
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

```
FFFFFFFFFF  RRRRRRR    EEEEEEEEEE  EEEEEEEEEE  PPPPPPPP      GGGGGGGG
FFFFFFFFFF  RRRRRRRR   EEEEEEEEEE  EEEEEEEEEE  PPPPPPPPP     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    PPPPPPPPP   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
```

```
;     1          0001  0
;     2          0002  0 MODULE FREEPG (LANGUAGE (BLISS32) ,
;     3          0003  0                IDENT = 'V04-000'
;     4          0004  0                ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !**************************************************************************
;     8          0008  1 !*                                                                        *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                              *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                  *
;    12          0012  1 !*                                                                        *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
;    18          0018  1 !*  TRANSFERRED.                                                           *
;    19          0019  1 !*                                                                        *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
;    22          0022  1 !*  CORPORATION.                                                           *
;    23          0023  1 !*                                                                        *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;    26          0026  1 !*                                                                        *
;    27          0027  1 !*                                                                        *
;    28          0028  1 !**************************************************************************
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 !
;    32          0032  1 ! FACILITY:  MTAACP
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !       This module handles the requesting and returning of virtual pages.
;    36          0036  1 !
;    37          0037  1 ! ENVIRONMENT:
;    38          0038  1 !
;    39          0039  1 !       Starlet operating system, including privileged system services
;    40          0040  1 !       and internal exec routines.
;    41          0041  1 !
;    42          0042  1 !--
;    43          0043  1
;    44          0044  1 !
;    45          0045  1 !
;    46          0046  1 ! AUTHOR:  D. H. GILLESPIE,      CREATION DATE:  9-JUN-77
;    47          0047  1 !
;    48          0048  1 ! MODIFIED BY:
;    49          0049  1 !
;    50          0050  1 !       V02-004 DMW00023        David Michael Walp      17-Jul-1981
;    51          0051  1 !               Included change shipped with 2.4 plus improvements.  Added
;    52          0052  1 !               additional comments through out the module.
;    53          0053  1 !
;    54          0054  1 !       V02-002 REFORMAT        Maria del C. Nasr       30-Jun-1980
;    55          0055  1 !
;    56          0056  1 !**
;    57          0057  1
```

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

```
;    63          0445  1 GLOBAL ROUTINE GET_FREE_PAGE (PAGES, ADDR) : COMMON_CALL NOVALUE =
;    64          0446  1
;    65          0447  1 !++
;    66          0448  1 !
;    67          0449  1 ! FUNCTIONAL DESCRIPTION:
;    68          0450  1 !       This routine gets the requested number of contiguous pages from
;    69          0451  1 !       the free page list.  If none are available, it expands virtual memory.
;    70          0452  1 !
;    71          0453  1 ! CALLING SEQUENCE:
;    72          0454  1 !       GET_FREE_PAGE(ARG1,ARG2)
;    73          0455  1 !
;    74          0456  1 ! INPUT PARAMETERS:
;    75          0457  1 !       ARG1 - number of pages
;    76          0458  1 !       ARG2 - address of long word in which to return address of free page
;    77          0459  1 !
;    78          0460  1 ! IMPLICIT INPUTS:
;    79          0461  1 !       FREE_PAGE_HEAD  - head of free_page list
;    80          0462  1 !       LAST_PAGE       - last page of virtual memory
;    81          0463  1 !
;    82          0464  1 ! OUTPUT PARAMETERS:
;    83          0465  1 !       ARG2 - address of long word in which to return address of free page
;    84          0466  1 !
;    85          0467  1 ! IMPLICIT OUTPUTS:
;    86          0468  1 !       none
;    87          0469  1 !
;    88          0470  1 ! ROUTINE VALUE:
;    89          0471  1 !       none
;    90          0472  1 !
;    91          0473  1 ! SIDE EFFECTS:
;    92          0474  1 !       none
;    93          0475  1 !
;    94          0476  1 !--
;    95          0477  1
;    96          0478  2     BEGIN
;    97          0479  2
;    98          0480  2     EXTERNAL REGISTER
;    99          0481  2         COMMON_REG;
;   100          0482  2
;   101          0483  2     EXTERNAL
;   102          0484  2         FREE_PAGE_HEAD  : REF BBLOCK,           ! free page list head
;   103          0485  2         LAST_PAGE;                              ! address of last page
;   104          0486  2
;   105          0487  2     EXTERNAL ROUTINE
;   106          0488  2         SYS$EXPREG : ADDRESSING_MODE (ABSOLUTE); ! expand region
;   107          0489  2
;   108          0490  2     LOCAL
;   109          0491  2         SIZE,                                   ! number of bytes requested
;   110          0492  2         FPAGE   : VECTOR [2],                   ! page references
;   111          0493  2         TOOBIG  : REF BBLOCK;                   ! address of space which is
;   112          0494  2                                                 !  bigger than need be
;   113          0495  2
;   114          0496  2     BIND
;   115          0497  2         FREEPAGE = FPAGE        : REF BBLOCK,
;   116          0498  2         ENDADDR = FPAGE[1];
;   117          0499  2
;   118          0500  2
;   119          0501  2     TOOBIG = 0;                                 ! initialize
```

```
;  120     0502  2     SIZE = 512*.PAGES;                          ! number of bytes requested
;  121     0503  2     FREEPAGE = .FREE_PAGE_HEAD;                 ! pickup first free page
;  122     0504  2
;  123     0505  2     ! Look down the freepage list for a region of the correct or larger size.
;  124     0506  2     ! If we find a region of the correct size return it.  Remember the first
;  125     0507  2     ! chuck which is too big, it will be cut down if we do not find a page of
;  126     0508  2     ! the correct size
;  127     0509  2     !
;  128     0510  2     WHILE .FREEPAGE NEQA FREE_PAGE_HEAD DO
;  129     0511  2         BEGIN
;  130     0512  2
;  131     0513  2         IF .SIZE EQLU .FREEPAGE[FVP$W_SIZE]
;  132     0514  2         THEN
;  133     0515  2
;  134     0516  2             ! we found a section of the correct size, remove it from the list
;  135     0517  3             ! and return it
;  136     0518  4             BEGIN
;  137     0519  4             REMQUE(.FREEPAGE, .ADDR);
;  138     0520  4             RETURN;
;  139     0521  3             END;
;  140     0522  2
;  141     0523  2         IF .SIZE LSSU .FREEPAGE[FVP$W_SIZE]
;  142     0524  2         THEN
;  143     0525  2
;  144     0526  2             ! this space is too big.  so if we do not already have a chuck to
;  145     0527  2             ! cut up if needed, then remember this one
;  146     0528  2             !
;  147     0529  2             IF .TOOBIG EQLA 0 THEN TOOBIG = .FREEPAGE;
;  148     0530  2
;  149     0531  2         FREEPAGE = .FREEPAGE[FVP$L_FORWARD];
;  150     0532  2         END;
;  151     0533  2
;  152     0534  2     IF .TOOBIG NEQ 0
;  153     0535  2     THEN
;  154     0536  2
;  155     0537  2         ! if there is entry that is too big, leave it in the free page list but
;  156     0538  2         ! make it smaller and use the end of the block to satisfy the request
;  157     0539  2         !
;  158     0540  2         BEGIN
;  159     0541  2         TOOBIG[FVP$W_SIZE] = .TOOBIG[FVP$W_SIZE] - .SIZE;
;  160     0542  2         FREEPAGE = .TOOBIG + .TOOBIG[FVP$W_SIZE];
;  161     0543  2         END
;  162     0544  2
;  163     0545  2     ELSE
;  164     0546  2
;  165     0547  2         ! otherwise expand the region and update last page pointer
;  166     0548  2         !
;  167     0549  2         BEGIN
;  168     0550  2         IF NOT SYS$EXPREG(.PAGES, FREEPAGE, EXEC_MODE, 0)
;  169     0551  2         THEN
;  170     0552  2             ERR_EXIT(SS$_ACPVAFUL);
;  171     0553  2         LAST_PAGE = .ENDADDR;
;  172     0554  2         END;
;  173     0555  2
;  174     0556  2     .ADDR = .FREEPAGE;
;  175     0557  2     FREEPAGE[FVP$W_SIZE] = .SIZE;
;  176     0558  1     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$: MOVL    FREEPAGE, R1                    ; 0523
                53      08 A1           10 00 ED 0C02D  CMPZV   #0, #16, 8(R1), SIZE            ;
                                           07 1B 00033  BLEQU   3$                              ;
                                        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
```

```
;  179     0560  1 GLOBAL ROUTINE RET_FREE_PAGE (ADDR,CONTRACT) : COMMON_CALL NOVALUE =
;  180     0561  1
;  181     0562  1 !++
;  182     0563  1 !
;  183     0564  1 ! FUNCTIONAL DESCRIPTION:
;  184     0565  1 !       This routine returns a block of contiguous pages to the free page list.
;  185     0566  1 !       If specified  and the page is the last page of virtual memory, then the
;  186     0567  1 !       program section is contracted.  Space is put back so that the highest
;  187     0568  1 !       address is at the tail of the queue.  Contiguous memory is represented
;  188     0569  1 !       by one free page block.
;  189     0570  1 !
;  190     0571  1 ! CALLING SEQUENCE:
;  191     0572  1 !       RET_FREE_PAGE(ARG1,ARG2)
;  192     0573  1 !
;  193     0574  1 ! INPUT PARAMETERS:
;  194     0575  1 !       ARG1 - address of block to return
;  195     0576  1 !       ARG2 - TRUE or FALSE value, signaling if we should try to contract P0
;  196     0577  1 !
;  197     0578  1 ! IMPLICIT INPUTS:
;  198     0579  1 !       The size of the block to be returned is contained in the block
;  199     0580  1 !       structure.
;  200     0581  1 !
;  201     0582  1 ! OUTPUT PARAMETERS:
;  202     0583  1 !       none
;  203     0584  1 !
;  204     0585  1 ! IMPLICIT OUTPUTS:
;  205     0586  1 !       if virtual memory is contracted, last_page is updated
;  206     0587  1 !
;  207     0588  1 ! ROUTINE VALUE:
;  208     0589  1 !       none
;  209     0590  1 !
;  210     0591  1 ! SIDE EFFECTS:
;  211     0592  1 !       none
;  212     0593  1 !
;  213     0594  1 !--
;  214     0595  1
;  215     0596  2     BEGIN
;  216     0597  2
;  217     0598  2     EXTERNAL REGISTER
;  218     0599  2         COMMON_REG;
;  219     0600  2
;  220     0601  2     EXTERNAL
;  221     0602  2         FREE_PAGE_HEAD  : REF BBLOCK,   ! addr of free page list head
;  222     0603  2         LAST_PAGE;                      ! addr of last page of virtual memory
;  223     0604  2
;  224     0605  2     EXTERNAL ROUTINE
;  225     0606  2         SYS$CNTREG      : ADDRESSING_MODE (ABSOLUTE);
;  226     0607  2
;  227     0608  2     MAP
;  228     0609  2         ADDR            : REF BBLOCK;   ! address of virtual memory to return
;  229     0610  2
;  230     0611  2     LOCAL
;  231     0612  2         FREEPAGE        : REF BBLOCK,   ! address of free block
;  232     0613  2         NEXTPAGE        : REF BBLOCK,   ! address of next page
;  233     0614  2         ENDFREE         : REF BBLOCK;   ! address of the last free page block
;  234     0615  2
;  235     0616  2     ! make this block a free block
```

```
;  236     0617  2      !
;  237     0618  2      ADDR[FVP$B_TYPE] = FVP_TYPE;
;  238     0619  2
;  239     0620  2      ! Search backwards through freepage queue.  Insert this page so that the
;  240     0621  2      ! highest address is at the end of the queue and all others are sorted.
;  241     0622  2      !
;  242     0623  2      FREEPAGE = .(FREE_PAGE_HEAD + 4);
;  243     0624  2
;  244     0625  2      WHILE .FREEPAGE NEQA FREE_PAGE_HEAD DO
;  245     0626  3          BEGIN
;  246     0627  3          IF .ADDR GTRA .FREEPAGE THEN EXITLOOP;
;  247     0628  3          FREEPAGE = .FREEPAGE[FVP$L_BACKWARD];
;  248     0629  2          END;                                    ! end of while
;  249     0630  2
;  250     0631  2      ! the previous entry has been found or may have either no entries in queue
;  251     0632  2      ! or this is the lowest address
;  252     0633  2      !
;  253     0634  2      NEXTPAGE = .FREEPAGE;                       ! previous or head of list
;  254     0635  2
;  255     0636  2      ! if not head of list calculate next entry addr
;  256     0637  2      !
;  257     0638  2      IF .NEXTPAGE NEQA FREE_PAGE_HEAD
;  258     0639  2      THEN NEXTPAGE = .FREEPAGE[FVP$W_SIZE] + .NEXTPAGE;
;  259     0640  2
;  260     0641  2
;  261     0642  2      ! if region being returned is contiguous after a currect entry in the list
;  262     0643  2      !
;  263     0644  2      IF .NEXTPAGE EQLA .ADDR
;  264     0645  2      THEN
;  265     0646  2
;  266     0647  2          ! append the new region to the old entry
;  267     0648  2          !
;  268     0649  2          FREEPAGE[FVP$W_SIZE] = .FREEPAGE[FVP$W_SIZE] + .ADDR[FVP$W_SIZE]
;  269     0650  2
;  270     0651  2      ELSE
;  271     0652  2
;  272     0653  2          ! if not contiguous put in queue and adjust FREEPAGE pointer
;  273     0654  2          !
;  274     0655  3          BEGIN
;  275     0656  3          INSQUE(.ADDR, .FREEPAGE);
;  276     0657  3          FREEPAGE = .ADDR;
;  277     0658  2          END;
;  278     0659  2
;  279     0660  2      !
;  280     0661  2      ! now if entry contiguous with following one, merge them
;  281     0662  2      !
;  282     0663  2
;  283     0664  2      NEXTPAGE = .FREEPAGE + .FREEPAGE[FVP$W_SIZE];
;  284     0665  2
;  285     0666  2      ! is it contiguous with next entry?
;  286     0667  2      !
;  287     0668  2      IF .NEXTPAGE EQLA .FREEPAGE[FVP$L_FORWARD]
;  288     0669  2      THEN
;  289     0670  3          BEGIN
;  290     0671  3
;  291     0672  3          ! remove next entry from queue
;  292     0673  3          !
```

```
;  293     0674  3          REMQUE(.FREEPAGE[FVP$L_FORWARD], NEXTPAGE);
;  294     0675  3
;  295     0676  3          ! inc size of current entry
;  296     0677  3          !
;  297     0678  3          FREEPAGE[FVP$W_SIZE] = .FREEPAGE[FVP$W_SIZE] + .NEXTPAGE[FVP$W_SIZE];
;  298     0679  3          END;
;  299     0680  2
;  300     0681  2      !  Should we try to contract the P0 virtual address space of the ACP
;  301     0682  2      !
;  302     0683  2      IF .CONTRACT
;  303     0684  2      THEN
;  304     0685  3          BEGIN
;  305     0686  3
;  306     0687  3          ! get highest free area start address
;  307     0688  3          !
;  308     0689  3          ENDFREE = .(FREE_PAGE_HEAD + 4);
;  309     0690  3          NEXTPAGE = .ENDFREE + .ENDFREE[FVP$W_SIZE] - 1;
;  310     0691  3
;  311     0692  3          IF .NEXTPAGE EQLA .LAST_PAGE
;  312     0693  3          THEN
;  313     0694  4              BEGIN
;  314     0695  4
;  315     0696  4              ! update last_page and remove last entry from queue
;  316     0697  4              !
;  317     0698  4              LAST_PAGE = .ENDFREE - 1;
;  318     0699  4              REMQUE(.ENDFREE, ENDFREE);
;  319     0700  4
;  320     0701  4              ! give back the space
;  321     0702  4              !
;  322     0703  4              NEXTPAGE = .ENDFREE[FVP$W_SIZE]/512;
;  323     0704  4              IF NOT SYS$CNTREG(.NEXTPAGE, 0, EXEC_MODE, 0)
;  324     0705  4              THEN
;  325     0706  4                  BUG_CHECK(ACPVAFAIL);
;  326     0707  4              END
;  327     0708  4
;  328     0709  3          ELSE
;  329     0710  3
;  330     0711  3      !**********
;  331     0712  3      !
;  332     0713  3      ! when making changes try to keep the following two CH$FILLs next to each other
;  333     0714  3      ! because BLISS will only generate the code once and branch to it from 2 places
;  334     0715  3      !
;  335     0716  3      !**********
;  336     0717  3
;  337     0718  3              ! The area return was not on the end on of the Virtual Address
;  338     0719  3              ! Space in P0.  So zero out the newly return pages, plus all
;  339     0720  3              ! pointer, size and type fields of the free pages that where
;  340     0721  3              ! appended (beacuse they were contiguous).
;  341     0722  3              !
;  342     0723  3              CH$FILL ( 0, .FREEPAGE[FVP$W_SIZE] - 12, .FREEPAGE + 12 );
;  343     0724  3
;  344     0725  3          END
;  345     0726  2      ELSE
;  346     0727  2
;  347     0728  2          ! We are not going to try to contract the P0 space.  So clean up the
;  348     0729  2          ! pages returned.  This will zero out the newly return pages, plus all
;  349     0730  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
```

FREEPG
LIS

DEACCS
LIS

FIND
LIS

DATECN
LIS

FRMOD1
LIS

CREATE
LIS

GETREQ
LIS

EXPIRE
LIS

CNTRL
LIS

FRMHDR
LIS

HEADER
LIS

COMLABPRC
LIS

ENDVOL
LIS

GETFIB
LIS