```
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
```

FILEID**BTMEM750

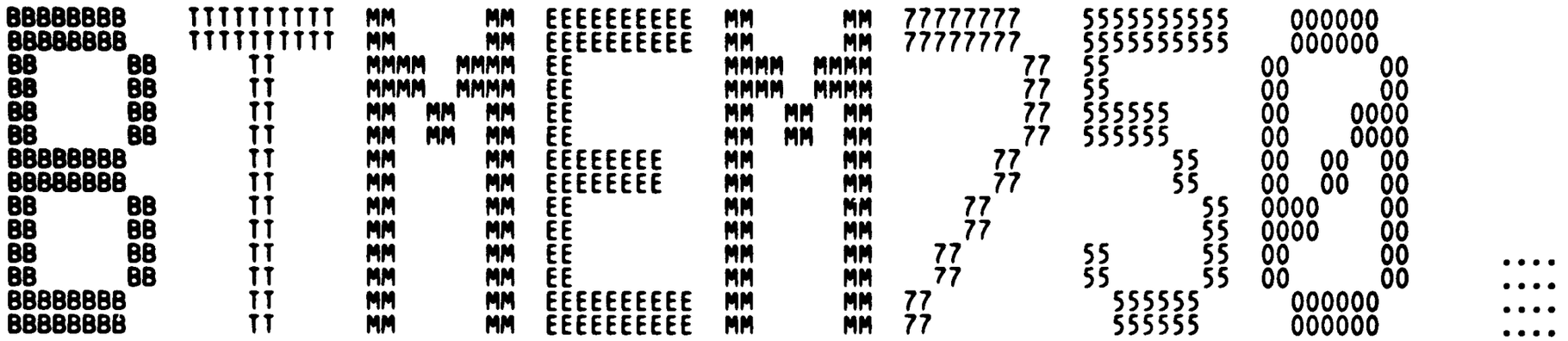

Table of contents

```
0000     1          .TITLE  BTMEM750 - Configure and Test 11/750 Memory
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:
0000    31  ;
0000    32  ;       Linked with VMB.EXE - part of the
0000    33  ;       bootstrap module for VAX 11/750 hardware.
0000    34  ;
0000    35  ; ENVIRONMENT:
0000    36  ;
0000    37  ;       Runs at IPL 31, kernel mode, memory management is OFF, IS=1
0000    38  ;       (running on interrupt stack), and code must be PIC.
0000    39  ;
0000    40  ;
0000    41  ; FUNCTIONAL DESCRIPTION:
0000    42  ;
0000    43  ;       This routine is 11/750 specific and
0000    44  ;       determines how many memory controllers are on the system,
0000    45  ;       where they are, how much memory they control, which pages
0000    46  ;       of that memory are present (and good). Then the routines
0000    47  ;       set bits in the PFN bitmap to represent each present (and
0000    48  ;       good) page of memory.
0000    49  ;
0000    50  ;       As a side effect, the routines store the type of adapter located
0000    51  ;       at each bus slot in the RPB.
0000    52  ;
0000    53  ; INPUTS:
0000    54  ;
0000    55  ;       R5      - address of 1st RPB configuration code field
0000    56  ;       R7      - address of the SCB
0000    57  ;       R11     - address of the RPB
```

```
0000     58 ;
0000     59 ; IMPLICIT INPUTS:
0000     60 ;
0000     61 ;       The positions on the 11/750 system bus are called slots, and are
0000     62 ;       identified by slot numbers 16-31. Slots 16-25 are called fixed
0000     63 ;       slots. If an adapter is present in a fixed slot, the adapter
0000     64 ;       must be of a predefined type. For example,
0000     65 ;
0000     66 ;               slot 16 = memory controller 0   = ^XF20000
0000     67 ;               slot 20 = MASSBUS adapter 0     = ^XF28000
0000     68 ;               slot 24 = UNIBUS adapter 0      = ^XF30000
0000     69 ;
0000     70 ;       Adapters at fixed slots do not have configuration registers.
0000     71 ;       This routine determines their presence by reading the first
0000     72 ;       longword of the slot and not having the read result in a
0000     73 ;       non-existent memory machine check. The data resulting from the
0000     74 ;       read will be garbage.
0000     75 ;
0000     76 ;       10 fixed system bus slots are currently defined for the 11/750:
0000     77 ;
0000     78 ;               4 memory controllers, starting at ^XF20000
0000     79 ;               4 MASSBUS adapters, starting at ^XF28000
0000     80 ;               2 UNIBUS adapters, starting at ^XF30000
0000     81 ;
0000     82 ;       The other 6 system bus slots are floating slots. If an adapter
0000     83 ;       is present, that adapter must have a configuration register as
0000     84 ;       the first location in the slot's address space. The register
0000     85 ;       must contain the adapter type in the low byte.
0000     86 ;
0000     87 ;       The 11/750 currently supports only 1 memory controller, located
0000     88 ;       at ^XF20000. The 11/750 does not have interleaved memory. This
0000     89 ;       routine tests all controller slots as though memory could exist
0000     90 ;       at them.
0000     91 ;
0000     92 ;       Memory controller registers also contain the starting page
0000     93 ;       number / 128. This routine determines where the end of memory
0000     94 ;       on a memory controller is by analyzing the memory present map
0000     95 ;       in the third memory controller register.
0000     96 ;
0000     97 ; OUTPUTS:
0000     98 ;
0000     99 ;       R7, R8, R11, and SP are preserved
0000    100 ;       All others (including AP and FP) are altered
0000    101 ;
0000    102 ; IMPLICIT OUTPUTS:
0000    103 ;
0000    104 ;       The PFN bitmap is modified to describe all of physical memory.
0000    105 ;
0000    106 ;       RPB$L_PFNCNT stores the number of pages of physical memory.
0000    107 ;
0000    108 ;       All single parity errors in memory are cleared.
0000    109 ;
0000    110 ;       RPB$B_CONFREG describes each NEXUS on the system bus with an
0000    111 ;       adapter type code.
0000    112 ;
0000    113 ; AUTHOR:
0000    114 ;
```

```
                        115 ;       C. A. Samuelson, creation date 24-April-1981
                        116 ;
                        117 ; REVISION HISTORY:
                        118 ;
                        119 ;       V03-002 TCM0003         Trudy C. Matthews       27-Apr-1983
                        120 ;               Change sense of CRDTEST flag from an enable to an inhibit, i.e.
                        121 ;               remove pages with CRD errors by default.
                        122 ;
                        123 ;       V03-001 TCM0002         Trudy C. Matthews       26-Jan-1983
                        124 ;               Add support for RPB$V_CRDTEST flag, which specifies that
                        125 ;               pages with CRD errors be removed.
                        126 ;
                        127 ;       V03-001 KDM0078         Kathleen D. Morse       15-Mar-1982
                        128 ;               Clear all MA780-specific boot flags.
                        129 ;--
```

```
                        .SBTTL  Declarations

                        .DEFAULT DISPLACEMENT, WORD

                ;
                ; Macros to describe VMS data structures
                ;

                        $DMPDEF                         ; System dump file header definitions
                        $I0750DEF                       ; 11/750 definitions
                        $NDTDEF                         ; Nexus device types
                        $PRDEF                          ; Processor registers
                        $RPBDEF                         ; Restart parameter block
                        $UBIDEF                         ; 11/750 UNIBUS adapter

                ;
                ; Macros
                ;

                        .MACRO  ERROR,STR               ; Outputs an error string to the
                        BSBW    ERROUT                  ; console terminal.
                        .ASCIZ  STR
                        .ENDM   ERROR

                        .PSECT  YBTMEM,LONG
```

```
                        0000   157          .SBTTL  CHECKMEM_750, Identify 11/750 memory
                        0000   158
                        0000   159 ;++
                        0000   160 ;
                        0000   161 ; CHECKMEM_750, Locate and test memory for 11/750
                        0000   162 ;
                        0000   163 ;--
                        0000   164
                        0000   165 ;
                        0000   166 ; The table that follows identifies the adapter type codes of adapters
                        0000   167 ; located in fixed 750 SBI slots. The last 6 slots contain zeroes to
                        0000   168 ; indicate floating slots.
                        0000   169 ;
                        0000   170
                        0000   171 ADAP_TYPE_750:                          ; Adapter type table for 11/750
                  10    0000   172          .BYTE   NDT$_MEM16NI           ; Memory controller 0.
                  40    0001   173          .BYTE   NDT$_MPM0              ; Multiport memory 0.
                  41    0002   174          .BYTE   NDT$_MPM1              ; Multiport memory 1.
                  42    0003   175          .BYTE   NDT$_MPM2              ; Multiport memory 2.
                  20    0004   176          .BYTE   NDT$_MB                ; MASSBUS adapter.
                  20    0005   177          .BYTE   NDT$_MB                ; MASSBUS adapter.
                  20    0006   178          .BYTE   NDT$_MB                ; MASSBUS adapter.
                  20    0007   179          .BYTE   NDT$_MB                ; MASSBUS adapter.
                  28    0008   180          .BYTE   NDT$_UB0               ; UNIBUS adapter 0.
                  29    0009   181          .BYTE   NDT$_UB1               ; UNIBUS adapter 1.
                  00    000A   182          .BYTE   0                      ; Floating slot.
                  00    000B   183          .BYTE   0                      ; Floating slot.
                  00    000C   184          .BYTE   0                      ; Floating slot.
                  00    000D   185          .BYTE   0                      ; Floating slot.
                  00    000E   186          .BYTE   0                      ; Floating slot.
                  00    000F   187          .BYTE   0                      ; Floating slot.
                        0010   188
                        0010   189
                        0010   190 ;
                        0010   191 ; Get address of appropriate fixed slot assignment taable for each
                        0010   192 ; CPU:
                        0010   193 ;
                        0010   194
                        0010   195 CHECKMEM_750::                          ; Entry for 11/750.
                52      ED AF   9E 0010   196          MOVAB   ADAP_TYPE_750,R2       ; Get 11/750 fixed slot assignment
                        0014   197                                         ; table
                        0014   198 ;
                        0014   199 ; Start testing slot positions to find adapters. First save the stack
                        0014   200 ; position so it can be restored after a machine check.
                        0014   201 ;
                        0014   202
                        0014   203 INIT_SEARCH:                            ; Start searching for adapters
                   5C     10   9A 0014   204          MOVZBL  #I0750$AL_NNEX,AP      ; Set up NEXUS loop counter.
                   5D     5E   D0 0017   205          MOVL    SP,FP                  ; Save current top of stack.
                        001A   206
                        001A   207 ;
                        001A   208 ; Set up the physical address of the 1st slot on the system bus and
                        001A   209 ; the address of the adapter type table.
                        001A   210 ;
                        001A   211
              54    00F20000 9F   9E 001A   212          MOVAB   @#I0750$AL_IOBASE,R4   ; Get address of 1st slot.
                        0021   213
```

```
                                0021   214 ;
                                0021   215 ; During this memory locate and test loop, the following registers are
                                0021   216 ; used:
                                0021   217 ;
                                0021   218 ;       R0      - the contents of the slot's configuration register;
                                0021   219 ;                 the 3rd memory controller register;
                                0021   220 ;       R1      - bit position within memory present map;
                                0021   221 ;       R2      - address of the next byte in the 750-specific adapter
                                0021   222 ;                 type table; address of TEST_QUAD_750 routine
                                0021   223 ;       R3      - the default adapter type for the current slot;
                                0021   224 ;                 the number of pages in this controller
                                0021   225 ;       R4      - address of the configuration register at the current
                                0021   226 ;                 slot position
                                0021   227 ;       R5      - address of next byte in RPB adapter type table
                                0021   228 ;       R7      - address of the SCB
                                0021   229 ;       R9      - bit setting in memory present map;
                                0021   230 ;                 starting page number in this controller
                                0021   231 ;       R10     - address of the memory description list in RPB (pagcnt & pfn)
                                0021   232 ;       R11     - address of the RPB
                                0021   233 ;
                                0021   234 ; Initialize the RPB slot field to a zero and obtain the default adapter
                                0021   235 ; type for this slot. Then set up a machine check fault handler to gain
                                0021   236 ; control if the loop addresses a non-existent configuration register
                                0021   237 ; (an empty slot). Then read the slot's configuration register.
                                0021   238 ;
                                0021   239 ; Initialize R10 (RPB  memory descripter list pointer) for search loop
                                0021   240 ;
          5A    00BC CB   9E    0021   241         MOVAB   RPB$L_MEMDSC(R11),R10   ; Set pointer to memory description list
    30 AB     00005800 8F   CA  0026   242         BICL    #<RPB$M_MPM ! RPB$M_USEMPM ! RPB$M_FINDMEM>, -
                                002E   243                 RPB$L_BOOTR5(R11)       ; Clear all MA780-specific boot flags
                                002E   244
                                002E   245
                                002E   246 TRY_NEXUS_750:                           ; Memory locate and test loop.
              85    94          002E   247         CLRB    (R5)+                   ; Assume nothing on slot.
          53    82    9A        0030   248         MOVZBL  (R2)+,R3                ; Get default adapter type.
04 A7     00B5'CF   9E          0033   249         MOVAB   DO_NEXT_750+1,4(R7)     ; Set up fault handler (+1 for
                                0039   250                                         ; handler execution on the
                                0039   251                                         ; interrupt stack).
          50    64    D0        0039   252         MOVL    (R4),R0                 ; Read CR at current slot.
                                003C   253
                                003C   254 ;
                                003C   255 ; Execution continues here if the configuration register is present.
                                003C   256 ; Load the adapter type into the RPB field. Then, if the adapter type
                                003C   257 ; is a memory controller, proceed to test memory. Otherwise, move to
                                003C   258 ; the next SBI slot.
                                003C   259 ;
                                003C   260
              53    D5          003C   261         TSTL    R3                      ; Is this a floating slot?
              06    12          003E   262         BNEQ    FIXED_SLOT              ; Branch if not
    FF A5     50    90          0040   263         MOVB    R0,-1(R5)               ; Save type read from config register
              04    11          0044   264         BRB     CHECK_TYPE              ; Check if memory controller
                                0046   265
                                0046   266 FIXED_SLOT:                            ; Slot is fixed assignment
    FF A5     53    90          0046   267         MOVB    R3,-1(R5)               ; Save fixed type
                                004A   268
                                004A   269 CHECK_TYPE:                            ; Check adapter type for memory
                                004A   270
```

```
                10   FF A5   91  004A   271          CMPB    -1(R5),#NDT$_MEM16NI    ; Memory controller?
                     64 12   12  004E   272          BNEQ    DO_NEXT_750             ; No, advance to next slot.
                             0050   273
                             0050   274 ;
                             0050   275 ; Memory controller found:
                             0050   276 ; Find out whether the memory addresses are within legal bounds by
                             0050   277 ; computing the number of pages on the controller and the starting page.
                             0050   278 ; number.  Each controller is assumed to contain up to 8 array
                             0050   279 ; boards.  Both 16Kb and 64Kb boards can be supported on the same
                             0050   280 ; controller.  Sizing is done by examining bits <15:0> in CSR2 in
                             0050   281 ; pairs.  Each pair is coded as follows:
                             0050   282 ;
                             0050   283 ;       00 --> no board, 0 Mbyte
                             0050   284 ;       11 --> 16Kb chips, 1/4 Mbyte
                             0050   285 ;       other --> 64Kb chips, 1Mbyte
                             0050   286 ;
                             0050   287
             50    08 A4     D0  0050   288          MOVL    8(R4),R0                ; Get starting address register.
                   53 D4     D4  0054   289          CLRL    R3                      ; Start with zero 128K chunks.
                51   0E D0     D0  0056   290          MOVL    #14,R1                  ; Start with top array slot in map.
                             0059   291
                             0059   292 TRY_NXT_ARRAY:
          59   50    02 51 EF  0059   293          EXTZV   R1,#2,R0,R9             ; Get next array size
                      0B 13    005E   294          BEQL    STEP_ARRAY              ; Branch if 0 (no board present)
                   53 02 C0     0060   295          ADDL    #2,R3                   ; Add at least 2 128 Kb chunks
                   03 59 D1     0063   296          CMPL    R9,#3                   ; Got 1/4 Mbyte of memory here?
                      03 13     0066   297          BEQL    STEP_ARRAY              ; Branch if so
                   53 06 C0     0068   298          ADDL    #6,R3                   ; No, add 6 more 128Kb chunks
                             006B   299                                          ;  for 1Mbyte of memory here
                             006B   300
                             006B   301 STEP_ARRAY:
                      51 D7     006B   302          DECL    R1                      ; Decrease bit # in sizing register
                   E9 51 F4     006D   303          SOBGEQ  R1,TRY_NXT_ARRAY        ;  by 2 and try next array, if any
             53    53 08 78     0070   304          ASHL    #8,R3,R3                ; Convert number of 128K byte
                             0074   305                                          ; chunks to number of pages.
          59   50    07 11 EF  0074   306          EXTZV   #17,#7,R0,R9            ; Starting page number/128
               59    59 07 78     0079   307          ASHL    #7,R9,R9                ; Multiply by 128 = page number.
                  8A 53 D0     007D   308          MOVL    R3,(R10)+               ; Save # of pages in this memory
                             0080   309          ASSUME  DMP$V_TR EQ 24
                             0080   310          ASSUME  DMP$S_TR EQ 8
                FF AA 10 90  0080   311          MOVB    #10750$AL_NNEX,-1(R10)  ; Compute the TR number for this
                FF AA 5C 82  0084   312          SUBB    AP,-1(R10)              ; memory and store in descriptor
                  8A 59 D0     0088   313          MOVL    R9,(R10)+               ; Save starting PFN for this memory
                             008B   314
                             008B   315 ;
                             008B   316 ; Before starting memory test, establish a page skipping handler for
                             008B   317 ; machine checks, and turn off the cache so that writes followed by
                             008B   318 ; reads to memory don't write to memory and then read from the cache.
                             008B   319 ; Also, enable CRD error reporting if requested by the RPB BOOTR5 flag.
                             008B   320 ;
          08 30 AB    10 E0  008B   321          BBS     #RPB$V_CRDTEST, -       ; Branch around CRD enable if inhibit
                             0090   322                  RPB$L_BOOTR5(R11),10$   ; set.
04 A4    10000000 8F  D0  0090   323          MOVL    #^X10000000,4(R4)       ; Enable reporting correctable errors.
   04 A7    00FD'CF 9E  0098   324 10$:     MOVAB   PAGE_MCHECK_750+1,4(R7) ; Set page skipping handler (+1
                             009E   325                                          ; for interrupt stack).
25    FFFFFFFF 8F  DA  009E   326          MTPR    #-1,#PR$_CADR           ; Turn off memory cache.
                   52 DD     00A5   327          PUSHL   R2                      ; Save address of fixed nexus table.
```

```
                52   00D2'CF   9E   00A7    328          MOVAB   TEST_QUAD_750,R2         ; Page test routine address
                       FF51'   30   00AC    329          BSBW    BOO$TEST_MEM            ; Test the specified range of PFN's
                          04   BA   00AF    330          POPR    #^M<R2>                 ; Retrieve fixed nexus table address.
                          01   11   00B1    331          BRB     DO_NEXT_750             ; Do the next controller if any
                               00B3    332
                               00B3    333          .ALIGN  LONG                    ; Longword-aligned handler.
                               00B4    334
                               00B4    335 ;
                               00B4    336 ; Fault handler for non-existent configuration register, or unreadable
                               00B4    337 ; registers, or a non-memory controller slot device. Restore stack
                               00B4    338 ; pointer, clear all errors, and try for another slot if any remain.
                               00B4    339 ;
                               00B4    340
                               00B4    341 DO_NEXT_750:                            ; Skip to next slot.
                5E    5D   D0   00B4    342          MOVL    FP,SP                   ; Restore stack pointer.
             26  FFFFFFFF 8F   DA   00B7    343          MTPR    #-1,#PR$_MCESR          ; Clear any faults.
                54   2000 C4   9E   00BE    344          MOVAB   I0750$AL_PERNEX(R4),R4  ; Move to next slot.
                      09 5C   F5   00C3    345          SOBGTR  AP,NEXT_NEXUS_750       ; If still a slot, loop.
                               00C6    346
                               00C6    347 ;
                               00C6    348 ; Reestablish the normal machine check fault handler.
                               00C6    349 ;
                               00C6    350
             04 A7   0001'CF   DE   00C6    351          MOVAL   UNEXP_MCHK+1,4(R7)      ; Reset SCB vector.
                          8A   D4   00CC    352          CLRL    (R10)+                  ; Indicate end of RPB memory descr list
                          05   00CE    353          RSB                             ; Return to main routine.
                               00CF    354
                               00CF    355 ;
                               00CF    356 ; Extra label and branch here to loop back through the slot testing
                               00CF    357 ; code.
                               00CF    358 ;
                               00CF    359
                               00CF    360 NEXT_NEXUS_750:                         ; Try the next slot.
             FF5C      31   00CF    361          BRW     TRY_NEXUS_750           ; Branch to top of loop.
                               00D2    362 ;++
                               00D2    363 ;
                               00D2    364 ; Functional Description:
                               00D2    365 ;
                               00D2    366 ;       Test a page of 780 memory,
                               00D2    367 ;
                               00D2    368 ; Calling Sequence:
                               00D2    369 ;
                               00D2    370 ;       JSB     TEST_QUAD_750
                               00D2    371 ;
                               00D2    372 ; Inputs:
                               00D2    373 ;
                               00D2    374 ;       R0 = starting address to test
                               00D2    375 ;       R1 = Quad word iteration count (64)
                               00D2    376 ;       R11= Address of RPB
                               00D2    377 ;
                               00D2    378 ; Outputs:
                               00D2    379 ;
                               00D2    380 ;       Returns via RSB if the entire page is OK
                               00D2    381 ;       Error exit via Machine Check code to BOO$PAGE_MCHECK
                               00D2    382 ;
                               00D2    383 ;--
                               00D2    384
```

```
                        00D2    385 TEST_QUAD_750:                          ; Test 1 quadword at a time.
                60  7C  00D2    386         CLRQ    (R0)                    ; Clear a quadword.
              80    80  D1  00D4    387         CMPL    (R0)+,(R0)+             ; Read both longwords, and
                        00D7    388                                         ; advance to next quadword.
                        00D7    389
                        00D7    390 ;
                        00D7    391 ; If no gross errors occur in the clear to the quadword or in the
                        00D7    392 ; subsequent read instruction, then execution continues below. Otherwise
                        00D7    393 ; execution goes to the fault handler.
                        00D7    394 ;
                        00D7    395
                 F8 51  F5  00D7    396         SOBGTR  R1,TEST_QUAD_750        ; Continue clearing unless done.
                    10  E1  00DA    397         BBC     #RPB$V_CRDTEST,-        ; Branch if CRD test is requested.
              01 30 AB          00DC    398                 RPB$L_BOOTR5(R11),10$
                        05  00DF    399 5$:     RSB
                        00E0    400 ;
                        00E0    401 ; Check if a CRD error occurred on this page.
                        00E0    402 ;
                        00E0    403 10$:
          7E    00F20000 9F   D0  00E0    404         MOVL    @#IO750$AL_IOBASE,-(SP) ; Get memory CSR 0.
          00F20000 9F   6E  D0  00E7    405         MOVL    (SP),@#IO750$AL_IOBASE  ; Clear errors, just in case.
          8E    20000000 8F   D3  00EE    406         BITL    #^X20000000,(SP)+       ; Check for CRD error.
                    E8  13  00F5    407         BEQL    5$                      ; Branch if no CRD error occurred.
                    0A  11  00F7    408         BRB     ERR_EXIT_750            ; Else take error path.
                        00F9    409
                        00F9    410         .ALIGN  LONG                        ; All handlers longword-aligned.
                        00FC    411
                        00FC    412 ;
                        00FC    413 ; Handler that gains control when a page has gross memory errors. Just
                        00FC    414 ; clear the error, recover the stack top, and advance to the next page.
                        00FC    415 ;
                        00FC    416
                        00FC    417 PAGE_MCHECK_750:                        ; Handle machine check.
       26   FFFFFFFF 8F   DA  00FC    418         MTPR    #-1,#PR$_MCESR          ; Clear error indicator.
                        0103    419 ERR_EXIT_750:
                  FEFA' 31  0103    420         BRW     BOO$PAGE_MCHECK         ; Exit to common bad page code
                        0106    421         .END
```

```
ADAP_TYPE_750                   00000000 R      02
BOO$PAGE_MCHECK                 ******** X      02
BOO$TEST_MEM                    ******** X      02
CHECKMEM_750                    00000010 RG     02
CHECK_TYPE                      0000004A R      02
DMP$S_TR                    =   00000008
DMP$V_TR                    =   00000018
DO_NEXT_750                     000000B4 R      02
ERR_EXIT_750                    00000103 R      02
FIXED_SLOT                      00000046 R      02
INIT_SEARCH                     00000014 R      02
I0750$AL_IOBASE             =   00F20000
I0750$AL_NNEX               =   00000010
I0750$AL_PERNEX             =   00002000
NDT$_MB                     =   00000020
NDT$_MEM16NI                =   00000010
NDT$_MPM0                   =   00000040
NDT$_MPM1                   =   00000041
NDT$_MPM2                   =   00000042
NDT$_UB0                    =   00000028
NDT$_UB1                    =   00000029
NEXT_NEXUS_750                  000000CF R      02
PAGE_MCHECK_750                 000000FC R      02
PR$_CADR                    =   00000025
PR$_MCESR                   =   00000026
RPB$L_BOOTR5                =   00000030
RPB$L_MEMDSC                =   000000BC
RPB$M_FINDMEM               =   00004000
RPB$M_MPM                   =   00000800
RPB$M_USEMPM                =   00001000
RPB$V_CRDTEST               =   00000010
STEP_ARRAY                      0000006B R      02
TEST_QUAD_750                   000000D2 R      02
TRY_NEXUS_750                   0000002E R      02
TRY_NXT_ARRAY                   00000059 R      02
UNEXP_MCHK                      ******** X      02
```

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
YBTMEM	00000106	(262.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	31	00:00:00.06	00:00:00.55
Command processing	113	00:00:00.67	00:00:02.30
Pass 1	185	00:00:03.89	00:00:09.83
Symbol table sort	0	00:00:00.37	00:00:00.59

Pass 2	86	00:00:01.15	00:00:02.57
Symbol table output	6	00:00:00.04	00:00:00.04
Psect synopsis output	2	00:00:00.03	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	425	00:00:06.21	00:00:15.91

The working set limit was 1350 pages.
20780 bytes (41 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 317 non-local and 3 local symbols.
421 source lines were read in Pass 1, producing 13 object records in Pass 2.
15 pages of virtual memory were used to define 14 macros.

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

Macro library name	Macros defined
_255DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
_255DUA28:[SYS.OBJ]LIB.MLB;1	4
_255DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	10

386 GETS were required to define 10 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:BTMEM750/OBJ=OBJ$:BTMEM750 MSRC$:BTMEM750/UPDATE=(ENH$:BTMEM750)+EXECML$/LIB+LIB$:BOOTS.MLB/LIB
```

CONFIG
LIS

BTMEM855
LIS

BTMEM790
LIS

CONFIGURE
LIS

BOOTDEF
LIS

BOOTIO
LIS

CONFIGMN
LIS

BOOTDRIVR
LIS

BTMEM730
LIS

BTMEM750
LIS

BTMEM780
LIS

BOOTBLOCK
LIS