G 6

```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSSS  RRRRRRRRRRRR     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     RRRRRRRRRRRR           TTT        LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           TTT        LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS     RRRRRRRRRRRR           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  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
PPP              AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
PPP              AAA         AAA  SSSSSSSSSSSS     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
_$2

Sym
---

PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS

PAS

PAS

PAS

PAS
PAS
PAS
PAS
PAS

PAS
PAS
PAS
PAS
PAS
PAS
PAS
PAS
```

```
**FILE**ID**PASHEAP                                   E 16
```

```
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
```

```
0 MODULE PAS$HEAP ( %TITLE 'NEW, DISPOSE, MARK and RELEASE procedures'
0                   IDENT = '1-002'                 ! File: PASHEAP.B32 Edit: SBL1002
0                   ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:     Pascal Language Support
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains the procedures which implement VAX-11 Pascal
1 !       heap storage management.  The language names for these procedures
1 !       are NEW, DISPOSE, MARK and RELEASE.
1 !
1 ! ENVIRONMENT:  User mode - AST reentrant
1 !
1 ! AUTHOR: Steven B. Lionel, CREATION DATE: 8-June-1981
1 !
1 ! MODIFIED BY:
1 !
1 ! 1-001 - Original.  SBL 8-June-1981
1 ! 1-002 - Add DISPOSE_HANDLER to turn ACCVIOs during DISPOSEs into ERRDURDIS.
1 !         SBL 12-July-1982
1 !--
1
```

```
1 %SBTTL 'Declarations'
1 !
1 ! PROLOGUE DEFINITIONS
1 !
1
1 REQUIRE 'RTLIN:PASPROLOG';                         ! Externals, linkages, PSECTs, structures
1
1 !
1 ! TABLE OF CONTENTS:
1 !
1
1 FORWARD ROUTINE
1     PAS$NEW2,                                      ! Allocate new storage
1     PAS$DISPOSE2: NOVALUE,                         ! Free a single item
1     PAS$MARK2,                                     ! Mark place on allocated list
1     PAS$RELEASE2: NOVALUE,                         ! Free all allocated since mark
1     INITIALIZE_QUEUE: NOVALUE,                     ! Initialize the queue
1     DISPOSE_HANDLER;                               ! Error handler for DISPOSE
1
1 !
1 ! MACROS:
1 !
1 !         NONE
1 !
1 ! EQUATED SYMBOLS:
1 !
1
1 LITERAL
1     PAS$K_HEAP_HDRSIZ = 8;                         ! Size of item header info (unmarked)
1
1 !
1 ! FIELDS:
1 !
1
1 !+
1 ! Fields in item header
1 !-
1
1 FIELD
1     PAS$HEAP_FIELDS =
1         SET
1
1         PAS$Q_HEAP_QLINK  = [-16,0,32,0],          ! Link in double-linked queue
1         PAS$Q_HEAP_HDR    = [-8,0,0,0],            ! Offset of non-marked header
1         PAS$L_HEAP_SIZE   = [-8,0,32,0],           ! Size of allocated storage
1         PAS$W_HEAP_FLAGS  = [-4,0,16,0],           ! Status flags
1         PAS$V_HEAP_DEALL  = [-4,0,1,0],            ! Item has been deallocated
1         PAS$V_HEAP_MARKER = [-4,1,1,0],            ! Item is a marker
1         PAS$V_HEAP_MARKED = [-4,2,1,0],            ! Item is on marked queue
1         PAS$W_ADDR_CHECK  = [-4,16,16,0]           ! Low word of item address
1                                                    ! (for validity check)
1
1         TES;
1
1 !
1 ! OWN STORAGE:
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);
```

```
1 %SBTTL 'PAS$NEW2 - Allocate new heap storage item'
1 GLOBAL ROUTINE PAS$NEW2(                              ! Allocate new heap storage item
1           SIZE                                        ! Item size in bytes
1       ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !         This procedure implements the Pascal NEW function.  It allocates
1 !         heap storage of the specified size and returns a pointer to that
1 !         storage to the caller.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !         pointer.wa.v = PAS$NEW2 (size.rlu.v)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !         size            The size of the requested item in bytes.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !         NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !         NONE
1 !
1 ! ROUTINE VALUE:
1 !
1 !         The pointer to the beginning of the user storage for the item.
1 !
1 ! SIDE EFFECTS:
1 !
1 !         Calls LIB$GET_VM to allocate heap storage.
1 !         May signal PAS$_ERRDURNEW, error during NEW
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS),
2                                                       ! Address of allocated storage
2         ALLOC_SIZE,                                   ! Size of allocated storage
2         MARKED,                                       ! 1 if to be placed on MARKED queue
2         STATUS;                                       ! Status return from LIB$GET_VM
2
2     BUILTIN
2         INSQUE;
2
2     !+
2     ! Set MARKED flag depending on whether or not a MARK is in effect.
2     ! At the same time, determine ALLOC_SIZE depending on whether or not
2     ! the item is to be marked.
2     !-
2
```

```
2       IF .MARKED_HEAP_QUEUE [0] NEQ 0      ! Queue not empty?
2       THEN
3           BEGIN
3           MARKED = 1;
3           ALLOC_SIZE = .SIZE + PAS$K_HEAP_HDRSIZ + 8;
3           END
2       ELSE
3           BEGIN
3           MARKED = 0;
3           ALLOC_SIZE = .SIZE + PAS$K_HEAP_HDRSIZ;
2           END;
2
2       !+
2       ! Allocate heap storage for item.
2       !-
2
2       STATUS = LIB$GET_VM (ALLOC_SIZE, ITEM);
2       IF NOT .STATUS
2       THEN
3           BEGIN
3           SIGNAL_STOP (PAS$_ERRDURNEW,0,.STATUS);
3           RETURN 0;
2           END;
2
2       !+
2       ! Zero-fill header and storage.
2       !-
2
3           BEGIN
3           LOCAL
3               PTR,                        ! Current pointer to item
3               BYTES_LEFT;                 ! Remaining size to fill
3           PTR = .ITEM;
3           BYTES_LEFT = .ALLOC_SIZE;
3           WHILE (.BYTES_LEFT GTRU 65535) DO
4               BEGIN
4               PTR = CH$FILL (0, 65535, .PTR);
4               BYTES_LEFT = .BYTES_LEFT - 65535
3               END;
3           CH$FILL (0, .BYTES_LEFT, .PTR);
2           END;
2
2       !+
2       ! Set ITEM to point to beginning of user storage.
2       !-
2
2       IF .MARKED
2       THEN
2           ITEM = .ITEM + PAS$K_HEAP_HDRSIZ + 8
2       ELSE
2           ITEM = .ITEM + PAS$K_HEAP_HDRSIZ;
2
2       !+
2       ! Set appropriate values in header.
2       !-
2
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          04  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>
```

```
1 %SBTTL 'PAS$DISPOSE2 - Deallocate heap storage item'
1 GLOBAL ROUTINE PAS$DISPOSE2(                                  ! Deallocate heap storage item
1         POINTER                                               ! Pointer expression
1     ) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure implements the Pascal DISPOSE function.  It deallocates
1 !       the specified item which is presumed to have been allocated using
1 !       the NEW function.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       PAS$DISPOSE2 (pointer.ra.v)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       pointer         The address of the item to dispose.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! ROUTINE VALUE:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       May call LIB$FREE_VM to deallocate heap storage.
1 !       May signal PAS$_ERRDURDIS, error during DISPOSE
1 !
1 !--
1
2    BEGIN
2
2    LOCAL
2         ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS), ! Allocated item
2         STATUS;                                   ! Status return from LIB$FREE_VM
2
2    !+
2    ! Enable an error handler to turn ACCVIOs into PAS$_ERRDURDIS.
2    !-
2
2    ENABLE DISPOSE_HANDLER;
2
2    !+
2    ! Get actual address of item.
2    !-
2
2    ITEM = .POINTER;
2
```

```
2       !+
2       ! If consistency check word does not match the low word of the item
2       ! address, signal an error.
2       !-
2
2       IF .ITEM [PAS$W_ADDR_CHECK] NEQ .ITEM<0,16>
2       THEN
3           BEGIN
3           SIGNAL_STOP (PAS$_ERRDURDIS,0,0);        ! Extra args to allow cross-jumping
3           RETURN;
2           END;
2
2       !+
2       ! If item is a marker, it's an error to try and DISPOSE it.  Also if
2       ! the item has already been disposed, then it's an error.
2       !-
2
2       IF .ITEM [PAS$V_HEAP_MARKER] OR .ITEM [PAS$V_HEAP_DEALL]
2       THEN
3           BEGIN
3           SIGNAL_STOP (PAS$_ERRDURDIS,0,0);        ! Extra args to allow cross-jumping
3           RETURN;
2           END;
2
2       !+
2       ! Set the DEALL flag so that it can't be DISPOSEd in the future.
2       !-
2
2       ITEM [PAS$V_HEAP_DEALL] = 1;
2
2       !+
2       ! If item is on the marked queue, just return.
2       ! We assume a future RELEASE will actually delete it.
2       !-
2
2       IF .ITEM [PAS$V_HEAP_MARKED]
2       THEN
2           RETURN;
2
2       !+
2       ! We know that it's not marked, so call LIB$FREE_VM to free the
2       ! allocated storage.
2       !-
2
2       ITEM [PAS$W_ADDR_CHECK] = 0;
2       STATUS = LIB$FREE_VM (ITEM [PAS$L_HEAP_SIZE], %REF(ITEM [PAS$Q_HEAP_HDR]));
2       IF NOT .STATUS
2       THEN
3           BEGIN
3           SIGNAL_STOP (PAS$_ERRDURDIS,0,.STATUS);
3           RETURN;
2           END;
2
2       RETURN;
2
1       END;                                         ! End of routine PAS$DISPOSE2
```

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

                                          000C 00000        .ENTRY  PAS$DISPOSE2, Save R2,R3                   ; 0314
                          5E             04  C2 00002        SUBL2   #4, SP                                     ;
                          6D      0047   CF  DE 00005        MOVAL   5$, (FP)                                   ; 0352
                          52        04   AC  D0 0000A        MOVL    POINTER, ITEM                              ; 0368
                          53        FC   A2  9E 0000E        MOVAB   -4(ITEM), R3                               ; 0375
                          52        02   A3  B1 00012        CMPW    2(R3), ITEM                                ;
                                         07  12 00016        BNEQ    1$                                         ;
             03           63             01  E0 00018        BBS     #1, (R3), 1$                               ; 0387
                          04             63  E9 0001C        BLBC    (R3), 2$                                   ;
                                         7E  D4 0001F 1$:    CLRL    -(SP)                                      ; 0390
                                         1D  11 00021        BRB     3$                                         ;
                          63             01  88 00023 2$:    BISB2   #1, (R3)                                   ; 0398
             25           63             02  E0 00026        BBS     #2, (R3), 4$                               ; 0405
                                    02   A3  B4 0002A        CLRW    2(R3)                                      ; 0414
                          6E             72  7E 0002D        MOVAQ   -(R2), (SP)                                ; 0415
                                  4004   8F  BB 00030        PUSHR   #^M<R2,SP>                                 ;
                 00000000G 00            02  FB 00034        CALLS   #2, LIB$FREE_VM                            ;
                          11             50  E8 0003B        BLBS    STATUS, 4$                                 ; 0416
                                         50  DD 0003E        PUSHL   STATUS                                     ; 0419
                                         7E  D4 00040 3$:    CLRL    -(SP)                                      ;
                             00000000G   8F  DD 00042        PUSHL   #PAS$_ERRDURDIS                            ;
                 00000000G 00            03  FB 00048        CALLS   #3, LIB$STOP                               ;
                                             04 0004F 4$:    RET                                                ; 0425
                                          0000 00050 5$:    .WORD   Save nothing                               ; 0352
                                         7E  D4 00052        CLRL    -(SP)                                      ;
                                         5E  DD 00054        PUSHL   SP                                         ;
                          7E        04   AC  7D 00056        MOVQ    4(AP), -(SP)                               ;
                     0000V CF            03  FB 0005A        CALLS   #3, DISPOSE_HANDLER                        ;
                                             04 0005F        RET                                                ;

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

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

```
1 %SBTTL 'PAS$MARK2 - Mark place on allocated list'
1 GLOBAL ROUTINE PAS$MARK2(                          ! Mark place on allocated list
1         SIZE                                       ! Item size in bytes
1      ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure implements the Pascal MARK function.  It
1 !       allocates new storage, like NEW, but marks it in such a
1 !       way that a future call to PAS$RELEASE2, specifying the
1 !       pointer value given by PAS$MARK, will free all storage
1 !       allocated since the call to PAS$MARK.
1 !
1 !       NOTE!  MARK and RELEASE are not supported as intrinsic
1 !       functions in the VAX-11 Pascal compiler.  They are provided
1 !       here solely for compatibility with the VAX-11 Pascal V1
1 !       compiler which used MARK and RELEASE in the compiler sources.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       pointer.wa.v = PAS$MARK2 (size.rlu.v)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       size            The size of the requested item in bytes.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       MARKED_HEAP_QUEUE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       A marker is created and linked onto the marked heap queue.
1 !
1 ! ROUTINE VALUE:
1 !
1 !       The pointer to the marker
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Calls LIB$GET_VM to allocate heap storage.
1 !       May signal PAS$_ERRDURMAR, error during MARK
1 !
1 !--
1
2    BEGIN
2
2    LOCAL
2       ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS),
2                                                    ! Address of item
2       STATUS;                                      ! Status return from LIB$GET_VM
2
2    BUILTIN
2       INSQUE;
2
```

```
 2      !+
 2      ! Allocate storage for the marker.
 2      !-
 2
 2      STATUS = LIB$GET_VM (%REF(.SIZE+PAS$K_HEAP_HDRSIZ+8), ITEM);
 2      IF NOT .STATUS
 2      THEN
 3          BEGIN
 3          SIGNAL_STOP (PAS$_ERRDURMAR,0,.STATUS);
 3          RETURN 0;
 2          END;
 2
 2      !+
 2      ! Zero-fill header and storage.
 2      !-
 2
 3          BEGIN
 3          LOCAL
 3              PTR,                        ! Current pointer to item
 3              BYTES_LEFT;                 ! Remaining size to fill
 3          PTR = .ITEM;
 3          BYTES_LEFT = .SIZE+PAS$K_HEAP_HDRSIZ+8;
 3          WHILE (.BYTES_LEFT GTRU 65535) DO
 4              BEGIN
 4              PTR = CH$FILL (0, 65535, .PTR);
 4              BYTES_LEFT = .BYTES_LEFT - 65535;
 3              END;
 3          CH$FILL (0, .BYTES_LEFT, .PTR);
 2          END;
 2
 2      !+
 2      ! Initialize the item
 2      !-
 2
 2      ITEM = .ITEM + PAS$K_HEAP_HDRSIZ + 8;
 2      ITEM [PAS$V_HEAP_MARKED] = 1;
 2      ITEM [PAS$V_HEAP_MARKER] = 1;
 2      ITEM [PAS$L_HEAP_SIZE] = .SIZE + PAS$K_HEAP_HDRSIZ + 8;
 2      ITEM [PAS$W_ADDR_CHECK] = .ITEM;     ! For consistency check
 2
 2      !+
 2      ! Insert it on the queue
 2      !-
 2
 2      IF NOT .QUEUE_INITIALIZED
 2      THEN
 2          INITIALIZE_QUEUE ();
 2      INSQUE (ITEM [PAS$Q_HEAP_QLINK], MARKED_HEAP_QUEUE);
 2
 2      RETURN .ITEM;                                   ! Return to caller
 2
 1      END;                                            ! End of routine PAS$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>
```

```
%SBTTL 'PAS$RELEASE2 - Free all allocated storage since MARK'
GLOBAL ROUTINE PAS$RELEASE2(                                    ! Free all allocated storage since MARK
        POINTER: REF VECTOR [, LONG]                            ! Marked item
    ) : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure implements the Pascal RELEASE function.  It deallocates
!       all storage allocated with NEW since the specified MARK was performed.
!
!       NOTE!  MARK and RELEASE are not defined as intrinsic functions by
!       the VAX-11 Pascal compiler.
!
! CALLING SEQUENCE:
!
!       PAS$DISPOSE2 (pointer.ra.r)
!
! FORMAL PARAMETERS:
!
!       pointer                 The address of the item allocated by a
!                               previous call to PAS$MARK2.
!
! IMPLICIT INPUTS:
!
!       MARKED_HEAP_QUEUE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! ROUTINE VALUE:
!
!       NONE
!
! SIDE EFFECTS:
!
!       Disables and reenables AST delivery.
!       Calls LIB$FREE_VM to deallocate heap storage.
!       Removes allocated items from the heap storage queue.
!       May signal PAS$_ERRDURREL, error during RELEASE
!
!--

    BEGIN

    LOCAL
        ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS),       ! Heap marker
        CUR_ITEM: REF BLOCK [, BYTE] FIELD (PAS$HEAP_FIELDS);   ! Current item

    BUILTIN
        REMQUE;

    !+
    ! Get actual address of item.
    !-

```

```
;  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;
```

```
        END

ELSE

    BEGIN

    LOCAL
        AST_STATUS;                     . Status of AST enable

    !+
    ! Disable AST delivery.
    !-

    AST_STATUS = $SETAST (ENBFLG=0);

    !+
    ! Start removing items from the tail of the marked heap queue and
    ! deallocating them until we come to the marker.
    !-

    IF NOT .QUEUE_INITIALIZED
    THEN
        INITIALIZE_QUEUE ();
    WHILE 1 DO
        BEGIN
        IF REMQUE (.MARKED_HEAP_QUEUE, CUR_ITEM)    ! TRUE if it fails (!)
        THEN
            BEGIN
            SIGNAL_STOP (PAS$_ERRDURREL);
            RETURN;
            END;
        CUR_ITEM = .CUR_ITEM + PAS$K_HEAP_HDRSIZ + 8;       ! Point to data area

        !+
        ! If this is a marker, but not the one we're releasing to,
        ! mark it for deallocation.  Otherwise, free the item.
        !-
        IF .CUR_ITEM [PAS$V_HEAP_MARKER] AND (.CUR_ITEM NEQA .ITEM)
        THEN
            CUR_ITEM [PAS$V_HEAP_DEALL] = 1
        ELSE
            BEGIN

            LOCAL
                STATUS;

            CUR_ITEM [PAS$V_HEAP_DEALL] = 1;        ! Set as protection against
                                                    ! another attempt to DISPOSE it.

            STATUS = LIB$FREE_VM (CUR_ITEM [PAS$L_HEAP_SIZE],
                                  %REF(CUR_ITEM [PAS$Q_HEAP_QLINK]));
            IF NOT .STATUS
            THEN
                BEGIN
                SIGNAL_STOP (PAS$_ERRDURREL,0,.STATUS);
                RETURN;
                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$RELEASE2, Save R2,R3,R4,R5,R6,R7,R8          ; 0538
                    58 0000000G  00   9E 00002                MOVAB    LIB$FREE_VM, R8                                   ;
                    57 0000000G  00   9E 00009                MOVAB    LIB$STOP, R7                                      ;
                    56 0000000G  8F   D0 00010                MOVL     #PAS$_ERRDURREL, R6                               ;
                    55 0000000G  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>
```

```
1 %SBTTL 'INITIALIZE_QUEUE - Initialize MARKED_HEAP_QUEUE'
1 ROUTINE INITIALIZE_QUEUE
1     : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Initializes MARKED_HEAP_QUEUE to be an empty queue.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       INITIALIZE_QUEUE ()
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       MARKED_HEAP_QUEUE
1 !       QUEUE_INITIALIZED
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       MARKED_HEAP_QUEUE
1 !       QUEUE_INITIALIZED
1 !
1 ! COMPLETION STATUS:
1 !
1 !       NONE
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Makes MARKED_HEAP_QUEUE an empty queue.
1 !
1 ! SIGNALLED ERRORS:
1 !
1 !       NONE
1 !--
1
2     BEGIN
2
2     LOCAL
2         AST_STATUS;                         ! Previous AST enable status
2
2     BUILTIN
2         TESTBITCS;
2
2     !+
2     ! Disable ASTs.
2     !-
2
2     AST_STATUS = $SETAST (ENBFLG = 0);
2
2     !+
2     ! If QUEUE_INITIALIZED is still clear, initialize MARKED_HEAP_QUEUE to
2     ! be an empty queue.  Set QUEUE_INITIALIZED.
```

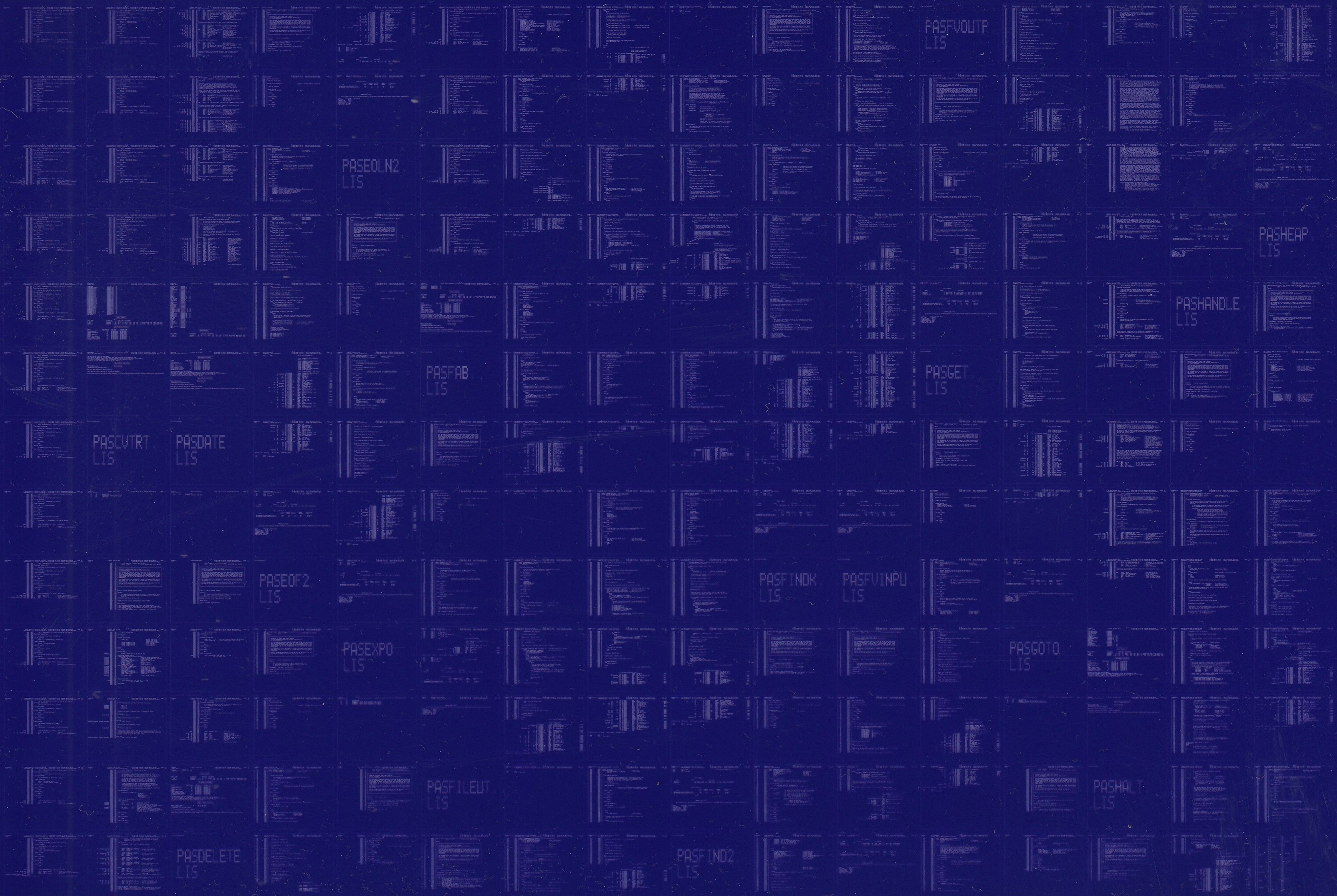
```
;  731          0788  2      !-
;  732          0789  2
;  733          0790  2      IF TESTBITCS (QUEUE_INITIALIZED)
;  734          0791  2      THEN
;  735          0792  3          BEGIN
;  736          0793  3          MARKED_HEAP_QUEUE [0] = MARKED_HEAP_QUEUE;       ! Set forward link
;  737          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>
```

```
1 %SBTTL 'DISPOSE_HANDLER - Error handler for DISPOSE'
1 ROUTINE DISPOSE_HANDLER (
1     SIGNAL_ARGS: REF BLOCK [, BYTE],               ! Signal arguments list
1     MECHANISM_ARGS: REF BLOCK [, BYTE]             ! Mechanism arguments list
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !        DISPOSE_HANDLER is a condition handler enabled by DISPOSE.  It converts
1 !        zero-level access violations into PAS$_ERRDURDIS.  It is presumed that
1 !        any access violations in DISPOSE are caused by invalid pointers.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !        ret_status.wlc.v = DISPOSE_HANDLER (signal_args.mz.r, mechanism_args.rz.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !        SIGNAL_ARGS     - The signal arguments list
1 !        MECHANISM_ARGS  - The mechanism arguments list
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !        NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !        NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_RESIGNAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !        NONE
1 !
1 ! SIGNALLED ERRORS:
1 !
1 !        NONE
1 !--
1
2     BEGIN
2
2     IF .SIGNAL_ARGS [CHF$L_SIG_NAME] EQLU SS$_ACCVIO AND
2        .MECHANISM_ARGS [CHF$L_MCH_DEPTH] EQL 0
2     THEN
3         BEGIN
3         !+
3         ! Change SS$_ACCVIO to PAS$_ERRDURDIS.
3         !-
3
3         SIGNAL_ARGS [CHF$L_SIG_NAME] = PAS$_ERRDURDIS;
3         SIGNAL_ARGS [12,0,32,0] = 0;     ! FAO Argument count
3         SIGNAL_ARGS [16,0,32,0] = 0;     ! Erase original SS$_ACCVIO arguments
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                             ;
                                              17   12 0000A           BNEQ     1$                                     ;
                              51        08    AC   D0 0000C           MOVL     MECHANISM_ARGS, R1                     ; 0856
                                        08    A1   D5 00010           TSTL     8(R1)                                  ;
                                              0E   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
```

0294 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY


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

0295 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY