F 5

```
BBBBBBBBBBBB        000000000       000000000     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBBBBBBBBBBB        000000000       000000000     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBBBBBBBBBBB        000000000       000000000     TTTTTTTTTTTTTTT     SSSSSSSSSSSS
BBB         BBB  000         000  000         000       TTT        SSS
BBB         BBB  000         000  000         000       TTT        SSS
BBB         BBB  000         000  000         000       TTT        SSS
BBB         BBB  000         000  000         000       TTT        SSS
BBB         BBB  000         000  000         000       TTT        SSS
BBB         BBB  000         000  000         000       TTT        SSS
BBBBBBBBBBBB     000         000  000         000       TTT           SSSSSSSSS
BBBBBBBBBBBB     000         000  000         000       TTT           SSSSSSSSS
BBBBBBBBBBBB     000         000  000         000       TTT           SSSSSSSSS
BBB         BBB  000         000  000         000       TTT                    SSS
BBB         BBB  000         000  000         000       TTT                    SSS
BBB         BBB  000         000  000         000       TTT                    SSS
BBB         BBB  000         000  000         000       TTT                    SSS
BBB         BBB  000         000  000         000       TTT                    SSS
BBB         BBB  000         000  000         000       TTT                    SSS
BBBBBBBBBBBB        000000000       000000000           TTT        SSSSSSSSSSSS
BBBBBBBBBBBB        000000000       000000000           TTT        SSSSSSSSSSSS
BBBBBBBBBBBB        000000000       000000000           TTT        SSSSSSSSSSSS
```

```
**FILE**ID**CONFIGURE                                    F 16




  CCCCCCCC    OOOOOO    NN      NN  FFFFFFFFFF    IIIIII      GGGGGGGG  UU      UU  RRRRRRRR    EEEEEEEEEE
  CCCCCCCC    OOOOOO    NN      NN  FFFFFFFFFF    IIIIII      GGGGGGGG  UU      UU  RRRRRRRR    EEEEEEEEEE
CC          OO      OO  NN      NN  FF              II      GG          UU      UU  RR      RR  EE
CC          OO      OO  NN      NN  FF              II      GG          UU      UU  RR      RR  EE
CC          OO      OO  NNNN    NN  FF              II      GG          UU      UU  RR      RR  EE
CC          OO      OO  NNNN    NN  FF              II      GG          UU      UU  RR      RR  EE
CC          OO      OO  NN  NN  NN  FFFFFFFF        II      GG          UU      UU  RRRRRRRR    EEEEEEEE
CC          OO      OO  NN  NN  NN  FFFFFFFF        II      GG          UU      UU  RRRRRRRR    EEEEEEEE
CC          OO      OO  NN    NNNN  FF              II      GG  GGGGGG  UU      UU  RR  RR      EE
CC          OO      OO  NN    NNNN  FF              II      GG  GGGGGG  UU      UU  RR  RR      EE
CC          OO      OO  NN      NN  FF              II      GG      GG  UU      UU  RR    RR    EE              ....
CC          OO      OO  NN      NN  FF              II      GG      GG  UU      UU  RR    RR    EE              ....
  CCCCCCCC    OOOOOO    NN      NN  FF            IIIIII      GGGGGG    UUUUUUUUUU  RR      RR  EEEEEEEEEE      ....
  CCCCCCCC    OOOOOO    NN      NN  FF            IIIIII      GGGGGG    UUUUUUUUUU  RR      RR  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
```

Table of contents

```
0000            .TITLE  CONFIGURE - PROCESS TO DYNAMICALLY CONFIGURE DEVICES
0000            .IDENT  'V04-000'
0000     ;
0000     ;****************************************************************************
0000     ;*                                                                          *
0000     ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000     ;*  ALL RIGHTS RESERVED.                                                    *
0000     ;*                                                                          *
0000     ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000     ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000     ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000     ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000     ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000     ;*  TRANSFERRED.                                                            *
0000     ;*                                                                          *
0000     ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000     ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000     ;*  CORPORATION.                                                            *
0000     ;*                                                                          *
0000     ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000     ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000     ;*                                                                          *
0000     ;*                                                                          *
0000     ;****************************************************************************
0000
0000     ;++
0000     ;
0000     ; Facility:  System configuration
0000     ;
0000     ; Abstract: CONFIGURE is used to dynamically configure VAX MSCP-served and HSC-
0000     ;           served disks and tapes.
0000     ;
0000     ; Environment: It is run as a process, in user, exec and kernel modes.
0000     ;
0000     ; Author:  Maryann Hinden, Creation date:  02-JUN-1983
0000     ;
0000     ; Modification History:
0000     ;
0000     ;       V03-004 WHM0001         Bill Matthews           11-Apr-1984
0000     ;               Purge working set before hibernating.
0000     ;
0000     ;       V03-003
0000     ;               Change value in BOO$GL_CONADP to indicate noadapter.
0000     ;
0000     ;       V03-002 WMC0001         Wayne Cardoza           11-Aug-1983
0000     ;               Polling must be reenabled in kernel mode.
0000     ;
0000     ;       V03-001 MSH0001         Maryann Hinden          14-Jul-1983
0000     ;               Add jacket routine BOO$CONFIGMN to image, and
0000     ;               remove some code.
0000     ;--
0000
0000     ;
0000     ; Include files:
0000     ;
0000            $ACFDEF                         ; Define autoconfiguration block
```

```
                    0000              $IODEF
                    0000              $IPLDEF
                    0000              $LCKDEF
                    0000              $PRCPOLDEF                      ; Define process poller mailbox offsets
                    0000              $SBDEF
                    0000              $SSDEF                          ; System status definitions
                    0000              $SYSGMSGDEF                     ; Sysgen messages
                    0000
                    0000      ;
                    0000      ; Equated Symbols
                    0000      ;
          00000123  0000      WRTATNFLG = <IO$_SETMODE!IO$M_WRTATTN>
          00000071  0000      READFLG   = <IO$_READVBLK!IO$M_NOW>
                    0000
          00000000  0000      SERVER = 0       ; Offsets into process info block
          00000010  0000      DEVICE = 16
          00000012  0000      DRIVER = 18
          0000001B  0000      SPPB   = 27
                    0000
                    0000      ;
                    0000      ; Macros
                    0000      ;
                    0000              .MACRO  PRCINFO  SERVER,DEVICE,DRIVER   ; Builds process info table
                    0000
                    0000              .PSECT  INFO_BLOCK                      ; Actual data area
                    0000
                    0000      $SERVERNAME$ = .
                    0000              .ASCII  /                /              ; Reserve room for all 16 chars
                    0000      $NAMEND$ = .                                    ; Remember end of block
                    0000      . = $SERVERNAME$                                ; Get back to beginning
                    0000              .ASCII  /SERVER/                        ; Now store the real name
                    0000      . = $NAMEND$                                    ; Go to end of block
                    0000              .ASCII  /DEVICE/                        ; Device name
                    0000              .ASCIC  /DRIVER/                        ; Driver name
                    0000              .LONG   0                               ; SPPB for polling this process
                    0000
                    0000              .PSECT  INFO_PTR                        ; A list of pointers to the data
                    0000
                    0000              .LONG   $SERVERNAME$
                    0000
                    0000              .ENDM
                    0000      ;
                    0000      ; Own Storage
                    0000      ;
              00000000                .PSECT  INFO_PTR
                    0000
                    0000      PROC_INFO:
                    0000              PRCINFO MSCP$DISK,DU,DUDRIVER           ; Build the process info table
                    0004              PRCINFO MSCP$TAPE,MU,TUDRIVER
                    0008
              00000008                .PSECT  INFO_PTR
          00000000  0008              .LONG   0                               ; Indicate end of table
          0000001C  000C              .BLKL   4                               ; Patch area for four more processes
                    001C
              0000003E                .PSECT  INFO_BLOCK
          000000BA  003E              .BLKB   <SPPB+4>*4                      ; Patch area for data
                    00BA
```

```
      00000000         .PSECT  NONPAGED_DATA,NOEXE,WRT
00000016  0000  FULL_NAME:      .BLKB   22              ; Storage area for cluster dev name
0000001B  0016  DEVNAME:        .BLKB    5              ; Storage area for short dev name
          001B
00000000  001B  EXIT_BLOCK:     .LONG   0               ; Data block for exit handler
00000251' 001F                  .LONG   EXIT_HANDLER
00000001  0023                  .LONG   1
0000002B' 0027                  .LONG   EXIT_STATUS
0000002F  002B  EXIT_STATUS:    .BLKL   1
          002F
00000033' 002F  KARGLST:        .LONG   SPPBARG         ; Argument list for CANCEL_POLL
00000000  0033  SPPBARG:        .LONG   0               ; kernel mode routine
          0037
00000038  0037  MSGBUFSIZ:      .LONG   PRCPOL$C_SIZ    ; Buffer used by mailbox read
00000073  003B  MSGBUF:         .BLKB   PRCPOL$C_SIZ
          0073
    0000  0073  MBXCHAN:        .WORD   0               ; Mailbox I/O channel
0000007D  0075  STATUS_BLOCK:   .BLKL   2               ; I/O completion status block
          007D
      00000000         .PSECT  PAGED_CODE,EXE,WRT
          0000  PURGE_LIMITS:                           ; Limits for purge working set
00000000  0000          .LONG   0                       ; Purge all of P0 and P1
7FFFFFFF  0004          .LONG   ^X7FFFFFFF
          0008
```

```
                     0008          .SBTTL  CONFIGURE - Configure devices
                     0008    ;++
                     0008    ;
                     0008    ;  PURPOSE
                     0008    ;       To start polling on cluster members in order to find out about
                     0008    ;       HSC- and MSCP-served devices on other systems.
                     0008    ;
                     0008    ;  INPUT
                     0008    ;       None
                     0008    ;
                     0008    ;  OUTPUT
                     0008    ;       None
                     0008    ;
                     0008    ;  FUNCTIONAL DESCRIPTION
                     0008    ;       This routine requests polling on all systems in the cluster
                     0008    ;       for all processes described in the process information table.
                     0008    ;       The process poller communicates with the CONFIGURE process via
                     0008    ;       a mailbox.  Once the polling requests have been sent out, a
                     0008    ;       write attention AST to the mailbox is issued, and the routine
                     0008    ;       hibernates waiting for input.
                     0008    ;
                     0008    ;       In order to cancel polling (and clean up properly) if the image
                     0008    ;       should terminate abnormally, this routine declares an exit handler.
                     0008    ;--
                     0008
                001C 0008          .ENTRY BOO$CONFIGURE, ^M<R2,R3,R4>
                     000A
                     000A    ;
                     000A    ;  Create mailbox used to communicate with process poller
                     000A    ;
                     000A          $CREMBX_S       prmflg  = #1,-
                     000A                          chan    = MBXCHAN,-
                     000A                          promsk  = #^XFF00
      60 50     E9   0025          BLBC    R0,10$
                     0028
                     0028    ;
                     0028    ;  Declare exit handler to be used when image exits
                     0028    ;
                     0028          $DCLEXH_S       desblk  = EXIT_BLOCK
      50 50     E9   0035          BLBC    R0,10$
                     0038
                     0038    ;
                     0038    ;  Now request polling on all processes
                     0038    ;
                     0038          $CMKRNL_S       REQ_POLL,(AP)
      3E 50     E9   0047          BLBC    R0,10$
                     004A
                     004A    ;
                     004A    ;  We are finished requesting polling.  Now set a write attention AST
                     004A    ;  and hibernate while waiting for responses from the poller.
                     004A    ;  (We assume that at least one call to SCS$POLL_MBX was successful).
                     004A    ;
                     004A          $QIO_S  chan    = MBXCHAN,-
                     004A                  func    = #WRTATNFLG,-
                     004A                          p1      = FOUND_PROC,-
                     004A                          p2      = PROC_INFO
      12 50     E9   0073          BLBC    R0,10$
```

```
                          0076
                          0076          $PURGWS_S inadr = PURGE_LIMITS  ; minimize system resources
                          0080          $HIBER_S
                      04  0087          RET
                          0088
                          0088
                          0088    ;
                          0088    ;  An error occured on the create mailbox, when calling the process
                          0088    ;  poller, or when issuing the QIO.  Send out the error message and terminate.
                          0088    ;  The exit handler (if declared at this point) will clean up.
                          0088    ;
50    007C8132 8F     D0  0088    10$:    MOVL    #SYSG$_CONFIGERR,R0
             FF6E'    30  008F            BSBW    PUTERROR
                          0092            $EXIT_S
                      04  009B            RET
                          009C    ;
                          009C    ;  Request polling on all processes we want to know about
                          009C    ;
                    0000  009C    REQ_POLL:       .WORD   0
                          009E
                          009E            SETIPL  #IPL$_ASTDEL
53    00000073'EF     3C  00A1            MOVZWL  MBXCHAN,R3                  ; Get channel address
54    00000000'EF     9E  00A8            MOVAB   PROC_INFO,R4                ; Get top of process table
        52    84      D0  00AF            MOVL    (R4)+,R2                    ; Get address of first process name
                          00B2
        50    53      D0  00B2    10$:    MOVL    R3,R0                       ; Channel # in R0 is arg to call
      00000000'EF     16  00B5            JSB     SCS$POLL_MBX                ; Request polling for this process
                          00BB
                          00BB    ;
                          00BB    ; R1 contains address of SPPB - need later to cancel polling
                          00BB    ; R2 is preserved and points to process info block
                          00BB    ;
         0C 50        E9  00BB            BLBC    R0,20$
     1B A2    51      D0  00BE            MOVL    R1,SPPB(R2)                 ; Save SPPB
        52    84      D0  00C2            MOVL    (R4)+,R2                    ; Get next process name
              EB      12  00C5            BNEQ    10$                         ; If NEQ, poll for it
        50    01      9A  00C7            MOVZBL  #SS$_NORMAL,R0              ; Indicate success
                          00CA
                          00CA    20$:
                          00CA            SETIPL  #0                          ; Lower IPL
                      04  00CD            RET                                 ; Return error to caller
                          00CE
```

```
                        00CE          .SBTTL  FOUND_PROC - A process has been found by the poller
                        00CE     ;++
                        00CE     ;
                        00CE     ;  PURPOSE
                        00CE     ;       Routine which is called when the process poller mailbox has been
                        00CE     ;       written into.
                        00CE     ;
                        00CE     ;  INPUT
                        00CE     ;       Mailbox messages - implicit
                        00CE     ;
                        00CE     ;  OUTPUT
                        00CE     ;       Processed messages
                        00CE     ;
                        00CE     ;  FUNCTIONAL DESCRIPTION
                        00CE     ;       This routine is called at AST level.  It first re-enables the
                        00CE     ;       write attention AST for the mailbox.  It then reads and processes
                        00CE     ;       messages until there are none left.
                        00CE     ;
                        00CE     ;--
                        00CE
                  007C  00CE          .ENTRY  FOUND_PROC, ^M<R2,R3,R4,R5,R6>
                        00D0
                        00D0     ;
                        00D0     ;  Before doing anything else, we requeue the write attention AST request
                        00D0     ;
                        00D0          $QIO_S  chan    = MBXCHAN,-
                        00D0                  func    = #WRTATNFLG,-
                        00D0                  p1      = FOUND_PROC,-
                        00D0                  p2      = PROC_INFO
           47 50    E9  00F6          BLBC    R0,30$
                        00F9
                        00F9     ;
                        00F9     ;  Now, read mailbox messages until there are none left
                        00F9     ;
                        00F9     10$:     $QIO_S  chan    = MBXCHAN,-
                        00F9                  func    = #READFLG,-
                        00F9                  iosb    = STATUS_BLOCK,-
                        00F9                  p1      = MSGBUF,-
                        00F9                  p2      = MSGBUFSIZ
           2B 50    E9  0126          BLBC    R0,40$
 54   00000075'EF   9E  0129          MOVAB   STATUS_BLOCK,R4         ; Get address of status block
     0870 8F   64   B1  0130          CMPW    (R4),#SS$_ENDOFFILE     ; Have we read all the msgs?
               08   13  0135          BEQL    20$                     ; If EQL, yes
            1A 64   E9  0137          BLBC    (R4),40$                ; If LBC, then some sort of error
             0022   30  013A          BSBW    PROCESS_MSG             ; Else the poller found something
               BA   11  013D          BRB     10$                     ; Look for more messages
                        013F
                    04  013F     20$:     RET
                        0140
                        0140     ;
                        0140     ;  An error has occurred when trying to requeue the write attention AST.
                        0140     ;  Have the image exit.
                        0140     ;
                        0140     30$:
 50   007C8132 8F   D0  0140          MOVL    #SYSG$_CONFIGERR,R0
            FEB6'   30  0147          BSBW    PUTERROR
                        014A          $EXIT_S
```

```
                                       04  0153   297          RET
                                           0154   298
                                           0154   299 ;
                                           0154   300 ;  An error has occurred when reading the mailbox message.  Send out the
                                           0154   301 ;  error message and dismiss the AST.
                                           0154   302 ;
                                           0154   303 40$:
                     50   007C8132 8F  D0  0154   304          MOVL    #SYSG$_CONFIGERR,R0
                                FEA2'  30  015B   305          BSBW    PUTERROR
                                       04  015E   306          RET
                                015F          .SBTTL  PROCESS_MSG - Do the work of configuring the device
                                015F     ;++
                                015F     ;
                                015F     ;  PURPOSE
                                015F     ;       Workhorse routine to actually configure the device database
                                015F     ;       for the server which has been found.
                                015F     ;
                                015F     ;  INPUT
                                015F     ;       MSGBUF - contains the actual message
                                015F     ;
                                015F     ;  OUTPUT
                                015F     ;       Configured device and driver
                                015F     ;
                                015F     ;  FUNCTIONAL DESCRIPTION
                                015F     ;       This routine uses the node name (contained in the message) together
                                015F     ;       with the information associated with the server process name to
                                015F     ;       construct a cluster device name.  It then calls the connect code
                                015F     ;       to actually construct the device database and load the class driver.
                                015F     ;--
                                015F
                                015F     PROCESS_MSG:
              007C 8F      BB   015F          PUSHR   #^M<R2,R3,R4,R5,R6>         ; Save registers touched here
       00000000'EF    00   FB   0163          CALLS   #0,BOO$CONRESET            ; Reset connect information
                                016A
                                016A     ;
                                016A     ;  Search through the list of processes we are looking for to see
                                016A     ;  if there is a match
                                016A     ;
   56    0000003B'EF       9E   016A          MOVAB   MSGBUF,R6                  ; Get address of message buffer
   54    00000000'EF       9E   0171          MOVAB   PROC_INFO,R4               ; Get address of process information
                                0178
              55   84      D0   0178     10$: MOVL    (R4)+,R5                   ; Get next entry
                   58      13   017B          BEQL    20$                 ; If EQL, no more entries & no match
     18 A6    65   10      29   017D          CMPC3   #16,SERVER(R5),PRCPOL$B_PRCNAM(R6)      ; Compare
                   F4           0182          BNEQ    10$                        ; If NEQ, try next one
                                0184
                                0184     ;
                                0184     ;  A match was found - save info needed for the connect call and build the
                                0184     ;  device name
                                0184     ;
       00000000'EF    01   CE   0184          MNEGL   #1,BOO$GL_CONADP           ; Don't use an adapter
            00000000'EF    D4   018B          CLRL    BOO$GL_CONCUNIT            ; Unit number always 0
            00000000'EF    D4   0191          CLRL    BOO$GL_CONAUNIT            ; Same for adapter unit
       00000000'EF    66   7D   0197          MOVQ    PRCPOL$L_SYSIDL(R6),BOO$GQ_CONSYSID ; Save the sys ID from msg
   00000000'EF    12 A5    9E   0[illegible]  MOVAB   DRIVER(R5),BOO$GL_CONDRV    ; Save the driver name from proc_info
            52    08 A6    9E   01A6          MOVAB   PRCPOL$T_NODNAM(R3),R2     ; Get node name arg from msg
                      20   BB   01AA          PUSHR   #^M<R5>                    ; Save pointer to proc_info
            55    10 A5    9E   01AC          MOVAB   DEVICE(R5),R5              ; Get device name arg from proc_info
                      55   10   01B0          BSBB    BLDNAME                    ; Construct the cluster device name
                      20   BA   01B2          POPR    #^M<R5>                    ; Restore
                                01B4
                                01B4     ;
                                01B4     ;  Connect the device - build the class device database, load the class driver,
                                01B4     ;  and initialize the device
                                01B4     ;
       00000000'EF    00   FB   01B4          CALLS   #0,BOO$CONNECT
                   12 50   E8   01BB          BLBS    R0,15$
```

```
                        01BE              $CMKRNL_S ROUTIN = 30$          ; Polling must be turned on from K mode
         0C 50     E9   01CD              BLBC    R0,25$
                        01D0
                        01D0     ;
                        01D0     ;  All done
                        01D0     ;
       007C 8F     BA   01D0     15$:     POPR    #^M<R2,R3,R4,R5,R6>     ; Restore registers touched here
                   05   01D4              RSB
                        01D5
                        01D5     ;
                        01D5     ;  There was no process name match - we got a spurious mailbox message
                        01D5     ;
                        01D5     20$:
 50   007C8132 8F  D0   01D5              MOVL    #SYSG$_CONFIGERR,R0
             FE21'  30  01DC     25$:     BSBW    PUTERROR
       007C 8F     BA   01DF              POPR    #^M<R2,R3,R4,R5,R6>     ; Restore registers touched here
                   05   01E3              RSB
                        01E4
                        01E4     ;
                        01E4     ;  There was an error connecting the device - CONNECT already let the
                        01E4     ;  world know.
                        01E4     ;
                        01E4     30$:
                 0000   01E4              .WORD   0
       52   66     9E   01E6              MOVAB   PRCPOL$L_SYSIDL(R6),R2  ; Get system ID
    51    1B A5    D0   01E9              MOVL    SPPB(R5),R1             ; Get SPPB
       50   01     9A   01ED              MOVZBL  #1,R0                   ; Re-enable polling
                        01F0              SETIPL  #IPL$_SCS               ; Raise IPL
   00000000'GF     16   01F3              JSB     G^SCS$POLL_MODE         ; Request polling again
                        01F9              SETIPL  #IPL$_ASTDEL            ; Restore IPL
         07 50     E8   01FC              BLBS    R0,35$
 50   007C813A 8F  D0   01FF              MOVL    #SYSG$_CANTPOLL,R0      ; Indicate unable to restart poll on
                   04   0206     35$:     RET
```

```
                                    0207          .SBTTL  BLDNAME
                                    0207     ;++
                                    0207     ;
                                    0207     ;  PURPOSE
                                    0207     ;        Construct cluster device name given the node name and the
                                    0207     ;        device prefix.
                                    0207     ;
                                    0207     ;  INPUT
                                    0207     ;        R2 - Address of the node name string (in counted ASCII)
                                    0207     ;        R5 - Address of the device prefix
                                    0207     ;
                                    0207     ;  OUTPUT
                                    0207     ;        FULL_NAME_PTR - contains address of complete device name string
                                    0207     ;        BOO$GL_CONDEV - contains pointer into complete device name string,
                                    0207     ;                        starting at device prefix
                                    0207     ;        All registers preserved.
                                    0207     ;
                                    0207     ;  FUNCTIONAL DESCRIPTION
                                    0207     ;        This routine builds a cluster device name of the form:
                                    0207     ;
                                    0207     ;  byte          0: count of chars in string
                                    0207     ;           1 to m: node name
                                    0207     ;              m+1: "$"
                                    0207     ;      m+2 to m+4: "xxA" , where xx is the device name used by a given server
                                    0207     ;--
                                    0207
                                    0207     BLDNAME:
                           3F   BB  0207          PUSHR   #^M<R0,R1,R2,R3,R4,R5>
          53   00000000'EF  9E  0209          MOVAB   FULL_NAME,R3            ; Pointer to output buffer
        00000000'GF    83   9E  0210          MOVAB   (R3)+,G^FULL_NAME_PTR   ; Set up ptr for connect
                                    0217          ASSUME  SB$T_NODENAME+16,EQ,SB$L_DDB ; Make sure size doesn't change
                  54   82   9A  0217          MOVZBL  (R2)+,R4                ; Get real length of string
                           34   BB  021A          PUSHR   #^M<R2,R4,R5>           ; Save regs destroyed by MOVC3
             63   62   54   28  021C          MOVC3   R4,(R2),(R3)            ; Store node name in buffer
                           34   BA  0220          POPR    #^M<R2,R4,R5>           ; Restore regs (R3 now points to next
                                    0222                                          ; byte in dest. buffer after node name)
                  83   24   90  0222          MOVB    #^A/$/,(R3)+            ; Set in separator
                  63   65   B0  0225          MOVW    (R5),(R3)               ; Store device prefix
           02 A3   41   8F  90  0228          MOVB    #^A/A/,2(R3)            ; Store controller letter
      00000000'EF   54   04   81  022D          ADDB3   #4,R4,FULL_NAME         ; String is ASCIC
        00000017'EF    63   D0  0235          MOVL    (R3),DEVNAME+1          ; Store device name
        00000016'EF    03   90  023C          MOVB    #3,DEVNAME              ; Store count
   00000000'EF   00000016'EF  9E  0243          MOVAB   DEVNAME,BOO$GL_CONDEV   ; Store address of device string
                           3F   BA  024E          POPR    #^M<R0,R1,R2,R3,R4,R5>
                                    0250
                                05  0250          RSB
                                    0251
```

```
                               0251              .SBTTL  EXIT_HANDLER
                               0251        ;++
                               0251        ;
                               0251        ;  PURPOSE
                               0251        ;         Cancel polling on mailbox (if any) at image exit.
                               0251        ;
                               0251        ;  INPUT
                               0251        ;         Saved SPPB addresses in PROC_INFO table.
                               0251        ;
                               0251        ;  OUTPUT
                               0251        ;         Cancelled polling.
                               0251        ;
                               0251        ;--
                               0251
                         001C  0251              .ENTRY  EXIT_HANDLER, ^M<R2,R3,R4>
                               0253
       53  00000000'EF     9E  0253              MOVAB   PROC_INFO,R3                 ; Get address of process info table
                               025A
              54     83    D0  025A    10$:      MOVL    (R3)+,R4                     ; Point to next info block
                     22    13  025D              BEQL    20$                          ; If EQL, end of table
   00000033'EF    1B A4    D0  025F              MOVL    SPPB(R4),SPPBARG             ; Get address of SPPB
                     F1    13  0267              BEQL    10$                          ; If EQL, we haven't polled for this process
                               0269              $CMKRNL_S       routin=CANCEL_POLL,- ; Cancel polling
                               0269                              arglst=KARGLST
                  1B A4    D4  027C              CLRL    SPPB(R4)                     ; Show no more polling for this process
                     D9    11  027F              BRB     10$                          ; Loop through table
                               0281
                               0281    20$:      $DELMBX_S       chan=MBXCHAN         ; Mark mailbox for deletion
                               028F              $CMEXEC_S       routin=DQLOCKS       ; Dequeue locks
              50     01    3C  029E              MOVZWL  #SS$_NORMAL,R0
                           04  02A1              RET
                               02A2
                               02A2        ;
                               02A2        ;  Kernel mode routine running at IPL$_ASTDEL which cancels the polling mailbox.
                               02A2        ;
                         0004  02A2              .ENTRY  CANCEL_POLL,^M<R2>
                               02A4
                               02A4              SETIPL  #IPL$_ASTDEL
            51    04 AC    D0  02A7              MOVL    4(AP),R1                     ; Get SPPB address
           00000000'EF     16  02AB              JSB     SCS$CANCEL_MBX               ; Cancel polling
                               02B1              SETIPL  #0
                           04  02B4              RET
                               02B5
                               02B5        ;
                               02B5        ;  Exec mode routine to dequeue all locks held
                               02B5        ;
                         0000  02B5              .ENTRY  DQLOCKS,^M<>
                               02B7
                               02B7              $DEQ_S  lkid    = #0,-
                               02B7                      flags   = #LCK$M_DEQALL
                           04  02C6              RET
                               02C7
                               02C7              .END
```

```
$$T1                      = 00000000                 SYS$CMEXEC                ********  GX    05
$NAMENDS                  = 0000002F R      03       SYS$CMKRNL                ********  GX    05
$SERVERNAMES              = 0000001F R      03       SYS$CREMBX                ********  GX    05
BLDNAME                     00000207 R      05       SYS$DCLEXH                ********  GX    05
BOO$CONFIGURE               00000008 RG     05       SYS$DELMBX                ********  GX    05
BOO$CONNECT                 ********    X   05       SYS$DEQ                   ********  GX    05
BOO$CONRESET                ********    X   05       SYS$EXIT                  ********  GX    05
BOO$GL_CONADP               ********    X   05       SYS$HIBER                 ********  GX    05
BOO$GL_CONAUNIT             ********    X   05       SYS$PURGWS                ********  GX    05
BOO$GL_CONCUNIT             ********    X   05       SYS$QIO                   ********  GX    05
BOO$GL_CONDEV               ********    X   05       SYSG$_CANTPOLL          = 007C813A
BOO$GL_CONDRV               ********    X   05       SYSG$_CONFIGERR         = 007C8132
BOO$GQ_CONSYSID             ********    X   05       WRTATNFLG               = 00000123
CANCEL_POLL                 000002A2 RG     05
DEVICE                    = 00000010
DEVNAME                     00000016 R      04
DQLOCKS                     000002B5 RG     05
DRIVER                    = 00000012
EXIT_BLOCK                  0000001B R      04
EXIT_HANDLER                00000251 RG     05
EXIT_STATUS                 0000002B R      04
FOUND_PROC                  000000CE RG     05
FULL_NAME                   00000000 R      04
FULL_NAME_PTR               ********    X   05
IO$M_NOW                  = 00000040
IO$M_WRTATTN              = 00000100
IO$_READVBLK              = 00000031
IO$_SETMODE               = 00000023
IPL$_ASTDEL               = 00000002
IPL$_SCS                  = 00000008
KARGLST                     0000002F R      04
LCK$M_DEQALL              = 00000001
MBXCHAN                     00000073 R      04
MSGBUF                      0000003B R      04
MSGBUFSIZ                   00000037 R      04
PR$_IPL                     ********    X   05
PRCPOL$B_PRCNAM           = 00000018
PRCPOL$C_SIZ              = 00000038
PRCPOL$L_SYSIDL           = 00000000
PRCPOL$T_NODNAM           = 00000008
PROCESS_MSG                 0000015F R      05
PROC_INFO                   00000000 R      02
PURGE_LIMITS                00000000 R      05
PUTERROR                    ********    X   05
READFLG                   = 00000071
REQ_POLL                    0000009C R      05
SB$C_DDB                  = 00000054
SB$T_NODENAME             = 00000044
SCS$CANCEL_MBX              ********    X   05
SCS$POLL_MBX                ********    X   05
SCS$POLL_MODE               ********    X   05
SERVER                    = 00000000
SPPB                      = 0000001B
SPPBARG                     00000033 R      04
SS$_ENDOFFILE             = 00000870
SS$_NORMAL                = 00000001
STATUS_BLOCK                00000075 R      04
```

```
                                        +----------------+
                                        ! Psect synopsis !
                                        +----------------+

PSECT name                        Allocation          PSECT No.  Attributes
----------                        ----------          ---------  ----------
.  ABS  .                         00000000  (    0.)  00 (   0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                             00000000  (    0.)  01 (   1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
INFO_PTR                          0000001C  (   28.)  02 (   2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
INFO_BLOCK                        000000BA  (  186.)  03 (   3.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
NONPAGED_DATA                     0000007D  (  125.)  04 (   4.)  NOPIC    USR    CON    REL    LCL NOSHR NOEXE   RD     WRT NOVEC BYTE
PAGED_CODE                        000002C7  (  711.)  05 (   5.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE

                                     +------------------------+
                                     ! Performance indicators !
                                     +------------------------+

Phase                       Page faults    CPU Time       Elapsed Time
-----                       -----------    --------       ------------
Initialization                      29     00:00:00.05    00:00:00.54
Command processing                 115     00:00:00.69    00:00:01.93
Pass 1                             320     00:00:10.32    00:00:21.76
Symbol table sort                    0     00:00:01.50    00:00:03.69
Pass 2                              98     00:00:02.12    00:00:03.49
Symbol table output                  9     00:00:00.12    00:00:02.67
Psect synopsis output                3     00:00:00.03    00:00:00.03
Cross-reference output               0     00:00:00.00    00:00:00.00
Assembler run totals               576     00:00:14.83    00:00:34.11

The working set limit was 1500 pages.
57546 bytes (113 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1005 non-local and 15 local symbols.
499 source lines were read in Pass 1, producing 35 object records in Pass 2.
33 pages of virtual memory were used to define 31 macros.

                                    +--------------------------+
                                    ! Macro library statistics !
                                    +--------------------------+

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[BOOTS.OBJ]BOOTS.MLB;1                  0
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                      6
_$255$DUA28:[SYSLIB]STARLET.MLB;2                  21
TOTALS (all libraries)                             27

1176 GETS were required to define 27 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:CONFIGURE/OBJ=OBJ$:CONFIGURE MSRC$:CONFIGURE/UPDATE=(ENH$:CONFIGURE)+EXECML$/LIB+LIB$:BOOTS.MLB/LIB
```

0037 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CONFIG
LIS

BTMEM8SS
LIS

BTMEM790
LIS

CONFIGURE
LIS

BOOTDEF
LIS

BOOTIO
LIS

CONFIGMN
LIS

BOOTDRIVR
LIS

BTMEM730
LIS

BTMEM750
LIS

BTMEM780
LIS

BOOTBLOCK
LIS

0038 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CVBTDRIVR
LIS

DDBTDRIVR
LIS

DMBTDRIVR
LIS

DQBTDRIVR
LIS

CONIO
LIS

LOADERSUB
LIS

CONFIGUTL
LIS

DLBTDRIVR
LIS

DXBTDRIVR
LIS

INITPGFIL
LIS

LOCKDATA
LIS

LOADDRIV
LIS

LOADER
LIS