

BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		000000000		000000000		TTTTTTTTTTTT		SSSSSSSSSS
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	
BBB	BBB	000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		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	
BBB	BBB	000	000	000	000	TTT	SSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSSSSSSSSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSSSSSSSSS	
BBBBBBBBBBBB		000000000		000000000		TTT	SSSSSSSSSS	

```
PPPPPPPP  UU      UU  BBBB8888  TTTTTTTTTT  DDDDDDDDD  RRRRRRRR  IIIIII  VV      VV  RRRRRRRR
PPPPPPPP  UU      UU  BBBB8888  TTTTTTTTTT  DDDDDDDDD  RRRRRRRR  IIIIII  VV      VV  RRRRRRRR
PP      PP  UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
PP      PP  UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
PP      PP  UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
PPPPPPPP  UU      UU  BBBB8888  TT      TT      DD      DD  RRRRRRRR  II      II  VV      VV  RRRRRRRR
PPPPPPPP  UU      UU  BBBB8888  TT      TT      DD      DD  RRRRRRRR  II      II  VV      VV  RRRRRRRR
PP      UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
PP      UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
PP      UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
PP      UU      UU  BB      BB      TT      DD      DD  RR      RR  II      II  VV      VV  RR      RR
UUUUUUUUUU  BBBB8888  TT      TT      DDDDDDDDD  RR      RR  IIIIII  VV      VV  RR      RR
UUUUUUUUUU  BBBB8888  TT      TT      DDDDDDDDD  RR      RR  IIIIII  VV      VV  RR      RR

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
LLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLL  IIIIII  SSSSSSSS
```

(2) 63
(3) 118
(4) 276
(5) 357

DECLARATIONS
UDA50 Bootstrap device initialization
UDA50 Bootstrap driver QIO
UDA50 Bootstrap device disconnect

```
0000 1      .TITLE PUBTDRIVR - UDA50 BOOT DRIVER
0000 2      .IDENT 'V04-000'
0000 3
0000 4
0000 5 :*****
0000 6 :*
0000 7 :*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 :*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 :*  ALL RIGHTS RESERVED.
0000 10 :*
0000 11 :*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 :*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 :*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 :*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 :*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 :*  TRANSFERRED.
0000 17 :*
0000 18 :*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 :*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 :*  CORPORATION.
0000 21 :*
0000 22 :*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 :*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 :*
0000 25 :*
0000 26 :*****
0000 27 :
0000 28
0000 29 :++
0000 30 : FACILITY:      BOOTS
0000 31
0000 32 : ABSTRACT:
0000 33 :   This module contains the bootstrap device driver for the
0000 34 :   UDA 50 disks.
0000 35
0000 36 : ENVIRONMENT:  IPL 31, kernel mode, code must be PIC
0000 37
0000 38 : AUTHOR:  Kerbey T. Altmann,   CREATION DATE:  20-Nov-1981
0000 39
0000 40 : MODIFIED BY:
0000 41
0000 42 :   V03-006 KTA3110      Kerbey T. Altmann      12-Mar-1984
0000 43 :   Set controller timeout to max for GEN booting.
0000 44
0000 45 :   V03-005 KDM0073      Kathleen D. Morse      23-Aug-1983
0000 46 :   Added $BQODEF for use by new version of TIMEDWAIT macro.
0000 47
0000 48 :   V03-004 KDM0059      Kathleen D. Morse      13-Jul-1983
0000 49 :   Replace time-wait loops that use IPR TODR with the
0000 50 :   new TIMEDWAIT macro.
0000 51
0000 52 :   V03-003 KTA3064      Kerbey T. Altmann      03-Jul-1983
0000 53 :   Fix page boundary problem.
0000 54
0000 55 :   V03-002 KTA3059      Kerbey T. Altmann      23-Jun-1983
0000 56 :   Add support for the boot device name.
0000 57 :
```

PUBTDRIVR
V04-000

- UDA50 BOOT DRIVER

C 5

15-SEP-1984 23:57:44 VAX/VMS Macro V04-00
4-SEP-1984 23:05:04 [BOOTS.SRC]PUBTDRIVR.MAR;1

Page 2
(1)

0000 58 :
0000 59 :
0000 60 :
0000 61 :--

V03-001 KTA3007 Kerbey T. Altmann
Fix problem with setting VMB\$V_SCS.

09-Oct-1982

.SBTTL DECLARATIONS

INCLUDE FILES:

```
0000 63
0000 64 :
0000 65 : INCLUDE FILES:
0000 66 :
0000 67 :
0000 68 $BQODEF ; Boot qio offset definitions
0000 69 $BTDDDEF ; Boot device types
0000 70 $IODEF ; I/O function codes
0000 71 $MSCPDEF ; MSCP definitions
0000 72 $PRDEF ; Processor registers
0000 73 $PTEDEF ; Page table entries
0000 74 $RPBDEF ; RPB offsets
0000 75 $SSDEF ; Status codes
0000 76 $UBADEF ; UBA definitions
0000 77 $SUBIDEF ; 11/750 UBA definitions
0000 78 $VADEF ; Virtual addresses
0000 79 $VMBARGDEF ; VMB argument list offsets
0000 80
0000 81 :
0000 82 : EQUATED SYMBOLS:
0000 83 :
0000 84 :
0000 85 UDAIP = 0
0000 86 UDASA = 2
0000 87 GO = 1
0000 88 OWN = 1015
0000 89 S1 = 11
0000 90 S4 = 14
0000 91 UMR = 494 ; Last-1 UNIBUS Mapping Register
0000 92
0000 93 :
0000 94 : OWN STORAGE:
0000 95 :
0000 96 :
0000 97 :
0000 98 : Boot driver table entry
0000 99 :
0000 100
0000 101 $BOOT_DRIVER DEVTYPE = BTDSK_UDA,- ; Device type (UDA50)
0000 102 SIZE = UD_DRVSIZE,- ; Driver size
0000 103 ADDR = START,- ; Driver starting address
0000 104 ENTRY = UD_DRIVER,- ; Driver entry point
0000 105 UNIT_INIT = UD_INIT,- ; Driver unit init entry
0000 106 UNIT_DISC = UD_DISC,- ; Driver disconnect entry
0000 107 DRVNAME = DSKDRVNAME,- ; Driver disk name
0000 108 AUXDRVNAME = PRTRDRVNAME,- ; Driver port name
0000 109 DEVNAME = DEVNAME ; Boot device name
0000 110
0000 111 START:
0000 112 DSKDRVNAME:
0000 113 .ASCIC /DUDRIVER.EXE/ ; Disk class driver filename
0000 114 PRTRDRVNAME:
0000 115 .ASCIC /PUDRIVER.EXE/ ; Port driver filename
```

```
00000000
00000002
00000001
00008000
0000000B
0000000E
000001EE
```

```
58 45 2E 52 45 56 49 52 44 55 44 00'
45 000C
0C 0000
000D
58 45 2E 52 45 56 49 52 44 55 50 00'
45 0019
0C 000D
```

PUBTDRIVR
V04-000

- UDA50 BOOT DRIVER
DECLARATIONS

E 5

15-SEP-1984 23:57:44 VAX/VMS Macro V04-00
4-SEP-1984 23:05:04 [BOOTS.SRC]PUBTDRIVR.MAR;1

Page 4
(2)

55 44 001A 116 DEVNAME:.ASCII /DU/

; Boot device name

```
001C 118 .SBTTL UDA50 Bootstrap device initialization
001C 119
001C 120 :++
001C 121 :
001C 122 : Inputs:
001C 123 :
001C 124 : R9 --> RPB
001C 125 : AP --> VMB argument list
001C 126 :
001C 127 : Outputs:
001C 128 :
001C 129 : R0 - status code
001C 130 :
001C 131 :--
001C 132 :
001C 133 UD_INIT:
001C 134 .ENABLE LSB
01FC 001C 135 .WORD ^M<R2,R3,R4,R5,R6,R7,R8>
001E 136
50 38 DB 001E 137 MFPR #PRS_MAPEN, R0 ; Get the mapping status
1A 50 E8 0021 138 BLBS R0,10$ ; If virtual, skip some set up
0024 139 :
0024 140 : Set up the SYSTEMID for the local UDA.
0024 141 :
51 34 A9 D0 0024 142 MOVL RPB$L_IOVEC(R9),R1 ; Point to iovec
00'A1 94 0028 143 CLRB B^<BOO$GB_UMR_DP-BOO$AL_VECTOR>(R1) ; Set for Direct Data Path
2C AC 01 D0 002B 144 ASSUME VMB$V_LOAD_SCS_EQ_0
24 A9 D0 002B 145 MOVL #1,VMB$L_F[AGS(AP) ; Set a flag to load SCS code
24 AC D0 002F 146 MOVL RPB$L_BOOTR2(R9),-
20 A9 B0 0032 147 VMB$B_SYSTEMID(AP) ; Low 32 bits is CSR of UDA
28 AC B0 0034 148 MOVW RPB$L_BOOTR1(R9),-
00 24 AC 2F E2 0037 149 VMB$B_SYSTEMID+4(AP) ; Hi 16 bits is TR of UBA
0039 150 BBSS #47,VMB$B_SYSTEMID(AP),10$ ; Set bit 47
003E 151 :
003E 152 : Set up an interrupt vector.
003E 153 :
1E A9 01FC 8F B0 003E 154 10$: MOVW #<127*4>,RPB$W_ROUBVEC(R9) ; Use the highest possible
0044 155 :
0044 156 : Set up a UNIBUS mapping register(s) to cover the ring and buffers.
0044 157 : To make things easy, we will grab the last two register (494 & 495).
0044 158 : These registers are necessary since the ring area will be accessed by
0044 159 : the controller which is a UNIBUS device that does not do any mapping.
0044 160 :
51 52 52 014A'CF 9E 0044 161 MOVAB W^INTTBL,R2 ; Get the address of the ring
02 A2 51 FFFFE00 8F CB 0049 162 BICL3 #^C<^X1FF>,R2,R1 ; Get the byte offset in page
52 0003DC00 8F C9 0051 163 BISL3 #<UMR@9>,R1,2(R2) ; Set in the UNIBUS addr
52 02 A2 10' C0 005A 164 ADDL S^#<RING-INTTBL>,2(R2) ; Make it the ring address
52 52 15 09 EF 005E 165 EXTZV #VAS$VPN,#VASS$VPN,R2,R2 ; Get the page frame
53 5C A940 D0 0063 166 ASSUME RPB$L_ADPVIR_EQ_RPB$L_ADPPHY+4
57 54 A940 D0 0063 167 MOVL RPB$L_ADPPHY(R9)[R0],R3 ; Get correct pointer to UBA reg
52 0C 50 E9 0068 168 ASSUME RPB$L_CSRVIR_EQ_RPB$L_CSRPHY+4
52 50 B942 D0 0068 169 MOVL RPB$L_CSRPHY(R9)[R0],R7 ; Get correct address of device CSR
52 FFE00000 8F CA 006D 170 BLBC R0,20$ ; If clr, then physical
54 0FB8 C3 DE 0070 171 MOVL @RPB$L_SVASPT(R9)[R2],R2 ; Virtual, get physical
84 52 000000DF'EF C9 0075 172 BICL #^C<PTESM_PFN>,R2 ; Now a physical PFN
0081 173 20$: MOVAL UBA$L_MAPT<UMR*4>(R3),R4 ; Get the last two UMR's
BISL3 VALID,R2,(R4)+ ; Set as valid w/PFN
```

```
64 52 000000DF'EF 52 D6 0089 175 INCL R2 ; Set next page just in case
C9 008B 176 BISL3 VALID,R2,(R4) ; Set as valid w/PFN
0093 177
0093 178 : Now go thru the ridiculously complicated startup sequence. This is a
0093 179 : fugue in four parts.
0093 180
53 58 02 D0 0093 181 MOVL #2,R8 ; Make two tries at this
014A'CF 9E 0096 182 RETRY: MOVAB W^INTTBL, R3
52 0B D0 009B 183 MOVL #S1, R2 ; Step flag
67 B4 009E 184 CLRW UDAIP(R7) ; Poke the controller's CSR
00A0 185
00A0 186 : Wait 100 microseconds.
00A0 187
00A0 188 TIME: TIMEDWAIT TIME=#1000*1000,- ; Wait for 10 seconds
00A0 189 INS1=<MOVW UDASA(R7),R4>,- ; Check the status register
00A0 190 INS2=<BLSS 25$>,- ; Bit 15 set is the error indicator
00A0 191 INS3=<BBS R2,R4,25$>,- ; Done with this step?
00A0 192 DONELBL=25$ ; Label for exiting wait loop
OD 50 E8 00D3 193 BLBS R0,26$ ; Br if not timed out
00D6 194
50 BD 58 F5 00D6 195 ERROR: SOBGTR R8,RETRY ; Try once again
0054 8F 3C 00D9 196 MOVZWL #SS$_CTRLERR,R0 ; Set failure status
04 00DE 197 RET
00DF 198
80000000 00DF 199 VALID: .LONG ^X80000000 ; Sign bit set
00E3 200
54 B5 00E3 201 26$: TSTW R4 ; Check status register for error
EF 19 00E5 202 BLSS ERROR ; Br if error
00E7 203
02 A7 83 B0 00E7 204 30$: MOVW (R3)+,UDASA(R7) ; Send the controller the next step
B1 52 0E F3 00EB 205 AOBLEQ #S4,R2,TIME ; Set for next step
00EF 206
00EF 207 : Initialization complete. Write the packet address in the ring.
00EF 208 : Writing the addresses must be deferred until this point because the
00EF 209 : hardware zeroes the entire ring as a memory write check.
00EF 210
00000040'8F C1 00EF 211 ADDL3 #<RSPPKT-RING>,-
5A'AF 4C'AF 00F5 212 B^INTTBL+2,B^RD ; Response packet
OC' C1 00F9 213 ADDL3 S^#<CMDPKT-RING>,-
5E'AF 4C'AF 00FB 214 B^INTTBL+2,B^CD ; Command packet
00FF 215
00FF 216 : Set the controller timeout to be the maximum
00FF 217
55 66'AF 9E 00FF 218 MOVAB B^CMDPKT,r5 ; Get the address of command packet
85 01 D0 0103 219 MOVL #1,(R5)+ ; Set command ref number
85 85 D4 0106 220 CLRL (R5)+ ; Reserved field
85 04 9A 0108 221 MOVZBL #MSCP$_K_OP_STCON,(R5)+ ; Set opcode to SET CTRL CHAR
85 85 D4 010B 222 CLRL (R5)+ ; Version and flags
85 FF 8F 9A 010D 223 MOVZBL #255,(R5)+ ; Set timeout to 255 seconds
85 85 7C 0111 224 CLRQ (R5)+ ; Clear time & date
65 D4 0113 225 CLRL (R5) ; Clear controller depend parameters
00CC 30 0115 226 BSBW IO ; Send it out
0118 227
0118 228 : Now bring the device on-line
0118 229
55 66'AF 9E 0118 230 MOVAB B^CMDPKT,r5 ; Get the address of command packet
85 01 D0 011C 231 MOVL #1,(R5)+ ; Set command ref number
```

```
85 64 A9 9A 011F 232 MOVZBL RPB$W UNIT(R9),(R5)+ ; Put unit number in cmd packet field
85 85 09 9A 0123 233 MOVZBL #MSCP$K_OP_ONLIN,(R5)+ ; Set opcode to bring drive online
      85 7C 0126 234 CLRQ (R5)+ ; Clear byte count, buff desc
      65 7C 0128 235 CLRQ (R5) ; buff desc and LBN
      00B7 30 012A 236 BSBW IO ; Send it out
      16 EF 012D 237 EXTZV #MSCP$V MTYP D1,-
      05 012F 238 #MSCP$S MTYP D1,-
      01B6'CF 0130 239 W^RSPPKT+ -
      51 51 08 78 0133 240 MSCP$L MEDIA_ID,R1 ; Pull out 2nd device character
      1B EF 0134 241 ASHL #8,R1,RT ; Stick it in high byte
      05 0138 242 EXTZV #MSCP$V MTYP D0,-
      B6'AF 013A 243 #MSCP$S MTYP D0,-
      52 013B 244 B^RSPPKT+ -
      51 4040 8F A8 013D 245 MSCP$L MEDIA_ID,R2 ; Pull out 1st device character
FED1 CF 52 51 A9 013E 246 BISW #^X4040,R1 ; Make ASCII characters
      04 0143 247 BISW3 R1,R2,DEVNAME ; Set into driver name
      0149 248 RET
      014A 249
      014A 250
      014A 251 .DISABLE LSB
      0000014A 014A 252 .=<. +1>8-2
      014A 253 ;
      014A 254 ; RINGS
      014A 255 ;
      014A 256 ;
      8000 014A 257 INTTBL: .WORD OWN ; Step 1 pattern
      00000000 014C 258 .LONG 0 ; Step 2 & 3 pattern
      0001 0150 259 .WORD GO
      0152 260
      0000 0000 0152 261 .WORD 0,0 ; Reserved
      0000 0156 262 CMDINT: .WORD 0 ; Command status word
      0000 0158 263 RSPINT: .WORD 0 ; Response status word
      015A 264 RING:
      00000000 015A 265 RD: .LONG 0 ; UNIBUS address of response ring
      00000000 015E 266 CD: .LONG 0 ; UNIBUS address of command ring
      0162 267 ;
      0030 0162 268 .WORD 48 ; Length of message
      0001 0164 269 .WORD 1 ; ID
      0001 0166 270 CMDPKT: .WORD 1
      00000196 0168 271 .BLKW 23 ; Full envelope
      0196 272 ;
      0000019A 0196 273 .BLKW 2
      000001CA 019A 274 RSPPKT: .BLKW 24
```

```
01CA 276      .SBTTL UDA50 Bootstrap driver Q10
01CA 277
01CA 278      :++
01CA 279
01CA 280      Inputs:
01CA 281
01CA 282      R3      - base address of adapter's register space
01CA 283      R5      - lbn for current piece of transfer
01CA 284      R6      - contains 0
01CA 285      R7      - address of the device's CSR
01CA 286      R8      - size of transfer in bytes
01CA 287      R9      - address of the RPB
01CA 288      R10     - starting address of transfer (byte offset in first
01CA 289                page 0Red with starting map register number)
01CA 290      R11     - LBN at start of transfer
01CA 291
01CA 292      FUNC(AP)- I/O operation (IOS_READBLK or IOS_WRITEBLK only)
01CA 293      SIZE(AP)- Size of transfer in bytes
01CA 294      MODE(AP)- Address interpretation mode (0 = physical, 1 = virtual)
01CA 295
01CA 296      Implicit inputs:
01CA 297
01CA 298      RPB$W_UNIT      - RPB field containing boot device unit number
01CA 299
01CA 300      Outputs:
01CA 301
01CA 302      R0 - status code
01CA 303
01CA 304      SSS_NORMAL      - successful transfer
01CA 305      SSS_NOSUCHDEV  - unsupported device
01CA 306      SSS_CTRLERR   - fatal controller error
01CA 307
01CA 308      R3 - must be preserved
01CA 309
01CA 310
01CA 311      --
01CA 312
00000010 01CA 313 FUNC = 16
00000014 01CA 314 MODE = 20
01CA 315
01CA 316 UD_DRIVER:      ; UDA50 device driver.
01CA 317
01CA 318
01CA 319      ; Translate the I/O function code into a device-dependent function
01CA 320      ; code for this disk.
01CA 321
01CA 322
A0 AF 21 90 01CA 323      MOVB      #MSCP$K_OP_READ, -      ; Assume read
01CE 324      CMDPKT+MSCP$B_OPCODE
20 10 AC D1 01CE 325      CMPL      FUNC(AP),#IOS_WRITEBLK ; Check for write function
04 12 01D2 326      BNEQ      20$      ; No, do read
96 AF 22 90 01D4 327      MOVB      #MSCP$K_OP_WRITE, -      ; Set write function code
01D8 328      CMDPKT+MSCP$B_OPCODE
A6 AF 55 D0 01D8 329 20$:      MOVL      R5,CMDPKT+MSCP$L_LBN      ; Set the logical block number
92 AF 58 D0 01DC 330      MOVL      R8,CMDPKT+MSCP$L_BYTE_CNT ; Set the byte count
92 AF 5A D0 01E0 331      MOVL      R10,CMDPKT+MSCP$B_BUFFER; Set the UNIBUS map register
54 02 A7 B0 01E4 332 10:      MOVW      UDA$A(R7),R4      ; Controller offline?
```

```

FF6F CF 8000 8F 2D 12 01E8 333 BNEQ ERROR1 ; Yep, error out
FF64 CF 8000 8F A8 01EA 334 BSW #OWN, CD+2 ; Set controller ownership
54 67 B0 01F1 335 BSW #OWN, RD+2 ; Ditto
02 A7 B0 01F8 336 MOVW UDAIP(R7),R4 ; Tell controller to read
16 12 01FF 337 MOVW UDASA(R7),R4 ; Any problems?
FF57 CF B5 0201 338 BNEQ ERROR1 ; Yes
FA 19 0205 339 30$: TSTW RD+2 ; Any response back?
9A AF B5 0207 340 BLSS 30$ ; No, spin until there is
OB 12 020A 341 TSTW RSPPKT+MSCP$W_STATUS ; Any drive errors?
FF49 CF 8000 8F A8 020C 342 BNEQ ERROR1 ; Yes
0213 343 BSW #OWN, RD+2 ; No, set controller ownship
0213 344 ;
0213 345 ; Transfer is complete. Return with success status code.
0213 346 ;
50 01 3C 0213 347 MOVZWL #SS$ _NORMAL,R0 ; SET COMPLETION CODE
05 0216 348 RSB ; AND RETURN
0217 349 ;
0217 350 ; Error occured during transfer. Return and retry.
0217 351 ;
0217 352 ;
0217 353 ERROR1:
50 0054 8F 3C 0217 354 MOVZWL #SS$ _CTRLERR,R0 ; Set failure status
05 021C 355 RSB ; Return to BOOTDRIVR
```

```
021D 357 .SBTTL UDA50 Bootstrap device disconnect
021D 358
021D 359 :++
021D 360 : This routine disconnect the boot device after a bugcheck dump.
021D 361 : It sends an AVAIL packet to the controller, in effect doing a
021D 362 : dismount of the system device. It is designed to be called
021D 363 : only from BUGCHECK immediately after the dump has finished.
021D 364 : It assumes virtual mapping turned on.
021D 365
021D 366 : Inputs:
021D 367
021D 368 : R9 --> RPB
021D 369
021D 370 : Outputs:
021D 371
021D 372 : R0 - status code
021D 373
021D 374 :--
021D 375
021D 376 UD_DISC:
021D 377 .ENABLE LSB
0090 021D 378 .WORD ^M<R4,R7>
021F 379
021F 380 MOVZBL #MSCP$K_OP_AVAIL,- : Make drive AVAILable
0221 381 CNDPKT+MSCP$B_OPCODE
57 FF4A CF 08 9A 0224 382 MOVL RPB$$_CSRVR(R9),R7 : Get correct address of device CSR
58 A9 D0 0228 383 BSBB IO : Send it out
BA 10 022A 384 RET
04 022B 385
0000022B 022B 386 UD_DRVSIZ=.-START
022B 387
022B 388 .END
```

PUBTDRIVR
Symbol table

- UDA50 BOOT DRIVER

L 5

15-SEP-1984 23:57:44 VAX/VMS Macro V04-00
4-SEP-1984 23:05:04 [BOOTS.SRC]PUBTDRIVR.MAR;1

Page 11
(5)

\$TABLE	= 00000000	R	02
BOOSAL_VECTOR	*****	X	03
BOOSGB_UMR_DP	*****	X	03
BQOSL_TENUSEC	= 0000003E		
BQOSL_UBDELAY	= 00000042		
BTDSK_UDA	= 00000011		
CD	0000015E	R	03
CMDINT	00000156	R	03
CMDPKT	00000166	R	03
DEVNAME	0000001A	R	03
DSKDRVNAME	00000000	R	03
ERROR	000000D6	R	03
ERROR1	00000217	R	03
FUNC	= 00000010		
GO	= 00000001		
INTTBL	0000014A	R	03
IO	000001E4	R	03
IOS_WRITEBLK	= 00000020		
MODE	= 00000014		
MSCPSB_BUFFER	= 00000010		
MSCPSB_OPCODE	= 00000008		
MSCPSK_OP_AVAIL	= 00000008		
MSCPSK_OP_ONLIN	= 00000009		
MSCPSK_OP_READ	= 00000021		
MSCPSK_OP_STCON	= 00000004		
MSCPSK_OP_WRITE	= 00000022		
MSCPSL_BYTE_CNT	= 0000000C		
MSCPSL_LBN	= 0000001C		
MSCPSL_MEDIA_ID	= 0000001C		
MSCPSS_MTYPE_D0	= 00000005		
MSCPSS_MTYPE_D1	= 00000005		
MSCPSV_MTYPE_D0	= 0000001B		
MSCPSV_MTYPE_D1	= 00000016		
MSCPSW_STATOS	= 0000000A		
OWN	= 00008000		
PR\$ MAPEN	= 00000038		
PRTDRVNAME	0000000D	R	03
PTE\$M_PFN	= 001FFFFFF		
RD	0000015A	R	03
RETRY	00000096	R	03
RING	0000015A	R	03
RPBSL_ADPPHY	= 0000005C		
RPBSL_ADPVIR	= 00000060		
RPBSL_BOOTR1	= 00000020		
RPBSL_BOOTR2	= 00000024		
RPBSL_CSRPHY	= 00000054		
RPBSL_CSRVIR	= 00000058		
RPBSL_IOVEC	= 00000034		
RPBSL_SVASPT	= 00000050		
RPBSW_ROUBVEC	= 0000001E		
RPBSW_UNIT	= 00000064		
RSPINT	00000158	R	03
RSPPKT	0000019A	R	03
S1	= 0000000B		
S4	= 0000000E		
SS\$_CTRLERR	= 00000054		
SS\$_NORMAL	= 00000001		

START	00000000	R	03
TIME	000000A0	R	03
UBASL_MAP	= 00000800		
UDAIP	= 00000000		
UDASA	= 00000002		
UD_DISC	0000021D	R	03
UD_DRIVER	000001CA	R	03
UD_DRVSI2	= 0000022B		
UD_INIT	0000001C	R	03
UMR	= 000001EE		
VASS_VPN	= 00000015		
VASV_VPN	= 00000009		
VALID	000000DF	R	03
VMBSB_SYSTEMID	00000024		
VMBSB_ARGBYTCNT	0000003C		
VMBSL_CI_HIPFN	00000030		
VMBSL_FLAGS	0000002C		
VMBSL_HI_PFN	00000010		
VMBSL_LO_PFN	0000000C		
VMBSQ_FILECACHE	00000004		
VMBSQ_NODENAME	00000034		
VMBSQ_PFNMAP	00000014		
VMBSQ_UCODE	0000001C		
VMBSV_LOAD_SCS	= 00000000		

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

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$AB\$\$	0000003C (60.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
BOOTDRIVR_4	00000028 (40.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
BOOTDRIVR_2	0000022B (555.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	32	00:00:00.09	00:00:00.31
Command processing	121	00:00:00.75	00:00:03.97
Pass 1	409	00:00:14.82	00:00:26.80
Symbol table sort	0	00:00:02.58	00:00:04.16
Pass 2	83	00:00:02.52	00:00:04.75
Symbol table output	11	00:00:00.10	00:00:00.10
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	660	00:00:20.89	00:00:40.12

The working set limit was 1500 pages.

83195 bytes (163 pages) of virtual memory were used to buffer the intermediate code.

There were 90 pages of symbol table space allocated to hold 1651 non-local and 9 local symbols.

388 source lines were read in Pass 1, producing 16 object records in Pass 2.

24 pages of virtual memory were used to define 21 macros.

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

Macro library name	Macros defined
_\$255\$DUA28:[BOOTS.OBJ]BOOTS.MLB;1	3
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	8
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	18

1804 GETS were required to define 18 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LISS:PUBTDRIVR/OBJ=OBJ\$:PUBTDRIVR MSRC\$:PUBTDRIVR/UPDATE=(ENH\$:PUBTDRIVR)+EXECMLS/LIB+LIB\$:BOOTS.MLB/LIB

0039 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY