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

FILEID**SETDEVICE

```
 SSSSSSSS   EEEEEEEEEE  TTTTTTTTTT  DDDDDDDD    EEEEEEEEEE  VV      VV    IIIIII      CCCCCCCC  EEEEEEEEEE
 SSSSSSSS   EEEEEEEEEE  TTTTTTTTTT  DDDDDDDD    EEEEEEEEEE  VV      VV    IIIIII      CCCCCCCC  EEEEEEEEEE
SS          EE              TT      DD      DD  EE          VV      VV      II      CC          EE
SS          EE              TT      DD      DD  EE          VV      VV      II      CC          EE
SS          EE              TT      DD      DD  EE          VV      VV      II      CC          EE
SS          EE              TT      DD      DD  EE          VV      VV      II      CC          EE
 SSSSSS     EEEEEEEE        TT      DD      DD  EEEEEEEE    VV      VV      II      CC          EEEEEEEE
 SSSSSS     EEEEEEEE        TT      DD      DD  EEEEEEEE    VV      VV      II      CC          EEEEEEEE
       SS   EE              TT      DD      DD  EE          VV      VV      II      CC          EE
       SS   EE              TT      DD      DD  EE          VV      VV      II      CC          EE
       SS   EE              TT      DD      DD  EE            VV  VV        II      CC          EE          ....
       SS   EE              TT      DD      DD  EE            VV  VV        II      CC          EE          ....
SSSSSSSS    EEEEEEEEEE      TT      DDDDDDDD    EEEEEEEEEE      VV        IIIIII      CCCCCCCC  EEEEEEEEEE  ....
SSSSSSSS    EEEEEEEEEE      TT      DDDDDDDD    EEEEEEEEEE      VV        IIIIII      CCCCCCCC  EEEEEEEEEE  ....

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 setdevice ( IDENT = 'V04-001',
0                    ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL=LONG_RELATIVE)
0                    ) =
1 BEGIN
1
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 ! FACILITY:  SET Command
1 !
1 ! ABSTRACT:
1 !
1 !       This module implements the DCL command SET DEVICE.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX/VMS operating system, user mode
1 !
1 ! AUTHOR:  Gerry Smith                            23-Feb-1983
1 !
1 ! Modified by:
1 !
1 !       V04-001 AEW0006         Anne E. Warner          05-Sep-1984
1 !               Make changes in routine SETSERVED so that MSCP served
1 !               devices cannot be reserved.
1 !
1 !       V03-011 AEW0005         Anne E. Warner          28-Aug-1984
1 !               Change the check of the reference count from 2 to 1
1 !               before trying to SET DEV/NOSPOOL.  This is to insure
1 !               that there are no open files or second channels to the
1 !               device.  This change is in routine SETNOSPOOL.
1 !
1 !       V03-010 AEW0004         Anne E. Warner          10-Aug-1984
1 !               Add a check in SETPROT that if the device is unowned
1 !               and has no ACL present to require SYSPRV before its
```

```
1 !                     protection can be changed.
1 !
1 !           V03-009 AEW0003           Anne E. Warner          07-Aug-1984
1 !                   Add the error message set$_mscpnotld for the SET DEV/SER
1 !                   return status is ss$_devoffline.  This message is much
1 !                   more clear to tell the user that the server code is not
1 !                   loaded
1 !
1 !           V03-008 AEW0002           Anne E. Warner          02-Aug-1984
1 !                   Comment out code concerning the /NOWRITE and /CONTROLLER
1 !                   qualifiers of SET DEVICE/SERVED.  These qualifiers are
1 !                   no longer legal qualifiers but the code remains for future
1 !                   work.
1 !
1 !           V03-007 AEW0001           Anne E. Warner          25-Jul-1984
1 !                   Add sanity checks in SETSERVED for SET DEVICE/SERVED.
1 !                   Replace cli$_ivdevtype error message with set$_invdev
1 !                   error message which explains what happened better.
1 !
1 !           V03-006 DAS0001           David Solomon           09-Jul-1984
1 !                   Fix truncation errors; make nonexternal refs LONG_RELATIVE.
1 !
1 !           V03-005 ROW0342           Ralph O. Weber          10-APR-1984
1 !                   Add SET DEVICE /[NO]DIAGNOSTIC, to shutdown ports for
1 !                   class/port devices where running the device-local diagnostics
1 !                   inteacts adversely with normal host polling functions.
1 !
1 !           V03-004 LMP0221           L. Mark Pilant,         10-Apr-1984  12:22
1 !                   Change UCB$L_OWNUIC to ORB$L_OWNER and UCB$W_VPROT to
1 !                   ORB$W_PROT.
1 !
1 !           V03-003 PCG0001 Peter George            30-Mar-1984
1 !                   Add SET DEVICE/SERVED/NOWRITE/CONTROLLER=A.
1 !
1 !           V03-002 TCM0001 Trudy C. Matthews       19-Aug-1983
1 !                   Set DEV$V_CLU bit on SET DEVICE/DUAL_PORT command.
1 !
1 !           V03-001 GAS0112                         29-Mar-1983
1 !                   Remove references to old CLI interface.
1 !--
```

```
0098 1 !
0099 1 ! Include files
0100 1 !
0101 1 LIBRARY 'SYS$LIBRARY:LIB';                           ! VAX/VMS common definitions
0102 1 REQUIRE 'SRC$:SHOWDEF';                              ! SHOW common definitions
0201 1
0202 1
0203 1 !
0204 1 ! Define bit settings for the flags longword
0205 1 !
0206 1 MACRO
0207 1     set$v_log           =       0, 0, 1, 0%,         ! /LOG
0208 1     set$v_availset      =       0, 1, 1, 0%,         !
0209 1     set$v_avail         =       0, 2, 1, 0%,         ! /[NO]AVAILABLE
0210 1     set$v_dualset       =       0, 3, 1, 0%,         !
0211 1     set$v_dual          =       0, 4, 1, 0%,         ! /[NO]DUAL_PORT
0212 1     set$v_errorset      =       0, 5, 1, 0%,         !
0213 1     set$v_error         =       0, 6, 1, 0%,         ! /[NO]ERROR_LOG
0214 1     set$v_spool         =       0, 7, 1, 0%,         ! /SPOOL
0215 1     set$v_nospool       =       1, 0, 1, 0%,         ! /NOSPOOL
0216 1     set$v_prot          =       1, 1, 1, 0%,         ! SET PROT/DEVICE
0217 1     set$v_uic           =       1, 2, 1, 0%,         ! UIC set explicitly
0218 1     set$v_nowrite       =       1, 3, 1, 0%,         ! /NOWRITE
0219 1     set$v_diagset       =       1, 4, 1, 0%,         ! /[NO]DIAGNOSTIC
0220 1     set$v_diag          =       1, 5, 1, 0%;         !
0221 1
0222 1
0223 1 !
0224 1 ! Define the linkages for the routines to lock and unlock the I/O database,
0225 1 ! as well as allocate and deallocate non-paged pool.
0226 1 !
0227 1 LINKAGE
0228 1     IOLOCK = JSB (REGISTER = 4):                     ! R4 = process PCB
0229 1                  NOPRESERVE (1,2,3),                 ! R1-R3 destroyed
0230 1     ALLO   = JSB (REGISTER = 1;                      ! R1 = size (on input)
0231 1                  REGISTER = 1,                       ! R1 = size of block
0232 1                  REGISTER = 2):                      ! R2 = address of block
0233 1                  NOPRESERVE (3,4,5),                 ! R3, R4, R5 destroyed
0234 1     DEALLO = JSB (REGISTER = 0):                     ! R0 = address of block
0235 1                  NOPRESERVE (1,2,3,4,5);             ! R1-R5 destroyed
0236 1
0237 1
```

```
  1 !
  1 ! Table of contents
  1 !
  1
  1 FORWARD ROUTINE
  1     set$device : NOVALUE,                   ! Main module of SET DEVICE
  1     setbits,                                ! Set device characteristics
  1     setspool,                               ! Make a device spooled
  1     setnospool,                             ! Make a device un-spooled
  1     setprot,                                ! Set device protection and ownership
  1     setserved,                              ! MSCP serve a device
  1     setgetportname;                         ! Given a device determine the device
  1                                             ! name of its port
  1
  1 !
  1 ! External routines
  1 !
  1 EXTERNAL ROUTINE
  1     tran_queue,                             ! Convert input name to queue name
  1     get_prot : NOVALUE,                     ! Convert ASCII protection to binary
  1     parse_uic,                              ! Convert ASCII UIC to longword
  1     expand_prot : NOVALUE,                  ! Convert binary prot to ASCII
  1     lib$cvt_dtb,                            ! Convert ASCII to binary
  1     cli$get_value,                          ! Get value from CLI
  1     cli$present,                            ! See if qualifier is present
  1     exe$alononpaged : ALLO,                 ! Get a chunk of non-paged pool
  1     exe$deanonpaged : DEALLO,               ! Deallocate non-paged pool
  1     sch$iolockw : IOLOCK,                   ! Lock the I/O database for write
  1     sch$iounlock : IOLOCK,                  ! Unlock the I/O database
  1     mscp$addunit;                           ! Alloc and fill in UCB for MSCP server
  1
  1 !
  1 ! External references
  1 !
  1 EXTERNAL
  1     ctl$gq_procpriv : $BBLOCK[8],           ! Process privilege mask
  1     ctl$gl_pcb      : REF $BBLOCK,          ! Address of process PCB
  1     ctl$gl_ccbbase;                         ! Beginning of channel control blocks
  1
  1 !
  1 ! Declare some shared messages
  1 !
P 1 $SHR_MSGDEF     (SET,119,LOCAL,
  1                 (invquaval,     error));
  1
  1 !
  1 ! Declare literals defined elsewhere
  1 !
  1 EXTERNAL LITERAL
  1     set$_writeerr,                          ! Error modifying device
  1     set$_devset1,                           ! characteristic set
  1     set$_devset2,
  1     set$_devpset,                           ! Device protection set
  1     set$_invdev,                            ! Device is invalid for requested operation
  1     set$_spooled,                           ! Device spooled
  1     set$_mscpnotld,                         ! MSCP-Server code not loaded
  1     set$_notuqport,                         ! Device doesn't have a UNIBUS/QBUS port
```

```
;  199      0295  1     cli$_devnotfor,                ! Device not mounted foreign
;  200      0296  1     cli$_devnotspl,                ! Device not spooled
;  201      0297  1     cli$_devalspl,                 ! Device already spooled
;  202      0298  1     cli$_absent,                   ! Qualifier absent
;  203      0299  1     cli$_negated,                  ! Qualifier explicitly negated
;  204      0300  1     cli$_present;                  ! Qualifier explicitly present
;  205      0301  1
```

```
1 GLOBAL ROUTINE set$device (dummy) : NOVALUE =
2 BEGIN
2
2 !++
2 ! Functional description
2 !
2 !       This is the routine for the SET DEVICE command.  It is called
2 !       from the SET command processor, and sets certain characteristics
2 !       of devices.
2 !
2 ! Inputs
2 !       None
2 !
2 ! Outputs
2 !       None
2 !
2 !----
2
2 BUILTIN
2     ACTUALCOUNT;
2
2 LOCAL
2     status,                                  ! Status return
2     uic : VOLATILE,                          ! UIC for device protection
2     prot_mask : WORD VOLATILE,               ! Mask to show what protections changed
2     new_prot : WORD VOLATILE,                ! New protections to set
2     result_prot : WORD VOLATILE,             ! Final device protection
2     dev_desc : $BBLOCK[dsc$c_s_bln],         ! Device descriptor
2     que_desc : $BBLOCK[dsc$c_s_bln],         ! Queuename descriptor
2     uic_desc : $BBLOCK[dsc$c_s_bln],         ! /OWNER_UIC descriptor
2     intdev_desc : $BBLOCK[dsc$c_s_bln],      ! Descriptor for intermediate device
2     flags : $BBLOCK[4] INITIAL(0),           ! Flags longword
2     info_desc : VECTOR[2],                   ! $GETCHN descriptor
2     info_block : $BBLOCK[12],                ! $GETCHN information block
2     chan : WORD,                             ! I/O channel
2     intchan : WORD,                          ! Intermediate device I/O channel
2     iosb : VECTOR[4,WORD],                   ! I/O status block
2     dev_string : VECTOR[16,BYTE],            ! Translated string for spooled dev
2     que_string : VECTOR[16,BYTE],            ! Translated string for que name
2     arglist : VECTOR[8],                     ! $CMKRNL argument list
2     cont_desc : $BBLOCK[dsc$c_s_bln],        ! Descriptor for controller name
2     controller : INITIAL(0),                 ! Controller letter
2     port_desc : $BBLOCK[dsc$c_s_bln],        ! Descriptor for port device name
2     port_string : VECTOR[ 20, BYTE ],        ! Port device name string
2     port_chan : WORD;                        ! Channel to port device
2
2 !
2 ! Collect the name of the device.
2 !
2 $init_dyndesc(dev_desc);                     ! Make the descriptors dynamic
2 $init_dyndesc(que_desc);
2 $init_dyndesc(uic_desc);
2 $init_dyndesc(intdev_desc);
2 $init_dyndesc(cont_desc);
2 $init_dyndesc(port_desc);
2
2 cli$get_value(%ASCID 'FILE',                 ! Get the device name
```

```
                                    dev_desc);

!
! See if the user has OPER privilege.
!
IF NOT .ctl$gq_procpriv[prv$v_oper]
THEN
    BEGIN
    SIGNAL(set$_writeerr, 1, dev_desc, ss$_nooper);
    RETURN;
    END;

!
! Assign a channel to the device.
!
IF NOT (status = $ASSIGN(DEVNAM = dev_desc,
                         CHAN   = chan))
THEN
    BEGIN
    SIGNAL(set$_writeerr, 1, dev_desc, .status);
    RETURN;
    END;


!
! See if logging is required.
!
flags[set$v_log] = cli$present(%ASCID 'LOG');

!
! If the device protection is to change, get the protections, as well as
! perhaps a UIC to give as the device's owner.  To get device protection
! changed, we first went thru the SET PROTECTION routine, which called
! this routine with a dummy argument.  So, if the number of actual arguments
! is 1, we know that it's really protection that should be changed.
!
IF ACTUALCOUNT() EQL 1
THEN
    BEGIN
    LOCAL
        buffer : VECTOR[20],
        desc : VECTOR[2];

    uic = 0;                                    ! Initialize UIC
    get_prot(prot_mask, new_prot);              ! Get protection masks

!
! If a UIC was specified, check it.
!
    IF (flags[set$v_uic] = cli$get_value(%ASCID 'OWNER_UIC', uic_desc))
    THEN IF NOT parse_uic(uic_desc, uic)
    THEN
        BEGIN
        SIGNAL(set$_invquaval, 2, uic_desc, %ASCID 'OWNER_UIC');
        RETURN;
        END;
```

```
   3 !
   3 ! Call the kernel mode routine to set the protection, and then simply
   3 ! return.
   3 !
   3      arglist[0] = 5;
   3      arglist[1] = flags;                           ! FLAGS says what to do
   3      arglist[2] = .ctl$gl_ccbbase - .chan;         ! Put CCB address here
   3      arglist[3] = prot_mask;                       ! Which class(es) to set
   3      arglist[4] = new_prot;                        ! New protections
   3      arglist[5] = uic;                             ! Final owner UIC
P  4      IF NOT (status = $CMKRNL(ROUTIN = setprot,
   4                               ARGLST = arglist))
   3      THEN
   4          BEGIN
   4          SIGNAL(set$_writeerr, 1, dev_desc, .status);
   4          END
   3      ELSE IF .flags[set$v_log]
   3      THEN
   4          BEGIN
   4          expand_prot(arglist, .new_prot, 1);
   4          SIGNAL(set$_devpset, 6, dev_desc,
   4                             .arglist[0],
   4                             .arglist[1],
   4                             .arglist[2],
   4                             .arglist[3],
   4                             .uic);
   3          END;
   3      RETURN;
   2      END;
   2 !
   2 ! Determine whether or not /[NO]DIAGNOSTIC was specified and whether it is
   2 ! setting or clearing the "diagnostic" state.
   2 !
   2 status = cli$present( %ASCID 'DIAGNOSTIC' );     ! DIAGNOSTIC qualifier present?
   2 IF .status NEQ cli$_absent                        ! If not absent,
   2 THEN                                              ! then ...
   3     BEGIN
   3     flags[set$v_diagset] = 1;                     ! indicate it is present
   3     flags[set$v_diag] = .status;                  ! save its value (yes/no)
   2     END;
   2 !
   2 ! One set of qualifiers do nothing more than set/clear bits in the
   2 ! device characteristics longword.
   2 !
   2 status = cli$present(%ASCID 'AVAILABLE');         ! See what CLI$PRESENT returned
   2 IF .status NEQ cli$_absent                        ! If the qualifier was
   2 THEN                                              ! explicitly mentioned,
   3     BEGIN                                         ! set the flag bit
   3     flags[set$v_availset] = 1;                    ! saying it was mentioned
   3     flags[set$v_avail] = .status;                 ! and set the yes/no bit
   2     END;
   2 !
   2 ! For all you Bliss hackers out there, a more concise way of expressing
   2 ! the above is shown below, for the /[NO]DUAL_PORT and /[NO]ERROR_LOGGING
   2 ! qualifiers.
   2 !
```

```
  2 flags[set$v_dualset] = ((flags[set$v_dual] = cli$present(%ASCID 'DUAL_PORT')) NEQ cli$_absent);
  2 flags[set$v_errorset] = ((flags[set$v_error] = cli$present(%ASCID 'ERROR_LOGGING')) NEQ cli$_absent);
  2
  2 !
  2 ! If a device is to SPOOLed, then a queue name and intermediate device
  2 ! must be acquired.
  2 !
  2 IF (flags[set$v_spool] = cli$present(%ASCID 'SPOOLED')) NEQ cli$_absent
  2 THEN
  3     BEGIN
  3     IF .flags[set$v_spool]
  3     THEN
  4         BEGIN
  4         IF NOT tran_queue(dev_desc, dev_string)
  4         THEN
  5             BEGIN
  5             SIGNAL(set$_invquaval, 2, dev_desc, %ASCID 'SPOOLED');
  5             RETURN;
  5             END
  4         ELSE
  5             BEGIN
  5             dev_desc[dsc$w_length] = .dev_string[0];
  5             dev_desc[dsc$a_pointer] = dev_string[1];
  4             END;
  4         IF NOT cli$get_value(%ASCID 'SPOOLED', que_desc)
  4         THEN
  5             BEGIN
  5             que_desc[dsc$w_length] = .dev_desc[dsc$w_length];
  5             que_desc[dsc$a_pointer] = .dev_desc[dsc$a_pointer];
  5             END
  4         ELSE
  5             BEGIN
  5             IF NOT tran_queue(que_desc, que_string)
  5             THEN
  6                 BEGIN
  6                 SIGNAL(set$_invquaval, 2, que_desc, %ASCID 'SPOOLED');
  6                 RETURN;
  6                 END
  5             ELSE
  6                 BEGIN
  6                 que_desc[dsc$w_length] = .que_string[0];
  6                 que_desc[dsc$a_pointer] = que_string[1];
  5                 END;
  4             END;
  4         IF NOT cli$get_value(%ASCID 'SPOOLED', intdev_desc)
  4         THEN
  5             BEGIN
  5             intdev_desc[dsc$w_length] = %CHARCOUNT('SYS$DISK');
  5             intdev_desc[dsc$a_pointer] = UPLIT BYTE('SYS$DISK');
  4             END;
P 5         IF NOT (status = $ASSIGN(DEVNAM = intdev_desc,
  5                                  CHAN   = intchan))
  4         THEN
  5             BEGIN
  5             SIGNAL(set$_writeerr, 1, intdev_desc, .status);
  5             RETURN;
  4             END;
```

```
  3         END;
  3 !
  3 ! If /NOSPOOL, then no additional information is  needed.
  3 !
  3     flags[set$v_nospool] = NOT .flags[set$v_spool];      ! Set the /NOSPOOL flag
  2     END;
  2
  2
  2 !
  2 ! If /[NO]DIAGNOSTIC was specified, then locate the port device and
  2 ! send is an IO$_STOP or an IO$_INITIALIZE, as specified by the qualifier.
  2 !
  2 IF .flags[set$v_diagset]
  2 THEN
  3     BEGIN
  3     IF NOT .ctl$gq_procpriv[prv$v_diagnose]
  3     THEN
  4         BEGIN
  4         SIGNAL(set$_writeerr, 1, dev_desc, ss$_nodiagnose);
  4         RETURN;
  3         END;
  3     IF NOT .ctl$gq_procpriv[prv$v_phy_io]
  3     THEN
  4         BEGIN
  4         SIGNAL(set$_writeerr, 1, dev_desc, ss$_nophy_io);
  4         RETURN;
  3         END;
  3
  3     arglist[0] = 3;
  3     arglist[1] = .ctl$gl_ccbbase - .chan;
  3     arglist[2] = 20;
  3     arglist[3] = port_string;
P 4     IF NOT (status = $CMKRNL( ROUTIN = setgetportname,
  4                               ARGLST = arglist ))
  3     THEN
  4         BEGIN
  4         SIGNAL(set$_writeerr, 1, dev_desc, .status);
  4         RETURN;
  3         END;
  3
  3     port_desc[dsc$w_length] = .arglist[2];
  3     port_desc[dsc$a_pointer] = port_string;
  3
P 4     IF NOT (status = $ASSIGN( DEVNAM = port_desc,
  4                               CHAN   = port_chan ))
  3     THEN
  4         BEGIN
  4         SIGNAL(set$_writeerr, 1, dev_desc, .status);
  4         RETURN;
  3         END;
  3
  4         BEGIN
  4         LOCAL port_func;
  4         IF .flags[set$v_diag]
  4         THEN  port_func = IO$_STOP
  4         ELSE  port_func = IO$_INITIALIZE;
P 5         IF NOT (status = $QIOW( CHAN = .port_chan,
```

```
                                        FUNC = .port_func ))
        THEN
            BEGIN
            SIGNAL(set$_writeerr, 1, dev_desc, .status);
            RETURN;
            END;
        END;

    $DASSGN( CHAN = .port_chan );

    END;


!
! If the bit-tweaking options were requested, call the subroutine
! that does that.  Signal any errors or successes.
!
IF .flags[set$v_availset]
OR .flags[set$v_dualset]
OR .flags[set$v_errorset]
OR .flags[set$v_diagset]
THEN
    BEGIN
    arglist[0] = 2;
    arglist[1] = flags;
    arglist[2] = .ctl$gl_ccbbase - .chan;
    status = $CMKRNL(ROUTIN = setbits,
                     ARGLST = arglist);
    IF .status NEQ 1
    THEN
        BEGIN
        SIGNAL(set$_writeerr, 1, dev_desc, .status);
        END
    ELSE IF .flags[set$v_log]
    THEN
        BEGIN
        IF .flags[set$v_availset]
        THEN
            BEGIN
            IF .flags[set$v_avail]
            THEN SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'AVAILABLE')
            ELSE SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'NOAVAILABLE');
            END;
        IF .flags[set$v_dualset]
        THEN
            BEGIN
            IF .flags[set$v_dual]
            THEN SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'DUAL_PORT')
            ELSE SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'NODUAL_PORT');
            END;
        IF .flags[set$v_errorset]
        THEN
            BEGIN
            IF .flags[set$v_error]
            THEN SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'ERROR_LOGGING')
            ELSE SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'NOERROR_LOGGING');
            END;
```

```
  3         END;
  2     END;
  2
  2 !
  2 ! If the device is to be spooled or despooled, then call the routine that
  2 ! performs those functions.  Signal any errors or successes.
  2 !
  2 IF .flags[set$v_spool]
  2 THEN
  3     BEGIN
  3     arglist[0] = 4;
  3     arglist[1] = .ctl$gl_ccbbase - .chan;          ! Put CCB address here
  3     arglist[2] = .ctl$gl_ccbbase - .intchan;       ! Need the CCB of the int device
  3     arglist[3] = que_desc;                         ! Name of  the queue
  3     arglist[4] = intdev_desc;                      ! Name of the intermediate device
P 4     IF NOT (status = $CMKRNL(ROUTIN = setspool,
  4                              ARGLST = arglist))
  3     THEN
  4         BEGIN
  4         SIGNAL(set$_writeerr, 1, dev_desc, .status);
  4         END
  3     ELSE IF .flags[set$v_log]
  3     THEN SIGNAL(set$_spooled, 3, dev_desc, que_desc, intdev_desc);
  2     END;
  2
  2
  2 IF .flags[set$v_nospool]
  2 THEN
  3     BEGIN
  3     arglist[0] = 1;
  3     arglist[1] = .ctl$gl_ccbbase - .chan;          ! Put CCB address here
P 4     IF NOT (status = $CMKRNL(ROUTIN = setnospool,
  4                              ARGLST = arglist))
  3     THEN
  4         BEGIN
  4         SIGNAL(set$_writeerr, 1, dev_desc, .status);
  4         END
  3     ELSE IF .flags[set$v_log]
  3     THEN SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'NOSPOOL');
  2     END;
  2
  2 !
  2 ! If a device is to SERVED, then serve it.
  2 !
  2 IF cli$present(%ASCID 'SERVED')
  2 THEN
  3     BEGIN                          ! The nowrite flag is not used so permanently set
  3                                    ! set to 0.  However the original code is left
  3                                    ! if anyone wants to change this back.
  3     flags[set$v_nowrite] = 0; !cli$present(%ASCID 'WRITE') EQL cli$_negated;
  3     controller = 0;
  3 !
  3 !   Controller is not presently a legal qualifier but the code is left incase
  3 !   this changes in the future
  3 !     IF cli$present(%ASCID 'CONTROLLER')
  3 !     THEN
  3 !         IF cli$get_value(%ASCID 'CONTROLLER', cont_desc)
```

```
  3 !         THEN
  3 !             IF .cont_desc [dsc$w_length] GEQ 1
  3 !             THEN controller = .(.cont_desc [dsc$a_pointer]) <0,8,0>;
  3
  3      arglist[0] = 3;
  3      arglist[1] = flags;                        ! Flags
  3      arglist[2] = .ctl$gl_ccbbase - .chan;      ! CCB address
  3      arglist[3] = .controller;                  ! Controller letter
  3
P 4      IF NOT (status = $CMKRNL(ROUTIN = setserved,
  4                               ARGLST = arglist))
  3      THEN
  4          BEGIN
  4          SIGNAL(set$_writeerr, 1, dev_desc, .status);
  4          END
  4
  3      ELSE IF .flags[set$v_log]
  3      THEN
  4          BEGIN
  4          SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'SERVED');
  4          IF .flags[set$v_nowrite]
  4          THEN SIGNAL(set$_devset1, 2, dev_desc, %ASCID 'NOWRITE');
  4          cont_desc [dsc$w_length] = 1;
  4          IF .controller NEQ 0
  4          THEN SIGNAL(set$_devset2, 3, dev_desc, %ASCID 'CONTROLLER', cont_desc);
  3          END;
  2      END;
  2
  2 RETURN;
  1 END;
```

```
                                                        .TITLE   SETDEVICE
                                                        .IDENT   \V04-001\

                                                        .PSECT   $PLIT$,NOWRT,NOEXE,2

                           45  4C  49  46  00000 P.AAB: .ASCII   \FILE\
                              010E0004     00004 P.AAA: .LONG    17694724
                              00000000'    00008        .ADDRESS P.AAB
                           00  47  4F  4C  0000C P.AAD: .ASCII   \LOG\<0>
                              010E0003     00010 P.AAC: .LONG    17694723
                              00000000'    00014        .ADDRESS P.AAD
00 00 00 43 49 55 5F 52 45 4E  57  4F      00018 P.AAF: .ASCII   \OWNER_UIC\<0><0><0>
                              010E0009     00024 P.AAE: .LONG    17694729
                              00000000'    00028        .ADDRESS P.AAF
00 00 00 43 49 55 5F 52 45 4E  57  4F      0002C P.AAH: .ASCII   \OWNER_UIC\<0><0><0>
                              010E0009     00038 P.AAG: .LONG    17694729
                              00000000'    0003C        .ADDRESS P.AAH
00 00 43 49 54 53 4F 4E 47 41  49  44      00040 P.AAJ: .ASCII   \DIAGNOSTIC\<0><0>
                              010E000A     0004C P.AAI: .LONG    17694730
                              00000000'    00050        .ADDRESS P.AAJ
00 00 00 45 4C 42 41 4C 49 41  56  41      00054 P.AAL: .ASCII   \AVAILABLE\<0><0><0>
                              010E0009     00060 P.AAK: .LONG    17694729
                              00000000'    00064        .ADDRESS P.AAL
00 00 00 54 52 4F 50 5F 4C 41  55  44      00068 P.AAN: .ASCII   \DUAL_PORT\<0><0><0>
                              010E0009     00074 P.AAM: .LONG    17694729
```

```
                                                00000000'  00078           .ADDRESS P.AAN
00 00 47 4E 49 47 47 4F 4C 5F 52 4F 52 52 45  0007C  P.AAP:    .ASCII  \ERROR_LOGGING\<0><0><0>
                                           00  0008B
                                     010E000D  0008C  P.AAO:    .LONG   17694733
                                    00000000'  00090            .ADDRESS P.AAP
                     00 44 45 4C 4F 4F 50 53  00094  P.AAR:    .ASCII  \SPOOLED\<0>
                                     010E0007  0009C  P.AAQ:    .LONG   17694727
                                    00000000'  000A0            .ADDRESS P.AAR
                     00 44 45 4C 4F 4F 50 53  000A4  P.AAT:    .ASCII  \SPOOLED\<0>
                                     010E0007  000AC  P.AAS:    .LONG   17694727
                                    00000000'  000B0            .ADDRESS P.AAT
                     00 44 45 4C 4F 4F 50 53  000B4  P.AAV:    .ASCII  \SPOOLED\<0>
                                     010E0007  000BC  P.AAU:    .LONG   17694727
                                    00000000'  000C0            .ADDRESS P.AAV
                     00 44 45 4C 4F 4F 50 53  000C4  P.AAX:    .ASCII  \SPOOLED\<0>
                                     010E0007  000CC  P.AAW:    .LONG   17694727
                                    00000000'  000D0            .ADDRESS P.AAX
                     00 44 45 4C 4F 4F 50 53  000D4  P.AAZ:    .ASCII  \SPOOLED\<0>
                                     010E0007  000DC  P.AAY:    .LONG   17694727
                                    00000000'  000E0            .ADDRESS P.AAZ
                     4B 53 49 44 24 53 59 53  000E4  P.ABA:    .ASCII  \SYS$DISK\
         00 00 00 45 4C 42 41 4C 49 41 56 41  000EC  P.ABC:    .ASCII  \AVAILABLE\<0><0><0>
                                     010E0009  000F8  P.ABB:    .LONG   17694729
                                    00000000'  000FC            .ADDRESS P.ABC
         00 45 4C 42 41 4C 49 41 56 41 4F 4E  00100  P.ABE:    .ASCII  \NOAVAILABLE\<0>
                                     010E000B  0010C  P.ABD:    .LONG   17694731
                                    00000000'  00110            .ADDRESS P.ABE
         00 00 00 54 52 4F 50 5F 4C 41 55 44  00114  P.ABG:    .ASCII  \DUAL_PORT\<0><0><0>
                                     010E0009  00120  P.ABF:    .LONG   17694729
                                    00000000'  00124            .ADDRESS P.ABG
         00 54 52 4F 50 5F 4C 41 55 44 4F 4E  00128  P.ABI:    .ASCII  \NODUAL_PORT\<0>
                                     010E000B  00134  P.ABH:    .LONG   17694731
                                    00000000'  00138            .ADDRESS P.ABI
00 00 47 4E 49 47 47 4F 4C 5F 52 4F 52 52 45  0013C  P.ABK:    .ASCII  \ERROR_LOGGING\<0><0><0>
                                           00  0014B
                                     010E000D  0014C  P.ABJ:    .LONG   17694733
                                    00000000'  00150            .ADDRESS P.ABK
47 4E 49 47 47 4F 4C 5F 52 4F 52 52 45 4F 4E  00154  P.ABM:    .ASCII  \NOERROR_LOGGING\<0>
                                           00  00163
                                     010E000F  00164  P.ABL:    .LONG   17694735
                                    00000000'  00168            .ADDRESS P.ABM
                     00 4C 4F 4F 50 53 4F 4E  0016C  P.ABO:    .ASCII  \NOSPOOL\<0>
                                     010E0007  00174  P.ABN:    .LONG   17694727
                                    00000000'  00178            .ADDRESS P.ABO
                     00 00 44 45 56 52 45 53  0017C  P.ABQ:    .ASCII  \SERVED\<0><0>
                                     010E0006  00184  P.ABP:    .LONG   17694726
                                    00000000'  00188            .ADDRESS P.ABQ
                     00 00 44 45 56 52 45 53  0018C  P.ABS:    .ASCII  \SERVED\<0><0>
                                     010E0006  00194  P.ABR:    .LONG   17694726
                                    00000000'  00198            .ADDRESS P.ABS
                     00 45 54 49 52 57 4F 4E  0019C  P.ABU:    .ASCII  \NOWRITE\<0>
                                     010E0007  001A4  P.ABT:    .LONG   17694727
                                    00000000'  001A8            .ADDRESS P.ABU
         00 00 52 45 4C 4C 4F 52 54 4E 4F 43  001AC  P.ABW:    .ASCII  \CONTROLLER\<0><0>
                                     010E000A  001B8  P.ABV:    .LONG   17694730
                                    00000000'  001BC            .ADDRESS P.ABW
```

```
                                                        .EXTRN  TRAN_QUEUE, GET_PROT
                                                        .EXTRN  PARSE_UIC, EXPAND_PROT
                                                        .EXTRN  LIB$CVT_DTB, CLI$GET_VALUE
                                                        .EXTRN  CLI$PRESENT, EXE$ALONONPAGED
                                                        .EXTRN  EXE$DEANONPAGED
                                                        .EXTRN  SCH$IOLOCKW, SCH$IOUNLOCK
                                                        .EXTRN  MSCP$ADDUNIT, CTL$GQ_PROCPRIV
                                                        .EXTRN  CTL$GL_PCB, CTL$GL_CCBBASE
                                                        .EXTRN  SET$_WRITEERR, SET$_DEVSET1
                                                        .EXTRN  SET$_DEVSET2, SET$_DEVPSET
                                                        .EXTRN  SET$_INVDEV, SET$_SPOOLED
                                                        .EXTRN  SET$_MSCPNOTLD, SET$_NOTUQPORT
                                                        .EXTRN  CLI$_DEVNOTFOR, CLI$_DEVNOTSPL
                                                        .EXTRN  CLI$_DEVALSPL, CLI$_ABSENT
                                                        .EXTRN  CLI$_NEGATED, CLI$_PRESENT
                                                        .EXTRN  SYS$ASSIGN, SYS$CMKRNL
                                                        .EXTRN  SYS$QIOW, SYS$DASSGN

                                                        .PSECT  $CODE$,NOWRT,2

                                 0FFC 00000             .ENTRY  SET$DEVICE, Save R2,R3,R4,R5,R6,R7,R8,R9,-   ; 0302
                                                                R10,R11
             5B 00000000G    00   9E 00002              MOVAB   CLI$GET_VALUE, R11                          ;
             5A 00000000G    8F   D0 00009              MOVL    #CLI$_ABSENT, R10                           ;
             59 00000000G    8F   D0 00010              MOVL    #SET$_DEVSET1, R9                           ;
             58 00000000G    00   9E 00017              MOVAB   SYS$CMKRNL, R8                              ;
             57 00000000G    00   9E 0001E              MOVAB   CTL$GL_CCBBASE, R7                          ;
             56 00000000G    00   9E 00025              MOVAB   CLI$PRESENT, R6                             ;
             55 00000000G    00   9E 0002C              MOVAB   LIB$SIGNAL, R5                              ;
             54 00000000'    EF   9E 00033              MOVAB   P.AAA, R4                                   ;
             5E       FEEC   CE   9E 0003A              MOVAB   -276(SP), SP                                ;
                        0C   AE   D4 0003F              CLRL    FLAGS                                       ; 0303
                             53   D4 00042              CLRL    CONTROLLER                                  ;
        EC   AD 020E0000     8F   D0 00044              MOVL    #34471936, DEV_DESC                         ; 0351
                        F0   AD   D4 0004C              CLRL    DEV_DESC+4                                  ;
        E4   AD 020E0000     8F   D0 0004F              MOVL    #34471936, QUE_DESC                         ; 0352
                        E8   AD   D4 00057              CLRL    QUE_DESC+4                                  ;
        DC   AD 020E0000     8F   D0 0005A              MOVL    #34471936, UIC_DESC                         ; 0353
                        E0   AD   D4 00062              CLRL    UIC_DESC+4                                  ;
        D4   AD 020E0000     8F   D0 00065              MOVL    #34471936, INTDEV_DESC                      ; 0354
                        D8   AD   D4 0006D              CLRL    INTDEV_DESC+4                               ;
      0084   CE 020E0000     8F   D0 00070              MOVL    #34471936, CONT_DESC                        ; 0355
                      0088   CE   D4 00079              CLRL    CONT_DESC+4                                 ;
        7C   AE 020E0000     8F   D0 0007D              MOVL    #34471936, PORT_DESC                        ; 0356
                      0080   CE   D4 00085              CLRL    PORT_DESC+4                                 ;
                        EC   AD   9F 00089              PUSHAB  DEV_DESC                                    ; 0358
                             54   DD 0008C              PUSHL   R4                                          ;
             6B              02   FB 0008E              CALLS   #2, CLI$GET_VALUE                           ;
   08 00000000G   00              02   E0 00091         BBS     #2, CTL$GQ_PROCPRIV+2, 1$                   ; 0364
             7E       2894   8F   3C 00099              MOVZWL  #10388, -(SP)                               ; 0367
                           0440   31 0009E              BRW     50$                                         ;
                             7E   7C 000A1   1$:        CLRQ    -(SP)                                       ; 0375
                        08   AE   9F 000A3              PUSHAB  CHAN                                        ;
                        EC   AD   9F 000A6              PUSHAB  DEV_DESC                                    ;
      00000000G   00         04   FB 000A9              CALLS   #4, SYS$ASSIGN                              ;
             52              50   D0 000B0              MOVL    R0, STATUS                                  ;
             03              52   E8 000B3              BLBS    STATUS, 2$                                  ;
```

```
                                        0426  31 000B6      BRW     49$
                                   0C     A4  9F 000B9 2$:  PUSHAB  P.AAC                                  ; 0386
                          66              01  FB 000BC      CALLS   #1, CLI$PRESENT
  0C    AE         01     00              50  F0 000BF      INSV    R0, #0, #1, FLAGS
                          01              6C  91 000C5      CMPB    (AP), #1                               ; 0395
                                          03  13 000C8      BEQL    3$
                                        00A4  31 000CA      BRW     7$
                                   FC     AD  D4 000CD 3$:  CLRL    UIC                                    ; 0402
                                   F8     AD  9F 000D0      PUSHAB  NEW_PROT                               ; 0403
                                   FA     AD  9F 000D3      PUSHAB  PROT_MASK
                   00000000G      00      02  FB 000D6      CALLS   #2, GET_PROT
                                   DC     AD  9F 000DD      PUSHAB  UIC_DESC                               ; 0408
                                   20     A4  9F 000E0      PUSHAB  P.AAE
                          6B              02  FB 000E3      CALLS   #2, CLI$GET_VALUE
  0D    AE         01     02              50  F0 000E6      INSV    R0, #2, #1, FLAGS+1
                          19              50  E9 000EC      BLBC    R0, 4$
                                   FC     AD  9F 000EF      PUSHAB  UIC                                    ; 0409
                                   DC     AD  9F 000F2      PUSHAB  UIC_DESC
                   00000000G      00      02  FB 000F5      CALLS   #2, PARSE_UIC
                          09              50  E8 000FC      BLBS    R0, 4$
                                   34     A4  9F 000FF      PUSHAB  P.AAG                                  ; 0412
                                   DC     AD  9F 00102      PUSHAB  UIC_DESC
                                        0140  31 00105      BRW     16$
                    FF78     CD           05  D0 00108 4$:  MOVL    #5, ARGLIST                            ; 0420
                    FF7C     CD    0C     AE  9E 0010D      MOVAB   FLAGS, ARGLIST+4                       ; 0421
                             50           6E  3C 00113      MOVZWL  CHAN, R0                               ; 0422
          80    AD           67           50  C3 00116      SUBL3   R0, CTL$GL_CCBBASE, ARGLIST+8
                      84     AD    FA     AD  9E 0011B      MOVAB   PROT_MASK, ARGLIST+12                  ; 0423
                      88     AD    F8     AD  9E 00120      MOVAB   NEW_PROT, ARGLIST+16                   ; 0424
                      8C     AD    FC     AD  9E 00125      MOVAB   UIC, ARGLIST+20                        ; 0425
                                 FF78     CD  9F 0012A      PUSHAB  ARGLIST                                ; 0427
                             00000000V    EF  9F 0012E      PUSHAB  SETPROT
                          68              02  FB 00134      CALLS   #2, SYS$CMKRNL
                          52              50  D0 00137      MOVL    R0, STATUS
                          03              52  E8 0013A      BLBS    STATUS, 5$
                                        039F  31 0013D      BRW     49$
                          01       0C     AE  E8 00140 5$:  BLBS    FLAGS, 6$                              ; 0432
                                              04 00144      RET
                                          01  DD 00145 6$:  PUSHL   #1                                     ; 0435
                          7E       F8     AD  3C 00147      MOVZWL  NEW_PROT, -(SP)
                                 FF78     CD  9F 0014B      PUSHAB  ARGLIST
                   00000000G      00      03  FB 0014F      CALLS   #3, EXPAND_PROT
                                   FC     AD  DD 00156      PUSHL   UIC                                    ; 0441
                          7E       80     AD  7D 00159      MOVQ    ARGLIST+8, -(SP)                       ; 0439
                          7E     FF78     CD  7D 0015D      MOVQ    ARGLIST, -(SP)                         ; 0437
                                   EC     AD  9F 00162      PUSHAB  DEV_DESC                               ; 0436
                                          06  DD 00165      PUSHL   #6
                             00000000G    8F  DD 00167      PUSHL   #SET$_DEVPSET
                          65              08  FB 0016D      CALLS   #8, LIB$SIGNAL
                                              04 00170      RET                                            ; 0397
                                   48     A4  9F 00171 7$:  PUSHAB  P.AAI                                  ; 0450
                          66              01  FB 00174      CALLS   #1, CLI$PRESENT
                          52              50  D0 00177      MOVL    R0, STATUS
                          5A              52  D1 0017A      CMPL    STATUS, R10                            ; 0451
                                          0A  13 0017D      BEQL    8$
                             0D   AE      10  88 0017F      BISB2   #16, FLAGS+1                           ; 0454
  0D    AE         01     05              52  F0 00183      INSV    STATUS, #5, #1, FLAGS+1                ; 0455
```

```
                                       5C   A4  9F 00189 8$:    PUSHAB  P.AAK                                   ; 0461
                              66            01  FB 0018C        CALLS   #1, CLISPRESENT                         ;
                              52            50  D0 0018F        MOVL    R0, STATUS                              ;
                              5A            52  D1 00192        CMPL    STATUS, R10                             ; 0462
                                            0A  13 00195        BEQL    9$                                      ;
                         0C   AE            02  88 00197        BISB2   #2, FLAGS                               ; 0465
      0C   AE        01       02            52  F0 0019B        INSV    STATUS, #2, #1, FLAGS                   ; 0466
                                       70   A4  9F 001A1 9$:    PUSHAB  P.AAM                                   ; 0473
                              66            01  FB 001A4        CALLS   #1, CLISPRESENT                         ;
      0C   AE        01       04            50  F0 001A7        INSV    R0, #4, #1, FLAGS                       ;
                                            51  D4 001AD        CLRL    R1                                      ;
                              5A            50  D1 001AF        CMPL    R0, R10                                 ;
                                            02  13 001B2        BEQL    10$                                     ;
                                            51  D6 001B4        INCL    R1                                      ;
      0C   AE        01       03            51  F0 001B6 10$:   INSV    R1, #3, #1, FLAGS                       ;
                                     0088   C4  9F 001BC        PUSHAB  P.AAO                                   ; 0474
                              66            01  FB 001C0        CALLS   #1, CLISPRESENT                         ;
      0C   AE        01       06            50  F0 001C3        INSV    R0, #6, #1, FLAGS                       ;
                                            51  D4 001C9        CLRL    R1                                      ;
                              5A            50  D1 001CB        CMPL    R0, R10                                 ;
                                            02  13 001CE        BEQL    11$                                     ;
                                            51  D6 001D0        INCL    R1                                      ;
      0C   AE        01       05            51  F0 001D2 11$:   INSV    R1, #5, #1, FLAGS                       ;
                                     0098   C4  9F 001D8        PUSHAB  P.AAQ                                   ; 0480
                              66            01  FB 001DC        CALLS   #1, CLISPRESENT                         ;
      0C   AE        01       07            50  F0 001DF        INSV    R0, #7, #1, FLAGS                       ;
                              5A            50  D1 001E5        CMPL    R0, R10                                 ;
                                            03  12 001E8        BNEQ    12$                                     ;
                                          00B3  31 001EA        BRW     21$                                     ;
                                       0C   AE  95 001ED 12$:   TSTB    FLAGS                                   ; 0483
                                            03  19 001F0        BLSS    13$                                     ;
                                          009C  31 001F2        BRW     20$                                     ;
                                       A8   AD  9F 001F5 13$:   PUSHAB  DEV_STRING                              ; 0486
                                       EC   AD  9F 001F8        PUSHAB  DEV_DESC                                ;
                   00000000G  00            02  FB 001FB        CALLS   #2, TRAN_QUEUE                          ;
                              09            50  E8 00202        BLBS    R0, 14$                                 ;
                                     00A8   C4  9F 00205        PUSHAB  P.AAS                                   ; 0489
                                       EC   AD  9F 00209        PUSHAB  DEV_DESC                                ;
                                            3A  11 0020C        BRB     16$                                     ;
                         EC   AD       A8   AD  9B 0020E 14$:   MOVZBW  DEV_STRING, DEV_DESC                    ; 0494
                         F0   AD       A9   AD  9E 00213        MOVAB   DEV_STRING+1, DEV_DESC+4                ; 0495
                                       E4   AD  9F 00218        PUSHAB  QUE_DESC                                ; 0497
                                     00B8   C4  9F 0021B        PUSHAB  P.AAU                                   ;
                              6B            02  FB 0021F        CALLS   #2, CLISGET_VALUE                       ;
                              0C            50  E8 00222        BLBS    R0, 15$                                 ;
                         E4   AD       EC   AD  B0 00225        MOVW    DEV_DESC, QUE_DESC                      ; 0500
                         E8   AD       F0   AD  D0 0022A        MOVL    DEV_DESC+4, QUE_DESC+4                  ; 0501
                                            2C  11 0022F        BRB     18$                                     ; 0497
                                       98   AD  9F 00231 15$:   PUSHAB  QUE_STRING                              ; 0505
                                       E4   AD  9F 00234        PUSHAB  QUE_DESC                                ;
                   00000000G  00            02  FB 00237        CALLS   #2, TRAN_QUEUE                          ;
                              12            50  E8 0023E        BLBS    R0, 17$                                 ;
                                     00C8   C4  9F 00241        PUSHAB  P.AAW                                   ; 0508
                                       E4   AD  9F 00245        PUSHAB  QUE_DESC                                ;
                                            02  DD 00248 16$:   PUSHL   #2                                      ;
                                0077132A    8F  DD 0024A        PUSHL   #7803690                                ;
                                          0299  31 00250        BRW     52$                                     ;
```

```
                      E4  AD        98  AD  9B 00253 17$:   MOVZBW  QUE_STRING, QUE_DESC                  ; 0513
                      E8  AD        99  AD  9E 00258        MOVAB   QUE_STRING+1, QUE_DESC+4              ; 0514
                                    D4  AD  9F 0025D 18$:   PUSHAB  INTDEV_DESC                           ; 0517
                                  00D8  C4  9F 00260        PUSHAB  P.AAY                                 ;
                          6B            02  FB 00264        CALLS   #2, CLI$GET_VALUE                     ;
                          0A            50  E8 00267        BLBS    R0, 19$                               ;
                      D4  AD            08  B0 0026A        MOVW    #8, INTDEV_DESC                       ; 0520
                      D8  AD      00E0  C4  9E 0026E        MOVAB   P.ABA, INTDEV_DESC+4                  ; 0521
                                        7E  7C 00274 19$:   CLRQ    -(SP)                                 ; 0524
                                    0C  AE  9F 00276        PUSHAB  INTCHAN                               ;
                                    D4  AD  9F 00279        PUSHAB  INTDEV_DESC                           ;
             00000000G    00            04  FB 0027C        CALLS   #4, SYS$ASSIGN                        ;
                          52            50  D0 00283        MOVL    R0, STATUS                            ;
                          08            52  E8 00286        BLBS    STATUS, 20$                           ;
                                        52  DD 00289        PUSHL   STATUS                                ; 0527
                                    D4  AD  9F 0028B        PUSHAB  INTDEV_DESC                           ;
                                      0253  31 0028E        BRW     51$                                   ;
      50        0C    AE  01            07  EF 00291 20$:   EXTZV   #7, #1, FLAGS, R0                     ; 0534
                          50            50  D2 00297        MCOML   R0, R0                                ;
  0D  AE          01      00            50  F0 0029A        INSV    R0, #0, #1, FLAGS+1                   ;
                  03      0D  AE        04  E0 002A0 21$:   BBS     #4, FLAGS+1, 22$                      ; 0542
                                      00A4  31 002A5        BRW     30$                                   ;
          07 00000000G    00            06  E0 002A8 22$:   BBS     #6, CTL$GQ_PROCPRIV, 23$              ; 0545
                          7E      2834  8F  3C 002B0        MOVZWL  #10292, -(SP)                         ; 0548
                                        0D  11 002B5        BRB     24$                                   ;
          08 00000000G    00            06  E0 002B7 23$:   BBS     #6, CTL$GQ_PROCPRIV+2, 25$            ; 0551
                          7E      28B4  8F  3C 002BF        MOVZWL  #10420, -(SP)                         ; 0554
                                      021A  31 002C4 24$:   BRW     50$                                   ;
                    FF78  CD            03  D0 002C7 25$:   MOVL    #3, ARGLIST                           ; 0558
                          50            6E  3C 002CC        MOVZWL  CHAN, R0                              ; 0559
           FF7C   CD      67            50  C3 002CF        SUBL3   R0, CTL$GL_CCBBASE, ARGLIST+4         ;
                      80  AD            14  D0 002D5        MOVL    #20, ARGLIST+8                        ; 0560
                      84  AD        68  AE  9E 002D9        MOVAB   PORT_STRING, ARGLIST+12               ; 0561
                                  FF78  CD  9F 002DE        PUSHAB  ARGLIST                               ; 0563
                             00000000V  EF  9F 002E2        PUSHAB  SETGETPORTNAME                        ;
                          68            02  FB 002E8        CALLS   #2, SYS$CMKRNL                        ;
                          52            50  D0 002EB        MOVL    R0, STATUS                            ;
                          4A            52  E9 002EE        BLBC    STATUS, 28$                           ;
                      7C  AE        80  AD  B0 002F1        MOVW    ARGLIST+8, PORT_DESC                  ; 0570
                    0080  CE        68  AE  9E 002F6        MOVAB   PORT_STRING, PORT_DESC+4              ; 0571
                                        7E  7C 002FC        CLRQ    -(SP)                                 ; 0574
                                    10  AE  9F 002FE        PUSHAB  PORT_CHAN                             ;
                                  0088  CE  9F 00301        PUSHAB  PORT_DESC                             ;
             00000000G    00            04  FB 00305        CALLS   #4, SYS$ASSIGN                        ;
                          52            50  D0 0030C        MOVL    R0, STATUS                            ;
                          29            52  E9 0030F        BLBC    STATUS, 28$                           ;
                  05      0D  AE        05  E1 00312        BBC     #5, FLAGS+1, 26$                      ; 0583
                          50            03  D0 00317        MOVL    #3, PORT_FUNC                         ; 0584
                                        03  11 0031A        BRB     27$                                   ;
                          50            04  D0 0031C 26$:   MOVL    #4, PORT_FUNC                         ; 0585
                                        7E  7C 0031F 27$:   CLRQ    -(SP)                                 ; 0587
                                        7E  7C 00321        CLRQ    -(SP)                                 ;
                                        7E  7C 00323        CLRQ    -(SP)                                 ;
                                        7E  7C 00325        CLRQ    -(SP)                                 ;
                                        7E  D4 00327        CLRL    -(SP)                                 ;
                                        50  DD 00329        PUSHL   PORT_FUNC                             ;
                          7E        30  AE  3C 0032B        MOVZWL  PORT_CHAN, -(SP)                      ;
```

```
                                    7E  D4 0032F          CLRL    -(SP)                              ;
           00000000G  00            0C  FB 00331          CALLS   #12, SYS$QIOW                      ;
                      52            50  D0 00338          MOVL    R0, STATUS                         ;
                      03            52  E8 0033B  28$:    BLBS    STATUS, 29$                        ;
                                  019E  31 0033E          BRW     49$                                ;
                      7E       08   AE  3C 00341  29$:    MOVZWL  PORT_CHAN, -(SP)                   ; 0595
           00000000G  00            01  FB 00345          CALLS   #1, SYS$DASSGN                     ;
      12        0C    AE            01  E0 0034C  30$:    BBS     #1, FLAGS, 31$                     ; 0604
      0D        0C    AE            03  E0 00351          BBS     #3, FLAGS, 31$                     ; 0605
      08        0C    AE            05  E0 00356          BBS     #5, FLAGS, 31$                     ; 0606
      03        0D    AE            04  E0 0035B          BBS     #4, FLAGS+1, 31$                   ; 0607
                                  0095  31 00360          BRW     42$                                ;
              FF78    CD            02  D0 00363  31$:    MOVL    #2, ARGLIST                        ; 0610
              FF7C    CD       0C   AE  9E 00368          MOVAB   FLAGS, ARGLIST+4                   ; 0611
                      50            6E  3C 0036E          MOVZWL  CHAN, R0                           ; 0612
  80  AD              67            50  C3 00371          SUBL3   R0, CTL$GL_CCBBASE, ARGLIST+8      ;
                             FF78   CD  9F 00376          PUSHAB  ARGLIST                            ; 0614
                        00000000V   EF  9F 0037A          PUSHAB  SETBITS                            ;
                      68            02  FB 00380          CALLS   #2, SYS$CMKRNL                     ;
                      52            50  D0 00383          MOVL    R0, STATUS                         ;
                      01            52  D1 00386          CMPL    STATUS, #1                         ; 0615
                                    0F  13 00389          BEQL    32$                                ;
                                    52  DD 0038B          PUSHL   STATUS                             ; 0618
                               EC   AD  9F 0038D          PUSHAB  DEV_DESC                           ;
                                    01  DD 00390          PUSHL   #1                                 ;
                        00000000G   8F  DD 00392          PUSHL   #SET$_WRITEERR                     ;
                                    5B  11 00398          BRB     41$                                ;
                      5A       0C   AE  E9 0039A  32$:    BLBC    FLAGS, 42$                         ; 0620
      19        0C    AE            01  E1 0039E          BBC     #1, FLAGS, 35$                     ; 0623
      06        0C    AE            02  E1 003A3          BBC     #2, FLAGS, 33$                     ; 0626
                             00F4   C4  9F 003A8          PUSHAB  P.ABB                              ; 0627
                                    04  11 003AC          BRB     34$                                ;
                             0108   C4  9F 003AE  33$:    PUSHAB  P.ABD                              ; 0628
                               EC   AD  9F 003B2  34$:    PUSHAB  DEV_DESC                           ;
                                    02  DD 003B5          PUSHL   #2                                 ;
                                    59  DD 003B7          PUSHL   R9                                 ;
                      65            04  FB 003B9          CALLS   #4, LIB$SIGNAL                     ;
      19        0C    AE            03  E1 003BC  35$:    BBC     #3, FLAGS, 38$                     ; 0630
      06        0C    AE            04  E1 003C1          BBC     #4, FLAGS, 36$                     ; 0633
                             011C   C4  9F 003C6          PUSHAB  P.ABF                              ; 0634
                                    04  11 003CA          BRB     37$                                ;
                             0130   C4  9F 003CC  36$:    PUSHAB  P.ABH                              ; 0635
                               EC   AD  9F 003D0  37$:    PUSHAB  DEV_DESC                           ;
                                    02  DD 003D3          PUSHL   #2                                 ;
                                    59  DD 003D5          PUSHL   R9                                 ;
                      65            04  FB 003D7          CALLS   #4, LIB$SIGNAL                     ;
      19        0C    AE            05  E1 003DA  38$:    BBC     #5, FLAGS, 42$                     ; 0637
      06        0C    AE            06  E1 003DF          BBC     #6, FLAGS, 39$                     ; 0640
                             0148   C4  9F 003E4          PUSHAB  P.ABJ                              ; 0641
                                    04  11 003E8          BRB     40$                                ;
                             0160   C4  9F 003EA  39$:    PUSHAB  P.ABL                              ; 0642
                               EC   AD  9F 003EE  40$:    PUSHAB  DEV_DESC                           ;
                                    02  DD 003F1          PUSHL   #2                                 ;
                                    59  DD 003F3          PUSHL   R9                                 ;
                      65            04  FB 003F5  41$:    CALLS   #4, LIB$SIGNAL                     ;
                               0C   AE  95 003F8  42$:    TSTB    FLAGS                              ; 0651
                                    61  18 003FB          BGEQ    44$                                ;
```

```
                FF78  CD             04  D0 003FD          MOVL    #4, ARGLIST                          ; 0654
                      50             67  D0 00402          MOVL    CTL$GL_CCBBASE, R0                   ; 0655
                      51             6E  3C 00405          MOVZWL  CHAN, R1                             ;
    FF7C  CD          50             51  C3 00408          SUBL3   R1, R0, ARGLIST+4                    ;
                      51       04    AE  3C 0040E          MOVZWL  INTCHAN, R1                          ; 0656
      80  AD          50             51  C3 00412          SUBL3   R1, R0, ARGLIST+8                    ;
                  84  AD       E4    AD  9E 00417          MOVAB   QUE_DESC, ARGLIST+12                 ; 0657
                  88  AD       D4    AD  9E 0041C          MOVAB   INTDEV_DESC, ARGLIST+16              ; 0658
                             FF78    CD  9F 00421          PUSHAB  ARGLIST                              ; 0660
                        00000000V    EF  9F 00425          PUSHAB  SETSPOOL                             ;
                      68             02  FB 0042B          CALLS   #2, SYS$CMKRNL                       ;
                      52             50  D0 0042E          MOVL    R0, STATUS                           ;
                      12             52  E8 00431          BLBS    STATUS, 43$                          ;
                                     52  DD 00434          PUSHL   STATUS                               ; 0663
                               EC    AD  9F 00436          PUSHAB  DEV_DESC                             ;
                                     01  DD 00439          PUSHL   #1                                   ;
                        00000000G    8F  DD 0043B          PUSHL   #SET$_WRITEERR                       ;
                      65             04  FB 00441          CALLS   #4, LIB$SIGNAL                       ;
                                     18  11 00444          BRB     44$                                  ; 0659
                      14       0C    AE  E9 00446  43$:    BLBC    FLAGS, 44$                           ; 0665
                               D4    AD  9F 0044A          PUSHAB  INTDEV_DESC                          ; 0666
                               E4    AD  9F 0044D          PUSHAB  QUE_DESC                             ;
                               EC    AD  9F 00450          PUSHAB  DEV_DESC                             ;
                                     03  DD 00453          PUSHL   #3                                   ;
                        00000000G    8F  DD 00455          PUSHL   #SET$_SPOOLED                        ;
                      65             05  FB 0045B          CALLS   #5, LIB$SIGNAL                       ;
                      42       0D    AE  E9 0045E  44$:    BLBC    FLAGS+1, 47$                         ; 0670
                FF78  CD             01  D0 00462          MOVL    #1, ARGLIST                          ; 0673
                      50             6E  3C 00467          MOVZWL  CHAN, R0                             ; 0674
    FF7C  CD          67             50  C3 0046A          SUBL3   R0, CTL$GL_CCBBASE, ARGLIST+4        ;
                             FF78    CD  9F 00470          PUSHAB  ARGLIST                              ; 0676
                        00000000V    EF  9F 00474          PUSHAB  SETNOSPOOL                           ;
                      68             02  FB 0047A          CALLS   #2, SYS$CMKRNL                       ;
                      52             50  D0 0047D          MOVL    R0, STATUS                           ;
                      0F             52  E8 00480          BLBS    STATUS, 45$                          ;
                                     52  DD 00483          PUSHL   STATUS                               ; 0679
                               EC    AD  9F 00485          PUSHAB  DEV_DESC                             ;
                                     01  DD 00488          PUSHL   #1                                   ;
                        00000000G    8F  DD 0048A          PUSHL   #SET$_WRITEERR                       ;
                                     0F  11 00490          BRB     46$                                  ;
                      0E       0C    AE  E9 00492  45$:    BLBC    FLAGS, 47$                           ; 0681
                             0170    C4  9F 00496          PUSHAB  P.ABN                                ; 0682
                               EC    AD  9F 0049A          PUSHAB  DEV_DESC                             ;
                                     02  DD 0049D          PUSHL   #2                                   ;
                                     59  DD 0049F          PUSHL   R9                                   ;
                      65             04  FB 004A1  46$:    CALLS   #4, LIB$SIGNAL                       ;
                             0180    C4  9F 004A4  47$:    PUSHAB  P.ABP                                ; 0688
                      66             01  FB 004A8          CALLS   #1, CLI$PRESENT                      ;
                      01             50  E8 004AB          BLBS    R0, 48$                              ;
                                         04 004AE          RET                                          ;
                  0D  AE             08  8A 004AF  48$:    BICB2   #8, FLAGS+1                          ; 0693
                                     53  D4 004B3          CLRL    CONTROLLER                           ; 0694
                FF78  CD             03  D0 004B5          MOVL    #3, ARGLIST                          ; 0705
                FF7C  CD       0C    AE  9E 004BA          MOVAB   FLAGS, ARGLIST+4                     ; 0706
                      50             6E  3C 004C0          MOVZWL  CHAN, R0                             ; 0707
      80  AD          67             50  C3 004C3          SUBL3   R0, CTL$GL_CCBBASE, ARGLIST+8        ;
                  84  AD             53  D0 004C8          MOVL    CONTROLLER, ARGLIST+12               ; 0708
```

```
                     FF78  CD  9F 004CC          PUSHAB  ARGLIST                    ; 0711
                00000000V  EF  9F 004D0          PUSHAB  SETSERVED                  ;
                       68  02  FB 004D6          CALLS   #2, SYS$CMKRNL             ;
                       52  50  D0 004D9          MOVL    R0, STATUS                 ;
                       11  52  E8 004DC          BLBS    STATUS, 53$                ;
                           52  DD 004DF  49$:    PUSHL   STATUS                     ; 0714
                       EC  AD  9F 004E1  50$:    PUSHAB  DEV_DESC                   ;
                           01  DD 004E4  51$:    PUSHL   #1                         ;
                00000000G  8F  DD 004E6          PUSHL   #SET$_WRITEERR             ;
                       65  04  FB 004EC  52$:    CALLS   #4, LIB$SIGNAL             ;
                               04 004EF          RET                                ; 0710
            40        0C  AE  E9 004F0  53$:    BLBC    FLAGS, 55$                 ; 0717
                     0190  C4  9F 004F4          PUSHAB  P.ABR                      ; 0720
                       EC  AD  9F 004F8          PUSHAB  DEV_DESC                   ;
                           02  DD 004FB          PUSHL   #2                         ;
                           59  DD 004FD          PUSHL   R9                         ;
                       65  04  FB 004FF          CALLS   #4, LIB$SIGNAL             ;
    0E         0D  AE  03  E1 00502          BBC     #3, FLAGS+1, 54$           ; 0721
                     01A0  C4  9F 00507          PUSHAB  P.ABT                      ; 0722
                       EC  AD  9F 0050B          PUSHAB  DEV_DESC                   ;
                           02  DD 0050E          PUSHL   #2                         ;
                           59  DD 00510          PUSHL   R9                         ;
                       65  04  FB 00512          CALLS   #4, LIB$SIGNAL             ;
           0084    CE      01  B0 00515  54$:    MOVW    #1, CONT_DESC              ; 0723
                           53  D5 0051A          TSTL    CONTROLLER                 ; 0724
                           16  13 0051C          BEQL    55$                        ;
                     0084  CE  9F 0051E          PUSHAB  CONT_DESC                  ; 0725
                     01B4  C4  9F 00522          PUSHAB  P.ABV                      ;
                       EC  AD  9F 00526          PUSHAB  DEV_DESC                   ;
                           03  DD 00529          PUSHL   #3                         ;
                00000000G  8F  DD 0052B          PUSHL   #SET$_DEVSET2              ;
                       65  05  FB 00531          CALLS   #5, LIB$SIGNAL             ;
                               04 00534  55$:    RET                                ; 0730

; Routine Size:  1333 bytes,    Routine Base:  $CODE$ + 0000
```

```
ROUTINE setbits (flags, ccb) =
BEGIN

!++
! Functional description
!
!       This is the kernel mode routine to set bits in the UCB. Based on
!       the bit settings of the FLAGS longword, the device database is
!       modified to show the specified characteristics.
!
! Inputs
!       FLAGS   - options longword
!
! Outputs
!       None.  The device database is modified.
!       If an error is detected, an appropriate error status is returned.
!
!----

MAP
    flags : REF $BBLOCK,
    ccb : REF $BBLOCK;

BIND
    ucb = .ccb[ccb$l_ucb] : $BBLOCK,
    char = ucb[ucb$l_devchar] : $BBLOCK,
    char2 = ucb[ucb$l_devchar2] : $BBLOCK;

!
! First some sanity checks.  If /[NO]AVAILABLE, the device must be a
! dismounted disk.  For /[NO]DUAL_PORT, the device must be a disk, and for
! /NODUAL, the disk must be dismounted
!
IF .flags[set$v_availset]                       ! If /[NO]AVAILABLE
OR .flags[set$v_dualset]                        ! or /[NO]DUAL_PORT
THEN
    BEGIN
    IF .ucb[ucb$b_devclass] NEQU dc$_disk       ! device must be disk
    THEN RETURN set$_invdev;
    IF .char[dev$v_mnt]                         ! disk must be dismounted
    THEN RETURN ss$_devnotdism;
    END;

!
! Set the bits as appropriate.
!
IF .flags[set$v_dualset]
    THEN BEGIN
        char[dev$v_dua] = .flags[set$v_dual];
        char2[dev$v_clu] = .flags[set$v_dual];
    END;
IF .flags[set$v_availset]
THEN char[dev$v_avl] = .flags[set$v_avail];
IF .flags[set$v_errorset]
THEN char[dev$v_elg] = .flags[set$v_error];
IF .flags[set$v_diagset]
THEN
```

```
;  694          0788  3       BEGIN
;  695          0789  3       ucb[ucb$v_online] = NOT .flags[set$v_diag];
;  696          0790  3       char[dev$v_avl] = NOT .flags[set$v_diag];
;  697          0791  2       END;
;  698          0792  2
;  699          0793  2  RETURN 1;
;  700          0794  1  END;



                                                  000C 00000 SETBITS:.WORD   Save R2,R3                      ; 0731
                              51         08   BC   D0 00002          MOVL    @CCB, R1                        ; 0755
                              52         38   A1   9E 00006          MOVAB   56(R1), R2                      ; 0756
                              50         04   AC   D0 0000A          MOVL    FLAGS, R0                       ; 0764
                 04           60              01   E0 0000E          BBS     #1, (R0), 1$                    ;
                 31           60              03   E1 00012          BBC     #3, (R0), 4$                    ; 0765
                              01         40   A1   91 00016 1$:      CMPB    64(R1), #1                      ; 0768
                                              08   13 0001A          BEQL    2$                              ;
                              50  00000000G   8F   D0 0001C          MOVL    #SET$_INVDEV, R0                ; 0769
                                                   04 00023          RET                                     ;
                 06           62              13   E1 00024 2$:      BBC     #19, (R2), 3$                   ; 0770
                              50       21B4   8F   3C 00028          MOVZWL  #8628, R0                       ; 0771
                                                   04 0002D          RET                                     ;
                 15           60              03   E1 0002E 3$:      BBC     #3, (R0), 4$                    ; 0777
         53      60           01              04   EF 00032          EXTZV   #4, #1, (R0), R3                ; 0779
         62      01           0F              53   F0 00037          INSV    R3, #15, #1, (R2)               ;
         53      60           01              04   EF 0003C          EXTZV   #4, #1, (R0), R3                ; 0780
    3C   A1      01           00              53   F0 00041          INSV    R3, #0, #1, 60(R1)              ;
                 0A           60              01   E1 00047 4$:      BBC     #1, (R0), 5$                    ; 0782
         53      60           01              02   EF 0004B          EXTZV   #2, #1, (R0), R3                ; 0783
         62      01           12              53   F0 00050          INSV    R3, #18, #1, (R2)               ;
                 0A           60              05   E1 00055 5$:      BBC     #5, (R0), 6$                    ; 0784
         53      60           01              06   EF 00059          EXTZV   #6, #1, (R0), R3                ; 0785
         62      01           16              53   F0 0005E          INSV    R3, #22, #1, (R2)               ;
                 14      01   A0              04   E1 00063 6$:      BBC     #4, 1(R0), 7$                   ; 0786
         50      01   A0      01              05   EF 00068          EXTZV   #5, #1, 1(R0), R0               ; 0789
                              50              50   D2 0006E          MCOML   R0, R0                          ;
    64   A1      01           04              50   F0 00071          INSV    R0, #4, #1, 100(R1)             ;
         62      01           12              50   F0 00077          INSV    R0, #18, #1, (R2)               ; 0790
                              50              01   D0 0007C 7$:      MOVL    #1, R0                          ; 0793
                                                   04 0007F          RET                                     ; 0794

; Routine Size:  128 bytes,    Routine Base:  $CODE$ + 0535
```

```
ROUTINE setspool (ccb, intccb, que_desc, int_desc) =
BEGIN

!++
! Functional description
!
!       This is the kernel mode routine to make a device spooled.
!
! Inputs
!       CCB      - address of the device's channel control block
!       INTCCB   - address of the intermediate device's CCB
!       QUE_DESC - descriptor for the queue name
!       INT_DESC - descriptor for the name of the intermediate device
!
! Outputs
!       None.  The device database is modified.
!       If an error is detected, an appropriate error status is returned.
!
!----

MAP
    ccb : REF $BBLOCK,
    intccb : REF $BBLOCK,
    que_desc : REF $BBLOCK,
    int_desc : REF $BBLOCK;

BIND
    ucb = .ccb[ccb$l_ucb] : $BBLOCK,
    char = ucb[ucb$l_devchar] : $BBLOCK,
    int_ucb = .intccb[ccb$l_ucb] : $BBLOCK,
    int_char = int_ucb[ucb$l_devchar] : $BBLOCK,
    int_vcb = .int_ucb[ucb$l_vcb] : $BBLOCK;

LOCAL
    status;


!
! Lock the I/O database for write access.
!
sch$iolockw(.ctl$gl_pcb);

!
! Make checks to insure that the device is the right kind of device,
! that it is not assigned to some other process, and that the intermediate
! device is a file-oriented disk.
!
status = 1;
IF .ucb[ucb$b_devclass] NEQU dc$_term              ! Device must be either a
AND .ucb[ucb$b_devclass] NEQU dc$_lp               ! terminal or printer
THEN status = set$_invdev
ELSE IF .char[dev$v_spl]
THEN status = cli$_devalspl                        ! Can't be already spooled
ELSE IF .ucb[ucb$w_refc] NEQ 1
THEN status = ss$_devassign                        ! Can't be assigned to anyone
ELSE IF NOT .int_char[dev$v_fod]                   ! Intermediate device must be
     OR NOT .int_char[dev$v_rnd]                   ! a file-oriented disk
```

```
2 THEN status = ss$_notfiledev
2 ELSE IF .int_char[dev$v_dmt]                          ! Intermediate device must be
2 OR NOT .int_char[dev$v_mnt]                           ! mounted
2 THEN status = ss$_devnotmount;
2
2 !
2 ! If the devices meet all the criteria, then try to allocate a chunk
2 ! of non-paged pool to put information about the queue.  If the
2 ! allocation request is not successful, indicate an error.
2 !
2 IF .status EQL 1
2 THEN
3     BEGIN
3     LOCAL
3         block : REF $BBLOCK,
3         size;
3
3     status = exe$alononpaged(irp$c_length; size, block);
3     IF NOT .status
3     THEN status = ss$_insfmem
3     ELSE
4         BEGIN
4 !
4 ! Move the queue's name into the block of non-paged pool, and put the
4 ! address of the block into the UCB.
4 !
4         status = 1;                           ! Put in return status = success
4         ucb[ucb$l_vcb] = .block;              ! Put block's address in UCB
4         block[vcb$w_size] = .size;            ! Set queue block's size
4         block[vcb$b_type] = dyn$c_vcb;  ! Say it's a VCB (pseudo)
4         block[vcb$b_status] = .que_desc[dsc$w_length];
4         CH$MOVE(.que_desc[dsc$w_length],
4                 .que_desc[dsc$a_pointer],
4                 block[vcb$w_trans]);
4
4 !
4 ! Now juggle the intermediate and spooled device UCB's to reflect this change.
4 ! Specifically, the intermediate device's reference count and transaction count
4 ! are incremented, the intermediate device UCB is stored in the spooled device
4 ! UCB, and the spooled device's spool bit is set.
4 !
4         int_ucb[ucb$w_refc] = .int_ucb[ucb$w_refc] + 1;
4         int_vcb[vcb$w_trans] = .int_vcb[vcb$w_trans] + 1;
4         ucb[ucb$l_amb] = int_ucb;                       ! Store intdev UCB
4         char[dev$v_spl] = 1;                            ! Set it spooled
4         ucb[ucb$l_pid] = 0;                             ! Clear owner field
3         END;
2     END;
2
2 !
2 ! Unlock the I/O database, set the IPL to 0, and return whatever
2 ! status.
2 !
2 sch$iounlock(.ctl$gl_pcb);
2 set_ipl(0);
2
2 RETURN .status;
```

```
;  816          0909  1 END;



                                        0FFC 00000 SETSPOOL:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11              ; 0795
                  5A 00000000G   00   9E 00002            MOVAB   CTL$GL_PCB, R10                                   ;
                  57         04   BC   D0 00009            MOVL    @CCB, R7                                          ; 0822
                  56         08   BC   D0 0000D            MOVL    @INTCCB, R6                                       ; 0824
                  58         34   A6   D0 00011            MOVL    52(R6), R8                                        ; 0826
                  54              6A   D0 00015            MOVL    CTL$GL_PCB, R4                                    ; 0835
                     00000000G   00   16 00018            JSB     SCH$IOLOCKW                                       ;
                  59              01   D0 0001E            MOVL    #1, STATUS                                        ; 0842
          42      8F         40   A7   91 00021            CMPB    64(R7), #66                                       ; 0843
                                  10   13 00026            BEQL    1$                                                ;
          43      8F         40   A7   91 00028            CMPB    64(R7), #67                                       ; 0844
                                  09   13 0002D            BEQL    1$                                                ;
                  59 00000000G   8F   D0 0002F            MOVL    #SET$_INVDEV, STATUS                              ; 0845
                                  3A   11 00036            BRB     7$                                                ;
    09    38      A7              06   E1 00038 1$:        BBC     #6, 56(R7), 2$                                    ; 0846
                  59 00000000G   8F   D0 0003D            MOVL    #CLI$_DEVALSPL, STATUS                            ; 0847
                                  2C   11 00044            BRB     7$                                                ;
          01      5C              A7   B1 00046 2$:        CMPW    92(R7), #1                                        ; 0848
                                  07   13 0004A            BEQL    3$                                                ;
          59      0848            8F   3C 0004C            MOVZWL  #2120, STATUS                                     ; 0849
                                  1F   11 00051            BRB     7$                                                ;
    05    39      A6              06   E1 00053 3$:        BBC     #6, 57(R6), 4$                                    ; 0850
    07    3B      A6              04   E0 00058            BBS     #4, 59(R6), 5$                                    ; 0851
                  59       01CC   8F   3C 0005D 4$:        MOVZWL  #460, STATUS                                      ; 0852
                                  0E   11 00062            BRB     7$                                                ;
    05    3A      A6              05   E0 00064 5$:        BBS     #5, 58(R6), 6$                                    ; 0853
    04    3A      A6              03   E0 00069            BBS     #3, 58(R6), 7$                                    ; 0854
                  59         7C   8F   9A 0006E 6$:        MOVZBL  #124, STATUS                                      ; 0855
                  01              59   D1 00072 7$:        CMPL    STATUS, #1                                        ; 0862
                                  46   12 00075            BNEQ    9$                                                ;
                  51         C4   8F   9A 00077            MOVZBL  #196, R1                                          ; 0869
                     00000000G   00   16 0007B            JSB     EXE$ALONONPAGED                                   ;
                  59              50   D0 00081            MOVL    R0, STATUS                                        ;
                  07              59   E8 00084            BLBS    STATUS, 8$                                        ; 0870
                  59       0124   8F   3C 00087            MOVZWL  #292, STATUS                                      ; 0871
                                  2F   11 0008C            BRB     9$                                                ;
                  59              01   D0 0008E 8$:        MOVL    #1, STATUS                                        ; 0878
          34      A7              52   D0 00091            MOVL    BLOCK, 52(R7)                                     ; 0879
          08      A2              51   B0 00095            MOVW    SIZE, 8(BLOCK)                                    ; 0880
          0A      A2              11   90 00099            MOVB    #17, 10(BLOCK)                                    ; 0881
                  50         0C   AC   D0 0009D            MOVL    QUE_DESC, R0                                      ; 0882
          0B      A2              60   90 000A1            MOVB    (R0), 11(BLOCK)                                   ;
0C  A2    04      B0              60   28 000A5            MOVC3   (R0), @4(R0), 12(BLOCK)                           ; 0885
                             5C   A6   B6 000AB            INCW    92(R6)                                            ; 0893
                             0C   A8   B6 000AE            INCW    12(R8)                                            ; 0894
          60      A7              56   D0 000B1            MOVL    R6, 96(R7)                                        ; 0895
          38      A7         40   8F   88 000B5            BISB2   #64, 56(R7)                                       ; 0896
                             2C   A7   D4 000BA            CLRL    44(R7)                                            ; 0897
                  54              6A   D0 000BD 9$:        MOVL    CTL$GL_PCB, R4                                    ; 0905
                     00000000G   00   16 000C0            JSB     SCH$IOUNLOCK                                      ;
```

```
                    12          00  DA 000C6        MTPR    #0, #18                               ; 0906
                    50          59  D0 000C9        MOVL    STATUS, R0                            ; 0908
                                    04 000CC        RET                                           ; 0909

; Routine Size:  205 bytes,    Routine Base:  $CODE$ + 05B5
```

```
ROUTINE setnospool (ccb) =
BEGIN

!++
! Functional description
!
!       This is the kernel mode routine to unspool a device.
!
! Inputs
!       CCB       - address of the device's channel control block
!
! Outputs
!       None.  The device database is modified.
!       If an error is detected, an appropriate error status is returned.
!
!----

MAP
    ccb : REF $BBLOCK;

BIND
    ucb = .ccb[ccb$l_ucb] : $BBLOCK,
    char = ucb[ucb$l_devchar] : $BBLOCK;

LOCAL
    int_ucb : REF $BBLOCK,
    int_vcb : REF $BBLOCK,
    status;


!
! Lock the I/O database for write access.
!
sch$iolockw(.ctl$gl_pcb);

!
! Make checks to insure that the device is the spooled, and that the reference
! count is correct (i.e., no one else has assigned a channel to it).
!
status = 1;                                     ! Assume everything's wonderful
IF NOT .char[dev$v_spl]
THEN status = cli$_devnotspl                    ! Must be spooled
ELSE IF .ucb[ucb$w_refc] GTR 1                  ! If the refcount is greater than
                                                ! one that means that there is a
                                                ! file open in which case the
                                                ! device cannot be despooled
THEN status = ss$_devassign                     ! Can't be assigned to anyone
ELSE

!
! If the device looks good, mark it as being no longer spooled, and
! reduce the reference count by one.
!
    BEGIN
    char[dev$v_spl] = 0;                        ! Device is not spooled
    ucb[ucb$w_refc] = .ucb[ucb$w_refc] - 1;     ! Decrement the ref count

```

```
0967 3 !
0968 3 ! Deallocate the chunk of pool that holds the queue name.
0969 3 !
0970 3
0971 3     exe$deanonpaged(.ucb[ucb$l_vcb]);          ! Return queue block to pool
0972 3     ucb[ucb$l_vcb] = 0;                        ! Clear pointer to queue block
0973 3
0974 3 !
0975 3 ! Decrement the transaction count of the intermediate device, since we're no
0976 3 ! longer spooling to it.
0977 3 !
0978 3     int_ucb = .ucb[ucb$l_amb];                 ! Locate intermediate device's
0979 3     int_vcb = .int_ucb[ucb$l_vcb];             ! UCB and VCB.
0980 3     int_vcb[vcb$w_trans] = .int_vcb[vcb$w_trans] -1;
0981 3
0982 3 !
0983 3 ! Clear the pointer to the intermediate device, since the soon-to-be-
0984 3 ! despooled device no longer needs it.  Also put the address of the
0985 3 ! intermediate device into the channel control block.
0986 3 !
0987 3     ccb[ccb$l_ucb] = .int_ucb;
0988 3     ucb[ucb$l_amb] = 0;
0989 2     END;
0990 2
0991 2 !
0992 2 ! Unlock the I/O database, set the IPL to 0, and return whatever
0993 2 ! status.
0994 2 !
0995 2 sch$iounlock(.ctl$gl_pcb);
0996 2 set_ipl(0);
0997 2
0998 2 RETURN .status;
0999 1 END;


                                 0FFC 00000 SETNOSPOOL:
                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 0910
               58 00000000G   00  9E 00002        MOVAB   CTL$GL_PCB, R8                          ;
               56        04   BC  D0 00009        MOVL    @CCB, R6                                ; 0931
               54             68  D0 0000D        MOVL    CTL$GL_PCB, R4                          ; 0943
                  00000000G   00  16 00010        JSB     SCH$IOLOCKW                             ;
               57             01  D0 00016        MOVL    #1, STATUS                              ; 0949
  09     38    A6             06  E0 00019        BBS     #6, 56(R6), 1$                          ; 0950
               57 00000000G   8F  D0 0001E        MOVL    #CLI$_DEVNOTSPL, STATUS                 ; 0951
                              34  11 00025        BRB     3$                                      ;
               01        5C   A6  B1 00027 1$:    CMPW    92(R6), #1                              ; 0952
                              07  1B 0002B        BLEQU   2$                                      ;
               57      0848   8F  3C 0002D        MOVZWL  #2120, STATUS                           ; 0956
                              27  11 00032        BRB     3$                                      ;
         38    A6        40   8F  8A 00034 2$:    BICB2   #64, 56(R6)                             ; 0964
                         5C   A6  B7 00039        DECW    92(R6)                                  ; 0965
               50        34   A6  D0 0003C        MOVL    52(R6), R0                              ; 0971
                  00000000G   00  16 00040        JSB     EXE$DEANONPAGED                         ;
                         34   A6  D4 00046        CLRL    52(R6)                                  ; 0972
```

```
              51        60  A6  D0 00049        MOVL    96(R6), INT_UCB                        ; 0978
              50        34  A1  D0 0004D        MOVL    52(INT_UCB), INT_VCB                   ; 0979
                        0C  A0  B7 00051        DECW    12(INT_VCB)                            ; 0980
        04    BC            51  D0 00054        MOVL    INT_UCB, @CCB                          ; 0987
                        60  A6  D4 00058        CLRL    96(R6)                                 ; 0988
              54            68  D0 0005B  3$:   MOVL    CTL$GL_PCB, R4                         ; 0995
                 00000000G  00  16 0005E        JSB     SCH$IOUNLOCK                           ;
              12            00  DA 00064        MTPR    #0, #18                                ; 0996
              50            57  D0 00067        MOVL    STATUS, R0                             ; 0998
                                04 0006A        RET                                            ; 0999
```

; Routine Size: 107 bytes,   Routine Base: $CODE$ + 0682

```
1 ROUTINE setprot (flags, ccb, mask, prot, uic) =
2 BEGIN
2
2 !++
2 ! Functional description
2 !
2 !       This is the kernel mode routine to set bits in the UCB. Based on
2 !       the bit settings of the FLAGS longword, the device database is
2 !       modified to show the specified characteristics.
2 !
2 ! Inputs
2 !       FLAGS    - options longword
2 !
2 ! Outputs
2 !       None.  The device database is modified.
2 !       If an error is detected, an appropriate error status is returned.
2 !
2 !----
2
2 MAP
2     flags : REF $BBLOCK,
2     ccb : REF $BBLOCK,
2     prot : REF VECTOR[,WORD],
2     mask : REF VECTOR[,WORD],
2     uic : REF VECTOR;
2
2 BIND
2     ucb = .ccb[ccb$l_ucb] : $BBLOCK,
2     orb = .ucb[ucb$l_orb] : $BBLOCK,
2     char = ucb[ucb$l_devchar] : $BBLOCK;
2
2 LOCAL
2     temp_prot : WORD;
2
2 !
2 ! Check that the device is not file-oriented.
2 !
2 IF .char[dev$v_fod]
2 THEN RETURN set$_invdev;
2
2 !
2 ! If the device is unowned and there is no ACL present, require SYSPRV to
2 ! change the ACL.
2 !
2
2 IF .orb[orb$l_owner] EQL 0
3 AND (IF .orb[orb$v_acl_queue]
3     THEN .orb[orb$l_aclfl] EQLA orb[orb$l_aclfl]
3     ELSE 1)
2 AND NOT .$BBLOCK [ctl$gl_pcb[pcb$q_priv], prv$v_sysprv]
2 THEN RETURN ss$_nopriv;
2
2 !
2 ! Set the new protection.  Then return the full device protection.
2 !
2 IF .orb[orb$v_prot_16]
2 THEN temp_prot = .orb[orb$w_prot]
```

```
;  966     1057  2 ELSE
;  967     1058  3     BEGIN
;  968     1059  3     temp_prot<0,4> = .(orb[orb$l_sys_prot])<0,4>;
;  969     1060  3     temp_prot<4,4> = .(orb[orb$l_own_prot])<0,4>;
;  970     1061  3     temp_prot<8,4> = .(orb[orb$l_grp_prot])<0,4>;
;  971     1062  3     temp_prot<12,4> = .(orb[orb$l_wor_prot])<0,4>;
;  972     1063  2     END;
;  973     1064  2 orb[orb$w_prot] = (.temp_prot AND NOT .mask[0]) OR (.prot[0] AND .mask[0]);
;  974     1065  2 prot[0] = .orb[orb$w_prot];
;  975     1066  2 orb[orb$v_prot_16] = 1;
;  976     1067  2
;  977     1068  2 !
;  978     1069  2 ! If no UIC was given, then take this process UIC as the owner.
;  979     1070  2 !
;  980     1071  2 IF NOT .flags[set$v_uic]
;  981     1072  2 THEN uic[0] = .$BBLOCK[.ctl$gl_pcb, pcb$l_uic];
;  982     1073  2 orb[orb$l_owner] = .uic[0];
;  983     1074  2
;  984     1075  2 RETURN 1;
;  985     1076  1 END;



                                             003C 00000 SETPROT:.WORD    Save R2,R3,R4,R5                       ; 1000
                     55 00000000G     00  9E 00002         MOVAB    CTL$GL_PCB, R5
                     51        08     BC  D0 00009         MOVL     @CCB, R1                               ; 1027
                     50        1C     A1  D0 0000D         MOVL     28(R1), R0                             ; 1028
          08    39   A1               06  E1 00011         BBC      #6, 57(R1), 1$                         ; 1037
                     50 00000000G     8F  D0 00016         MOVL     #SET$_INVDEV, R0                       ; 1038
                                          04 0001D         RET
                                      60  D5 0001E 1$:     TSTL     (R0)                                   ; 1045
                                      1C  12 00020         BNEQ     3$
          0A    0B   A0               01  E1 00022         BBC      #1, 11(R0), 2$                         ; 1046
                     51        28     A0  9E 00027         MOVAB    40(R0), R1                             ; 1047
                     51        28     A0  D1 0002B         CMPL     40(R0), R1
                                      0D  12 0002F         BNEQ     3$
                     51               65  D0 00031 2$:     MOVL     CTL$GL_PCB, R1                         ; 1049
          04  0087   C1               04  E0 00034         BBS      #4, 135(R1), 3$
                     50               24  D0 0003A         MOVL     #36, R0                                ; 1050
                                          04 0003D         RET
                     09        0B     A0  E9 0003E 3$:     BLBC     11(R0), 4$                             ; 1055
                     52        18     A0  9E 00042         MOVAB    24(R0), R2                             ; 1056
                     51               62  B0 00046         MOVW     (R2), TEMP_PROT
                                      1B  11 00049         BRB      5$
                     52        18     A0  9E 0004B 4$:     MOVAB    24(R0), R2                             ; 1059
  51          04                  00  62  F0 0004F         INSV     (R2), #0, #4, TEMP_PROT
  51          04      04       1C     A0  F0 00054         INSV     28(R0), #4, #4, TEMP_PROT              ; 1060
  51          04      08       20     A0  F0 0005A         INSV     32(R0), #8, #4, TEMP_PROT              ; 1061
  51          04      0C       24     A0  F0 00060         INSV     36(R0), #12, #4, TEMP_PROT             ; 1062
                     53               51  3C 00066 5$:     MOVZWL   TEMP_PROT, R3                          ; 1064
                     54        0C     BC  3C 00069         MOVZWL   @MASK, R4                              ;
                     53               54  CA 0006D         BICL2    R4, R3                                 ;
                     51        10     BC  3C 00070         MOVZWL   @PROT, R1                              ;
                     54        0C     BC  3C 00074         MOVZWL   @MASK, R4                              ;
                     54               54  D2 00078         MCOML    R4, R4                                 ;
```

```
                     51          54  CA 0007B         BICL2   R4, R1                        ;
          62         51          53  A9 0007E         BISW3   R3, R1, (R2)                  ;
                10   BC          62  B0 00082         MOVW    (R2), @PROT                   ; 1065
                0B   A0          01  88 00086         BISB2   #1, 11(R0)                    ; 1066
                     51      04  AC  D0 0008A         MOVL    FLAGS, R1                     ; 1071
          09    01   A1          02  E0 0008E         BBS     #2, 1(R1), 6$                 ;
                     51          65  D0 00093         MOVL    CTL$GL_PCB, R1                ; 1072
                14   BC    00BC  C1  D0 00096         MOVL    188(R1), @UIC                 ;
                     60      14  BC  D0 0009C 6$:     MOVL    @UIC, (R0)                    ; 1073
                     50          01  D0 000A0         MOVL    #1, R0                        ; 1075
                                 04  000A3            RET                                   ; 1076

; Routine Size:  164 bytes,    Routine Base:  $CODE$ + 06ED
```

```
1 ROUTINE setserved (flags, ccb, letter) =
2 BEGIN
2
2 !++
2 ! Functional description
2 !
2 !       This is the kernel mode routine to serve a device.
2 !
2 ! Inputs
2 !       FLAGS     - options longword
2 !       CCB       - address of the device's channel control block
2 !       LETTER    - controller letter
2 !
2 ! Outputs
2 !       None.  The device database is modified.
2 !       If an error is detected, an appropriate error status is returned.
2 !
2 !----
2
2 MAP
2     ccb : REF $BBLOCK,
2     flags : REF $BBLOCK;
2
2 BIND
2     ucb = .ccb[ccb$l_ucb] : $BBLOCK,
2     char = ucb[ucb$l_devchar] : $BBLOCK,
2     char2 = ucb[ucb$l_devchar2] : $BBLOCK,
2     cddb = .ucb[ucb$l_cddb] : $BBLOCK,
2     p_cddb =.ucb[ucb$l_2p_cddb] : $BBLOCK;
2
2 LOCAL
2     status,
2     stat1,
2     stat2;
2
2 !
2 ! First check to see if the device can be served
2 !
2 status = 0;
2
3 IF ((.ucb[ucb$b_devclass] eqlu dc$_disk) and    ! If all of these conditions
3     (.ucb[ucb$b_devtype]  nequ dt$_rx01) and    ! then ok so far
3     (.ucb[ucb$b_devtype]  nequ dt$_rx02) and
3     (.ucb[ucb$b_devtype]  nequ dt$_rx04) and
3     (.char2[dev$v_srv]    nequ 1))
2 THEN status = 1;
2
2 IF .status
2 THEN
2     IF  .char[dev$v_mnt] eqlu 1                 ! If the device is mounted
2     THEN                                        ! then it must be cluster
3         BEGIN
3         status = .char2[dev$v_clu];             ! accessible to be served.
3         IF NOT .status THEN RETURN ss$_devnotdism;
2         END;
2
2
```

```
2 stat1 = stat2 = 0;
2
2 IF .status                                    ! If ok so far then check rest
2 THEN                                          ! of conditions
2     IF .char2[dev$v_mscp]
2     THEN
3     BEGIN
3         stat1 = .cddb[cddb$w_cntrlflgs] AND mscp$m_cf_mlths;
3         IF .char2[dev$v_2p]
3         THEN
3             stat2 = .p_cddb[cddb$w_cntrlflgs] AND mscp$m_cf_mlths;
3         IF (.stat1 OR .stat2) NEQU 0          ! If the above conditions are
3         THEN                                  ! true then cannot serve device
3             status = 0;
2     END;
2 IF NOT .status                                ! If not a valid device to serve
2 THEN                                          ! return with invalid device
2     RETURN set$_invdev;                       ! status.  Else continue.
2 !
2 ! Lock the I/O database for write access.
2 !
2 sch$iolockw(.ctl$gl_pcb);
2
2 !
2 ! Call the routine to set the specified characteristics.
2 !
2 status = mscp$addunit(.ccb[ccb$l_ucb], .flags[set$v_nowrite], .letter);
2
2 !
2 ! Unlock the I/O database, set IPL back to 0, and return whatever status.
2 !
2 sch$iounlock(.ctl$gl_pcb);
2 set_ipl(0);
2 IF .status EQLU ss$_devoffline
2 THEN
2     RETURN set$_mscpnotld
2 ELSE
2     RETURN .status;
1 END;
```

```
                          0FFC 00000 SETSERVED:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 1077
     57 00000000G   00  9E 00002           MOVAB   CTL$GL_PCB, R7                          ;
     55        08   BC  D0 00009           MOVL    @CCB, R5                                ; 1101
     52      00BC   C5  D0 0000D           MOVL    188(R5), R2                             ; 1104
     51      00C0   C5  D0 00012           MOVL    192(R5), R1                             ; 1105
                    56  D4 00017           CLRL    STATUS                                  ; 1115
     01        40   A5  91 00019           CMPB    64(R5), #1                              ; 1117
                    1A  12 0001D           BNEQ    1$                                      ;
     10        41   A5  91 0001F           CMPB    65(R5), #16                             ; 1118
                    14  13 00023           BEQL    1$                                      ;
     0B        41   A5  91 00025           CMPB    65(R5), #11                             ; 1119
                    0E  13 00029           BEQL    1$                                      ;
```

```
                        0C         41    A5  91 0002B            CMPB    65(R5), #12                      ; 1120
                                         08  13 0002F            BEQL    1$                               ;
                                   3C    A5  95 00031            TSTB    60(R5)                           ; 1121
                                         03  19 00034            BLSS    1$                               ;
                        56               01  D0 00036            MOVL    #1, STATUS                       ; 1122
                        14               56  E9 00039 1$:        BLBC    STATUS, 2$                       ; 1124
           0F      3A   A5               03  E1 0003C            BBC     #3, 58(R5), 2$                   ; 1126
56    3C   A5           01               00  EF 00041            EXTZV   #0, #1, 60(R5), STATUS           ; 1129
                        06               56  E8 00047            BLBS    STATUS, 2$                       ; 1130
                        50       21B4    8F  3C 0004A            MOVZWL  #8628, R0                        ;
                                         04 0004F                RET                                      ;
                                         53  D4 00050 2$:        CLRL    STAT2                            ; 1134
                                         50  D4 00052            CLRL    STAT1                            ;
                        2A               56  E9 00054            BLBC    STATUS, 5$                       ; 1136
           22      3C   A5               05  E1 00057            BBC     #5, 60(R5), 4$                   ; 1138
                        50         28    A2  3C 0005C            MOVZWL  40(R2), STAT1                    ; 1141
                        50 FFFFFFFB      8F  CA 00060            BICL2   #-5, STAT1                       ;
           0B      3C   A5               04  E1 00067            BBC     #4, 60(R5), 3$                   ; 1142
                        53         28    A1  3C 0006C            MOVZWL  40(R1), STAT2                    ; 1144
                        53 FFFFFFFB      8F  CA 00070            BICL2   #-5, STAT2                       ;
                        50               53  C8 00077 3$:        BISL2   STAT2, R0                        ; 1145
                                         02  13 0007A            BEQL    4$                               ;
                                         56  D4 0007C            CLRL    STATUS                           ; 1147
                        08               56  E8 0007E 4$:        BLBS    STATUS, 6$                       ; 1149
                        50 00000000G     8F  D0 00081 5$:        MOVL    #SET$_INVDEV, R0                 ; 1151
                                         04 00088                RET                                      ;
                        54               67  D0 00089 6$:        MOVL    CTL$GL_PCB, R4                   ; 1155
                           00000000G     00  16 0008C            JSB     SCH$IOLOCKW                      ;
                                   0C    AC  DD 00092            PUSHL   LETTER                           ; 1160
                        50         04    AC  D0 00095            MOVL    FLAGS, R0                        ;
7E    01   A0           01               03  EF 00099            EXTZV   #3, #1, 1(R0), -(SP)             ;
                                         55  DD 0009F            PUSHL   R5                               ;
      00000000G         00               03  FB 000A1            CALLS   #3, MSCP$ADDUNIT                 ;
                        56               50  D0 000A8            MOVL    R0, STATUS                       ;
                        54               67  D0 000AB            MOVL    CTL$GL_PCB, R4                   ; 1165
                           00000000G     00  16 000AE            JSB     SCH$IOUNLOCK                     ;
                        12               00  DA 000B4            MTPR    #0, #18                          ; 1166
      00000084          8F               56  D1 000B7            CMPL    STATUS, #132                     ; 1167
                                         08  12 000BE            BNEQ    7$                               ;
                        50 00000000G     8F  D0 000C0            MOVL    #SET$_MSCPNOTLD, R0              ; 1171
                                         04 000C7                RET                                      ;
                        50               56  D0 000C8 7$:        MOVL    STATUS, R0                       ;
                                         04 000CB                RET                                      ; 1172

; Routine Size:  204 bytes,    Routine Base:  $CODE$ + 0791
```

```
1 ROUTINE setgetportname ( ccb, length, string ) =
2 BEGIN
2
2 !++
2 ! Functional description:
2 !
2 !       This routine uses the UCB stored in the input CCB to locate a PDT and
2 !       finally a port device UCB.  The port device UCB address is fed to
2 !       IOC$CVT_DEVNAM to build a device name string suitable for use in a
2 !       channel assignment.  Several consistancy checks are made along the way
2 !       and if any of them fails and error status is returned.
2 !
2 ! Inputs:
2 !
2 !       CCB     Channel Control Block address
2 !       LENGTH  maximum device name string length
2 !       STRING  beginning address for device name string
2 !
2 ! Outputs:
2 !
2 !       LENGTH  actual length of device name string
2 !       .STRING device name string
2 !--
2
2 MAP
2     ccb : REF $BBLOCK,
2     length : REF VECTOR[ 1, WORD ],
2     string : REF VECTOR[ 20, BYTE ];
2
2 LINKAGE
2     DEVNAM = JSB ( REGISTER = 0,                       ! R0 = length of name buffer
2                    REGISTER = 1,                       ! R1 = address of name buffer
2                    REGISTER = 4,                       ! R4 = conversion type flag
2                    REGISTER = 5;                       ! R5 = UCB address
2                    REGISTER = 1 );                     ! R1 = device name length
2
2 EXTERNAL ROUTINE
2     ioc$cvt_devnam : DEVNAM;                           ! Convert UCB to device name
2
2 BIND
2     ucb = .ccb[ccb$l_ucb] : $BBLOCK,
2     char2 = ucb[ucb$l_devchar2] : $BBLOCK;
2
2 LOCAL
2     status;
2
2 !
2 ! Only MSCP devices can be connected to a U/Q port.
2 !
2
2 IF NOT .char2[dev$v_mscp] THEN RETURN set$_notuqport;
2
2 !
2 ! Now, a decent contents for UCB$L_PDT has been assured.
2 !
2
3     BEGIN
```

```
; 1141          1230  3
; 1142          1231  3     BIND
; 1143          1232  3         pdt = .ucb[ucb$l_pdt] : $BBLOCK,
; 1144          1233  3         port_ucb = .pdt[pdt$l_ucb0] : $BBLOCK;
; 1145          1234  3
; 1146          1235  3     IF .pdt[pdt$b_pdt_type] NEQ pdt$c_pu
; 1147          1236  3     OR .port_ucb[ucb$b_devclass] NEQ dc$_bus
; 1148          1237  3     THEN RETURN set$_notuqport;
; 1149          1238  3
; 1150          1239  3     ! Lock the I/O database for write access, convert the UCB address to
; 1151          1240  3     ! a device name, and release the I/O database.
; 1152          1241  3
; 1153          1242  3     sch$iolockw(.ctl$gl_pcb);
; 1154          1243  3     status = IOC$CVT_DEVNAM( .length, .string, 0, port_ucb; length );
; 1155          1244  3     IF .status EQL ss$_bufferovf THEN status = set$_notuqport;
; 1156          1245  3     sch$iounlock(.ctl$gl_pcb);
; 1157          1246  3     set_ipl(0);
; 1158          1247  3
; 1159          1248  2     END;
; 1160          1249  2
; 1161          1250  2 RETURN .status;
; 1162          1251  1 END;


                                                          .EXTRN  IOC$CVT_DEVNAM

                                         OFFC 00000 SETGETPORTNAME:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1173
                     57 00000000G  8F  D0 00002           MOVL    #SET$_NOTUQPORT, R7                          ;
                     56 00000000G  00  9E 00009           MOVAB   CTL$GL_PCB, R6                               ;
                     50        04  BC  D0 00010           MOVL    @CCB, R0                                     ; 1213
          17     3C  A0            05  E1 00014           BBC     #5, 60(R0), 1$                               ; 1223
                     50      0084  C0  D0 00019           MOVL    132(R0), R0                                  ; 1232
                     55      00DC  C0  D0 0001E           MOVL    220(R0), R5                                  ; 1233
                     02        07  A0  91 00023           CMPB    7(R0), #2                                    ; 1235
                                   07  12 00027           BNEQ    1$                                           ;
          80     8F        40      A5  91 00029           CMPB    64(R5), #128                                 ; 1236
                                   04  13 0002E           BEQL    2$                                           ;
                     50            57  D0 00030 1$:       MOVL    R7, R0                                       ; 1237
                                       04 00033           RET                                                  ;
                     54            66  D0 00034 2$:       MOVL    CTL$GL_PCB, R4                               ; 1242
                        00000000G  00  16 00037           JSB     SCH$IOLOCKW                                  ;
                                   54  D4 0003D           CLRL    R4                                           ; 1243
                     50        08  AC  7D 0003F           MOVQ    LENGTH, R0                                   ;
                        00000000G  00  16 00043           JSB     IOC$CVT_DEVNAM                               ;
                     55            50  D0 00049           MOVL    R0, STATUS                                   ;
              08     AC            51  D0 0004C           MOVL    R1, LENGTH                                   ;
        00000601     8F            55  D1 00050           CMPL    STATUS, #1537                                ; 1244
                                   03  12 00057           BNEQ    3$                                           ;
                     55            57  D0 00059           MOVL    R7, STATUS                                   ;
                     54            66  D0 0005C 3$:       MOVL    CTL$GL_PCB, R4                               ; 1245
                        00000000G  00  16 0005F           JSB     SCH$IOUNLOCK                                 ;
                     12            00  DA 00065           MTPR    #0, #18                                      ; 1246
                     50            55  D0 00068           MOVL    STATUS, R0                                   ; 1250
                                       04 0006B           RET                                                  ; 1251
```

```
; Routine Size:  108 bytes,    Routine Base:  $CODE$ + 085D
```

```
; 1164          1252  1 END
; 1165          1253  0 ELUDOM

                                                        .EXTRN  LIB$SIGNAL

;                                PSECT SUMMARY
;
;       Name                   Bytes                         Attributes
;
;    $PLIT$                         448  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;    $CODE$                        2249  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        98         0        1000           00:01.9



;                                  COMMAND QUALIFIERS

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

; Size:          2249 code + 448 data bytes
; Run Time:         00:42.1
; Elapsed Time:     02:42.0
; Lines/CPU Min:     1787
; Lexemes/CPU-Min: 20352
; Memory Used:  380 pages
; Compilation Complete
```

0052 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SETDEVICE
LIS

SETDEVS
LIS

SETCLUSUB
LIS

SETDIR
LIS

SETACT
LIS

SETAUDIT
LIS

SETCLUSTR
LIS