```
                                            I 3

   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
CCC             LLL                   III       UUU         UUU       TTT       LLL
   CCCCCCCCCCCC LLLLLLLLLLLLLL     IIIIIIIII    UUUUUUUUUUUUUUU       TTT       LLLLLLLLLLLLLL
   CCCCCCCCCCCC LLLLLLLLLLLLLL     IIIIIIIII    UUUUUUUUUUUUUUU       TTT       LLLLLLLLLLLLLL
   CCCCCCCCCCCC LLLLLLLLLLLLLL     IIIIIIIII    UUUUUUUUUUUUUUU       TTT       LLLLLLLLLLLLLL
```

```
                                     I 16
**FILE**ID**SHODEVUTL




 SSSSSSSS   HH      HH    OOOOOO    DDDDDDDD    EEEEEEEEEE  VV      VV  UU      UU  TTTTTTTTTT  LL
 SSSSSSSS   HH      HH    OOOOOO    DDDDDDDD    EEEEEEEEEE  VV      VV  UU      UU  TTTTTTTTTT  LL
SS          HH      HH  OO      OO  DD      DD  EE          VV      VV  UU      UU      TT      LL
SS          HH      HH  OO      OO  DD      DD  EE          VV      VV  UU      UU      TT      LL
SS          HH      HH  OO      OO  DD      DD  EE          VV      VV  UU      UU      TT      LL
SS          HH      HH  OO      OO  DD      DD  EE          VV      VV  UU      UU      TT      LL
 SSSSSS     HHHHHHHHHH  OO      OO  DD      DD  EEEEEEEE    VV      VV  UU      UU      TT      LL
 SSSSSS     HHHHHHHHHH  OO      OO  DD      DD  EEEEEEEE    VV      VV  UU      UU      TT      LL
        SS  HH      HH  OO      OO  DD      DD  EE          VV      VV  UU      UU      TT      LL
        SS  HH      HH  OO      OO  DD      DD  EE          VV      VV  UU      UU      TT      LL
        SS  HH      HH  OO      OO  DD      DD  EE            VV  VV    UU      UU      TT      LL
        SS  HH      HH  OO      OO  DD      DD  EE            VV  VV    UU      UU      TT      LL              ....
SSSSSSSS    HH      HH    OOOOOO    DDDDDDDD    EEEEEEEEEE      VV      UUUUUUUUUU      TT      LLLLLLLLLL      ....
SSSSSSSS    HH      HH    OOOOOO    DDDDDDDD    EEEEEEEEEE      VV      UUUUUUUUUU      TT      LLLLLLLLLL      ....

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
```

```
0 MODULE shodevutl(IDENT = 'V04-000',
0                     ADDRESSING_MODE (EXTERNAL = GENERAL)) =
0
1 BEGIN
1
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  SHOW utility
1 !
1 ! ABSTRACT:
1 !       This module contains the routines for the SHOW DEVICES command.
1 !
1 ! ENVIRONMENT:
1 !       VAX native, user and kernel mode
1 !
1 ! AUTHOR:  Gerry Smith            CREATION DATE:  28-Jul-1982
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-018 CWH3018                 CW Hobbs                24-Jul-1984
1 !               Add orb flags, max block, and ACP extent info to items
1 !               which are collected.
1 !
1 !       V03-017 LMP0221         L. Mark Pilant,         12-Apr-1984  15:01
1 !               Change UCB$L_OWNUIC to ORB$L_OWNER and UCB$W_VPROT to
1 !               ORB$W_PROT.
1 !
1 !       V03-016 CWH3016                 CW Hobbs                12-Apr-1984
1 !               Move test for /MOUNT and /ALLOC to SHODEVPRT, make the routine
1 !               suspicious of the PID in the UCB.
1 !
1 !       V03-015 CWH3015                 CW Hobbs                3-Mar-1984
1 !               Fix dual-path logic so that when getting data the "ddb"
1 !               parameter is always the primary ddb.  Also support allocation
```

```
1 !                 class device names for file-oriented devices and sorted
1 !                 device displays.
1 !
1 !         V03-014 CWH3014                      CW Hobbs                    29-Feb-1984
1 !                 Remove reference to D_L_VOLLKID, used during trial builds but
1 !                 not needed after EXE$DVI_FREEBLOCKS is built into the system.
1 !
1 !         V03-013 CWH3013                      CW Hobbs                    27-Feb-1984
1 !                 Collect more information for remote and dual-path devices.
1 !                 Fix linkages for calls to the exec, and add a handler to
1 !                 trap and dismiss kernel mode access violations.
1 !
1 !         V03-012 TCM0001                      Trudy C. Matthews           10-Oct-1983
1 !                 If there are two paths to the same device, find the name of
1 !                 the alternate path (i.e. the device's alias).
1 !
1 !         V03-011 GAS0178                      Gerry Smith                 7-Sep-1983
1 !                 Fix quota caching for ODS2 disks.  The quota cache size was
1 !                 being taken from the wrong cell.
1 !
1 !         V03-010 GAS0167                      Gerry Smith                 22-Aug-1983
1 !                 Fix the journal device name: get rid of the underscore that
1 !                 ioc$cvt_devnam returns, and make the device name into an
1 !                 ASCIC string.
1 !
1 !         V03-009 GAS0160                      Gerry Smith                 27-Jul-1983
1 !                 Show template devices by default.
1 !
1 !         V03-008 GAS0149                      Gerry Smith                 28-Jun-1983
1 !                 Use IOC$CVT_DEVNAM to obtain the device name.
1 !
1 !         V03-007 GAS0133                      Gerry Smith                 14-May-1983
1 !                 Add retention period, default extend quantity, default file
1 !                 protection.
1 !
1 !         V03-006 GAS0114                      Gerry Smith                 1-Apr-1983
1 !                 Modify the cluster_device logic so that less checking and
1 !                 testing is done in kernel mode.
1 !
1 !         V03-005 GAS0110                      Gerry Smith                 28-Feb-1983
1 !                 Add support for cluster devices.
1 !
1 !         V03-004 GAS0107                      Gerry Smith                 3-Feb-1983
1 !                 Add support for journals.
1 !
1 !         V03-003 GAS0106                      Gerry Smith                 24-Jan-1983
1 !                 In the case of multivolume sets, check to make sure that
1 !                 the volume is mounted.  Also tighten up the bounds checking.
1 !
1 !         V03-002 GAS00104                     Gerry Smith                 17-Jan-1983
1 !                 Fix the logic path for /ALLOCATED and /MOUNTED
1 !
1 !         V03-001 GAS00101                     Gerry Smith                 13-Jan-1983
1 !                 Only check for an RVN if the device is file-oriented.
1 !
1 !--
```

```
  1
  1 !
  1 ! Include files
  1 !
  1
  1 LIBRARY 'SYS$LIBRARY:LIB';                     ! VAX/VMS system definitions
  1 REQUIRE 'SRC$:SHOWDEF';                        ! SHOW common definitions
  1 REQUIRE 'SRC$:SHODEVDEF';                      ! SHOW DEVICES common definitions
  1 REQUIRE 'SHRLIB$:JNLDEFINT';                   ! Journal definitions
  1
  1 !
  1 ! Define the linkage for the routines to lock and unlock the I/O database,
  1 ! scan the I/O database, and obtain the device name.
  1 !
  1 LINKAGE
  1     IOLOCK = JSB (REGISTER = 4)
  1                 : NOPRESERVE(0,1,2,3,4,5) PRESERVE(6,7,8,9,10,11),
  1     CVTDEV = JSB (REGISTER = 0,                           ! Length of output buffer,
  1                   REGISTER = 1,                           ! Address of output buffer
  1                   REGISTER = 4,                           ! Format of device name
  1                   REGISTER = 5;                           ! Address of UCB
  1                   REGISTER = 1)                           ! Length of final name
  1                 : PRESERVE(0,2,3,4,5,6,7,8,9,10,11),
  1     IOSCAN = JSB (REGISTER = 11,                          ! Call with DDB,
  1                   REGISTER = 10;                          ! UCB,
  1                   REGISTER = 11,                          ! Return with DDB,
  1                   REGISTER = 10)                          ! UCB
  1                 : NOPRESERVE(0,10,11) PRESERVE(1,2,3,4,5,6,7,8,9);
  1
  1 !
  1 ! The following macro makes it easier to copy stuff to the scratch area.
  1 !
M 1 MACRO copy_data (source, dest) [item] =
  1     dest[%NAME('d_', item)] = .source[%NAME(source, '$', item)]%;
  1
```

```
1 FORWARD ROUTINE
1         kernel_handler;            ! Turn kernel mode signals to returns
1
1 FORWARD ROUTINE
1     io_scan,
1     utl_get_data;
1
1 EXTERNAL ROUTINE
1     show$write_line : NOVALUE,
1     exe$dvi_freeblocks,
1     sch$iolockr : IOLOCK,
1     sch$iounlock : IOLOCK,
1     ioc$cvt_devnam : CVTDEV,
1     ioc$scan_iodb_2p : IOSCAN;
1
1 EXTERNAL
1     scs$gq_config,
1     scs$ga_localsb,
1     sch$gl_maxpix,
1     sch$gl_pcbvec : REF VECTOR,
1     sch$gl_curpcb,
1     ioc$gl_devlist;
1
1 GLOBAL
1     kernel_accvio : VECTOR [4, LONG] ADDRESSING_MODE (GENERAL);
1
```

```
;  178          1577  1 GLOBAL ROUTINE kernel_handler (sig : REF BLOCK[,BYTE], mech : REF BLOCK[,BYTE]) =
;  179          1578  2 BEGIN
;  180          1579  2 !++
;  181          1580  2 !
;  182          1581  2 ! FUNCTIONAL DESCRIPTION:
;  183          1582  2 !
;  184          1583  2 !       This routine intercepts kernel mode signals.
;  185          1584  2 !
;  186          1585  2 ! INPUTS:
;  187          1586  2 !
;  188          1587  2 !       sig  - signal argument list
;  189          1588  2 !       mech - mechanism argument list
;  190          1589  2 !
;  191          1590  2 ! SIDE EFFECTS:
;  192          1591  2 !
;  193          1592  2 !       A return is made to user mode code.
;  194          1593  2 !--
;  195          1594  2
;  196          1595  2 EXTERNAL ROUTINE
;  197          1596  2     LIB$SIG_TO_RET : ADDRESSING_MODE (GENERAL);
;  198          1597  2
;  199          1598  2 ! If the signal name is an accvio, then clean up
;  200          1599  2 !
;  201          1600  2 IF .sig [chf$l_sig_name] EQL ss$_accvio          ! Is it an accvio?
;  202          1601  2 THEN
;  203          1602  3     BEGIN
;  204          1603  3     SCH$IOUNLOCK(.sch$gl_curpcb);                    ! Unlock I/O database
;  205          1604  3     SET_IPL(0);                                       ! Lower IPL
;  206          1605  3     CH$MOVE (4*4, sig[chf$l_sig_arg1], kernel_accvio[0]);
;  207          1606  3     RETURN LIB$SIG_TO_RET (.sig, .mech);             ! Convert signal to return
;  208          1607  2     END;
;  209          1608  2
;  210          1609  2 RETURN ss$_resignal;
;  211          1610  1 END;


                                                                        .TITLE  SHODEVUTL
                                                                        .IDENT  \V04-000\

                                                                        .PSECT  $GLOBAL$,NOEXE,2

                                                      00000 KERNEL_ACCVIO::
                                                                        .BLKB   16

                                                                        .EXTRN  SHOW$WRITE_LINE
                                                                        .EXTRN  EXE$DVI_FREEBLOCKS
                                                                        .EXTRN  SCH$IOLOCKR, SCH$IOUNLOCK
                                                                        .EXTRN  IOC$CVT_DEVNAM, IOC$SCAN_IODB_2P
                                                                        .EXTRN  SCS$GQ_CONFIG, SCS$GA_LOCALSB
                                                                        .EXTRN  SCH$GL_MAXPIX, SCH$GL_PCBVEC
                                                                        .EXTRN  SCH$GL_CURPCB, IOC$GL_DEVLIST
                                                                        .EXTRN  LIB$SIG_TO_RET

                                                                        .PSECT  $CODE$,NOWRT,2

                                                 0FFC 00000             .ENTRY  KERNEL_HANDLER, Save R2,R3,R4,R5,R6,R7,R8,-  ; 1577
                                                                                R9,R10,R11                                      ;
```

```
                         56         04   AC  D0 00002          MOVL    SIG, R6                                  ; 1600
                         0C         04   A6  D1 00006          CMPL    4(R6), #12                               ;
                                         26  12 0000A          BNEQ    1$                                       ;
                         54  00000000G   00  D0 0000C          MOVL    SCH$GL_CURPCB, R4                        ; 1603
                             00000000G   00  16 00013          JSB     SCH$IOUNLOCK                             ;
                         12              00  DA 00019          MTPR    #0, #18                                  ; 1604
        00000000'  00          08  A6        10  28 0001C          MOVC3   #16, 8(R6), KERNEL_ACCVIO                ; 1605
                                    08   AC  DD 00025          PUSHL   MECH                                     ; 1606
                                         56  DD 00028          PUSHL   R6                                       ;
                   00000000G  00         02  FB 0002A          CALLS   #2, LIB$SIG_TO_RET                       ;
                                         04  00031             RET                                              ;
                         50       0918   8F  3C 00032 1$:      MOVZWL  #2328, R0                                ; 1609
                                         04  00037             RET                                              ; 1610

; Routine Size:  56 bytes,    Routine Base:  $CODE$ + 0000
```

```
GLOBAL ROUTINE io_scan (node, device, unit, flags, data) =
BEGIN

!---
!
! This routine is called in KERNEL mode to scan the device data base and
! determine which devices to collect information about.  Once a likely
! candidate for data collection is determined, control is passed to
! another routine, UTL_GET_DATA, where, based on the type of device and
! the qualifiers selected, device-specific data is stuffed into the scratch
! area.  This continues until either the end of the device database is
! reached, or an error status (STATUS low bit clear) is obtained.  Typical
! reasons for an error status are running out of scratch area, or having
! obtained all the data that is required of the caller.
!
! Inputs
!       NODE      - address of ASCIC of node part of device name, or allocation
!                   class if FLAGS[DEVI$V_ALLOCLS]
!       DEVICE    - address of ASCIC of device part of device name
!       UNIT      - address of unit number. (-1 => no unit number)
!       FLAGS     - address of options longword
!       DATA      - address of scratch area.
!
! Outputs
!       DATA      - is full of all sorts of useful data about devices
!
!---

MAP
    data : REF VECTOR,
    node : REF VECTOR[,BYTE],
    device : REF VECTOR[,BYTE],
    flags : REF $BBLOCK;

LOCAL
    status,
    limit,
    ptr : REF VECTOR[,BYTE],                        ! Data area pointer
    scratch : REF $BBLOCK,                          ! Scratch pointer
    ucb : REF $BBLOCK,                              ! UCB pointer
    ddb : REF $BBLOCK,                              ! DDB pointer
    sb : REF $BBLOCK;                               ! System block pointer

!
! Trap anything weird, and turn it into a return
!
ENABLE
    kernel_handler;

!
! Set up the scratch area so that is can be addressed easily.  Also, calculate
! a limit toward the end of the scratch area, so that we don't write beyond the
! area.
!
scratch = data[1];                              ! Point to beginning of scratch area
limit = .data[0] + data[0] - d_k_length;        ! Set the limit

```

```
2  !
2  ! Lock the I/O data base.  Upon return from the call to SCH$IOLOCKR, the
2  ! IPL will be 2, so that pagefaults are still allowed.
2  !
2  SCH$IOLOCKR(.sch$gl_curpcb);                       ! Lock the I/O database
2
2  !
2  ! Start at the beginning of the I/O database and initiate the I/O scan.
2  !
2  status = IOC$SCAN_IODB_2P(0, 0; ddb, ucb);
2
2  !
2  ! For each UCB in the I/O database, determine if it might contain devices of
2  ! interest.  If so, then call the data-gathering dispatch routine.  Upon
2  ! return from the data-gathering, STATUS must be checked, to see if any
2  ! further scan is necessary.  If not, then exit the DDB/UCB loops.
2  !
2  WHILE .status DO                                   ! As long as the scan returns
3      BEGIN                                          ! a success, stay in the loop.
3      IF                                             ! For each device found, make
4          BEGIN                                      ! some checks.
4          IF .flags[devi$v_allocls]                  ! If an allocation class is desired
4          THEN
5              BEGIN
6              IF .ddb[ddb$l_allocls] EQL .(node[0])! If the allocation class matches
5              THEN true                              ! then the device is ok, otherwise
5              ELSE ucb = 0                           ! go to the next DDB.
5              END
4          ELSE
5              BEGIN
5              IF .node[0] EQL 0                      ! If no node specified, then
5              THEN true                              ! continue.
5              ELSE                                   ! Otherwise check to see if
6                  BEGIN                              ! this node is one we want.
6                  IF (sb = .ddb[ddb$l_sb]) EQL 0     ! If no node, go to
6                  THEN ucb = 0                       ! next DDB.
6                  ELSE
7                      BEGIN
7                      IF CH$EQL(.node[0], node[1],
7                              .(sb[sb$t_nodename])<0,8>, sb[sb$t_nodename] + 1)
7                      THEN true                      ! If nodenames match, good
7                      ELSE ucb = 0                   ! Else get next DDB
7                      END
6                  END
5              END
4          END
3      AND
4          BEGIN
4          IF .device[0] EQL 0                        ! If no device specified, then
4          THEN
5              BEGIN                                      ! Don't display mailbox
5              IF .$BBLOCK[ucb[ucb$l_devchar], dev$v_mbx]  ! UCB's, and get to
5              THEN ucb = 0                               ! next DDB
5              ELSE true
5              END
4          ELSE                                           ! If a device was
5              BEGIN                                      ! specified, check for
```

```
5               IF CH$EQL(.device[0], device[1],                ! a match.
5                       .device[0], ddb[ddb$t_name] + 1)
5               THEN true                                       ! If a match, good
5               ELSE ucb = 0                                    ! Otherwise, go to
5               END                                             ! next DDB
4           END
3       AND
4           BEGIN                                               ! If a unit specified,
4           IF .unit NEQ -1                                     ! check for a match
5           THEN (.unit EQL .ucb[ucb$w_unit])
4           ELSE true                                           ! If no unit, ok
4           END
3       THEN
4           BEGIN
4           IF .scratch GEQA .limit             ! Before getting data, check
4           THEN                                ! that there is room.
5               BEGIN                           ! If no room, set status to
5               status = SS$_VASFULL;           ! appropriate error
5               EXITLOOP                        ! and get out.
5               END;
4   !
4   ! Determine how much data to get.  If no complete device was specified,
4   ! return just information about this device.  However, if a complete device
4   ! was specified, check to see if this is perhaps a multi-volume set.  If so,
4   ! then return data about the entire set.
4   !
4   ! So, if no explicit device was given, or if the device is not file-oriented,
4   ! or there's no VCB, or there is no Relative Volume Table, then
4   ! collect data on one device.  Otherwise, rip thru the UCB list associated
4   ! with the RVT, and get data about each device in the set.
4   !
4           IF .unit EQL -1                                     ! If not explicit
4           OR NOT .$BBLOCK[ucb[ucb$l_devchar], dev$v_fod]      ! or not Files-11
4           OR
5               BEGIN
5               BIND vcb = ucb[ucb$l_vcb] : REF $BBLOCK;
5               IF .vcb EQL 0                                   ! or no VCB
5               THEN true
5               ELSE
6                   BEGIN
6                   IF .vcb[vcb$w_rvn] EQL 0                    ! or not an RVN
6                   THEN true                                   ! then do one
6                   ELSE false
6                   END
5               END
4           THEN
5               BEGIN
5               status = utl_get_data(.ucb, .ddb, .flags, .scratch, .data);
5                                                               ! Get device data
5               IF .status                                      ! If we got data,
5               THEN                                            ! update the pointer
6                   BEGIN
6                   IF .scratch[d_b_devclass] EQLU dc$_journal
6                   THEN scratch = .scratch+d_k_length;         ! Skip an extra
6                   scratch = .scratch + d_k_length;            ! block if journal
6                   END
```

```
5              ELSE status = 1;                          ! The only time FALSE
5                                                        ! is returned is if
5                                                        ! /MOUNTED or /ALLOCATED
5                                                        ! was specified and
5                                                        ! the device wasn't either
5              IF .unit NEQ -1                           ! If explicit device
5              AND .status                               ! (don't mask error)
5              THEN ucb = 0;                             ! then we're done with
5              END                                       ! this DDB.
4          ELSE
5              BEGIN
5              LOCAL
5                  vcb : REF $BBLOCK,
5                  rvt : REF $BBLOCK,
5                  ucblist : REF VECTOR;
5
5              vcb = .ucb[ucb$l_vcb];
5              rvt = .vcb[vcb$l_rvt];
5              ucblist = rvt[rvt$l_ucblst];
5
5              INCR index FROM 0 TO .rvt[rvt$b_nvols] - 1 DO
6                  BEGIN
6                  IF .scratch GEQA .limit               ! Check limit
7                  THEN (status = SS$_VASFULL; EXITLOOP)
6                  ELSE IF .ucblist[.index] NEQ 0        ! If volume mounted,
6                  THEN                                  ! get data
7                      BEGIN
7                      status = utl_get_data(.ucblist[.index], .ddb, .flags, .scratch, .data);
7                      IF .status
7                      THEN
8                          BEGIN
8                          IF .scratch[d_b_devclass] EQLU dc$_journal
8                          THEN scratch = .scratch + d_k_length;
8                          scratch = .scratch + d_k_length;
7                          END;
6                      END;
5                  END;
5              status = 0;                   ! To indicate finished with
4              END;                          ! this volume set
4          IF NOT .status THEN EXITLOOP;     ! Go away?
3          END;
3      status = IOC$SCAN_IODB_2P(.ddb, .ucb; ddb, ucb);
2      END;
2
2  scratch[d_t_device] = 0;                      ! To show end of list
2
2  !
2  ! Now to clean up.  Unlock the I/O database, then lower the IPL
2  ! to zero.
2  !
2  SCH$IOUNLOCK(.sch$gl_curpcb);                 ! Unlock I/O database
2  SET_IPL(0);                                   ! Lower IPL
2
2  IF .scratch EQLA data[1]                      ! If no data,
2  THEN status = SS$_NOSUCHDEV                   ! return an error
2  ELSE status = true;
2
```

```
:  441           1839  2 RETURN .status;                                    ! Return with status
:  442           1840  1 END;



                                           0FFC 00000            .ENTRY  IO_SCAN, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,- ; 1611
                                                                         R11                                         ;
                            5E          08  C2 00002             SUBL2   #8, SP                                      ;
                            6D    018A  CF  DE 00005             MOVAL   27$, (FP)                                   ; 1612
              04   AE   14  AC          04  C1 0000A             ADDL3   #4, DATA, 4(SP)                             ; 1665
                            57      04  AE  D0 00010             MOVL    4(SP), SCRATCH                              ;
                   56   14  BC      14  AC  C1 00014             ADDL3   DATA, @DATA, R6                             ; 1666
                            56    FEF9  C6  9E 0001A             MOVAB   -263(R6), LIMIT                             ;
                            54 00000000G 00 D0 0001F             MOVL    SCH$GL_CURPCB, R4                           ; 1672
                               00000000G 00 16 00026             JSB     SCH$IOLOCKR                                 ;
                                        5A  7C 0002C             CLRQ    R10                                         ; 1677
                               00000000G 00 16 0002E             JSB     IOC$SCAN_IODB_2P                            ;
                            58          50  D0 00034             MOVL    R0, STATUS                                  ;
                            6E      10  AC  D0 00037             MOVL    FLAGS, (SP)                                 ; 1689
                            55      04  AC  D0 0003B             MOVL    NODE, R5                                    ; 1692
                            6C          58  E9 0003F  1$:        BLBC    STATUS, 10$                                 ; 1685
                   59       6E          01  C1 00042             ADDL3   #1, (SP), R9                                ; 1689
                   08       69          04  E1 00046             BBC     #4, (R9), 3$                                ;
                            65      3C  AB  D1 0004A             CMPL    60(DDB), (R5)                               ; 1692
                                        1D  13 0004E  2$:        BEQL    4$                                          ;
                                        3A  11 00050             BRB     6$                                          ; 1694
                                        65  95 00052  3$:        TSTB    (R5)                                        ; 1698
                                        17  13 00054             BEQL    4$                                          ;
                   54               34  AB  D0 00056             MOVL    52(DDB), SB                                 ; 1702
                                        30  13 0005A             BEQL    6$                                          ;
                            51          65  9A 0005C             MOVZBL  (R5), R1                                    ; 1706
                            50      44  A4  9A 0005F             MOVZBL  68(SB), R0                                  ; 1707
        50         00   01  A5          51  2D 00063             CMPC5   R1, 1(R5), #0, R0, 69(SB)                   ; 1706
                                    45  A4     00069                                                                 ;
                                        E1  11 0006B             BRB     2$                                          ;
                   50               08  AC  D0 0006D  4$:        MOVL    DEVICE, R0                                  ; 1716
                                        60  95 00071             TSTB    (R0)                                        ;
                                        07  12 00073             BNEQ    5$                                          ;
                   17       3A  AA          04  E1 00075         BBC     #4, 58(UCB), 8$                             ; 1719
                                        10  11 0007A             BRB     6$                                          ; 1720
                            52          60  9A 0007C  5$:        MOVZBL  (R0), R2                                    ; 1725
                            51          60  9A 0007F             MOVZBL  (R0), R1                                    ; 1726
        51         00   01  A0          52  2D 00082             CMPC5   R2, 1(R0), #0, R1, 21(DDB)                  ; 1725
                                    15  AB     00088                                                                 ;
                                        05  13 0008A             BEQL    8$                                          ;
                                        5A  D4 0008C  6$:        CLRL    UCB                                         ; 1728
                                      00CF  31 0008E  7$:        BRW     23$                                         ;
                       FFFFFFFF  8F     0C  AC  D1 00091  8$:    CMPL    UNIT, #-1                                   ; 1733
                                        09  13 00099             BEQL    9$                                          ;
    0C  AC         54   AA          10      00  ED 0009B         CMPZV   #0, #16, 84(UCB), UNIT                      ; 1734
                                        EA  12 000A2             BNEQ    7$                                          ;
                            56          57  D1 000A4  9$:        CMPL    SCRATCH, LIMIT                              ; 1739
                                        08  1F 000A7             BLSSU   11$                                         ;
                            58    0244  8F  3C 000A9             MOVZWL  #580, STATUS                                ; 1742
                                      00BB  31 000AE  10$:       BRW     24$                                         ; 1741
```

```
          FFFFFFFF    8F      0C   AC   D1 000B1 11$:    CMPL    UNIT, #-1                              ; 1757
                                   10   13 000B9         BEQL    12$                                    ;
     0B        39     AA           06   E1 000BB         BBC     #6, 57(UCB), 12$                       ; 1758
                      50      34   AA   D0 000C0         MOVL    52(UCB), R0                            ; 1762
                                   05   13 000C4         BEQL    12$                                    ;
                              0E   A0   B5 000C6         TSTW    14(R0)                                 ; 1766
                                   3D   12 000C9         BNEQ    16$                                    ;
                              14   AC   DD 000CB 12$:    PUSHL   DATA                                   ; 1773
                                   57   DD 000CE         PUSHL   SCRATCH                                ;
                              10   AC   DD 000D0         PUSHL   FLAGS                                  ;
                      7E           5A   7D 000D3         MOVQ    UCB, -(SP)                             ;
              0000V   CF           05   FB 000D6         CALLS   #5, UTL_GET_DATA                       ;
                      58           50   D0 000DB         MOVL    R0, STATUS                             ;
                      13           58   E9 000DE         BLBC    STATUS, 14$                            ; 1775
                A1    8F      78   A7   91 000E1         CMPB    120(SCRATCH), #161                     ; 1778
                                   05   12 000E6         BNEQ    13$                                    ;
                      57    0107   C7   9E 000E8         MOVAB   263(R7), SCRATCH                       ; 1779
                      57    0107   C7   9E 000ED 13$:    MOVAB   263(R7), SCRATCH                       ; 1780
                                   03   11 000F2         BRB     15$                                    ; 1775
                      58           01   D0 000F4 14$:    MOVL    #1, STATUS                             ; 1782
          FFFFFFFF    8F      0C   AC   D1 000F7 15$:    CMPL    UNIT, #-1                              ; 1787
                                   5C   13 000FF         BEQL    22$                                    ;
                      68           58   E9 00101         BLBC    STATUS, 24$                            ; 1788
                                   5A   D4 00104         CLRL    UCB                                    ; 1789
                                   55   11 00106         BRB     22$                                    ; 1757
                      50      34   AA   D0 00108 16$:    MOVL    52(UCB), VCB                           ; 1798
                      50      20   A0   D0 0010C         MOVL    32(VCB), RVT                           ; 1799
                      52      44   A0   9E 00110         MOVAB   68(R0), UCBLIST                        ; 1800
                      59      0B   A0   9A 00114         MOVZBL  11(RVT), R9                            ; 1802
                      53           01   CE 00118         MNEGL   #1, INDEX                              ;
                                   3A   11 0011B         BRB     20$                                    ;
                      56           57   D1 0011D 17$:    CMPL    SCRATCH, LIMIT                         ; 1804
                                   07   1F 00120         BLSSU   18$                                    ;
                      58    0244   8F   3C 00122         MOVZWL  #580, STATUS                           ; 1805
                                   32   11 00127         BRB     21$                                    ;
                                 6243   D5 00129 18$:    TSTL    (UCBLIST)[INDEX]                       ; 1806
                                   29   13 0012C         BEQL    20$                                    ;
                              14   AC   DD 0012E         PUSHL   DATA                                   ; 1809
                                   57   DD 00131         PUSHL   SCRATCH                                ;
                              10   AC   DD 00133         PUSHL   FLAGS                                  ;
                                   5B   DD 00136         PUSHL   DDB                                    ;
                                 6243   DD 00138         PUSHL   (UCBLIST)[INDEX]                       ;
              0000V   CF           05   FB 0013B         CALLS   #5, UTL_GET_DATA                       ;
                      58           50   D0 00140         MOVL    R0, STATUS                             ;
                      11           58   E9 00143         BLBC    STATUS, 20$                            ; 1810
                A1    8F      78   A7   91 00146         CMPB    120(SCRATCH), #161                     ; 1813
                                   05   12 0014B         BNEQ    19$                                    ;
                      57    0107   C7   9E 0014D         MOVAB   263(R7), SCRATCH                       ; 1814
                      57    0107   C7   9E 00152 19$:    MOVAB   263(R7), SCRATCH                       ; 1815
     C2               53           59   F2 00157 20$:    AOBLSS  R9, INDEX, 17$                         ; 1802
                                   58   D4 0015B 21$:    CLRL    STATUS                                 ; 1819
                      0C           58   E9 0015D 22$:    BLBC    STATUS, 24$                            ; 1821
                       00000000G   00   16 00160 23$:    JSB     IOC$SCAN_IODB_2P                       ; 1823
                      58           50   D0 00166         MOVL    R0, STATUS                             ;
                                 FED3   31 00169         BRW     1$                                     ; 1685
                              08   A7   94 0016C 24$:    CLRB    8(SCRATCH)                             ; 1826
                      54 00000000G 00   D0 0016F         MOVL    SCH$GL_CURPCB, R4                      ; 1832
```

```
                              00000000G  00  16 00176           JSB      SCH$IOUNLOCK                           ;
                   12                    00  DA 0017C           MTPR     #0, #18                                ; 1833
           04      AE                    57  D1 0017F           CMPL     SCRATCH, 4(SP)                         ; 1835
                                         07  12 00183           BNEQ     25$                                    ;
                   58        0908        8F  3C 00185           MOVZWL   #2312, STATUS                          ; 1836
                                         03  11 0018A           BRB      26$                                    ;
                   58                    01  D0 0018C  25$:     MOVL     #1, STATUS                             ; 1837
                   50                    58  D0 0018F  26$:     MOVL     STATUS, R0                             ; 1839
                                             04 00192           RET                                             ; 1840
                                           0000 00193  27$:     .WORD    Save nothing                           ; 1612
                                         7E  D4 00195           CLRL     -(SP)                                  ;
                                         5E  DD 00197           PUSHL    SP                                     ;
                   7E          04        AC  7D 00199           MOVQ     4(AP), -(SP)                           ;
         FE26      CF                    03  FB 0019D           CALLS    #3, KERNEL_HANDLER                     ;
                                             04 001A2           RET                                             ;

; Routine Size:  419 bytes,    Routine Base:  $CODE$ + 0038
```

```
1 GLOBAL ROUTINE utl_get_data (in_ucb, in_ddb, flags, scratch, data) =
2 BEGIN
2
2 !---
2 !
2 ! This routine executes in KERNEL mode, and is called by IO_SCAN to dispatch
2 ! to specific data-gathering routines, based on the qualifiers and the type of
2 ! device.
2 !
2 ! Inputs
2 !       IN_UCB          - address of the UCB of the device of interest
2 !       IN_DDB          - address of the DDB whose UCB chain we are following
2 !       FLAGS           - pointer to flags longword
2 !       SCRATCH         - location of scratch area where data can be stored
2 !       DATA            - pointer to start of scratch area
2 !
2 ! Outputs
2 !       SCRATCH - has data possibly stored into it.  Also, the value of
2 !                 SCRATCH will have changed, to show the next place where
2 !                 data can be stored.
2 !
2 !---
2
2 MAP
2     data : REF VECTOR,
2     scratch : REF $BBLOCK,
2     flags : REF $BBLOCK;
2
2 LOCAL
2     status,
2     aqb : REF $BBLOCK,
2     ddb : REF $BBLOCK,
2     scr : REF $BBLOCK,
2     ucb : REF $BBLOCK,
2     orb : REF $BBLOCK,
2     vcb : REF $BBLOCK;
2
2 !
2 ! Move the input parameters to the local pointers.  Check if the ucb is marked as the class driver
2 ! copy, used for dual-pathed massbus disks.  If so, substitute the primary UCB and DDB for the
2 ! input parameters.
2 !
2 ucb = .in_ucb;
2 IF .$BBLOCK[ucb[ucb$l_devchar2], dev$v_cdp]      ! Is it the class driver path?
2 THEN ucb = .ucb[ucb$l_2p_altucb];                ! Get the "real" ucb address
2 orb = .ucb[ucb$l_orb];                           ! Save a pointer to the object's rights block
2 ddb = .ucb[ucb$l_ddb];                           ! Always use the ddb hanging from the ucb we are actually us
2 vcb = .ucb[ucb$l_vcb];                           ! Save a pointer to the volume control block
2
2 !
2 ! Collect data about this device.  Initialize the SHOW DEVICE
2 ! control areas in the scratch cell.
2 !
2 scratch[d_w_bits] = 0;                           ! Clear all the bits
2 scratch[d_l_ucb]  = .ucb;                        ! Save the ucb address
2
2 !
```

```
2 ! First, determine if an alternate path to the device exists.  If so,
2 ! next check that the UCB for the device is not already in the scratch
2 ! area.  If it is, return without saving this device.  If not, get the
2 ! secondary host information
2 !
2 IF .$BBLOCK[ucb[ucb$l_devchar2], dev$v_2p]       ! If device is dual-pathed
2 THEN
3     BEGIN
3     REGISTER
3         l,
3         scr : REF $BBLOCK;
3     scr = data[1];                               ! Start at the front of the data
3     WHILE .scr LSSA .scratch                     ! Look up to the current pointer
3     DO
4         BEGIN
4         IF .scr[d_l_ucb] EQLA .ucb               ! The UCB is already there,
4         THEN RETURN false;                       ! so we can simply exit now.
4         IF .scr[d_b_devclass] EQLU dc$_journal   ! If the device is a journal
4         THEN scr = .scr + d_k_length;            ! skip over the journal's device
4         scr = .scr + d_k_length;                 ! Skip to the next device
3         END;
3     !
3     ! First time we've seen this UCB, start stashing some info away.
3     !
3     scr = .ucb[ucb$l_2p_ddb];                    ! Get the ddb for the second path
3     scr = .scr[ddb$l_sb];                        ! Get the sb for the second host
3     !
3     ! Copy the node name and length
3     !
3     CH$MOVE (sb$s_nodename, scr[sb$t_nodename], scratch[d_t_host2_name]);
3     !
3     ! Copy the node type, a blank-padded string sitting in a long-word
3     !
3     scratch[d_l_host2_type] = .scr[sb$t_hwtype];
3     !
3     ! Tell if the host is available, i.e. if an SCS connection exists
3     !
4     scratch[d_v_host2_avail] = (IF .$BBLOCK[ucb[ucb$l_devchar2], dev$v_mscp]
4                                 THEN
5                                     BEGIN
5                                     scr = .ucb[ucb$l_2p_cddb];  ! Move the pointer to the CDDB for the devic
6                                     (NOT .scr[cddb$v_noconn])
5                                     END
3                                 ELSE 0);
2     END;                                         ! of code for dual-pathed devices
2
2 !
2 ! Save host info for the primary host.  We don't need to save the nodename, since that will be
2 ! part of the device name we return.
2 !
2 scr = .ddb[ddb$l_sb];                            ! Get the sb for the host
2 scratch[d_v_remote_device] = (.scr NEQ scs$ga_localsb);
2 CH$MOVE (sb$s_nodename, scr[sb$t_nodename], scratch[d_t_host_name]);
2 scratch[d_l_host_type] = .scr[sb$t_hwtype];      ! Copy the node type, a blank-padded string
2 scratch[d_v_host_avail] =  1;                     ! Assume that a connection exists (local node alway true)
2
2 !
```

```
  2 ! Check out some things only valid for MSCP devices
  2 !
  2 IF .$BBLOCK[ucb[ucb$l_devchar2], dev$v_mscp]
  2 THEN
  3     BEGIN
  3     scratch[d_v_shadow_master] = (.ucb[ucb$w_mscpunit] LSS 0);  ! Shadow masters have negative unit #s
  3     scr = .ucb[ucb$l_cddb];                                      ! Move the pointer to the CDDB for the devic
  3     scratch[d_v_host_avail] = (NOT .scr[cddb$v_noconn]);         ! Does a connection really exist?
  2     END;
  2
  2 !
  2 ! Now get the device name.
  2 !
  2 ioc$cvt_devnam(20,                                    ! Get device name, max this long
  2                scratch[d_t_device],                   ! put it here,
  3                (IF .$BBLOCK[ucb[ucb$l_devchar], dev$v_fod] ! If file-oriented
  3                 THEN 0                                ! then try for "$n$ddcu" format
  2                 ELSE -1),                             ! else select "node$ddcu" display format
  2                .ucb;                                  ! UCB is here
  2                scratch[d_b_devlen]);                  ! final length here
  2
  2 !
  2 ! Copy standard cells from the UCB to the scratch area
  2 !
P 2 copy_data (ucb, scratch, l_pid,                      ! Copy all the necessary
P 2                          l_devchar,                  ! information from the UCB.
P 2                          l_devchar2,
P 2                          b_devclass,
P 2                          b_devtype,
P 2                          w_unit,
P 2                          w_devbufsiz,
P 2                          l_devdepend,
P 2                          l_devdepnd2,
P 2                          w_refc,
P 2                          l_sts,
P 2                          w_devsts,
P 2                          l_opcnt,
  2                          w_errcnt);
  2
  2 !
  2 ! Copy ORB information to the scratch area
  2 !
  2 IF .orb[orb$v_prot_16]
  2 THEN scratch[d_w_vprot] = .orb[orb$w_prot]
  2 ELSE
  3     BEGIN
  3     (scratch[d_w_vprot])<0,4> = .(orb[orb$l_sys_prot])<0,4>;
  3     (scratch[d_w_vprot])<4,4> = .(orb[orb$l_own_prot])<0,4>;
  3     (scratch[d_w_vprot])<8,4> = .(orb[orb$l_grp_prot])<0,4>;
  3     (scratch[d_w_vprot])<12,4> = .(orb[orb$l_wor_prot])<0,4>;
  2     END;
  2 scratch[d_l_ownuic] = .orb[orb$l_owner];
  2 scratch[d_b_orb_flags] = .orb[orb$b_flags];
  2
  2 !
  2 ! Remember whether or not an ACL exists on the device
  2 !
  3 scratch[d_v_acl_present] = (IF .orb[orb$v_acl_queue]
  4                             THEN (.orb[orb$l_aclfl] NEQ orb[orb$l_aclfl])
  2                             ELSE 0);                ! Someday maybe (.orb[orb$l_acl_count] NEQ 0)
  2
  2 !
  2 ! Copy standard cells from the DDB to the scratch area
  2 !
  2 copy_data (ddb, scratch, l_allocls);
  2
  2 !
  2 ! If the device is owned, get the process name
  2 !
  2 IF .ucb[ucb$l_pid] NEQ 0
  2 THEN
  3     BEGIN
  3     LOCAL
  3         pix,
  3         pcb : REF $BBLOCK;
  3     pix = .(ucb[ucb$l_pid])<0,16>;
  3     IF .pix LEQU .sch$gl_maxpix
  3     THEN
  4         BEGIN
  4         pcb = .sch$gl_pcbvec[.pix];
  4         CH$MOVE(pcb$s_lname,
  4                 pcb[pcb$t_lname],
  4                 scratch[d_t_prcnam]);
  4         IF .pcb[pcb$l_pid] NEQ .ucb[ucb$l_pid]  ! Consistency check: do PIDs
  4         THEN scratch[d_t_prcnam] = 0;           ! Still match?  If no, don't
  3         END;                                    ! print the procname.
  2     END;
  2
  2 !
  2 ! For journals, get journal-specific information.
  2 !
  2 IF .ucb[ucb$b_devclass] EQLU dc$_journal
  2 THEN
  3     BEGIN
P 3     copy_data (ucb, scratch, l_jnl_mask,
P 3                              l_jnl_seqno,
P 3                              l_jnl_asid,
P 3                              l_jnl_quot,
P 3                              l_jnl_refc,
P 3                              l_jnl_trefc,
P 3                              w_jnl_id,
P 3                              w_devsts,
  3                              b_amod);
  3     IF NOT .ucb[ucb$v_jnl_slv]              ! If not a slave UCB
  3     AND .vcb NEQ 0                          ! and there's a VCB
  3     THEN
  4         BEGIN
  4         LOCAL
  4             first_jmt,
  4             jmt : REF $BBLOCK;
P 4         copy_data(vcb, scratch, l_jnl_char,
  4                                 w_jnl_cop);
  4         IF (first_jmt = jmt = .vcb[vcb$l_jnl_jmtfl]) NEQ 0
  4         THEN
```

```
5             BEGIN
5             LOCAL
5                 pointer : REF VECTOR[,BYTE],
5                 wcb : REF $BBLOCK,
5                 jnlucb : REF $BBLOCK,
5                 jnlddb : REF $BBLOCK;
5             CH$MOVE(.(jmt[jmt$t_grpnam])<0,8> + 1,
5                      jmt[jmt$t_grpnam],
5                      scratch[d_t_grpnam]);
5             scratch[d_l_fil_mxvbn] = .jmt[jmt$l_fil_mxvbn];
5             scratch[d_b_jnl_spl] = .jmt[jmt$v_spooled];
5             pointer = .scratch + d_k_length;
5             scratch[d_b_jnl_avl] = 0;
5             DO
6                 BEGIN
6                 IF .jmt[jmt$v_avl]
6                 THEN scratch[d_b_jnl_avl] = .scratch[d_b_jnl_avl] + 1;
6                 IF (wcb = .jmt[jmt$l_fil_wcb]) NEQ 0
6                 THEN
7                     BEGIN
7                     IF (jnlucb = .jmt[jmt$l_fil_ucb]) NEQ 0
7                     THEN IF (jnlddb = .jnlucb[ucb$l_ddb]) NEQ 0
7                     THEN
8                         BEGIN
8                         LOCAL
8                             count;
8                         ioc$cvt_devnam(20,
8                                        pointer[0],
8                                        -1,
8                                        .jnlucb;
8                                        count);
8                         pointer[0] = .count - 1;
8                         pointer = pointer[.count];
7                         END;
6                     END;
6                 jmt = .jmt[jmt$l_forjnllnk];
6                 END
5             UNTIL (.jmt EQL .first_jmt) OR (.jmt EQL 0);
4             END;
3         END;
2     END;
2
2 !
2 ! If this is a disk, get the maxblock value
2 !
2 IF .ucb[ucb$b_devclass] EQLU dc$_disk
2 THEN
2     scratch[d_l_maxblock] = .ucb[ucb$l_maxblock];
2
2 !
2 ! If this is a disk, tape, or journal, collect common information in the VCB.
2 !
2 IF .ucb[ucb$b_devclass] EQLU dc$_disk
2 OR .ucb[ucb$b_devclass] EQLU dc$_tape
2 OR .ucb[ucb$b_devclass] EQLU dc$_journal
2 THEN
3     BEGIN
```

```
;      3      IF .vcb EQL 0
;      3      THEN (scratch[d_b_cont] = 0; RETURN true);   ! If no VCB, go away.
;      3      scratch[d_b_cont] = 1;                       ! Say there's more
;   P  3      copy_data (vcb, scratch, b_status,           ! Copy VCB stuff
;   P  3                               w_rvn,
;   P  3                               w_mcount,
;      3                               w_trans);
;      3      IF .ucb[ucb$b_devclass] NEQ dc$_journal
;      3      THEN CH$MOVE(vcb$s_volname,                  ! Get the volume label
;      3                   vcb[vcb$t_volname],
;      3                   scratch[d_t_volnam])
;      3      ELSE CH$MOVE(ucb$s_jnl_nam,
;      3                   ucb[ucb$b_jnl_nam],
;      3                   scratch[d_t_volnam]);
;      3
;      3      scratch[d_b_aqbtype] = scratch[d_t_acpnam] = 0;      ! Assume no AQB, therefore no ACP name
;      3      IF (aqb = .vcb[vcb$l_aqb]) EQL 0                     ! If no AQB, then no more
;      3      THEN RETURN true;                                    ! Go away
;      3
;      3      scratch[d_b_aqbtype] = .aqb[aqb$b_acptype];          ! Stash the ACP type
;      3      IF .aqb[aqb$l_acppid] NEQ 0                          ! If the pid checks pass, get the ACP process name
;      3      THEN
;      4          BEGIN
;      4          LOCAL
;      4              pcb : REF $BBLOCK;
;      4          pcb = .sch$gl_pcbvec[.(aqb[aqb$l_acppid])<0,16>];
;      4          IF .pcb[pcb$l_pid] EQL .aqb[aqb$l_acppid]
;      4          THEN
;      4              CH$MOVE(pcb$s_lname,
;      4                      pcb[pcb$t_lname],
;      4                      scratch[d_t_acpnam]);
;      3          END;
;      3
;      3  !
;      3  ! If a magtape, get magtape-specific data from the Magtape Volume List (MVL).
;      3  ! This is rather involved, since there is no direct link between the MVL and
;      3  ! the UCB in question.  Instead, the list of UCB's in the Relative Volume
;      3  ! Table are scanned in index order, until this UCB is found.  The mounted tape
;      3  ! in the MVL with the same index is then found.
;      3  !
;      3      IF .aqb[aqb$b_acptype] EQL aqb$k_mta
;      3      THEN
;      4          BEGIN
;      4          BIND
;      4              rvt = vcb[vcb$l_rvt] : REF $BBLOCK,
;      4              ucblst = rvt[rvt$l_ucblst] : VECTOR;
;      4          LOCAL
;      4              index;
;      4          index = -1;
;      4          INCR i FROM 0 TO .rvt[rvt$b_nvols] -1 DO
;      5          (IF .ucblst[.i] EQL .ucb
;      4           THEN (index = .i; EXITLOOP));
;      4          IF .index EQL -1
;      4          THEN
;      5              BEGIN
;      5              scratch[d_t_volnam] = 0;
;      5              scratch[d_w_rvn] = 0;
```

```
  5            END
  4        ELSE
  5            BEGIN
  5            LOCAL
  5                limit,
  5                mvl : REF $BBLOCK;
  5            mvl = .vcb[vcb$l_mvl] + mvl$k_fixlen;
  5            limit = .mvl[mvl$b_nvols] - 1;
  5            INCR mvli FROM 0 TO .limit DO                     ! Find an entry with
  6                BEGIN                                         ! the same RVN and
  6                IF .mvl[mvl$b_rvn] EQL .index                 ! that is mounted
  6                AND .mvl[mvl$b_status]                        ! (low bit set -> mounted)
  6                THEN
  7                    BEGIN
  7                    scratch[d_w_rvn] = .mvli + 1;             ! Because RVN's start at 1
  7                    CH$MOVE(mvl$s_vollbl,                     ! Get the volume label
  7                            mvl[mvl$t_vollbl],
  7                            scratch[d_t_volnam]);
  7                    EXITLOOP                                  ! Skip the rest of the MVL
  7                    END
  6                ELSE mvl = .mvl + mvl$k_length;               ! Otherwise, go to next
  5                END;                                          ! MVL entry
  4            END;
  4        scratch[d_w_recordsz] = .vcb[vcb$w_recordsz];         ! Get record size
  4        RETURN true;                                          ! Go away.
  3        END;
  3  !
  3  ! If this is a disk, collect disk-specific information
  3  !
  3    IF .aqb[aqb$b_acptype] EQL aqb$k_f11v1
  3    OR .aqb[aqb$b_acptype] EQL aqb$k_f11v2
  3    THEN
  4        BEGIN
P 4        copy_data (vcb, scratch, w_cluster,
P 4                                 w_extend,
P 4                                 l_free,
P 4                                 l_maxfiles,
P 4                                 b_window,
  4                                 b_lru_lim);
  3        END;
  3
  3  !
  3  ! For ODS-2 disks, there is more information to collect, namely the retention
  3  ! periods and caching parameters.
  3  !
  3    IF .aqb[aqb$b_acptype] EQL aqb$k_f11v2
  3    THEN
  4        BEGIN
  4        LOCAL vca : REF $BBLOCK;
  4        !
  4        ! For ODS-2 disks, get the correct free blocks from the value block associated with
  4        ! the volume lock.  We call an internal routine in GETDVI which will use $GETLKI to
  4        ! grab the value from the XQP's lock value block.  This routine expects to be called
  4        ! at IPL = IPL$_ASTDEL.
  4        !
  4        exe$dvi_freeblocks (.vcb[vcb$l_vollkid], scratch[d_l_free]);
  4        copy_data (vcb, scratch, b_status2);
```

```
;  843      2240  4            CH$MOVE(vcb$s_retainmin + vcb$s_retainmax,
;  844      2241  4                    vcb[vcb$q_retainmin],
;  845      2242  4                    scratch[d_q_retainmin]);
;  846      2243  4            scratch[d_w_fidsize] = scratch[d_w_quosize]
;  847      2244  4                                 = scratch[d_w_extsize]
;  848      2245  4                                 = 0;
;  849      2246  4            IF (vca = .vcb[vcb$l_cache]) NEQ 0                  ! If fid/ext cache
;  850      2247  4            THEN                                                ! present, get those
;  851      2248  5                BEGIN
;  852      2249  5                LOCAL cache : REF $BBLOCK;
;  853      2250  5                IF (cache = .vca[vca$l_fidcache]) NEQ 0
;  854      2251  5                THEN scratch[d_w_fidsize] = .cache[vca$w_fidsize];
;  855      2252  5                IF (cache = .vca[vca$l_extcache]) NEQ 0
;  856      2253  5                THEN
;  857      2254  6                    BEGIN
;  858      2255  6                    scratch[d_w_extsize] = .cache[vca$w_extsize];
;  859      2256  6                    scratch[d_w_extlimit] = .cache[vca$w_extlimit];
;  860      2257  6                    scratch[d_l_exttotal] = .cache[vca$l_exttotal];
;  861      2258  5                    END;
;  862      2259  4                END;
;  863      2260  4            IF (vca = .vcb[vcb$l_quocache]) NEQ 0               ! If quota cache,
;  864      2261  4            THEN scratch[d_w_quosize] = .vca[vca$w_quosize];! get quota size.
;  865      2262  4            $ASSUME (d_s_acpnam, GEQ, f11bc$s_cachename);     ! Make sure it is large enough
;  866      2263  5            IF  ((vca = .aqb[aqb$l_bufcache]) NEQ 0)          ! If buffer cache exists get the cache name
;  867      2264  4              AND
;  868      2265  5                (.aqb[aqb$l_acppid] EQL 0)                      ! if the acp didn't have a name
;  869      2266  4            THEN
;  870      2267  5                BEGIN
;  871      2268  5                scratch[d_v_cachename] = 1;                     ! Remember that it is cache name and not ACP name
;  872      2269  5                CH$MOVE (f11bc$s_cachename,
;  873      2270  5                         vca[f11bc$t_cachename],
;  874      2271  5                         scratch[d_t_acpnam]);
;  875      2272  5                scratch[d_w_bfrcnt] = .vca[f11bc$w_bfrcnt]; ! Number of buffer cache blocks
;  876      2273  4                END;
;  877      2274  3            END;
;  878      2275  2        END;
;  879      2276  2
;  880      2277  2    !
;  881      2278  2    ! In the event that that the device is spooled, the VCB field actually
;  882      2279  2    ! points to a block containing the name of the queue to which this device
;  883      2280  2    ! is spooled, and UCB$L_AMB contains the address of the UCB of the
;  884      2281  2    ! intermediate device.
;  885      2282  2    !
;  886      2283  2    IF .$BBLOCK[ucb[ucb$l_devchar], dev$v_spl]
;  887      2284  2    THEN
;  888      2285  3        BEGIN
;  889      2286  3        BIND
;  890      2287  3            int_ucb = ucb[ucb$l_amb] : REF $BBLOCK,
;  891      2288  3            int_ddb = int_ucb[ucb$l_ddb] : REF $BBLOCK;
;  892      2289  3        ioc$cvt_devnam(20,                                      ! Get device name, max this long
;  893      2290  3                       scratch[d_t_intdev],                     ! put it here,
;  894      2291  3                       -1,                                      ! in standard display format
;  895      2292  3                       .int_ucb;                                ! UCB is here
;  896      2293  3                       scratch[d_l_intlen]);                    ! final length here
;  897      2294  3        IF .vcb NEQ 0
;  898      2295  3        THEN CH$MOVE(.vcb[vcb$b_qnamecnt] + 1,
;  899      2296  3                     vcb[vcb$b_qnamecnt],
;  900          2297  3                    scratch[d_t_qname])
;  901          2298  3           ELSE scratch[d_t_qname] = 0;
;  902          2299  3           RETURN true;
;  903          2300  2           END;
;  904          2301  2
;  905          2302  2   RETURN true;
;  906          2303  1   END;



                                          0FFC 00000          .ENTRY  UTL_GET_DATA, Save R2,R3,R4,R5,R6,R7,R8,R9,-;  1841
                                                                      R10,R11                                       ;
                        5E            20  C2 00002          SUBL2   #32, SP                                       ;
                        58       04   AC  D0 00005          MOVL    IN_UCB, UCB                                   ;  1883
            05     3C   A8            03  E1 00009          BBC     #3, 60(UCB), 1$                               ;  1884
                        58     00A8   C8  D0 0000E          MOVL    168(UCB), UCB                                 ;  1885
                        59       1C   A8  D0 00013 1$:      MOVL    28(UCB), ORB                                  ;  1886
                        5B       28   A8  D0 00017          MOVL    40(UCB), DDB                                  ;  1887
                        5A       34   A8  D0 0001B          MOVL    52(UCB), VCB                                  ;  1888
                        57       10   AC  D0 0001F          MOVL    SCRATCH, R7                                   ;  1894
                   14   AE       04   A7  9E 00023          MOVAB   4(R7), 20(SP)                                 ;
                                 14   BE  B4 00028          CLRW    @20(SP)                                       ;
                        67            58  D0 0002B          MOVL    UCB, (R7)                                     ;  1895
            56     3C   A8            04  E1 0002E          BBC     #4, 60(UCB), 8$                               ;  1903
            56     14   AC            04  C1 00033          ADDL3   #4, DATA, SCR                                 ;  1909
                        57            56  D1 00038 2$:      CMPL    SCR, R7                                       ;  1910
                                      1B  1E 0003B          BGEQU   5$                                            ;
                        58            66  D1 0003D          CMPL    (SCR), UCB                                    ;  1913
                                      03  12 00040          BNEQ    3$                                            ;
                                    0422  31 00042          BRW     47$                                           ;
                   A1   8F       78   A6  91 00045 3$:      CMPB    120(SCR), #161                                ;  1915
                                      05  12 0004A          BNEQ    4$                                            ;
                        56     0107   C6  9E 0004C          MOVAB   263(R6), SCR                                  ;  1916
                        56     0107   C6  9E 00051 4$:      MOVAB   263(R6), SCR                                  ;  1917
                                      E0  11 00056          BRB     2$                                            ;  1910
                        56     00A0   C8  D0 00058 5$:      MOVL    160(UCB), SCR                                 ;  1922
                        56       34   A6  D0 0005D          MOVL    52(SCR), SCR                                  ;  1923
       30    A7    44   A6            10  28 00061          MOVC3   #16, 68(SCR), 48(R7)                          ;  1927
                   40   A7       34   A6  D0 00067          MOVL    52(SCR), 64(R7)                               ;  1931
             10    3C   A8            05  E1 0006C          BBC     #5, 60(UCB), 6$                               ;  1935
                        56     00C0   C8  D0 00071          MOVL    192(UCB), SCR                                 ;  1938
  50   12    A6         01            07  EF 00076          EXTZV   #7, #1, 18(SCR), R0                           ;  1939
                        50            50  D2 0007C          MCOML   R0, R0                                        ;
                                      02  11 0007F          BRB     7$                                            ;
                                      50  D4 00081 6$:      CLRL    R0                                            ;  1935
  14   BE          01   02            50  F0 00083 7$:      INSV    R0, #2, #1, @20(SP)                           ;
                        56       34   AB  D0 00089 8$:      MOVL    52(DDB), SCR                                  ;  1948
                        51  00000000G 00  9E 0008D          MOVAB   SCS$GA_LOCALSB, R1                            ;  1949
                                      50  D4 00094          CLRL    R0                                            ;
                        51            56  D1 00096          CMPL    SCR, R1                                       ;
                                      02  13 00099          BEQL    9$                                            ;
                                      50  D6 0009B          INCL    R0                                            ;
  14   BE          01   03            50  F0 0009D 9$:      INSV    R0, #3, #1, @20(SP)                           ;
             1C    A7   44   A6       10  28 000A3          MOVC3   #16, 68(SCR), 28(R7)                          ;  1950
                   2C   A7       34   A6  D0 000A9          MOVL    52(SCR), 44(R7)                               ;  1951
```

```
                            14    BE                   02   88 000AE          BISB2    #2, @20(SP)                       ; 1952
                  18        3C    A8                   05   E1 000B2          BBC      #5, 60(UCB), 10$                  ; 1957
                            14    BE                   10   8A 000B7          BICB2    #16, @20(SP)                      ; 1960
                                  56        00BC       C8   D0 000BB          MOVL     188(UCB), SCR                     ; 1961
      50          12        A6                         01   07   EF 000C0     EXTZV    #7, #1, 18(SCR), R0               ; 1962
                                  50                   50   D2 000C6          MCOML    R0, R0                            ;
      14    BE              01    01                   50   F0 000C9          INSV     R0, #1, #1, @20(SP)               ;
                  04        39    A8                   06   E1 000CF  10$:    BBC      #6, 57(UCB), 11$                  ; 1970
                                                       54   D4 000D4          CLRL     R4                                ;
                                                       03   11 000D6          BRB      12$                               ;
                                  54                   01   CE 000D8  11$:    MNEGL    #1, R4                            ; 1972
                                  51          08       A7   9E 000DB  12$:    MOVAB    8(R7), R1                         ; 1969
                                  55                   58   D0 000DF          MOVL     UCB, R5                           ; 1974
                                  50                   14   D0 000E2          MOVL     #20, R0                           ;
                                        00000000G      00   16 000E5          JSB      IOC$CVT_DEVNAM                    ;
                            06    A7                   51   90 000EB          MOVB     R1, 6(R7)                         ;
                            5C    A7          2C       A8   D0 000EF          MOVL     44(UCB), 92(R7)                   ; 1992
                            70    A7          38       A8   7D 000F4          MOVQ     56(UCB), 112(R7)                  ;
                                  56          40       A8   9A 000F9          MOVZBL   64(UCB), R6                       ;
                            78    A7                   56   90 000FD          MOVB     R6, 120(R7)                       ;
                            79    A7          41       A8   90 00101          MOVB     65(UCB), 121(R7)                  ;
                            52    A7          54       A8   B0 00106          MOVW     84(UCB), 82(R7)                   ;
                            7A    A7          42       A8   B0 0010B          MOVW     66(UCB), 122(R7)                  ;
                            7C    A7          44       A8   7D 00110          MOVQ     68(UCB), 124(R7)                  ;
                          0086    C7          5C       A8   B0 00115          MOVW     92(UCB), 134(R7)                  ;
                          0088    C7          64       A8   D0 0011B          MOVL     100(UCB), 136(R7)                 ;
                          0090    C7          68       A8   B0 00121          MOVW     104(UCB), 144(R7)                 ;
                          008C    C7          70       A8   D0 00127          MOVL     112(UCB), 140(R7)                 ;
                          0092    C7        0082       C8   B0 0012D          MOVW     130(UCB), 146(R7)                 ;
                                  50        0084       C7   9E 00134          MOVAB    132(R7), R0                       ; 1998
                                  06          0B       A9   E9 00139          BLBC     11(ORB), 13$                      ; 1997
                                  60          18       A9   B0 0013D          MOVW     24(ORB), (R0)                     ; 1998
                                                       19   11 00141          BRB      14$                               ;
            60              04    00          18       A9   F0 00143  13$:    INSV     24(ORB), #0, #4, (R0)             ; 2001
            60              04    04          1C       A9   F0 00149          INSV     28(ORB), #4, #4, (R0)             ; 2002
      01    A0              04    00          20       A9   F0 0014F          INSV     32(ORB), #0, #4, 1(R0)            ; 2003
            60              04    0C          24       A9   F0 00156          INSV     36(ORB), #12, #4, (R0)            ; 2004
                            58    A7                   69   DC 0015C  14$:    MOVL     (ORB), 88(R7)                     ; 2006
                          0098    C7          0B       A9   90 00160          MOVB     11(ORB), 152(R7)                  ; 2007
                  10        0B    A9                   01   E1 00166          BBC      #1, 11(ORB), 15$                  ; 2012
                                  51          28       A9   9E 0016B          MOVAB    40(ORB), R1                       ; 2013
                                                       50   D4 0016F          CLRL     R0                                ;
                                  51          28       A9   D1 00171          CMPL     40(ORB), R1                       ;
                                                       06   13 00175          BEQL     16$                               ;
                                                       50   D6 00177          INCL     R0                                ;
                                                       02   11 00179          BRB      16$                               ;
                                                       50   D4 0017B  15$:    CLRL     R0                                ; 2012
      14    BE              01    09                   50   F0 0017D  16$:    INSV     R0, #9, #1, @20(SP)               ;
                            54    A7          3C       AB   D0 00183          MOVL     60(DDB), 84(R7)                   ; 2019
                                              2C       A8   D5 00188          TSTL     44(UCB)                           ; 2024
                                                       28   13 0018B          BEQL     17$                               ;
                                  50          2C       A8   3C 0018D          MOVZWL   44(UCB), PIX                      ; 2030
                       00000000G  00                   50   D1 00191          CMPL     PIX, SCH$GL_MAXPIX                ; 2031
                                                       1B   1A 00198          BGTRU    17$                               ;
                                  51 00000000G         00   D0 0019A          MOVL     SCH$GL_PCBVEC, R1                 ; 2034
                                  59            6140        D0 001A1          MOVL     (R1)[PIX], PCB                    ;
                  60        A7    70          A9       10   28 001A5          MOVC3    #16, 112(PCB), 96(R7)             ; 2037
```

```
                      2C   A8         60   A9   D1 001AB           CMPL    96(PCB), 44(UCB)                        : 2038
                                           03   13 001B0           BEQL    17$                                     :
                                      60   A7   94 001B2           CLRB    96(R7)                                  : 2039
                                      1C   AE   D4 001B5 17$:      CLRL    28(SP)                                  : 2046
                      A1   8F              56   91 001B8           CMPB    R6, #161                                :
                                           03   13 001BC           BEQL    18$                                     :
                                         00B7   31 001BE           BRW     22$                                     :
                                      1C   AE   D6 001C1 18$:      INCL    28(SP)                                  :
                    00E4   C7       00D4   C8   D0 001C4           MOVL    212(UCB), 228(R7)                       : 2057
                    00E8   C7         44   A8   D0 001CB           MOVL    68(UCB), 232(R7)                        :
                    00EC   C7       00D8   C8   D0 001D1           MOVL    216(UCB), 236(R7)                       :
                    00F0   C7       00CC   C8   D0 001D8           MOVL    204(UCB), 240(R7)                       :
                    00F4   C7       00DC   C8   7D 001DF           MOVQ    220(UCB), 244(R7)                       :
                    00FC   C7       00D0   C8   B0 001E6           MOVW    208(UCB), 252(R7)                       :
                    0090   C7         68   A8   B0 001ED           MOVW    104(UCB), 144(R7)                       :
                    0100   C7         5F   A8   90 001F3           MOVB    95(UCB), 256(R7)                        :
                                      68   A8   95 001F9           TSTB    104(UCB)                                : 2058
                                           7A   19 001FC           BLSS    22$                                     :
                                           5A   D5 001FE           TSTL    VCB                                     : 2059
                                           76   13 00200           BEQL    22$                                     :
                    00E0   C7         24   AA   D0 00202           MOVL    36(VCB), 224(R7)                        : 2066
                    0101   C7         45   AA   B0 00208           MOVW    69(VCB), 257(R7)                        :
                           59         3C   AA   D0 0020E           MOVL    60(VCB), JMT                            : 2067
                           5B              59   D0 00212           MOVL    JMT, FIRST_JMT                          :
                                           61   13 00215           BEQL    22$                                     :
                           50         7A   A9   9A 00217           MOVZBL  122(JMT), R0                            : 2075
                                           50   D6 0021B           INCL    R0                                      :
            00CE    C7        7A   A9      50   28 0021D           MOVC3   R0, 122(JMT), 206(R7)                   : 2077
                    00DC   C7         58   A9   D0 00224           MOVL    88(JMT), 220(R7)                        : 2078
   50        2D    A9      01              03   EF 0022A           EXTZV   #3, #1, 45(JMT), R0                     : 2079
                           52       0107   C7   9E 00230           MOVAB   263(R7), POINTER                        : 2080
                    0103   C7              50   9B 00235           MOVZBW  R0, 259(R7)                             : 2079
                     04    2E   A9         01   E1 0023A 19$:      BBC     #1, 46(JMT), 20$                        : 2084
                                    0104   C7   96 0023F           INCB    260(R7)                                 : 2085
                      18   AE         50   A9   D0 00243 20$:      MOVL    80(JMT), WCB                            : 2086
                                           22   13 00248           BEQL    21$                                     :
                           55         54   A9   D0 0024A           MOVL    84(JMT), JNLUCB                         : 2089
                                           1C   13 0024E           BEQL    21$                                     :
                           53         28   A5   D0 00250           MOVL    40(JNLUCB), JNLDDB                      : 2090
                                           16   13 00254           BEQL    21$                                     :
                           54              01   CE 00256           MNEGL   #1, R4                                  : 2096
                           51              52   D0 00259           MOVL    POINTER, R1                             :
                           50              14   D0 0025C           MOVL    #20, R0                                 :
                                00000000G  00   16 0025F           JSB     IOC$CVT_DEVNAM                          :
                      62   51              01   83 00265           SUBB3   #1, COUNT, (POINTER)                    : 2100
                           52              51   C0 00269           ADDL2   COUNT, POINTER                          : 2101
                           59              69   D0 0026C 21$:      MOVL    (JMT), JMT                              : 2104
                           5B              59   D1 0026F           CMPL    JMT, FIRST_JMT                          : 2106
                                           04   13 00272           BEQL    22$                                     :
                                           59   D5 00274           TSTL    JMT                                     :
                                           C2   12 00276           BNEQ    19$                                     :
                                           50   D4 00278 22$:      CLRL    R0                                      : 2114
                           01              56   91 0027A           CMPB    R6, #1                                  :
                                           09   12 0027D           BNEQ    23$                                     :
                                           50   D6 0027F           INCL    R0                                      :
                    0094   C7       00B0   C8   D0 00281           MOVL    176(UCB), 148(R7)                       : 2116
                           0C              50   E8 00288 23$:      BLBS    R0, 24$                                 : 2121
```

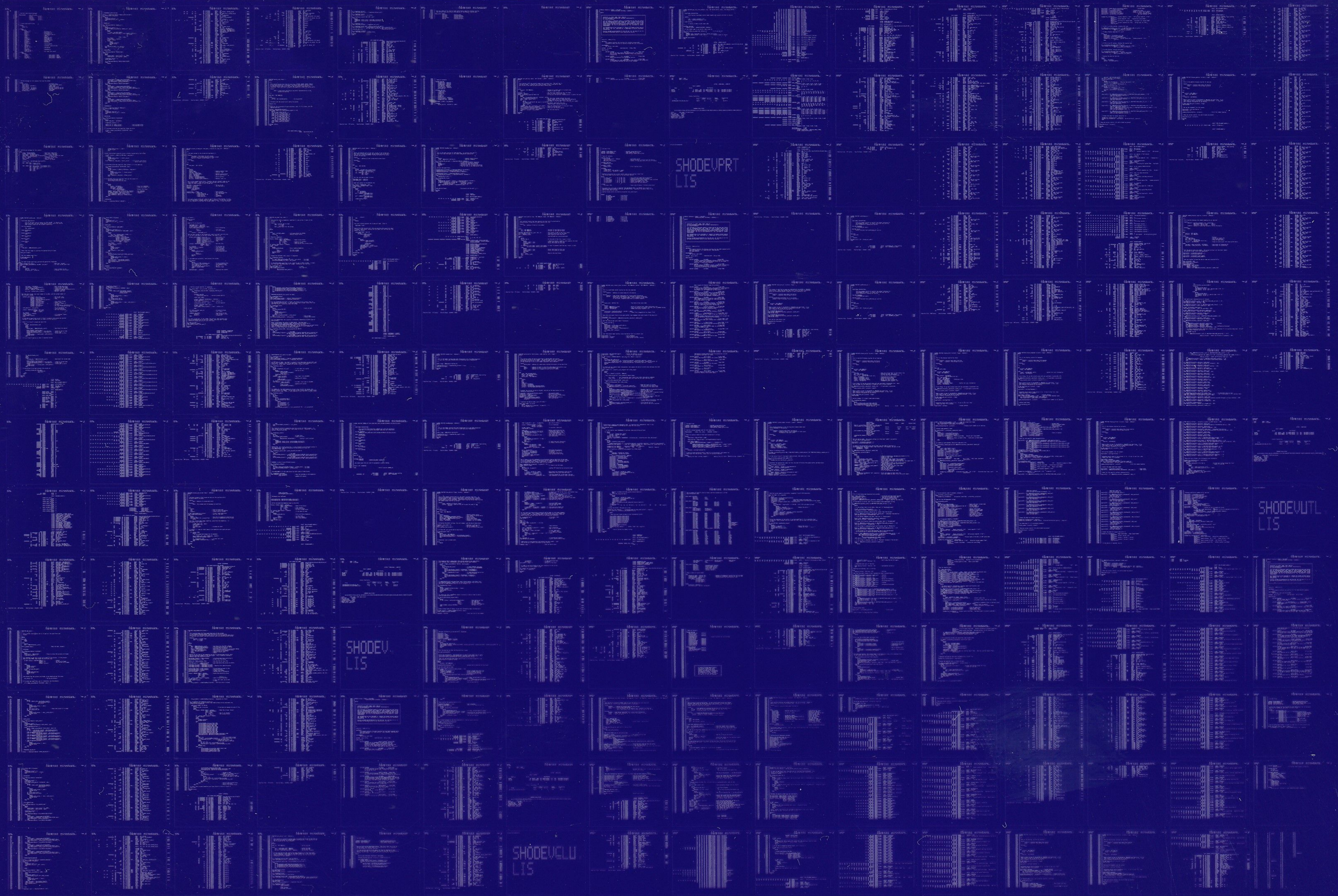
```
                        02              56  91 0028B        CMPB    R6, #2                              ; 2122
                                        07  13 0028E        BEQL    24$
                        03         1C   AE  E8 00290        BLBS    28(SP), 24$                         ; 2123
                                      0196  31 00294        BRW     44$
                                        5A  D5 00297 24$:   TSTL    VCB                                 ; 2126
                                        07  12 00299        BNEQ    26$
                                 0099   C7  94 0029B        CLRB    153(R7)                             ; 2127
                                      01C1  31 0029F 25$:   BRW     46$
                  0099  C7              01  90 002A2 26$:   MOVB    #1, 153(R7)                         ; 2128
                  009A  C7         0B   AA  90 002A7        MOVB    11(VCB), 154(R7)                    ; 2132
                    0C  AE       00B6   C7  9E 002AD        MOVAB   182(R7), 12(SP)
                    0C  BE         0E   AA  B0 002B3        MOVW    14(VCB), @12(SP)
                  00CC  C7         4C   AA  B0 002B8        MOVW    76(VCB), 204(R7)
                  009B  C7         0C   AA  B0 002BE        MOVW    12(VCB), 155(R7)
                    6E       00B8       C7  9E 002C4        MOVAB   184(R7), (SP)                       ; 2136
                    A1  8F              56  91 002C9        CMPB    R6, #161                            ; 2133
                                        08  13 002CD        BEQL    27$
    00      BE      14  AA              0C  28 002CF        MOVC3   #12, 20(VCB), @0(SP)                ; 2136
                                        07  11 002D5        BRB     28$
    00      BE    00B9  C8              12  28 002D7 27$:   MOVC3   #18, 185(UCB), @0(SP)               ; 2139
                    18  AE       009E   C7  9E 002DE 28$:   MOVAB   158(R7), 24(SP)                     ; 2141
                                   18   BE  94 002E4        CLRB    @24(SP)
                                 009D   C7  94 002E7        CLRB    157(R7)
                        59         10   AA  D0 002EB        MOVL    16(VCB), AQB                        ; 2142
                                        AE  13 002EF        BEQL    25$
                    08  AE         15   A9  9A 002F1        MOVZBL  21(AQB), 8(SP)                      ; 2145
                  009D  C7         08   AE  90 002F6        MOVB    8(SP), 157(R7)
                    10  AE         0C   A9  D0 002FC        MOVL    12(AQB), 16(SP)                     ; 2146
                                        1C  13 00301        BEQL    29$
                        51 00000000G    00  D0 00303        MOVL    SCH$GL_PCBVEC, R1                   ; 2151
                        50         0C   A9  3C 0030A        MOVZWL  12(AQB), R0
                        50            6140  D0 0030E        MOVL    (R1)[R0], PCB
                    10  AE         60   A0  D1 00312        CMPL    96(PCB), 16(SP)                     ; 2152
                                        06  12 00317        BNEQ    29$
    18      BE      70  A0              10  28 00319        MOVC3   #16, 112(PCB), @24(SP)              ; 2156
                        03         08   AE  91 0031F 29$:   CMPB    8(SP), #3                           ; 2166
                                        73  12 00323        BNEQ    38$
                        50         20   AA  D0 00325        MOVL    32(VCB), R0                         ; 2171
                        52         44   A0  9E 00329        MOVAB   68(R0), R2
                    04  AE              01  CE 0032D        MNEGL   #1, INDEX                           ; 2174
                        51         0B   A0  9A 00331        MOVZBL  11(R0), R1                          ; 2175
                        50              01  CE 00335        MNEGL   #1, I
                                        0C  11 00338        BRB     31$
                        58            6240  D1 0033A 30$:   CMPL    (R2)[I], UCB                        ; 2176
                                        06  12 0033E        BNEQ    31$
                    04  AE              50  D0 00340        MOVL    I, INDEX                            ; 2177
                                        04  11 00344        BRB     32$
            F0          50              51  F2 00346 31$:   AOBLSS  R1, I, 30$                          ; 2176
              FFFFFFFF  8F         04   AE  D1 0034A 32$:   CMPL    INDEX, #-1                          ; 2178
                                        08  12 00352        BNEQ    33$
                                   00   BE  94 00354        CLRB    @0(SP)                              ; 2181
                                   0C   BE  B4 00357        CLRW    @12(SP)                             ; 2182
                                        33  11 0035A        BRB     37$                                 ; 2178
            56      34  AA              24  C1 0035C 33$:   ADDL3   #36, 52(VCB), MVL                   ; 2189
                    1C  AE         0B   A6  9A 00361        MOVZBL  11(MVL), LIMIT                      ; 2190
                                   1C   AE  D7 00366        DECL    LIMIT
                        5B              01  CE 00369        MNEGL   #1, MVLI                            ; 2191
```

```
                                                    1C  11 0036C        BRB     36$
      04   AE       06   A6           08            00  ED 0036E 34$:   CMPZV   #0, #8, 6(MVL), INDEX                 ; 2193
                                                    10  12 00375        BNEQ    35$
                                      0C         07 A6  E9 00377        BLBC    7(MVL), 35$                           ; 2194
                    0C   BE           5B            01  A1 0037B        ADDW3   #1, MVLI, @12(SP)                     ; 2197
                    00   BE           66            06  28 00380        MOVC3   #6, (MVL), @0(SP)                     ; 2200
                                                    08  11 00385        BRB     37$                                   ; 2196
                                      56            08  C0 00387 35$:   ADDL2   #8, MVL                               ; 2203
                         DF           5B         1C AE  F3 0038A 36$:   AOBLEQ  LIMIT, MVLI, 34$                      ; 2191
                                00CE  C7         50 AA  B0 0038F 37$:   MOVW    80(VCB), 206(R7)                      ; 2206
                                                  00CB  31 00395        BRW     46$                                   ; 2207
                                      01         08 AE  91 00398 38$:   CMPB    8(SP), #1                             ; 2212
                                                    06  13 0039C        BEQL    39$
                                      02         08 AE  91 0039E        CMPB    8(SP), #2                             ; 2213
                                                    12  12 003A2        BNEQ    40$
                                00CE  C7         3C AA  7D 003A4 39$:   MOVQ    60(VCB), 206(R7)                      ; 2221
                                00D6  C7         44 AA  D0 003AA        MOVL    68(VCB), 214(R7)
                                00DA  C7         48 AA  B0 003B0        MOVW    72(VCB), 218(R7)
                                      02         08 AE  91 003B6 40$:   CMPB    8(SP), #2                             ; 2228
                                                    71  12 003BA        BNEQ    44$
                                               00D2 C7  9F 003BC        PUSHAB  210(R7)                               ; 2238
                                                 7C AA  DD 003C0        PUSHL   124(VCB)
                           00000000G  00            02  FB 003C3        CALLS   #2, EXE$DVI_FREEBLOCKS
                                00DC  C7         53 AA  90 003CA        MOVB    83(VCB), 220(R7)                      ; 2239
                  00DD   C7       6C  AA            10  28 003D0        MOVC3   #16, 108(VCB), 221(R7)                ; 2242
                                               00F7 C7  B4 003D7        CLRW    247(R7)                               ; 2245
                                               00ED C7  D4 003DB        CLRL    237(R7)                               ; 2244
                                      5B         58 AA  D0 003DF        MOVL    88(VCB), VCA                          ; 2246
                                                    21  13 003E3        BEQL    42$
                                      50            6B  D0 003E5        MOVL    (VCA), CACHE                          ; 2250
                                                    05  13 003E8        BEQL    41$
                                00ED  C7            60  B0 003EA        MOVW    (CACHE), 237(R7)                      ; 2251
                                      50         04 AB  D0 003EF 41$:   MOVL    4(VCA), CACHE                         ; 2252
                                                    11  13 003F3        BEQL    42$
                                00EF  C7            60  B0 003F5        MOVW    (CACHE), 239(R7)                      ; 2255
                                00F1  C7         08 A0  B0 003FA        MOVW    8(CACHE), 241(R7)                     ; 2256
                                00F3  C7         04 A0  D0 00400        MOVL    4(CACHE), 243(R7)                     ; 2257
                                      5B         5C AA  D0 00406 42$:   MOVL    92(VCB), VCA                          ; 2260
                                                    05  13 0040A        BEQL    43$
                                00F7  C7            6B  B0 0040C        MOVW    (VCA), 247(R7)                        ; 2261
                                      5B         18 A9  D0 00411 43$:   MOVL    24(AQB), VCA                          ; 2263
                                                    16  13 00415        BEQL    44$
                                                 10 AE  D5 00417        TSTL    16(SP)                                ; 2265
                                                    11  12 0041A        BNEQ    44$
                                  14  BE            20  88 0041C        BISB2   #32, @20(SP)                          ; 2268
                    18   BE     00AC  CB            18  28 00420        MOVC3   #24, 172(VCA), @24(SP)                ; 2271
                                00F9  C7         16 AB  B0 00427        MOVW    22(VCA), 249(R7)                      ; 2272
                         31       38  A8            06  E1 0042D 44$:   BBC     #6, 56(UCB), 46$                      ; 2283
                                      51       009A C7  9E 00432        MOVAB   154(R7), R1                           ; 2290
                                      55         60 A8  D0 00437        MOVL    96(UCB), R5                           ; 2293
                                      54            01  CE 0043B        MNEGL   #1, R4
                                      50            14  D0 0043E        MOVL    #20, R0
                                          00000000G 00  16 00441        JSB     IOC$CVT_DEVNAM
                                00AE  C7            51  D0 00447        MOVL    R1, 174(R7)
                                                    5A  D5 0044C        TSTL    VCB                                   ; 2294
                                                    0F  13 0044E        BEQL    45$
                                      50         0B AA  9A 00450        MOVZBL  11(VCB), R0                           ; 2295
```

```
                                            50 D6 00454        INCL    R0                                ;
            00B2   C7       0B   AA         50 28 00456        MOVC3   R0, 11(VCB), 178(R7)              ; 2297
                                            04 11 0045D        BRB     46$                               ;
                                 00B2       C7 94 0045F 45$:   CLRB    178(R7)                           ; 2298
                          50                01 D0 00463 46$:   MOVL    #1, R0                            ; 2302
                                               04 00466        RET                                       ;
                                            50 D4 00467 47$:   CLRL    R0                                ; 2303
                                               04 00469        RET                                       ;

; Routine Size:  1130 bytes,    Routine Base:  $CODE$ + 01DB
;  908          2304  1 END
;  909          2305  0 ELUDOM




;                                         PSECT SUMMARY
;
;       Name                        Bytes                          Attributes
;
;  $GLOBAL$                            16  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                            1605  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       136         0        1000           00:01.9




;                                         COMMAND QUALIFIERS

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

; Size:          1605 code + 16 data bytes
; Run Time:         00:55.9
; Elapsed Time:     03:00.4
; Lines/CPU Min:      2476
; Lexemes/CPU-Min: 39481
; Memory Used:  564 pages
; Compilation Complete
```

0055 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



0056 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SHOMSGUTL
LIS

SHONET
LIS

SHOWAUDIT
LIS

SHOWIO
LIS

SHOWLOG
LIS

SHOWERROR
LIS

SHOWFILES
LIS

SHOMEMORY
LIS