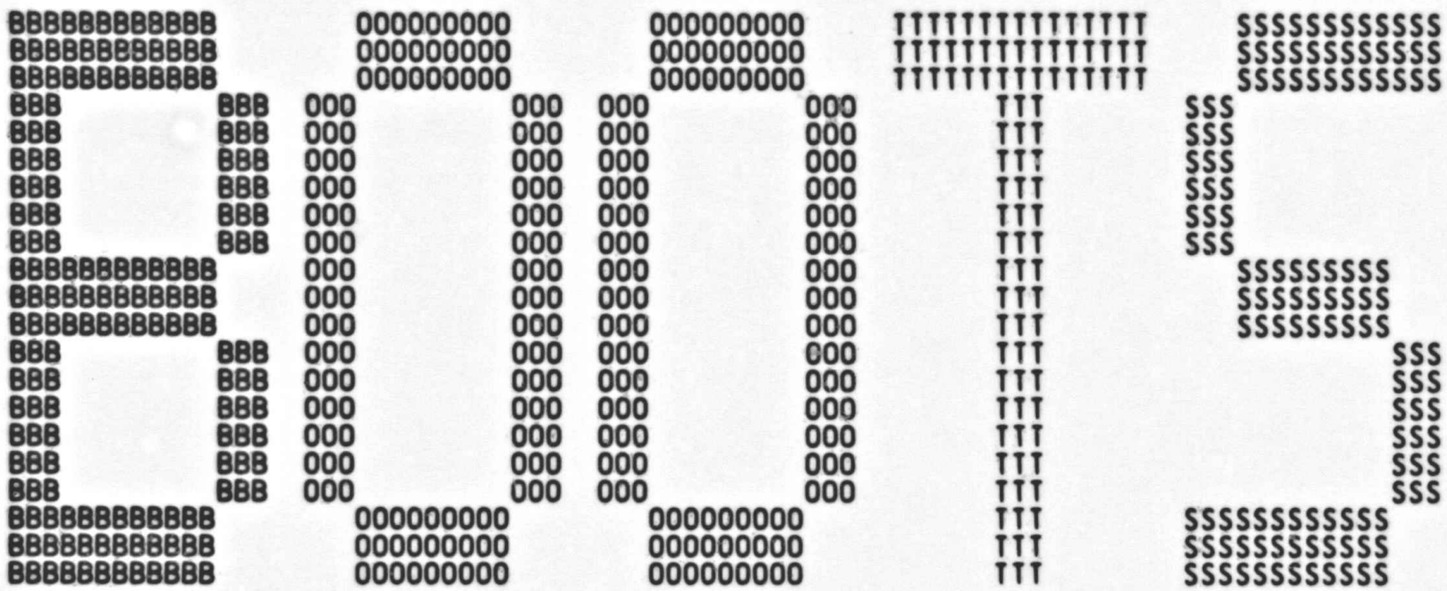
BBBBBBBBBBB OOOOOOOOO OOOOOOOOO TTTTTTTTTTTTTTT SSSSSSSSSSS
BBB BBB OOO OOO OOO OOO TTT SSS
BBBBBBBBBBB OOO OOO OOO OOO TTT SSSSSSSSS
BBB BBB OOO OOO OOO OOO TTT SSS
BBBBBBBBBBB OOOOOOOOO OOOOOOOOO TTT SSSSSSSSSSS

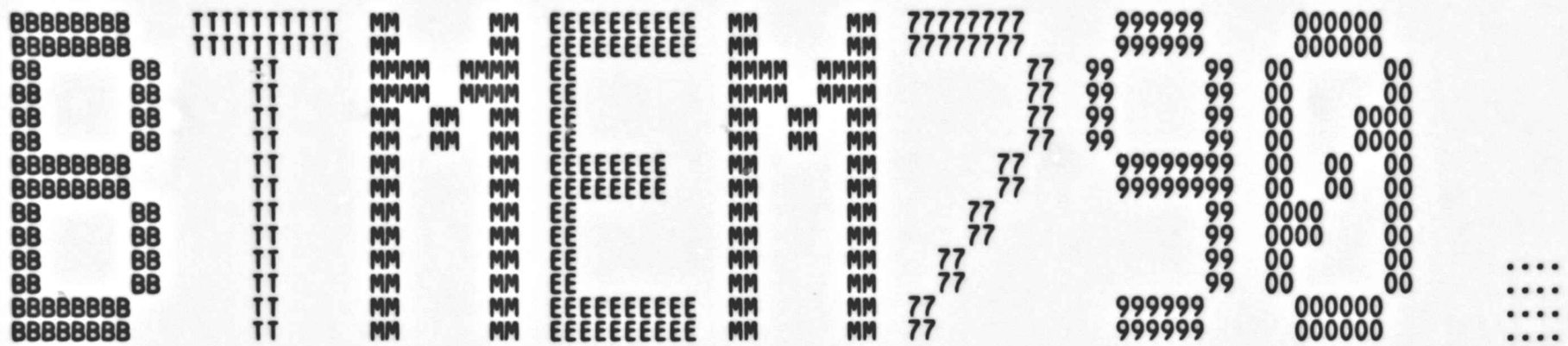
BBBBBBBB TTTTTTTTTT MM MM EEEEEEEEEE MM MM 77777777 999999 000000
BBBBBBBB TTTTTTTTTT MM MM EEEEEEEEEE MM MM 77777777 999999 000000
BB BB TT MMMM MMMM EE MMMM MMMM 77 99 99 00 00
BB BB TT MMMM MMMM EE MMMM MMMM 77 99 99 00 00
BB BB TT MM MM MM EE MM MM MM 77 99 99 00 0000
BB BB TT MM MM MM EE MM MM MM 77 99 99 00 0000
BBBBBBBB TT MM MM EEEEEEEE MM MM 77 99999999 00 00 00
BBBBBBBB TT MM MM EEEEEEEE MM MM 77 99999999 00 00 00
BB BB TT MM MM EE MM MM 77 99 0000 00
BB BB TT MM MM EE MM MM 77 99 0000 00
BB BB TT MM MM EE MM MM 77 99 00 00
BB BB TT MM MM EE MM MM 77 99 00 00
BBBBBBBB TT MM MM EEEEEEEEEE MM MM 77 999999 000000
BBBBBBBB TT MM MM EEEEEEEEEE MM MM 77 999999 000000
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
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

Table of contents

```
0000     1          .TITLE  BTMEM790 - Configure and Test 11/790 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  ;       BOOTS
0000    33  ;
0000    34  ; ENVIRONMENT:
0000    35  ;
0000    36  ;       Linked with VMB.EXE; runs at IPL 31, kernel mode, memory management
0000    37  ;       OFF,  PSL<IS>=1 (on interrupt stack), and code must be PIC.
0000    38  ;
0000    39  ; ABSTRACT:
0000    40  ;
0000    41  ;       This module is 11/790-specific.  It contains routines that:
0000    42  ;               - locate all of 11/790 physical memory
0000    43  ;               - test a range of 11/790 memory for hard (RDS) errors
0000    44  ;               - handle 11/790 machine checks generated when encountering
0000    45  ;                 hard memory errors
0000    46  ;
0000    47  ;       The routines in this module, in conjunction with common memory routines
0000    48  ;       in VMB.EXE, build a PFN bitmap that identifies each page of good 11/790
0000    49  ;       memory.
0000    50  ;
0000    51  ; AUTHOR:  TRUDY MATTHEWS,      CREATION DATE: 14-July-1982
0000    52  ;
0000    53  ; MODIFIED BY:
0000    54  ;
0000    55  ;       V03-009 TCM0008         Trudy C. Matthews       23-Jul-1984
0000    56  ;               Turn off cache before testing memory.  Test memory a page
0000    57  ;               at a time instead of a quadword at a time.
```

```
0000     58 ;
0000     59 ;       V03-008 TCM0007         Trudy C. Matthews       28-Nov-1983
0000     60 ;               Fix "EXTZV" instruction that extracts the PFN field from a
0000     61 ;               physical address.
0000     62 ;
0000     63 ;       V03-007 TCM0006         Trudy C. Matthews       17-Oct-1983
0000     64 ;               Use "MCOML #0,dest" instead of "MCOML #1,dest" to write a
0000     65 ;               pattern of all 1's.
0000     66 ;
0000     67 ;       V03-006 TCM0005         Trudy C. Mattehws       02-Jun-1983
0000     68 ;               Correct bug in TCM0001; we were writing the wrong bit to
0000     69 ;               disable ECC correction.  Also correct algorithm that tests
0000     70 ;               a quadword of memory by writing all 1's and then all 0's.
0000     71 ;
0000     72 ;       V03-005 TCM0004         Trudy C. Matthews       19-May-1983
0000     73 ;               Don't allow disable of CRDTEST to skip R4 initialization.
0000     74 ;
0000     75 ;       V03-004 TCM0003         Trudy C. Matthews       27-Apr-1983
0000     76 ;               Change sense of CRDTEST flag from an enable to an inhibit; i.e.
0000     77 ;               remove pages with CRD errors by default.
0000     78 ;
0000     79 ;       V03-003 TCM0002         Trudy C. Matthews       26-Jan-1983
0000     80 ;               Correct bug in TCM0001; didn't clear the "machine check on
0000     81 ;               CRD error" flag after memory testing was complete.
0000     82 ;
0000     83 ;       V03-002 TCM0001         Trudy C. Matthews       20-Oct-1982
0000     84 ;               Added optional ability to remove pages with single-bit
0000     85 ;               memory errors, in addition to always removing pages with
0000     86 ;               double-bit memory errors.
0000     87 ;
0000     88 ;--
```

```
                        .SBTTL  Declarations
                ;
                ; Macros to describe VMS data structures.
                ;
                        $CSWPDEF                        ; 11/790 cache sweep register
                        $EHSRDEF                        ; 11/790 Error Handling status reg
                        $IO790DEF                       ; 11/790 I/O spce definitions
                        $MDCTLDEF                       ; Define memory data control register.
                        $PAMMDEF                        ; Physical Address Memory Map defs
                        $PR790DEF                       ; 11/790-specific processor registers
                        $RPBDEF                         ; Restart Parameter Block definitions

                        .PSECT  YBTMEM,LONG
```

```
0000   105          .SBTTL  CHECKMEM_790, Locate 11/790 memory
0000   106 ;++
0000   107 ;
0000   108 ; Routine CHECKMEM_790
0000   109 ;
0000   110 ; VENUS address space and the PAMM
0000   111 ; --------------------------------
0000   112 ;       VENUS's design provides for 512 Mb of memory physical address space,
0000   113 ; and 512 Mb of I/O physical address space.  Memory physical addresses can
0000   114 ; range from 00000000 to 1FFFFFFF; I/O space physical addresses can range from
0000   115 ; 20000000 to 3FFFFFFF.
0000   116 ;
0000   117 ;       On VENUS systems, VMS will determine the memory configuration by using
0000   118 ; a structure called the PAMM (physical array memory map), which is set up by
0000   119 ; the VENUS console to map VENUS address space.  The PAMM is an array of 1024
0000   120 ; locations, each of which corresponds to 1 Mb of physical address space (i.e.
0000   121 ; the index to each PAMM location can also be thought of as bits <29:20> of the
0000   122 ; physical address of the corresponding Mb of address space; the first 512 PAMM
0000   123 ; locations correspond to memory physical address space and the last 512 PAMM
0000   124 ; locations correspond to I/O physical address space).
0000   125 ;
0000   126 ;       Each PAMM location contains a 5-bit type code that identifies what
0000   127 ; entity is referenced in that Mb of physical address space.  The low 4 bits
0000   128 ; of the PAMM type codes are:
0000   129 ;
0000   130 ;       CODE            SELECTS
0000   131 ;       ----            -------
0000   132 ;        0              Memory array slot 0
0000   133 ;        1              Memory array slot 1
0000   134 ;        .                      .
0000   135 ;        .                      .
0000   136 ;        7              Memory array slot 7
0000   137 ;        8              ABUS adapter slot 0
0000   138 ;        9              ABUS adapter slot 1
0000   139 ;        A              ABUS adapter slot 2
0000   140 ;        B              ABUS adapter slot 3
0000   141 ;        C,D,E          unused
0000   142 ;        F              non-existent memory
0000   143 ;
0000   144 ;
0000   145 ;       The high bit of the PAMM type code is the CACHE bit; if set it
0000   146 ; disables the use of cache for that Mb of physical address space.  The console
0000   147 ; will initialize the cache bit to 1 for all valid I/O space addresses (I/O
0000   148 ; space data should not be cached) and to 0 for all valid memory addresses
0000   149 ; (memory data should be cached).
0000   150 ;
0000   151 ; VENUS Memory System
0000   152 ; ------------------
0000   153 ;       VENUS will have 8 memory array slots in its CPU cab; each array slot
0000   154 ; can contain 1 or 4 Mb of memory (depending on whether 64k or 256k memory
0000   155 ; chips are used).  This limits the maximum amount of VENUS physical memory
0000   156 ; to 32Mb.
0000   157 ;
0000   158 ;       The VENUS console will always set up the PAMM so that physical memory
0000   159 ; addresses start at physical address 0 and are contiguous.
0000   160 ;
0000   161 ; FUNCTIONAL DESCRIPTION:
```

```
                                        0000   162 ;
                                        0000   163 ;       Starting at PAMM location 0, search the PAMM for memory type codes.
                                        0000   164 ; For each memory type code found, call BOO$TEST_MEM to test 1 Mb worth of
                                        0000   165 ; memory and record its configuration in the PFN bitmap.
                                        0000   166 ;
                                        0000   167 ; INPUTS:
                                        0000   168 ;
                                        0000   169 ;       R7      - address of System Control Block (SCB)
                                        0000   170 ;       R11     - address of Restart Parameter Block (RPB)
                                        0000   171 ;       SP      - current top of stack
                                        0000   172 ;
                                        0000   173 ; OUTPUTS:
                                        0000   174 ;
                                        0000   175 ;       BOO$TEST_MEM modifies the PFN bitmap to describe all of physical memory
                                        0000   176 ;       described by the PAMM and found to be good (RDS and/or CRD free).
                                        0000   177 ;
                                        0000   178 ;       RPB$L_PFNCNT stores the number of good pages of physical memory.
                                        0000   179 ;
                                        0000   180 ;       Memory descriptor array in RPB is filled in with starting PFN and
                                        0000   181 ;       total number of pages of memory.
                                        0000   182 ;
                                        0000   183 ;       R7,R8,R10,R11,AP,FP preserved.
                                        0000   184 ;       All others may be altered.
                                        0000   185 ;
                                        0000   186 ;--
                                        0000   187
                                        0000   188 CHECKMEM_790::
                                        0000   189 ;
                                        0000   190 ; During the memory locate and test loop, the following registers are used:
                                        0000   191 ;
                                        0000   192 ;       R2      - address of VENUS-specific page test routine
                                        0000   193 ;       R3      - number of pages to test (2048 pages = 1Mb)
                                        0000   194 ;       R4      - physical address of the Mb of memory being tested
                                        0000   195 ;       R5      - PAMM type code
                                        0000   196 ;       R7      - address of SCB
                                        0000   197 ;       R9      - PFN of Mb of memory being tested
                                        0000   198 ;       R11     - address of RPB
                                        0000   199 ;
                                        0000   200
      30 AB   00005800 8F    CA       0000   201         BICL    #<RPB$M_MPM!RPB$M_USEMPM!RPB$M_FINDMEM>, -
                                        0008   202                 RPB$L_BOOTR5(R11)       ; Clear all MA780 specific boot flags.
                  00BC CB    7C       0008   203         CLRQ    RPB$L_MEMDSC(R11)       ; Zero # of pages in this memory, TR#,
                                        000C   204                                         ; and starting PFN.
      04 A7   000000AD'EF    9E       000C   205         MOVAB   PAGE_MCHECK_790+1,4(R7) ; Establish RDS machine check handler.
                                        0014   206                                         ; (+1 to execute on interrupt stack)
                        54    D4       0014   207         CLRL    R4                      ; Start at physical address 0.
            00000042 8F    08    DA    0016   208         MTPR    #CSWP$M_INV,#PR790$_CSWP; Turn off the cache.
               0B 30 AB    10    E0    001D   209         BBS     #RPB$V_CRDTEST, -       ; Should we also remove pages with
                                        0022   210                 RPB$L_BOOTR5(R11), -    ; single bit memory errors?
                                        0022   211                 TRY_NEXT_790            ; Branch if no.
00000045 8F    00000400 8F    DA       0022   212         MTPR    #MDCTL$M_DISECC, -      ; Disable ECC correction (so single
                                        002D   213                 #PR790$_MDCTL           ; bit errors cause machine checks).
                                        002D   214 ;
                                        002D   215 ; Read the PAMM and check the type code.
                                        002D   216 ;
                                        002D   217 TRY_NEXT_790:
            00000041 8F    54    DA    002D   218         MTPR    R4,#PR790$_PAMLOC       ; Request type code from next PAMM loc.
```

```
                55   00000040 8F   DB   0034   219          MFPR    #PR790$_PAMACC,R5       ; Get PAMM type code for this Mb.
              55   55    04   00   EF   003B   220          EXTZV   #PAMM$V_CODE,#PAMM$S_CODE,R5,R5
                                    0040   221                                              ; Isolate type code.
                        0F   55   D1   0040   222          CMPL    R5,#PAMM$C_NEXM         ; Non-existent memory?
                           2D   13   0043   223          BEQL    ALL_DONE_790            ; Yes, we've found all of memory.
                        07   55   D1   0045   224          CMPL    R5,#PAMM$C_MEM7         ; Higher than highest memory type code?
                           1D   14   0048   225          BGTR    DO_NEXT_790             ; Yes, ignore this Mb.
                                    004A   226 ;
                                    004A   227 ; Call BOO$TEST_MEM to test 1 Mb worth of memory.
                                    004A   228 ;
                                    004A   229                                              ; Inputs to BOO$TEST_MEM:
                52   00000086'EF   DE   004A   230          MOVAL   TEST_QUAD_790,R2        ; Address of page test routine.
              53    0800 8F   3C   0051   231          MOVZWL  #2048,R3                ; # of pages to test.
            59   54    15   09   EF   0056   232          EXTZV   #9,#21,R4,R9            ; Starting PFN for this Mb of memory.
                        FFA2'   30   005B   233          BSBW    BOO$TEST_MEM            ; Test 1 Mb worth of memory.
        00BC CB    00000800 8F   C0   005E   234          ADDL2   #2048,RPB$L_MEMDSC(R11) ; Add 1Mb to total page count.
                                    0067   235 ;
                                    0067   236 ; Step to next megabyte of memory and loop.
                                    0067   237 ;
                                    0067   238 DO_NEXT_790:
                54   00100000 8F   C0   0067   239          ADDL2   #^X100000,R4            ; Increment to next Mb boundary.
                     BB 54    1D   E1   006E   240          BBC     #29,R4,TRY_NEXT_790     ; If we're not at the end of memory
                                    0072   241                                              ; address space, go try next Mb.
                                    0072   242 ALL_DONE_790:
             04 A7   00000001'EF   DE   0072   243          MOVAL   UNEXP_MCHK+1,4(R7)      ; Restore normal machine check handler.
                        00C4 CB   D4   007A   244          CLRL    RPB$L_MEMDSC+8(R11)     ; Signal end of memory descriptor list.
                00000045 8F   00   DA   007E   245          MTPR    #0,#PR790$_MDCTL        ; Clear diagnostic bit that turned
                                    0085   246                                              ; single-bit errors into machine checks.
                                    05   0085   247          RSB
```

```
                                0086   249          .SBTTL  TEST_QUAD_790, Test 11/790 Memory
                                0086   250 ;++
                                0086   251 ; Routine TEST_QUAD_790
                                0086   252 ;
                                0086   253 ; FUNCTIONAL DESCRIPTION:
                                0086   254 ;
                                0086   255 ;       Test specified number of quadwords of memory for hard memory errors,
                                0086   256 ; by first writing to and then reading back from the specified location.
                                0086   257 ;
                                0086   258 ; INPUTS:
                                0086   259 ;
                                0086   260 ;       R0      - starting address to test
                                0086   261 ;       R1      - quadword iteration count (64 for one page)
                                0086   262 ;
                                0086   263 ; OUTPUTS:
                                0086   264 ;
                                0086   265 ;       Returns via RSB if page is ok.
                                0086   266 ;       Else error exit via machine check to BOO$PAGE_MCHECK.
                                0086   267 ;       R0,R1 destroyed.
                                0086   268 ;--
                                0086   269
                                0086   270 TEST_QUAD_790:
                51   51   03   78  0086   271          ASHL    #3,R1,R1                ; Convert quad count to byte count.
                     7E   54   7D  008A   272          MOVQ    R4,-(SP)                ; Save R4 and R5.
                     7E   52   7D  008D   273          MOVQ    R2,-(SP)                ; Save R2 and R3.
                     7E   50   7D  0090   274          MOVQ    R0,-(SP)                ; Save R0 and R1.
 60   51   FF 8F   00 8F   00   2C  0093   275          MOVC5   #0,#0,#-1,R1,(R0)       ; Write a bit pattern of all 1's.
                     50   8E   7D  009B   276          MOVQ    (SP)+,R0                ; Get original R0 and R1.
    60   51   00   00 8F   00   2C  009E   277          MOVC5   #0,#0,#0,R1,(R0)        ; Write a bit pattern of all 0's.
                                00A5   278 ;
                                00A5   279 ; If no gross errors occur, then execution continues below.  Otherwise,
                                00A5   280 ; control is transferred to the fault handler PAGE_MCHECK_790.
                                00A5   281 ;
                     52   8E   7D  00A5   282          MOVQ    (SP)+,R2                ; Restore R2 and R3.
                     54   8E   7D  00A8   283          MOVQ    (SP)+,R4                ; Restore R4 and R5.
                          05        00AB   284          RSB
                                00AC   285
                                00AC   286
                                00AC   287 ;
                                00AC   288 ; Handler that gains control if page has a hard memory error.
                                00AC   289 ;
                                00AC   290          .ALIGN  LONG                    ; All handlers longword-aligned.
                                00AC   291 PAGE_MCHECK_790:
          50    0000004A 8F   DB  00AC   292          MFPR    #PR790$_EHSR,R0         ; Get Error Handling Status Register.
                00 50    05   E5  00B3   293          BBCC    #EHSR$V_VMS,R0,10$      ; Clear bit to indicate VMS machine
                                00B7   294 10$:                                    ; check handling complete.
0000004A 8F   50    DA  00B7   295          MTPR    R0,#PR790$_EHSR         ; Write register back.
                FF3F'   31  00BE   296          BRW     BOO$PAGE_MCHECK         ; Goto common page error handler.
                                00C1   297
                                00C1   298          .end
```

```
ALL_DONE_790        00000072 R      02
BOO$PAGE_MCHECK     ********   X    02
BOO$TEST_MEM        ********   X    02
CHECKMEM_790        00000000 RG     02
CSWP$M_INV      =   00000008
DO_NEXT_790         00000067 R      02
EHSR$V_VMS      =   00000005
MDCTL$M_DISECC  =   00000400
PAGE_MCHECK_790     000000AC R      02
PAMM$C_MEM7     =   00000007
PAMM$C_NEXM     =   0000000F
PAMM$S_CODE     =   00000004
PAMM$V_CODE     =   00000000
PR790$_CSWP     =   00000042
PR790$_EHSR     =   0000004A
PR790$_MDCTL    =   00000045
PR790$_PAMACC   =   00000040
PR790$_PAMLOC   =   00000041
RPB$L_BOOTR5    =   00000030
RPB$L_MEMDSC    =   000000BC
RPB$M_FINDMEM   =   00004000
RPB$M_MPM       =   00000800
RPB$M_USEMPM    =   00001000
RPB$V_CRDTEST   =   00000010
TEST_QUAD_790       00000086 R      02
TRY_NEXT_790        0000002D 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	000000C1 (193.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.09	00:00:02.17
Command processing	155	00:00:00.79	00:00:04.82
Pass 1	187	00:00:03.18	00:00:10.10
Symbol table sort	0	00:00:00.23	00:00:00.29
Pass 2	67	00:00:00.86	00:00:02.79
Symbol table output	3	00:00:00.04	00:00:00.15
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	451	00:00:05.23	00:00:20.34

The working set limit was 1350 pages.
15299 bytes (30 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 214 non-local and 1 local symbols.

298 source lines were read in Pass 1, producing 13 object records in Pass 2.
14 pages of virtual memory were used to define 13 macros.

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

Macro library name	Macros defined
_255DUA28:[SHRLIB]790DEF.MLB;1	4
_255DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
_255DUA28:[SYS.OBJ]LIB.MLB;1	2
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	10

304 GETS were required to define 10 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:BTMEM790/OBJ=OBJ$:BTMEM790 MSRC$:BTMEM790/UPDATE=(ENH$:BTMEM790)+EXECML$/LIB+LIB$:BOOTS.MLB/LIB+SHRLIB$:790DEF.MLB/LI
```

CONFIG
LIS

BTMEM855
LIS

BTMEM790
LIS

CONFIGURE
LIS

BOOTDEF
LIS

BOOTIO
LIS

CONFIGMN
LIS

BOOTDRIVR
LIS

BTMEM730
LIS

BTMEM750
LIS

BTMEM780
LIS

BOOTBLOCK
LIS