

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	

```
MM      MM  BBBB BBBB  BBBB BBBB  TTTTTTTTTT  DDDDDDDD  RRRRRRRR  IIIIII  VV      VV  RRRRRRRR
MM      MM  BBBB BBBB  BBBB BBBB  TTTTTTTTTT  DDDDDDDD  RRRRRRRR  IIIIII  VV      VV  RRRRRRRR
MMM     MMM  BB      BB  BB      BB  TT          DD      DD  RR      RR  II      II  VV      VV  RR      RR
MMM     MMM  BB      BB  BB      BB  TT          DD      DD  RR      RR  II      II  VV      VV  RR      RR
MM      MM  BB      BB  BB      BB  TT          DD      DD  RR      RR  II      II  VV      VV  RR      RR
MM      MM  BBBB BBBB  BBBB BBBB  TT          DD      DD  RRRRRRRR  II      II  VV      VV  RRRRRRRR
MM      MM  BBBB BBBB  BBBB BBBB  TT          DD      DD  RRRRRRRR  II      II  VV      VV  RRRRRRRR
MM      MM  BB      BB  BB      BB  TT          DD      DD  RR      RR  II      II  VV      VV  RR      RR
MM      MM  BB      BB  BB      BB  TT          DD      DD  RR      RR  II      II  VV      VV  RR      RR
MM      MM  BB      BB  BB      BB  TT          DD      DD  RR      RR  II      II  VV      VV  RR      RR
MM      MM  BBBB BBBB  BBBB BBBB  TT          DDDDDDDD  RR      RR  IIIIII  VV      VV  RR      RR
MM      MM  BBBB BBBB  BBBB BBBB  TT          DDDDDDDD  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
LL      II      SS
LLLLLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLLLLL  IIIIII  SSSSSSSS
```

(2) 56
(3) 208
(4) 366
(5) 453

DECLARATIONS
RM03/5/80, RP04/5/6/7 Bootstrap driver code
ECC - PERFORM ECC ERROR CORRECTION
SSERROR - RM80 SKIP SECTOR ERROR ROUTINE

```

0000 1      .TITLE MBBTDRIVR - MASSBUS DISK BOOT DRIVER
0000 2      .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 *
0000 8 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 9 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 10 *  ALL RIGHTS RESERVED.
0000 11 *
0000 12 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 13 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 14 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 15 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 16 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 17 *  TRANSFERRED.
0000 18 *
0000 19 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 20 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 21 *  CORPORATION.
0000 22 *
0000 23 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 24 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
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
0000 34      MASSBUS disks. It supports the RP04/5/6 and RM03/5/80.
0000 35
0000 36 ENVIRONMENT:  IPL 31, kernel mode, code must be PIC
0000 37
0000 38 AUTHOR:  Steve Beckhardt,      CREATION DATE: 31-Oct-1979
0000 39      (Original authors: Carol Peters and Charlie Franks)
0000 40
0000 41 MODIFIED BY:
0000 42
0000 43      V02-004 KTA0041      Kerbey T. Altmann      10-Nov-1981
0000 44      Recode drive test routine to loop on Medium-On-Line
0000 45      bit thus allowing VMB to stall while drive spins up.
0000 46
0000 47      02-03  CAS0002      C.A. Samuelson      08-May-1980
0000 48      Fix driver size calculation so entire driver is copied
0000 49      into non-paged pool
0000 50
0000 51      02-02  CAS0001      C.A. Samuelson      30-Apr-1980
0000 52      Change interface to BOOTDRIVR for UBA purge datapath
0000 53
0000 54 --

```

```

0000 56          .SBTTL  DECLARATIONS
0000 57          :
0000 58          : INCLUDE FILES:
0000 59          :
0000 60          :
0000 61          $BTDDDEF          : Define boot device types
0000 62          $IODEF           : Define I/O function codes
0000 63          $MBADEF          : Define MASSBUS adapter registers
0000 64          $PRDEF           : Define processor registers
0000 65          $RPBDEF          : Define RPB offsets
0000 66          $SSDEF           : Define system status codes
0000 67          :
0000 68          :
0000 69          : MACROS:
0000 70          :
0000 71          :
0000 72          .MACRO  MBADEV,TYPE,SPC,SPT
0000 73          .BYTE   TYPE          : MASSBUS DEVICE CODE
0000 74          .WORD   SPC           : SECTORS PER CYLINDER
0000 75          .BYTE   SPT           : SECTORS PER TRACK
0000 76          .ENDM  MBADEV
0000 77          :
0000 78          :
0000 79          : EQUATED SYMBOLS:
0000 80          :
0000 81          :
0000 82          :
0000 83          : RP04/05/06 MASSBUS REGISTER OFFSETS
0000 84          :
0000 85          :
0000 86          $DEFINI RP
0000 87          :
0000 88  $DEF  RP_CS1          .BLKL  1          : DRIVE CONTROL REGISTER
0004 89          _VIELD  RP_CS1,0,<-          : DRIVE CONTROL REGISTER BIT DEFINITIONS
0004 90          <GO,,M>,-          : GO BIT
0004 91          <FCODE,5>-          : FUNCTION CODE
0004 92          >
0004 93  $DEF  RP_DS          .BLKL  1          : DRIVE STATUS REGISTER
0008 94          _VIELD  RP_DS,6,<-          : DRIVE STATUS REGISTER BIT DEFINITIONS
0008 95          <VV,,M>,-          : VOLUME VALID
0008 96          <DRY,,M>,-          : DRIVE READY
0008 97          <DPR,,M>,-          : DRIVE PRESENT
0008 98          <PGM,,M>,-          : PROGRAMMABLE
0008 99          <LST,,M>,-          : LAST SECTOR TRANSFERRED
0008 100         <WRL,,M>,-          : DRIVE WRITE LOCKED
0008 101         <MOL,,M>,-          : MEDIUM ONLINE
0008 102         <PIP,,M>,-          : POSITIONING IN PROGRESS
0008 103         <ERR,,M>,-          : COMPOSITE ERROR
0008 104         <ATA,,M>-          : ATTENTION ACTIVE
0008 105         >
0008 106  $DEF  RP_ER1          .BLKL  1          : ERROR REGISTER 1
000C 107         _VIELD  RP_ER1,0,<-          : ERROR REGISTER 1 BIT DEFINITIONS
000C 108         <ICF,,M>,-          : ILLEGAL FUNCTION
000C 109         <ILR,,M>,-          : ILLEGAL REGISTER
000C 110         <RMR,,M>,-          : REGISTER MODIFY REFUSED
000C 111         <PAR,,M>,-          : PARITY ERROR
000C 112         <FER,,M>-          : FORMAT ERROR

```

```

000C 113 <WCF,,M>,- : WRITE CLOCK FAIL
000C 114 <ECH,,M>,- : ECC HARD ERROR
000C 115 <HCE,,M>,- : HEADER COMPARE ERROR
000C 116 <HCRC,,M>,- : HEADER CRC ERROR
000C 117 <AOE,,M>,- : ADDRESS OVERFLOW ERROR
000C 118 <IAE,,M>,- : ILLEGAL ADDRESS ERROR
000C 119 <WLE,,M>,- : WRITE LOCK ERROR
000C 120 <DTE,,M>,- : DRIVE TIMING ERROR
000C 121 <OPI,,M>,- : OPERATION INCOMPLETE
000C 122 <UNS,,M>,- : DRIVE UNSAFE
000C 123 <DCK,,M>,- : DATA CHECK ERROR
000C 124 > :
000C 125 $DEF RP_MR .BLKL 1 : MAINTENANCE REGISTER
0010 126 $DEF RP_AS .BLKL 1 : ATTENTION SUMMARY REGISTER
0014 127 $DEF RP_DA .BLKL 1 : DESIRED SECTOR/TRACK ADDRESS REGISTER
0018 128 -VIELD RP_DA,0,<- : DESIRED ADDRESS FIELD DEFINITIONS
0018 129 <SA,5>,- : DESIRED SECTOR ADDRESS
0018 130 <3>,- : RESERVED BITS
0018 131 <TA,5>,- : DESIRED TRACK ADDRESS
0018 132 > :
0018 133 $DEF RP_DT .BLKL 1 : DRIVE TYPE REGISTER
001C 134 -VIELD RP_DT,0,<- : DRIVE TYPE REGISTER FIELD DEFINITIONS
001C 135 <DTN,9>,- : DRIVE TYPE NUMBER
001C 136 <2>,- : RESERVED BITS
001C 137 <DRQ,,M>,- : DRIVE REQUEST REQUIRED
001C 138 > :
001C 139 $DEF RP_LA .BLKL 1 : LOOKAHEAD REGISTER
0020 140 $DEF RP_ER2 .BLKL 1 : ERROR REGISTER 2
0024 141 $DEF RP_OF .BLKL 1 : OFFSET REGISTER
0028 142 -VIELD RP_OF,0,<- : OFFSET REGISTER BIT DEFINITIONS
0028 143 <OFF,8>,- : OFFSET VALUE
0028 144 <DCK,,M>,- : DATA CHECK IN PROGRESS (SOFTWARE)
0028 145 <1>,- : RESERVED BIT
0028 146 <HCI,,M>,- : HEADER COMPARE INHIBIT
0028 147 <ECI,,M>,- : ECC INHIBIT
0028 148 <FMT,,M>,- : 16-BIT FORMAT
0028 149 > :
0028 150 $DEF RP_DC .BLKL 1 : DESIRED CYLINDER ADDRESS
002C 151 $DEF RP_CC .BLKL 1 : CURRENT CYLINDER ADDRESS
0030 152 $DEF RP_SN .BLKL 1 : DRIVE SERIAL NUMBER
0034 153 $DEF RP_ER3 .BLKL 1 : ERROR REGISTER 3
0038 154 -VIELD RP_ER3,14,<- : ERROR REGISTER 3 BIT DEFINITIONS
0038 155 <SKI,,M>,- : SEEK INCOMPLETE
0038 156 > :
0038 157 $DEF RP_EC1 .BLKL 1 : ECC POSITION REGISTER
003C 158 VIELD RP_EC1,0,<<POS,13>> : ECC POSITION FIELD
003C 159 $DEF RP_EC2 .BLKL 1 : ECC PATTERN REGISTER
0040 160 -VIELD RP_EC2,0,<<PAT,11>> : ECC PATTERN FIELD
0040 161 $DEFEND RP
0000 162
0000 163
0000 164 :
0000 165 : RM80 ADDITIONAL MASSBUS REGISTER BIT DEFINITIONS
0000 166 :
0000 167 :
00000200 0000 168 RM_OF_M_SSEI = ^X200 : SKIP SECTOR ERROR INHIBIT (RP_ER3)
00000020 0000 169 RM_ER2_M_SSE = ^X20 : SKIP SECTOR ERROR (RP_OF)

```

```

00000005 0000 170 RM_ER2_V_SSE = 5 ; ...
          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 DEVTYP = BTDSK_MB,- ; Device type (MASSBUS)
          0000 181 SIZE = MBA_DRVSTZ,- ; Driver size
          0000 182 ADDR = DRIVERNAME,- ; Driver address
          0000 183 ENTRY = MBA_DISK_DRIVER,- ; Entry point
          0000 184 DRIVERNAME = DRIVERNAME ; Driver name
          0000 185
          0000 186
          0000 187 DRIVERNAME: ; Boot device driver name
          0000 188 .ASCII /DRDRIVER.EXE/ ; Assume DRDRIVER
          0000 189
          0000 190
          0000 191 :
          0000 192 : GEOMETRY TABLE FOR MASSBUS DISKS
          0000 193 :
          0000 194 :
          0000 195 .SHOW BINARY ; Show MACRO expansions.
          0000 196
          0000 197 MBAGEOM:
          0000 198 MBADEV <^X10>,<22*19>,<22> ; RP04
          10 0000 .BYTE ^X10 ; MASSBUS DEVICE CODE
          01A2 000E .WORD 22*19 ; SECTORS PER CYLINDER
          16 0010 .BYTE 22 ; SECTORS PER TRACK
          0011 199 MBADEV <^X11>,<22*19>,<22> ; RP05
          11 0011 .BYTE ^X11 ; MASSBUS DEVICE CODE
          01A2 0012 .WORD 22*19 ; SECTORS PER CYLINDER
          16 0014 .BYTE 22 ; SECTORS PER TRACK
          0015 200 MBADEV <^X12>,<22*19>,<22> ; RP06
          12 0015 .BYTE ^X12 ; MASSBUS DEVICE CODE
          01A2 0016 .WORD 22*19 ; SECTORS PER CYLINDER
          16 0018 .BYTE 22 ; SECTORS PER TRACK
          0019 201 MBADEV <^X14>,<5*32>,<32> ; RM03
          14 0019 .BYTE ^X14 ; MASSBUS DEVICE CODE
          00A0 001A .WORD 5*32 ; SECTORS PER CYLINDER
          20 001C .BYTE 32 ; SECTORS PER TRACK
          001D 202 MBADEV <^X16>,<14*31>,<31> ; RM80
          16 001D .BYTE ^X16 ; MASSBUS DEVICE CODE
          01B2 001E .WORD 14*31 ; SECTORS PER CYLINDER
          1F 0020 .BYTE 31 ; SECTORS PER TRACK
          0021 203 MBADEV <^X17>,<19*32>,<32> ; RM05
          17 0021 .BYTE ^X17 ; MASSBUS DEVICE CODE
          0260 0022 .WORD 19*32 ; SECTORS PER CYLINDER
          20 0024 .BYTE 32 ; SECTORS PER TRACK
          0025 204 MBADEV <^X22>,<50*32>,<50> ; RP07 (W/O HEAD PER TRACK)
          22 0025 .BYTE ^X22 ; MASSBUS DEVICE CODE
          0640 0026 .WORD 50*32 ; SECTORS PER CYLINDER

```

MBBTDRIVR
V04-000

- MASSBUS DISK BOOT DRIVER
DECLARATIONS

B 2

15-SEP-1984 23:56:04 VAX/VMS Macro V04-00
4-SEP-1984 23:04:51 [BOOTS.SRC]MBBTDRIVR.MAR;1

Page 5
(2)

32	0028		.BYTE	50	:	SECTORS PER TRACK
	0029	205	MBADEV	<^X21>,<50*32>,<50	:	RP07 (WITH HEAD PER TRACK OPTION)
21	0029		.BYTE	^X21	:	MASSBUS DEVICE CODE
0640	002A		.WORD	50*32	:	SECTORS PER CYLINDER
32	002C		.BYTE	50	:	SECTORS PER TRACK
00	002D	206	.BYTE	0	:	TERMINATE LIST

```

002E 208 .SBTTL RM03/5/80, RP04/5/6/7 Bootstrap driver code
002E 209
002E 210 :++
002E 211 :
002E 212 : Inputs:
002E 213 :
002E 214 : R3 - base address of adapter's register space
002E 215 : R5 - LBN for current piece of transfer
002E 216 : R6 - contains 0
002E 217 : R8 - size of transfer in bytes
002E 218 : R9 - address of the RPB
002E 219 : R10 - starting address of transfer (byte offset in first
002E 220 : page ORed with starting map register number)
002E 221 : R11 - Block number at start of transfer
002E 222 :
002E 223 : FUNC(AP)- I/O operation (IO$ READLBLK or IO$ WRITELBLK only)
002E 224 : MODE(AP)- Address interpretation mode (0 = physical, 1 = virtual)
002E 225 : SIZE(AP)- Size of buffer in bytes
002E 226 : BUF(AP) - Address of buffer
002E 227 :
002E 228 : Implicit inputs:
002E 229 :
002E 230 : RPB$W_UNIT - RPB field containing boot device unit number
002E 231 :
002E 232 : MBAGEOM - a table describe the geometries of the
002E 233 : MASSBUS disks
002E 234 :
002E 235 : Outputs:
002E 236 :
002E 237 : R0 - status code
002E 238 :
002E 239 : SSS_NORMAL - successful transfer
002E 240 : SSS_NOSUCHDEV - unsupported device
002E 241 : SSS_CTRLERR - fatal controller error on transfer
002E 242 :
002E 243 : This routine destroys R1, R5, R6, and R7. Within the routine,
002E 244 : register usage is as follows:
002E 245 :
002E 246 : R0 - drive type from the geometry table
002E 247 : MASSBUS device function code
002E 248 : status code
002E 249 : R1 - address of an entry in the MBA geometry table
002E 250 : R5 - used in logical to physical calculation
002E 251 : R6 - drive type according to device register
002E 252 : used in logical to physical calculation
002E 253 : R7 - address of the 1st MBA external register for
002E 254 : this unit on the MBA
002E 255 :
002E 256 : --
002E 257 :
00000004 002E 258 BUF = 4
00000008 002E 259 SIZE = 8
0000000C 002E 260 LBN = 12
00000010 002E 261 FUNC = 16
00000014 002E 262 MODE = 20
002E 263
002E 264 MBA_DISK_DRIVER:

```

; RM03/5/80, RP04/5/6/7 Boot driver.

```

      57 57 64 A9 3C 002E 265      MOVZWL RPB$W_UNIT(R9),R7      : GET UNIT NUMBER OF BOOT DEVICE
      57 57 07 78 0032 266      ASHL #7,R7,R7      : ALIGN TO UNIT NUMBER FIELD
      57 0400 C347 9E 0036 267      MOVAB MBASL_ERB(R3)[R7],R7      : FORM ADDRESS OF DRIVE REGISTERS
      003C 268      :
      003C 269      : NOTE: The following test works because the MBA OR's the upper 16 bits of
      003C 270      : the MBA register SR and the lower 16 bits of the driver status
      003C 271      : register to make up the 32 bits sensed on a longword access.
      003C 272      :
      04 A7 00040000 8F D3 003C 273      BITL #MBAS$M_SR_NED,RP_DS(R7) : DOES DRIVE EXIST?
      03 13 0044 274      BEQL 10$      : YES, CONTINUE
      00A3 31 0046 275      BRW NOSUCHDEV      : NO, ERROR
      0049 276      :
      0049 277      :
      0049 278      : Check to see that Medium-On-Line is set. If not, spin until it is.
      0049 279      :
      0049 280      :
      04 A7 00001000 8F D3 0049 281 10$: BITL #RP_DS_M_MOL,RP_DS(R7) : IS MEDIUM ON LINE?
      F6 13 0051 282      BEQL 10$      : NO, SPIN UNTIL IT IS
      67 13 D0 0053 283      MOVL #<9*2>+1,RP_CS1(R7) : SET VOLUME VALID
      56 18 A7 D0 0056 284 15$: MOVL RP_DT(R7),R6 : GET DRIVE TYPE
      12 56 91 005A 285      CMPB R6,#^X12 : IS IT RP04/5/6?
      06 1A 005D 286      BGTRU 17$ : NO
      9C AF 4244 8F B0 005F 287      MOVW #^A/DB/,DRIVERNAME+1 : YES, CHANGE DRIVERNAME
      51 A2 AF 9E 0065 288 17$: MOVAB MBAGEOM-3,R1 : POINT TO START OF GEOMETRY TABLE
      51 03 C0 0069 289 20$: ADDL #3,R1 : ADVANCE TO NEXT TYPE CODE
      50 81 90 006C 290      MOVB (R1)+,R0 : GET TYPE CODE
      03 12 006F 291      BNEQ 30$ : IF NEQ, LEGAL TYPE FOUND - CONTINUE
      0078 31 0071 292      BRW NOSUCHDEV : TYPE NOT FOUND, UNSUPPORTED DEVICE
      50 56 91 0074 293 30$: CMPB R6,R0 : CHECK FOR MATCH
      F0 12 0077 294      BNEQ 20$ : NO, TRY ANOTHER
      24 A7 1000 8F 3C 0079 295      MOVZWL #RP_OF_M_FMT,RP_OF(R7) : SELECT 16 BIT FORMAT
      007F 296      :
      007F 297      :
      007F 298      : Convert LBN to physical device address. Start the transfer.
      007F 299      :
      007F 300      :
      56 D4 007F 301      CLRL R6 : CLEAR HIGH ORDER BITS OF DISK ADDRESS
      50 81 3C 0081 302      MOVZWL (R1)+,R0 : GET SECTORS PER CYLINDER
      55 50 7B 0084 303      EDIV R0,R5,RP_DC(R7),R5 : CALCULATE AND SET CYLINDER ADDRESS
      50 81 9A 008A 304      MOVZBL (R1)+,R0 : GET SECTORS PER TRACK
      56 55 55 50 7B 008D 305      EDIV R0,R5,R5,R6 : CALCULATE SECTOR AND TRACK ADDRESS
      0092 306 MCOMMON:
      56 08 08 55 F0 0092 307      INSV R5,#8,#8,R6 : MERGE SECTOR AND TRACK ADDRESS
      14 A7 56 D0 0097 308      MOVL R6,RP_DA(R7) : SET DESIRED SECTOR AND TRACK ADDRESS
      10 A3 58 CE 009B 309      MNEGL R8,MBAS$B_BCR(R3) : SET TRANSFER BYTE COUNT
      0C A3 5A D0 009F 310      MOVL R10,MBAS$VAR(R3) : AND SET AS VIRTUAL STARTING ADDRESS
      50 39 D0 00A3 311      MOVL #<28*2>+1,R0 : ASSUME READ
      20 10 AC D1 00A6 312      CMPL FUNC(AP),#10$_WRITEBLK : IS FUNCTION WRITE?
      03 12 00AA 313      BNEQ 10$ : NO, THEN READ
      50 31 D0 00AC 314      MOVL #<24*2>+1,R0 : SET WRITE FUNCTION
      67 50 D0 00AF 315 10$: MOVL R0,RP_CS1(R7) : START TRANSFER
      00B2 316      :
      00B2 317      :
      00B2 318      : Loop until the transfer completes.
      00B2 319      :
      00B2 320      :
      08 A3 D5 00B2 321 20$: TSTL MBAS$SR(R3) : DATA TRANSFER BUSY?
```

```

FB 19 00B5 322          BLSS 20$          ; IF LSS YES
      00B7 323
      00B7 324 :
      00B7 325 : If the boot device is an RM80, check for a skip sector error.
      00B7 326 :
      00B7 327
56 18 A7 D0 00B7 328          MOVL RP_DT(R7),R6          ; GET DEVICE TYPE
56 16 91 00BB 329          CMPB #^X16,R6          ; IS BOOT DEVICE = RM80?
      09 12 00BE 330          BNEQ 30$          ; IF NEQ, NO
34 A7 20 D3 00C0 331          BITL #RM_ER2_M_SSE,RP_ER3(R7) ; SKIP SECTOR ERROR?
      03 13 00C4 332          BEQL 30$          ; IF EQL, NO
      00BE 31 00C6 333          BRW SSERROR          ; SKIP SECTOR ERROR
      00C9 334
      00C9 335 :
      00C9 336 : See if the transfer was a success.
      00C9 337 :
      00C9 338
08 A3 50 01 D0 00C9 339 30$: MOVL #SS$ NORMAL,R0          ; ASSUME NORMAL
      000E5FFF 8F D3 00CC 340          BITL #MBA$M_ERROR,MBA$L_SR(R3) ; ANY CONTROLLER ERRORS?
      1B 13 00D4 341          BEQL RETURN          ; IF EQL NO
      00D6 342
      00D6 343 :
      00D6 344 : An error occurred in the transfer. Prepare 4 input registers and call
      00D6 345 : the ECC correction routine.
      00D6 346 :
      00D6 347
50 08 A7 D0 00D6 348          MOVL RP_ER1(R7),R0          ; GET ERROR STATUS
51 10 A3 D0 00DA 349          MOVL MBA$L_BCR(R3),R1          ; AND BYTE COUNT REMAINING
51 51 32 00DE 350          CVTWL R1,R1          ; EXTEND SIGN
55 38 A7 D0 00E1 351          MOVL RP_EC1(R7),R5          ; GET ECC POSITION
56 3C A7 D0 00E5 352          MOVL RP_EC2(R7),R6          ; AND PATTERN
      0006 31 00E9 353          BRW ECC          ; ATTEMPT ECC CORRECTION
      00EC 354
      00EC 355 :
      00EC 356 : The specified boot device unit does not exist, or it is of an
      00EC 357 : unsupported drive type.
      00EC 358 :
      00EC 359
50 0908 8F 3C 00EC 360 NOSUCHDEV:          ; UNSUPPORTED DEVICE
      00F1 361          MOVZWL #SS$_NOSUCHDEV,R0          ; SET ERROR CODE
      00F1 362
      00F1 363 RETURN:
      05 00F1 364          RSB          ; AND RETURN

```

```

00F2 366 .SBTTL ECC - PERFORM ECC ERROR CORRECTION
00F2 367
00F2 368 :++
00F2 369
00F2 370 : Functional description:
00F2 371
00F2 372 : ATTEMPT ECC ERROR CORRECTION
00F2 373
00F2 374 : INPUTS:
00F2 375
00F2 376 : R0 - RP_ER1 ERROR STATUS REGISTER
00F2 377 : R1 - NEGATIVE BYTE COUNT REMAINING
00F2 378 : R5 - ECC POSITION
00F2 379 : R6 - ECC PATTERN
00F2 380 : R8 - BYTE COUNT REMAINING AT START OF LAST TRANSFER
00F2 381 : R10 - starting address of transfer
00F2 382 : R11 - BLOCK NUMBER AT START OF TRANSFER
00F2 383
00F2 384 : Outputs:
00F2 385
00F2 386 :--
00F2 387
00F2 388 ECC: ; ATTEMPT ECC CORRECTION
00F2 389
00F2 390 :
00F2 391 : Don't attempt an ECC correction if any of the following conditions apply:
00F2 392 :
00F2 393 : the error was not a data check
00F2 394 : the error was a hard ECC error
00F2 395 : the transfer mode does not match the map-enabled position, i.e.,
00F2 396 : transfer is virtual, and mapping is not enabled, or v.v.
00F2 397 : no bytes have been transferred yet
00F2 398 :
00F2 399 :
5B 50 0F E1 00F2 400 BBC #RP_ER1_V_DCK,R0,RETRY ; NOT DATA CHECK, RETRY
57 50 06 E0 00F6 401 BBS #RP_ER1_V_ECH,R0,RETRY ; HARD ECC ERROR, RETRY
50 38 DB 00FA 402 MFPR #PR$ MAPEN,R0 ; GET MAP ENABLE STATE
14 AC 50 D1 00FD 403 CMPL R0,MODE(AP) ; SAME AS I/O MODE
51 4E 12 0101 404 BNEQ RETRY ; NO, CANT DO SIMPLE ECC
51 58 C0 0103 405 ADDL R8,R1 ; COMPUTE BYTES TRANSFERRED
49 13 0106 406 BEQL RETRY ; NONE, RETRY
0108 407
0108 408 :
0108 409 : Appears to be a correctable data check. Attempt the correction.
0108 410 :
0108 411 :
50 51 F7 8F 78 0108 412 ASHL #-9,R1,R0 ; CONVERT TO PAGE COUNT
5B 50 C0 010D 413 ADDL R0,R11 ; UPDATE BLOCK NUMBER
5A 51 C0 0110 414 ADDL R1,R10 ; UPDATE BYTE ADDRESS
58 51 C2 0113 415 SUBL R1,R8 ; DECREASE BYTES REMAINING
55 D7 0116 416 DECL R5 ; MAKE POSITION 1 ORIGIN
50 0200 C8 9E 0118 417 MOVAB 512(R8),R0 ; REMAINING BYTES AT START OF BAD SECTOR
52 08 AC 50 C3 011D 418 SUBL3 R0,SIZE(AP),R2 ; BYTES TRANSFERRED AT START OF ERROR SECTOR
50 08 C4 0122 419 MULL #8,R0 ; CONVERT BYTE COUNT TO BIT COUNT
50 55 C2 0125 420 SUBL R5,R0 ; COMPUTE CORRECTION FIELD WIDTH
19 15 0128 421 BLEQ 20$ ; BR IF NO CORRECTION NEEDED
0D 50 D1 012A 422 CMPL R0,#RP_EC1_S_POS ; MINIMUM OF 13 AND BUFFER REMAINING

```

```

51 04 BC42 50 03 15 012D 423 BLEQ 10$ ; KEEP MINIMUM VALUE
50 0D D0 012F 424 MOVL #RP_EC1_S_POS,R0 ; LIMIT FIELD TO RP_EC1_S_POS BITS
50 55 EF 0132 425 10$: EXTZV R5,R0,@BUF(AP)[R2],R1 ; GET FIELD TO BE CORRECTED
51 56 CC 0139 426 XORL R6,R1 ; APPLY CORRECTION CODE
04 BC42 50 55 51 F0 013C 427 INSV R1,R5,R0,@BUF(AP)[R2] ; AND STORE IN BUFFER
0143 428
0143 429
0143 430 ; If the transfer is not complete, branch back to retry it.
0143 431
0143 432
58 D5 0143 433 20$: TSTL R8 ; CHECK FOR COUNT REMAINING
06 15 0145 434 BLEQ 30$ ; NONE, EXIT
55 5B D0 0147 435 MOVL R11,R5 ; GET WORKING COPY OF LBN
FEE1 31 014A 436 BRW MBA_DISK_DRIVER ; CONTINUE TRANSFER
014D 437
014D 438
014D 439 ; Transfer is complete. Return with success status code.
014D 440
014D 441
50 01 3C 014D 442 30$: MOVZWL #SS$_NORMAL,R0 ; SET COMPLETION CODE
05 05 0150 443 RSB ; AND RETURN
0151 444
0151 445 ;
0151 446 ; No ECC correction was possible. Return and retry.
0151 447 ;
0151 448
50 0054 8F 3C 0151 449 RETRY: MOVZWL #SS$_CTRLERR,R0 ; SET FAILURE
05 05 0156 450 RSB ; RETURN
0156 451

```

```

0157 453 .SBTTL SSERROR - RM80 SKIP SECTOR ERROR ROUTINE
0157 454
0157 455 :++
0157 456 : FUNCTIONAL DESCRIPTION:
0157 457 :
0157 458 : SSERROR is entered if an RM80 boot device encounters a skip
0157 459 : sector error (SSE). SSE's are inhibited for the remainder of
0157 460 : the current track, transfer parameters are updated to the last
0157 461 : successful sector, and the sector address is incremented to skip
0157 462 : the error sector. The transfer is then allowed to continue.
0157 463 :
0157 464 : Note: The first bad spot on each track of an RM80 is marked at
0157 465 : manufacturing time as a "skipped sector", and causes an
0157 466 : SSE. Data for the remaining sectors on that track are
0157 467 : incremented by one physical sector, the last (31st) sector
0157 468 : mapping into a 32nd sector reserved for this use.
0157 469 :
0157 470 : INPUTS:
0157 471 :
0157 472 : R3 - base address of the adapter's register space
0157 473 : R7 - address of the first external MBA register for this unit
0157 474 : R8 - byte count remaining at start of last transfer
0157 475 : R10 - starting address of transfer
0157 476 : R11 - updated block number (LBN)
0157 477 :
0157 478 : OUTPUTS:
0157 479 :
0157 480 : Skip sector error is cleared in the error register (RP_ER3).
0157 481 : Skip sector error inhibit is set in the offset register (RP_OF).
0157 482 : R5 - current track address
0157 483 : R6 - sector address incremented past skipped sector
0157 484 : R8 - updated byte count remaining to transfer
0157 485 : R10 - updated starting address
0157 486 : R11 - updated block number (LBN)
0157 487 :
0157 488 : --
0157 489 :
0157 490 SSERROR:
0157 491 BISL #RM_OF_M_SSE1,RP_OF(R7) ; RM80 SKIP SECTOR ERROR ROUTINE
0157 492 MOVL #<4*2>+1,RP_CS1(R7) ; Set skip sector error inhibit
0157 493 : Clear drive errors
0157 494 :
0157 495 : Update transfer parameters
0157 496 :
0157 497 :
0157 498 MOVL MBA$L_BCR(R3),R1 ; Get negative bytes remaining
0157 499 CVTWL R1,R1 ; Extend sign
0157 500 BNEQ 10$ ; If NEQ, partial transfer
0157 501 MOVL #-1,R1 ; Force a partial transfer
0157 502 ADDL R8,R1 ; Calculate bytes transferred
0157 503 BICW #X1FF,R1 ; Truncate start of sector
0157 504 ASHL #-9,R1,R0 ; Convert to page count
0157 505 ADDL R0,R11 ; Update LBN
0157 506 ADDL R1,R10 ; Update starting address
0157 507 SUBL R1,R8 ; Decrease bytes remaining
0157 508
0157 509 :

24 A7 00000200 8F C8 0157 491
67 09 D0 0157 492
0162 493
0162 494
0162 495
0162 496
0162 497
51 10 A3 D0 0162 498
51 51 32 0166 499
51 07 12 0169 500
51 FFFFFFFF 8F D0 016B 501
51 51 58 C0 0172 502 10$:
51 01FF 8F AA 0175 503
50 51 F7 8F 78 017A 504
5B 50 C0 017F 505
5A 51 C0 0182 506
58 51 C2 0185 507
0188 508
0188 509 :

```

```

0188 510 ; Convert updated LBN to physical device address.
0188 511 ; Increment past skipped sector and re-start.
0188 512 ;
0188 513 ;
55 5B D0 0188 514 MOVL R11,R5 ; Get working copy of LBN
56 56 D4 0188 515 CLRL R6 ; Clear high order bits of disk address
55 01B2 8F 3C 0188 516 MOVZWL #<14*31>,R0 ; Get sectors per cylinder
55 50 7B 018D 517 EDIV R0,R5,RP_DC(R7),R5 ; Calculate and set cylinder address
50 1F 9A 0192 518 MOVZBL #31,R0 ; Get sectors per track
56 55 55 50 7B 0198 519 EDIV R0,R5,R5,R6 ; Calculate R5=track R6=sector
56 56 96 01A0 520 INCB R6 ; Increment sector address
FEED 31 01A2 521 ;
000001A5 01A2 522 BRW MBCOMMON ; Restart the function
01A5 523
01A5 524 MBA_DRVSIZ=-DRIVERNAME
01A5 525
01A5 526 .END

```

MBBTDRIVR
Symbol table

- MASSBUS DISK BOOT DRIVER

J 2

15-SEP-1984 23:56:04
4-SEP-1984 23:04:51

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

Page 13
(5)

\$TABLE	= 00000000	R	02
BTDSK_MB	= 00000000		
BUF	= 00000004		
DRIVERNAME	00000000	R	03
ECC	000000F2	R	03
FUNC	= 00000010		
IOS_WRITELBLK	= 00000020		
LBN	= 0000000C		
MBASL_BCR	= 00000010		
MBASL_ERB	= 00000400		
MBASL_SR	= 00000008		
MBASL_VAR	= 0000000C		
MBASM_ERROR	= 000E5FFF		
MBASM_SR_NED	= 00040000		
MBAGEOM	0000000D	R	03
MBA_DISK_DRIVER	0000002E	R	03
MBA_DRVSIZ	= 000001A5		
MBCOMMON	00000092	R	03
MODE	= 00000014		
NOSUCHDEV	000000EC	R	03
PR\$ MAPEN	= 00000038		
RETRY	00000151	R	03
RETURN	000000F1	R	03
RM_ER2_M_SSE	= 00000020		
RM_ER2_V_SSE	= 00000005		
RM_OF_M_SSEI	= 00000200		
RPBSW_UNIT	= 00000064		
RP_AS	00000010		
RP_CC	0000002C		
RP_CS1	00000000		
RP_DA	00000014		
RP_DC	00000026		
RP_DS	00000004		
RP_DS_M_MOL	= 00001000		
RP_DT	00000018		
RP_EC1	00000038		
RP_EC1_S_POS	= 0000000D		
RP_EC2	0000003C		
RP_ER1	00000008		
RP_ER1_V_DCK	= 0000000F		
RP_ER1_V_ECH	= 00000006		
RP_ER2	00000020		
RP_ER3	00000034		
RP_LA	0000001C		
RP_MR	0000000C		
RP_OF	00000024		
RP_OF_M_FMT	= 00001000		
RP_SN	00000030		
SIZ...	= 0000000B		
SIZE	= 00000008		
SS\$_CTRLERR	= 00000054		
SS\$_NORMAL	= 00000001		
SS\$_NOSUCHDEV	= 00000908		
SSERROR	00000157	R	03

+-----+
! 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	00000040 (64.)	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	000001A5 (421.)	03 (3.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE					

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.06	00:00:00.50
Command processing	111	00:00:00.64	00:00:02.23
Pass 1	354	00:00:11.55	00:00:23.78
Symbol table sort	0	00:00:01.81	00:00:02.81
Pass 2	109	00:00:02.35	00:00:04.49
Symbol table output	8	00:00:00.08	00:00:00.09
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	617	00:00:16.54	00:00:33.95

The working set limit was 1650 pages.
65133 bytes (128 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1159 non-local and 12 local symbols.
526 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	3
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	11

1175 GETS were required to define 11 macros.

There were no errors, warnings or information messages.

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

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY