

```
UUU      UUU VVV      VVV      111      RRRRRRRRRRRR      0000000000      MMM      MMM
UUU      UUU VVV      VVV      111      RRRRRRRRRRRR      0000000000      MMM      MMM
UUU      UUU VVV      VVV      111      RRRRRRRRRRRR      0000000000      MMM      MMM
UUU      UUU VVV      VVV      111111      RRR      RRR      000      000      MMMMMM      MMMMMM
UUU      UUU VVV      VVV      111111      RRR      RRR      000      000      MMMMMM      MMMMMM
UUU      UUU VVV      VVV      111111      RRR      RRR      000      000      MMMMMM      MMMMMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUU      UUU VVV      VVV      111      RRR      RRR      000      000      MMM      MMM      MMM
UUUUUUUUUUUUUUUUUU      VVV      1111111111      RRR      RRR      0000000000      MMM      MMM
UUUUUUUUUUUUUUUUUU      VVV      1111111111      RRR      RRR      0000000000      MMM      MMM
UUUUUUUUUUUUUUUUUU      VVV      1111111111      RRR      RRR      0000000000      MMM      MMM
```

[illegible]

(1)	72	RDWRTLBN - READ/WRITE LOGICAL BLOCK NUMBER
(1)	136	BOO\$CACHE_INIT - INIT FILEREAD CACHE
(1)	274	BOO\$IMAGE_ATT - Get image attributes from image header
(1)	359	Common Globals for VMB and SYSBOOT

```

00000001 0000 1      BOOT_UV1_SWITCH = 1      ; Build Micro-VAX I bootstrap emulator
00000001 0000 2      PQ      == 1
0000 0000 1      .NLIST CND
0000 0000 6      .TITLE BOOTIOUV1 - BOOTSTRAP FILEREAD IO MODULE FOR MICRO-VAX I
0000 0000 8      .IDENT 'V03-002'
0000 0000 9      :
0000 0000 10     :*****
0000 0000 11     :*
0000 0000 12     :*  COPYRIGHT (c) 1978, 1980, 1982, 1983 BY
0000 0000 13     :*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 0000 14     :*  ALL RIGHTS RESERVED.
0000 0000 15     :*
0000 0000 16     :*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 0000 17     :*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 0000 18     :*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 0000 19     :*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 0000 20     :*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 0000 21     :*  TRANSFERRED.
0000 0000 22     :*
0000 0000 23     :*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 0000 24     :*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 0000 25     :*  CORPORATION.
0000 0000 26     :*
0000 0000 27     :*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 0000 28     :*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 0000 29     :*
0000 0000 30     :*
0000 0000 31     :*****
0000 0000 32     :
0000 0000 33     :++
0000 0000 34     : FACILITY:      SYSTEM BOOTSTRAPPING
0000 0000 35     :
0000 0000 36     : ABSTRACT:
0000 0000 37     :
0000 0000 38     :      THIS MODULE PERFORMS LOGICAL BLOCK I/O FOR FILEREAD
0000 0000 39     :
0000 0000 40     : ENVIRONMENT:  KERNEL MODE, UNMAPPED, IPL=31
0000 0000 41     :
0000 0000 42     : AUTHOR: RICHARD I. HUSTVEDT   , CREATION DATE: 14-APR-78
0000 0000 43     :
0000 0000 44     : MODIFIED BY:
0000 0000 45     :
0000 0000 46     :      V03-002 KDM0070      Kathleen D. Morse      11-Aug-1983
0000 0000 47     :      Create Micro-VAX I version, via assembly switch.
0000 0000 48     :
0000 0000 49     :--

```


BOOTIOUV1
V03-002

- BOOTSTRAP FILEREAD IO MODULE FOR MICRO 10-AUG-1984 18:05:02 VAX/VMS Macro V04-00 Page 2
9-JUL-1984 11:36:49 [GAMACHE.UV1ROM.VMS]BOOTIO.MAR;1 (1)

```
0000 51 :  
0000 52 : INCLUDE FILES:  
0000 53 :  
0000 54 $IHDEF ; IMAGE HEADER DEFINITIONS  
0000 55 $IHSDEF ; IMAGE HEADER SYMBOL TABLE DEFS  
0000 56 $IHPDEF ; IMAGE HEADER PATCH CONTROL DEFS  
0000 57 $RPBDEF ; DEFINE RESTART PARAMETER BLOCK  
0000 58 :  
0000 59 : MACROS:  
0000 60 :  
0000 61 : Define Memory Size to File Cache Parameter table entry  
0000 62 :  
0000 63 .MACRO MEM_FILE_CACHE MEM_PAGE_CNT,CACHE_PAGE_NUM,CACHE_PAGE_CNT,MAX_PAGE  
0000 64 .LONG MEM_PAGE_CNT-<MEM_PAGE_CNT/10>  
0000 65 .WORD CACHE_PAGE_NUM  
0000 66 .WORD <<CACHE_PAGE_CNT+3>&^C<3>>  
0000 67 .LONG MAX_PAGE  
0000 68 .ENDM MEM_FILE_CACHE
```



```

00000000 70      .PSECT  YFILEREAD,BYTE,EXE
0000      71
0000      72      .SBTTL  RDWRTLBN - READ/WRITE LOGICAL BLOCK NUMBER
0000      73      ;++
0000      74      ; FUNCTIONAL DESCRIPTION:
0000      75
0000      76      ;     THIS ROUTINE READS/Writes N BYTES FROM/TO THE SPECIFIED
0000      77      ; LOGICAL BLOCK NUMBER OF THE VOLUME ASSIGNED TO THE SPECIFIED CHANNEL
0000      78
0000      79      ; CALLING SEQUENCE:
0000      80
0000      81      ;     CALLG  ARGLIST,FIL$RDWRITLBN
0000      82
0000      83      ; INPUT PARAMETERS:
0000      84
0000      85      ;     CHAN(AP)      =                ;CHANNEL ASSIGNED TO THE VOLUME TO READ
0000      86      ;     LBN(AP)      =                ;LOGICAL BLOCK NUMBER TO READ
0000      87      ;     BUFADR(AP)   =                ;ADDRESS OF BUFFER TO READ INTO
0000      88      ;     IOFUNC(AP)  =                ;I/O FUNCTION CODE
0000      89      ;     BYTCNT(AP)  =                ;NUMBER OF BYTES TO TRANSFER
0000      90
0000      91      ; IMPLICIT INPUTS:
0000      92
0000      93      ;     NONE
0000      94
0000      95      ; OUTPUT PARAMETERS:
0000      96
0000      97      ;     RO = SYSTEM STATUS CODE
0000      98
0000      99      ; IMPLICIT OUTPUTS:
0000     100
0000     101      ;     NONE
0000     102
0000     103      ; COMPLETION CODES:
0000     104
0000     105      ;     NONE
0000     106
0000     107      ; SIDE EFFECTS:
0000     108
0000     109      ;     NONE
0000     110
0000     111      ; EQUATED SYMBOLS:
0000     112
0000     113      ;     OFFSETS FROM AP
0000     114      ;
00000004 0000     115      ;     CHAN      =      4      ;CHANNEL TO WHICH VOLUME IS ASSIGNED
00000008 0000     116      ;     LBN      =      8      ;LOGICAL BLOCK NUMBER
0000000C 0000     117      ;     BUFADR   =     12     ;BUFFER ADDRESS TO READ INTO
00000010 0000     118      ;     IOFUNC   =     16     ;FUNCTION CODE FOR THE QIO
00000014 0000     119      ;     BYTCNT   =     20     ;NUMBER OF BYTES TO TRANSFER
0000     120      ;
0000     121      ;--
0000     122
0000     123      FIL$RDWRTLBN::
0000     124      ;     .WORD  0
0000     125      ;     PUSHL  CHAN(AP)      ; ADDRESS OF RPB
0000     126      ;     MOVL   (SP),RO      ; GET ADDRESS OF RPB

```


BOOTIOUV1
V03-002

D 6
- BOOTSTRAP FILEREAD IO MODULE FOR MICRO 10-AUG-1984 18:05:02 VAX/VMS Macro V04-00 Page 4
RDWRTLBN - READ/WRITE LOGICAL BLOCK NUMB 9-JUL-1984 11:36:49 [GAMACHE.UV1ROM.VMS]BOOTIO.MAR;1 (1)

50	34	A0	D0	0008	127	MOVL	RPBSL_IOVEC(R0),R0	:	GET POINTER TO I/O ROUTINE VECTOR
		00	DD	000C	128	PUSHL	#0	:	SET MODE TO PHYSICAL ADDRESS
	10	AC	DD	000E	129	PUSHL	IOFUNC(AP)	:	SET FUNCTION
	08	AC	DD	0011	130	PUSHL	LBN(AP)	:	LOGICAL BLOCK NUMBER
	14	AC	DD	0014	131	PUSHL	BYTCNT(AP)	:	SET NUMBER OF BYTES
	0C	BC	DF	0017	132	PUSHAL	@BUFADR(AP)	:	SET BUFFER ADDRESS
00	B040	06	FB	001A	133	CALLS	#6,@(R0)[R0]	:	CALL BOOTSTRAP DRIVER
			04	001F	134	RET			


```

0020 136      .SBTTL BOO$CACHE_INIT - INIT FILEREAD CACHE
0020 137      :++
0020 138      :
0020 139      : Functional description:
0020 140      :
0020 141      :     This routine establishes a desired FILEREAD cache size and
0020 142      :     base address according to the size of memory. It finds
0020 143      :     good contiguous pages at or near the desired place and
0020 144      :     calls the FIL$CACHE_INIT routine to initialize the cache.
0020 145      :     The routine is further divided into two pieces: one to do
0020 146      :     cache allocation, and one to do the actual mount and open.
0020 147      :     This is necessary for VMB needs to allocate the cache long
0020 148      :     before it is ready to accept IO to the device.
0020 149      :
0020 150      : Calling Sequence:
0020 151      :
0020 152      :     BSBW      BOO$CACHE_INIT
0020 153      :
0020 154      : Inputs:
0020 155      :
0020 156      :     R11          - RPB base address
0020 157      :     RPB$$_PFNCNT(R11) - actual number of good pages in memory
0020 158      :     RPB$$_PFNMAP+4(R11) - base address of PFN bitmap
0020 159      :
0020 160      : Implicit inputs:
0020 161      :
0020 162      :     none
0020 163      :
0020 164      : Outputs:
0020 165      :
0020 166      :     R0-R4 altered
0020 167      :     FIL$GQ_CACHE set up with size and address of cache
0020 168      :
0020 169      : Implicit outputs:
0020 170      :
0020 171      : --
0020 172      :
0020 173      : Table of memory sizes to file cache parameters
0020 174      :
0020 175      : NOTE: If this table is modified, a corresponding table in VMB around
0020 176      : label MEM_TAB should be checked for consistency.
0020 177      :
0020 178      : MEM_CACHE TABLE:
0020 183      : MEM_FILE_CACHE 4096, 640, 64, 1024      : More than 2 megabyte
0020 184      : MEM_FILE_CACHE 2048, 512, 64, 768          : More than 1 megabyte
0038 185      : MEM_FILE_CACHE 1024, 256, 32, 512      : More than 512k bytes
0044 186      : MEM_FILE_CACHE 512, 256, 16, 256        : More than 256k bytes
0050 191      : MEM_FILE_CACHE 0, 0, 0, 0              :

```



```

005C 193 :
005C 194 : BOO$CACHE_ALLOC - The piece that does the allocation.
005C 195 :
005C 196 : Outputs:
005C 197 :     FIL$GQ_CACHE filled in with size/address in blocks
005C 198 :
005C 199 BOO$CACHE_ALLOC::
50 0000'CF 55 DD 005C 200     PUSHL R5 ; Save a register
51 BB AF 7C 005E 201     CLRQ W^FIL$GQ_CACHE ; Assume no cache available
51 80 7D 0062 202     MOVAL B^MEM_CACHE_TABLE,R0 ; Adr of memory size to cache params tbl
51 1F 13 0066 203 10$:     MOVQ (R0)+,R1 ; Get the next table entry
51 80 D0 0069 204     BEQL 20$ ; Branch if memory too small for cache
51 4C AB D1 006B 205     MOVL (R0)+,R5 ; Max page
51 F2 19 006E 206     CMPL RPB$$_PFNCNT(R11),R1 ; More memory than this entry?
51 52 3C 0072 207     BLSS 10$ ; Branch if not, get next one
51 52 8F 78 0074 208     MOVZWL R2,R0 ; Starting relative bit (page) in PFNMAP
51 51 FF 8F 78 0077 209     ASHL #-16,R2,R1 ; Count of bits (pages) to look for
54 51 37 10 0081 210     ASHL #-1,R1,R4 ; Settle for half if can't find all
0000'CF 05 19 0083 211     BSBB BOO$ALLOC_PAGES ; Go get the pages
51 52 7D 0085 212     BLSS 20$ ; Failed
51 8ED0 008A 213     MOVQ R2,W^FIL$GQ_CACHE ; Success, record the values
51 05 008D 214 20$:     POPL R5 ; Restore a register
008E 215     RSB
008E 216 :
008E 217 : BOO$CACHE_INIT - Full routine to both allocate and open the cache
008E 218 :
008E 219 : BOO$CACHE_INIT::
CC 10 008E 220     BSBB BOO$CACHE_ALLOC ; Allocate the cache
0090 221     ; Fall thru to finish
0090 222 :
0090 223 : BOO$CACHE_OPEN - Actually mount the device and fill the cache
0090 224 :
0090 225 : BOO$CACHE_OPEN::
52 0000'CF D0 0090 226     MOVL W^FIL$GQ_CACHE,R2 ; Pick up size
52 22 13 0095 227     BEQL 10$ ; Zero length implies none
52 5E 04 C2 0097 228     SUBL #4,SP ; Location to store channel
52 50 5E D0 009A 229     MOVL SP,R0 ; Address to store channel
7E 52 02 C3 009D 230     SUBL3 #2,R2,-(SP) ; Blocks in directory LBN cache
7E 0004'CF 00' DD 00A1 231     PUSHL S^#<<1024-FIL$C_SIZE>/FIL$C_DIR_SIZE>; No. of dir cache entries
7E 52 09 78 00A3 232     ASHL #9,W^FIL$GQ_CACHE+4,-(SP) ; Byte address from page number
7E 52 09 78 00A9 233     ASHL #9,R2,-(SP) ; Size of cache in bytes
7E 7E D4 00AD 234     CLRL -(SP) ; Null device name string descriptor
0000'CF 50 DD 00AF 235     PUSHL R0 ; Address to store channel
0000'CF 06 FB 00B1 236     CALLS #6,W^FIL$CACHE_INIT ; Init the FIL$OPENFILE cache
5E 04 C0 00B6 237     ADDL #4,SP ; descriptor returned in FIL$GQ_CACHE
00B6 238     RSB ; Clean off channel
00B9 239 10$:
00BA 240 :
00BA 241 : BOO$ALLOC_PAGES - Find a run of contiguous, good pages
00BA 242 :
00BA 243 : Inputs:
00BA 244 :     R0 - Page to start at
00BA 245 :     R1 - Number of pages needed
00BA 246 :     R4 - Number willing to settle for
00BA 247 :
00BA 248 :
00BA 249 :

```



```

00BA 250 : R5 - Maximum page
00BA 251 : Outputs:
00BA 252 : CC - Status (BLSS to an error routine)
00BA 253 : R2 - Number found
00BA 254 : R3 - Starting page number
00BA 255 :
00BA 256 BOO$ALLOC PAGES::
52 5B 17 9C 00BA 257 ROTL #<32-9>,R11,R2 ; PFN of the RPB
50 52 C0 00BE 258 ADDL R2,R0 ; Convert relative PFN to absolute
53 50 D0 00C1 259 MOVL R0,R3 ; Make a copy of starting bit
55 52 C0 00C4 260 ADDL R2,R5 ; NEW*** Bias out of range max value
50 55 D1 00C7 261 30$: CMPL R5,R0 ; Less than max page
1E 19 00CA 262 BLSS 50$ ; No, failure
OD 48 BB 50 E0 00CC 263 BBS R0,@RPB$Q_PFNMAP+4(R11),40$ ; Branch if this is a good page
52 50 53 C3 00D1 264 SUBL3 R3,R0,R2 ; Count of bits (pages) found
54 52 D1 00D5 265 CMPL R2,R4 ; Found enough?
10 18 00D8 266 BGEQ 50$ ; Branch if yes
53 50 01 C1 00DA 267 ADDL3 #1,R0,R3 ; No, reset starting base
50 D6 00DE 268 40$: INCL R0 ; Next bit (page)
E4 51 F5 00E0 269 SOBGTR R1,30$ ; Branch if more to check
52 50 53 C3 00E3 270 SUBL3 R3,R0,R2 ; Count of bits (pages) found
54 52 D1 00E7 271 CMPL R2,R4 ; Found enough?
05 00EA 272 50$: RSB ; Return (Status in CC)

```



```

00EB 274      .SBTTL BOO$IMAGE_ATT - Get image attributes from image header
00EB 275      :++
00EB 276      : Functional Description:
00EB 277      :
00EB 278      : BOO$IMAGE_ATT returns to the caller some attributes of the image
00EB 279      :
00EB 280      : Calling Sequence:
00EB 281      :
00EB 282      : BSBW BOO$IMAGE_ATT
00EB 283      :
00EB 284      : Inputs:
00EB 285      :
00EB 286      : R2 = Size of file in blocks
00EB 287      : R3 = Address of image header block (first one only)
00EB 288      :
00EB 289      : Outputs:
00EB 290      :
00EB 291      : R1 = Number of image header blocks at the front of the image
00EB 292      : R2 = Size of image in blocks excluding the blocks at the end
00EB 293      : containing local symbols, global symbols, or patch text
00EB 294      :
00EB 295      :--
00EB 296
00EB 297 BOO$IMAGE_ATT::
50  04 A3 3C 00EB 298      MOVZWL IHD$W_SYMDBGOFF(R3),R0 ; ANY SYMBOL TABLE INFORMATION?
      0D 13 00EF 299      BEQL 20$ ; BRANCH IF NOT
51  6043 9E 00F1 300      MOVAB IHS$L_DSTVBN(R0)[R3],R1 ; ADR OF 1ST VBN IN DEBUG SYMBOL TABLE
      19 10 00F5 301      BSBB 40$ ; PROCESS IT
51  04 A043 9E 00F7 302      MOVAB IHS$L_GSTVBN(R0)[R3],R1 ; ADR OF 1ST VBN IN GLOBAL SYMBOL TABLE
      12 10 00FC 303      BSBB 40$ ; PROCESS IT
50  08 A3 3C 00FE 304 20$: MOVZWL IHD$W_PATCHOFF(R3),R0 ; ANY PATCH CONTROL INFORMATION?
      07 13 0102 305      BEQL 30$ ; BRANCH IF NOT
51  20 A043 9E 0104 306      MOVAB IHP$L_PATCOMTXT(R0)[R3],R1 ; ADR OF 1ST VBN OF PATCH COMMAND TEXT
      05 10 0109 307      BSBB 40$ ; PROCESS IT
51  10 A3 9A 010B 308 30$: MOVZBL IHD$B_HDRBLKCNT(R3),R1 ; GET IMAGE HEADER BLOCK COUNT
      05 010F 309      RSB
      0110 310
      0110 311 ; SEE IF VBN IS NON ZERO AND THEN IF IT IS SMALLER THAN THE CURRENT SMALLEST
      0110 312
51  61 01 C3 0110 313 40$: SUBL3 #1,(R1),R1 ; FETCH VBN - 1
      08 19 0114 314      BLSS 50$ ; BRANCH IF NO VBN IS PRESENT
51  52 D1 0116 315      CMPL R2,R1 ; IS IT SMALLER THAN THE CURRENT ONE
      03 15 0119 316      BLEQ 50$ ; BRANCH IF NOT
52  51 D0 011B 317      MOVL R1,R2 ; YES, USE IT
      05 011E 318 50$: RSB

```


BOOTIOUV1
V03-002

I 6

- BOOTSTRAP FILEREAD IO MODULE FOR MICRO 10-AUG-1984 18:05:02 VAX/VMS Macro V04-00 Page 9
Common Globals for VMB and SYSBOOT 9-JUL-1984 11:36:49 [GAMACHE.UV1ROM.VMS]BOOTIO.MAR;1 (1)

```
011F 359 .SBTTL Common Globals for VMB and SYSBOOT
011F 360 :
011F 361 : The following globals are common to VMB and SYSBOOT and are
011F 362 : defined here to avoid replicate definitions.
011F 363 :
011F 364 FIL$GT_DDSTRING:: ; Default directory string.
SD 45 58 45 53 59 53 5B 00' 011F 365 .ASCIC /[SYSEXEC]/
08 011F
00 0128 366 FIL$GT_DDDEV:: ; Default device name
0128 367 .BYTE 0 ; Null ASCII string
0129 368
0129 369 .END
```


BOOTIOUV1
Symbol table

BOOSALLOC_PAGES	000000BA	RG	02
BOOSCACHE_ALLOC	0000005C	RG	02
BOOSCACHE_INIT	0000008E	RG	02
BOOSCACHE_OPEN	00000090	RG	02
BOOSIMAGE_ATT	000000EB	RG	02
BOOT_UV1_SWITCH	= 00000001		
BUFADR	= 0000000C		
BYTCNT	= 00000014		
CHAN	= 00000004		
FILSCACHE_INIT	*****	X	02
FILSC_DIR_SIZE	*****	X	02
FILSC_SIZE	*****	X	02
FILSGD_CACHE	*****	X	02
FILSGT_DDDEV	00000128	RG	02
FILSGT_DDSTRING	0000011F	RG	02
FILSRDQRTLBN	00000000	RG	02
IHDSB_HDRBLKCNT	= 00000010		
IHDSW_PATCHOFF	= 00000008		
IHDSW_SYMDBGOFF	= 00000004		
IHPSL_PATCOMTXT	= 00000020		
IHSSL_DSTVBN	= 00000000		
IHSSL_GSTVBN	= 00000004		
IOFUNC	= 00000010		
LBN	= 00000008		
MEM_CACHE_TABLE	00000020	R	02
PQ	= 00000001	G	
RPBSL_IOVEC	= 00000034		
RPBSL_PFNcnt	= 0000004C		
RPBSQ_PFNMAP	= 00000044		

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
\$ABSS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
YFILEREAD	00000129 (297.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	9	00:00:00.07	00:00:00.27
Command processing	78	00:00:00.58	00:00:01.19
Pass 1	130	00:00:02.83	00:00:04.36
Symbol table sort	0	00:00:00.21	00:00:00.24
Pass 2	61	00:00:00.97	00:00:01.53
Symbol table output	4	00:00:00.04	00:00:00.04
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	284	00:00:04.74	00:00:07.66

The working set limit was 900 pages.

15815 bytes (31 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 186 non-local and 10 local symbols.
371 source lines were read in Pass 1, producing 13 object records in Pass 2.
12 pages of virtual memory were used to define 11 macros.

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

Macro library name	Macros defined
-----	-----
DISK\$STARWORK03:[GAMACHE.UV1ROM.VMS]LIBUV1.ML	4
DISK\$STARWORK03:[GAMACHE.UV1ROM.OBJ]VMB.MLB;3	0
SYS\$SYSROOT:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	7

237 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

MAC/LIS=LIS\$:BOOTIOUV1/OBJ=OBJ\$:BOOTIOUV1 VMS\$:BOOUV1SWT+VMS\$:BOOTIO+OBJ\$:VMB/LIB+VMS\$:LIBUV1/LIB

0430 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

