```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPP          PPP   AAA         AAA  SSS          RRR         RRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPPPPPPPPPPP       AAA         AAA     SSSSSSSSS     RRRRRRRRRRR        TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAAAAAAAAAAAAAA           SSS   RRR   RRR          TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA           SSS   RRR      RRR       TTT        LLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
PPP                AAA         AAA  SSSSSSSSSSS   RRR         RRR     TTT        LLLLLLLLLLLLLLL
```

FILEID**PASHEAP

```
PPPPPPPP      AAAAAA      SSSSSSSS  HH      HH  EEEEEEEEEE    AAAAAA    PPPPPPPP
PPPPPPPP      AAAAAA      SSSSSSSS  HH      HH  EEEEEEEEEE    AAAAAA    PPPPPPPP
PP      PP   AA      AA   SS        HH      HH  EE            AA    AA  PP      PP
PP      PP   AA      AA   SS        HH      HH  EE            AA    AA  PP      PP
PP      PP   AA      AA   SS        HH      HH  EE            AA    AA  PP      PP
PP      PP   AA      AA   SS        HH      HH  EE            AA    AA  PP      PP
PPPPPPPP     AA      AA     SSSSSS  HHHHHHHHHH  EEEEEEEE      AA    AA  PPPPPPPP
PPPPPPPP     AA      AA     SSSSSS  HHHHHHHHHH  EEEEEEEE      AA    AA  PPPPPPPP
PP           AAAAAAAAAA           SS  HH      HH  EE            AAAAAAAAAA  PP
PP           AAAAAAAAAA           SS  HH      HH  EE            AAAAAAAAAA  PP
PP           AA      AA           SS  HH      HH  EE            AA    AA  PP              ....
PP           AA      AA           SS  HH      HH  EE            AA    AA  PP              ....
PP           AA      AA   SSSSSSSS  HH      HH  EEEEEEEEEE    AA    AA  PP              ....
PP           AA      AA   SSSSSSSS  HH      HH  EEEEEEEEEE    AA    AA  PP              ....

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 MODULE PAS$HEAP ( %TITLE 'NEW, DISPOSE, MARK and RELEASE procedures'
;     2          0002  0                  IDENT = '1-002'                    ! File: PASHEAP.B32 Edit: SBL1002
;     3          0003  0                  ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !******************************************************************************
;     7          0007  1 !*                                                                            *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                   *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                     *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                       *
;    11          0011  1 !*                                                                            *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    17          0017  1 !*  TRANSFERRED.                                                               *
;    18          0018  1 !*                                                                            *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    21          0021  1 !*  CORPORATION.                                                               *
;    22          0022  1 !*                                                                            *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                    *
;    25          0025  1 !*                                                                            *
;    26          0026  1 !*                                                                            *
;    27          0027  1 !******************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:     Pascal Language Support
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This module contains the procedures which implement VAX-11 Pascal
;    36          0036  1 !       heap storage management.  The language names for these procedures
;    37          0037  1 !       are NEW, DISPOSE, MARK and RELEASE.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:  User mode - AST reentrant
;    40          0040  1 !
;    41          0041  1 ! AUTHOR: Steven B. Lionel, CREATION DATE: 8-June-1981
;    42          0042  1 !
;    43          0043  1 ! MODIFIED BY:
;    44          0044  1 !
;    45          0045  1 ! 1-001 - Original.  SBL 8-June-1981
;    46          0046  1 ! 1-002 - Add DISPOSE_HANDLER to turn ACCVIOs during DISPOSEs into ERRDURDIS.
;    47          0047  1 !         SBL 12-July-1982
;    48          0048  1 !--
;    49          0049  1
```

```
;    51          0050  1 %SBTTL 'Declarations'
;    52          0051  1 !
;    53          0052  1 ! PROLOGUE DEFINITIONS
;    54          0053  1 !
;    55          0054  1
;    56          0055  1 REQUIRE 'RTLIN:PASPROLOG';                      ! Externals, linkages, PSECTs, structures
;    57          0119  1
;    58          0120  1 !
;    59          0121  1 ! TABLE OF CONTENTS:
;    60          0122  1 !
;    61          0123  1
;    62          0124  1 FORWARD ROUTINE
;    63          0125  1     PAS$NEW2,                                   ! Allocate new storage
;    64          0126  1     PAS$DISPOSE2: NOVALUE,                      ! Free a single item
;    65          0127  1     PAS$MARK2,                                  ! Mark place on allocated list
;    66          0128  1     PAS$RELEASE2: NOVALUE,                      ! Free all allocated since mark
;    67          0129  1     INITIALIZE_QUEUE: NOVALUE,                  ! Initialize the queue
;    68          0130  1     DISPOSE_HANDLER;                            ! Error handler for DISPOSE
;    69          0131  1
;    70          0132  1 !
;    71          0133  1 ! MACROS:
;    72          0134  1 !
;    73          0135  1 !       NONE
;    74          0136  1 !
;    75          0137  1 ! EQUATED SYMBOLS:
;    76          0138  1 !
;    77          0139  1
;    78          0140  1 LITERAL
;    79          0141  1     PAS$K_HEAP_HDRSIZ = 8;                      ! Size of item header info (unmarked)
;    80          0142  1
;    81          0143  1 !
;    82          0144  1 ! FIELDS:
;    83          0145  1 !
;    84          0146  1
;    85          0147  1 !+
;    86          0148  1 ! Fields in item header
;    87          0149  1 !-
;    88          0150  1
;    89          0151  1 FIELD
;    90          0152  1     PAS$HEAP_FIELDS =
;    91          0153  1         SET
;    92          0154  1
;    93          0155  1         PAS$Q_HEAP_QLINK  = [-16,0,32,0],       ! Link in double-linked queue
;    94          0156  1         PAS$Q_HEAP_HDR    = [-8,0,0,0],         ! Offset of non-marked header
;    95          0157  1         PAS$L_HEAP_SIZE   = [-8,0,32,0],        ! Size of allocated storage
;    96          0158  1         PAS$W_HEAP_FLAGS  = [-4,0,16,0],        ! Status flags
;    97          0159  1         PAS$V_HEAP_DEALL  = [-4,0,1,0],         ! Item has been deallocated
;    98          0160  1         PAS$V_HEAP_MARKER = [-4,1,1,0],         ! Item is a marker
;    99          0161  1         PAS$V_HEAP_MARKED = [-4,2,1,0],         ! Item is on marked queue
;   100          0162  1         PAS$W_ADDR_CHECK  = [-4,16,16,0]        ! Low word of item address
;   101          0163  1                                                 ! (for validity check)
;   102          0164  1
;   103          0165  1         TES;
;   104          0166  1
;   105          0167  1 !
;   106          0168  1 ! OWN STORAGE:
;   107          0169  1 !
```

```
;  108          0170  1
;  109          0171  1 !+
;  110          0172  1 ! Declare head of queue to which we will link items which have been
;  111          0173  1 ! allocated since a MARK.
;  112          0174  1 !-
;  113          0175  1 OWN
;  114          0176  1     MARKED_HEAP_QUEUE: VECTOR [2, LONG],
;  115          0177  1     QUEUE_INITIALIZED: INITIAL (0);
```

```
;  117          0178  1 %SBTTL 'PAS$NEW2 - Allocate new heap storage item'
;  118          0179  1 GLOBAL ROUTINE PAS$NEW2(                              ! Allocate new heap storage item
;  119          0180  1         SIZE                                          ! Item size in bytes
;  120          0181  1     ) =
;  121          0182  1
;  122          0183  1 !++
;  123          0184  1 ! FUNCTIONAL DESCRIPTION:
;  124          0185  1 !
;  125          0186  1 !       This procedure implements the Pascal NEW function.  It allocates
;  126          0187  1 !       heap storage of the specified size and returns a pointer to that
;  127          0188  1 !       storage to the caller.
;  128          0189  1 !
;  129          0190  1 ! CALLING SEQUENCE:
;  130          0191  1 !
;  131          0192  1 !       pointer.wa.v = PAS$NEW2 (size.rlu.v)
;  132          0193  1 !
;  133          0194  1 ! FORMAL PARAMETERS:
;  134          0195  1 !
;  135          0196  1 !       size            The size of the requested item in bytes.
;  136          0197  1 !
;  137          0198  1 ! IMPLICIT INPUTS:
;  138          0199  1 !
;  139          0200  1 !       NONE
;  140          0201  1 !
;  141          0202  1 ! IMPLICIT OUTPUTS:
;  142          0203  1 !
;  143          0204  1 !       NONE
;  144          0205  1 !
;  145          0206  1 ! ROUTINE VALUE:
;  146          0207  1 !
;  147          0208  1 !       The pointer to the beginning of the user storage for the item.
;  148          0209  1 !
;  149          0210  1 ! SIDE EFFECTS:
;  150          0211  1 !
;  151          0212  1 !       Calls LIB$GET_VM to allocate heap storage.
;  152          0213  1 !       May signal PAS$_ERRDURNEW, error during NEW
;  153          0214  1 !
;  154          0215  1 !--
;  155          0216  1
;  156          0217  2     BEGIN
;  157          0218  2
;  158          0219  2     LOCAL
;  159          0220  2         ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS),
;  160          0221  2                                                       ! Address of allocated storage
;  161          0222  2         ALLOC_SIZE,                                   ! Size of allocated storage
;  162          0223  2         MARKED,                                       ! 1 if to be placed on MARKED queue
;  163          0224  2         STATUS;                                       ! Status return from LIB$GET_VM
;  164          0225  2
;  165          0226  2     BUILTIN
;  166          0227  2         INSQUE;
;  167          0228  2
;  168          0229  2     !+
;  169          0230  2     ! Set MARKED flag depending on whether or not a MARK is in effect.
;  170          0231  2     ! At the same time, determine ALLOC_SIZE depending on whether or not
;  171          0232  2     ! the item is to be marked.
;  172          0233  2     !-
;  173          0234  2
```

```
;  174          0235  2     IF .MARKED_HEAP_QUEUE [0] NEQ 0     ! Queue not empty?
;  175          0236  2     THEN
;  176          0237  3         BEGIN
;  177          0238  3         MARKED = 1;
;  178          0239  3         ALLOC_SIZE = .SIZE + PAS$K_HEAP_HDRSIZ + 8;
;  179          0240  3         END
;  180          0241  2     ELSE
;  181          0242  3         BEGIN
;  182          0243  3         MARKED = 0;
;  183          0244  3         ALLOC_SIZE = .SIZE + PAS$K_HEAP_HDRSIZ;
;  184          0245  2         END;
;  185          0246  2
;  186          0247  2     !+
;  187          0248  2     ! Allocate heap storage for item.
;  188          0249  2     !-
;  189          0250  2
;  190          0251  2     STATUS = LIB$GET_VM (ALLOC_SIZE, ITEM);
;  191          0252  2     IF NOT .STATUS
;  192          0253  2     THEN
;  193          0254  3         BEGIN
;  194          0255  3         SIGNAL_STOP (PAS$_ERRDURNEW,0,.STATUS);
;  195          0256  3         RETURN 0;
;  196          0257  2         END;
;  197          0258  2
;  198          0259  2     !+
;  199          0260  2     ! Zero-fill header and storage.
;  200          0261  2     !-
;  201          0262  2
;  202          0263  3         BEGIN
;  203          0264  3         LOCAL
;  204          0265  3             PTR,                            ! Current pointer to item
;  205          0266  3             BYTES_LEFT;                     ! Remaining size to fill
;  206          0267  3         PTR = .ITEM;
;  207          0268  3         BYTES_LEFT = .ALLOC_SIZE;
;  208          0269  3         WHILE (.BYTES_LEFT GTRU 65535) DO
;  209          0270  4             BEGIN
;  210          0271  4             PTR = CH$FILL (0, 65535, .PTR);
;  211          0272  4             BYTES_LEFT = .BYTES_LEFT - 65535
;  212          0273  3             END;
;  213          0274  3         CH$FILL (0, .BYTES_LEFT, .PTR);
;  214          0275  2         END;
;  215          0276  2
;  216          0277  2     !+
;  217          0278  2     ! Set ITEM to point to beginning of user storage.
;  218          0279  2     !-
;  219          0280  2
;  220          0281  2     IF .MARKED
;  221          0282  2     THEN
;  222          0283  2         ITEM = .ITEM + PAS$K_HEAP_HDRSIZ + 8
;  223          0284  2     ELSE
;  224          0285  2         ITEM = .ITEM + PAS$K_HEAP_HDRSIZ;
;  225          0286  2
;  226          0287  2     !+
;  227          0288  2     ! Set appropriate values in header.
;  228          0289  2     !-
;  229          0290  2
;  230          0291  2     ITEM [PAS$L_HEAP_SIZE] = .ALLOC_SIZE;
```

```
;  231          0292  2     ITEM [PAS$W_ADDR_CHECK] = .ITEM;                ! Low word of item address
;  232          0293  2                                                     ! for consistency check
;  233          0294  2
;  234          0295  2     !+
;  235          0296  2     ! If a MARK is in effect, link this item on the queue.
;  236          0297  2     !-
;  237          0298  2
;  238          0299  2     IF .MARKED
;  239          0300  2     THEN
;  240          0301  3         BEGIN
;  241          0302  3         IF NOT .QUEUE_INITIALIZED
;  242          0303  3         THEN
;  243          0304  3             INITIALIZE_QUEUE ();
;  244          0305  3         ITEM [PAS$V_HEAP_MARKED] = 1;               ! Note item as marked
;  245          0306  3         INSQUE (ITEM [PAS$Q_HEAP_QLINK], MARKED_HEAP_QUEUE);    ! Insert at head
;  246          0307  2         END;
;  247          0308  2
;  248          0309  2     RETURN .ITEM;                                   ! Return pointer to user storage
;  249          0310  2
;  250          0311  1     END;                                            ! End of routine PAS$NEW2


                                                                .TITLE  PAS$HEAP NEW, DISPOSE, MARK and RELEASE procedu
;                                                                                       res
                                                                .IDENT  \1-002\

                                                                .PSECT  _PAS$DATA,NOEXE,  PIC,2

                                                        00000 MARKED_HEAP_QUEUE:
                                                                .BLKB   8
                                                        00000000  00008 QUEUE_INITIALIZED:
                                                                .LONG   0                                               ;

                                                                .EXTRN  PAS$NEW2, PAS$DISPOSE2
                                                                .EXTRN  PAS$MARK2, PAS$RELEASE2
                                                                .EXTRN  LIB$GET_VM, PAS$_ERRDURNEW

                                                                .PSECT  _PAS$CODE,NOWRT,  SHR,  PIC,2

                                                                  01FC 00000            .ENTRY  PAS$NEW2, Save R2,R3,R4,R5,R6,R7,R8     ; 0179
                                                 58 00000000'   EF   9E 00002           MOVAB   MARKED_HEAP_QUEUE, R8                   ;
                                                 5E             08   C2 00009           SUBL2   #8, SP                                  ;
                                                                68   D5 0000C           TSTL    MARKED_HEAP_QUEUE                       ; 0235
                                                                0B   13 0000E           BEQL    1$                                      ;
                                                 57             01   D0 00010           MOVL    #1, MARKED                              ; 0238
                                  04    AE      04   AC        10   C1 00013           ADDL3   #16, SIZE, ALLOC_SIZE                   ; 0239
                                                                08   11 00019           BRB     2$                                      ; 0235
                                                                57   D4 0001B 1$:       CLRL    MARKED                                  ; 0243
                                  04    AE      04   AC        08   C1 0001D           ADDL3   #8, SIZE, ALLOC_SIZE                    ; 0244
                                                                5E   DD 00023 2$:       PUSHL   SP                                      ; 0251
                                                     08         AE   9F 00025           PUSHAB  ALLOC_SIZE                              ;
                                        00000000G    00         02   FB 00028           CALLS   #2, LIB$GET_VM                          ;
                                                     13                   50   E8 0002F  BLBS    STATUS, 3$                            ; 0252
                                                                50   DD 00032           PUSHL   STATUS                                  ; 0255
                                                                7E   D4 00034           CLRL    -(SP)                                   ;
                                                00000000G       8F   DD 00036           PUSHL   #PAS$_ERRDURNEW                         ;
                                        00000000G    00         03   FB 0003C           CALLS   #3, LIB$STOP                            ;
```

```
                                                  56  11 00043          BRB     10$                                     ; 0256
                                              53  6E  D0 00045 3$:      MOVL    ITEM, PTR                               ; 0267
                                              56  04  AE  D0 00048      MOVL    ALLOC_SIZE, BYTES_LEFT                  ; 0268
                                   0000FFFF   8F      56  D1 0004C 4$:      CMPL    BYTES_LEFT, #65535                      ; 0269
                                                      11  1B 00053          BLEQU   5$                                      ;
    FFFF    8F          00                    6E      00  2C 00055          MOVC5   #0, (SP), #0, #65535, (PTR)             ; 0271
                                                      63      0005C                                                          ;
                                              56 FFFF0001   E6  9E 0005D      MOVAB   -65535(R6), BYTES_LEFT                  ; 0272
                                                      E6  11 00064          BRB     4$                                      ;
            56          00                    6E      00  2C 00066 5$:      MOVC5   #0, (SP), #0, BYTES_LEFT, (PTR)         ; 0274
                                                      63      0006B                                                          ;
                                              05      57  E9 0006C          BLBC    MARKED, 6$                              ; 0281
                                              6E      10  C0 0006F          ADDL2   #16, ITEM                               ; 0283
                                                      03  11 00072          BRB     7$                                      ;
                                              6E      08  C0 00074 6$:      ADDL2   #8, ITEM                                ; 0285
                                              52      6E  D0 00077 7$:      MOVL    ITEM, R2                                ; 0291
                                        F8    A2  04  AE  D0 0007A          MOVL    ALLOC_SIZE, -8(R2)                      ;
                                        FE    A2      52  B0 0007F          MOVW    R2, -2(R2)                              ; 0292
                                              11      57  E9 00083          BLBC    MARKED, 9$                              ; 0299
                                              05  08  A8  E8 00086          BLBS    QUEUE_INITIALIZED, 8$                   ; 0302
                                     0000V    CF      00  FB 0008A          CALLS   #0, INITIALIZE_QUEUE                    ; 0304
                                        FC    A2      C4  88 0008F 8$:      BISB2   #4, -4(R2)                              ; 0305
                                              68  F0  A2  0E 00093          INSQUE  -16(R2), MARKED_HEAP_QUEUE              ; 0306
                                              50      6E  D0 00097 9$:      MOVL    ITEM, R0                                ; 0309
                                                      04  0009A          RET                                             ;
                                                      50  D4 0009B 10$:     CLRL    R0                                      ; 0311
                                                      04  0009D          RET                                             ;

; Routine Size:  158 bytes,    Routine Base:  _PAS$CODE + 0000

;  251          0312  1 !<BLF/PAGE>
```

```
;  253          0313  1 %SBTTL 'PAS$DISPOSE2 - Deallocate heap storage item'
;  254          0314  1 GLOBAL ROUTINE PAS$DISPOSE2(                                  ! Deallocate heap storage item
;  255          0315  1         POINTER                                               ! Pointer expression
;  256          0316  1     ) : NOVALUE =
;  257          0317  1
;  258          0318  1 !++
;  259          0319  1 ! FUNCTIONAL DESCRIPTION:
;  260          0320  1 !
;  261          0321  1 !       This procedure implements the Pascal DISPOSE function.  It deallocates
;  262          0322  1 !       the specified item which is presumed to have been allocated using
;  263          0323  1 !       the NEW function.
;  264          0324  1 !
;  265          0325  1 ! CALLING SEQUENCE:
;  266          0326  1 !
;  267          0327  1 !       PAS$DISPOSE2 (pointer.ra.v)
;  268          0328  1 !
;  269          0329  1 ! FORMAL PARAMETERS:
;  270          0330  1 !
;  271          0331  1 !       pointer         The address of the item to dispose.
;  272          0332  1 !
;  273          0333  1 ! IMPLICIT INPUTS:
;  274          0334  1 !
;  275          0335  1 !       NONE
;  276          0336  1 !
;  277          0337  1 ! IMPLICIT OUTPUTS:
;  278          0338  1 !
;  279          0339  1 !       NONE
;  280          0340  1 !
;  281          0341  1 ! ROUTINE VALUE:
;  282          0342  1 !
;  283          0343  1 !       NONE
;  284          0344  1 !
;  285          0345  1 ! SIDE EFFECTS:
;  286          0346  1 !
;  287          0347  1 !       May call LIB$FREE_VM to deallocate heap storage.
;  288          0348  1 !       May signal PAS$_ERRDURDIS, error during DISPOSE
;  289          0349  1 !
;  290          0350  1 !--
;  291          0351  1
;  292          0352  2     BEGIN
;  293          0353  2
;  294          0354  2     LOCAL
;  295          0355  2         ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS), ! Allocated item
;  296          0356  2         STATUS;                                           ! Status return from LIB$FREE_VM
;  297          0357  2
;  298          0358  2     !+
;  299          0359  2     ! Enable an error handler to turn ACCVIOs into PAS$_ERRDURDIS.
;  300          0360  2     !-
;  301          0361  2
;  302          0362  2     ENABLE DISPOSE_HANDLER;
;  303          0363  2
;  304          0364  2     !+
;  305          0365  2     ! Get actual address of item.
;  306          0366  2     !-
;  307          0367  2
;  308          0368  2     ITEM = .POINTER;
;  309          0369  2
```

```
;  310     0370  2      !+
;  311     0371  2      ! If consistency check word does not match the low word of the item
;  312     0372  2      ! address, signal an error.
;  313     0373  2      !-
;  314     0374  2
;  315     0375  2      IF .ITEM [PAS$W_ADDR_CHECK] NEQ .ITEM<0,16>
;  316     0376  2      THEN
;  317     0377  3          BEGIN
;  318     0378  3          SIGNAL_STOP (PAS$_ERRDURDIS,0,0);       ! Extra args to allow cross-jumping
;  319     0379  3          RETURN;
;  320     0380  2          END;
;  321     0381  2
;  322     0382  2      !+
;  323     0383  2      ! If item is a marker, it's an error to try and DISPOSE it.  Also if
;  324     0384  2      ! the item has already been disposed, then it's an error.
;  325     0385  2      !-
;  326     0386  2
;  327     0387  2      IF .ITEM [PAS$V_HEAP_MARKER] OR .ITEM [PAS$V_HEAP_DEALL]
;  328     0388  2      THEN
;  329     0389  3          BEGIN
;  330     0390  3          SIGNAL_STOP (PAS$_ERRDURDIS,0,0);       ! Extra args to allow cross-jumping
;  331     0391  3          RETURN;
;  332     0392  2          END;
;  333     0393  2
;  334     0394  2      !+
;  335     0395  2      ! Set the DEALL flag so that it can't be DISPOSEd in the future.
;  336     0396  2      !-
;  337     0397  2
;  338     0398  2      ITEM [PAS$V_HEAP_DEALL] = 1;
;  339     0399  2
;  340     0400  2      !+
;  341     0401  2      ! If item is on the marked queue, just return.
;  342     0402  2      ! We assume a future RELEASE will actually delete it.
;  343     0403  2      !-
;  344     0404  2
;  345     0405  2      IF .ITEM [PAS$V_HEAP_MARKED]
;  346     0406  2      THEN
;  347     0407  2          RETURN;
;  348     0408  2
;  349     0409  2      !+
;  350     0410  2      ! We know that it's not marked, so call LIB$FREE_VM to free the
;  351     0411  2      ! allocated storage.
;  352     0412  2      !-
;  353     0413  2
;  354     0414  2      ITEM [PAS$W_ADDR_CHECK] = 0;
;  355     0415  2      STATUS = LIB$FREE_VM (ITEM [PAS$L_HEAP_SIZE], %REF(ITEM [PAS$Q_HEAP_HDR]));
;  356     0416  2      IF NOT .STATUS
;  357     0417  2      THEN
;  358     0418  3          BEGIN
;  359     0419  3          SIGNAL_STOP (PAS$_ERRDURDIS,0,.STATUS);
;  360     0420  3          RETURN;
;  361     0421  2          END;
;  362     0422  2
;  363     0423  2      RETURN;
;  364     0424  2
;  365     0425  1      END;                                        ! End of routine PAS$DISPOSE2
```

```
                                                        .EXTRN  PAS$_ERRDURDIS, LIB$FREE_VM

                                                        .ENTRY  PAS$DISPOSE2, Save R2,R3        ; 0314
                                                        SUBL2   #4, SP                          ;
                                                        MOVAL   5$, (FP)                        ; 0352
                                                        MOVL    POINTER, ITEM                   ; 0368
                                                        MOVAB   -4(ITEM), R3                    ; 0375
                                                        CMPW    2(R3), ITEM                     ;
                                                        BNEQ    1$                              ;
                                                        BBS     #1, (R3), 1$                    ; 0387
                                                        BLBC    (R3), 2$                        ;
                                                1$:     CLRL    -(SP)                           ; 0390
                                                        BRB     3$                              ;
                                                2$:     BISB2   #1, (R3)                        ; 0398
                                                        BBS     #2, (R3), 4$                    ; 0405
                                                        CLRW    2(R3)                           ; 0414
                                                        MOVAQ   -(R2), (SP)                     ; 0415
                                                        PUSHR   #^M<R2,SP>                      ;
                                                        CALLS   #2, LIB$FREE_VM                 ;
                                                        BLBS    STATUS, 4$                      ; 0416
                                                        PUSHL   STATUS                          ; 0419
                                                3$:     CLRL    -(SP)                           ;
                                                        PUSHL   #PAS$_ERRDURDIS                 ;
                                                        CALLS   #3, LIB$STOP                    ;
                                                4$:     RET                                     ; 0425
                                                5$:     .WORD   Save nothing                    ; 0352
                                                        CLRL    -(SP)                           ;
                                                        PUSHL   SP                              ;
                                                        MOVQ    4(AP), -(SP)                    ;
                                                        CALLS   #3, DISPOSE_HANDLER             ;
                                                        RET                                     ;

; Routine Size:  96 bytes,    Routine Base:  _PAS$CODE + 009E

;  366          0426  1 !<BLF/PAGE>
```

```
;  368          0427  1 %SBTTL 'PAS$MARK2 - Mark place on allocated list'
;  369          0428  1 GLOBAL ROUTINE PAS$MARK2(                             ! Mark place on allocated list
;  370          0429  1         SIZE                                          ! Item size in bytes
;  371          0430  1       ) =
;  372          0431  1
;  373          0432  1 !++
;  374          0433  1 ! FUNCTIONAL DESCRIPTION:
;  375          0434  1 !
;  376          0435  1 !       This procedure implements the Pascal MARK function.  It
;  377          0436  1 !       allocates new storage, like NEW, but marks it in such a
;  378          0437  1 !       way that a future call to PAS$RELEASE2, specifying the
;  379          0438  1 !       pointer value given by PAS$MARK, will free all storage
;  380          0439  1 !       allocated since the call to PAS$MARK.
;  381          0440  1 !
;  382          0441  1 !       NOTE!  MARK and RELEASE are not supported as intrinsic
;  383          0442  1 !       functions in the VAX-11 Pascal compiler.  They are provided
;  384          0443  1 !       here solely for compatibility with the VAX-11 Pascal V1
;  385          0444  1 !       compiler which used MARK and RELEASE in the compiler sources.
;  386          0445  1 !
;  387          0446  1 ! CALLING SEQUENCE:
;  388          0447  1 !
;  389          0448  1 !       pointer.wa.v = PAS$MARK2 (size.rlu.v)
;  390          0449  1 !
;  391          0450  1 ! FORMAL PARAMETERS:
;  392          0451  1 !
;  393          0452  1 !       size            The size of the requested item in bytes.
;  394          0453  1 !
;  395          0454  1 ! IMPLICIT INPUTS:
;  396          0455  1 !
;  397          0456  1 !       MARKED_HEAP_QUEUE
;  398          0457  1 !
;  399          0458  1 ! IMPLICIT OUTPUTS:
;  400          0459  1 !
;  401          0460  1 !       A marker is created and linked onto the marked heap queue.
;  402          0461  1 !
;  403          0462  1 ! ROUTINE VALUE:
;  404          0463  1 !
;  405          0464  1 !       The pointer to the marker
;  406          0465  1 !
;  407          0466  1 ! SIDE EFFECTS:
;  408          0467  1 !
;  409          0468  1 !       Calls LIB$GET_VM to allocate heap storage.
;  410          0469  1 !       May signal PAS$_ERRDURMAR, error during MARK
;  411          0470  1 !
;  412          0471  1 !--
;  413          0472  1
;  414          0473  2     BEGIN
;  415          0474  2
;  416          0475  2     LOCAL
;  417          0476  2         ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS),
;  418          0477  2                                                       ! Address of item
;  419          0478  2         STATUS;                                       ! Status return from LIB$GET_VM
;  420          0479  2
;  421          0480  2     BUILTIN
;  422          0481  2         INSQUE;
;  423          0482  2
;  424          0483  2
```

```
;  425          0484  2      !+
;  426          0485  2      ! Allocate storage for the marker.
;  427          0486  2      !-
;  428          0487  2
;  429          0488  2      STATUS = LIB$GET_VM (%REF(.SIZE+PAS$K_HEAP_HDRSIZ+8), ITEM);
;  430          0489  2      IF NOT .STATUS
;  431          0490  2      THEN
;  432          0491  3          BEGIN
;  433          0492  3          SIGNAL_STOP (PAS$_ERRDURMAR,0,.STATUS);
;  434          0493  3          RETURN 0;
;  435          0494  2          END;
;  436          0495  2
;  437          0496  2      !+
;  438          0497  2      ! Zero-fill header and storage.
;  439          0498  2      !-
;  440          0499  2
;  441          0500  3          BEGIN
;  442          0501  3          LOCAL
;  443          0502  3              PTR,                            ! Current pointer to item
;  444          0503  3              BYTES_LEFT;                     ! Remaining size to fill
;  445          0504  3          PTR = .ITEM;
;  446          0505  3          BYTES_LEFT = .SIZE+PAS$K_HEAP_HDRSIZ+8;
;  447          0506  3          WHILE (.BYTES_LEFT GTRU 65535) DO
;  448          0507  4              BEGIN
;  449          0508  4              PTR = CH$FILL (0, 65535, .PTR);
;  450          0509  4              BYTES_LEFT = .BYTES_LEFT - 65535;
;  451          0510  3              END;
;  452          0511  3          CH$FILL (0, .BYTES_LEFT, .PTR);
;  453          0512  2          END;
;  454          0513  2
;  455          0514  2      !+
;  456          0515  2      ! Initialize the item
;  457          0516  2      !-
;  458          0517  2
;  459          0518  2      ITEM = .ITEM + PAS$K_HEAP_HDRSIZ + 8;
;  460          0519  2      ITEM [PAS$V_HEAP_MARKED] = 1;
;  461          0520  2      ITEM [PAS$V_HEAP_MARKER] = 1;
;  462          0521  2      ITEM [PAS$L_HEAP_SIZE] = .SIZE + PAS$K_HEAP_HDRSIZ + 8;
;  463          0522  2      ITEM [PAS$W_ADDR_CHECK] = .ITEM;    ! For consistency check
;  464          0523  2
;  465          0524  2      !+
;  466          0525  2      ! Insert it on the queue
;  467          0526  2      !-
;  468          0527  2
;  469          0528  2      IF NOT .QUEUE_INITIALIZED
;  470          0529  2      THEN
;  471          0530  2          INITIALIZE_QUEUE ();
;  472          0531  2      INSQUE (ITEM [PAS$Q_HEAP_QLINK], MARKED_HEAP_QUEUE);
;  473          0532  2
;  474          0533  2      RETURN .ITEM;                                   ! Return to caller
;  475          0534  2
;  476          0535  1      END;                                            ! End of routine PAS$MARK2


                                                                    .EXTRN  PAS$_ERRDURMAR
```

```
                                                00FC 00000          .ENTRY  PAS$MARK2, Save R2,R3,R4,R5,R6,R7       ; 0428
                                          5E        08 C2 00002     SUBL2   #8, SP                                  ;
                                             04     AE 9F 00005     PUSHAB  ITEM                                    ; 0488
                                       57    04  AC  10 C1 00008     ADDL3   #16, SIZE, R7                           ;
                                             04  AE  57 D0 0000D     MOVL    R7, 4(SP)                               ;
                                             04     AE 9F 00011     PUSHAB  4(SP)                                   ;
                                 00000000G   00     02 FB 00014     CALLS   #2, LIB$GET_VM                          ;
                                          13        50 E8 0001B     BLBS    STATUS, 1$                              ; 0489
                                                    50 DD 0001E     PUSHL   STATUS                                  ; 0492
                                                    7E D4 00020     CLRL    -(SP)                                   ;
                                          00000000G 8F DD 00022     PUSHL   #PAS$_ERRDURMAR                         ;
                                 00000000G  00      03 FB 00028     CALLS   #3, LIB$STOP                            ;
                                                    54 11 0002F     BRB     5$                                      ; 0493
                                          53    04  AE D0 00031 1$: MOVL    ITEM, PTR                               ; 0504
                                          56        57 D0 00035     MOVL    R7, BYTES_LEFT                          ; 0505
                                 0000FFFF 8F        56 D1 00038 2$: CMPL    BYTES_LEFT, #65535                      ; 0506
                                                    11 1B 0003F     BLEQU   3$                                      ;
    FFFF  8F          00        6E                  00 2C 00041     MOVC5   #0, (SP), #0, #65535, (PTR)             ; 0508
                                                    63    00048     
                                       56 FFFF0001  E6 9E 00049     MOVAB   -65535(R6), BYTES_LEFT                  ; 0509
                                                    E6 11 00050     BRB     2$                                      ; 0506
          56          00        6E                  00 2C 00052 3$: MOVC5   #0, (SP), #0, BYTES_LEFT, (PTR)         ; 0511
                                                    63    00057     
                                             04     AE 10 C0 00058     ADDL2   #16, ITEM                               ; 0518
                                          52    04  AE D0 0005C     MOVL    ITEM, R2                                ; 0519
                                             FC     A2 06 88 00060     BISB2   #6, -4(R2)                              ; 0520
                                             F8     A2 57 D0 00064     MOVL    R7, -8(R2)                              ; 0521
                                             FE     A2 52 B0 00068     MOVW    R2, -2(R2)                              ; 0522
                                          05 00000000' EF E8 0006C     BLBS    QUEUE_INITIALIZED, 4$                   ; 0528
                                       0000V  CF        00 FB 00073     CALLS   #0, INITIALIZE_QUEUE                    ; 0530
                                 00000000'  EF    F0  A2 0E 00078 4$: INSQUE  -16(R2), MARKED_HEAP_QUEUE              ; 0531
                                          50    04  AE D0 00080     MOVL    ITEM, R0                                ; 0533
                                                    04 00084     RET                                             ;
                                                    50 D4 00085 5$: CLRL    R0                                      ; 0535
                                                    04 00087     RET                                             ;

; Routine Size:  136 bytes,    Routine Base:  _PAS$CODE + 00FE

;  477          0536  1 !<BLF/PAGE>
```

```
;  479          0537  1 %SBTTL 'PAS$RELEASE2 - Free all allocated storage since MARK'
;  480          0538  1 GLOBAL ROUTINE PAS$RELEASE2(                                  ! Free all allocated storage since MARK
;  481          0539  1         POINTER: REF VECTOR [, LONG]                          ! Marked item
;  482          0540  1     ) : NOVALUE =
;  483          0541  1
;  484          0542  1 !++
;  485          0543  1 ! FUNCTIONAL DESCRIPTION:
;  486          0544  1 !
;  487          0545  1 !       This procedure implements the Pascal RELEASE function.  It deallocates
;  488          0546  1 !       all storage allocated with NEW since the specified MARK was performed.
;  489          0547  1 !
;  490          0548  1 !       NOTE!  MARK and RELEASE are not defined as intrinsic functions by
;  491          0549  1 !       the VAX-11 Pascal compiler.
;  492          0550  1 !
;  493          0551  1 ! CALLING SEQUENCE:
;  494          0552  1 !
;  495          0553  1 !       PAS$DISPOSE2 (pointer.ra.r)
;  496          0554  1 !
;  497          0555  1 ! FORMAL PARAMETERS:
;  498          0556  1 !
;  499          0557  1 !       pointer                 The address of the item allocated by a
;  500          0558  1 !                               previous call to PAS$MARK2.
;  501          0559  1 !
;  502          0560  1 ! IMPLICIT INPUTS:
;  503          0561  1 !
;  504          0562  1 !       MARKED_HEAP_QUEUE
;  505          0563  1 !
;  506          0564  1 ! IMPLICIT OUTPUTS:
;  507          0565  1 !
;  508          0566  1 !       NONE
;  509          0567  1 !
;  510          0568  1 ! ROUTINE VALUE:
;  511          0569  1 !
;  512          0570  1 !       NONE
;  513          0571  1 !
;  514          0572  1 ! SIDE EFFECTS:
;  515          0573  1 !
;  516          0574  1 !       Disables and reenables AST delivery.
;  517          0575  1 !       Calls LIB$FREE_VM to deallocate heap storage.
;  518          0576  1 !       Removes allocated items from the heap storage queue.
;  519          0577  1 !       May signal PAS$_ERRDURREL, error during RELEASE
;  520          0578  1 !
;  521          0579  1 !--
;  522          0580  1
;  523          0581  2     BEGIN
;  524          0582  2
;  525          0583  2     LOCAL
;  526          0584  2         ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS),       ! Heap marker
;  527          0585  2         CUR_ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS);   ! Current item
;  528          0586  2
;  529          0587  2     BUILTIN
;  530          0588  2         REMQUE;
;  531          0589  2
;  532          0590  2     !+
;  533          0591  2     ! Get actual address of item.
;  534          0592  2     !-
;  535          0593  2
```

```
;  536     0594  2      ITEM = .POINTER [0];
;  537     0595  2
;  538     0596  2      !+
;  539     0597  2      ! If the pointer is zero, it isn't an allocated item.
;  540     0598  2      !-
;  541     0599  2
;  542     0600  2      IF .ITEM EQL 0
;  543     0601  2      THEN
;  544     0602  3          BEGIN
;  545     0603  3          SIGNAL_STOP (PAS$_ERRDURREL);
;  546     0604  3          RETURN;
;  547     0605  2          END;
;  548     0606  2
;  549     0607  2      !+
;  550     0608  2      ! If consistency check word doesn't match low word of item
;  551     0609  2      ! address, signal an error.
;  552     0610  2      !-
;  553     0611  2
;  554     0612  2      IF .ITEM [PAS$W_ADDR_CHECK] NEQ .ITEM<0,16>
;  555     0613  2      THEN
;  556     0614  3          BEGIN
;  557     0615  3          SIGNAL_STOP (PAS$_ERRDURREL);
;  558     0616  3          RETURN;
;  559     0617  2          END;
;  560     0618  2
;  561     0619  2      !+
;  562     0620  2      ! If ITEM is in fact not a marker, signal an error.
;  563     0621  2      !-
;  564     0622  2
;  565     0623  2      IF NOT .ITEM [PAS$V_HEAP_MARKER]
;  566     0624  2      THEN
;  567     0625  3          BEGIN
;  568     0626  3          SIGNAL_STOP (PAS$_ERRDURREL);
;  569     0627  3          RETURN;
;  570     0628  2          END;
;  571     0629  2
;  572     0630  2      !+
;  573     0631  2      ! If marker has already been "deallocated" by a previous RELEASE, free
;  574     0632  2      ! the storage it uses.
;  575     0633  2      !-
;  576     0634  2
;  577     0635  2      IF .ITEM [PAS$V_HEAP_DEALL]
;  578     0636  2      THEN
;  579     0637  3          BEGIN
;  580     0638  3          LOCAL
;  581     0639  3              STATUS;
;  582     0640  3
;  583     0641  3          ITEM [PAS$V_HEAP_MARKER] = 0;    ! Set so that it can't be RELEASEd
;  584     0642  3                                           ! again.
;  585     0643  3          STATUS = LIB$FREE_VM (ITEM [PAS$L_HEAP_SIZE],
;  586     0644  3                                %REF(ITEM [PAS$Q_HEAP_QLINK]));
;  587     0645  3          IF NOT .STATUS
;  588     0646  3          THEN
;  589     0647  4              BEGIN
;  590     0648  4              SIGNAL_STOP (PAS$_ERRDURREL,0,.STATUS);
;  591     0649  4              RETURN;
;  592     0650  3              END;
```

```
;  593          0651  3          END
;  594          0652  3
;  595          0653  2      ELSE
;  596          0654  2
;  597          0655  3          BEGIN
;  598          0656  3
;  599          0657  3          LOCAL
;  600          0658  3              AST_STATUS;                 . Status of AST enable
;  601          0659  3
;  602          0660  3          !+
;  603          0661  3          ! Disable AST delivery.
;  604          0662  3          !-
;  605          0663  3
;  606          0664  3          AST_STATUS = $SETAST (ENBFLG=0);
;  607          0665  3
;  608          0666  3          !+
;  609          0667  3          ! Start removing items from the tail of the marked heap queue and
;  610          0668  3          ! deallocating them until we come to the marker.
;  611          0669  3          !-
;  612          0670  3
;  613          0671  3          IF NOT .QUEUE_INITIALIZED
;  614          0672  3          THEN
;  615          0673  3              INITIALIZE_QUEUE ();
;  616          0674  3          WHILE 1 DO
;  617          0675  4              BEGIN
;  618          0676  4              IF REMQUE (.MARKED_HEAP_QUEUE, CUR_ITEM)    ! TRUE if it fails (!)
;  619          0677  4              THEN
;  620          0678  5                  BEGIN
;  621          0679  5                  SIGNAL_STOP (PAS$_ERRDURREL);
;  622          0680  5                  RETURN;
;  623          0681  4                  END;
;  624          0682  4              CUR_ITEM = .CUR_ITEM + PAS$K_HEAP_HDRSIZ + 8;       ! Point to data area
;  625          0683  4
;  626          0684  4              !+
;  627          0685  4              ! If this is a marker, but not the one we're releasing to,
;  628          0686  4              ! mark it for deallocation.  Otherwise, free the item.
;  629          0687  4              !-
;  630          0688  5              IF .CUR_ITEM [PAS$V_HEAP_MARKER] AND (.CUR_ITEM NEQA .ITEM)
;  631          0689  4              THEN
;  632          0690  4                  CUR_ITEM [PAS$V_HEAP_DEALL] = 1
;  633          0691  4              ELSE
;  634          0692  5                  BEGIN
;  635          0693  5
;  636          0694  5                  LOCAL
;  637          0695  5                      STATUS;
;  638          0696  5
;  639          0697  5                  CUR_ITEM [PAS$V_HEAP_DEALL] = 1;        ! Set as protection against
;  640          0698  5                                                          ! another attempt to DISPOSE it.
;  641          0699  5
;  642          0700  5                  STATUS = LIB$FREE_VM (CUR_ITEM [PAS$L_HEAP_SIZE],
;  643          0701  5                                        %REF(CUR_ITEM [PAS$Q_HEAP_QLINK]));
;  644          0702  5                  IF NOT .STATUS
;  645          0703  5                  THEN
;  646          0704  6                      BEGIN
;  647          0705  6                      SIGNAL_STOP (PAS$_ERRDURREL,0,.STATUS);
;  648          0706  6                      RETURN;
;  649          0707  5                      END;
```

```
;  650     0708  5
;  651     0709  4                    END;
;  652     0710  4
;  653     0711  4                IF .CUR_ITEM EQLA .ITEM
;  654     0712  4                THEN
;  655     0713  4                    EXITLOOP;
;  656     0714  4
;  657     0715  3                END;
;  658     0716  3
;  659     0717  3            !+
;  660     0718  3            ! Reenable ASTs if previously enabled.
;  661     0719  3            !-
;  662     0720  3
;  663     0721  3            IF .AST_STATUS EQL SS$_WASSET
;  664     0722  3            THEN
;  665     0723  3                $SETAST (ENBFLG=1);
;  666     0724  3
;  667     0725  2            END;
;  668     0726  2
;  669     0727  2        RETURN;
;  670     0728  2
;  671     0729  1        END;                                    ! End of routine PAS$RELEASE2


                                                               .EXTRN  PAS$_ERRDURREL, SYS$SETAST

                                                 01FC 00000    .ENTRY  PAS$RELEASE2, Save R2,R3,R4,R5,R6,R7,R8      ; 0538
                    58 00000000G   00   9E 00002             MOVAB   LIB$FREE_VM, R8                              ;
                    57 00000000G   00   9E 00009             MOVAB   LIB$STOP, R7                                 ;
                    56 00000000G   8F   D0 00010             MOVL    #PAS$_ERRDURREL, R6                          ;
                    55 00000000G   00   9E 00017             MOVAB   SYS$SETAST, R5                               ;
                    5E             04   C2 0001E             SUBL2   #4, SP                                       ;
                    53       04    BC   D0 00021             MOVL    @POINTER, ITEM                               ; 0594
                                   40   13 00025             BEQL    3$                                           ; 0600
                    53       FE    A3   B1 00027             CMPW    -2(ITEM), ITEM                               ; 0612
                                   3A   12 0002B             BNEQ    3$                                           ;
                35       FC    A3        01 E1 0002D        BBC     #1, -4(ITEM), 3$                             ; 0623
                               14        FC A3 E9 00032      BLBC    -4(ITEM), 1$                                 ; 0635
                          FC   A3        02 8A 00036        BICB2   #2, -4(ITEM)                                 ; 0641
                               6E        F0 A3 9E 0003A     MOVAB   -16(R3), (SP)                                ; 0644
                                           5E DD 0003E      PUSHL   SP                                           ;
                                           F8 A3 9F 00040   PUSHAB  -8(ITEM)                                     ; 0643
                               68          02 FB 00043      CALLS   #2, LIB$FREE_VM                              ;
                               4A          50 E9 00046      BLBC    STATUS, 6$                                   ; 0645
                                           04 00049         RET                                                  ; 0648
                                           7E D4 0004A 1$:  CLRL    -(SP)                                        ; 0664
                               65          01 FB 0004C      CALLS   #1, SYS$SETAST                               ;
                               54          50 D0 0004F      MOVL    R0, AST_STATUS                               ;
                               05 00000000' EF E8 00052     BLBS    QUEUE_INITIALIZED, 2$                        ; 0671
                          0000V CF         00 FB 00059      CALLS   #0, INITIALIZE_QUEUE                         ; 0673
                               52 00000000' FF 0F 0005E 2$: REMQUE  @MARKED_HEAP_QUEUE, CUR_ITEM                 ; 0676
                                           06 1C 00065      BVC     4$                                           ;
                                           56 DD 00067 3$:  PUSHL   R6                                           ; 0679
                               67          01 FB 00069      CALLS   #1, LIB$STOP                                 ;
                                           04 0006C         RET                                                  ; 0678
                               52          10 C0 0006D 4$:  ADDL2   #16, CUR_ITEM                                ; 0682
```

```
                 0B     FC   A2            01  E1 00070          BBC     #1, -4(CUR_ITEM), 5$                    ; 0688
                        53                 52  D1 00075          CMPL    CUR_ITEM, ITEM                          ;
                                           06  13 00078          BEQL    5$                                      ;
                        FC   A2            01  88 0007A          BISB2   #1, -4(CUR_ITEM)                        ; 0690
                                           1D  11 0007E          BRB     7$                                      ;
                        FC   A2            01  88 00080 5$:      BISB2   #1, -4(CUR_ITEM)                        ; 0697
                             6E       F0   A2  9E 00084          MOVAB   -16(R2), (SP)                           ; 0701
                                           5E  DD 00088          PUSHL   SP                                      ;
                                      F8   A2  9F 0008A          PUSHAB  -8(CUR_ITEM)                            ; 0700
                             68            02  FB 0008D          CALLS   #2, LIB$FREE_VM                         ;
                             0A            50  E8 00090          BLBS    STATUS, 7$                              ; 0702
                                           50  DD 00093 6$:      PUSHL   STATUS                                  ; 0705
                                           7E  D4 00095          CLRL    -(SP)                                   ;
                                           56  DD 00097          PUSHL   R6                                      ;
                             67            03  FB 00099          CALLS   #3, LIB$STOP                            ;
                                           04  0009C            RET                                             ; 0704
                             53            52  D1 0009D 7$:      CMPL    CUR_ITEM, ITEM                          ; 0711
                                           BC  12 000A0          BNEQ    2$                                      ;
                             09            54  D1 000A2          CMPL    AST_STATUS, #9                          ; 0721
                                           05  12 000A5          BNEQ    8$                                      ;
                                           01  DD 000A7          PUSHL   #1                                      ; 0723
                             65            01  FB 000A9          CALLS   #1, SYS$SETAST                          ;
                                           04  000AC 8$:         RET                                             ; 0729

; Routine Size:  173 bytes,    Routine Base:  _PAS$CODE + 0186

;  672          0730  1 !<BLF/PAGE>
```

```
;  674          0731  1 %SBTTL 'INITIALIZE_QUEUE - Initialize MARKED_HEAP_QUEUE'
;  675          0732  1 ROUTINE INITIALIZE_QUEUE
;  676          0733  1     : NOVALUE =
;  677          0734  1
;  678          0735  1 !++
;  679          0736  1 ! FUNCTIONAL DESCRIPTION:
;  680          0737  1 !
;  681          0738  1 !       Initializes MARKED_HEAP_QUEUE to be an empty queue.
;  682          0739  1 !
;  683          0740  1 ! CALLING SEQUENCE:
;  684          0741  1 !
;  685          0742  1 !       INITIALIZE_QUEUE ()
;  686          0743  1 !
;  687          0744  1 ! FORMAL PARAMETERS:
;  688          0745  1 !
;  689          0746  1 !       NONE
;  690          0747  1 !
;  691          0748  1 ! IMPLICIT INPUTS:
;  692          0749  1 !
;  693          0750  1 !       MARKED_HEAP_QUEUE
;  694          0751  1 !       QUEUE_INITIALIZED
;  695          0752  1 !
;  696          0753  1 ! IMPLICIT OUTPUTS:
;  697          0754  1 !
;  698          0755  1 !       MARKED_HEAP_QUEUE
;  699          0756  1 !       QUEUE_INITIALIZED
;  700          0757  1 !
;  701          0758  1 ! COMPLETION STATUS:
;  702          0759  1 !
;  703          0760  1 !       NONE
;  704          0761  1 !
;  705          0762  1 ! SIDE EFFECTS:
;  706          0763  1 !
;  707          0764  1 !       Makes MARKED_HEAP_QUEUE an empty queue.
;  708          0765  1 !
;  709          0766  1 ! SIGNALLED ERRORS:
;  710          0767  1 !
;  711          0768  1 !       NONE
;  712          0769  1 !--
;  713          0770  1
;  714          0771  2     BEGIN
;  715          0772  2
;  716          0773  2     LOCAL
;  717          0774  2         AST_STATUS;                             ! Previous AST enable status
;  718          0775  2
;  719          0776  2     BUILTIN
;  720          0777  2         TESTBITCS;
;  721          0778  2
;  722          0779  2     !+
;  723          0780  2     ! Disable ASTs.
;  724          0781  2     !-
;  725          0782  2
;  726          0783  2     AST_STATUS = $SETAST (ENBFLG = 0);
;  727          0784  2
;  728          0785  2     !+
;  729          0786  2     ! If QUEUE_INITIALIZED is still clear, initialize MARKED_HEAP_QUEUE to
;  730          0787  2     ! be an empty queue.  Set QUEUE_INITIALIZED.
```

```
;  731          0788  2      !-
;  732          0789  2
;  733          0790  2      IF TESTBITCS (QUEUE_INITIALIZED)
;  734          0791  2      THEN
;  735          0792  3          BEGIN
;  736          0793  3          MARKED_HEAP_QUEUE [0] = MARKED_HEAP_QUEUE;      ! Set forward link
;  737          0794  3          MARKED_HEAP_QUEUE [1] = .MARKED_HEAP_QUEUE [0]; ! Set backward link
;  738          0795  2          END;
;  739          0796  2
;  740          0797  2      !+
;  741          0798  2      ! Reenable ASTs if previously enabled.
;  742          0799  2      !-
;  743          0800  2
;  744          0801  2      IF .AST_STATUS EQL SS$_WASSET
;  745          0802  2      THEN
;  746          0803  2          $SETAST (ENB_LG = 1);
;  747          0804  2
;  748          0805  2      RETURN;
;  749          0806  2
;  750          0807  1      END;                                    ! End of routine INITIALIZE_QUEUE



                                000C 00000 INITIALIZE_QUEUE:
                                           .WORD   Save R2,R3                                          ; 0732
                          53 00000000G  00  9E 00002           MOVAB   SYS$SETAST, R3
                          52 00000000'  EF  9E 00009           MOVAB   MARKED_HEAP_QUEUE, R2
                                        7E  D4 00010           CLRL    -(SP)                           ; 0783
                                        63  01 FB 00012        CALLS   #1, SYS$SETAST
                                07  08  A2  00 E2 00015        BBSS    #0, QUEUE_INITIALIZED, 1$       ; 0790
                                        62  62 9E 0001A        MOVAB   MARKED_HEAP_QUEUE, MARKED_HEAP_QUEUE ; 0793
                                    04  A2  62 D0 0001D        MOVL    MARKED_HEAP_QUEUE, MARKED_HEAP_QUEUE+4 ; 0794
                                        09  50 D1 00021 1$:    CMPL    AST_STATUS, #9                  ; 0801
                                            05 12 00024        BNEQ    2$
                                            01 DD 00026        PUSHL   #1                              ; 0803
                                        63  01 FB 00028        CALLS   #1, SYS$SETAST
                                            04 0002B 2$:       RET                                     ; 0807

; Routine Size:  44 bytes,    Routine Base:  _PAS$CODE + 0233

;  751          0808  1
;  752          0809  1 !<BLF/PAGE>
```

```
;  754          0810  1 %SBTTL 'DISPOSE_HANDLER - Error handler for DISPOSE'
;  755          0811  1 ROUTINE DISPOSE_HANDLER (
;  756          0812  1     SIGNAL_ARGS: REF BLOCK [, BYTE],          ! Signal arguments list
;  757          0813  1     MECHANISM_ARGS: REF BLOCK [, BYTE]        ! Mechanism arguments list
;  758          0814  1     ) =
;  759          0815  1
;  760          0816  1 !++
;  761          0817  1 ! FUNCTIONAL DESCRIPTION:
;  762          0818  1 !
;  763          0819  1 !       DISPOSE_HANDLER is a condition handler enabled by DISPOSE.  It converts
;  764          0820  1 !       zero-level access violations into PAS$_ERRDURDIS.  It is presumed that
;  765          0821  1 !       any access violations in DISPOSE are caused by invalid pointers.
;  766          0822  1 !
;  767          0823  1 ! CALLING SEQUENCE:
;  768          0824  1 !
;  769          0825  1 !       ret_status.wlc.v = DISPOSE_HANDLER (signal_args.mz.r, mechanism_args.rz.r)
;  770          0826  1 !
;  771          0827  1 ! FORMAL PARAMETERS:
;  772          0828  1 !
;  773          0829  1 !       SIGNAL_ARGS     - The signal arguments list
;  774          0830  1 !       MECHANISM_ARGS  - The mechanism arguments list
;  775          0831  1 !
;  776          0832  1 ! IMPLICIT INPUTS:
;  777          0833  1 !
;  778          0834  1 !       NONE
;  779          0835  1 !
;  780          0836  1 ! IMPLICIT OUTPUTS:
;  781          0837  1 !
;  782          0838  1 !       NONE
;  783          0839  1 !
;  784          0840  1 ! COMPLETION STATUS:
;  785          0841  1 !
;  786          0842  1 !       SS$_RESIGNAL
;  787          0843  1 !
;  788          0844  1 ! SIDE EFFECTS:
;  789          0845  1 !
;  790          0846  1 !       NONE
;  791          0847  1 !
;  792          0848  1 ! SIGNALLED ERRORS:
;  793          0849  1 !
;  794          0850  1 !       NONE
;  795          0851  1 !--
;  796          0852  1
;  797          0853  2     BEGIN
;  798          0854  2
;  799          0855  2     IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_ACCVIO AND
;  800          0856  2        .MECHANISM_ARGS [CHF$L_MCH_DEPTH] EQL 0
;  801          0857  2     THEN
;  802          0858  3         BEGIN
;  803          0859  3         !+
;  804          0860  3         ! Change SS$_ACCVIO to PAS$_ERRDURDIS.
;  805          0861  3         !-
;  806          0862  3
;  807          0863  3         SIGNAL_ARGS [CHF$L_SIG_NAME] = PAS$_ERRDURDIS;
;  808          0864  3         SIGNAL_ARGS [12,0,32,0] = 0;     ! FAO Argument count
;  809          0865  3         SIGNAL_ARGS [16,0,32,0] = 0;     ! Erase original SS$_ACCVIO arguments
;  810          0866  3         SIGNAL_ARGS [20,0,32,0] = 0;
```

```
;  811          0867  2          END;
;  812          0868  2
;  813          0869  2      RETURN SS$_RESIGNAL;
;  814          0870  2
;  815          0871  1      END;                                    End of routine DISPOSE_HANDLER



                                        0000 00000 DISPOSE_HANDLER:
                                                   .WORD    Save nothing                                ; 0811
                              50        04    AC   D0 00002          MOVL     SIGNAL_ARGS, R0          ; 0855
                              0C        04    AC   D1 00006          CMPL     4(R0), #12               ;
                                                   12 0000A          BNEQ     1$                       ;
                              51        08    AC   D0 0000C          MOVL     MECHANISM_ARGS, R1       ; 0856
                                        08    A1   D5 00010          TSTL     8(R1)                    ;
                                                   12 00013          BNEQ     1$                       ;
                         04   A0 00000000G   8F   D0 00015          MOVL     #PAS$_ERRDURDIS, 4(R0)   ; 0863
                                        0C    A0   7C 0001D          CLRQ     12(R0)                   ; 0864
                                        14    A0   D4 00020          CLRL     20(R0)                   ; 0866
                              50      0918    8F   3C 00023 1$:      MOVZWL   #2328, R0                ; 0869
                                                   04 00028          RET                               ; 0871

; Routine Size:  41 bytes,    Routine Base:  _PAS$CODE + 025F

;  816          0872  1
;  817          0873  1 !<BLF/PAGE>
```

```
;  819          0874  1 END                                   ! End of module PAS$HEAP
;  820          0875  1
;  821          0876  0 ELUDOM


                                                        .EXTRN  LIB$STOP

;                                PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  _PAS$DATA                      12  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _PAS$CODE                     648  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                              Library Statistics

;                                        -------- Symbols --------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]STARLET.L32;1       9776         8         0         581       00:01.0
;  _$255$DUA28:[PASRTL.OBJ]PASLIB.L32;1     427        10         2          33       00:00.4


;                                  COMMAND QUALIFIERS

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

; Size:          648 code + 12 data bytes
; Run Time:          00:14.1
; Elapsed Time:      00:50.7
; Lines/CPU Min:      3740
; Lexemes/CPU-Min: 13306
; Memory Used:  92 pages
; Compilation Complete
```

PASFVOUTP
LIS

PASEOLN2
LIS

PASHEAP
LIS

PASHANDLE
LIS

PASFAB
LIS

PASGET
LIS

PASCVTRT
LIS

PASDATE
LIS

PASEOF2
LIS

PASFINDK
LIS

PASFVINPU
LIS

PASEXPO
LIS

PASGOTO
LIS

PASFILEUT
LIS

PASHALT
LIS

PASDELETE
LIS

PASFIND2
LIS

PASPAGE2
LIS

PASODD
LIS

PASHEX
LIS

PASLINELI
LIS

PASMSGPTR
LIS

PASLIB
LIS

PASLOOKAH
LIS

PASOCT
LIS

PASOPEN2
LIS

PASPUT
LIS

PASTOHAND
LIS

PASLOCATE
LIS

PASMSGTXT
LIS