73  79  53  06  00DE3 P.AJU:  .ASCII  <6>\System\                  ;
                        73  73  65  63  6F  72  50  07  00DEA P.AJV:  .ASCII  <7>\Process\                 ;
                31  31  2D  6C  61  6D  72  6F  4E  09  00DF2 P.AJW:  .ASCII  <9>\Normal-11\              ;
```

```
                                                30 30 38 03  00DFC P.AJX:  .ASCII  <3>\800\
                                             30 30 36 31 04  00E00 P.AJY:  .ASCII  <4>\1600\
                                             30 35 32 36 04  00E05 P.AJZ:  .ASCII  <4>\6250\
                                    6E 77 6F 6E 6B 6E 75 07  00E0A P.AKA:  .ASCII  <7>\unknown\
                                    6E 77 6F 6E 6B 6E 75 07  00E12 P.AKB:  .ASCII  <7>\unknown\
                                                            00E1A          .BLKB   2
65 62 61 6C 20 65 6D 75 6C 6F 56 20 20 20 20  00E1C P.AKD:  .ASCII  \    Volume label            "!AF"    Rel\
21 22 20 20 20 20 20 20 20 20 20 20 20 20 6C  00E2B
                  6C 65 52 20 20 20 20 22 46 41  00E3A
6F 6E 20 65 6D 75 6C 6F 76 20 65 76 69 74 61  00E44          .ASCII  \ative volume no.!20UW\<0><0><0>
                  00 00 00 57 55 30 32 21 2E  00E53
                                    010E003D  00E5C P.AKC:  .LONG   17694781
                                    00000000'  00E60          .ADDRESS P.AKD
65 7A 69 73 20 64 72 6F 63 65 52 20 20 20 20  00E64 P.AKF:  .ASCII  \    Record size!21UW    Transaction coun\
61 73 6E 61 72 54 20 20 20 20 57 55 31 32 21  00E73
                  6E 75 6F 63 20 6E 6F 69 74 63  00E82
                  00 00 57 55 32 32 21 74  00E8C          .ASCII  \t!22UW\<0><0>
                                    010E002E  00E94 P.AKE:  .LONG   17694766
                                    00000000'  00E98          .ADDRESS P.AKF
75 74 61 74 73 20 74 6E 75 6F 4D 20 20 20 20  00E9C P.AKH:  .ASCII  \    Mount status!#< !>!AC    Mount count\
4D 20 20 20 20 43 41 21 3E 21 20 3C 23 21 73  00EAB
                  74 6E 75 6F 63 20 74 6E 75 6F  00EBA
                  00 00 00 57 55 38 32 21  00EC4          .ASCII  \!28UW\<0><0><0>
                                    010E002D  00ECC P.AKG:  .LONG   17694765
                                    00000000'  00ED0          .ADDRESS P.AKH
73 73 65 63 6F 72 70 20 50 43 41 20 20 20 20  00ED4 P.AKJ:  .ASCII  \    ACP process name!#< !>"!AF"\<0>
46 41 21 22 3E 21 20 3C 23 21 65 6D 61 6E 20  00EE3
                                    00 22  00EF2
                                    010E001F  00EF4 P.AKI:  .LONG   17694751
                                    00000000'  00EF8          .ADDRESS P.AKJ
20 3C 23 21 79 74 69 73 6E 65 44 20 20 20 20  00EFC P.AKL:  .ASCII  \    Density!#< !>!AC    Format!#< !>!AC-
74 61 6D 72 6F 46 20 20 20 20 43 41 21 3E 21  00F0B                  \<0>
                  00 43 41 21 3E 21 20 3C 23 21  00F1A
                                    010E0027  00F24 P.AKK:  .LONG   17694759
                                    00000000'  00F28          .ADDRESS P.AKL
                                             00  00F2C P.AKM:  .BYTE   0
64 20 6E 6F 20 64 61 6F 6C 6E 75 2D 6F 6E 17  00F2D P.AKN:  .ASCII  <23>\no-unload on dismount, \
                  20 2C 74 6E 75 6F 6D 73 69  00F3C
74 2D 66 6F 2D 67 6E 69 6E 6E 69 67 65 62 13  00F45 P.AKO:  .ASCII  <19>\beginning-of-tape, \
                                    20 2C 65 70 61  00F54
                                             00  00F59 P.AKP:  .BYTE   0
     20 2C 65 6C 69 66 2D 66 6F 2D 64 6E 65 0D  00F5A P.AKQ:  .ASCII  <13>\end-of-file, \
                                             00  00F68 P.AKR:  .BYTE   0
     20 2C 65 70 61 74 2D 66 6F 2D 64 6E 65 0D  00F69 P.AKS:  .ASCII  <13>\end-of-tape, \
                                             00  00F77 P.AKT:  .BYTE   0
20 2C 64 65 6B 63 6F 6C 2D 65 74 69 72 77 0E  00F78 P.AKU:  .ASCII  <14>\write-locked, \
                                             00  00F87 P.AKV:  .BYTE   0
2C 74 73 6F 6C 20 6E 6F 69 74 69 73 6F 70 0F  00F88 P.AKW:  .ASCII  <15>\position lost, \
                                             20  00F97
                                             00  00F98 P.AKX:  .BYTE   0
                                    6E 65 76 65 04  00F99 P.AKY:  .ASCII  <4>\even\
                                       64 64 6F 03  00F9E P.AKZ:  .ASCII  <3>\odd\
                                                   00FA2          .BLKB   2
73 75 74 61 74 73 20 65 6D 75 6C 6F 56 20 20  00FA4 P.ALB:  .ASCII  \  Volume status:  !AC!AC!AC!AC!AC!AC!AC \
43 41 21 43 41 21 43 41 21 43 41 21 20 20 3A  00FB3
                  20 43 41 21 43 41 21 43 41 21  00FC2
                        00 2E 79 74 69 72 61 70  00FCC          .ASCII  \parity.\<0>
```

```
                                010E002F  00FD4 P.ALA:  .LONG   17694767                                          ;
                                00000000' 00FD8         .ADDRESS P.ALB                                         ;


                                                        .PSECT  $CODE$,NOWRT,2

                                    003C 00000          .ENTRY  DISPLAY_MAGTAPE, Save R2,R3,R4,R5              ; 1758
                          55  0000'  CF   9E 00002      MOVAB   FORMAT_LABEL, R5                               ;
                          54 00000000G 00   9E 00007    MOVAB   SHOW$WRITE_LINE, R4                            ;
                          5E   FBB0   CE   9E 0000E     MOVAB   -1104(SP), SP                                  ;
                          52     04   AC   D0 00013     MOVL    SCRATCH, R2                                    ; 1785
                          0C     74   A2   E9 00017     BLBC    116(R2), 1$                                    ;
                                 50   AE   9F 0001B     PUSHAB  CLUBUF                                         ; 1787
                                      52   DD 0001E     PUSHL   R2                                             ;
                00000000G  00         02   FB 00020     CALLS   #2, SCAN_CLUSTER_LOCKS                         ;
                                      08   AC   DD 00027 1$:    PUSHL   FLAGS                                  ; 1794
                                      52   DD 0002A     PUSHL   R2                                             ;
                    F5DF  CF          02   FB 0002C     CALLS   #2, CHECK_MNT_ALL                              ;
                          01          50   E8 00031     BLBS    R0, 2$                                         ;
                                      04   00034        RET                                                    ;
                          6E  0000'   CF   9E 00035 2$:  MOVAB   P.AJR, ARGLIST                                 ; 1797
                        18    AE    0000' CF   9E 0003A  MOVAB   P.AJS, ARGLIST+24                              ; 1803
                                      50   D4 00040     CLRL    I                                              ; 1805
                        C8 A540       79   A2   91 00042 3$:    CMPB    121(R2), TAPE_TYPE[I]                  ;
                                      08   12 00048     BNEQ    4$                                             ;
                        18    AE      D4 A540  D0 0004A  MOVL    TAPE_LABEL[I], ARGLIST+24                      ; 1806
                                      04   11 00050     BRB     5$                                             ;
                    EC          50          09   F3 00052 4$:    AOBLEQ  #9, I, 3$                              ; 1805
                                      52   DD 00056 5$:    PUSHL   R2                                             ; 1808
                                04    AE   9F 00058     PUSHAB  ARGLIST                                        ;
                              0000V   CF         02   FB 0005B     CALLS   #2, COMMON_DISPLAY                     ;
                                      66   0099  C2   E9 00060     BLBC    153(R2), 9$                            ; 1813
                                      6E         06   D0 00065     MOVL    #6, ARGLIST                            ; 1816
                                04    AE   00B8  C2   9E 00068     MOVAB   184(R2), ARGLIST+4                     ; 1817
                                08    AE   00B6  C2   3C 0006E     MOVZWL  182(R2), ARGLIST+8                     ; 1818
                                0C    AE   00CE  C2   3C 00074     MOVZWL  206(R2), ARGLIST+12                    ; 1819
                                10    AE   009B  C2   3C 0007A     MOVZWL  155(R2), ARGLIST+16                    ; 1820
                              08      009A C2         06   E1 00080     BBC     #6, 154(R2), 6$                ; 1821
                                18    AE   0000' CF   9E 00086     MOVAB   P.AJT, ARGLIST+24                      ; 1822
                                      14   11 0008C     BRB     8$                                             ;
                                      009A C2   95 0008E 6$:    TSTB    154(R2)                                ; 1823
                                      08   18 00092     BGEQ    7$                                             ;
                                18    AE   0000' CF   9E 00094     MOVAB   P.AJU, ARGLIST+24                      ; 1824
                                      06   11 0009A     BRB     8$                                             ;
                                18    AE   0000' CF   9E 0009C 7$:    MOVAB   P.AJV, ARGLIST+24                  ; 1825
                                14    AE     18   BE   9A 000A2 8$:    MOVZBL  @ARGLIST+24, ARGLIST+20            ; 1826
                          14    AE          14   14   AE   C3 000A7     SUBL3   ARGLIST+20, #20, ARGLIST+20            ;
                                1C    AE   00CC  C2   3C 000AD     MOVZWL  204(R2), ARGLIST+28                    ; 1827
                                20    AE   009E  C2   9A 000B3     MOVZBL  158(R2), ARGLIST+32                    ; 1828
                          20    AE          0E   20   AE   C3 000B9     SUBL3   ARGLIST+32, #14, ARGLIST+32            ;
                                24    AE   009E  C2   9A 000BF     MOVZBL  158(R2), ARGLIST+36                    ; 1829
                                28    AE   009F  C2   9E 000C5     MOVAB   159(R2), ARGLIST+40                    ; 1830
                                      53   7C   A2   9E 000CB 9$:    MOVAB   124(R2), R3                        ; 1835
                                      1F   01   A3   93 000CF     BITB    1(R3), #31                             ;
                                      08   13 000D3     BEQL    10$                                            ;
                0C        01    A3          05         00   ED 000D5     CMPZV   #0, #5, 1(R3), #12             ; 1836
```

```
                                          07  12 000DB        BNEQ    11$                                  ;
                          50     0000'    CF  9E 000DD 10$:   MOVAB   P.AJW, R0                            ; 1837
                                          32  11 000E2        BRB     15$                                  ;
   03     01     A3       05              00  ED 000E4 11$:   CMPZV   #0, #5, 1(R3), #3                    ; 1838
                                          07  12 000EA        BNEQ    12$                                  ;
                          50     0000'    CF  9E 000EC        MOVAB   P.AJX, R0                            ; 1839
                                          23  11 000F1        BRB     15$                                  ;
   04     01     A3       05              00  ED 000F3 12$:   CMPZV   #0, #5, 1(R3), #4                    ; 1840
                                          07  12 000F9        BNEQ    13$                                  ;
                          50     0000'    CF  9E 000FB        MOVAB   P.AJY, R0                            ; 1841
                                          14  11 00100        BRB     15$                                  ;
   05     01     A3       05              00  ED 00102 13$:   CMPZV   #0, #5, 1(R3), #5                    ; 1842
                                          07  12 00108        BNEQ    14$                                  ;
                          50     0000'    CF  9E 0010A        MOVAB   P.AJZ, R0                            ; 1843
                                          05  11 0010F        BRB     15$                                  ;
                          50     0000'    CF  9E 00111 14$:   MOVAB   P.AKA, R0                            ; 1844
                    30    AE              50  D0 00116 15$:   MOVL    R0, ARGLIST+48                       ; 1835
                    2C    AE       30     BE  9A 0011A        MOVZBL  @ARGLIST+48, ARGLIST+44              ; 1845
          2C     AE       19       2C     AE  C3 0011F        SUBL3   ARGLIST+44, #25, ARGLIST+44          ;
                    F0    8F              63  93 00125        BITB    (R3), #240                           ; 1846
                                          06  12 00129        BNEQ    16$                                  ;
                    38    AE              65  D0 0012B        MOVL    FORMAT_LABEL, ARGLIST+56             ; 1847
                                          1F  11 0012F        BRB     19$                                  ;
                    38    AE     0000'    CF  9E 00131 16$:   MOVAB   P.AKB, ARGLIST+56                    ; 1850
                                          50  D4 00137        CLRL    I                                    ; 1851
                          51        FC A540  9A 00139 17$:   MOVZBL  FORMAT_TYPE[I], R1                   ; 1852
   51            63       04              04  ED 0013E        CMPZV   #4, #4, (R3), R1                     ;
                                          07  12 00143        BNEQ    18$                                  ;
                    38    AE            6540  D0 00145        MOVL    FORMAT_LABEL[I], ARGLIST+56          ; 1853
                                          04  11 0014A        BRB     19$                                  ;
                 E9       50              02  F3 0014C 18$:   AOBLEQ  #2, I, 17$                           ; 1852
                    34    AE       38     BE  9A 00150 19$:   MOVZBL  @ARGLIST+56, ARGLIST+52              ; 1855
          34     AE       21       34     AE  C3 00155        SUBL3   ARGLIST+52, #33, ARGLIST+52          ;
                          26     0099     C2  E9 0015B        BLBC    153(R2), 20$                         ; 1857
                                   20     AE  9F 00160        PUSHAB  ARGLIST+32                           ; 1866
                                 0000'    CF  9F 00163        PUSHAB  P.AKI                                ; 1864
                                   1C     AE  9F 00167        PUSHAB  ARGLIST+20                           ;
                                 0000'    CF  9F 0016A        PUSHAB  P.AKG                                ; 1862
                                   1C     AE  9F 0016E        PUSHAB  ARGLIST+12                           ;
                                 0000'    CF  9F 00171        PUSHAB  P.AKE                                ; 1860
                                   18     AE  9F 00175        PUSHAB  ARGLIST                              ;
                                 0000'    CF  9F 00178        PUSHAB  P.AKC                                ; 1858
                                   20     AE  9F 0017C        PUSHAB  ARGLIST                              ;
                                 0000'    CF  9F 0017F        PUSHAB  ASCID_NULL                           ;
                          64              0A  FB 00183        CALLS   #10, SHOW$WRITE_LINE                 ;
                                   2C     AE  9F 00186 20$:   PUSHAB  ARGLIST+44                           ; 1869
                                 0000'    CF  9F 00189        PUSHAB  P.AKK                                ; 1868
                          64              02  FB 0018D        CALLS   #2, SHOW$WRITE_LINE                  ;
                 07     0089      C2      04  E1 00190        BBC     #4, 137(R2), 21$                     ; 1871
                          50     0000'    CF  9E 00196        MOVAB   P.AKM, R0                            ; 1872
                                          05  11 0019B        BRB     22$                                  ;
                          50     0000'    CF  9E 0019D 21$:   MOVAB   P.AKN, R0                            ; 1873
                          6E              50  D0 001A2 22$:   MOVL    R0, ARGLIST                          ; 1871
                          07       02     A3  E9 001A5        BLBC    2(R3), 23$                           ; 1875
                          50     0000'    CF  9E 001A9        MOVAB   P.AKO, R0                            ; 1876
                                          05  11 001AE        BRB     24$                                  ;
                          50     0000'    CF  9E 001B0 23$:   MOVAB   P.AKP, R0                            ; 1877
```

```
                04   AE           50   D0 001B5 24$:    MOVL    R0, ARGLIST+4                   ; 1875
          07         63           11   E1 001B9         BBC     #17, (R3), 25$                  ; 1879
                     50     0000'  CF   9E 001BD         MOVAB   P.AKQ, R0                       ; 1880
                                   05   11 001C2         BRB     26$                             ;
                     50     0000'  CF   9E 001C4 25$:    MOVAB   P.AKR, R0                       ; 1881
                08   AE           50   D0 001C9 26$:    MOVL    R0, ARGLIST+8                   ; 1879
          07         63           12   E1 001CD         BBC     #18, (R3), 27$                  ; 1883
                     50     0000'  CF   9E 001D1         MOVAB   P.AKS, R0                       ; 1884
                                   05   11 001D6         BRB     28$                             ;
                     50     0000'  CF   9E 001D8 27$:    MOVAB   P.AKT, R0                       ; 1885
                0C   AE           50   D0 001DD 28$:    MOVL    R0, ARGLIST+12                  ; 1883
          07         63           13   E1 001E1         BBC     #19, (R3), 29$                  ; 1887
                     50     0000'  CF   9E 001E5         MOVAB   P.AKU, R0                       ; 1888
                                   05   11 001EA         BRB     30$                             ;
                     50     0000'  CF   9E 001EC 29$:    MOVAB   P.AKV, R0                       ; 1889
                10   AE           50   D0 001F1 30$:    MOVL    R0, ARGLIST+16                  ; 1887
          07         63           14   E1 001F5         BBC     #20, (R3), 31$                  ; 1891
                     50     0000'  CF   9E 001F9         MOVAB   P.AKW, R0                       ; 1892
                                   05   11 001FE         BRB     32$                             ;
                     50     0000'  CF   9E 00200 31$:    MOVAB   P.AKX, R0                       ; 1893
                14   AE           50   D0 00205 32$:    MOVL    R0, ARGLIST+20                  ; 1891
          07         63           03   E1 00209         BBC     #3, (R3), 33$                   ; 1895
                     50     0000'  CF   9E 0020D         MOVAB   P.AKY, R0                       ; 1896
                                   05   11 00212         BRB     34$                             ;
                     50     0000'  CF   9E 00214 33$:    MOVAB   P.AKZ, R0                       ; 1897
                18   AE           50   D0 00219 34$:    MOVL    R0, ARGLIST+24                  ; 1895
                                   5E   DD 0021D         PUSHL   SP                              ; 1903
                          0000'    CF   9F 0021F         PUSHAB  ASCID_NULL                      ;
                     64            02   FB 00223         CALLS   #2, SHOW$WRITE_LINE             ;
                                   5E   DD 00226         PUSHL   SP                              ; 1904
                          0000'    CF   9F 00228         PUSHAB  P.ALA                           ;
                     64            02   FB 0022C         CALLS   #2, SHOW$WRITE_LINE             ;
                           50      AE   9F 0022F         PUSHAB  CLUBUF                          ; 1909
                                   52   DD 00232         PUSHL   R2                              ;
                F401       CF       02   FB 00234         CALLS   #2, CLUSTER_NODES               ;
                                   52   DD 00239         PUSHL   R2                              ; 1914
                F4F6       CF       01   FB 0023B         CALLS   #1, SHOW_DEVICE_ACL             ;
                                   04   00240         RET                                     ; 1917

; Routine Size:  577 bytes,    Routine Base:  $CODE$ + 0A9D
```

```
;  1543          1918  1 GLOBAL ROUTINE display_journal (scratch, flags) : NOVALUE =
;  1544          1919  2 BEGIN
;  1545          1920  2
;  1546          1921  2 !---
;  1547          1922  2 !
;  1548          1923  2 ! This is the display routine for journals
;  1549          1924  2 !
;  1550          1925  2 ! Inputs
;  1551          1926  2 !       SCRATCH - contains data about the journal
;  1552          1927  2 !       FLAGS   - contains info on what to print
;  1553          1928  2 !
;  1554          1929  2 ! Outputs
;  1555          1930  2 !       None
;  1556          1931  2 !
;  1557          1932  2 !---
;  1558          1933  2
;  1559          1934  2 MAP
;  1560          1935  2     scratch : REF $BBLOCK,
;  1561          1936  2     flags : REF $BBLOCK;
;  1562          1937  2
;  1563          1938  2 LOCAL
;  1564          1939  2     arglist : VECTOR[20];
;  1565          1940  2
;  1566          1941  2 !
;  1567          1942  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;  1568          1943  2 ! then check to see if the device is mounted/allocated.  If not, then
;  1569          1944  2 ! don't display information on the device.
;  1570          1945  2 !
;  1571          1946  2 IF NOT check_mnt_all(.scratch, .flags)
;  1572          1947  2 THEN RETURN;
;  1573          1948  2
;  1574          1949  2 arglist[0] = cstring('Journal ');
;  1575          1950  2
;  1576          1951  2 !
;  1577          1952  2 ! Determine which type of journal it is.  If not in the list, then
;  1578          1953  2 ! simply say that it is unknown.
;  1579          1954  2 !
;  1580          1955  2 arglist[6] = unknown;                    ! Assume unknown
;  1581          1956  2 INCR i FROM 0 TO journal_number - 1 DO  ! Loop thru known journal types
;  1582          1957  3     (IF .scratch[d_b_devtype] EQLU .journal_type[.i]
;  1583          1958  2     THEN (arglist[6] = .journal_label[.i]; EXITLOOP));
;  1584          1959  2
;  1585          1960  2 scratch[d_w_devbufsiz] = .scratch[d_w_devbufsiz]/512;   ! Fiddle buffer size
;  1586          1961  2 common_display(arglist, .scratch);       ! Display stuff common to all devices
;  1587          1962  2
;  1588          1963  2
;  1589          1964  2 !
;  1590          1965  2 ! If there's more (i.e., not a control journal), continue
;  1591          1966  2 !
;  1592          1967  2 IF .scratch[d_b_cont]
;  1593          1968  2 THEN
;  1594          1969  3     BEGIN
;  1595          1970  3     arglist[0] = ucb$s_jnl_nam;                          ! The volume label may
;  1596          1971  3     arglist[1] = scratch[d_t_volnam];
;  1597          1972  3     arglist[2] = 21 - .scratch[d_t_acpnam];
;  1598          1973  3     arglist[3] = .scratch[d_t_acpnam];                   ! Process name of the ACP
;  1599          1974  3     arglist[4] = scratch[d_t_acpnam] + 1;
```

```
;1600     1975  3      arglist[6] = .mode_table[.scratch[d_b_amod]];                ! Access mode
;1601     1976  3      arglist[5] = 21 - .(.arglist[6])<0,8>;
;1602     1977  3      arglist[7] = .scratch[d_w_trans];                    ! Transaction count
;1603     1978  3      arglist[8] = .scratch[d_w_jnl_cop];                  ! Number of existing jnl copies
;1604     1979  3      arglist[9] = .scratch[d_b_jnl_avl];                  ! Number of avail. jnl copies
;1605     1980  3      arglist[10] = .scratch[d_l_jnl_segno];                       ! Current sequence number
;1606     1981  3      arglist[11] = .scratch[d_l_jnl_asid];                        ! How many assigns so far
;1607     1982  3
;1608     1983  3
;1609     1984  3      show$write_line(%ASCID '    Journal name!2< !>":AF"    ACP process name!#< !>"!AF"',
;1610     1985  3                      arglist[0],
;1611     1986  3                      %ASCID '    Access mode!#< !>!AC    Transaction count!22UW',
;1612     1987  3                      arglist[5],
;1613     1988  3                      %ASCID '    Existing jnl copies!13UW    Available jnl copies!19UB',
;1614     1989  3                      arglist[8],
;1615     1990  3                      %ASCID '    Current sequence no.!12UL    Assign seq. no.!24UL',
;1616     1991  3                      arglist[10]);
;1617     1992  3
;1618     1993  3      !
;1619     1994  3      ! If this is a disk journal, display the size.
;1620     1995  3      !
;1621     1996  3      IF .$BBLOCK[scratch[d_l_jnl_char], 0, $BITPOSITION(vcb$v_jnl_disk), 1, 0]
;1622     1997  3      THEN show$write_line(%ASCID '    Journal size!20UL',
;1623     1998  3                           %REF(.scratch[d_l_fil_mxvbn]))
;1624     1999  3      !
;1625     2000  3      ! If tape, display tape-specific stuff
;1626     2001  3      !
;1627     2002  3      ELSE
;1628     2003  4          BEGIN
;1629     2004  4          arglist[0] = .scratch[d_w_jnl_id];
;1630     2005  4          arglist[2] = scratch[d_t_grpnam];
;1631     2006  4          arglist[1] = 27 - .(.arglist[2])<0,8>;
;1632     2007  4          show$write_line(%ASCID '    Journal quota!19UL    Shadow group!#< !>!AC',
;1633     2008  4                          arglist);
;1634     2009  3          END;
;1635     2010  3
;1636     2011  3      !
;1637     2012  3      ! If this is an AT journal, display the journal mask
;1638     2013  3      !
;1639     2014  3      IF .scratch[d_b_devtype] EQLU dt$_atjnl
;1640     2015  3      THEN show$write_line(%ASCID '    Journal mask!12< !>!8XL',
;1641     2016  3                           %REF(.scratch[d_l_jnl_mask]));
;1642     2017  3
;1643     2018  3      !
;1644     2019  3      ! If an RU journal, display the quota
;1645     2020  3      !
;1646     2021  3      IF .scratch[d_b_devtyp•] EQLU dt$_rujnl
;1647     2022  3      THEN show$write_line(%ASCID '    Journal quota!19UL',
;1648     2023  3                           %REF(.scratch[d_l_jnl_quot]));
;1649     2024  3
;1650     2025  3      !
;1651     2026  3      ! Display the physical devices
;1652     2027  3      !
;1653     2028  4      BEGIN
;1654     2029  4      LOCAL
;1655     2030  4          desc : VECTOR[2],
;1656     2031  4          buffer : VECTOR[25,BYTE],
```

```
;1657     2032  4          pointer : REF VECTOR[,BYTE];
;1658     2033  4      pointer = .scratch + d_k_length;
;1659     2034  4      IF .pointer[0] NEQ 0
;1660     2035  4      THEN
;1661     2036  5          BEGIN
;1662     2037  5          show$write_line(%ASCID '    Physical device(s): !AC',
;1663     2038  5                          pointer);
;1664     2039  5          pointer = pointer[0] + .pointer[0] +1;
;1665     2040  4          END;
;1666     2041  4      WHILE .pointer[0] NEQ 0 DO
;1667     2042  5          BEGIN
;1668     2043  5          show$wri'e_line(%ASCID '!24< !>!AC',
;1669     2044  5                          pointer);
;1670     2045  5          pointer = pointer[0] + .pointer[0] + 1;
;1671     2046  4          END;
;1672     2047  3      END;
;1673     2048  3
;1674     2049  3
;1675     2050  4      arglist[0] = (IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_deadmo), 1, 0]
;1676     2051  4                    THEN cstring('delete pending, ')
;1677     2052  3                    ELSE UPLIT BYTE(0));
;1678     2053  3
;1679     2054  4      arglist[1] = (IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_jnl_cls), 1, 0]
;1680     2055  4                    THEN cstring('cluster journal, ')
;1681     2056  3                    ELSE UPLIT BYTE(0));
;1682     2057  3
;1683     2058  4      arglist[2] = (IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_jnl_slv), 1, 0]
;1684     2059  4                    THEN cstring('slave journal, ')
;1685     2060  3                    ELSE UPLIT BYTE(0));
;1686     2061  3
;1687     2062  4      arglist[3] = (IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_known_jnl), 1, 0]
;1688     2063  4                    THEN cstring('known journal, ')
;1689     2064  4                    ELSE IF .$BBLOCK[scratch[d_l_jnl_char], 0, $BITPOSITION(vcb$v_jnl_tmpfi), 1, 0]
;1690     2065  4                    THEN cstring('temporary file, ')
;1691     2066  3                    ELSE cstring('permanent file, '));
;1692     2067  3
;1693     2068  4      arglist[4] = (IF .$BBLOCK[scratch[d_w_devsts], 0, $BITPOSITION(ucb$v_perm_jnl), 1, 0]
;1694     2069  4                    THEN cstring('permanent journal.')
;1695     2070  3                    ELSE cstring('temporary journal.'));
;1696     2071  3
;1697     2072  3      !
;1698     2073  3      ! Format the data for output and print it.
;1699     2074  3      !
;1700     2075  3      show$write_line(ascid_null, arglist);
;1701     2076  3      show$write_line(%ASCID '  Attributes:  !AC!AC!AC!AC!AC', arglist);
;1702     2077  2      END;
;1703     2078  2
;1704     2079  2 !
;1705     2080  2 ! Show any access control list which might be present
;1706     2081  2 !
;1707     2082  2 show_device_acl (.scratch);
;1708     2083  2
;1709     2084  2 RETURN;
;1710     2085  1 END;
```

```
                                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                20  6C  61  6E  72  75  6F  4A  08  00FDC P.ALC:   .ASCII  <8>\Journal \
                                                    6E  77  6F  6E  6B  6E  75  07  00FE5 P.ALD:   .ASCII  <7>\unknown\
                                                                                00FED           .BLKB   3
6D  61  6E  20  6C  61  6E  72  75  6F  4A  20  20  20  20  00FF0 P.ALF:   .ASCII  \    Journal name!2< !>'"!AF"    ACP proce\
20  20  20  22  46  41  21  22  3E  21  20  3C  32  21  65  00FFF
                        65  63  6F  72  70  20  50  43  41  20  0100E
21  22  3E  21  20  3C  23  21  65  6D  61  6E  20  73  73  01018           .ASCII  \ss name!#< !>'"!AF"\<0><0>
                                        00  00  22  46  41  01027
                                                    010E003A  0102C P.ALE:   .LONG   17694778
                                                    00000000' 01030           .ADDRESS P.ALF
65  64  6F  6D  20  73  73  65  63  63  41  20  20  20  20  01034 P.ALH:   .ASCII  \    Access mode!#< !>!AC    Transaction \
72  54  20  20  20  20  43  41  21  3E  21  20  3C  23  21  01043
                        20  6E  6F  69  74  63  61  73  6E  61  01052
            00  00  57  55  32  32  21  74  6E  75  6F  63  0105C           .ASCII  \count!22UW\<0><0>
                                                    010E0032  01068 P.ALG:   .LONG   17694770
                                                    00000000' 0106C           .ADDRESS P.ALH
6E  6A  20  67  6E  69  74  73  69  78  45  20  20  20  20  01070 P.ALJ:   .ASCII  \    Existing jnl copies!13UW    Availabl\
20  20  57  55  33  31  21  73  65  69  70  6F  63  20  6C  0107F
                        6C  62  61  6C  69  61  76  41  20  20  0108E
39  31  21  73  65  69  70  6F  63  20  6C  6E  6A  20  65  01098           .ASCII  \e jnl copies!19UB\<0><0><0>
                                        00  00  00  42  55  010A7
                                                    010E0039  010AC P.ALI:   .LONG   17694777
                                                    00000000' 010B0           .ADDRESS P.ALJ
71  65  73  20  74  6E  65  72  72  75  43  20  20  20  20  010B4 P.ALL:   .ASCII  \    Current sequence no.!12UL    Assign \
20  4C  55  32  31  21  2E  6F  6E  20  65  63  6E  65  75  010C3
                        20  6E  67  69  73  73  41  20  20  20  010D2
00  00  4C  55  34  32  21  2E  6F  6E  20  2E  71  65  73  010DC           .ASCII  \seq. no.!24UL\<0><0><0>
                                                    00  010EB
                                                    010E0035  010EC P.ALK:   .LONG   17694773
                                                    00000000' 010F0           .ADDRESS P.ALL
7A  69  73  20  6C  61  6E  72  75  6F  4A  20  20  20  20  010F4 P.ALN:   .ASCII  \    Journal size!20UL\<0><0><0>
                            00  00  00  4C  55  30  32  21  65  01103
                                                    010E0015  0110C P.ALM:   .LONG   17694741
                                                    00000000' 01110           .ADDRESS P.ALN
6F  75  71  20  6C  61  6E  72  75  6F  4A  20  20  20  20  01114 P.ALP:   .ASCII  \    Journal quota!19UL    Shadow group!#\
64  61  68  53  20  20  20  20  4C  55  39  31  21  61  74  01123
                        23  21  70  75  6F  72  67  20  77  6F  01132
                                00  43  41  21  3E  21  20  3C  0113C           .ASCII  \< !>!AC\<0>
                                                    010E002F  01144 P.ALO:   .LONG   17694767
                                                    00000000' 01148           .ADDRESS P.ALP
73  61  6D  20  6C  61  6E  72  75  6F  4A  20  20  20  20  0114C P.ALR:   .ASCII  \    Journal mask!12< !>!8XL\<0>
        00  4C  58  38  21  3E  21  20  3C  32  31  21  6B  0115B
                                                    010E001B  01168 P.ALQ:   .LONG   17694747
                                                    00000000' 0116C           .ADDRESS P.ALR
6F  75  71  20  6C  61  6E  72  75  6F  4A  20  20  20  20  01170 P.ALT:   .ASCII  \    Journal quota!19UL\<0><0>
                            00  00  4C  55  39  31  21  61  74  0117F
                                                    010E0016  01188 P.ALS:   .LONG   17694742
                                                    00000000' 0118C           .ADDRESS P.ALT
65  64  20  6C  61  63  69  73  79  68  50  20  20  20  20  01190 P.ALV:   .ASCII  \    Physical device(s): !AC\<0>
        00  43  41  21  20  3A  29  73  28  65  63  69  76  0119F
                                                    010E001B  011AC P.ALU:   .LONG   17694747
                                                    00000000' 011B0           .ADDRESS P.ALV
            00  00  43  41  21  3E  21  20  3C  34  32  21  011B4 P.ALX:   .ASCII  \!24< !>!AC\<0><0>
                                                    010E000A  011C0 P.ALW:   .LONG   17694730
                                                    00000000' 011C4           .ADDRESS P.ALX
```

```
67 6E 69 64 6E 65 70 20 65 74 65 6C 65 64 10  011C8 P.ALY:  .ASCII  <16>\delete pending, \         ;
                                        20 2C  011D7                                              ;
                                           00  011D9 P.ALZ:  .BYTE   0                            ;
61 6E 72 75 6F 6A 20 72 65 74 73 75 6C 63 11  011DA P.AMA:  .ASCII  <17>\cluster journal, \        ;
                                     20 2C 6C  011E9                                              ;
                                           00  011EC P.AMB:  .BYTE   0                            ;
2C 6C 61 6E 72 75 6F 6A 20 65 76 61 6C 73 0F  011ED P.AMC:  .ASCII  <15>\slave journal, \          ;
                                           20  011FC                                              ;
                                           00  011FD P.AMD:  .BYTE   0                            ;
2C 6C 61 6E 72 75 6F 6A 20 6E 77 6F 6E 6B 0F  011FE P.AME:  .ASCII  <15>\known journal, \          ;
                                           20  0120D                                              ;
65 6C 69 66 20 79 72 61 72 6F 70 6D 65 74 10  0120E P.AMF:  .ASCII  <16>\temporary file, \         ;
                                        20 2C  0121D                                              ;
65 6C 69 66 20 74 6E 65 6E 61 6D 72 65 70 10  0121F P.AMG:  .ASCII  <16>\permanent file, \         ;
                                        20 2C  0122E                                              ;
72 75 6F 6A 20 74 6E 65 6E 61 6D 72 65 70 12  01230 P.AMH:  .ASCII  <18>\permanent journal.\       ;
                                  2E 6C 61 6E  0123F                                              ;
72 75 6F 6A 20 79 72 61 72 6F 70 6D 65 74 12  01243 P.AMI:  .ASCII  <18>\temporary journal.\       ;
                                  2E 6C 61 6E  01252                                              ;
                                               01256         .BLKB   2                            ;
20 20 3A 73 65 74 75 62 69 72 74 74 41 20 20  01258 P.AMK:  .ASCII  \  Attributes:  !AC!AC!AC!AC!AC\<0><0>  ;
43 41 21 43 41 21 43 41 21 43 41 21 43 41 21  01267                                              ;
                                        00 00  01276                                              ;
                                     010E001E  01278 P.AMJ:  .LONG   17694750                     ;
                                    00000000'  0127C         .ADDRESS P.AMK                       ;

                                                              .PSECT  $CODE$,NOWRT,2

                                         000C 00000          .ENTRY  DISPLAY_JOURNAL, Save R2,R3  ; 1918
                53 00000000G        00   9E 00002          MOVAB   SHOW$WRITE_LINE, R3          ;
                5E          84      AE   9E 00009          MOVAB   -124(SP), SP                 ;
                            08      AC   DD 0000D          PUSHL   FLAGS                        ; 1946
                52          04      AC   D0 00010          MOVL    SCRATCH, R2                  ;
                                    52   DD 00014          PUSHL   R2                           ;
        F3B4    CF                  02   FB 00016          CALLS   #2, CHECK_MNT_ALL            ;
                01                  50   E8 0001B          BLBS    R0, 1$                       ;
                                         04 0001E          RET                                  ;
          2C    AE        0000'     CF   9E 0001F 1$:      MOVAB   P.ALC, ARGLIST               ; 1949
          44    AE        0000'     CF   9E 00025          MOVAB   P.ALD, ARGLIST+24            ; 1955
                                    50   D4 0002B          CLRL    I                            ; 1957
     0000'CF40              79      A2   91 0002D 2$:      CMPB    121(R2), JOURNAL_TYPE[I]     ;
                                    09   12 00034          BNEQ    3$                           ;
          44    AE        0000'CF40      D0 00036          MOVL    JOURNAL_LABEL[I], ARGLIST+24 ; 1958
                                    04   11 0003D          BRB     4$                           ;
    EA          50                  05   F3 0003F 3$:      AOBLEQ  #5, I, 2$                    ; 1957
                50          7A      A2   3C 00043 4$:      MOVZWL  122(R2), R0                  ; 1960
                50 00000200         8F   C6 00047          DIVL2   #512, R0                     ;
          7A    A2                  50   B0 0004E          MOVW    R0, 122(R2)                  ;
                                    52   DD 00052          PUSHL   R2                           ; 1961
                            30      AE   9F 00054          PUSHAB  ARGLIST                      ;
        0000V   CF                  02   FB 00057          CALLS   #2, COMMON_DISPLAY           ;
                03        0099      C2   E8 0005C          BLBS    153(R2), 5$                  ; 1967
                                  0187   31 00061          BRW     24$                          ;
          2C    AE                  12   D0 00064 5$:      MOVL    #18, ARGLIST                 ; 1970
          30    AE        00B8      C2   9E 00068          MOVAB   184(R2), ARGLIST+4           ; 1971
```

```
                34   AE      009E   C2  9A 0006E          MOVZBL  158(R2), ARGLIST+8                          ; 1972
        34   AE      15        34   AE  C3 00074          SUBL3   ARGLIST+8, #21, ARGLIST+8                   ;
                38   AE      009E   C2  9A 0007A          MOVZBL  158(R2), ARGLIST+12                         ; 1973
                3C   AE      009F   C2  9E 00080          MOVAB   159(R2), ARGLIST+16                         ; 1974
                50      0100   C2  9A 00086          MOVZBL  256(R2), R0                                 ; 1975
                44   AE    0000'CF40  D0 0008B          MOVL    MODE_TABLE[R0], ARGLIST+24                  ;
                40   AE        44   BE  9A 00092          MOVZBL  @ARGLIST+24, ARGLIST+20                     ; 1976
        40   AE      15        40   AE  C3 00097          SUBL3   ARGLIST+20, #21, ARGLIST+20                 ;
                48   AE      009B   C2  3C 0009D          MOVZWL  155(R2), ARGLIST+28                         ; 1977
                4C   AE      0101   C2  3C 000A3          MOVZWL  257(R2), ARGLIST+32                         ; 1978
                50   AE      0104   C2  9A 000A9          MOVZBL  260(R2), ARGLIST+36                         ; 1979
                54   AE      00E8   C2  7D 000AF          MOVQ    232(R2), ARGLIST+40                         ; 1980
                          54   AE  9F 000B5          PUSHAB  ARGLIST+40                                  ; 1991
                        0000'  CF  9F 000B8          PUSHAB  P.ALK                                       ; 1989
                          54   AE  9F 000BC          PUSHAB  ARGLIST+32                                  ;
                        0000'  CF  9F 000BF          PUSHAB  P.ALI                                       ; 1987
                          50   AE  9F 000C3          PUSHAB  ARGLIST+20                                  ;
                        0000'  CF  9F 000C6          PUSHAB  P.ALG                                       ; 1985
                          44   AE  9F 000CA          PUSHAB  ARGLIST                                     ;
                        0000'  CF  9F 000CD          PUSHAB  P.ALE                                       ; 1984
                      63           08  FB 000D1          CALLS   #8, SHOW$WRITE_LINE                         ;
                      0D    00E0   C2  E9 000D4          BLBC    224(R2), 6$                                 ; 1996
                      6E    00DC   C2  D0 000D9          MOVL    220(R2), (SP)                               ; 1998
                                5E   DD 000DE          PUSHL   SP                                          ;
                        0000'  CF  9F 000E0          PUSHAB  P.ALM                                       ; 1997
                                1E   11 000E4          BRB     7$                                          ;
                2C   AE      00FC   C2  3C 000E6 6$:     MOVZWL  252(R2), ARGLIST                            ; 2004
                34   AE      00CE   C2  9E 000EC          MOVAB   206(R2), ARGLIST+8                          ; 2005
                30   AE        34   BE  9A 000F2          MOVZBL  @ARGLIST+8, ARGLIST+4                       ; 2006
        30   AE      1B        30   AE  C3 000F7          SUBL3   ARGLIST+4, #27, ARGLIST+4                   ;
                          2C   AE  9F 000FD          PUSHAB  ARGLIST                                     ; 2007
                        0000'  CF  9F 00100          PUSHAB  P.ALO                                       ;
                      63           02  FB 00104 7$:     CALLS   #2, SHOW$WRITE_LINE                         ;
                      04     79   A2  91 00107          CMPB    121(R2), #4                                 ; 2014
                                0E   12 0010B          BNEQ    8$                                          ;
                      6E    00E4   C2  D0 0010D          MOVL    228(R2), (SP)                               ; 2016
                                5E   DD 00112          PUSHL   SP                                          ;
                        0000'  CF  9F 00114          PUSHAB  P.ALQ                                       ; 2015
                      63           02  FB 00118          CALLS   #2, SHOW$WRITE_LINE                         ;
                      01     79   A2  91 0011B 8$:     CMPB    121(R2), #1                                 ; 2021
                                0E   12 0011F          BNEQ    9$                                          ;
                      6E    00F0   C2  D0 00121          MOVL    240(R2), (SP)                               ; 2023
                                5E   DD 00126          PUSHL   SP                                          ;
                        0000'  CF  9F 00128          PUSHAB  P.ALS                                       ; 2022
                      63           02  FB 0012C          CALLS   #2, SHOW$WRITE_LINE                         ;
        04   AE      0107   C2  9E 0012F 9$:     MOVAB   263(R2), POINTER                            ; 2033
                          04   BE  95 00135          TSTB    @POINTER                                    ; 2034
                                17   13 00138          BEQL    11$                                         ;
                          04   AE  9F 0013A          PUSHAB  POINTER                                     ; 2037
                        0000'  CF  9F 0013D          PUSHAB  P.ALU                                       ;
                      63           02  FB 00141 10$:     CALLS   #2, SHOW$WRITE_LINE                         ;
                      50     04   BE  9A 00144          MOVZBL  @POINTER, R0                                ; 2039
                      50     04   AE  C0 00148          ADDL2   POINTER, R0                                 ;
        04   AE      01     A0   9E 0014C          MOVAB   1(R0), POINTER                              ;
                          04   BE  95 00151 11$:     TSTB    @POINTER                                    ; 2041
                                09   13 00154          BEQL    12$                                         ;
                          04   AE  9F 00156          PUSHAB  POINTER                                     ; 2043
```

```
                        0000'  CF  9F 00159          PUSHAB  P.ALW                                   ;
                               E2  11 0015D          BRB     10$                                     ;
07      0089    C2             02  E1 0015F 12$:     BBC     #2, 137(R2), 13$                        ; 2050
                50      0000'  CF  9E 00165          MOVAB   P.ALY, R0                               ; 2051
                               05  11 0016A          BRB     14$                                     ;
                50      0000'  CF  9E 0016C 13$:     MOVAB   P.ALZ, R0                               ; 2052
              2C    AE       50  D0 00171 14$:     MOVL    R0, ARGLIST                             ; 2050
                51      0090   C2  9E 00175          MOVAB   144(R2), R1                             ; 2054
07              61             06  E1 0017A          BBC     #6, (R1), 15$                           ;
                50      0000'  CF  9E 0017E          MOVAB   P.AMA, R0                               ; 2055
                               05  11 00183          BRB     16$                                     ;
                50      0000'  CF  9E 00185 15$:     MOVAB   P.AMB, R0                               ; 2056
              30    AE       50  D0 0018A 16$:     MOVL    R0, ARGLIST+4                           ; 2054
                               61  95 0018E          TSTB    (R1)                                    ; 2058
                               07  18 00190          BGEQ    17$                                     ;
                50      0000'  CF  9E 00192          MOVAB   P.AMC, R0                               ; 2059
                               05  11 00197          BRB     18$                                     ;
                50      0000'  CF  9E 00199 17$:     MOVAB   P.AMD, R0                               ; 2060
              34    AE       50  D0 0019E 18$:     MOVL    R0, ARGLIST+8                           ; 2058
07              61             05  E1 001A2          BBC     #5, (R1), 19$                           ; 2062
                50      0000'  CF  9E 001A6          MOVAB   P.AME, R0                               ; 2063
                               12  11 001AB          BRB     21$                                     ;
07      00E0    C2             02  E1 001AD 19$:     BBC     #2, 224(R2), 20$                        ; 2064
                50      0000'  CF  9E 001B3          MOVAB   P.AMF, R0                               ; 2065
                               05  11 001B8          BRB     21$                                     ;
                50      0000'  CF  9E 001BA 20$:     MOVAB   P.AMG, R0                               ; 2066
              38    AE       50  D0 001BF 21$:     MOVL    R0, ARGLIST+12                          ; 2062
07              61             04  E1 001C3          BBC     #4, (R1), 22$                           ; 2068
                50      0000'  CF  9E 001C7          MOVAB   P.AMH, R0                               ; 2069
                               05  11 001CC          BRB     23$                                     ;
                50      0000'  CF  9E 001CE 22$:     MOVAB   P.AMI, R0                               ; 2070
              3C    AE       50  D0 001D3 23$:     MOVL    R0, ARGLIST+16                          ; 2068
                        2C     AE  9F 001D7          PUSHAB  ARGLIST                                 ; 2075
                        0000'  CF  9F 001DA          PUSHAB  ASCID_NULL                              ;
                63             02  FB 001DE          CALLS   #2, SHOW$WRITE_LINE                     ;
                        2C     AE  9F 001E1          PUSHAB  ARGLIST                                 ; 2076
                        0000'  CF  9F 001E4          PUSHAB  P.AMJ                                   ;
                63             02  FB 001E8          CALLS   #2, SHOW$WRITE_LINE                     ;
                               52  DD 001EB 24$:     PUSHL   R2                                      ; 2082
            F303    CF         01  FB 001ED          CALLS   #1, SHOW_DEVICE_ACL                     ;
                               04  001F2            RET                                             ; 2085

; Routine Size:  499 bytes,    Routine Base:  $CODE$ + 0CDE
```

```
;  1712          2086  1 GLOBAL ROUTINE display_printer (scratch, flags) : NOVALUE =
;  1713          2087  2 BEGIN
;  1714          2088  2
;  1715          2089  2 !---
;  1716          2090  2 !
;  1717          2091  2 ! This is the display routine for printers.
;  1718          2092  2 !
;  1719          2093  2 ! Inputs
;  1720          2094  2 !       SCRATCH - contains data about the printer device
;  1721          2095  2 !       FLAGS   - contains info on what to print
;  1722          2096  2 !
;  1723          2097  2 ! Outputs
;  1724          2098  2 !       None
;  1725          2099  2 !
;  1726          2100  2 !---
;  1727          2101  2
;  1728          2102  2 MAP
;  1729          2103  2     scratch : REF $BBLOCK,
;  1730          2104  2     flags : REF $BBLOCK;
;  1731          2105  2
;  1732          2106  2 LOCAL
;  1733          2107  2     arglist : VECTOR[20];
;  1734          2108  2
;  1735          2109  2 !
;  1736          2110  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;  1737          2111  2 ! then check to see if the device is mounted/allocated.  If not, then
;  1738          2112  2 ! don't display information on the device.
;  1739          2113  2 !
;  1740          2114  2 IF NOT check_mnt_all(.scratch, .flags)
;  1741          2115  2 THEN RETURN;
;  1742          2116  2
;  1743          2117  2 !
;  1744          2118  2 ! Set up the argument list for output.
;  1745          2119  2 !
;  1746          2120  2 arglist[0] = cstring('Printer ');
;  1747          2121  2
;  1748          2122  2 !
;  1749          2123  2 ! Determine which type of line printer it is.  If not in the list, then
;  1750          2124  2 ! simply say that it is unknown.
;  1751          2125  2 !
;  1752          2126  2 arglist[6] = unknown;
;  1753          2127  2 INCR i FROM 0 TO printer_number - 1
;  1754          2128  3 DO (IF .scratch[d_b_devtype] EQLU .printer_type[.i]
;  1755          2129  2     THEN (arglist[6] = .printer_label[.i]; EXITLOOP));
;  1756          2130  2
;  1757          2131  2 common_display(arglist, .scratch);              ! Print the common heading
;  1758          2132  2
;  1759          2133  2 !
;  1760          2134  2 ! Insert the page width and length
;  1761          2135  2 !
;  1762          2136  2 arglist[0] = .scratch[d_w_devbufsiz];                          ! Page width
;  1763          2137  2 arglist[1] = .$BBLOCK[scratch[d_l_devdepend], lp$v_page_l];    ! Page length
;  1764          2138  2
;  1765          2139  2 !
;  1766          2140  2 ! Check for the specific printer characteristics
;  1767          2141  2 !
;  1768          2142  3 arglist[2] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_cr]
```

```
;1769        2143  3                THEN UPLIT BYTE (0)
;1770        2144  2                ELSE cstring('No '));
;1771        2145  2 arglist[3] = cstring('Carriage_return');
;1772        2146  2
;1773        2147  3 arglist[4] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_mechform]
;1774        2148  3                THEN UPLIT BYTE (0)
;1775        2149  2                ELSE cstring('No '));
;1776        2150  2 arglist[5] = cstring('Formfeed');
;1777        2151  2
;1778        2152  3 arglist[6] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_lower]
;1779        2153  3                THEN cstring('Lowercase')
;1780        2154  2                ELSE cstring('Uppercase'));
;1781        2155  2
;1782        2156  3 arglist[7] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_passall]
;1783        2157  3                THEN UPLIT BYTE (0)
;1784        2158  2                ELSE cstring('No '));
;1785        2159  2 arglist[8] = cstring('Passall');
;1786        2160  2
;1787        2161  3 arglist[9] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_wrap]
;1788        2162  3                 THEN UPLIT BYTE (0)
;1789        2163  2                 ELSE cstring ('No '));
;1790        2164  2 arglist[10] = cstring('Wrap');
;1791        2165  2
;1792        2166  3 arglist[11] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_printall]
;1793        2167  3                 THEN UPLIT BYTE (0)
;1794        2168  2                 ELSE cstring('No '));
;1795        2169  2 arglist[12] = cstring('Printall );
;1796        2170  2
;1797        2171  3 arglist[13] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_fallback]
;1798        2172  3                 THEN UPLIT BYTE (0)
;1799        2173  2                 ELSE cstring('No '));
;1800        2174  2 arglist[14] = cstring('Fallback');
;1801        2175  2
;1802        2176  3 arglist[15] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_tab]
;1803        2177  3                 THEN UPLIT BYTE (0)
;1804        2178  2                 ELSE cstring('No '));
;1805        2179  2 arglist[16] = cstring('Tab');
;1806        2180  2
;1807        2181  3 arglist[17] = (IF .$BBLOCK[scratch[d_l_devdepend], lp$v_truncate]
;1808        2182  3                 THEN UPLIT BYTE (0)
;1809        2183  2                 ELSE cstring('No '));
;1810        2184  2 arglist[18] = cstring('Truncate');
;1811        2185  2
;1812        2186  2 !
;1813        2187  2 ! Format the data for output and print it.
;1814        2188  2 !
;1815        2189  2 show$write_line(%ASCID '    Page width!22UW    Page Length!28UB',
;1816        2190  2                  arglist[0],
;1817        2191  2                  %ASCID '    !AC!AC!_!AC!AC!_!AC',
;1818        2192  2                  arglist[2],
;1819        2193  2                  %ASCID '    !AC!AC!_!_!AC!AC!_!_!AC!AC',
;1820        2194  2                  arglist[7],
;1821        2195  2                  %ASCID '    !AC!AC!_!_!AC!AC!_!_!AC!AC',
;1822        2196  2                  arglist[13]);
;1823        2197  2
;1824        2198  2 !
;1825        2199  2 ! If the device is spooled, say so.
```

```
; 1826          2200  2 !
; 1827          2201  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_spl]
; 1828          2202  2 THEN
; 1829          2203  3     BEGIN
; 1830          2204  3     arglist[0] = .scratch[d_l_intlen] - 1;
; 1831          2205  3     arglist[1] = scratch[d_t_intdev] + 1;
; 1832          2206  3     arglist[2] = scratch[d_t_qname];
; 1833          2207  3     show$write_line(%ASCID '    Intermediate device: !AF', arglist,
; 1834          2208  3                     %ASCID '    Associated queue:    !AC', arglist[2]);
; 1835          2209  3     END;
; 1836          2210  2
; 1837          2211  2 !
; 1838          2212  2 ! Show any access control list which might be present
; 1839          2213  2 !
; 1840          2214  2 show_device_acl (.scratch);
; 1841          2215  2
; 1842          2216  2 RETURN;
; 1843          2217  1 END;
```

```
                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                20  72  65  74  6E  69  72  50  08  01280 P.AML:  .ASCII  <8>\Printer \
                                    6E  77  6F  6E  6B  6E  75  07  01289 P.AMM:  .ASCII  <7>\unknown\
                                                                00  01291 P.AMN:  .BYTE   0
                                                        20  6F  4E  03  01292 P.AMO:  .ASCII  <3>\No \
72  75  74  65  72  5F  65  67  61  69  72  72  61  43  0F  01296 P.AMP:  .ASCII  <15>\Carriage_return\
                                                                6E  012A5
                                                                00  012A6 P.AMQ:  .BYTE   0
                                                        20  6F  4E  03  012A7 P.AMR:  .ASCII  <3>\No \
                                64  65  65  66  6D  72  6F  46  08  012AB P.AMS:  .ASCII  <8>\Formfeed\
                            65  73  61  63  72  65  77  6F  4C  09  012B4 P.AMT:  .ASCII  <9>\Lowercase\
                            65  73  61  63  72  65  70  70  55  09  012BE P.AMU:  .ASCII  <9>\Uppercase\
                                                                00  012C8 P.AMV:  .BYTE   0
                                                        20  6F  4E  03  012C9 P.AMW:  .ASCII  <3>\No \
                                    6C  6C  61  73  73  61  50  07  012CD P.AMX:  .ASCII  <7>\Passall\
                                                                00  012D5 P.AMY:  .BYTE   0
                                                        20  6F  4E  03  012D6 P.AMZ:  .ASCII  <3>\No \
                                                    70  61  72  57  04  012DA P.ANA:  .ASCII  <4>\Wrap\
                                                                00  012DF P.ANB:  .BYTE   0
                                                        20  6F  4E  03  012E0 P.ANC:  .ASCII  <3>\No \
                                6C  6C  61  74  6E  69  72  50  08  012E4 P.AND:  .ASCII  <8>\Printall\
                                                                00  012ED P.ANE:  .BYTE   0
                                                        20  6F  4E  03  012EE P.ANF:  .ASCII  <3>\No \
                                6B  63  61  62  6C  6C  61  46  08  012F2 P.ANG:  .ASCII  <8>\Fallback\
                                                                00  012FB P.ANH:  .BYTE   0
                                                        20  6F  4E  03  012FC P.ANI:  .ASCII  <3>\No \
                                                        62  61  54  03  01300 P.ANJ:  .ASCII  <3>\Tab\
                                                                00  01304 P.ANK:  .BYTE   0
                                                        20  6F  4E  03  01305 P.ANL:  .ASCII  <3>\No \
                                65  74  61  63  6E  75  72  54  08  01309 P.ANM:  .ASCII  <8>\Truncate\
                                                                    01312          .BLKB   2
21  68  74  64  69  77  20  65  67  61  50  20  20  20  20  01314 P.ANO:  .ASCII  \     Page width!22UW    Page Length!28UB-
65  4C  20  65  67  61  50  20  20  20  20  57  55  32  32  01323                  \<0>
                        00  42  55  38  32  21  68  74  67  6E  01332
                                                    010E0027  0133C P.ANN:  .LONG   17694759
```

```
                                              00000000' 01340          .ADDRESS P.ANO                                  ;
43 41 21 5F 21 43 41 21 43 41 21 20 20 20 20   01344 P.ANQ:   .ASCII  \    !AC!AC!_!AC!AC!_!AC\<0>              ;
               00 43 41 21 5F 21 43 41 21   01353                                                                  ;
                                              010E0017  0135C P.ANP:   .LONG   17694743                              ;
                                              00000000' 01360          .ADDRESS P.ANQ                                  ;
21 5F 21 5F 21 43 41 21 43 41 21 20 20 20 20   01364 P.ANS:   .ASCII  \    !AC!AC!_!_!AC!AC!_!_!AC!AC\<0><0>    ;
43 41 21 43 41 21 5F 21 5F 21 43 41 21 43 41   01373                                                                  ;
                                         00 00  01382                                                                  ;
                                              010E001E  01384 P.ANR:   .LONG   17694750                              ;
                                              00000000' 01388          .ADDRESS P.ANS                                  ;
21 5F 21 5F 21 43 41 21 43 41 21 20 20 20 20   0138C P.ANU:   .ASCII  \    !AC!AC!_!_!AC!AC!_!_!AC!AC\<0><0>    ;
43 41 21 43 41 21 5F 21 5F 21 43 41 21 43 41   0139B                                                                  ;
                                         00 00  013AA                                                                  ;
                                              010E001E  013AC P.ANT:   .LONG   17694750                              ;
                                              00000000' 013B0          .ADDRESS P.ANU                                  ;
74 61 69 64 65 6D 72 65 74 6E 49 20 20 20 20   013B4 P.ANW:   .ASCII  \    Intermediate device: !AF\            ;
      46 41 21 20 3A 65 63 69 76 65 64 20 65   013C3                                                                  ;
                                              010E001C  013D0 P.ANV:   .LONG   17694748                              ;
                                              00000000' 013D4          .ADDRESS P.ANW                                  ;
20 64 65 74 61 69 63 6F 73 73 41 20 20 20 20   013D8 P.ANY:   .ASCII  \    Associated queue:     !AC\           ;
      43 41 21 20 20 20 20 3A 65 75 65 75 71   013E7                                                                  ;
                                              010E001C  013F4 P.ANX:   .LONG   17694748                              ;
                                              00000000' 013F8          .ADDRESS P.ANY                                  ;


                                                               .PSECT  $CODE$,NOWRT,2

                                          001C 00000           .ENTRY  DISPLAY_PRINTER, Save R2,R3,R4           ; 2086
                 54 00000000G  00  9E 00002           MOVAB   SHOW$WRITE_LINE, R4                              ;
                 53     0000'  CF  9E 00009           MOVAB   P.AML, R3                                        ;
                 5E        B0  AE  9E 0000E           MOVAB   -80(SP), SP                                      ;
                           08  AC  DD 00012           PUSHL   FLAGS                                            ; 2114
                 52        04  AC  D0 00015           MOVL    SCRATCH, R2                                      ;
                               52  DD 00019           PUSHL   R2                                               ;
            F1BC CF            02  FB 0001B           CALLS   #2, CHECK_MNT_ALL                                ;
                 01            50  E8 00020           BLBS    R0, 1$                                           ;
                               04  00023              RET                                                      ;
                 6E            63  9E 00024 1$:       MOVAB   P.AML, ARGLIST                                   ; 2120
            18   AE        09  A3  9E 00027           MOVAB   P.AMM, ARGLIST+24                                ; 2126
                               50  D4 0002C           CLRL    I                                                ; 2128
       0000'CF40           79  A2  91 0002E 2$:       CMPB    121(R2), PRINTER_TYPE[I]                         ;
                               09  12 00035           BNEQ    3$                                               ;
            18   AE    0000'CF40  D0 00037           MOVL    PRINTER_LABEL[I], ARGLIST+24                     ; 2129
                               04  11 0003E           BRB     4$                                               ;
       EA        50            02  F3 00040 3$:       AOBLEQ  #2, I, 2$                                        ; 2128
                               52  DD 00044 4$:       PUSHL   R2                                               ; 2131
                           04  AE  9F 00046           PUSHAB  ARGLIST                                          ;
            0000V CF           02  FB 00049           CALLS   #2, COMMON_DISPLAY                               ;
                 6E        7A  A2  3C 0004E           MOVZWL  122(R2), ARGLIST                                 ; 2136
                 50        7C  A2  9E 00052           MOVAB   124(R2), R0                                      ; 2137
            04   AE        03  A0  9A 00056           MOVZBL  3(R0), ARGLIST+4                                 ;
                 06            60  E9 0005B           BLBC    (R0), 5$                                         ; 2142
                 51        11  A3  9E 0005E           MOVAB   P.AMN, R1                                        ; 2143
                               04  11 00062           BRB     6$                                               ;
                 51        12  A3  9E 00064 5$:       MOVAB   P.AMO, R1                                        ; 2144
            08   AE            51  D0 00068 6$:       MOVL    R1, ARGLIST+8                                    ; 2142
```

```
                0C  AE      16  A3  9E 0006C          MOVAB   P.AMP, ARGLIST+12                ; 2145
            06      60      01  E1  00071            BBC     #1, (R0), 7$                    ; 2147
                    51      26  A3  9E 00075          MOVAB   P.AMQ, R1                       ; 2148
                            04  11  00079            BRB     8$                              ;
                    51      27  A3  9E 0007B 7$:      MOVAB   P.AMR, R1                       ; 2149
                10  AE          51  D0  0007F 8$:      MOVL    R1, ARGLIST+16                  ; 2147
                14  AE      2B  A3  9E 00083          MOVAB   P.AMS, ARGLIST+20               ; 2150
                            60  95  00088            TSTB    (R0)                            ; 2152
                            06  18  0008A            BGEQ    9$                              ;
                    51      34  A3  9E 0008C          MOVAB   P.AMT, R1                       ; 2153
                            04  11  00090            BRB     10$                             ;
                    51      3E  A3  9E 00092 9$:      MOVAB   P.AMU, R1                       ; 2154
                18  AE          51  D0  00096 10$:     MOVL    R1, ARGLIST+24                  ; 2152
                    06      01  A0  E9 0009A          BLBC    1(R0), 11$                      ; 2156
                    51      48  A3  9E 0009E          MOVAB   P.AMV, R1                       ; 2157
                            04  11  000A2            BRB     12$                             ;
                    51      49  A3  9E 000A4 11$:     MOVAB   P.AMW, R1                       ; 2158
                1C  AE          51  D0  000A8 12$:     MOVL    R1, ARGLIST+28                  ; 2156
                20  AE      4D  A3  9E 000AC          MOVAB   P.AMX, ARGLIST+32               ; 2159
            06      60      04  E1  000B1            BBC     #4, (R0), 13$                   ; 2161
                    51      55  A3  9E 000B5          MOVAB   P.AMY, R1                       ; 2162
                            04  11  000B9            BRB     14$                             ;
                    51      56  A3  9E 000BB 13$:     MOVAB   P.AMZ, R1                       ; 2163
                24  AE          51  D0  000BF 14$:     MOVL    R1, ARGLIST+36                  ; 2161
                28  AE      5A  A3  9E 000C3          MOVAB   P.ANA, ARGLIST+40               ; 2164
            06      60      02  E1  000C8            BBC     #2, (R0), 15$                   ; 2166
                    51      5F  A3  9E 000CC          MOVAB   P.ANB, R1                       ; 2167
                            04  11  000D0            BRB     16$                             ;
                    51      60  A3  9E 000D2 15$:     MOVAB   P.ANC, R1                       ; 2168
                2C  AE          51  D0  000D6 16$:     MOVL    R1, ARGLIST+44                  ; 2166
                30  AE      64  A3  9E 000DA          MOVAB   P.AND, ARGLIST+48               ; 2169
            06      60      09  E1  000DF            BBC     #9, (R0), 17$                   ; 2171
                    51      6D  A3  9E 000E3          MOVAB   P.ANE, R1                       ; 2172
                            04  11  000E7            BRB     18$                             ;
                    51      6E  A3  9E 000E9 17$:     MOVAB   P.ANF, R1                       ; 2173
                34  AE          51  D0  000ED 18$:     MOVL    R1, ARGLIST+52                  ; 2171
                38  AE      72  A3  9E 000F1          MOVAB   P.ANG, ARGLIST+56               ; 2174
            06      60      05  E1  000F6            BBC     #5, (R0), 19$                   ; 2176
                    51      7B  A3  9E 000FA          MOVAB   P.ANH, R1                       ; 2177
                            04  11  000FE            BRB     20$                             ;
                    51      7C  A3  9E 00100 19$:     MOVAB   P.ANI, R1                       ; 2178
                3C  AE          51  D0  00104 20$:     MOVL    R1, ARGLIST+60                  ; 2176
                40  AE  0080    C3  9E 00108          MOVAB   P.ANJ, ARGLIST+64               ; 2179
            07      60      06  E1  0010E            BBC     #6, (R0), 21$                   ; 2181
                    50  0084    C3  9E 00112          MOVAB   P.ANK, R0                       ; 2182
                            05  11  00117            BRB     22$                             ;
                    50  0085    C3  9E 00119 21$:     MOVAB   P.ANL, R0                       ; 2183
                44  AE          50  D0  0011E 22$:     MOVL    R0, ARGLIST+68                  ; 2181
                48  AE  0089    C3  9E 00122          MOVAB   P.ANM, ARGLIST+72               ; 2184
                        34  AE  9F  00128            PUSHAB  ARGLIST+52                      ; 2196
                    012C    C3  9F  0012B            PUSHAB  P.ANT                           ; 2194
                        24  AE  9F  0012F            PUSHAB  ARGLIST+28                      ;
                    0104    C3  9F  00132            PUSHAB  P.ANR                           ; 2192
                        18  AE  9F  00136            PUSHAB  ARGLIST+8                       ;
                    00DC    C3  9F  00139            PUSHAB  P.ANP                           ; 2190
                        18  AE  9F  0013D            PUSHAB  ARGLIST                         ;
                    00BC    C3  9F  00140            PUSHAB  P.ANN                           ; 2189
```

```
                        64              08  FB 00144          CALLS   #8, SHOW$WRITE_LINE                    ;
                23      70      A2      06  E1 00147          BBC     #6, 112(R2), 23$                       ; 2201
                6E    00AE      C2      01  C3 0014C          SUBL3   #1, 174(R2), ARGLIST                   ; 2204
                        04      AE  009B  C2  9E 00152          MOVAB   155(R2), ARGLIST+4                     ; 2205
                        08      AE  00B2  C2  9E 00158          MOVAB   178(R2), ARGLIST+8                     ; 2206
                                    08    AE  9F 0015E          PUSHAB  ARGLIST+8                              ; 2208
                                  0174    C3  9F 00161          PUSHAB  P.ANX                                  ; 2207
                                    08    AE  9F 00165          PUSHAB  ARGLIST                                ;
                                  0150    C3  9F 00168          PUSHAB  P.ANV                                  ;
                                        64      04  FB 0016C          CALLS   #4, SHOW$WRITE_LINE            ;
                                                52  DD 0016F 23$:     PUSHL   R2                             ; 2214
                        F18C    CF              01  FB 00171          CALLS   #1, SHOW_DEVICE_ACL            ;
                                                    04 00176          RET                                    ; 2217

; Routine Size:  375 bytes,    Routine Base:  $CODE$ + 0ED1
```

```
;  1845          2218  1 GLOBAL ROUTINE display_terminal (scratch, flags) : NOVALUE =
;  1846          2219  2 BEGIN
;  1847          2220  2
;  1848          2221  2 !---
;  1849          2222  2 !
;  1850          2223  2 ! This is the display routine for terminals
;  1851          2224  2 !
;  1852          2225  2 ! Inputs
;  1853          2226  2 !       SCRATCH - contains data about the terminal
;  1854          2227  2 !       FLAGS   - contains info on what to print
;  1855          2228  2 !
;  1856          2229  2 ! Outputs
;  1857          2230  2 !       None
;  1858          2231  2 !
;  1859          2232  2 !---
;  1860          2233  2
;  1861          2234  2 MAP
;  1862          2235  2     scratch : REF $BBLOCK,
;  1863          2236  2     flags : REF $BBLOCK;
;  1864          2237  2
;  1865          2238  2 LOCAL
;  1866          2239  2     arglist : VECTOR[20];
;  1867          2240  2
;  1868          2241  2 !
;  1869          2242  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;  1870          2243  2 ! then check to see if the device is mounted/allocated.  If not, then
;  1871          2244  2 ! don't display information on the device.
;  1872          2245  2 !
;  1873          2246  2 IF NOT check_mnt_all(.scratch, .flags)  .
;  1874          2247  2 THEN RETURN;
;  1875          2248  2
;  1876          2249  2 !
;  1877          2250  2 ! Set up the argument list for output.
;  1878          2251  2 !
;  1879          2252  2 arglist[0] = cstring('Terminal ');
;  1880          2253  2
;  1881          2254  2 !
;  1882          2255  2 ! Determine which type of terminal it is.  If not in the list, then
;  1883          2256  2 ! simply say that it is unknown.
;  1884          2257  2 !
;  1885          2258  2 arglist[6] = unknown;
;  1886          2259  2 INCR i FROM 0 TO terminal_number - 1
;  1887          2260  3 DO (IF .scratch[d_b_devtype] EQLU .terminal_type[.i]
;  1888          2261  2     THEN (arglist[6] = .terminal_label[.i]; EXITLOOP));
;  1889          2262  2
;  1890          2263  2 common_display(arglist, .scratch);              ! Print the common heading
;  1891          2264  2
;  1892          2265  2 !
;  1893          2266  2 ! Show any access control list which might be present
;  1894          2267  2 !
;  1895          2268  2 show_device_acl (.scratch);
;  1896          2269  2
;  1897          2270  2 RETURN;
;  1898          2271  1 END;
```

```
                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                20 6C 61 6E 69 6D 72 65 54 09  013FC P.ANZ:   .ASCII  <9>\Terminal \
                      6E 77 6F 6E 6B 6E 75 07  01406 P.AOA:   .ASCII  <7>\unknown\                                ;


                                                        .PSECT  $CODE$,NOWRT,2

                                          0000 00000            .ENTRY  DISPLAY_TERMINAL, Save nothing             ; 2218
                                    5E      B0  AE    9E 00002  MOVAB   -80(SP), SP                                ;
                                    7E      04  AC    7D 00006  MOVQ    SCRATCH, -(SP)                           ; 2246
                              F056  CF          02    FB 0000A  CALLS   #2, CHECK_MNT_ALL                        ;
                                    3A          50    E9 0000F  BLBC    R0, 4$                                   ;
                                    6E  0000'   CF    9E 00012  MOVAB   P.ANZ, ARGLIST                           ; 2252
                                18  AE  0000'   CF    9E 00017  MOVAB   P.AOA, ARGLIST+24                        ; 2258
                                    51    04    AC    D0 0001D  MOVL    SCRATCH, R1                              ; 2260
                                          50    D4 00021        CLRL    I                                        ;
                            0000'CF40     79    A1    91 00023 1$:    CMPB    121(R1), TERMINAL_TYPE[I]          ;
                                          09    12 0002A        BNEQ    2$                                       ;
                                18  AE  0000'CF40   D0 0002C        MOVL    TERMINAL_LABEL[I], ARGLIST+24        ; 2261
                                          04    11 00033        BRB     3$                                       ;
                          EA              50    22    F3 00035 2$:    AOBLEQ  #34, I, 1$                       ; 2260
                                    04    AC    DD 00039 3$:    PUSHL   SCRATCH                                  ; 2263
                                    04    AE    9F 0003C        PUSHAB  ARGLIST                                  ;
                            0000V   CF          02    FB 0003F  CALLS   #2, COMMON_DISPLAY                       ;
                                    04    AC    DD 00044        PUSHL   SCRATCH                                  ; 2268
                              F13F  CF          01    FB 00047  CALLS   #1, SHOW_DEVICE_ACL                      ;
                                          04 0004C 4$:          RET                                              ; 2271

; Routine Size:  77 bytes,    Routine Base:  $CODE$ + 1048
```

```
;  1900          2272  1 GLOBAL ROUTINE display_general (scratch, flags) : NOVALUE =
;  1901          2273  2 BEGIN
;  1902          2274  2
;  1903          2275  2 !---
;  1904          2276  2 !
;  1905          2277  2 ! This is the general display routine for devices
;  1906          2278  2 !
;  1907          2279  2 ! Inputs
;  1908          2280  2 !       SCRATCH - contains data about the device
;  1909          2281  2 !       FLAGS   - contains info on what to print
;  1910          2282  2 !
;  1911          2283  2 ! Outputs
;  1912          2284  2 !       None
;  1913          2285  2 !
;  1914          2286  2 !---
;  1915          2287  2
;  1916          2288  2 MAP
;  1917          2289  2     scratch : REF $BBLOCK,
;  1918          2290  2     flags : REF $BBLOCK;
;  1919          2291  2
;  1920          2292  2 LOCAL
;  1921          2293  2     arglist : VECTOR[20];
;  1922          2294  2
;  1923          2295  2 !
;  1924          2296  2 ! Make a check to see if /ALLOCATED or /MOUNTED was specified.  If so,
;  1925          2297  2 ! then check to see if the device is mounted/allocated.  If not, then
;  1926          2298  2 ! don't display information on the device.
;  1927          2299  2 !
;  1928          2300  2 IF NOT check_mnt_all(.scratch, .flags)
;  1929          2301  2 THEN RETURN;
;  1930          2302  2
;  1931          2303  2 !
;  1932          2304  2 ! Set up the argument list for output.
;  1933          2305  2 !
;  1934          2306  2 arglist[0] = cstring('Device ');
;  1935          2307  2 arglist[6] = 0;
;  1936          2308  2
;  1937          2309  2 common_display(arglist, .scratch);              ! Print the common heading
;  1938          2310  2
;  1939          2311  2 !
;  1940          2312  2 ! Show any access control list which might be present
;  1941          2313  2 !
;  1942          2314  2 show_device_acl (.scratch);
;  1943          2315  2
;  1944          2316  2 RETURN;
;  1945          2317  1 END;


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
                    20  65  63  69  76  65  44  07  0140E P.AOB:  .ASCII  <7>\Device \                          ;

                                                        .PSECT  $CODE$,NOWRT,2
```

```
                                0000 00000          .ENTRY  DISPLAY_GENERAL, Save nothing                ; 2272
                          5E    B0   AE   9E 00002  MOVAB   -80(SP), SP                                  ;
                          7E    04   AC   7D 00006  MOVQ    SCRATCH, -(SP)                              ; 2300
                F009      CF         02   FB 0000A  CALLS   #2, CHECK_MNT_ALL                           ;
                          1B         50   E9 0000F  BLBC    R0, 1$                                      ;
                          6E  0000'  CF   9E 00012  MOVAB   P.AOB, ARGLIST                              ; 2306
                              18     AE   D4 00017  CLRL    ARGLIST+24                                  ; 2307
                              04     AC   DD 0001A  PUSHL   SCRATCH                                     ; 2309
                              04     AE   9F 0001D  PUSHAB  ARGLIST                                     ;
                0000V         CF         02   FB 00020  CALLS   #2, COMMON_DISPLAY                      ;
                              04     AC   DD 00025  PUSHL   SCRATCH                                     ; 2314
                F111          CF         01   FB 00028  CALLS   #1, SHOW_DEVICE_ACL                     ;
                                         04 0002D 1$:    RET                                            ; 2317

; Routine Size:  46 bytes,    Routine Base:  $CODE$ + 1095
```

```
;1947     2318  1 ROUTINE common_display (arglist, scratch) : NOVALUE =
;1948     2319  2 BEGIN
;1949     2320  2
;1950     2321  2 !---
;1951     2322  2 !
;1952     2323  2 ! This routine displays the common heading for all devices
;1953     2324  2 !
;1954     2325  2 ! Inputs
;1955     2326  2 !       ARGLIST - partial argument list, containing
;1956     2327  2 !                 ARGLST[0] = device class      (ASCIC)
;1957     2328  2 !                 ARGLST[6] = device type       (ASCIC)
;1958     2329  2 !       SCRATCH - address of area that contains info on the device
;1959     2330  2 !
;1960     2331  2 ! Outputs
;1961     2332  2 !       None
;1962     2333  2 !
;1963     2334  2 !---
;1964     2335  2
;1965     2336  2 MAP
;1966     2337  2     arglist : REF VECTOR,
;1967     2338  2     scratch : REF $BBLOCK;
;1968     2339  2
;1969     2340  2 LOCAL
;1970     2341  2     index,                                  ! Pointer
;1971     2342  2     count_pointer,                          ! Pointer to count cell
;1972     2343  2     ptr : REF VECTOR,                       ! Descriptor pointer
;1973     2344  2     desc_vector : VECTOR[10]                ! Array of descriptors
;1974     2345  2                   INITIAL(REP 10 OF (0)),
;1975     2346  2     fao_buff : VECTOR[200];                 ! Place to store stuff (10 lines by 80 chars)
;1976     2347  2
;1977     2348  2 BIND
;1978     2349  2     fao_desc = desc_vector[0] : VECTOR[2],  ! Descriptor of formatted UIC
;1979     2350  2     prot_desc = desc_vector[2] : VECTOR[2]; ! Descriptor for protection
;1980     2351  2
;1981     2352  2
;1982     2353  2
;1983     2354  2 !
;1984     2355  2 ! Get the device and alias names.  The actual string is of the form "_DDCU",
;1985     2356  2 ! so shuffle to get rid of the underscore.
;1986     2357  2 !
;1987     2358  2 arglist[1] = .scratch[d_b_devlen] - 1;
;1988     2359  2 arglist[2] = scratch[d_t_device] + 1;
;1989     2360  2
;1990     2361  2 arglist[3] = .scratch[d_t_host_name];
;1991     2362  2 arglist[4] = scratch[d_t_host_name]+1;
;1992     2363  2
;1993     2364  2 !
;1994     2365  2 ! Set up a buffer to hold the device name and characteristics
;1995     2366  2 !
;1996     2367  2 desc_vector[0] = 80;
;1997     2368  2 desc_vector[1] = fao_buff;
;1998     2369  2 ptr = desc_vector;
;1999     2370  2 leading_blanks = 4;
;2000   P 2371  2 $FAOL(OUTBUF = desc_vector,
;2001   P 2372  2       OUTLEN = desc_vector,
;2002   P 2373  2       PRMLST = .arglist,
;2003   P 2374  2       CTRSTR = (IF .$BBLOCK[scratch[d_l_devchar], dev$v_fod]
```

```
;  2004        P 2375  2                    AND
;  2005        P 2376  2                        .scratch[d_l_allocls] NEQ 0
;  2006        P 2377  2                    THEN
;  2007        P 2378  2                        BEGIN
;  2008        P 2379  2                        IF .arglist[6] NEQ 0
;  2009        P 2380  2                        THEN %ASCID '!AC!AF (!AF), device type !+!AC, is '
;  2010        P 2381  2                        ELSE %ASCID '!AC!AF (!AF) is '
;  2011        P 2382  2                        END
;  2012        P 2383  2                    ELSE
;  2013        P 2384  2                        BEGIN
;  2014        P 2385  2                        IF .arglist[6] NEQ 0
;  2015        P 2386  2                        THEN %ASCID '!AC!AF!+!+, device type !+!AC, is '
;  2016        P 2387  2                        ELSE %ASCID '!AC!AF is '
;  2017          2388  2                        END));
;  2018          2389  2
;  2019          2390  2 !
;  2020          2391  2 ! Get particular characteristics of interest
;  2021          2392  2 !
;  2022          2393  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_online), 1, 0]
;  2023          2394  2 THEN add_to_list(ptr, 'online,', 80)
;  2024          2395  2 ELSE add_to_list(ptr, 'offline,', 80);
;  2025          2396  2
;  2026          2397  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_timout), 1, 0]
;  2027          2398  2 THEN add_to_list(ptr, ' unit timed out,', 80);
;  2028          2399  2
;  2029          2400  2 IF NOT .scratch[d_v_host_avail]
;  2030          2401  2 THEN add_to_list(ptr, ' host not available,', 80);
;  2031          2402  2
;  2032          2403  2 IF NOT .$BBLOCK[scratch[d_l_devchar], dev$v_avl]
;  2033          2404  2 THEN add_to_list(ptr, ' device set /NOAVAILABLE,', 80);
;  2034          2405  2
;  2035          2406  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_all]
;  2036          2407  2 THEN add_to_list(ptr, ' allocated,', 80);
;  2037          2408  2
;  2038          2409  2 IF .$BBLOCK[scratch[d_l_sts], 0,$BITPOSITION(ucb$v_deadmo),1,0]
;  2039          2410  2   AND
;  2040          2411  2     .$BBLOCK[scratch[d_l_devchar], dev$v_all]
;  2041          2412  2 THEN add_to_list(ptr, ' deallocate on dismount,', 80);
;  2042          2413  2
;  2043          2414  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_mnt]
;  2044          2415  2 THEN
;  2045          2416  3     BEGIN
;  2046          2417  3     IF .$BBLOCK[scratch[d_l_devchar], dev$v_for]        ! Foreign
;  2047          2418  3     THEN add_to_list(ptr, ' mounted foreign,', 80)
;  2048          2419  3     ELSE add_to_list(ptr, ' mounted,', 80);
;  2049          2420  3     IF .$BBLOCK[scratch[d_l_devchar], dev$v_swl]        ! Software-writelocked
;  2050          2421  3     THEN add_to_list(ptr, ' software write-locked,', 80);
;  2051          2422  3     !
;  2052          2423  3     ! If mount verification has timed out, display that, otherwise if it is active show that.  Do not
;  2053          2424  3     ! show both timed out and active.
;  2054          2425  3     !
;  2055          2426  3     SELECTONE true OF
;  2056          2427  3     SET
;  2057          2428  3         [ .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_mntverip), 1, 0] ]
;  2058          2429  3
;  2059          2430  3                             : add_to_list(ptr, ' mount verification in progress,', 80);
;  2060          2431  3
```

```
;2061     2432  4          [ (NOT .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_valid), 1, 0])
;2062     2433  3              AND
;2063     2434  3            ((.scratch[d_b_aqbtype] EQL aqb$k_f11v1) OR (.scratch[d_b_aqbtype] EQL aqb$k_f11v2)) ]
;2064     2435  3
;2065     2436  3                                : add_to_list(ptr, ' mount verification timed out,', 80);
;2066     2437  3      TES;
;2067     2438  3      IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_wrongvol), 1, 0]
;2068     2439  3      THEN add_to_list(ptr, ' mount verification found wrong volume,', 80);
;2069     2440  2      END;
;2070     2441  2
;2071     2442  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_dmt]
;2072     2443  2  THEN add_to_list(ptr, ' volume is marked for dismount,', 80);
;2073     2444  2
;2074     2445  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_opr]
;2075     2446  2  THEN add_to_list(ptr, ' enabled as operator terminal,', 80);
;2076     2447  2
;2077     2448  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_dua]
;2078     2449  2  THEN add_to_list(ptr, ' device is set /DUAL_PORT,', 80);
;2079     2450  2
;2080     2451  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_rck]
;2081     2452  2  THEN add_to_list(ptr, ' data check on reads,', 80);
;2082     2453  2
;2083     2454  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_wck]
;2084     2455  2  THEN add_to_list(ptr, ' data check on writes,', 80);
;2085     2456  2
;2086     2457  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_rec]
;2087     2458  2  THEN add_to_list(ptr, ' record-oriented device,', 80);
;2088     2459  2
;2089     2460  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_ccl]
;2090     2461  2  THEN add_to_list(ptr, ' carriage control,', 80);
;2091     2462  2
;2092     2463  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_net]
;2093     2464  2  THEN add_to_list(ptr, ' network device,', 80);
;2094     2465  2
;2095     2466  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_fod]
;2096     2467  2  THEN add_to_list(ptr, ' file-oriented device,', 80);
;2097     2468  2
;2098     2469  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_shr]
;2099     2470  2  THEN add_to_list(ptr, ' shareable,', 80);
;2100     2471  2
;2101     2472  2  IF .$BBLOCK[scratch[d_l_devchar], dev$v_rtm]
;2102     2473  2  THEN add_to_list(ptr, ' real time device,', 80);
;2103     2474  2
;2104     2475  2  IF .$BBLOCK[scratch[d_l_devchar2], dev$v_clu]          ! If available show so, but
;2105     2476  3   AND (NOT .$BBLOCK[scratch[d_l_devchar2], dev$v_srv])   !  not if we served it (it's obvious then)
;2106     2477  2  THEN add_to_list(ptr, ' available to cluster,', 80);
;2107     2478  2
;2108     2479  2  IF .$BBLOCK[scratch[d_l_devchar2], dev$v_red]
;2109     2480  2  THEN add_to_list(ptr, ' device is a physical terminal accessed through a virtual terminal,', 80);
;2110     2481  2
;2111     2482  2  IF .$BBLOCK[scratch[d_l_devchar2], dev$v_ssm]
;2112     2483  2  THEN add_to_list(ptr, ' member of a shadow set,', 80);
;2113     2484  2
;2114     2485  2  IF .scratch[d_v_shadow_master]
;2115     2486  2  THEN add_to_list(ptr, ' master of a shadow set,', 80);
;2116     2487  2
;2117     2488  2  IF .$BBLOCK[scratch[d_l_devchar2], dev$v_srv]
```

```
;2118        2489  2 THEN add_to_list(ptr, ' served to cluster via MSCP Server,', 80);
;2119        2490  2
;2120        2491  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_spl]
;2121        2492  2 THEN add_to_list(ptr, ' device is spooled through an intermediate device,', 80);
;2122        2493  2
;2123        2494  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_elg]
;2124        2495  2 THEN add_to_list(ptr, ' error logging is enabled,', 80);
;2125        2496  2
;2126        2497  2 IF .$BBLOCK[scratch[d_l_devchar2], dev$v_det]   ! detached from physical
;2127        2498  2 THEN add_to_list(ptr, ' virtual device is disconnected from physical device,', 80);
;2128        2499  2
;2129        2500  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_gen]
;2130        2501  2 THEN add_to_list(ptr, ' generic device,', 80);
;2131        2502  2
;2132        2503  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_mbx]
;2133        2504  2 THEN add_to_list(ptr, ' mailbox device,', 80);
;2134        2505  2
;2135        2506  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_erlogip), 1, 0]
;2136        2507  2 THEN add_to_list(ptr, ' error currently being logged,', 80);
;2137        2508  2
;2138        2509  2 IF .$BBLOCK[scratch[d_b_orb_flags], 0, $BITPOSITION(orb$v_noacl), 1, 0]
;2139        2510  2 THEN add_to_list(ptr, ' access control list not supported on this type of device,', 80);
;2140        2511  2
;2141        2512  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_cancel), 1, 0]
;2142        2513  2 THEN add_to_list(ptr, ' operations being canceled,', 80);
;2143        2514  2
;2144        2515  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_power), 1, 0]
;2145        2516  2 THEN add_to_list(ptr, ' power failure has been recorded,', 80);
;2146        2517  2
;2147        2518  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_bsy), 1, 0]
;2148        2519  2 THEN add_to_list(ptr, ' device is busy,', 80);
;2149        2520  2
;2150        2521  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_mounting), 1, 0]
;2151        2522  2 THEN add_to_list(ptr, ' mount operation in progress,', 80);
;2152        2523  2
;2153        2524  2 IF .$BBLOCK[scratch[d_l_sts], 0, $BITPOSITION(ucb$v_template), 1, 0]
;2154        2525  2 THEN add_to_list(ptr, ' device is a template only,', 80);
;2155        2526  2
;2156        2527  2 !
;2157        2528  2 ! Now print out a blank line followed by the information
;2158        2529  2 !
;2159        2530  2 show$write_line(ascid_null, arglist[0]);
;2160        2531  2 INCR i FROM 0 TO 2 DO
;2161        2532  3     BEGIN
;2162        2533  3     IF .desc_vector[2*(.i+1)] EQL 0              ! On the last string change the final
;2163        2534  3     THEN                                         !  comma to a period.
;2164        2535  4         BEGIN
;2165        2536  4         BIND
;2166        2537  4             str = .desc_vector[(2*.i)+1],
;2167        2538  4             dot = (str+.desc_vector[2*.i]-1) : BYTE;
;2168        2539  4         dot = '.';
;2169        2540  3         END;
;2170        2541  3     arglist[0] = desc_vector[2*.i];
;2171        2542  3     show$write_line(%ASCID '!AS', arglist[0]);
;2172        2543  3     IF .desc_vector[2*(.i+1)] EQL 0
;2173        2544  3     THEN EXITLOOP
;2174        2545  2     END;
```

```
;2175          2546  2
;2176          2547  2 !
;2177          2548  2 ! Display statistics common to all devices.
;2178          2549  2 !
;2179          2550  2 arglist[0] = .scratch[d_w_errcnt];
;2180          2551  2 arglist[1] = .scratch[d_l_opcnt];
;2181          2552  2 arglist[2] = 15 - .scratch[d_t_prcnam];
;2182          2553  2 arglist[3] = .scratch[d_t_prcnam];
;2183          2554  2 arglist[4] = scratch[d_t_prcnam] + 1;
;2184          2555  2 fao_desc[0] = 80;
;2185          2556  2 fao_desc[1] = fao_buff;
;2186        P 2557  2 $FAOL(CTRSTR = %ASCID '!%I',
;2187        P 2558  2       OUTBUF = fao_desc,
;2188        P 2559  2       OUTLEN = fao_desc,
;2189          2560  2       PRMLST = %REF(.scratch[d_l_ownuic]));
;2190          2561  3 arglist[5] = (IF .FAO_DESC[0] GEQ 30
;2191          2562  2                THEN 1 ELSE 30 - .FAO_DESC[0]);
;2192          2563  2 arglist[6] = fao_desc;
;2193          2564  4 arglist[7] =   (BEGIN          ! convert to the extended pid format
;2194          2565  4                  LINKAGE
;2195          2566  4                      cvt_lnk = JSB (REGISTER=0) : PRESERVE (1,2,3,4,5) NOTUSED (6,7,8,9,10,11);
;2196          2567  4                  EXTERNAL ROUTINE
;2197          2568  4                      EXE$IPID_TO_EPID : cvt_lnk ADDRESSING_MODE (GENERAL);
;2198          2569  4                  EXE$IPID_TO_EPID (.scratch[d_l_pid])
;2199          2570  2                  END);
;2200          2571  2 arglist[9] = prot_desc;
;2201          2572  2 IF .$BBLOCK[scratch[d_l_devchar], dev$v_net]
;2202          2573  3 THEN (arglist[8] = 1; prot_desc[0] = 0)
;2203          2574  2 ELSE
;2204          2575  3     BEGIN
;2205          2576  3     LOCAL
;2206          2577  3         key,
;2207          2578  3         prot_addr : VECTOR[4];
;2208          2579  3     IF .$BBLOCK[scratch[d_l_devchar], dev$v_fod]
;2209          2580  3     THEN key = 0
;2210          2581  3     ELSE key = 1;
;2211          2582  3     expand_prot(prot_addr, .scratch[d_w_vprot], .key);
;2212          2583  3     prot_desc[0] = 80;
;2213          2584  3     prot_desc[1] = fao_buff + 80;
;2214        P 2585  3     $FAOL(CTRSTR = %ASCID 'S:!AS,O:!AS,G:!AS,W:!AS',
;2215        P 2586  3           OUTLEN = prot_desc,
;2216        P 2587  3           OUTBUF = prot_desc,
;2217          2588  3           PRMLST = prot_addr);
;2218          2589  2     END;
;2219          2590  2 arglist[8] = 31 - .prot_desc[0];
;2220          2591  2 arglist[10] = .scratch[d_w_refc];
;2221          2592  2 arglist[11] = .scratch[d_w_devbufsiz];
;2222          2593  2
;2223          2594  2 show$write_line(ascid_null, arglist[0],
;2224          2595  2                  %ASCID '    Error count!21UW    Operations completed!19UL',
;2225          2596  2                  arglist[0],
;2226          2597  2                  %ASCID '    Owner process  !#< !>"!AF"    Owner UIC!#< !>!AS',
;2227          2598  2                  arglist[2],
;2228          2599  2                  %ASCID '    Owner process ID!16XL    Dev Prot!#< !>!AS',
;2229          2600  2                  arglist[7],
;2230          2601  2                  %ASCID '    Reference count!17UL    Default buffer size!20UW',
;2231          2602  2                  arglist[10]);
```

```
;  2232          2603  2
;  2233          2604  2 RETURN;
;  2234          2605  1 END;


                                                                .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                        01416           .BLKB   2
                                                00000000# 01418 P.AOC:  .LONG   0[10]
64  20  2C  29  46  41  21  28  20  46  41  21  43  41  21  01440 P.AOE:  .ASCII  \!AC!AF (!AF), device type !+!AC, is \
41  21  2B  21  20  65  70  79  74  20  65  63  69  76  65  0144F
                                    20  73  69  20  2C  43  0145E
                                                010E0024  01464 P.AOD:  .LONG   17694756
                                                00000000' 01468         .ADDRESS P.AOE
73  69  20  29  46  41  21  28  20  46  41  21  43  41  21  0146C P.AOG:  .ASCII  \!AC!AF (!AF) is \
                                                        20  0147B
                                                010E0010  0147C P.AOF:  .LONG   17694736
                                                00000000' 01480         .ADDRESS P.AOG
76  65  64  20  2C  2B  21  2B  21  46  41  21  43  41  21  01484 P.AOI:  .ASCII  \!AC!AF!+!+, device type !+!AC, is \<0>
2C  43  41  21  2B  21  20  65  70  79  74  20  65  63  69  01493
                                        00  20  73  69  20  014A2
                                                        00  014A7        .ASCII  <0>
                                                010E0022  014A8 P.AOH:  .LONG   17694754
                                                00000000' 014AC         .ADDRESS P.AOI
            00  00  20  73  69  20  46  41  21  43  41  21  014B0 P.AOK:  .ASCII  \!AC!AF is \<0><0>
                                                010E000A  014BC P.AOJ:  .LONG   17694730
                                                00000000' 014C0         .ADDRESS P.AOK
                        00  2C  65  6E  69  6C  6E  6F  014C4 P.AOM:  .ASCII  \online,\<0>
                                                010E0007  014CC P.AOL:  .LONG   17694727
                                                00000000' 014D0         .ADDRESS P.AOM
                            2C  65  6E  69  6C  66  66  6F  014D4 P.AOO:  .ASCII  \offline,\
                                                010E0008  014DC P.AON:  .LONG   17694728
                                                00000000' 014E0         .ADDRESS P.AOO
74  75  6F  20  64  65  6D  69  74  20  74  69  6E  75  20  014E4 P.AOQ:  .ASCII  \ unit timed out,\
                                                            2C  014F3
                                                010E0010  014F4 P.AOP:  .LONG   17694736
                                                00000000' 014F8         .ADDRESS P.AOQ
6C  69  61  76  61  20  74  6F  6E  20  74  73  6F  73  20  014FC P.AOS:  .ASCII  \ host not available,\
                                                    2C  65  6C  62  61  0150B
                                                010E0014  01510 P.AOR:  .LONG   17694740
                                                00000000' 01514         .ADDRESS P.AOS
4F  4E  2F  20  74  65  73  20  65  63  69  76  65  64  20  01518 P.AOU:  .ASCII  \ device set /NOAVAILABLE,\<0><0><0>
        00  00  00  2C  45  4C  42  41  4C  49  41  56  41  01527
                                                010E0019  01534 P.AOT:  .LONG   17694745
                                                00000000' 01538         .ADDRESS P.AOU
            00  2C  64  65  74  61  63  6F  6C  6C  61  20  0153C P.AOW:  .ASCII  \ allocated,\<0>
                                                010E000B  01548 P.AOV:  .LONG   17694731
                                                00000000' 0154C         .ADDRESS P.AOW
20  6E  6F  20  65  74  61  63  6F  6C  6C  61  65  64  20  01550 P.AOY:  .ASCII  \ deallocate on dismount,\
                                    2C  74  6E  75  6F  6D  73  69  64  0155F
                                                010E0018  01568 P.AOX:  .LONG   17694744
                                                00000000' 0156C         .ADDRESS P.AOY
67  69  65  72  6F  66  20  64  65  74  6E  75  6F  6D  20  01570 P.APA:  .ASCII  \ mounted foreign,\<0><0><0>
                                            00  00  00  2C  6E  0157F
                                                010E0011  01584 P.AOZ:  .LONG   17694737
                                                00000000' 01588         .ADDRESS P.APA
```

```
                00 00 00 2C 64 65 74 6E 75 6F 6D 20  0158C P.APC:  .ASCII  \ mounted,\<0><0><0>
                                                 010E0009  01598 P.APB:  .LONG   17694729
                                                 00000000'  0159C          .ADDRESS P.APC
65 74 69 72 77 20 65 72 61 77 74 66 6F 73 20  015A0 P.APE:  .ASCII  \ software write-locked,\<0>
                   00 2C 64 65 6B 63 6F 6C 2D  015AF
                                                 010E0017  015B8 P.APD:  .LONG   17694743
                                                 00000000'  015BC          .ADDRESS P.APE
61 63 69 66 69 72 65 76 20 74 6E 75 6F 6D 20  015C0 P.APG:  .ASCII  \ mount verification in progress,\
73 65 72 67 6F 72 70 20 6E 69 20 6E 6F 69 74  015CF
                                                 2C 73  015DE
                                                 010E0020  015E0 P.APF:  .LONG   17694752
                                                 00000000'  015E4          .ADDRESS P.APG
61 63 69 66 69 72 65 76 20 74 6E 75 6F 6D 20  015E8 P.API:  .ASCII  \ mount verification timed out,\<0><0>
2C 74 75 6F 20 64 65 6D 69 74 20 6E 6F 69 74  015F7
                                                 00 00  01606
                                                 010E001E  01608 P.APH:  .LONG   17694750
                                                 00000000'  0160C          .ADDRESS P.API
61 63 69 66 69 72 65 76 20 74 6E 75 6F 6D 20  01610 P.APK:  .ASCII  \ mount verification found wrong volume,-
6E 6F 72 77 20 64 6E 75 6F 66 20 6E 6F 69 74  0161F                \<0>
                   00 2C 65 6D 75 6C 6F 76 20 67  0162E
                                                 010E0027  01638 P.APJ:  .LONG   17694759
                                                 00000000'  0163C          .ADDRESS P.APK
6B 72 61 6D 20 73 69 20 65 6D 75 6C 6F 76 20  01640 P.APM:  .ASCII  \ volume is marked for dismount,\<0>
74 6E 75 6F 6D 73 69 64 20 72 6F 66 20 64 65  0164F
                                                 00 2C  0165E
                                                 010E001F  01660 P.APL:  .LONG   17694751
                                                 00000000'  01664          .ADDRESS P.APM
65 70 6F 20 73 61 20 64 65 6C 62 61 6E 65 20  01668 P.APO:  .ASCII  \ enabled as operator terminal,\<0><0>
2C 6C 61 6E 69 6D 72 65 74 20 72 6F 74 61 72  01677
                                                 00 00  01686
                                                 010E001E  01688 P.APN:  .LONG   17694750
                                                 00000000'  0168C          .ADDRESS P.APO
20 74 65 73 20 73 69 20 65 63 69 76 65 64 20  01690 P.APQ:  .ASCII  \ device is set /DUAL_PORT,\<0><0>
   00 00 2C 54 52 4F 50 5F 4C 41 55 44 2F 54  0169F
                                                 010E001A  016AC P.APP:  .LONG   17694746
                                                 00000000'  016B0          .ADDRESS P.APQ
20 6E 6F 20 6B 63 65 68 63 20 61 74 61 64 20  016B4 P.APS:  .ASCII  \ data check on reads,\<0><0><0>
                   00 00 00 2C 73 64 61 65 72  016C3
                                                 010E0015  016CC P.APR:  .LONG   17694741
                                                 00000000'  016D0          .ADDRESS P.APS
20 6E 6F 20 6B 63 65 68 63 20 61 74 61 64 20  016D4 P.APU:  .ASCII  \ data check on writes,\<0><0>
                   00 00 2C 73 65 74 69 72 77  016E3
                                                 010E0016  016EC P.APT:  .LONG   17694742
                                                 00000000'  016F0          .ADDRESS P.APU
65 74 6E 65 69 72 6F 2D 64 72 6F 63 65 72 20  016F4 P.APW:  .ASCII  \ record-oriented device,\
                   2C 65 63 69 76 65 64 20 64  01703
                                                 010E0018  0170C P.APV:  .LONG   17694744
                                                 00000000'  01710          .ADDRESS P.APW
72 74 6E 6F 63 20 65 67 61 69 72 72 61 63 20  01714 P.APY:  .ASCII  \ carriage control,\<0><0>
                                  00 00 2C 6C 6F  01723
                                                 010E0012  01728 P.APX:  .LONG   17694738
                                                 00000000'  0172C          .ADDRESS P.APY
65 63 69 76 65 64 20 6B 72 6F 77 74 65 6E 20  01730 P.AQA:  .ASCII  \ network device,\
                                                 2C  0173F
                                                 010E0010  01740 P.APZ:  .LONG   17694736
                                                 00000000'  01744          .ADDRESS P.AQA
20 64 65 74 6E 65 69 72 6F 2D 65 6C 69 66 20  01748 P.AQC:  .ASCII  \ file-oriented device,\<0><0>
```

```
                                00 00 2C 65 63 69 76 65 64  01757
                                                010E0016   01760 P.AQB:  .LONG   17694742
                                                00000000'  01764         .ADDRESS P.AQC
                   00 2C 65 6C 62 61 65 72 61 68 73 20     01768 P.AQE:  .ASCII  \ shareable,\<0>
                                                010E000B   01774 P.AQD:  .LONG   17694731
                                                00000000'  01778         .ADDRESS P.AQE
69 76 65 64 20 65 6D 69 74 20 6C 61 65 72 20    0177C P.AQG:  .ASCII  \ real time device,\<0><0>
                                     00 00 2C 65 63     0178B
                                                010E0012   01790 P.AQF:  .LONG   17694738
                                                00000000'  01794         .ADDRESS P.AQG
63 20 6F 74 20 65 6C 62 61 6C 69 61 76 61 20    01798 P.AQI:  .ASCII  \ available to cluster,\<0><0>
                                  00 00 2C 72 65 74 73 75 6C    017A7
                                                010E0016   017B0 P.AQH:  .LONG   17694742
                                                00000000'  017B4         .ADDRESS P.AQI
68 70 20 61 20 73 69 20 65 63 69 76 65 64 20    017B8 P.AQK:  .ASCII  \ device is a physical terminal accessed \
6C 61 6E 69 6D 72 65 74 20 6C 61 63 69 73 79    017C7
                        20 64 65 73 73 65 63 63 61 20  017D6
75 74 72 69 76 20 61 20 68 67 75 6F 72 68 74    017E0         .ASCII  \through a virtual terminal,\<0>
      00 2C 6C 61 6E 69 6D 72 65 74 20 6C 61    017EF
                                                010E0043   017FC P.AQJ:  .LONG   17694787
                                                00000000'  01800         .ADDRESS P.AQK
68 73 20 61 20 66 6F 20 72 65 62 6D 65 6D 20    01804 P.AQM:  .ASCII  \ member of a shadow set,\
                                  2C 74 65 73 20 77 6F 64 61    01813
                                                010E0018   0181C P.AQL:  .LONG   17694744
                                                00000000'  01820         .ADDRESS P.AQM
68 73 20 61 20 66 6F 20 72 65 74 73 61 6D 20    01824 P.AQO:  .ASCII  \ master of a shadow set,\
                                  2C 74 65 73 20 77 6F 64 61    01833
                                                010E0018   0183C P.AQN:  .LONG   17694744
                                                00000000'  01840         .ADDRESS P.AQO
73 75 6C 63 20 6F 74 20 64 65 76 72 65 73 20    01844 P.AQQ:  .ASCII  \ served to cluster via MSCP Server,\<0>
65 53 20 50 43 53 4D 20 61 69 76 20 72 65 74    01853
                                     00 2C 72 65 76 72  01862
                                                010E0023   01868 P.AQP:  .LONG   17694755
                                                00000000'  0186C         .ADDRESS P.AQQ
6F 6F 70 73 20 73 69 20 65 63 69 76 65 64 20    01870 P.AQS:  .ASCII  \ device is spooled through an intermedia\
20 6E 61 20 68 67 75 6F 72 68 74 20 64 65 6C    0187F
                        61 69 64 65 6D 72 65 74 6E 69  0188E
      00 00 2C 65 63 69 76 65 64 20 65 74       01898         .ASCII  \te device,\<0><0>
                                                010E0032   018A4 P.AQR:  .LONG   17694770
                                                00000000'  018A8         .ADDRESS P.AQS
20 67 6E 69 67 67 6F 6C 20 72 6F 72 72 65 20    018AC P.AQU:  .ASCII  \ error logging is enabled,\<0><0>
      00 00 2C 64 65 6C 62 61 6E 65 20 73 69    018BB
                                                010E001A   018C8 P.AQT:  .LONG   17694746
                                                00000000'  018CC         .ADDRESS P.AQU
65 63 69 76 65 64 20 6C 61 75 74 72 69 76 20    018D0 P.AQW:  .ASCII  \ virtual device is disconnected from phy\
65 74 63 65 6E 6E 6F 63 73 69 64 20 73 69 20    018DF
                        79 68 70 20 6D 6F 72 66 20 64  018EE
00 00 2C 65 63 69 76 65 64 20 6C 61 63 69 73    018F8         .ASCII  \sical device,\<0><0><0>
                                     00         01907
                                                010E0035   01908 P.AQV:  .LONG   17694773
                                                00000000'  0190C         .ADDRESS P.AQW
65 63 69 76 65 64 20 63 69 72 65 6E 65 67 20    01910 P.AQY:  .ASCII  \ generic device,\
                                              2C    0191F
                                                010E0010   01920 P.AQX:  .LONG   17694736
                                                00000000'  01924         .ADDRESS P.AQY
65 63 69 76 65 64 20 78 6F 62 6C 69 61 6D 20    01928 P.ARA:  .ASCII  \ mailbox device,\
                                              2C    01937
```

```
                                                    010E0010  01938 P.AQZ:  .LONG   17694736
                                                    00000000' 0193C        .ADDRESS P.ARA
6C 74 6E 65 72 72 75 63 20 72 6F 72 72  65  20  01940 P.ARC:  .ASCII  \ error currently being logged,\<0><0>
2C 64 65 67 67 6F 6C 20 67 6E 69 65 62  20  79  0194F
                                                       00  00  0195E
                                                    010E001E  01960 P.ARB:  .LONG   17694750
                                                    00000000' 01964        .ADDRESS P.ARC
6C 6F 72 74 6E 6F 63 20 73 73 65 63 63  61  20  01968 P.ARE:  .ASCII  \ access control list not supported on th\
6F 70 70 75 73 20 74 6F 6E 20 74 73 69  6C  20  01977
                  68 74 20 6E 6F 20 64 65 74 72  01986
69 76 65 64 20 66 6F 20 65 70 79 74 20  73  69  01990        .ASCII  \is type of device,\<0><0>
                        00 00 2C 65 63  61  6C  019A1
                                                    010E003A  019A4 P.ARD:  .LONG   17694778
                                                    00000000' 019A8        .ADDRESS P.ARE
69 65 62 20 73 6E 6F 69 74 61 72 65 70  6F  20  019AC P.ARG:  .ASCII  \ operations being canceled,\<0>
      00 2C 64 65 6C 65 63 6E 61 63 20  67  6E  019BB
                                                    010E001B  019C8 P.ARF:  .LONG   17694747
                                                    00000000' 019CC        .ADDRESS P.ARG
20 65 72 75 6C 69 61 66 20 72 65 77 6F  70  20  019D0 P.ARI:  .ASCII  \ power failure has been recorded,\<0><0>
64 72 6F 63 65 72 20 6E 65 65 62 20 73  61  68  019DF
                        00 00 2C 64 65  65  6C  019EE
                                          00    019F3        .ASCII  <0>
                                                    010E0021  019F4 P.ARH:  .LONG   17694753
                                                    00000000' 019F8        .ADDRESS P.ARI
79 73 75 62 20 73 69 20 65 63 69 76 65  64  20  019FC P.ARK:  .ASCII  \ device is busy,\
                                             2C  01A0B
                                                    010E0010  01A0C P.ARJ:  .LONG   17694736
                                                    00000000' 01A10        .ADDRESS P.ARK
6F 69 74 61 72 65 70 6F 20 74 6E 75 6F  6D  20  01A14 P.ARM:  .ASCII  \ mount operation in progress,\<0><0>
00 2C 73 73 65 72 67 6F 72 70 20 6E 69  20  6E  01A23
                                          00    01A32
                                          00    01A33        .ASCII  <0>
                                                    010E001D  01A34 P.ARL:  .LONG   17694749
                                                    00000000' 01A38        .ADDRESS P.ARM
65 74 20 61 20 73 69 20 65 63 69 76 65  64  20  01A3C P.ARO:  .ASCII  \ device is a template only,\<0>
      00 2C 79 6C 6E 6F 20 65 74 61 6C  70  6D  01A4B
                                                    010E001B  01A58 P.ARN:  .LONG   17694747
                                                    00000000' 01A5C        .ADDRESS P.ARO
                                             00 53  41  21  01A60 P.ARQ:  .ASCII  \!AS\<0>
                                                    010E0003  01A64 P.ARP:  .LONG   17694723
                                                    00000000' 01A68        .ADDRESS P.ARQ
                                             00 49  25  21  01A6C P.ARS:  .ASCII  \!%I\<0>
                                                    010E0003  01A70 P.ARR:  .LONG   17694723
                                                    00000000' 01A74        .ADDRESS P.ARS
21 3A 47 2C 53 41 21 3A 4F 2C 53 41 21  3A  53  01A78 P.ARU:  .ASCII  \S:!AS,O:!AS,G:!AS,W:!AS\<0>
                  00 53 41 21 3A 57 2C  53  41  01A87
                                                    010E0017  01A90 P.ART:  .LONG   17694743
                                                    00000000' 01A94        .ADDRESS P.ARU
74 6E 75 6F 63 20 72 6F 72 72 45 20 20  20  20  01A98 P.ARW:  .ASCII  \    Error count!21UW    Operations compl\
74 61 72 65 70 4F 20 20 20 20 57 55 31  32  21  01AA7
                  6C 70 6D 6F 63 20 73 6E 6F  69  01AB6
                  00 00 00 4C 55 39 31 21 64 65  74  65  01AC0        .ASCII  \eted!19UL\<0><0><0>
                                                    010E0031  01ACC P.ARV:  .LONG   17694769
                                                    00000000' 01AD0        .ADDRESS P.ARW
65 6F 6F 72 70 20 72 65 6E 77 4F 20 20  20  20  01AD4 P.ARY:  .ASCII  \    Owner process  !#< !>"'!AF"    Owner \
22 46 41 21 22 3E 21 20 3C 23 21 20 20  73  73  01AE3
                  20 72 65 6E 77 4F 20 20 20 20  01AF2
```

```
             53 41 21 3E 21 20 3C 23 21 43 49 55  01AFC          .ASCII  \UIC!#< !>!AS\                         ;
                                         010E0034  01B08 P.ARX:   .LONG   17694772                               ;
                                         00000000' 01B0C          .ADDRESS P.ARY                                 ;
65 63 6F 72 70 20 72 65 6E 77 4F 20 20 20 20  01B10 P.ASA:   .ASCII  \    Owner process ID!16XL    Dev Prot!#<\ ;
44 20 20 20 20 4C 58 36 31 21 44 49 20 73 73  01B1F                                                  ;
               3C 23 21 74 6F 72 50 20 76 65  01B2E                                                  ;
                  00 00 53 41 21 3E 21 20     01B38          .ASCII  \ !>!AS\<0><0>                         ;
                                         010E002E  01B40 P.ARZ:   .LONG   17694766                               ;
                                         00000000' 01B44          .ADDRESS P.ASA                                 ;
63 20 65 63 6E 65 72 65 66 65 52 20 20 20 20  01B48 P.ASC:   .ASCII  \    Reference count!17UL    Default buff\ ;
65 44 20 20 20 20 4C 55 37 31 21 74 6E 75 6F  01B57                                                  ;
               66 66 75 62 20 74 6C 75 61 66  01B66                                                  ;
         57 55 30 32 21 65 7A 69 73 20 72 65  01B70          .ASCII  \er size!20UW\                         ;
                                         010E0034  01B7C P.ASB:   .LONG   17694772                               ;
                                         00000000' 01B80          .ADDRESS P.ASC                                 ;

                                                               .EXTRN  SYS$FAOL, EXE$IPID_TO_EPID

                                                               .PSECT  $CODE$,NOWRT,2

                                           07FC 00000 COMMON_DISPLAY:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10       ; 2318
                     5A 00000000G   00  9E 00002          MOVAB   SHOW$WRITE_LINE, R10                   ;
                     59 00000000G   00  9E 00009          MOVAB   SYS$FAOL, R9                           ;
                     58       EF29  CF  9E 00010          MOVAB   ADD_TO_LIST_RTN, R8                    ;
                     5E       FCA0  CE  9E 00015          MOVAB   -864(SP), SP                           ;
          D8   AD    0000'  CF  28  28 0001A          MOVC3   #40, P.AOC, DESC_VECTOR                ; 2345
                     53         04  AC  7D 00021          MOVQ    ARGLIST, R3                            ; 2358
                04   A3         06  A4  9A 00025          MOVZBL  6(R4), 4(R3)                           ;
                                04  A3  D7 0002A          DECL    4(R3)                                  ;
                08   A3         09  A4  9E 0002D          MOVAB   9(R4), 8(R3)                           ; 2359
                0C   A3         1C  A4  9A 00032          MOVZBL  28(R4), 12(R3)                         ; 2361
                10   A3         1D  A4  9E 00037          MOVAB   29(R4), 16(R3)                         ; 2362
                D8   AD         50  8F  9A 0003C          MOVZBL  #80, DESC_VECTOR                       ; 2367
                DC   AD         18  AE  9E 00041          MOVAB   FAO_BUFF, DESC_VECTOR+4                ; 2368
                04   AE         D8  AD  9E 00046          MOVAB   DESC_VECTOR, PTR                       ; 2369
              0000'  CF             04  D0 0004B          MOVL    #4, LEADING_BLANKS                     ; 2370
                                    53  DD 00050          PUSHL   R3                                     ; 2388
                                D8  AD  9F 00052          PUSHAB  DESC_VECTOR                            ;
                                D8  AD  9F 00055          PUSHAB  DESC_VECTOR                            ;
                     57         18  A3  9E 00058          MOVAB   24(R3), R7                             ;
                     55         70  A4  9E 0005C          MOVAB   112(R4), R5                            ;
          17         65             0E  E1 00060          BBC     #14, (R5), 2$                          ;
                                54  A4  D5 00064          TSTL    84(R4)                                 ;
                                    12  13 00067          BEQL    2$                                     ;
                                    67  D5 00069          TSTL    (R7)                                   ;
                                    07  13 0006B          BEQL    1$                                     ;
                     50      0000'  CF  9E 0006D          MOVAB   P.AOD, R0                              ;
                                    17  11 00072          BRB     4$                                     ;
                     50      0000'  CF  9E 00074 1$:      MOVAB   P.AOF, R0                              ;
                                    10  11 00079          BRB     4$                                     ;
                                    67  D5 0007B 2$:      TSTL    (R7)                                   ;
                                    07  13 0007D          BEQL    3$                                     ;
                     50      0000'  CF  9E 0007F          MOVAB   P.AOH, R0                              ;
                                    05  11 00084          BRB     4$                                     ;
                     50      0000'  CF  9E 00086 3$:      MOVAB   P.AOJ, R0                              ;
                                    50  DD 0008B 4$:      PUSHL   R0                                     ;
```

```
                69          04  FB 0008D          CALLS   #4, SYS$FAOL                                    ;
                56    0088  C4  9E 00090          MOVAB   136(R4), R6                             ; 2393
            0A  66          04  E1 00095          BBC     #4, (R6), 5$                            ;
                7E      50  8F  9A 00099          MOVZBL  #80, -(SP)                              ; 2394
                      0000' CF  9F 0009D          PUSHAB  P.AOL                                   ;
                            08  11 000A1          BRB     6$                                      ;
                7E      50  8F  9A 000A3 5$:      MOVZBL  #80, -(SP)                              ; 2395
                      0000' CF  9F 000A7          PUSHAB  P.AON                                   ;
                        0C  AE  9F 000AB 6$:      PUSHAB  PTR                                     ;
                68          03  FB 000AE          CALLS   #3, ADD_TO_LIST_RTN                     ;
            0E  66          06  E1 000B1          BBC     #6, (R6), 7$                            ; 2397
                7E      50  8F  9A 000B5          MOVZBL  #80, -(SP)                              ; 2398
                      0000' CF  9F 000B9          PUSHAB  P.AOP                                   ;
                        0C  AE  9F 000BD          PUSHAB  PTR                                     ;
                68          03  FB 000C0          CALLS   #3, ADD_TO_LIST_RTN                     ;
            0E  04  A4          01  E0 000C3 7$:      BBS     #1, 4(R4), 8$                       ; 2400
                7E      50  8F  9A 000C8          MOVZBL  #80, -(SP)                              ; 2401
                      0000' CF  9F 000CC          PUSHAB  P.AOR                                   ;
                        0C  AE  9F 000D0          PUSHAB  PTR                                     ;
                68          03  FB 000D3          CALLS   #3, ADD_TO_LIST_RTN                     ;
            0E  65          12  E0 000D6 8$:      BBS     #18, (R5), 9$                           ; 2403
                7E      50  8F  9A 000DA          MOVZBL  #80, -(SP)                              ; 2404
                      0000' CF  9F 000DE          PUSHAB  P.AOT                                   ;
                        0C  AE  9F 000E2          PUSHAB  PTR                                     ;
                68          03  FB 000E5          CALLS   #3, ADD_TO_LIST_RTN                     ;
            0E  65          17  E1 000E8 9$:      BBC     #23, (R5), 10$                          ; 2406
                7E      50  8F  9A 000EC          MOVZBL  #80, -(SP)                              ; 2407
                      0000' CF  9F 000F0          PUSHAB  P.AOV                                   ;
                        0C  AE  9F 000F4          PUSHAB  PTR                                     ;
                68          03  FB 000F7          CALLS   #3, ADD_TO_LIST_RTN                     ;
        12  66          0A  E1 000FA 10$:      BBC     #10, (R6), 11$                          ; 2409
            0E  65          17  E1 000FE          BBC     #23, (R5), 11$                          ; 2411
                7E      50  8F  9A 00102          MOVZBL  #80, -(SP)                              ; 2412
                      0000' CF  9F 00106          PUSHAB  P.AOX                                   ;
                        0C  AE  9F 0010A          PUSHAB  PTR                                     ;
                68          03  FB 0010D          CALLS   #3, ADD_TO_LIST_RTN                     ;
        03  65          13  E0 00110 11$:      BBS     #19, (R5), 12$                          ; 2414
                        0082 31  00114          BRW     21$                                     ;
                0A      03  A5  E9 00117 12$:      BLBC    3(R5), 13$                          ; 2417
                7E      50  8F  9A 0011B          MOVZBL  #80, -(SP)                              ; 2418
                      0000' CF  9F 0011F          PUSHAB  P.AOZ                                   ;
                            08  11 00123          BRB     14$                                     ;
                7E      50  8F  9A 00125 13$:      MOVZBL  #80, -(SP)                              ; 2419
                      0000' CF  9F 00129          PUSHAB  P.APB                                   ;
                        0C  AE  9F 0012D 14$:      PUSHAB  PTR                                     ;
                68          03  FB 00130          CALLS   #3, ADD_TO_LIST_RTN                     ;
            0E  65          19  E1 00133          BBC     #25, (R5), 15$                          ; 2420
                7E      50  8F  9A 00137          MOVZBL  #80, -(SP)                              ; 2421
                      0000' CF  9F 0013B          PUSHAB  P.APD                                   ;
                        0C  AE  9F 0013F          PUSHAB  PTR                                     ;
                68          03  FB 00142          CALLS   #3, ADD_TO_LIST_RTN                     ;
            0A  66          0E  E1 00145 15$:      BBC     #14, (R6), 16$                          ; 2428
                7E      50  8F  9A 00149          MOVZBL  #80, -(SP)                              ; 2430
                      0000' CF  9F 0014D          PUSHAB  P.APF                                   ;
                            2E  11 00151          BRB     19$                                     ;
                            51  D4 00153 16$:      CLRL    R1                                      ; 2434
                01    009D  C4  91 00155          CMPB    157(R4), #1                             ;
```

```
                        02  12 0015A          BNEQ    17$                                 ;
                        51  D6 0015C          INCL    R1                                  ;
                        50  D4 0015E 17$:     CLRL    R0                                  ;
                02      009D  C4 91 00160      CMPB    157(R4), #2                         ;
                        02  12 00165          BNEQ    18$                                 ;
                        50  D6 00167          INCL    R0                                  ;
                50          51  C8 00169 18$:     BISL2   R1, R0                          ;
        52      66      01      0B  EF 0016C      EXTZV   #11, #1, (R6), R2               ;
                50          52  CA 00171        BICL2   R2, R0                          ;
                01          50  D1 00174        CMPL    R0, #1                          ; 2433
                        0E  12 00177          BNEQ    20$                                 ;
                7E      50  8F  9A 00179      MOVZBL  #80, -(SP)                          ; 2436
                    0000'   CF  9F 0017D      PUSHAB  P.APH                               ;
                        0C  AE  9F 00181 19$:     PUSHAB  PTR                             ;
                68          03  FB 00184      CALLS   #3, ADD_TO_LIST_RTN             ;
                            66  B5 00187 20$:     TSTW    (R6)                            ; 2438
                        0E  18 00189          BGEQ    21$                                 ;
                7E      50  8F  9A 0018B      MOVZBL  #80, -(SP)                          ; 2439
                    0000'   CF  9F 0018F      PUSHAB  P.APJ                               ;
                        0C  AE  9F 00193      PUSHAB  PTR                             ;
                68          03  FB 00196      CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      65          15  E1 00199 21$:     BBC     #21, (R5), 22$                  ; 2442
                7E      50  8F  9A 0019D      MOVZBL  #80, -(SP)                          ; 2443
                    0000'   CF  9F 001A1      PUSHAB  P.APL                               ;
                        0C  AE  9F 001A5      PUSHAB  PTR                             ;
                68          03  FB 001A8      CALLS   #3, ADD_TO_LIST_RTN             ;
                            65  95 001AB 22$:     TSTB    (R5)                            ; 2445
                        0E  18 001AD          BGEQ    23$                                 ;
                7E      50  8F  9A 001AF      MOVZBL  #80, -(SP)                          ; 2446
                    0000'   CF  9F 001B3      PUSHAB  P.APN                               ;
                        0C  AE  9F 001B7      PUSHAB  PTR                             ;
                68          03  FB 001BA      CALLS   #3, ADD_TO_LIST_RTN             ;
                            65  B5 001BD 23$:     TSTW    (R5)                            ; 2448
                        0E  18 001BF          BGEQ    24$                                 ;
                7E      50  8F  9A 001C1      MOVZBL  #80, -(SP)                          ; 2449
                    0000'   CF  9F 001C5      PUSHAB  P.APP                               ;
                        0C  AE  9F 001C9      PUSHAB  PTR                             ;
                68          03  FB 001CC      CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      65          1E  E1 001CF 24$:     BBC     #30, (R5), 25$                  ; 2451
                7E      50  8F  9A 001D3      MOVZBL  #80, -(SP)                          ; 2452
                    0000'   CF  9F 001D7      PUSHAB  P.APR                               ;
                        0C  AE  9F 001DB      PUSHAB  PTR                             ;
                68          03  FB 001DE      CALLS   #3, ADD_TO_LIST_RTN             ;
                            65  D5 001E1 25$:     TSTL    (R5)                            ; 2454
                        0E  18 001E3          BGEQ    26$                                 ;
                7E      50  8F  9A 001E5      MOVZBL  #80, -(SP)                          ; 2455
                    0000'   CF  9F 001E9      PUSHAB  P.APT                               ;
                        0C  AE  9F 001ED      PUSHAB  PTR                             ;
                68          03  FB 001F0      CALLS   #3, ADD_TO_LIST_RTN             ;
                0E          65  E9 001F3 26$:     BLBC    (R5), 27$                       ; 2457
                7E      50  8F  9A 001F6      MOVZBL  #80, -(SP)                          ; 2458
                    0000'   CF  9F 001FA      PUSHAB  P.APV                               ;
                        0C  AE  9F 001FE      PUSHAB  PTR                             ;
                68          03  FB 00201      CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      65          01  E1 00204 27$:     BBC     #1, (R5), 28$                   ; 2460
                7E      50  8F  9A 00208      MOVZBL  #80, -(SP)                          ; 2461
                    0000'   CF  9F 0020C      PUSHAB  P.APX                               ;
```

```
                 0C   AE   9F 00210          PUSHAB  PTR
             68       03   FB 00213          CALLS   #3, ADD_TO_LIST_RTN
        0E   65       0D   E1 00216 28$:     BBC     #13, (R5), 29$                  ;2463
             7E   50  8F   9A 0021A          MOVZBL  #80, -(SP)                      ;2464
                  0000' CF   9F 0021E          PUSHAB  P.APZ
                 0C   AE   9F 00222          PUSHAB  PTR
             68       03   FB 00225          CALLS   #3, ADD_TO_LIST_RTN
        0E   65       0E   E1 00228 29$:     BBC     #14, (R5), 30$                  ;2466
             7E   50  8F   9A 0022C          MOVZBL  #80, -(SP)                      ;2467
                  0000' CF   9F 00230          PUSHAB  P.AQB
                 0C   AE   9F 00234          PUSHAB  PTR
             68       03   FB 00237          CALLS   #3, ADD_TO_LIST_RTN
             0E   02  A5   E9 0023A 30$:     BLBC    2(R5), 31$                      ;2469
             7E   50  8F   9A 0023E          MOVZBL  #80, -(SP)                      ;2470
                  0000' CF   9F 00242          PUSHAB  P.AQD
                 0C   AE   9F 00246          PUSHAB  PTR
             68       03   FB 00249          CALLS   #3, ADD_TO_LIST_RTN
        0E   65       1D   E1 0024C 31$:     BBC     #29, (R5), 32$                  ;2472
             7E   50  8F   9A 00250          MOVZBL  #80, -(SP)                      ;2473
                  0000' CF   9F 00254          PUSHAB  P.AQF
                 0C   AE   9F 00258          PUSHAB  PTR
             68       03   FB 0025B          CALLS   #3, ADD_TO_LIST_RTN
             52   74  A4   9E 0025E 32$:     MOVAB   116(R4), R2                     ;2475
             12       62   E9 00262          BLBC    (R2), 33$
                      62   95 00265          TSTB    (R2)                            ;2476
                      0E   19 00267          BLSS    33$
             7E   50  8F   9A 00269          MOVZBL  #80, -(SP)                      ;2477
                  0000' CF   9F 0026D          PUSHAB  P.AQH
                 0C   AE   9F 00271          PUSHAB  PTR
             68       03   FB 00274          CALLS   #3, ADD_TO_LIST_RTN
             0E   01  A2   E9 00277 33$:     BLBC    1(R2), 34$                      ;2479
             7E   50  8F   9A 0027B          MOVZBL  #80, -(SP)                      ;2480
                  0000' CF   9F 0027F          PUSHAB  P.AQJ
                 0C   AE   9F 00283          PUSHAB  PTR
             68       03   FB 00286          CALLS   #3, ADD_TO_LIST_RTN
        0E   62       06   E1 00289 34$:     BBC     #6, (R2), 35$                   ;2482
             7E   50  8F   9A 0028D          MOVZBL  #80, -(SP)                      ;2483
                  0000' CF   9F 00291          PUSHAB  P.AQL
                 0C   AE   9F 00295          PUSHAB  PTR
             68       03   FB 00298          CALLS   #3, ADD_TO_LIST_RTN
        0E   04  A4       04   E1 0029B 35$:     BBC     #4, 4(R2), 36$              ;2485
             7E   50  8F   9A 002A0          MOVZBL  #80, -(SP)                      ;2486
                  0000' CF   9F 002A4          PUSHAB  P.AQN
                 0C   AE   9F 002A8          PUSHAB  PTR
             68       03   FB 002AB          CALLS   #3, ADD_TO_LIST_RTN
                      62   95 002AE 36$:     TSTB    (R2)                            ;2488
                      0E   18 002B0          BGEQ    37$
             7E   50  8F   9A 002B2          MOVZBL  #80, -(SP)                      ;2489
                  0000' CF   9F 002B6          PUSHAB  P.AQP
                 0C   AE   9F 002BA          PUSHAB  PTR
             68       03   FB 002BD          CALLS   #3, ADD_TO_LIST_RTN
        0E   65       06   E1 002C0 37$:     BBC     #6, (R5), 38$                   ;2491
             7E   50  8F   9A 002C4          MOVZBL  #80, -(SP)                      ;2492
                  0000' CF   9F 002C8          PUSHAB  P.AQR
                 0C   AE   9F 002CC          PUSHAB  PTR
             68       03   FB 002CF          CALLS   #3, ADD_TO_LIST_RTN
        0E   65       16   E1 002D2 38$:     BBC     #22, (R5), 39$                  ;2494
```

```
                7E      50   8F  9A 002D6          MOVZBL  #80, -(SP)                       ; 2495
              0000'     CF   9F 002DA             PUSHAB  P.AQT                           ;
                0C      AE   9F 002DE             PUSHAB  PTR                             ;
                68           03 FB 002E1          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      62           01 E1 002E4 39$:     BBC     #1, (R2), 40$                   ; 2497
                7E      50   8F  9A 002E8          MOVZBL  #80, -(SP)                      ; 2498
              0000'     CF   9F 002EC             PUSHAB  P.AQV                           ;
                0C      AE   9F 002F0             PUSHAB  PTR                             ;
                68           03 FB 002F3          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      65           11 E1 002F6 40$:     BBC     #17, (R5), 41$                  ; 2500
                7E      50   8F  9A 002FA          MOVZBL  #80, -(SP)                      ; 2501
              0000'     CF   9F 002FE             PUSHAB  P.AQX                           ;
                0C      AE   9F 00302             PUSHAB  PTR                             ;
                68           03 FB 00305          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      65           14 E1 00308 41$:     BBC     #20, (R5), 42$                  ; 2503
                7E      50   8F  9A 0030C          MOVZBL  #80, -(SP)                      ; 2504
              0000'     CF   9F 00310             PUSHAB  P.AQZ                           ;
                0C      AE   9F 00314             PUSHAB  PTR                             ;
                68           03 FB 00317          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      66           02 E1 0031A 42$:     BBC     #2, (R6), 43$                   ; 2506
                7E      50   8F  9A 0031E          MOVZBL  #80, -(SP)                      ; 2507
              0000'     CF   9F 00322             PUSHAB  P.ARB                           ;
                0C      AE   9F 00326             PUSHAB  PTR                             ;
                68           03 FB 00329          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      0098    C4   03 E1 0032C 43$:     BBC     #3, 152(R4), 44$                ; 2509
                7E      50   8F  9A 00332          MOVZBL  #80, -(SP)                      ; 2510
              0000'     CF   9F 00336             PUSHAB  P.ARD                           ;
                0C      AE   9F 0033A             PUSHAB  PTR                             ;
                68           03 FB 0033D          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      66           03 E1 00340 44$:     BBC     #3, (R6), 45$                   ; 2512
                7E      50   8F  9A 00344          MOVZBL  #80, -(SP)                      ; 2513
              0000'     CF   9F 00348             PUSHAB  P.ARF                           ;
                0C      AE   9F 0034C             PUSHAB  PTR                             ;
                68           03 FB 0034F          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      66           05 E1 00352 45$:     BBC     #5, (R6), 46$                   ; 2515
                7E      50   8F  9A 00356          MOVZBL  #80, -(SP)                      ; 2516
              0000'     CF   9F 0035A             PUSHAB  P.ARH                           ;
                0C      AE   9F 0035E             PUSHAB  PTR                             ;
                68           03 FB 00361          CALLS   #3, ADD_TO_LIST_RTN             ;
                0E      01   A6 E9 00364 46$:     BLBC    1(R6), 47$                      ; 2518
                7E      50   8F  9A 00368          MOVZBL  #80, -(SP)                      ; 2519
              0000'     CF   9F 0036C             PUSHAB  P.ARJ                           ;
                0C      AE   9F 00370             PUSHAB  PTR                             ;
                68           03 FB 00373          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      66           09 E1 00376 47$:     BBC     #9, (R6), 48$                   ; 2521
                7E      50   8F  9A 0037A          MOVZBL  #80, -(SP)                      ; 2522
              0000'     CF   9F 0037E             PUSHAB  P.ARL                           ;
                0C      AE   9F 00382             PUSHAB  PTR                             ;
                68           03 FB 00385          CALLS   #3, ADD_TO_LIST_RTN             ;
        0E      66           0D E1 00388 48$:     BBC     #13, (R6), 49$                  ; 2524
                7E      50   8F  9A 0038C          MOVZBL  #80, -(SP)                      ; 2525
              0000'     CF   9F 00390             PUSHAB  P.ARN                           ;
                0C      AE   9F 00394             PUSHAB  PTR                             ;
                68           03 FB 00397          CALLS   #3, ADD_TO_LIST_RTN             ;
                53           DD 0039A 49$:        PUSHL   R3                              ; 2530
              0000'     CF   9F 0039C             PUSHAB  ASCID_NULL                      ;
                6A           02 FB 003A0          CALLS   #2, SHOW$WRITE_LINE             ;
```

```
                                52  D4 003A3                CLRL    I                                               ;2531
          50                    52  01 78 003A5 50$:        ASHL    #1, I, R0                                       ;2533
                                56  D4 003A9                CLRL    R6
                             E0 AD40  D5 003AB              TSTL    DESC_VECTOR+8[R0]
                                16  12 003AF                BNEQ    51$
                                56  D6 003B1                INCL    R6
          51                    52  01 78 003B3             ASHL    #1, I, R1                                       ;2537
          50                    52  01 78 003B7             ASHL    #1, I, R0                                       ;2538
          50            DC AD41     D8 AD40  C1 003BB       ADDL3   DESC_VECTOR[R0], DESC_VECTOR+4[R1], R0
                        FF   A0        2E  90 003C3         MOVB    #46, -1(R0)                                     ;2539
          50                    52  01 78 003C7 51$:        ASHL    #1, I, R0                                       ;2541
                                63     D8 AD40  DE 003CB    MOVAL   DESC_VECTOR[R0], (R3)
                                53  DD 003D0                PUSHL   R3                                              ;2542
                          0000'  CF  9F 003D2               PUSHAB  P.ARP
                                6A  02  FB 003D6            CALLS   #2, SHOW$WRITE_LINE
                                04  56  E8 003D9            BLBS    R6, 52$                                         ;2543
          C5                    52  02  F3 003DC            AOBLEQ  #2, I, 50$
                                63  0092  C4  3C 003E0 52$: MOVZWL  146(R4), (R3)                                   ;2550
                          04    A3  008C  C4  D0 003E5      MOVL    140(R4), 4(R3)                                  ;2551
                          08    A3    60  A4  9A 003EB      MOVZBL  96(R4), 8(R3)                                   ;2552
          08    A3                0F    08  A3  C3 003F0    SUBL3   8(R3), #15, 8(R3)
                          0C    A3    60  A4  9A 003F6      MOVZBL  96(R4), 12(R3)                                  ;2553
                          10    A3    61  A4  9E 003FB      MOVAB   97(R4), 16(R3)                                  ;2554
                          D8    AD    50  8F  9A 00400      MOVZBL  #80, FAO_DESC                                   ;2555
                          DC    AD    18  AE  9E 00405      MOVAB   FAO_BUFF, FAO_DESC+4                            ;2556
                                6E    58  A4  D0 0040A      MOVL    88(R4), (SP)                                    ;2560
                                      5E  DD 0040E          PUSHL   SP
                                      D8  AD  9F 00410      PUSHAB  FAO_DESC
                                      D8  AD  9F 00413      PUSHAB  FAO_DESC
                                    0000'  CF  9F 00416     PUSHAB  P.ARR
                                69        04  FB 0041A      CALLS   #4, SYS$FAOL
                                1E    D8  AD  D1 0041D      CMPL    FAO_DESC, #30                                   ;2561
                                      05  19 00421          BLSS    53$
                                50        01  D0 00423      MOVL    #1, R0
                                      05  11 00426          BRB     54$
          50                    1E    D8  AD  C3 00428 53$: SUBL3   FAO_DESC, #30, R0                               ;2562
                          14    A3        50  D0 0042D 54$: MOVL    R0, 20(R3)                                      ;2561
                                67    D8  AD  9E 00431      MOVAB   FAO_DESC, (R7)                                  ;2563
                                50    5C  A4  D0 00435      MOVL    92(R4), R0                                      ;2569
                                  00000000G   00  16 00439  JSB     EXE$IPID_TO_EPID
                          1C    A3        50  D0 0043F      MOVL    R0, 28(R3)
                          24    A3    E0  AD  9E 00443      MOVAB   PROT_DESC, 36(R3)                               ;2571
          09                    65        0D  E1 00448      BBC     #13, (R5), 55$                                  ;2572
                          20    A3        01  D0 0044C      MOVL    #1, 32(R3)                                      ;2573
                                      E0  AD  D4 00450      CLRL    PROT_DESC
                                      36  11 00453          BRB     58$
          04                    65        0E  E1 00455 55$: BBC     #14, (R5), 56$                                  ;2579
                                      50  D4 00459          CLRL    KEY                                             ;2580
                                      03  11 0045B          BRB     57$
                                50        01  D0 0045D 56$: MOVL    #1, KEY                                         ;2581
                                      50  DD 00460 57$:     PUSHL   KEY                                             ;2582
                                7E    0084  C4  3C 00462    MOVZWL  132(R4), -(SP)
                                      10  AE  9F 00467      PUSHAB  PROT_ADDR
                    00000000G   00        03  FB 0046A      CALLS   #3, EXPAND_PROT
                          E0    AD    50  8F  9A 00471      MOVZBL  #80, PROT_DESC                                  ;2583
                          E4    AD    68  AE  9E 00476      MOVAB   FAO_BUFF+80, PROT_DESC+4                        ;2584
                                      08  AE  9F 0047B      PUSHAB  PROT_ADDR                                       ;2588
```

```
                                    E0  AD  9F 0047E          PUSHAB  PROT_DESC                                ;
                                    E0  AD  9F 00481          PUSHAB  PROT_DESC                               ;
                                0000'   CF  9F 00484          PUSHAB  P.ART                                   ;
                                    69      04  FB 00488      CALLS   #4, SYS$FAOL                            ;
                20  A3                1F    E0  AD  C3 0048B 58$:    SUBL3   PROT_DESC, #31, 32(R3)          ; 2590
                            28  A3      0086  C4  3C 00491          MOVZWL  134(R4), 40(R3)                 ; 2591
                            2C  A3        7A  A4  3C 00497          MOVZWL  122(R4), 44(R3)                 ; 2592
                                          28  A3  9F 0049C          PUSHAB  40(R3)                          ; 2602
                                    0000'   CF  9F 0049F          PUSHAB  P.ASB                           ; 2600
                                          1C  A3  9F 004A3          PUSHAB  28(R3)                          ;
                                    0000'   CF  9F 004A6          PUSHAB  P.ARZ                           ; 2598
                                          08  A3  9F 004AA          PUSHAB  8(R3)                           ;
                                    0000'   CF  9F 004AD          PUSHAB  P.ARX                           ; 2596
                                                53  DD 004B1      PUSHL   R3                              ; 2602
                                    0000'   CF  9F 004B3          PUSHAB  P.ARV                           ; 2594
                                                53  DD 004B7      PUSHL   R3                              ; 2602
                                    0000'   CF  9F 004B9          PUSHAB  ASCID_NULL                      ; 2594
                                        6A      0A  FB 004BD      CALLS   #10, SHOW$WRITE_LINE            ; 2602
                                                04 004C0          RET                                     ; 2605

; Routine Size:  1217 bytes,    Routine Base:  $CODE$ + 10C3
```

```
; 2236          2606  1 END
; 2237          2607  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY

;       Name                    Bytes                   Attributes

;  $OWN$                         492  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                       7044  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                       5508  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       217         1      1000          00:01.9


;                                   COMMAND QUALIFIERS

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

; Size:          5508 code + 7536 data bytes
; Run Time:         01:57.5
; Elapsed Time:     05:27.0
; Lines/CPU Min:     1330
; Lexemes/CPU-Min: 31932
; Memory Used:  488 pages
; Compilation Complete
```

SHODEVPRT
LIS

SHODEVUTL
LIS

SHODEV
LIS

SHODEVCLU
LIS