J 6

```
MMM         MMM  TTTTTTTTTTTTTTT     AAAAAAAAA       AAAAAAAAA       CCCCCCCCCCCC  PPPPPPPPPPPP
MMM         MMM  TTTTTTTTTTTTTTT     AAAAAAAAA       AAAAAAAAA       CCCCCCCCCCCC  PPPPPPPPPPPP
MMM         MMM  TTTTTTTTTTTTTTT     AAAAAAAAA       AAAAAAAAA       CCCCCCCCCCCC  PPPPPPPPPPPP
MMMMMM   MMMMMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP         PPP
MMMMMM   MMMMMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP         PPP
MMMMMM   MMMMMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP         PPP
MMM   MMM   MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP         PPP
MMM   MMM   MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP         PPP
MMM   MMM   MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP         PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPPPPPPPPPPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPPPPPPPPPPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPPPPPPPPPPP
MMM         MMM        TTT        AAAAAAAAAAAAAAA  AAAAAAAAAAAAAAA  CCC              PPP
MMM         MMM        TTT        AAAAAAAAAAAAAAA  AAAAAAAAAAAAAAA  CCC              PPP
MMM         MMM        TTT        AAAAAAAAAAAAAAA  AAAAAAAAAAAAAAA  CCC              PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA  CCC              PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA     CCCCCCCCCCCC  PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA     CCCCCCCCCCCC  PPP
MMM         MMM        TTT        AAA         AAA  AAA         AAA     CCCCCCCCCCCC  PPP
```

```
**FILE**ID**FREEPG
```

```
FFFFFFFFFF  RRRRRRRR    EEEEEEEEEE  EEEEEEEEEE  PPPPPPPP      GGGGGGGG
FFFFFFFFFF  RRRRRRRR    EEEEEEEEEE  EEEEEEEEEE  PPPPPPPP      GGGGGGGG
FF          RR      RR  EE          EE          PP      PP  GG
FF          RR      RR  EE          EE          PP      PP  GG
FF          RR      RR  EE          EE          PP      PP  GG
FF          RR      RR  EE          EE          PP      PP  GG
FFFFFFFF    RRRRRRRR    EEEEEEEE    EEEEEEEE    PPPPPPPP    GG
FFFFFFFF    RRRRRRRR    EEEEEEEE    EEEEEEEE    PPPPPPPP    GG
FF          RR  RR      EE          EE          PP          GG  GGGGGG
FF          RR  RR      EE          EE          PP          GG  GGGGGG
FF          RR    RR    EE          EE          PP          GG      GG
FF          RR    RR    EE          EE          PP          GG      GG      ....
FF          RR      RR  EEEEEEEEEE  EEEEEEEEEE  PP            GGGGGG        ....
FF          RR      RR  EEEEEEEEEE  EEEEEEEEEE  PP            GGGGGG        ....
                                                                            ....

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
```

```
0 MODULE FREEPG (LANGUAGE (BLISS32) ,
0                IDENT = 'V04-000'
0                ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  MTAACP
1 !
1 ! ABSTRACT:
1 !       This module handles the requesting and returning of virtual pages.
1 !
1 ! ENVIRONMENT:
1 !
1 !       Starlet operating system, including privileged system services
1 !       and internal exec routines.
1 !
1 !--
1
1 !
1 !
1 ! AUTHOR:  D. H. GILLESPIE,      CREATION DATE:  9-JUN-77
1 !
1 ! MODIFIED BY:
1 !
1 !       V02-004 DMW00023        David Michael Walp      17-Jul-1981
1 !               Included change shipped with 2.4 plus improvements.  Added
1 !               additional comments through out the module.
1 !
1 !       V02-002 REFORMAT        Maria del C. Nasr       30-Jun-1980
1 !
1 !**
1
```

```
;    58          0058  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    59          0059  1
;    60          0060  1 REQUIRE 'SRC$:MTADEF.B32';
;    61          0444  1
```

```
GLOBAL ROUTINE GET_FREE_PAGE (PAGES, ADDR) : COMMON_CALL NOVALUE =

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine gets the requested number of contiguous pages from
!       the free page list.  If none are available, it expands virtual memory.
!
! CALLING SEQUENCE:
!       GET_FREE_PAGE(ARG1,ARG2)
!
! INPUT PARAMETERS:
!       ARG1 - number of pages
!       ARG2 - address of long word in which to return address of free page
!
! IMPLICIT INPUTS:
!       FREE_PAGE_HEAD  - head of free_page list
!       LAST_PAGE       - last page of virtual memory
!
! OUTPUT PARAMETERS:
!       ARG2 - address of long word in which to return address of free page
!
! IMPLICIT OUTPUTS:
!       none
!
! ROUTINE VALUE:
!       none
!
! SIDE EFFECTS:
!       none
!
!--

    BEGIN

    EXTERNAL REGISTER
        COMMON_REG;

    EXTERNAL
        FREE_PAGE_HEAD  : REF BBLOCK,           ! free page list head
        LAST_PAGE;                              ! address of last page

    EXTERNAL ROUTINE
        SYS$EXPREG : ADDRESSING_MODE (ABSOLUTE); ! expand region

    LOCAL
        SIZE,                                   ! number of bytes requested
        FPAGE   : VECTOR [2],                   ! page references
        TOOBIG  : REF BBLOCK;                   ! address of space which is
                                                !  bigger than need be

    BIND
        FREEPAGE = FPAGE        : REF BBLOCK,
        ENDADDR = FPAGE[1];


    TOOBIG = 0;                                 ! initialize
```

```
SIZE = 512*.PAGES;                              ! number of bytes requested
FREEPAGE = .FREE_PAGE_HEAD;                     ! pickup first free page

! Look down the freepage list for a region of the correct or larger size.
! If we find a region of the correct size return it.  Remember the first
! chuck which is too big, it will be cut down if we do not find a page of
! the correct size
!
WHILE .FREEPAGE NEQA FREE_PAGE_HEAD DO
    BEGIN

    IF .SIZE EQLU .FREEPAGE[FVP$W_SIZE]
    THEN

        ! we found a section of the correct size, remove it from the list
        ! and return it
        BEGIN
        REMQUE(.FREEPAGE, .ADDR);
        RETURN;
        END;

    IF .SIZE LSSU .FREEPAGE[FVP$W_SIZE]
    THEN

        ! this space is too big.  so if we do not already have a chuck to
        ! cut up if needed, then remember this one
        !
        IF .TOOBIG EQLA 0 THEN TOOBIG = .FREEPAGE;

    FREEPAGE = .FREEPAGE[FVP$L_FORWARD];
    END;

IF .TOOBIG NEQ 0
THEN

    ! if there is entry that is too big, leave it in the free page list but
    ! make it smaller and use the end of the block to satisfy the request
    !
    BEGIN
    TOOBIG[FVP$W_SIZE] = .TOOBIG[FVP$W_SIZE] - .SIZE;
    FREEPAGE = .TOOBIG + .TOOBIG[FVP$W_SIZE];
    END

ELSE

    ! otherwise expand the region and update last page pointer
    !
    BEGIN
    IF NOT SYS$EXPREG(.PAGES, FREEPAGE, EXEC_MODE, 0)
    THEN
        ERR_EXIT(SS$_ACPVAFUL);
    LAST_PAGE = .ENDADDR;
    END;

.ADDR = .FREEPAGE;
FREEPAGE[FVP$W_SIZE] = .SIZE;
END;                                            ! end of routine
```

```
                                                                  .TITLE   FREEPG
                                                                  .IDENT   \V04-000\

                                                                  .EXTRN   FREE_PAGE_HEAD, LAST_PAGE
                                                                  .EXTRN   SYS$EXPREG

                                                                  .PSECT   $CODE$,NOWRT,2

                                             000C 00000           .ENTRY   GET_FREE_PAGE, Save R2,R3                    ; 0445
                                 5E       04   C2 00002           SUBL2    #4, SP                                       ;
                                          50   D4 00005           CLRL     TOOBIG                                       ; 0501
                   53      04    AC       09   78 00007           ASHL     #9, PAGES, SIZE                              ; 0502
                                   0000G  CF   DD 0000C           PUSHL    FREE_PAGE_HEAD                               ; 0503
                                 51       6E   D0 00010 1$:       MOVL     FREEPAGE, R1                                 ; 0510
                                 52 0000G CF   9E 00013           MOVAB    FREE_PAGE_HEAD, R2                           ;
                                 52       51   D1 00018           CMPL     R1, R2                                       ;
                                          24   13 0001B           BEQL     4$                                           ;
         53      08    A1        10       00   ED 0001D           CMPZV    #0, #16, 8(R1), SIZE                         ; 0513
                                          05   12 00023           BNEQ     2$                                           ;
                         08      BC       61   0F 00025           REMQUE   (R1), @ADDR                                  ; 0519
                                          04      00029           RET                                                   ; 0518
                                 51       6E   D0 0002A 2$:       MOVL     FREEPAGE, R1                                 ; 0523
         53      08    A1        10       00   ED 0002D           CMPZV    #0, #16, 8(R1), SIZE                         ;
                                          07   1B 00033           BLEQU    3$                                           ;
                                          50   D5 00035           TSTL     TOOBIG                                       ; 0529
                                          03   12 00037           BNEQ     3$                                           ;
                                 50       51   D0 00039           MOVL     R1, TOOBIG                                   ;
                                 6E       61   D0 0003C 3$:       MOVL     (R1), FREEPAGE                               ; 0531
                                          CF   11 0003F           BRB      1$                                           ; 0510
                                          50   D5 00041 4$:       TSTL     TOOBIG                                       ; 0534
                                          0E   13 00043           BEQL     5$                                           ;
                         08      A0       53   A2 00045           SUBW2    SIZE, 8(TOOBIG)                              ; 0541
                                 51    08 A0   3C 00049           MOVZWL   8(TOOBIG), R1                                ; 0542
                   6E            50       51   C1 0004D           ADDL3    R1, TOOBIG, FREEPAGE                         ;
                                          1D   11 00051           BRB      7$                                           ; 0534
                                 7E       01   7D 00053 5$:       MOVQ     #1, -(SP)                                    ; 0550
                                       08 AE   9F 00056           PUSHAB   FREEPAGE                                     ;
                                       04 AC   DD 00059           PUSHL    PAGES                                        ;
                      00000000G  9F       04   FB 0005C           CALLS    #4, @#SYS$EXPREG                             ;
                                 04       50   E8 00063           BLBS     R0, 6$                                       ;
                                     02FC 8F   BF 00066           CHMU     #764                                         ; 0552
                        0000G    CF    04 AE   D0 0006A 6$:       MOVL     ENDADDR, LAST_PAGE                           ; 0553
                                 50       6E   D0 00070 7$:       MOVL     FREEPAGE, R0                                 ; 0556
                         08      BC       50   D0 00073           MOVL     R0, @ADDR                                    ;
                         08      A0       53   B0 00077           MOVW     SIZE, 8(R0)                                  ; 0557
                                          04      0007B           RET                                                   ; 0558

; Routine Size:  124 bytes,    Routine Base:  $CODE$ + 0000


;  177          0559  1
```

```
GLOBAL ROUTINE RET_FREE_PAGE (ADDR,CONTRACT) : COMMON_CALL NOVALUE =

!++
!
! FUNCTIONAL DESCRIPTION:
!       This routine returns a block of contiguous pages to the free page list.
!       If specified  and the page is the last page of virtual memory, then the
!       program section is contracted.  Space is put back so that the highest
!       address is at the tail of the queue.  Contiguous memory is represented
!       by one free page block.
!
! CALLING SEQUENCE:
!       RET_FREE_PAGE(ARG1,ARG2)
!
! INPUT PARAMETERS:
!       ARG1 - address of block to return
!       ARG2 - TRUE or FALSE value, signaling if we should try to contract P0
!
! IMPLICIT INPUTS:
!       The size of the block to be returned is contained in the block
!       structure.
!
! OUTPUT PARAMETERS:
!       none
!
! IMPLICIT OUTPUTS:
!       if virtual memory is contracted, last_page is updated
!
! ROUTINE VALUE:
!       none
!
! SIDE EFFECTS:
!       none
!
!--

    BEGIN

    EXTERNAL REGISTER
        COMMON_REG;

    EXTERNAL
        FREE_PAGE_HEAD  : REF BBLOCK,   ! addr of free page list head
        LAST_PAGE;                      ! addr of last page of virtual memory

    EXTERNAL ROUTINE
        SYS$CNTREG      : ADDRESSING_MODE (ABSOLUTE);

    MAP
        ADDR            : REF BBLOCK;   ! address of virtual memory to return

    LOCAL
        FREEPAGE        : REF BBLOCK,   ! address of free block
        NEXTPAGE        : REF BBLOCK,   ! address of next page
        ENDFREE         : REF BBLOCK;   ! address of the last free page block

    ! make this block a free block
```

```
;  2      !
;  2      ADDR[FVP$B_TYPE] = FVP_TYPE;
;  2
;  2      ! Search backwards through freepage queue.  Insert this page so that the
;  2      ! highest address is at the end of the queue and all others are sorted.
;  2      !
;  2      FREEPAGE = .(FREE_PAGE_HEAD + 4);
;  2
;  2      WHILE .FREEPAGE NEQA FREE_PAGE_HEAD DO
;  3          BEGIN
;  3          IF .ADDR GTRA .FREEPAGE THEN EXITLOOP;
;  3          FREEPAGE = .FREEPAGE[FVP$L_BACKWARD];
;  2          END;                                      ! end of while
;  2
;  2      ! the previous entry has been found or may have either no entries in queue
;  2      ! or this is the lowest address
;  2      !
;  2      NEXTPAGE = .FREEPAGE;                         ! previous or head of list
;  2
;  2      ! if not head of list calculate next entry addr
;  2      !
;  2      IF .NEXTPAGE NEQA FREE_PAGE_HEAD
;  2      THEN NEXTPAGE = .FREEPAGE[FVP$W_SIZE] + .NEXTPAGE;
;  2
;  2
;  2      ! if region being returned is contiguous after a currect entry in the list
;  2      !
;  2      IF .NEXTPAGE EQLA .ADDR
;  2      THEN
;  2
;  2          ! append the new region to the old entry
;  2          !
;  2          FREEPAGE[FVP$W_SIZE] = .FREEPAGE[FVP$W_SIZE] + .ADDR[FVP$W_SIZE]
;  2
;  2      ELSE
;  2
;  2          ! if not contiguous put in queue and adjust FREEPAGE pointer
;  2          !
;  3          BEGIN
;  3          INSQUE(.ADDR, .FREEPAGE);
;  3          FREEPAGE = .ADDR;
;  2          END;
;  2
;  2      !
;  2      ! now if entry contiguous with following one, merge them
;  2      !
;  2
;  2      NEXTPAGE = .FREEPAGE + .FREEPAGE[FVP$W_SIZE];
;  2
;  2      ! is it contiguous with next entry?
;  2      !
;  2      IF .NEXTPAGE EQLA .FREEPAGE[FVP$L_FORWARD]
;  2      THEN
;  3          BEGIN
;  3
;  3          ! remove next entry from queue
;  3          !
```

```
3           REMQUE(.FREEPAGE[FVP$L_FORWARD], NEXTPAGE);
3
3           ! inc size of current entry
3           !
3           FREEPAGE[FVP$W_SIZE] = .FREEPAGE[FVP$W_SIZE] + .NEXTPAGE[FVP$W_SIZE];
2           END;
2
2      !  Should we try to contract the PO virtual address space of the ACP
2      !
2      IF .CONTRACT
2      THEN
3          BEGIN
3
3          ! get highest free area start address
3          !
3          ENDFREE = .(FREE_PAGE_HEAD + 4);
3          NEXTPAGE = .ENDFREE + .ENDFREE[FVP$W_SIZE] - 1;
3
3          IF .NEXTPAGE EQLA .LAST_PAGE
3          THEN
4              BEGIN
4
4              ! update last_page and remove last entry from queue
4              !
4              LAST_PAGE = .ENDFREE - 1;
4              REMQUE(.ENDFREE, ENDFREE);
4
4              ! give back the space
4              !
4              NEXTPAGE = .ENDFREE[FVP$W_SIZE]/512;
4              IF NOT SYS$CNTREG(.NEXTPAGE, 0, EXEC_MODE, 0)
4              THEN
4                  BUG_CHECK(ACPVAFAIL);
4              END
4
3          ELSE
3
3  !**********
3  !
3  ! when making changes try to keep the following two CH$FILLs next to each other
3  ! because BLISS will only generate the code once and branch to it from 2 places
3  !
3  !**********
3
3              ! The area return was not on the end on of the Virtual Address
3              ! Space in PO.  So zero out the newly return pages, plus all
3              ! pointer, size and type fields of the free pages that where
3              ! appended (beacuse they were contiguous).
3              !
3              CH$FILL ( 0, .FREEPAGE[FVP$W_SIZE] - 12, .FREEPAGE + 12 );
3
3          END
2      ELSE
2
2          ! We are not going to try to contract the PO space.  So clean up the
2          ! pages returned.  This will zero out the newly return pages, plus all
2          ! pointer, size and type fields of other free pages that were appended.
```

```
;  350          0731  2          !
;  351          0732  2          CH$FILL ( 0, .FREEPAGE[FVP$W_SIZE] - 12, .FREEPAGE + 12 );
;  352          0733  2
;  353          0734  1      END;                                         ! end of routine


                                                              .EXTRN  SYS$CNTREG, BUG$_ACPVAFAIL

                                               007C 00000      .ENTRY  RET_FREE_PAGE, Save R2,R3,R4,R5,R6            ; 0560
                      56       0000G  CF   9E 00002            MOVAB   FREE_PAGE_HEAD+4, R6                          ;
                      51          04  AC   D0 00007            MOVL    ADDR, R1                                      ; 0618
               0A     A1              01   90 0000B            MOVB    #1, 10(R1)                                    ;
                      50              66   D0 0000F            MOVL    FREE_PAGE_HEAD+4, FREEPAGE                    ; 0623
                      52          FC  A6   9E 00012 1$:        MOVAB   FREE_PAGE_HEAD, R2                            ; 0625
                      52              50   D1 00016            CMPL    FREEPAGE, R2                                  ;
                                      0B   13 00019            BEQL    2$                                            ;
                      50              51   D1 0001B            CMPL    R1, FREEPAGE                                  ; 0627
                                      06   1A 0001E            BGTRU   2$                                            ;
                      50          04  A0   D0 00020            MOVL    4(FREEPAGE), FREEPAGE                         ; 0628
                                      EC   11 00024            BRB     1$                                            ; 0625
                      53              50   D0 00026 2$:        MOVL    FREEPAGE, NEXTPAGE                            ; 0634
                      52          FC  A6   9E 00029            MOVAB   FREE_PAGE_HEAD, R2                            ; 0638
                      52              53   D1 0002D            CMPL    NEXTPAGE, R2                                  ;
                                      07   13 00030            BEQL    3$                                            ;
                      52          08  A0   3C 00032            MOVZWL  8(FREEPAGE), R2                               ; 0639
                      53              52   C0 00036            ADDL2   R2, NEXTPAGE                                  ;
                      51              53   D1 00039 3$:        CMPL    NEXTPAGE, R1                                  ; 0644
                                      07   12 0003C            BNEQ    4$                                            ;
               08     A0          08  A1   A0 0003E            ADDW2   8(R1), 8(FREEPAGE)                            ; 0649
                                      07   11 00043            BRB     5$                                            ;
                      60              61   0E 00045 4$:        INSQUE  (R1), (FREEPAGE)                              ; 0656
                      50          04  AC   D0 00048            MOVL    ADDR, FREEPAGE                                ; 0657
                      53          08  A0   3C 0004C 5$:        MOVZWL  8(FREEPAGE), NEXTPAGE                         ; 0664
                      53              50   C0 00050            ADDL2   FREEPAGE, NEXTPAGE                            ;
                      60              53   D1 00053            CMPL    NEXTPAGE, (FREEPAGE)                          ; 0668
                                      09   12 00056            BNEQ    6$                                            ;
                      53          00  B0   0F 00058            REMQUE  @0(FREEPAGE), NEXTPAGE                        ; 0674
               08     A0          08  A3   A0 0005C            ADDW2   8(NEXTPAGE), 8(FREEPAGE)                      ; 0678
                      3D          08  AC   E9 00061 6$:        BLBC    CONTRACT, 7$                                  ; 0683
                      51              66   D0 00065            MOVL    FREE_PAGE_HEAD+4, ENDFREE                     ; 0689
                      52          08  A1   3C 00068            MOVZWL  8(ENDFREE), R2                                ; 0690
                      53         FF A241   9E 0006C            MOVAB   -1(R2)[ENDFREE], NEXTPAGE                     ;
            0000G     CF              53   D1 00071            CMPL    NEXTPAGE, LAST_PAGE                           ; 0692
                                      2A   12 00076            BNEQ    7$                                            ;
            0000G     CF          FF  A1   9E 00078            MOVAB   -1(R1), LAST_PAGE                             ; 0698
                      51              61   0F 0007E            REMQUE  (ENDFREE), ENDFREE                            ; 0699
                      53          08  A1   3C 00081            MOVZWL  8(ENDFREE), NEXTPAGE                          ; 0703
                      53 00000200     8F   C6 00085            DIVL2   #512, NEXTPAGE                                ;
                      7E              01   7D 0008C            MOVQ    #1, -(SP)                                     ; 0704
                                      7E   D4 0008F            CLRL    -(SP)                                         ;
                                      53   DD 00091            PUSHL   NEXTPAGE                                      ;
        00000000G     9F              04   FB 00093            CALLS   #4, @#SYS$CNTREG                              ;
                      13              50   E8 0009A            BLBS    R0, 8$                                        ;
                                         FEFF 0009D            BUGW                                                  ; 0706
                                        0000* 0009F            .WORD   <BUG$_ACPVAFAIL!4>                            ;
                                           04 000A1            RET                                                   ; 0692
```

```
                                    51      08   A0  3C 000A2 7$:     MOVZWL  8(FREEPAGE), R1                 ; 0732
                                    51           0C  C2 000A6         SUBL2   #12, R1                         ;
        51            00            6E           00  2C 000A9         MOVC5   #0, (SP), #0, R1, 12(FREEPAGE)  ;
                                            0C   A0     000AE                                                 ;
                                                     04 000B0 8$:     RET                                     ; 0734

; Routine Size:  177 bytes,    Routine Base:  $CODE$ + 007C


;  354          0735  1
;  355          0736  1 END
;  356          0737  1
;  357          0738  0 ELUDOM




;                                  PSECT SUMMARY
;
;       Name                    Bytes                       Attributes
;
;  $CODE$                           301  NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)


;                            Library Statistics
;
;                                           -------- Symbols --------      Pages      Processing
;       File                                Total    Loaded   Percent      Mapped     Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1            18619         5         0      1000         00:01.9




;                                  COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:FREEPG/OBJ=OBJ$:FREEPG MSRC$:FREEPG/UPDATE=(ENH$:FREEPG)

; Size:          301 code + 0 data bytes
; Run Time:         00:10.4
; Elapsed Time:     00:30.5
; Lines/CPU Min:     4265
; Lexemes/CPU-Min: 18635
; Memory Used:  110 pages
; Compilation Complete
```

0254 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY