

BBBBBBBBBBBB		00000000		00000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		00000000		00000000		TTTTTTTTTTTT		SSSSSSSSSS
BBBBBBBBBBBB		00000000		00000000		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		00000000		00000000		TTT	SSSSSSSSSS	
BBBBBBBBBBBB		00000000		00000000		TTT	SSSSSSSSSS	
BBBBBBBBBBBB		00000000		00000000		TTT	SSSSSSSSSS	

```

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
LLLLLLLLLLLL          IIIIII                      SSSSSSSS
LLLLLLLLLLLL          IIIIII                      SSSSSSSS

```


(2) 48
(3) 185
(4) 331

DECLARATIONS
RK06/7 Bootstrap driver code
ECC - PERFORM ECC ERROR CORRECTION

```
0000 1      .TITLE DMBTDRIVR - RK06/7 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         RK06/7 disks.
0000 35
0000 36     ENVIRONMENT:  IPL 31, kernel mode, code must be PIC
0000 37
0000 38     AUTHOR:  Steve Beckhardt,      CREATION DATE:  1-Nov-1979
0000 39     (Original author: Carol Peters)
0000 40
0000 41     MODIFIED BY:
0000 42
0000 43         02-02  CAS0001      C.A. Samuelson      30-Apr-1980
0000 44         Change interface to BOOTDRIVR for purge of UBA datapath
0000 45
0000 46     --
```



```
0000 48      .SBTTL  DECLARATIONS
0000 49      :
0000 50      : INCLUDE FILES:
0000 51      :
0000 52      :
0000 53      $BTDDDEF      : Boot device types
0000 54      $IODEF        : I/O function codes
0000 55      $PRDEF        : Processor registers
0000 56      $RPBDEF       : RPB offsets
0000 57      $SSDEF        : Status codes
0000 58      $UBADEF       : UBA definitions
0000 59      $SUBIDEF      : 11/750 UBA definitions
0000 60      :
0000 61      :
0000 62      : MACROS:
0000 63      :
0000 64      :
0000 65      :
0000 66      : EQUATED SYMBOLS:
0000 67      :
0000 68      :
0000 69      : RK611/RK06 CONTROLLER REGISTER OFFSETS
0000 70      :
0000 71      :
0000 72      $DEFINI RK
0000 73      :
0000 74 $DEF      RK_CS1      .BLKW  1      : CONTROL STATUS REGISTER 1
0002 75      _VIELD      RK_CS1,0,<-      : CONTROL STATUS REGISTER 1 FIELD DEFINITION
0002 76      <GO,,M>,-      : GO BIT
0002 77      <FCODE,4>,-      : FUNCTION CODE
0002 78      <DPPE,,M>,-      : DATA PATH PURGE ERROR
0002 79      <IE,,M>,-      : INTERRUPT ENABLE
0002 80      <RDY,,M>,-      : CONTROLLER READY
0002 81      <MEX,2>,-      : MEMORY EXTENSION BITS
0002 82      <CDT,,M>,-      : CONTROLLER DRIVE TYPE
0002 83      <CTO,,M>,-      : CONTROLLER TIME OUT
0002 84      <CFMT,,M>,-      : CONTROLLER FORMAT ERROR
0002 85      <SPAR,,M>,-      : SERIAL BUS PARITY ERROR
0002 86      <DI,,M>,-      : DRIVE INTERRUPT
0002 87      <CERR,,M>-      : CONTROLLER ERROR
0002 88      >
0002 89 $DEF      RK_WC      .BLKW  1      : WORD COUNT REGISTER
0004 90 $DEF      RK_BA      .BLKW  1      : BUFFER ADDRESS REGISTER
0006 91 $DEF      RK_DA      .BLKW  1      : DESIRED SECTOR/TRACK ADDRESS REGISTER
0008 92      _VIELD      RK_DA,0,<-      : DESIRED ADDRESS FIELD DEFINITIONS
0008 93      <SA,5>,-      : DESIRED SECTOR ADDRESS
0008 94      <3>-      : RESERVED BITS
0008 95      <TA,3>-      : DESIRED TRACK ADDRESS
0008 96      >
0008 97 $DEF      RK_CS2      .BLKW  1      : CONTROL STATUS REGISTER 2
000A 98      _VIELD      RK_CS2,0,<-      : CONTROL STATUS REGISTER 2 FIELD DEFINITION
000A 99      <DS,3>,-      : DRIVE SELECT
000A 100      <RLS,,M>,-      : RELEASE DRIVE
000A 101      <BAI,,M>,-      : BUFFER ADDRESS INCREMENT INHIBIT
000A 102      <SCLR,,M>,-      : SUBSYSTEM CLEAR
000A 103      <IR,,M>,-      : INPUT READY
000A 104      <OR,,M>-      : OUTPUT READY
```

```
000A 105 <UFE,,M>,- : UNIT FIELD ERROR
000A 106 <MDS,,M>,- : MULTIPLE DRIVE SELECT
000A 107 <PGE,,M>,- : PROGRAMMING ERROR
000A 108 <NEM,,M>,- : NONEXISTENT MEMORY
000A 109 <NED,,M>,- : NONEXISTENT DRIVE
000A 110 <UPE,,M>,- : UNIBUS PARITY ERROR
000A 111 <WCE,,M>,- : WRITE CHECK ERROR
000A 112 <DLT,,M>,- : DATA LATE ERROR
000A 113 >
000A 114 $DEF RK_DS .BLKW 1 : DRIVE STATUS REGISTER
000C 115 _VIELD RK_DS,0,<- : DRIVE STATUS REGISTER BIT DEFINITIONS
000C 116 <DRA,,M>,- : DRIVE AVAILABLE
000C 117 <,1>,- : RESERVED BIT
000C 118 <OFST,,M>,- : DRIVE OFFSET
000C 119 <ACLO,,M>,- : DRIVE AC LOW
000C 120 <DCLO,,M>,- : DRIVE DC LOW
000C 121 <DROT,,M>,- : DRIVE OFF TRACK
000C 122 <VV,,M>,- : VOLUME VALID
000C 123 <DRDY,,M>,- : DRIVE READY
000C 124 <DDT,,M>,- : DRIVE DRIVE TYPE
000C 125 <,2>,- : RESERVED BITS
000C 126 <WRL,,M>,- : DRIVE WRITE LOCKED
000C 127 <,1>,- : RESERVED BIT
000C 128 <PIP,,M>,- : POSITIONING IN PROGRESS
000C 129 <DSC,,M>,- : DRIVE STATUS CHANGE
000C 130 <SVAL,,M>,- : DRIVE STATUS VALID
000C 131 >
000C 132 $DEF RK_ER .BLKW 1 : ERROR REGISTER
000E 133 _VIELD RK_ER,0,<- : ERROR REGISTER BIT DEFINITIONS
000E 134 <ICF,,M>,- : ILLEGAL FUNCTION
000E 135 <SKI,,M>,- : SEEK INCOMPLETE
000E 136 <NXF,,M>,- : NONEXECUTABLE FUNCTION
000E 137 <DRPAR,,M>,- : DRIVE PARITY ERROR
000E 138 <FMTE,,M>,- : FORMAT ERROR
000E 139 <DTYE,,M>,- : DRIVE TYPE ERROR
000E 140 <ECH,,M>,- : ECC HARD ERROR
000E 141 <BSE,,M>,- : BAD SECTOR ERROR
000E 142 <HVRC,,M>,- : HEADER VRC ERROR
000E 143 <COE,,M>,- : CYLINDER OVERFLOW ERROR
000E 144 <IDAE,,M>,- : INVALID DISK ADDRESS ERROR
000E 145 <WLE,,M>,- : WRITE LOCK ERROR
000E 146 <DTE,,M>,- : DRIVE TIMING ERROR
000E 147 <OPI,,M>,- : OPERATION INCOMPLETE
000E 148 <UNS,,M>,- : DRIVE UNSAFE
000E 149 <DCK,,M>,- : DATA CHECK ERROR
000E 150 >
000E 151 $DEF RK_AS .BLKW 1 : ATTENTION SUMMARY/OFFSET REGISTER
0010 152 _VIELD RK_AS,0,<- : ATTENTION SUMMARY/OFFSET REGISTER FIELDS
0010 153 <OF,7>,- : DRIVE OFFSET
0010 154 <,1>,- : RESERVED BIT
0010 155 <ATTN,8,M>- : DRIVE ATTENTION SUMMARY
0010 156 >
0010 157 $DEF RK_DC .BLKW 1 : DESIRED CYLINDER ADDRESS
0012 158 $DEF RK_SPR .BLKW 1 : UNUSED REGISTER
0014 159 $DEF RK_DB .BLKW 1 : DATA BUFFER REGISTER
0016 160 $DEF RK_MR1 .BLKW 1 : MAINTENANCE REGISTER 1
0018 161 _VIELD RK_MR1,0,<<MS,3>> : MAINTENANCE REGISTER 1 FIELD DEFINITION
```



```
0018 162 $DEF RK_EC1 .BLKW 1 ;ECC POSITION REGISTER
001A 163 VIELD RK_EC1,0,<<EPS,13>> ; ECC POSITION FIELD
001A 164 $DEF RK_EC2 .BLKW 1 ;ECC PATTERN REGISTER
001C 165 VIELD RK_EC2,0,<<EPT,11>> ; ECC PATTERN FIELD
001C 166 $DEF RK_MR2 .BLKW 1 ;MAINTENANCE REGISTER 2
001E 167 $DEF RK_MR3 .BLKW 1 ;MAINTENANCE REGISTER 3
0020 168
0020 169 $DEFEND RK
0000 170
0000 171
0000 172 ;
0000 173 ; OWN STORAGE:
0000 174 ;
0000 175 ;
0000 176 ;
0000 177 ; Boot driver table entry
0000 178 ;
0000 179
0000 180 $BOOT_DRIVER DEVTPE = BTDSK_DM,- ; Device type (DM)
0000 181 SIZE = DM_DRVSIZ,- ; Driver size
0000 182 ADDR = DM_DRIVER,- ; Driver address
0000 183 DRVRNAME = DMNAME ; Driver name
```

```
0000 185      .SBTTL RK06/7 Bootstrap driver code
0000 186
0000 187      :++
0000 188      :
0000 189      Inputs:
0000 190      :
0000 191      R3      - base address of adapter's register space
0000 192      R5      - LBN FOR CURRENT PIECE OF TRANSFER
0000 193      R6      - contains 0
0000 194      R7      - address of the device's CSR
0000 195      R8      - SIZE OF TRANSFER IN BYTES
0000 196      R9      - address of the RPB
0000 197      R10     - starting address of transfer (byte offset in first
0000 198                page ORed with starting map register number)
0000 199      R11     - LBN at start of transfer
0000 200
0000 201      FUNC(AP)- I/O operation (IOS_READBLK or IOS_WRITEBLK only)
0000 202      BUF(AP)- Buffer address
0000 203      SIZE(AP)- Size of transfer in bytes
0000 204      MODE(AP)- Address interpretation mode (0 = physical, 1 = virtual)
0000 205
0000 206      Implicit inputs:
0000 207
0000 208      RPB$W_UNIT - RPB field containing boot device unit number
0000 209
0000 210      Outputs:
0000 211
0000 212      R0 - status code
0000 213
0000 214      SSS_NORMAL - successful transfer
0000 215      SSS_NOSUCHDEV - unsupported device
0000 216      SSS_CTRLERR - fatal controller error
0000 217
0000 218      R3 - must be preserved
0000 219
0000 220      This routine destroys R1, R2, R4, R5, R6. Within the
0000 221      routine, register usage is as follows:
0000 222
0000 223      R0      - mapping enabled flag
0000 224                device unit number
0000 225                status code
0000 226      R1      - device function code
0000 227      R2      - drive type according to device register
0000 228      R4      - used in logical to physical calculation
0000 229      R5      - used in logical to physical calculation
0000 230      R6      - used in logical to physical calculation
0000 231                transfer word count
0000 232
0000 233      :--
0000 234      BUF = 4
0000 235      SIZE = 8
0000 236      LBN = 12
0000 237      FUNC = 16
0000 238      MODE = 20
0000 239
0000 240      DM_DRIVER:
0000 241
```

; RK06/7 device driver.


```
0000 242 :  
0000 243 : Translate the I/O function code into a device-dependent function  
0000 244 : code for this disk.  
0000 245 :  
0000 246 :  
20 51 11 D0 0000 247 10$: MOVL #17,R1 : ASSUME READ  
08 10 AC D1 0003 248 CMPL FUNC(AP),#IOS_WRITEBLK : CHECK FOR WRITE FUNCTION  
03 12 0007 249 BNEQ 20$ : NO, DO READ  
51 13 D0 0009 250 MOVL #19,R1 : SET WRITE FUNCTION CODE  
000C 251 :  
000C 252 :  
000C 253 : Clear controller and drive status. Confirm that the drive exists and  
000C 254 : is ready for a transfer.  
000C 255 :  
000C 256 :  
50 64 A9 3C 000C 257 20$: MOVZWL RPB$W_UNIT(R9),R0 : GET UNIT NUMBER  
08 A7 20 B0 0010 258 MOVW #RK_CS2_M_SCLR,RK_CS2(R7) : CLEAR CONTROLLER AND ALL DRIVES  
0092 30 0014 259 BSBW READY : WAIT FOR CONTROLLER READY  
08 A7 50 B0 0017 260 MOVW R0,RK_CS2(R7) : SET DRIVE NUMBER  
001B 261 :  
67 05 B0 001B 262 30$: : Clear drive and find type.  
0088 30 001E 263 MOVW #5,RK_CS1(R7) : Clear drive.  
52 D4 0021 264 BSBW READY : Wait for controller ready.  
0A A7 0100 8F B3 0023 265 CLRL R2 : Assume RK06 drive.  
05 13 0029 266 BITW #RK_DS_M_DDT,RK_DS(R7) : Is drive RK07?  
52 0400 8F 3C 002B 267 BEQL 33$ : No. Branch.  
0030 268 MOVZWL #RK_CS1_M_CDT,R2 : Yes. Set drive code.  
0030 269 :  
08 A7 1000 8F B3 0030 270 33$: : Check for existence of drive.  
06 13 0036 271 BITW #RK_CS2_M_NED,RK_CS2(R7) : Does drive exist?  
50 0908 8F 3C 0038 272 BEQL 35$ : Yes. Branch.  
05 003D 273 MOVZWL #SS$_NOSUCHDEV,R0 : No. Exit with error  
003E 274 RSB :  
003E 275 :  
8000 8F B0 003E 276 35$: : Clear drive and acknowledge.  
08 A7 50 B0 0042 277 MOVW #RK_CS1_M_CERR,- : Clear controller error  
67 52 05 A9 0043 278 RK_CS1(R7) : status.  
005B 30 004B 279 MOVW R0,RK_CS2(R7) : Set unit number code again.  
0A A7 0080 8F B3 004E 280 BISW3 #5,R2,RK_CS1(R7) : Clear drive.  
E8 13 0054 281 BSBW READY : Wait for controller ready.  
67 52 03 A9 0056 282 BITW #RK_DS_M_DRDY,RK_DS(R7) : Is the drive ready?  
005A 283 BEQL 35$ : No. Clear drive again.  
005A 284 BISW3 #3,R2,RK_CS1(R7) : Acknowledge pack and set  
4D 10 005A 285 : volume valid.  
005C 286 BSBB READY : Wait for controller ready.  
005C 287 :  
005C 288 :  
005C 289 : Compute the cylinder, track, and sector addresses. Load device  
005C 290 : registers with transfer parameters and start transfer.  
005C 291 :  
005C 292 :  
55 54 55 00000042 56 D4 005C 293 CLRL R6 : Clear register for EDIV.  
10 A7 54 B0 005E 294 EDIV #22*3,R5,R4,R5 : COMPUTE DESIRED CYLINDER  
56 55 55 16 7B 0067 295 MOVW R4,RK_DC(R7) : AND SET IN DEVICE  
56 08 08 55 F0 006B 296 EDIV #22,R5,R5,R6 : CALCULATE DESIRED TRACK/SECTOR  
06 A7 56 B0 0070 297 INSV R5,#8,#8,R6 : MERGE TRACK AND SECTOR  
0075 298 MOVW R6,RK_DA(R7) : SET DESIRED TRACK SECTOR
```



```
04 A7 5A B0 0079 299      MOVW    R10,RK_BA(R7)      ; SET STARTING BUFFER ADDRESS
56 58 02 C7 007D 300      DIVL3   #2,R8,R6          ; COMPUTE WORD COUNT
02 A7 56 AE 0081 301      MNEGW   R6,RK_WC(R7)       ; SET NUMBER OF WORDS TO TRANSFER
67 52 51 A9 0085 302      BISW3   R1,R2,RK_CS1(R7)    ; Start disk function.
1E 10 0089 303      BSBB     READY                   ; WAIT FOR CONTROLLER READY
008B 304
008B 305
008B 306      ; Transfer is complete. See whether the transfer completed without
008B 307      ; error. If not, prepare input registers for the ECC correction routine
008B 308      ; and branch to that routine.
008B 309
008B 310
50 01 3C 008B 311      MOVZWL   #SS$,NORMAL,R0        ; ASSUME NORMAL COMPLETION
67 67 B5 008E 312      TSTW     RK_CS1(R7)            ; CHECK COMPOSITE ERROR
01 19 0090 313      BLSS     50$                      ; CONTINUE IF NO ERROR
05 0092 314      RSB
50 0C A7 B0 0093 315 50$: MOVW    RK_ER(R7),R0        ; GET ERROR STATUS
51 02 A7 32 0097 316      CVTWL   RK_WC(R7),R1        ; GET NEGATED COUNT REMAINING
51 51 02 C4 009B 317      MULL    #2,R1                ; CONVERT TO BYTES
55 18 A7 3C 009E 318      MOVZWL   RK_EC1(R7),R5       ; GET POSITION OF ERROR
56 1A A7 B0 00A2 319      MOVW     RK_EC2(R7),R6       ; GET PATTERN
0008 31 00A6 320      BRW       ECC                   ; AND ATTEMPT ECC CORRECTION
00A9 321
00A9 322
00A9 323      ; SUBROUTINE TO WAIT FOR CONTROLLER READY OR ERROR
00A9 324
00A9 325
00A9 326
67 8080 8F B3 00A9 327 READY: BITW    #^X8080,(R7)      ; CONTROLLER READY OR ERROR?
F9 13 00AE 328      BEQL     READY                   ; IF EQL NO
05 00B0 329      RSB
```



```
00B1 331 .SBTTL ECC - PERFORM ECC ERROR CORRECTION
00B1 332
00B1 333 :++
00B1 334 :
00B1 335 : Functional description:
00B1 336 :
00B1 337 : ATTEMPT ECC ERROR CORRECTION
00B1 338 :
00B1 339 : INPUTS:
00B1 340 :
00B1 341 : R0 - RK_ER/RP_ER1 ERROR STATUS REGISTER
00B1 342 : R1 - NEGATIVE-BYTE COUNT REMAINING
00B1 343 : R5 - ECC POSITION
00B1 344 : R6 - ECC PATTERN
00B1 345 : R8 - BYTE COUNT REMAINING AT START OF LAST TRANSFER
00B1 346 : R10 - starting address of transfer
00B1 347 : R11 - BLOCK NUMBER AT START OF TRANSFER
00B1 348 :
00B1 349 : Outputs:
00B1 350 :
00B1 351 : --
00B1 352 :
00B1 353 ECC: ; ATTEMPT ECC CORRECTION
00B1 354 :
00B1 355 :
00B1 356 : Don't attempt an ECC correction if any of the following conditions apply:
00B1 357 :
00B1 358 : the error was not a data check
00B1 359 : the error was a hard ECC error
00B1 360 : the transfer mode does not match the map-enabled position, i.e.,
00B1 361 : transfer is virtual; and mapping is not enabled, or v.v.
00B1 362 : no bytes have been transferred yet
00B1 363 :
00B1 364 :
5B 50 0F E1 00B1 365 BBC #RK_ER_V_DCK,R0,RETRY ; NOT DATA CHECK, RETRY
57 50 06 E0 00B5 366 BBS #RK_ER_V_ECH,R0,RETRY ; HARD ECC ERROR, RETRY
50 38 DB 00B9 367 MFPR #PR$ MAPEN,R0 ; GET MAP ENABLE STATE
14 AC 50 D1 00BC 368 CMPL R0,MODE(AP) ; SAME AS I/O MODE
51 4E 12 00C0 369 BNEQ RETRY ; NO, CANT DO SIMPLE ECC
51 58 C0 00C2 370 ADDL R8,R1 ; COMPUTE BYTES TRANSFERRED
49 13 00C5 371 BEQL RETRY ; NONE, RETRY
00C7 372 :
00C7 373 :
00C7 374 : Appears to be a correctable data check. Attempt the correction.
00C7 375 :
00C7 376 :
50 51 F7 8F 78 00C7 377 ASHL #-9,R1,R0 ; CONVERT TO PAGE COUNT
5B 50 C0 00CC 378 ADDL R0,R11 ; UPDATE BLOCK NUMBER
5A 51 C0 00CF 379 ADDL R1,R10 ; UPDATE BYTE ADDRESS
58 51 C2 00D2 380 SUBL R1,R8 ; DECREASE BYTES REMAINING
55 D7 00D5 381 DECL R5 ; MAKE POSITION 1 ORIGIN
50 0200 C8 9E 00D7 382 MOVAB 512(R8),R0 ; REMAINING BYTES AT START OF BAD SECTOR
52 08 AC 50 C3 00DC 383 SUBL3 R0,SIZE(AP),R2 ; BYTES TRANSFERRED AT START OF ERROR SECTOR
50 08 C4 00E1 384 MULL #8,R0 ; CONVERT BYTE COUNT TO BIT COUNT
50 55 C2 00E4 385 SUBL R5,R0 ; COMPUTE CORRECTION FIELD WIDTH
OD 19 15 00E7 386 BLEQ 20$ ; BR IF NO CORRECTION NEEDED
OD 50 D1 00E9 387 CMPL R0,#RK_EC1_S_EPS ; MINIMUM OF 13 AND BUFFER REMAINING
```

```

      03 15 00EC 388      BLEQ 10$      ; KEEP MINIMUM VALUE
51 04 BC42 50 0D D0 00EE 389      MOVL #RK_EC1,S,EPS,R0 ; LIMIT FIELD TO RK_EC1,S,EPS BITS
51 55 EF 00F1 390 10$: EXTZV R5,R0,@BUF(AP)[R2],R1 ; GET FIELD TO BE CORRECTED
51 56 CC 00F8 391      XORL R6,R1 ; APPLY CORRECTION CODE
04 BC42 50 55 51 F0 00FB 392      INSV R1,R5,R0,@BUF(AP)[R2] ; AND STORE IN BUFFER
      0102 393
      0102 394
      0102 395 ; If the transfer is not complete, branch back to retry it.
      0102 396
      0102 397
      58 D5 0102 398 20$: TSTL R8 ; CHECK FOR COUNT REMAINING
      06 15 0104 399      BLEQ 30$ ; NONE, EXIT
55 5B D0 0106 400      MOVL R11,R5 ; GET WORKING COPY OF LBN
      FEF4 31 0109 401      BRW DM_DRIVER ; CONTINUE TRANSFER
      010C 402
      010C 403 ;
      010C 404 ; Transfer is complete. Return with success status code.
      010C 405 ;
      010C 406
50 01 3C 010C 407 30$: MOVZWL #SS$_NORMAL,R0 ; SET COMPLETION CODE
      05 010F 408      RSB ; AND RETURN
      0110 409
      0110 410 ;
      0110 411 ; No ECC correction was possible. Return and retry.
      0110 412 ;
      0110 413
      0110 414 RETRY:
50 0054 8F 3C 0110 415      MOVZWL #SS$_CTRLERR,R0 ; Set failure status
      05 0115 416      RSB ; Return to BOOTDRIVR
58 45 2E 52 45 56 49 52 44 4D 44 00' 0116 417
      45 0122
      0C 0116
      0123 419
00000123 0123 420 DM_DRVSIZ=-DM_DRIVER
      0123 421
      0123 422      .END
```


DMBTDRIVR
Symbol table

- RK06/7 BOOT DRIVER

E 9

15-SEP-1984 23:51:15
4-SEP-1984 23:04:09

VAX/VMS Macro V04-00
[BOOTS.SRC]DMBTDRIVR.MAR;1

Page 10
(4)

\$TABLE	= 00000000	R	02
BTD\$K_DM	= 00000001		
BUF	= 00000004		
DMNAME	00000116	R	03
DM_DRIVER	00000000	R	03
DM_DRVSIZ	= 00000123		
ECC	000000B1	R	03
FUNC	= 00000010		
IOS_WRITEBLK	= 00000020		
LBN	= 0000000C		
MODE	= 00000014		
PR\$ MAPEN	= 00000038		
READY	000000A9	R	03
RETRY	00000110	R	03
RK_AS	0000000E		
RK_BA	00000004		
RK_CS1	00000000		
RK_CS1_M_CDT	= 00000400		
RK_CS1_M_CERR	= 00008000		
RK_CS2	00000008		
RK_CS2_M_NED	= 00001000		
RK_CS2_M_SCLR	= 00000020		
RK_DA	00000006		
RK_DB	00000014		
RK_DC	00000010		
RK_DS	0000000A		
RK_DS_M_DDT	= 00000100		
RK_DS_M_DRDY	= 00000080		
RK_ECT	00000018		
RK_EC1_S_EPS	= 0000000D		
RK_EC2	0000001A		
RK_ER	0000000C		
RK_ER_V_DCK	= 0000000F		
RK_ER_V_ECH	= 00000006		
RK_MRT	00000016		
RK_MR2	0000001C		
RK_MR3	0000001E		
RK_SPR	00000012		
RK_WC	00000002		
RP\$W_UNIT	= 00000064		
SIZ...	= 0000000B		
SIZE	= 00000008		
SS\$_CTRLERR	= 00000054		
SS\$_NORMAL	= 00000001		
SS\$_NOSUCHDEV	= 00000908		

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000020 (32.)	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	00000123 (291.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	36	00:00:00.10	00:00:00.37
Command processing	138	00:00:00.64	00:00:01.78
Pass 1	363	00:00:12.13	00:00:24.87
Symbol table sort	0	00:00:01.98	00:00:03.84
Pass 2	88	00:00:02.28	00:00:03.99
Symbol table output	7	00:00:00.08	00:00:00.10
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	636	00:00:17.24	00:00:34.99

The working set limit was 1650 pages.
68937 bytes (135 pages) of virtual memory were used to buffer the intermediate code.
There were 70 pages of symbol table space allocated to hold 1262 non-local and 9 local symbols.
422 source lines were read in Pass 1, producing 14 object records in Pass 2.
18 pages of virtual memory were used to define 16 macros.

! Macro library statistics !

Macro library name	Macros defined
-----	-----
-\$255\$DUA28:[BOOTS.OBJ]BOOTS.MLB;1	1
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	4
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	12

1246 GETS were required to define 12 macros.

There were no errors, warnings or information messages.

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

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY