```
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**BOOTIO

```
BBBBBBBB      000000      000000    TTTTTTTTTT    IIIIII      000000
BBBBBBBB      000000      000000    TTTTTTTTTT    IIIIII      000000
BB      BB  00      00  00      00      TT        II      00      00
BB      BB  00      00  00      00      TT        II      00      00
BB      BB  00      00  00      00      TT        II      00      00
BB      BB  00      00  00      00      TT        II      00      00
BBBBBBBB    00      00  00      00      TT        II      00      00
BBBBBBBB    00      00  00      00      TT        II      00      00
BB      BB  00      00  00      00      TT        II      00      00
BB      BB  00      00  00      00      TT        II      00      00
BB      BB  00      00  00      00      TT        II      00      00      ....
BB      BB  00      00  00      00      TT        II      00      00      ....
BBBBBBBB      000000      000000        TT      IIIIII      000000        ....
BBBBBBBB      000000      000000        TT      IIIIII      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  BOOTIO - BOOTSTRAP FILEREAD IO MODULE
0000     2          .IDENT  'V04-000'
0000     3 ;
0000     4 ;*****************************************************************************
0000     5 ;*                                                                           *
0000     6 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8 ;*  ALL RIGHTS RESERVED.                                                     *
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:     SYSTEM BOOTSTRAPPING
0000    29 ;
0000    30 ; ABSTRACT:
0000    31 ;
0000    32 ;       THIS MODULE PERFORMS LOGICAL BLOCK I/O FOR FILEREAD
0000    33 ;
0000    34 ; ENVIRONMENT:  KERNEL MODE, UNMAPPED, IPL=31
0000    35 ;
0000    36 ; AUTHOR: RICHARD I. HUSTVEDT    , CREATION DATE: 14-APR-78
0000    37 ;
0000    38 ; MODIFIED BY:
0000    39 ;
0000    40 ;       V03-002 KDM0097         Kathleen D. Morse       09-Apr-1984
0000    41 ;                Bias the out of range value by the initial page number.
0000    42 ;
0000    43 ;--
```

```
0000     45 ;
0000     46 ; INCLUDE FILES:
0000     47 ;
0000     48         $IHDDEF                         ; IMAGE HEADER DEFINITIONS
0000     49         $IHSDEF                         ; IMAGE HEADER SYMBOL TABLE DEFS
0000     50         $IHPDEF                         ; IMAGE HEADER PATCH CONTROL DEFS
0000     51         $RPBDEF                         ; DEFINE RESTART PARAMETER BLOCK
0000     52 ;
0000     53 ; MACROS:
0000     54 ;
0000     55 ; Define Memory Size to File Cache Parameter table entry
0000     56 ;
0000     57         .MACRO  MEM_FILE_CACHE MEM_PAGE_CNT,CACHE_PAGE_NUM,CACHE_PAGE_CNT,MAX_PAGE
0000     58         .LONG   MEM_PAGE_CNT-<MEM_PAGE_CNT/10>
0000     59         .WORD   CACHE_PAGE_NUM
0000     60         .WORD   <<CACHE_PAGE_CNT+3>&^C<3>>
0000     61         .LONG   MAX_PAGE
0000     62         .ENDM   MEM_FILE_CACHE
```

```
00000000     64          .PSECT  YFILEREAD,BYTE,EXE
     0000    65
     0000    66          .SBTTL  RDWRTLBN - READ/WRITE LOGICAL BLOCK NUMBER
     0000    67  ;++
     0000    68  ; FUNCTIONAL DESCRIPTION:
     0000    69  ;
     0000    70  ;       THIS ROUTINE READS/WRITES N BYTES FROM/TO THE SPECIFIED
     0000    71  ; LOGICAL BLOCK NUMBER OF THE VOLUME ASSIGNED TO THE SPECIFIED CHANNEL
     0000    72  ;
     0000    73  ; CALLING SEQUENCE:
     0000    74  ;
     0000    75  ;       CALLG   ARGLIST,FIL$RDWRITLBN
     0000    76  ;
     0000    77  ; INPUT PARAMETERS:
     0000    78  ;
     0000    79  ;       CHAN(AP)        =               ;CHANNEL ASSIGNED TO THE VOLUME TO READ
     0000    80  ;       LBN(AP)         =               ;LOGICAL BLOCK NUMBER TO READ
     0000    81  ;       BUFADR(AP)      =               ;ADDRESS OF BUFFER TO READ INTO
     0000    82  ;       IOFUNC(AP)      =               ;I/O FUNCTION CODE
     0000    83  ;       BYTCNT(AP)      =               ;NUMBER OF BYTES TO TRANSFER
     0000    84  ;
     0000    85  ; IMPLICIT INPUTS:
     0000    86  ;
     0000    87  ;       NONE
     0000    88  ;
     0000    89  ; OUTPUT PARAMETERS:
     0000    90  ;
     0000    91  ;       R0 = SYSTEM STATUS CODE
     0000    92  ;
     0000    93  ; IMPLICIT OUTPUTS:
     0000    94  ;
     0000    95  ;       NONE
     0000    96  ;
     0000    97  ; COMPLETION CODES:
     0000    98  ;
     0000    99  ;       NONE
     0000   100  ;
     0000   101  ; SIDE EFFECTS:
     0000   102  ;
     0000   103  ;       NONE
     0000   104  ;
     0000   105  ; EQUATED SYMBOLS:
     0000   106  ;
     0C00   107  ;    OFFSETS FROM AP
     0000   108  ;
00000004  0000   109          CHAN            =       4       ;CHANNEL TO WHICH VOLUME IS ASSIGNED
00000008  0000   110          LBN             =       8       ;LOGICAL BLOCK NUMBER
0000000C  0000   111          BUFADR          =       12      ;BUFFER ADDRESS TO READ INTO
00000010  0000   112          IOFUNC          =       16      ;FUNCTION CODE FOR THE QIO
00000014  0000   113          BYTCNT          =       20      ;NUMBER OF BYTES TO TRANSFER
          0000   114  ;
          0000   115  ;--
          0000   116
          0000   117  FIL$RDWRTLBN::
     0000 0000   118          .WORD   0
   04 AC   DD 0002   119          PUSHL   CHAN(AP)                ; ADDRESS OF RPB
50    6E   D0 0005   120          MOVL    (SP),R0                 ; GET ADDRESS OF RPB
```

```
                50    34 A0    DO  0008    121          MOVL    RPB$L_IOVEC(R0),R0      ; GET POINTER TO I/O ROUTINE VECTOR
                         00    DD  000C    122          PUSHL   #0                      ; SET MODE TO PHYSICAL ADDRESS
                      10 AC    DD  000E    123          PUSHL   IOFUNC(AP)              ; SET FUNCTION
                      08 AC    DD  0011    124          PUSHL   LBN(AP)                 ; LOGICAL BLOCK NUMBER
                      14 AC    DD  0014    125          PUSHL   BYTCNT(AP)              ; SET NUMBER OF BYTES
                      0C BC    DF  0017    126          PUSHAL  @BUFADR(AP)             ; SET BUFFER ADDRESS
          00 B040      06    FB  001A    127          CALLS   #6,@(R0)[R0]            ; CALL BOOTSTRAP DRIVER
                               04  001F    128          RET
```

```
                        .SBTTL  BOO$CACHE_INIT - INIT FILEREAD CACHE
                ;++
                ;
                ; Functional description:
                ;
                ;       This routine establishes a desired FILEREAD cache size and
                ;       base address according to the size of memory.  It finds
                ;       good contiguous pages at or near the desired place and
                ;       calls the FIL$CACHE_INIT routine to initialize the cache.
                ;       The routine is further divided into two pieces: one to do
                ;       cache allocation, and one to do the actual mount and open.
                ;       This is necessary for VMB needs to allocate the cache long
                ;       before it is ready to accept IO to the device.
                ;
                ; Calling Sequence:
                ;
                ;       BSBW    BOO$CACHE_INIT
                ;
                ; Inputs:
                ;
                ;       R11                     - RPB base address
                ;       RPB$L_PFNCNT(R11)       - actual number of good pages in memory
                ;       RPB$Q_PFNMAP+4(R11)     - base address of PFN bitmap
                ;
                ; Implicit inputs:
                ;
                ;       none
                ;
                ; Outputs:
                ;
                ;       R0-R4 altered
                ;       FIL$GQ_CACHE set up with size and address of cache
                ;
                ; Implicit outputs:
                ;
                ;--
                ;
                ; Table of memory sizes to file cache parameters
                ;
                ; NOTE: If this table is modified, a corresponding table in VMB around
                ;       label MEM_TAB should be checked for consitency.
                ;
0020    172 MEM_CACHE_TABLE:
0020    173             MEM_FILE_CACHE 16384,2048,64,4096       ; More than 8 megabyte
002C    174             MEM_FILE_CACHE  8192,1024,64,2048       ; More than 4 megabyte
0038    175             MEM_FILE_CACHE  4096, 640,64,1024       ; More than 2 megabyte
0044    176             MEM_FILE_CACHE  2048, 512,64, 768       ; More than 1 megabyte
0050    177             MEM_FILE_CACHE  1024, 256,32, 512       ; More than 512k bytes
005C    178             MEM_FILE_CACHE   512, 256,16, 256       ; More than 256k bytes
0068    179             MEM_FILE_CACHE   384, 256, 8, 192       ; More than 192k bytes
0074    180             MEM_FILE_CACHE   256, 128, 4, 128       ; More than 128k bytes
0080    181             MEM_FILE_CACHE     0,   0, 0,   0       ;
```

```
                        008C    183 ;
                        008C    184 ; BOO$CACHE_ALLOC - The piece that does the allocation.
                        008C    185 ;
                        008C    186 ; Outputs:
                        008C    187 ;       FIL$GQ_CACHE filled in with size/address in blocks
                        008C    188 ;
                        008C    189 BOO$CACHE_ALLOC::
                  55    DD  008C    190         PUSHL   R5                              ; Save a register
          0000'CF       7C  008E    191         CLRQ    W^FIL$GQ_CACHE                  ; Assume no cache available
       50   8B AF       DE  0092    192         MOVAL   B^MEM_CACHE_TABLE,R0            ; Adr of memory size to cache params tbl
            51 80       7D  0096    193 10$:    MOVQ    (R0)+,R1                        ; Get the next table entry
               1F       13  0099    194         BEQL    20$                             ; Branch if memory too small for cache
            55 80       D0  009B    195         MOVL    (R0)+,R5                        ; Max page
            51 4C AB    D1  009E    196         CMPL    RPB$L_PFNCNT(R11),R1            ; More memory than this entry?
               F2       19  00A2    197         BLSS    10$                             ; Branch if not, get next one
            50 52       3C  00A4    198         MOVZWL  R2,R0                           ; Starting relative bit (page) in PFNMAP
51  52    F0 8F       78  00A7    199         ASHL    #-16,R2,R1                      ; Count of bits (pages) to look for
54  51    FF 8F       78  00AC    200         ASHL    #-1,R1,R4                       ; Settle for half if can't find all
               37       10  00B1    201         BSBB    BOO$ALLOC_PAGES                 ; Go get the pages
               05       19  00B3    202         BLSS    20$                             ; Failed
     0000'CF   52       7D  00B5    203         MOVQ    R2,W^FIL$GQ_CACHE               ; Success, record the values
               55 8ED0  00BA    204 20$:    POPL    R5                              ; Restore a register
                  05    00BD    205         RSB
                        00BE    206
                        00BE    207 ;
                        00BE    208 ; BOO$CACHE_INIT - Full routine to both allocate and open the cache
                        00BE    209 ;
                        00BE    210 BOO$CACHE_INIT::
               CC       10  00BE    211         BSBB    BOO$CACHE_ALLOC                 ; Allocate the cache
                        00C0    212                                                     ; Fall thru to finish
                        00C0    213
                        00C0    214 ;
                        00C0    215 ; BOO$CACHE_OPEN - Actually mount the device and fill the cache
                        00C0    216 ;
                        00C0    217 BOO$CACHE_OPEN::
       52       0000'CF  D0  00C0    218         MOVL    W^FIL$GQ_CACHE,R2               ; Pick up size
                   22    13  00C5    219         BEQL    10$                             ; Zero length implies none
            5E     04    C2  00C7    220         SUBL    #4,SP                           ; Location to store channel
            50     5E    D0  00CA    221         MOVL    SP,R0                           ; Address to store channel
       7E   52     02    C3  00CD    222         SUBL3   #2,R2,-(SP)                     ; Blocks in directory LBN cache
                   00'   DD  00D1    223         PUSHL   S^#<<1024-FIL$C_SIZE>/FIL$C_DIR_SIZE> ; No. of dir cache entries
7E     0004'CF     09    78  00D3    224         ASHL    #9,W^FIL$GQ_CACHE+4,-(SP) ; Byte address from page number
       7E   52     09    78  00D9    225         ASHL    #9,R2,-(SP)                     ; Size of cache in bytes
                   7E    D4  00DD    226         CLRL    -(SP)                           ; Null device name string descriptor
                   50    DD  00DF    227         PUSHL   R0                              ; Address to store channel
            0000'CF    06    FB  00E1    228         CALLS   #6,W^FIL$CACHE_INIT             ; Init the FIL$OPENFILE cache
                                00E6    229                                             ; descriptor returned in FIL$GQ_CACHE
            5E     04    C0  00E6    230         ADDL    #4,SP                           ; Clean off channel
                   05    00E9    231 10$:    RSB
                        00EA    232
                        00EA    233 ;
                        00EA    234 ; BOO$ALLOC_PAGES - Find a run of contiguous, good pages
                        00EA    235 ;
                        00EA    236 ; Inputs:
                        00EA    237 ;       R0 - Page to start at
                        00EA    238 ;       R1 - Number of pages needed
                        00EA    239 ;       R4 - Number willing to settle for
```

```
                     00EA   240 ;       R5 - Maximum page
                     00EA   241 ; Outputs:
                     00EA   242 ;       CC - Status (BLSS to an error routine)
                     00EA   243 ;       R2 - Number found
                     00EA   244 ;       R3 - Starting page number
                     00EA   245 ;
                     00EA   246 BOO$ALLOC_PAGES::
 52   5B   17   9C  00EA   247           ROTL    #<32-9>,R11,R2          ; PFN of the RPB
      50   52   C0  00EE   248           ADDL    R2,R0                   ; Convert relative PFN to absolute
      53   50   D0  00F1   249           MOVL    R0,R3                   ; Make a copy of starting bit
      55   52   C0  00F4   250           ADDL    R2,R5                   ; Convert max relative PFN to absolute
      50   55   D1  00F7   251 30$:      CMPL    R5,R0                   ; Less than max page
           1E   19  00FA   252           BLSS    50$                     ; No, failure
0D 48 BB   50   E0  00FC   253           BBS     R0,@RPB$Q_PFNMAP+4(R11),40$ ; Branch if this is a good page
 52   50   53   C3  0101   254           SUBL3   R3,R0,R2                ; Count of bits (pages) found
      54   52   D1  0105   255           CMPL    R2,R4                   ; Found enough?
           10   18  0108   256           BGEQ    50$                     ; Branch if yes
 53   50   01   C1  010A   257           ADDL3   #1,R0,R3                ; No, reset starting base
           50   D6  010E   258 40$:      INCL    R0                      ; Next bit (page)
        E4 51   F5  0110   259           SOBGTR  R1,30$                  ; Branch if more to check
 52   50   53   C3  0113   260           SUBL3   R3,R0,R2                ; Count of bits (pages) found
      54   52   D1  0117   261           CMPL    R2,R4                   ; Found enough?
           05   11  011A   262 50$:      RSB                             ; Return (Status in CC)
```

```
                     011B    264          .SBTTL  BOO$IMAGE_ATT - Get image attributes from image header
                     011B    265 ;++
                     011B    266 ; Functional Description:
                     011B    267 ;
                     011B    268 ;       BOO$IMAGE_ATT returns to the caller some attributes of the image
                     011B    269 ;
                     011B    270 ; Calling Sequence:
                     011B    271 ;
                     011B    272 ;       BSBW    BOO$IMAGE_ATT
                     011B    273 ;
                     011B    274 ; Inputs:
                     011B    275 ;
                     011B    276 ;       R2 = Size of file in blocks
                     011B    277 ;       R3 = Address of image header block (first one only)
                     011B    278 ;
                     011B    279 ; Outputs:
                     011B    280 ;
                     011B    281 ;       R1 = Number of image header blocks at the front of the image
                     011B    282 ;       R2 = Size of image in blocks excluding the blocks at the end
                     011B    283 ;            containing local symbols, global symbols, or patch text
                     011B    284 ;
                     011B    285 ;--
                     011B    286
                     011B    287 BOO$IMAGE_ATT::
   50    04 A3   3C  011B    288          MOVZWL  IHD$W_SYMDBGOFF(R3),R0  ; ANY SYMBOL TABLE INFORMATION?
         0D  13  011F    289          BEQL    20$                     ; BRANCH IF NOT
   51    6043    9E  0121    290          MOVAB   IHS$L_DSTVBN(R0)[R3],R1 ; ADR OF 1ST VBN IN DEBUG SYMBOL TABLE
         19  10  0125    291          BSBB    40$                     ; PROCESS IT
51   04 A043    9E  0127    292          MOVAB   IHS$L_GSTVBN(R0)[R3],R1 ; ADR OF 1ST VBN IN GLOBAL SYMBOL TABLE
         12  10  012C    293          BSBB    40$                     ; PROCESS IT
   50    08 A3   3C  012E    294 20$:     MOVZWL  IHD$W_PATCHOFF(R3),R0   ; ANY PATCH CONTROL INFORMATION?
         07  13  0132    295          BEQL    30$                     ; BRANCH IF NOT
51   20 A043    9E  0134    296          MOVAB   IHP$L_PATCOMTXT(R0)[R3],R1 ; ADR OF 1ST VBN OF PATCH COMMAND TEXT
         05  10  0139    297          BSBB    40$                     ; PROCESS IT
   51    10 A3   9A  013B    298 30$:     MOVZBL  IHD$B_HDRBLKCNT(R3),R1  ; GET IMAGE HEADER BLOCK COUNT
         05  013F    299          RSB
                     0140    300 ;
                     0140    301 ; SEE IF VBN IS NON ZERO AND THEN IF IT IS SMALLER THAN THE CURRENT SMALLEST
                     0140    302 ;
51   61    01    C3  0140    303 40$:     SUBL3   #1,(R1),R1              ; FETCH VBN - 1
         08  19  0144    304          BLSS    50$                     ; BRANCH IF NO VBN IS PRESENT
      51    52    D1  0146    305          CMPL    R2,R1                   ; IS IT SMALLER THAN THE CURRENT ONE
         03  15  0149    306          BLEQ    50$                     ; BRANCH IF NOT
      52    51    D0  014B    307          MOVL    R1,R2                   ; YES, USE IT
         05  014E    308 50$:     RSB
```

```
                                014F    310          .SBTTL  SYS$ASSIGN, Dummy assign device system service
                                014F    311
                                014F    312 ;++
                                014F    313 ;
                                014F    314 ; Functional description:
                                014F    315 ;
                                014F    316 ;       SYS$ASSIGN is a dummy routine to satisfy the requirements of
                                014F    317 ;       FIL$OPENFILE.
                                014F    318 ;
                                014F    319 ; Inputs:
                                014F    320 ;
                                014F    321 ;       CHAN(AP)        - address at which to return channel
                                014F    322 ;
                                014F    323 ; Implicit inputs:
                                014F    324 ;
                                014F    325 ;       The label BOO$GL_RPBBASE contains the physical address of the RPB.
                                014F    326 ;
                                014F    327 ; Outputs:
                                014F    328 ;
                                014F    329 ;       R0      - success status code
                                014F    330 ;
                                014F    331 ; Implicit outputs:
                                014F    332 ;
                                014F    333 ;       The channel returned is not a channel. It is instead the base
                                014F    334 ;       address of the RPB.
                                014F    335 ;
                                014F    336 ;--
                                014F    337
                   00000008     014F    338 CHAN    = 8
                                014F    339
                                014F    340 SYS$ASSIGN::                            ; Dummy system service.
                        0000    014F    341          .WORD   0
                                0151    342
08 BC    0000'CF    D0          0151    343          MOVL    W^BOO$GL_RPBBASE,@CHAN(AP) ; Store RPB address as channel.
         50   00'   D0          0157    344          MOVL    S^#SS$_NORMAL,R0           ; Return success status.
                    04          015A    345          RET                                ; Return to caller.
```

```
                                                                I  6
                     015B   347          .SBTTL  Common Globals for VMB and SYSBOOT
                     015B   348 ;
                     015B   349 ; The following globals are common to VMB and SYSBOOT and are
                     015B   350 ; defined here to avoid replicate definitions.
                     015B   351 ;
                     015B   352 FIL$GT_DDSTRING::                            ; Default directory string.
5D 45 58 45 53 59 53 5B 00'  015B   353          .ASCIC  /[SYSEXE]/
                       08    015B
                     0164   354 FIL$GT_DDDEV::                               ; Default device name
                       00    0164   355          .BYTE   0                   ; Null ASCIC string
                     0165   356
                     0165   357          .END
```

```
BOO$ALLOC_PAGES                 000000EA RG     02
BOO$CACHE_ALLOC                 0000008C RG     02
BOO$CACHE_INIT                  000000BE RG     02
BOO$CACHE_OPEN                  000000C0 RG     02
BOO$GL_RPBBASE                  ********   X    02
BOO$IMAGE_ATT                   0000011B RG     02
BUFADR                        = 0000000C
BYTCNT                        = 00000014
CHAN                          = 00000008
FIL$CACHE_INIT                  ********   X    02
FIL$C_DIR_SIZE                  ********   X    02
FIL$C_SIZE                      ********   X    02
FIL$GQ_CACHE                    ********   X    02
FIL$GT_DDDEV                    00000164 RG     02
FIL$GT_DDSTRING                 0000015B RG     02
FIL$RDWRTLBN                    00000000 RG     02
IHD$B_HDRBLKCNT               = 00000010
IHD$W_PATCHOFF                = 00000008
IHD$W_SYMDBGOFF               = 00000004
IHP$L_PATCOMTXT               = 00000020
IHS$L_DSTVBN                  = 00000000
IHS$L_GSTVBN                  = 00000004
IOFUNC                        = 00000010
LBN                           = 00000008
MEM_CACHE_TABLE                 00000020 R      02
OP$_ACBD                      = 0000006F
OP$_ACBF                      = 0000004F
OP$_ACBG                      = 00004FFD
OP$_ACBH                      = 00006FFD
OP$_ADDD2                     = 00000060
OP$_ADDD3                     = 00000061
OP$_ADDF2                     = 00000040
OP$_ADDF3                     = 00000041
OP$_ADDG2                     = 000040FD
OP$_ADDG3                     = 000041FD
OP$_ADDH2                     = 000060FD
OP$_ADDH3                     = 000061FD
OP$_ADDP4                     = 00000020
OP$_ADDP6                     = 00000021
OP$_ASHP                      = 000000F8
OP$_CLRD                      = 0000007C
OP$_CLRF                      = 000000D4
OP$_CLRG                      = 0000007C
OP$_CLRH                      = 00007CFD
OP$_CMPD                      = 00000071
OP$_CMPF                      = 00000051
OP$_CMPG                      = 000051FD
OP$_CMPH                      = 000071FD
OP$_CMPP3                     = 00000035
OP$_CMPP4                     = 00000037
OP$_CRC                       = 0000000B
OP$_CVTBD                     = 0000006C
OP$_CVTBF                     = 0000004C
OP$_CVTBG                     = 00004CFD
OP$_CVTBH                     = 00006CFD
OP$_CVTDB                     = 00000068
OP$_CVTDF                     = 00000076
OP$_CVTDH                     = 000032FD
OP$_CVTDL                     = 0000006A
OP$_CVTDW                     = 00000069
OP$_CVTFB                     = 00000048
OP$_CVTFD                     = 00000056
OP$_CVTFG                     = 000099FD
OP$_CVTFH                     = 000098FD
OP$_CVTFL                     = 0000004A
OP$_CVTFW                     = 00000049
OP$_CVTGB                     = 000048FD
OP$_CVTGF                     = 000033FD
OP$_CVTGH                     = 000056FD
OP$_CVTGL                     = 00004AFD
OP$_CVTGW                     = 000049FD
OP$_CVTHB                     = 000068FD
OP$_CVTHD                     = 0000F7FD
OP$_CVTHF                     = 0000F6FD
OP$_CVTHG                     = 000076FD
OP$_CVTHL                     = 00006AFD
OP$_CVTHW                     = 000069FD
OP$_CVTLD                     = 0000006E
OP$_CVTLF                     = 0000004E
OP$_CVTLG                     = 00004EFD
OP$_CVTLH                     = 00006EFD
OP$_CVTLP                     = 000000F9
OP$_CVTPL                     = 00000036
OP$_CVTPS                     = 00000008
OP$_CVTPT                     = 00000024
OP$_CVTRDL                    = 0000006B
OP$_CVTRFL                    = 0000004B
OP$_CVTRGL                    = 00004BFD
OP$_CVTRHL                    = 00006BFD
OP$_CVTSP                     = 00000009
OP$_CVTTP                     = 00000026
OP$_CVTWD                     = 0000006D
OP$_CVTWF                     = 0000004D
OP$_CVTWG                     = 00004DFD
OP$_CVTWH                     = 00006DFD
OP$_DIVD2                     = 00000066
OP$_DIVD3                     = 00000067
OP$_DIVF2                     = 00000046
OP$_DIVF3                     = 00000047
OP$_DIVG2                     = 000046FD
OP$_DIVG3                     = 000047FD
OP$_DIVH2                     = 000066FD
OP$_DIVH3                     = 000067FD
OP$_DIVP                      = 00000027
OP$_EDITPC                    = 00000038
OP$_EMODD                     = 00000074
OP$_EMODF                     = 00000054
OP$_EMODG                     = 000054FD
OP$_EMODH                     = 000074FD
OP$_MATCHC                    = 00000039
OP$_MNEGD                     = 00000072
OP$_MNEGF                     = 00000052
OP$_MNEGG                     = 000052FD
OP$_MNEGH                     = 000072FD
```

```
OP$_MOVD                        = 00000070
OP$_MOVF                        = 00000050
OP$_MOVG                        = 000050FD
OP$_MOVH                        = 000070FD
OP$_MOVP                        = 00000034
OP$_MOVTC                       = 0000002E
OP$_MOVTUC                      = 0000002F
OP$_MULD2                       = 00000064
OP$_MULD3                       = 00000065
OP$_MULF2                       = 00000044
OP$_MULF3                       = 00000045
OP$_MULG2                       = 000044FD
OP$_MULG3                       = 000045FD
OP$_MULH2                       = 000064FD
OP$_MULH3                       = 000065FD
OP$_MULP                        = 00000025
OP$_POLYD                       = 00000075
OP$_POLYF                       = 00000055
OP$_POLYG                       = 000055FD
OP$_POLYH                       = 000075FD
OP$_SCANC                       = 0000002A
OP$_SKPC                        = 0000003B
OP$_SPANC                       = 0000002B
OP$_SUBD2                       = 00000062
OP$_SUBD3                       = 00000063
OP$_SUBF2                       = 00000042
OP$_SUBF3                       = 00000043
OP$_SUBG2                       = 000042FD
OP$_SUBG3                       = 000043FD
OP$_SUBH2                       = 000062FD
OP$_SUBH3                       = 000063FD
OP$_SUBP4                       = 00000022
OP$_SUBP6                       = 00000023
OP$_TSTD                        = 00000073
OP$_TSTF                        = 00000053
OP$_TSTG                        = 000053FD
OP$_TSTH                        = 000073FD
RPB$L_IOVEC                     = 00000034
RPB$L_PFNCNT                    = 0000004C
RPB$Q_PFNMAP                    = 00000044
SS$_NORMAL                        ********   X   02
SYS$ASSIGN                        0000014F  RG   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
YFILEREAD	00000165 (357.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.10	00:00:00.31
Command processing	110	00:00:00.72	00:00:01.86
Pass 1	406	00:00:11.31	00:00:20.88
Symbol table sort	0	00:00:00.79	00:00:01.02
Pass 2	76	00:00:03.69	00:00:07.01
Symbol table output	19	00:00:00.15	00:00:00.20
Psect synopsis output	1	00:00:00.03	00:00:00.34
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	644	00:00:16.80	00:00:31.63

The working set limit was 1500 pages.
49198 bytes (97 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 559 non-local and 10 local symbols.
3109 source lines were read in Pass 1, producing 14 object records in Pass 2.
139 pages of virtual memory were used to define 138 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	4
TOTALS (all libraries)	8

613 GETS were required to define 8 macros.

There were no errors, warnings or information messages.

```
MACRO/LIS=LIS$:BOOTIO/OBJ=OBJ$:BOOTIO MASD$:[EMULAT.SRC]MISSING/UPDATE=(MASD$:[EMULAT.ENH]MISSING)+MASD$:[BOOTS.SRC]BOOTIO/UPDATE=(M
```

CONFIG
LIS

BTMEM855
LIS

BTMEM790
LIS

CONFIGURE
LIS

BOOTDEF
LIS

BOOTIO
LIS

CONFIGMN
LIS

BOOTDRIVR
LIS

BTMEM730
LIS

BTMEM750
LIS

BTMEM780
LIS

BOOTBLOCK
LIS