ECVMB·1.0 VAX 11750 BOOT BLOCKS EP·ECVMB·DL 1.0 FICHE 1 OF 1

IDENTIFICATION

PRODUCT CODE: ZZ-ECVMB-1.0

PRODUCT TITLE: BOOT BLOCK LISTING

PRODUCT DATE: APRIL, 1981

DEPARTMENT: VAX ENGINEERING

Table of contents

```
0000     1          .TITLE  BOOTBLOCK
0000     2          .IDENT  'V02-001'
0000     3
0000     4  ;
0000     5  ;*****************************************************************************
0000     6  ;*                                                                           *
0000     7  ;*  COPYRIGHT (c) 1979, 1980                                                 *
0000     8  ;*  BY DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.                         *
0000     9  ;*                                                                           *
0000    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND  COPIED    *
0000    11  ;*  ONLY  IN  ACCORDANCE  WITH  THE  TERMS  OF  SUCH  LICENSE AND WITH THE   *
0000    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE.  THIS SOFTWARE OR  ANY  OTHER    *
0000    13  ;*  COPIES  THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    14  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE  IS  HEREBY    *
0000    15  ;*  TRANSFERRED.                                                             *
0000    16  ;*                                                                           *
0000    17  ;*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE  WITHOUT  NOTICE    *
0000    18  ;*  AND  SHOULD  NOT  BE  CONSTRUED  AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    19  ;*  CORPORATION.                                                             *
0000    20  ;*                                                                           *
0000    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR  RELIABILITY  OF  ITS    *
0000    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23  ;*                                                                           *
0000    24  ;*****************************************************************************
0000    25  ;
0000    26  ;++
0000    27  ;
0000    28  ; FACILITY:
0000    29  ;
0000    30  ;       Device-independent boot block for VAX
0000    31  ;
0000    32  ; ABSTRACT:
0000    33  ;
0000    34  ;       Reads a file (usually VMB.EXE) off the booting medium into
0000    35  ;       memory and transfers control to the VMB code.
0000    36  ;
0000    37  ; AUTHOR:
0000    38  ;
0000    39  ;       Carol Peters    23 August 1979
0000    40  ;
0000    41  ; REVISION HISTORY:
0000    42  ;
0000    43  ;       Robert Rappaport        13 Sept 1979
0000    44  ;               Simplified references to local data items.
0000    45  ;
0000    46  ;--
```

```
                        0000     48          .SBTTL  Declarations
                        0000     49
                        0000     50 ;
                        0000     51 ; Own storage.
                        0000     52 ;
                        0000     53
                        0000     54 FILE_STATS:                          ; Reserve space to contain
                        0000     55 FILE_SIZE:
                00000000 0000     56          .LONG   0                 ; # blocks in primary boot.
                        0004     57 START_LBN:
                00000000 0004     58          .LONG   0                 ; Swapped words of start LBN.
                        0008     59 LOAD_ADDR:
                00000000 0008     60          .LONG   0                 ; Load address for primary boot.
                        000C     61                                    ; NOTE - the load address here is
                        000C     62                                    ; relative to the base of the 64KB
                        000C     63                                    ; of physical memory in which we are
                        000C     64                                    ; currently running.  As explained
                        000C     65                                    ; below, the UNIBUS (or MASSBUS) map
                        000C     66                                    ; registers numbered 0-127 will be
                        000C     67                                    ; mapped to this same 64KB.  Therefore,
                        000C     68                                    ; this address, as is, can be used by
                        000C     69                                    ; UNIBUS (or MASSBUS) devices to
                        000C     70                                    ; pinpoint where to load the primary
                        000C     71                                    ; bootstrap program.  However, to
                        000C     72                                    ; calculate the physical memory address
                        000C     73                                    ; corresponding to this relative
                        000C     74                                    ; address, we must add in the physical
                        000C     75                                    ; offset of the base of the 64KB.
```

```
                        77          .SBTTL  BOO$BLOCK - reads in and starts boot code
                        78
                        79 ;++
                        80 ; Functional description:
                        81 ;
                        82 ;       The boot block code reads the primary bootstrap file into
                        83 ;       physical memory a block at a time. The code calls the device-
                        84 ;       dependent ROM subroutine once for each block in the bootstrap
                        85 ;       file. Then the routine jumps to byte 0 of the loaded code.
                        86 ;
                        87 ; Inputs:
                        88 ;
                        89 ;       R0      - type of boot device
                        90 ;       R1      - (UNIBUS) address of the I/O page for the boot device's
                        91 ;                         UNIBUS
                        92 ;                 (MASSBUS) address of the device's MASSBUS adapter
                        93 ;       R2      - (UNIBUS) 32-bit physical address of the boot device's
                        94 ;                         CSR (bits <31:24> must be zero)
                        95 ;                 (MASSBUS) adapter's controller/formatter number
                        96 ;       R3      - unit number of the boot device
                        97 ;       R5      - software boot control flags
                        98 ;       R6      - physical address of the device-dependent ROM routine
                        99 ;                 that reads an arbitrary LBN into memory
                       100 ;
                       101 ;       SP      - <base_address + ^X200> of 64kb of good memory
                       102 ;
                       103 ; Implicit inputs:
                       104 ;
                       105 ;       UNIBUS adapter map registers 0-127 are mapped to the 64kb of
                       106 ;       good memory. MR 0 maps to first page of memory, etc.
                       107 ;
                       108 ;       The boot block is loaded into the 1st page of the 64KB of
                       109 ;       memory, i.e. the page which corresponds to MR 0.
                       110 ;
                       111 ;       The first longword (bytes 0-3) of the boot block contains
                       112 ;       the size of the primary bootstrap.
                       113 ;
                       114 ;       The second longword (bytes 4-7) contains the starting LBN of
                       115 ;       the bootstrap file, expressed as swapped words.
                       116 ;
                       117 ;       The third longword (bytes 8-11) contains the relative offset
                       118 ;       from the base of the 64KB of memory into which we should load
                       119 ;       the primary bootstrap program.  This must be a positive
                       120 ;       number less than or equal to 64KB-(size*512) where size is the
                       121 ;       size of the primary bootstrap.
                       122 ;
                       123 ;       The starting LBN format is defined by DSC and cannot be
                       124 ;       changed. The load address is defined by WRITEBOOT and cannot
                       125 ;       be changed.
                       126 ;
                       127 ; Outputs:
                       128 ;
                       129 ;       R0      - type of boot device
                       130 ;       R1      - (UNIBUS) address of the I/O page for the boot device's
                       131 ;                         UNIBUS
                       132 ;                 (MASSBUS) address of the device's MASSBUS adapter
                       133 ;       R2      - (UNIBUS) 18-bit UNIBUS address of the boot device's
```

```
                                000C   134 ;                       CSR
                                000C   135 ;                         (MASSBUS) adapter's controller/formatter number
                                000C   136 ;       R3      - unit number of the boot device
                                000C   137 ;       R5      - software boot control flags
                                000C   138 ;       R6      - physical address of the device-dependent ROM routine
                                000C   139 ;                 that reads an arbitrary LBN into memory
                                000C   140 ;
                                000C   141 ;       SP      - <base_address + ^X200> of 64kb of good memory
                                000C   142 ;
                                000C   143 ; Implicit outputs:
                                000C   144 ;
                                000C   145 ;       The routine preserves R0-R1, R3, R4, R5-R6, R8, R10-R11, AP, and SP.
                                000C   146 ;
                                000C   147 ;       Transfers control to the 0th byte of the primary bootstrap
                                000C   148 ;       program.
                                000C   149 ;
                                000C   150 ;--
                                000C   151
                                000C   152 BOO$BLOCK_CODE:                                ; Start of device independent code.
                   F1 AF   9F   000C   153         PUSHAB  FILE_STATS                    ; Move physical address of base of
                                000F   154                                               ; 64KB of memory.
                6E    F6 AF   C0   000F   155         ADDL    LOAD_ADDR,(SP)            ; Add in relative load address. Result
                                0013   156                                               ; is physical address of load point.
                                0013   157                                               ; Leave on stack for final JMP inst.
                                0013   158
                  0131 8F   BB   0013   159         PUSHR   #^M<R0,R4,R5,R8>          ; Save 4 registers for temps.
                54    E6 AF   D0   0017   160         MOVL    FILE_SIZE,R4              ; Get # of blocks in VMB.
                58    E6 AF   B0   001B   161         MOVW    START_LBN,R8              ; Get upper word value of LBN.
             E0 AF    E4 AF   B0   001F   162         MOVW    START_LBN+2,START_LBN      ; Move lower word value of LBN into
                                0024   163                                               ; lower word position.
                DE AF    58   B0   0024   164         MOVW    R8,START_LBN+2            ; Move upper word value to upper
                                0028   165                                               ; word position.
                58    D9 AF   D0   0028   166         MOVL    START_LBN,R8              ; Pickup the swapped LBN.
                55    D9 AF   D0   002C   167         MOVL    LOAD_ADDR,R5              ; Get primary boot relative load addr.
                      10 AE   DD   0030   168         PUSHL   16(SP)                    ; Copy physical transfer address to
                                0033   169                                               ; top of stack for those devices such
                                0033   170                                               ; as the TU58 which need physical
                                0033   171                                               ; rather than UNIBUS virtual addresses.
                                0033   172
                                0033   173 READ_BLOCK:                                  ; VMB read loop.
                         66     16   0033   174         JSB     (R6)                      ; Call ROM read LBN routine.
                      01 50     E8   0035   175         BLBS    R0,NEXT_BLOCK             ; Branch on successful read.
                                00   0038   176         HALT                              ; Halt on failure to read.
                                0039   177
                                0039   178 NEXT_BLOCK:                                  ; Read next block.
          55    00000200 8F   C0   0039   179         ADDL    #^X200,R5                 ; Increment relative address 512 bytes.
          6E    00000200 8F   C0   0040   180         ADDL    #^X200,(SP)               ; Increment physical address one page.
                      BA AF   D6   0047   181         INCL    START_LBN                 ; Increment LBN number.
                58    B7 AF   D0   004A   182         MOVL    START_LBN,R8              ; Next LBN is LBN+1.
                      E2 54   F5   004E   183         SOBGTR  R4,READ_BLOCK             ; If more blocks, loop.
                         8E   D5   0051   184         TSTL    (SP)+                     ; Pop now useless data from stack.
                                0053   185
                                0053   186 ;
                                0053   187 ; The primary bootstrap program is now in physical memory starting at
                                0053   188 ; the specified load address. Restore the saved registers, convert the
                                0053   189 ; CSR address to an 18-bit UNIBUS address, and transfer control to the
                                0053   190 ; program.
```

```
                                0053   191 ;
                                0053   192
                     0131 8F    BA  0053   193          POPR    #^M<R0,R4,R5,R8>        ; Restore registers.
                52   FFFC0000 8F    CA  0057   194          BICL    #^XFFFC0000,R2          ; Reduce 32-bit CSR to 18-bit
                                0057   195                                                ; CSR that VMB expects.
                          9E    17  005E   196          JMP     @(SP)+                  ; Jump to primary bootstrap program.
                                0060   197
                                0060   198          .END
```

```
BOOTBLOCK_CODE   0000000C R     01
FILE_SIZE        00000000 R     01
FILE_STATS       00000000 R     01
LOAD_ADDR        00000008 R     01
NEXT_BLOCK       00000039 R     01
READ_BLOCK       00000033 R     01
START_LBN        00000004 R     01
```

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000060 (96.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	17	00:00:00.04	00:00:00.84
Command processing	24	00:00:00.25	00:00:01.69
Pass 1	34	00:00:00.40	00:00:02.45
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	57	00:00:00.36	00:00:02.02
Symbol table output	1	00:00:00.03	00:00:00.40
Psect synopsis output	1	00:00:00.00	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	134	00:00:01.08	00:00:07.42

```
The working set limit was 500 pages.
2142 bytes (5 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 7 non-local and 0 local symbols.
198 source lines were read in Pass 1, producing 9 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.
```

! Macro library statistics !

Macro library name	Macros defined
_DRA7:[BOOTS.OBJ]BOOTS.MLB;1	0
_DRA7:[SYS.OBJ]LIB.MLB;1	0
_DRA7:[SYSLIB]STARLET.MLB;1	0
TOTALS (all libraries)	0

```
0 GETS were required to define 0 macros.
There were no errors, warnings or information messages.
/LIS=LIS$:BOOTBLOCK/OBJ=OBJ$:BOOTBLOCK MSRC$:BOOTBLOCK/UPDATE=(ENH$:BOOTBLOCK)+EXECML$/LIB+LIB$:BOOTS.MLB/LIB
```