```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS     CCCCCCCCCCC     AAAAAAAAA     LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS     CCCCCCCCCCC     AAAAAAAAA     LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSS     CCCCCCCCCCC     AAAAAAAAA     LLL
PPP         PPP  AAA         AAA  SSS             CCC             AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS             CCC             AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS             CCC             AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS             CCC             AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS             CCC             AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS             CCC             AAA         AAA  LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS    CCC             AAA         AAA  LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS    CCC             AAA         AAA  LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS    CCC             AAA         AAA  LLL
PPP              AAAAAAAAAAAAAAA             SSS  CCC             AAAAAAAAAAAAAAA  LLL
PPP              AAAAAAAAAAAAAAA             SSS  CCC             AAAAAAAAAAAAAAA  LLL
PPP              AAAAAAAAAAAAAAA             SSS  CCC             AAAAAAAAAAAAAAA  LLL
PPP              AAA         AAA             SSS  CCC             AAA         AAA  LLL
PPP              AAA         AAA             SSS  CCC             AAA         AAA  LLL
PPP              AAA         AAA             SSS  CCC             AAA         AAA  LLL
PPP              AAA         AAA  SSSSSSSSSSS        CCCCCCCCCCC  AAA         AAA  LLLLLLLLLLLLLLL
PPP              AAA         AAA  SSSSSSSSSSS        CCCCCCCCCCC  AAA         AAA  LLLLLLLLLLLLLLL
PPP              AAA         AAA  SSSSSSSSSSS        CCCCCCCCCCC  AAA         AAA  LLLLLLLLLLLLLLL
```

```
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      OOOOOO    44    44
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      OOOOOO    44    44
PP      PP  AA      AA  SS              II      OO      OO  44    44
PP      PP  AA      AA  SS              II      OO      OO  44    44
PP      PP  AA      AA  SS              II      OO      OO  44    44
PP      PP  AA      AA  SS              II      OO      OO  44    44
PPPPPPPP    AA      AA    SSSSSS        II      OO      OO  444444444
PPPPPPPP    AA      AA    SSSSSS        II      OO      OO  444444444
PP          AAAAAAAAAA          SS      II      OO      OO        44
PP          AAAAAAAAAA          SS      II      OO      OO        44
PP          AA      AA          SS      II      OO      OO        44        ....
PP          AA      AA          SS      II      OO      OO        44        ....
PP          AA      AA  SSSSSSSS      IIIIII      OOOOOO          44        ....
PP          AA      AA  SSSSSSSS      IIIIII      OOOOOO          44        ....

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
```

```
0000    ;
0000    ;*****************************************************************************
0000    ;*                                                                           *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000    ;*  ALL RIGHTS RESERVED.                                                     *
0000    ;*                                                                           *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    ;*  TRANSFERRED.                                                             *
0000    ;*                                                                           *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    ;*  CORPORATION.                                                             *
0000    ;*                                                                           *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    ;*                                                                           *
0000    ;*                                                                           *
0000    ;*****************************************************************************
0000    ;
0000            .TITLE  PAS$IO_COMP             ; PASCAL RMS linkage
0000            .IDENT  'V04-000'
0000    ;
0000    ;*********************************************************************
0000    ;*********************************************************************
0000    ;**                                                                 **
0000    ;**              PASCAL RMS LINKAGE FOR VAX-11/780                  **
0000    ;**              ================================                   **
0000    ;**                                                                 **
0000    ;**                                                                 **
0000    ;**                VERSION V1.2 -- JANUARY 1981                     **
0000    ;**                                                                 **
0000    ;**   DEVELOPED BY:  COMPUTER SCIENCE DEPARTMENT                    **
0000    ;**                  UNIVERSITY OF WASHINGTON                       **
0000    ;**                  SEATTLE, WA 98195                              **
0000    ;**                                                                 **
0000    ;**   AUTHORS:       MARK BAILEY, JOHN CHAN, HELLMUT GOLDE          **
0000    ;**                                                                 **
0000    ;*********************************************************************
0000    ;*********************************************************************
0000    ;
0000    ; Modified: 16Oct80 Change the value of PRN_CRLF so that a line on the
0000    ;                   terminal is printed as <LF> <text> <CR> Susan Azibert
0000    ;
0000    ;           13Jan81 Change all tests of status from FMS to BLBC R0,label
0000    ;                   and BLBS R0,label from CMPL R0,#RMS$_NORMAL;BNEQ label,etc.
0000    ;
0000    ;       3. Fix PAS$CONCAT to request read-only access instead of read-write access
0000    ;
0000    ;       1-005 11-Aug-81 Paul Hohensee   Changed to general addressing of external ro
0000    ;
0000    ;*****************************************************************
0000    ;*****************************************************************
```

```
0000    ;**                                                                **
0000    ;**                                                                **
0000    ;**                    SECTION 4                                   **
0000    ;**                                                                **
0000    ;**              SPECIAL COMPILER PROCEDURES                       **
0000    ;**                                                                **
0000    ;**                                                                **
0000    ;*******************************************************************
0000    ;*******************************************************************
```

```
0000    ;For any file variable the following storage is assumed:
0000    ;
0000    ;                    ------------------
0000    ;          FSB:     |     POINTER      |
0000    ;                    ------------------
0000    ;                   |   STATUS WORD    |
0000    ;                    ------------------
0000    ;                   |      LAST        |
0000    ;                    ------------------
0000    ;                   |    LINELIMIT     |
0000    ;                    ------------------
0000    ;                   |    LINECOUNT     |
0000    ;                    ------------------
0000    ;                   |  RECORD NUMBER   |
0000    ;                    ------------------
0000    ;          RAB:     |                  |
0000    ;                     44(HEX) BYTES
0000    ;                   |        .         |
0000    ;                   |        .         |
0000    ;                   |        .         |
0000    ;                    ------------------
0000    ;          FAB:     |                  |
0000    ;                     50(HEX) BYTES
0000    ;                   |        .         |
0000    ;                   |        .         |
0000    ;                   |        .         |
0000    ;                    ------------------
0000    ;          NAM:     |                  |     NOTE: The NAM block is allocated
0000    ;                     38(HEX) BYTES          for the PASCAL logical files
0000    ;                   |        .         |     'INPUT' and 'OUTPUT' only.
0000    ;                   |        .         |
0000    ;                   |                  |
0000    ;                    ------------------
0000    ;
0000    ; Macro options
0000    ;
0000            .DSABL  GBL                     ; no undefined references
0000            .ENABL  FPT                     ; rounded arithmetic
0000    ;
0000    ; External references
0000    ;
0000            .EXTRN  PAS$IOERROR
0000            .EXTRN  PAS$WRITELN
0000            .EXTRN  PAS$OPEN
0000            .EXTRN  PAS$CLOSE
0000            .EXTRN PAS$CLOSEINOUT
0000            .EXTRN  PAS$RESET
0000    ;
0000            .EXTRN  SYS$PUTMSG              ; messages to terminal
0000            .EXTRN  SYS$TRNLOG
0000            .EXTRN  LIB$GET_VM
0000    ;
0000    ; Provide definitions of system values
0000    ;
0000            $FABDEF
0000            $NAMDEF
0000            $RABDEF
```

```
                0000          $RMSDEF                          ; for status code checking
                0000          $SSDEF                           ; system definitions
                0000   ;
                0000   ; PASCAL compiler constants
                0000   ;
                0000   ; NOTE: The constants below with the names 'PAS$C_XXXXX' are
                0000   ; used in the PASCAL compiler with the names 'XXXXX'. If the
                0000   ; values in the compiler are altered then the values below
                0000   ; must be altered accordingly.
                0000   ;
00000101        0000          PAS$C_DFLTRECSI = 257;           ; default buffer size
                0000   ;      PAS$C_NIL = 0                    ; NIL pointer
                0000   ;      PAS$C_TRUE = 1                   ; TRUE
                0000   ;      PAS$C_FALSE = 0                  ; FALSE
00000000        0000          PAS$C_NOCARR = 0                 ; no carriage control
                0000   ;      PAS$C_CARRIAGE = 1               ; FORTRAN carriage control
                0000   ;      PAS$C_LIST = 2                   ; LIST carriage control
                0000   ;      PAS$C_PRN = 3                    ; PRN carriage control
                0000   ;
                0000   ; PRN carriage control constants
                0000   ;
                0000   ;      PRN_CRLF = ^X8D01                ; PRN carriage control constant
                0000                                           ; for <LF> <text> <CR>
                0000   ;      PRN_NULL = ^X0000                ; PRN carriage control constant
                0000                                           ; for no carriage control
                0000   ;
                0000   ; File status block constants
                0000   ;
00000018        0000          FSB$C_BLN = ^X18                 ; FSB block length
                0000   ;      FSB$V_OPEN = 5
                0000   ;      FSB$V_EOF = 1
                0000   ;      FSB$V_EOLN = 2
                0000   ;      FSB$V_GET = 3
                0000   ;      FSB$V_TXT = 4                    ; textfile flag
                0000   ;      FSB$V_RDLN = 0                   ; last access READLN
                0000   ;      FSB$V_DIR = 6                    ; direct access flag
                0000   ;      FSB$V_PUT = 7
                0000   ;      FSB$V_INT = 8                    ; internal flag
                0000   ;      FSB$V_PRMT = 9                   ; prompt flag
                0000   ;      FSB$V_OUTPUT = 10                ; OUTPUT file flag
                0000   ;      FSB$V_ACTIN = 11                 ; actual input flag
                0000   ;      FSB$V_DELZ = 30                  ; delete file if empty
0000001F        0000          FSB$V_INC = 31                   ; included file flag
                0000   ;      FSB$B_CC = 6                     ; carriage control byte offset
                0000   ;      FSB$M_OPEN = ^X0020
00000002        0000          FSB$M_EOF = ^X0002
                0000   ;      FSB$M_EOLN = ^X0004
                0000   ;      FSB$M_GET = ^X0008
                0000   ;      FSB$M_PRMT = ^X0200
                0000   ;      FSB$M_PUT = ^X00000080
00000010        0000          FSB$M_TXT = ^X0010
00000001        0000          FSB$M_RDLN = ^X0001
                0000   ;      FSB$M_DIR = ^X00000040
                0000   ;      FSB$M_INT = ^X00000100
                0000   ;      FSB$M_OUTPUT = ^X0400
                0000   ;      FSB$M_ACTIN = ^X0800
40000000        0000          FSB$M_DELZ = ^X40000000
```

```
80000000  0000          FSB$M_INC = ^X80000000
          0000  ;       FSB$L_CNT = 16                  ; line count (textfiles)
00000014  0000          FSB$L_INC = 20                  ; %INCLUDE block address
          0000  ;       FSB$L_LIM = 12                  ; LINELIMIT
          0000  ;       FSB$L_LST = 8                   ; LAST WORD OFFSET
          0000  ;       FSB$L_PFSB = 20                 ; Related file FSB for prompting
          0000                                          ; for INPUT, has address of OUTPUT FSB
          0000                                          ; for OUTPUT, has address of INPUT FSB
          0000                                          ; (shares storage with include address
          0000                                          ; and direct access record
          0000                                          ; buffer address)
          0000  ;       FSB$L_REC = 20                  ; record buffer address for
          0000                                          ; direct access (shares storage
          0000                                          ; with include address and related
          0000                                          ; file FSB)
00000004  0000          FSB$L_STA = 4                   ; status word offset
          0000  ;
          0000  ; Character constants
          0000  ;
          0000  ;       TAB = ^X09
00000020  0000          SPACE = ^X20
          0000  ;       DOLLAR = ^X24
          0000  ;       FORMFEED = ^XC
          0000  ;       STAR = ^X2A
          0000  ;       PLUS = ^X2B
          0000  ;       MINUS = ^X2D
          0000  ;       POINT = ^X2E
          0000  ;       ZERO = ^X30
          0000  ;       ONE = ^X31
          0000  ;       NINE = ^X39
          0000  ;       AA = ^X41
          0000  ;       DD = ^X44
          0000  ;       EE = ^X45
          0000  ;       ZZ = ^X5A
          0000  ;       UNDERSCORE = ^X5F
          0000  ;       AA_SMALL = ^X61
          0000  ;       ZZ_SMALL = ^X7A
          0000  ;
          0000  ;
00000000  0000          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
          0000  ;
          0000  ;       ********************
          0000  ;       *                  *
          0000  ;       *   PAS$COMPINIT   *
          0000  ;       *                  *
          0000  ;       ********************
          0000  ;
          0000  ; Initializes the file control blocks for certain files used by the
          0000  ; compiler.  Specifically, it
          0000  ;
          0000  ;       (1) clears the RDLN flag for the input file so that PAS$EOF and
          0000  ;           PAS$EOLN will never retrieve the next record
          0000  ;
          0000  ;       (2) sets a flag to delete the output file if it is empty
          0000  ;
          0000  ;       (3) clears the textfile flag for the object file
          0000  ;
```

```
                                  0000  ; In addition, determines if the listing is going to the terminal.  If
                                  0000  ; the listing is going to the terminal, the fourth (return) argument is
                                  0000  ; set to 1, otherwise it is set to 0.
                                  0000  ;
                                  0000  ; Argument offsets
                                  0000  ;
                                  0000  ;       AP                              ; number of arguments (4)
                     00000004     0000          INP_DISP = 04                   ; input file FSB
                     00000008     0000          OUT_DISP = 08                   ; output file FSB
                     0000000C     0000          OBJ_DISP = 12                   ; object file FSB
                     00000010     0000          TRM_DISP = 16                   ; list_to_terminal flag (by reference)
                                  0000  ;
                                  0000  ; Other constants
                                  0000  ;
                                  0000  PASOUT:
54 55 50 54 55 4F 24 53 41 50     0000          .ASCII  /PAS$OUTPUT/            ; listing file logical name
                                  000A  SYSOUT:
                          000A'   000A          .WORD   20$-10$                 ; length of string
54 55 50 54 55 4F 24 53 59 53     000C  10$:    .ASCII  /SYS$OUTPUT/            ; standard output logical name
                                  0016  20$:
                                  0016  SYSERR:
                          0009'   0016          .WORD   40$-30$                 ; length of string
   52 4F 52 52 45 24 53 59 53     0018  30$:    .ASCII  /SYS$ERROR/             ; error output logical name
                                  0021  40$:
                          00FC    0021          .ENTRY  PAS$COMPINIT,^M<R2,R3,R4,R5,R6,R7>
        5E   00000088 8F    C2    0023          SUBL2   #136,SP                 ; make room for translated names
              56    04 AC   D0    002A          MOVL    INP_DISP(AP),R6
              04 A6    01   CA    002E          BICL2   #FSB$M_RDLN,FSB$L_STA(R6); clear input RDLN flag
              56    0C AC   D0    0032          MOVL    OBJ_DISP(AP),R6
              04 A6    10   CA    0036          BICL2   #FSB$M_TXT,FSB$L_STA(R6) ; clear textfile flag
              56    08 AC   D0    003A          MOVL    OUT_DISP(AP),R6
     04 A6   40000000 8F    C8    003E          BISL2   #FSB$M_DELZ,FSB$L_STA(R6); set to delete if empty
                                  0046  ;
                                  0046  ; Now determine if listing is going to terminal.
                                  0046  ;
  BC AD    B6 AF    0A    28      0046          MOVC3   #10,PASOUT,-68(FP)      ; move listing logical name
           FC AD    0A    D0      004C          MOVL    #10,-4(FP)              ; set length of logical name
                     00   DD      0050          PUSHL   #0                      ; arg3 (process-permanent flag)
                                  0052                                          ; is set to false
              FC AD    DF         0052          PUSHAL  -4(FP)                  ; arg2 (address of length)
              BC AD    DF         0055          PUSHAL  -68(FP)                 ; arg1 (address of string)
    000000A3'EF    03    FB       0058          CALLS   #3,PAS$TRANS            ; translate logical name
                                  005F  ;
                                  005F  ;  Loop through names SYS$OUTPUT and SYS$ERROR, translating them and
                                  005F  ;  comparing the result to the translation of PAS$OUTPUT.  If either
                                  005F  ;  is equal, return TRUE; otherwise return FALSE.
                                  005F  ;
              56    02    D0      005F          MOVL    #2,R6                   ; loop index
           57    A5 AF    DE      0062          MOVAL   SYSOUT,R7               ; address of logical name string
                                  0066  COMPLOOP:                               ; compare translated names
FF78 CD    02 A7    67    28      0066          MOVC3   (R7),2(R7),-136(FP)     ; move logical name
           B8 AD    67    3C      006D          MOVZWL  (R7),-72(FP)            ; set length of logical name
                     01   DD      0071          PUSHL   #1                      ; arg3 (process-permanent flag)
                                  0073                                          ; is set to TRUE
              B8 AD    DF         0073          PUSHAL  -72(FP)                 ; arg2 (address of length)
           FF78 CD    DF         0076          PUSHAL  -136(FP)                ; arg1 (address of string)
    000000A3'GF    03    FB       007A          CALLS   #3,G^PAS$TRANS          ; translate logical name
```

```
                B8 AD    FC AD    D1  0081          CMPL    -4(FP),-72(FP)            ; are lengths equal?
                              0F    12  0086          BNEQ    TRYNXT                    ; no--try next logical name
        FF78 CD    BC AD    FC AD    29  0088          CMPC3   -4(FP),-68(FP),-136(FP)   ; else, are strings equal?
                              05    12  0090          BNEQ    TRYNXT                    ; no--try next logical name
                   10 BC     01    D0  0092          MOVL    #1,@TRM_DISP(AP)          ; yes--set return value TRUE
                              04    0096              RET                               ; return TRUE to caller
                                    0097  TRYNXT:                                       ; try next logical name
                 57    FF7B CF    DE  0097          MOVAL   SYSERR,R7                 ; address of next logical name
                          C7 56    F5  009C          SOBGTR  R6,COMPLOOP               ; loop through all logical names
                          10 BC    D4  009F          CLRL    @TRM_DISP(AP)             ; set return value FALSE
                              04    00A2              RET                               ; return FALSE to caller
                                    00A3  ;
                                    00A3  ;  Routine to translate logical names, used by PAS$COMPINIT above.
                                    00A3  ;
                                    00A3  PAS$TRANS:                                    ; routine to translate names
                             0700   00A3              .WORD   ^M<R8,R9,R10>
                       5E    10     C2  00A5          SUBL2   #16,SP                    ; make room for $TRNLOG args
                FC AD       04 AC  D0  00A8          MOVL    4(AP),-4(FP)              ; address of string
                F8 AD       08 BC  D0  00AD          MOVL    @8(AP),-8(FP)             ; length of logical name
                F4 AD       04 AC  D0  00B2          MOVL    4(AP),-12(FP)             ; address of result
                F0 AD       50 8F  9A  00B7          MOVZBL  #80,-16(FP)               ; length of result
                                58  D4  00BC          CLRL    R8                        ; clear colon flag
                                    00BE  TRANS:      $TRNLOG_S LOGNAM=-8(FP),-         ; translate the name
                                    00BE                        RSLLEN=@8(AP),-
                                    00BE                        RSLBUF=-16(FP)
                          1B 58     E9  00D4          BLBC    R8,CHKCOL                 ; branch if not trying without colon
                00000629 8F    50  D1  00D7          CMPL    R0,#SS$_NOTRAN            ; was translation successful?
                               0E  12  00DE          BNEQ    10$                       ; yes--continue
                       59   08 BC  D0  00E0          MOVL    @8(AP),R9                 ; else put colon back and quit
                    04 BC49    3A  90  00E4          MOVB    #^A/:/,@4(AP)[R9]         ; put colon back
                 08 BC    59    01  C1  00E9          ADDL3   #1,R9,@8(AP)              ; set return length
                               58  D4  00EE  10$:    CLRL    R8                        ; not trying without colon any more
                               23  11  00F0          BRB     CHKHDR                    ; go check for 4-byte header
                                    00F2  CHKCOL:                                       ; check if translate failed due to colon
                       1F 0C AC    E8  00F2          BLBS    12(AP),CHKHDR             ; if process-perm name, continue
                00000629 8F    50  D1  00F6          CMPL    R0,#SS$_NOTRAN            ; translation successful?
                               16  12  00FD          BNEQ    CHKHDR                    ; yes--continue
                       59   08 BC  D0  00FF          MOVL    @8(AP),R9                 ; translated length
                               59  D7  0103          DECL    R9                        ; index of last character
                    3A   04 BC49   91  0105          CMPB    @4(AP)[R9],#^A/:/         ; last character a colon?
                               09  12  010A          BNEQ    CHKHDR                    ; no--continue
                          08 BC    D7  010C          DECL    @8(AP)                    ; else remove colon and try again
                       58    01    D0  010F          MOVL    #1,R8                     ; set flag to show trying again
                       50    01    D0  0112          MOVL    #1,R0                     ; set status to continue
                                    0115  CHKHDR:                                       ; check for 4-byte header
                    1B   04 BC     91  0115          CMPB    @4(AP),#^X1B              ; translated name begins with esc?
                               2F  12  0119          BNEQ    CHKPRM                    ; no--make sure not process-perm
                       59    04    D0  011B          MOVL    #4,R9                     ; initialize number of bytes to remove
                 5F 8F    04 BC49   91  011E          CMPB    @4(AP)[R9],#^A/_/         ; translated name begins with '_'?
                               02  12  0124          BNEQ    10$                       ; no--remove just first 4 bytes
                               59  D6  0126          INCL    R9                        ; else remove 5 bytes
                 5A    08 BC    59  C3  0128  10$:    SUBL3   R9,@8(AP),R10             ; r10 <- new string length
        04 BC    04 BC49    5A  28  012D          MOVC3   R10,@4(AP)[R9],@4(AP)     ; move remaining chars to front
                               5A  D7  0134          DECL    R10                       ; r10 <- index of last character
                    3A   04 BC4A   91  0136          CMPB    @4(AP)[R10],#^A/:/        ; name ends with colon?
                               07  13  013B          BEQL    20$                       ; yes--continue
                               5A  D6  013D          INCL    R10                       ; else add colon at end
```

```
      04 BC4A   3A   90  013F          MOVB    #^A/:/,@4(AP)[R10]
   08 BC   5A   01   C1  0144  20$:    ADDL3   #1,R10,@8(AP)          ; update length
                     04  0149          RET                            ; return
                         014A  CHKPRM:                               ; check if process permanent name
         11 0C AC   E8  014A          BLBS    12(AP),RET             ; return if process permanent name
00000629 8F    50   D1  014E          CMPL    R0,#SS$_NOTRAN         ; translation was successful?
               08   13  0155          BEQL    RET                    ; no--return
   F8 AD   08 BC   D0  0157          MOVL    @8(AP),-8(FP)          ; update length of $TRNLOG arg
           FF5F    31  015C          BRW     TRANS                  ; translate recursively
               04  C15F  RET:        RET                            ; return to caller
                         0160  ;
                         0160  ;
                 00000160          .PSECT  _PAS$CODE,             PIC,EXE,SHR,NOWRT
                         0160  ;
                         0160  ;       ********************
                         0160  ;       *                  *
                         0160  ;       *  PAS$READINPUT   *
                         0160  ;       *                  *
                         0160  ;       ********************
                         0160  ;
                         0160  ; Reads the next record into the user supplieed buffer returning the
                         0160  ; length read. The end-of-file flag is altered as need be.
                         0160  ;
                         0160  ; Argument offsets
                         0160  ;
                         0160  ;       AP                             ; number of arguments (4)
                 00000004 0160          FSB_DISP = 04                  ; FSB address
                 00000008 0160          BUF_DISP = 08                  ; buffer address
                 0000000C 0160          LEN_DISP = 12                  ; length of buffer (by reference)
                 00000010 0160          SOS_DISP = 16                  ; SOS line buffer (by reference)
                 00000014 0160          incl_eof_disp = 20             ; eof on INCLUDE'd file flag
                         0160                                         ; (byte by ref)
                         0160                                         ;    0 = no eof
                         0160                                         ;    1 = eof hit
                         0160  ;
                     003C 0160          .ENTRY  PAS$READINPUT,^M<R2,R3,R4,R5>
           14 BC   94  0162          clrb    @incl_eof_disp(ap)     ; assume no eof
      54   04 AC   D0  0165          MOVL    FSB_DISP(AP),R4        ; R4 = address of FSB
   52   18   54   C1  0169          ADDL3   R4,#FSB$C_BLN,R2       ; R2 = address of RAB
55 00000044 8F   52   C1  016D          ADDL3   R2,#RAB$C_BLN,R5       ; R5 = address of FAB
      53   24 A2   D0  0175          MOVL    RAB$L_UBF(R2),R3       ; save buffer address
   24 A2   08 AC   D0  0179          MOVL    BUF_DISP(AP),RAB$L_UBF(R2); set new buffer address
      02   3F A5   91  017E          CMPB    FAB$B_FSZ(R5),#2       ; check for SOS line numbers
               07   12  0182          BNEQ    120$
   2C A2   10 AC   D0  0184          MOVL    SOS_DISP(AP),RAB$L_RHB(R2)
               03   11  0189          BRB     130$
                         018B  120$:
           2C A2   D4  018B          CLRL    RAB$L_RHB(R2)
                         018E  130$:
                         018E          $GET    RAB=R2
0001827A 8F    50   D1  0197          CMPL    R0,#RMS$_EOF
               18   12  019E          BNEQ    110$
   0D 04 A4    1F   E1  01A0          BBC     #FSB$V_INC,FSB$L_STA(R4),100$
0000037B'EF    6C   FA  01A5          CALLG   (AP),PAS$EXCLUDE
      14 BC    01   90  01AC          movb    #1,@incl_eof_disp(ap)  ; set eof flag
               09   11  01B0          BRB     111$
                         01B2  100$:
```

```
04 A4    02   C8  01B2   410          BISL    #FSB$M_EOF,FSB$L_STA(R4); set EOF flag
         03   11  01B6   411          BRB     111$
                 01B8   412   110$:
      0A 50   E9  01B8   413          BLBC    R0,910$                   ; branch if error
                 01BB   414   111$:
0C BC  22 A2   32  01BB   415          CVTWL   RAB$W_RSZ(R2),@LEN_DISP(AP); store length
   24 A2   53  D0  01C0   416          MOVL    R3,RAB$L_UBF(R2)          ; restore buffer address
           04  01C4   417          RET
               01C5   418 ;
               01C5   419 ; Input error
               01C5   420 ;
               01C5   421   910$:
      50   DD  01C5   422          PUSHL   R0
   7E 78 A4   9A  01C7   423          MOVZBL  <RAB$C_BLN+FAB$B_FNS>(R4),-(SP)
      70 A4   DD  01CB   424          PUSHL   <RAB$C_BLN+FAB$L_FNA>(R4)
00000000'GF   03  FB  01CE   425          CALLS   #3,G^PAS$IOERROR
               01D5   426 ;
               01D5   427 ;
           000001D5   428          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
               01D5   429 ;
               01D5   430 ;        ********************
               01D5   431 ;        *                  *
               01D5   432 ;        * PAS$WRITEOUTPUT  *
               01D5   433 ;        * PAS$WRITEOBJ     *
               01D5   434 ;        *                  *
               01D5   435 ;        ********************
               01D5   436 ;
               01D5   437 ; Writes the string to the designated file. The buffer is NOT checked
               01D5   438 ; for overflow.
               01D5   439 ;
               01D5   440 ; Argument list
               01D5   441 ;
               01D5   442 ;        AP                              ; number of arguments (3)
         00000004  01D5   443          FSB_DISP = 04                   ; FSB address
         00000008  01D5   444          BUF_DISP = 08                   ; buffer address
         0000000C  01D5   445          LEN_DISP = 12                   ; buffer length (by reference)
               01D5   446  PAS$WRITEOBJ::
           007C  01D5   447          .ENTRY  PAS$WRITEOUTPUT,^M<R2,R3,R4,R5,R6>
      56   04 AC   D0  01D7   448          MOVL    FSB_DISP(AP),R6         ; R6 = address of FSB
00 B6  08 BC   0C BC   28  01DB   449          MOVC3   @LEN_DISP(AP),@BUF_DISP(AP),@(R6); move string
         66   53  D0  01E2   450          MOVL    R3,(R6)                 ; set pointer
         56   DD  01E5   451          PUSHL   R6
00000000'GF   01  FB  01E7   452          CALLS   #1,G^PAS$WRITELN
           04  01EE   453          RET
               01EF   454 ;
               01EF   455 ;
           000001EF   456          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
               01EF   457 ;
               01EF   458 ;        ********************
               01EF   459 ;        *                  *
               01EF   460 ;        *   PAS$PUTMSG     *
               01EF   461 ;        *                  *
               01EF   462 ;        ********************
               01EF   463 ;
               01EF   464 ; Writes a string to the terminal using the SYS$PUTMSG system service.
               01EF   465 ; A Pascal file (FSB) is passed as argument, and the string is stolen
               01EF   466 ; from its buffer.  If the fourth argument is true the buffer is cleared,
```

```
                                01EF   467 ; otherwise it is left alone.  As for PAS$WRITEOUTPUT, buffer overflow
                                01EF   468 ; is NOT checked.
                                01EF   469 ;
                                01EF   470 ; Argument list
                                01EF   471 ;
                                01EF   472 ;        AP                                  ; number of arguments (4)
                   00000004     01EF   473          FSB_DISP = 04                       ; FSB address
                   00000008     01EF   474          BUF_DISP = 08                       ; buffer address
                   0000000C     01EF   475          LEN_DISP = 12                       ; buffer length (by reference)
                   00000010     01EF   476          RST_DISP = 16                       ; reset flag (by value)
                                01EF   477                                              ; 0: do not reset line to be empty
                                01EF   478                                              ; 1: reset line after sending to terminal
                                01EF   479 ;
                        007C    01EF   480          .ENTRY  PAS$PUTMSG,^M<R2,R3,R4,R5,R6>
                 5E   18   C2   01F1   481          SUBL2   #24,SP                      ; room for message vector
                                01F4   482                                              ; and descriptor
              56   04 AC   D0   01F4   483          MOVL    FSB_DISP(AP),R6             ; R6 = address of FSB
        00 B6   08 BC   0C BC   28   01F8   484          MOVC3   @LEN_DISP(AP),@BUF_DISP(AP),@(R6); move string
                 66      53   D0   01FF   485          MOVL    R3,(R6)                     ; set pointer
     E8 AD   00010003 8F   D0   0202   486          MOVL    #^X10003,-24(FP)            ; vector length and flags
     EC AD   00801134 8F   D0   020A   487          MOVL    #^X801134,-20(FP)           ; message identification
              F0 AD      01   D0   0212   488          MOVL    #1,-16(FP)                  ; number of fao arguments
           F4 AD   F8 AD   DE   0216   489          MOVAL   -8(FP),-12(FP)              ; address of descriptor
                   40 A6   C3   021B   490          SUBL3   <FSB$C_BLN+RAB$L_RBF>(R6),-; compute and store length
              F8 AD      53      021E   491                   R3,-8(FP)                  ; in descriptor
                   F8 AD   D7   0221   492          DECL    -8(FP)                      ; skip carriage control byte
                   40 A6   D0   0224   493          MOVL    <FSB$C_BLN+RAB$L_RBF>(R6),-; move address to descriptor
                   FC AD        0227   494                   -4(FP)
                   FC AD   D6   0229   495          INCL    -4(FP)                      ; skip carriage control byte
                      7E   7C   022C   496          CLRQ    -(SP)                       ; arg2 and arg3 are NULL
                   E8 AD   DF   022E   497          PUSHAL  -24(FP)                     ; arg1: address of message vector
   00000000'GF        03   FB   0231   498          CALLS   #3,G^SYS$PUTMSG             ; send message to terminal
            04 10 AC   E9   0238   499          BLBC    RST_DISP(AP),10$            ; done if not to reset
                   40 A6   D0   023C   500          MOVL    <FSB$C_BLN+RAB$L_RBF>(R6),-; reset line to be empty
                      66        023F   501                   (R6)
                      04        0240   502 10$:     RET                                 ; all done
                                0241   503 ;
                                0241   504 ;
                   00000241     0241   505          .PSECT  _PAS$CODE,                  PIC,EXE,SHR,NOWRT
                                0241   506 ;
                                0241   507 ;        ********************
                                0241   508 ;        *                  *
                                0241   509 ;        *     PAS$TAB      *
                                0241   510 ;        *                  *
                                0241   511 ;        ********************
                                0241   512 ;
                                0241   513 ; Advances the pointer to a new position blank filling the spaces
                                0241   514 ; skipped. If the tab position is either less than the current position
                                0241   515 ; or greater than the end-of-line position the tab is ignored.
                                0241   516 ;
                                0241   517 ; Argument offsets
                                0241   518 ;
                                0241   519 ;        AP                                  ; number of arguments (2)
                   00000004     0241   520          FSB_DISP = 04                       ; FSB address
                   00000008     0241   521          TAB_DISP = 08                       ; TAB position
                                0241   522 ;
                        007C    0241   523          .ENTRY  PAS$TAB,^M<R2,R3,R4,R5,R6>
```

```
                56   04 AC   D0  0243   524          MOVL    FSB_DISP(AP),R6          ; address of next byte
             53   18   56   C1  0247   525          ADDL3   R6,#FSB$C_BLN,R3         ; address of RAB to R3
          50   66   28 A3   C3  024B   526          SUBL3   RAB$L_RBF(R3),(R6),R0    ; old position to R0
             51   08 AC   D0  0250   527          MOVL    TAB_DISP(AP),R1          ; new position
             50   51   50   C3  0254   528          SUBL3   R0,R1,R0                 ; amount to move pointer
                       13   15  0258   529          BLEQ    $420                     ; done if negative
          00000101 8F   51   D1  025A   530          CMPL    R1,#PAS$C_DFLTRECSI      ; check if too far to go
                       0A   14  0261   531          BGTR    $420                     ; ignore request
   00 B6   50   20   66   00   2C  0263   532          MOVC5   #0,(R6),#SPACE,R0,@(R6) ; move blanks
                   66   53   D0  026A   533          MOVL    R3,(R6)                  ; update pointer
                             026D   534 $420:
                        04   026D   535          RET                              ; done
                             026E   536 ;
                             026E   537 ;
                        0000026E   538          .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                             026E   539 ;
                             026E   540 ;          ********************
                             026E   541 ;          *                  *
                             026E   542 ;          *   PAS$INCLUDE    *
                             026E   543 ;          *                  *
                             026E   544 ;          ********************
                             026E   545 ;
                             026E   546 ; Saves the necessary information about the current file so that it may
                             026E   547 ; be restored, closes the current file, and opens/resets the included
                             026E   548 ; file. The included block has the following storage allocation:
                             026E   549 ;
                             026E   550 ;           3322222222221111111111
                             026E   551 ;           1098765432109876543210987654321 0
                             026E   552 ;           --------------------------------
                             026E   553 ;          |   next include block address   |
                             026E   554 ;           --------------------------------
                             026E   555 ;          |   current position in buffer   |
                             026E   556 ;           --------------------------------
                             026E   557 ;          |       record file address      |
                             026E   558 ;           ----------------
                             026E   559 ;          |                |               |
                             026E   560 ;           -               ----------------
                             026E   561 ;          |      directory identification  |
                             026E   562 ;           --------------------------------
                             026E   563 ;          |                                |
                             026E   564 ;           -                              -
                             026E   565 ;          |            device              |
                             026E   566 ;           -                              -
                             026E   567 ;          |         identification         |
                             026E   568 ;           -                              -
                             026E   569 ;          |                                |
                             026E   570 ;           --------------------------------
                             026E   571 ;          |     file identification        |
                             026E   572 ;           ----------------               -
                             026E   573 ;                          |               |
                             026E   574 ;                           ----------------
                             026E   575 ;
                 0000002A    026E   576          INC$C_BLN = 42
                 00000000    026E   577          INC$L_NXT = 00
                 00000004    026E   578          INC$L_POS = 04
                 00000008    026E   579          INC$T_RFA = 08
                 0000000E    026E   580          INC$T_DID = 14
```

```
                    00000014  026E        INC$X_DVI = 20
                    00000024  026E        INC$T_FID = 36
                              026E  ;
                              026E  ; Argument offsets
                              026E  ;
                              026E  ;     AP                          ; number of arguments (7)
                    00000004  026E        FSB_DISP = 04               ; FSB address
                    00000008  026E        BUF_DISP = 08               ; character buffer address
                    0000000C  026E        POS_DISP = 12               ; current buffer position (by value)
                    00000010  026E        LST_DISP = 16               ; last buffer position (by reference)
                    00000014  026E        spec_disp = 20              ; include specification string length
                              026E                                    ;       (by value)
                    00000018  026E        LEN_DISP = 24               ; file name string length (by value)
                    0000001C  026E        INC_DISP = 28               ; recursive include flag
                              026E                                    ;    0 => ok
                              026E                                    ;    1 => recursive
                              026E  ;
                              026E  ; Constants
                              026E  ;
                    0000002A  026E  INCLBLN: .LONG  INC$C_BLN
                              0272  ;
                        07FC  0272        .ENTRY  PAS$INCLUDE,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
                              0274
            56   04 AC   D0   0274        MOVL    FSB_DISP(AP),R6     ; R6 = address of FSB
         57   18   56   C1   0278        ADDL3   R6,#FSB$C_BLN,R7    ; R7 = address of RAB
58  00000044 8F   57   C1   027C        ADDL3   R7,#RAB$C_BLN,R8    ; R8 = address of FAB
59  00000050 8F   58   C1   0284        ADDL3   R8,#FAB$C_BLN,R9    ; R9 = address of NAM
                              028C  ;
                              028C  ; Allocate memory for include block
                              028C  ;
                 5E   04   C2 028C        SUBL2   #4,SP
                 5A   5E   D0 028F        MOVL    SP,R10              ; R10 = address of memory address
                      5A   DD 0292        PUSHL   R10
                   D7 AF   DF 0294        PUSHAL  INCLBLN
         00000000'GF   02   FB 0297        CALLS   #2,G^LIB$GET_VM
                   03 50   E8 029E        BLBS    R0,101$
                    0091   31 02A1        BRW     910$
                              02A4  ;
                              02A4  ; Save include information
                              02A4  ;
                              02A4    101$:
                 5A   6A   D0 02A4        MOVL    (R10),R10           ; R10 = address of include block
            6A   14 A6   D0 02A7        MOVL    FSB$L_INC(R6),INC$L_NXT(R10)
         04 AA   0C AC   D0 02AB        MOVL    POS_DISP(AP),INC$L_POS(R10)
      08 AA   10 A7   06   28 02B0        MOVC3   #06,RAB$W_RFA(R7),INC$T_RFA(R10)
      0E AA   2A A9   06   28 02B6        MOVC3   #06,NAM$W_DID(R9),INC$T_DID(R10)
      14 AA   14 A9   10   28 02BC        MOVC3   #16,NAM$T_DVI(R9),INC$X_DVI(R10)
      24 AA   24 A9   06   28 02C2        MOVC3   #06,NAM$W_FID(R9),INC$T_FID(R10)
                              02C8  ;
                              02C8  ; Blank out remaining part of line
                              02C8  ;
         51   0C AC   01   C3 02C8        SUBL3   #1,POS_DISP(AP),R1  ; R1 = substring offset
      50   10 BC   0C AC   C3 02CD        SUBL3   POS_DISP(AP),@LST_DISP(AP),R0
                      50   D6 02D3        INCL    R0                  ; R0 = substring length
08 BC41   50   20   66   00   2C 02D5        MOVC5   #0,(R6),#SPACE,R0,@BUF_DISP(AP)[R1]; blank out substring
                              02DD  ;
                              02DD  ; Fix FSB
```

```
                                02DD   638 ;
            14 A6    5A   D0    02DD   639          MOVL    R10,FSB$L_INC(R6)
     04 A6  80000000 8F   C8    02E1   640          BISL2   #FSB$M_INC,FSB$L_STA(R6); set include flag
                                02E9   641 ;
                                02E9   642 ; Close current file and open included file
                                02E9   643 ;
                     56   DD    02E9   644          PUSHL   R6
     00000000'GF     01   FB    02EB   645          CALLS   #1,G^PAS$CLOSEINOUT
                                02F2   646 ;
                     00   DD    02F2   647          PUSHL   #PAS$C_NOCARR           ; carriage control -- not used
          FFFFFEFF   8F   DD    02F4   648          PUSHL   #-PAS$C_DFLTRECSI       ; no buffer allocated
                     7E   7C    02FA   649          CLRD    -(SP)                   ; variable length, sequential access
                 18  AC   DD    02FC   650          PUSHL   LEN_DISP(AP)            ; file name string length
                 0C  AC   D7    02FF   651          DECL    POS_DISP(AP)
     50    0C AC    14  AC   C3  0302   652          SUBL3   spec_DISP(AP),POS_DISP(AP),R0
                     50   D7    0308   653          DECL    R0
               08 BC40    9F    030A   654          PUSHAB  @BUF_DISP(AP)[R0]       ; file name string address
                     56   DD    030E   655          PUSHL   R6                      ; fsb address
     00000000'GF     07   FB    0310   656          CALLS   #7,G^PAS$OPEN
                                0317   657 ;
                                0317   658 ; Check for recursive includes
                                0317   659 ;
                 14  A6   DD    0317   660          PUSHL   FSB$L_INC(R6)           ; head of include list
                     59   DD    031A   661          PUSHL   R9                      ; address of NAM block
     0000034A'EF     02   FB    031C   662          CALLS   #2,PAS$INCHECK
           1C BC     50   90    0323   663          MOVb    R0,@INC_DISP(AP)        ; return recursive flag
                                0327   664 ;
                                0327   665 ; Reset input file
                                0327   666 ;
                     56   DD    0327   667          PUSHL   R6
     00000000'GF     01   FB    0329   668          CALLS   #1,G^PAS$RESET
           04 A6     01   CA    0330   669          BICL2   #FSB$M_RDLN,FSB$L_STA(R6); clear RDLN flag
                     04         0334   670          RET
                                0335   671 ;
                                0335   672 ; Heap overflow on include
                                0335   673 ;
                                0335   674   910$:
                     50   DD    0335   675          PUSHL   R0
       7E     8324   8F   3C    0337   676          MOVZWL  #^X8324,-(SP)
         7E     34   A8   9A    033C   677          MOVZBL  FAB$B_FNS(R8),-(SP)
                 2C  A8   DD    0340   678          PUSHL   FAB$L_FNA(R8)
     00000000'GF     04   FB    0343   679          CALLS   #4,G^PAS$IOERROR
                                034A   680 ;
                                034A   681 ;
                                0000034A   682          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
                                034A   683 ;
                                034A   684 ;        ********************
                                034A   685 ;        *                  *
                                034A   686 ;        *   PAS$INCHECK    *
                                034A   687 ;        *                  *
                                034A   688 ;        ********************
                                034A   689 ;
                                034A   690 ; Checks for recursive included files by traversing the linked list
                                034A   691 ; of included files.
                                034A   692 ;
                                034A   693 ; Argument offsets
                                034A   694 ;        AP                              ; number of arguments (2)
```

```
                    00000004  034A          NAM_DISP = 04                    ; current input file name block
                    00000008  034A          LNK_DISP = 08                    ; head of included file list
                              034A  ;
                        00C0  034A          .ENTRY  PAS$INCHECK,^M<R6,R7>
          56   04 AC    D0   034C          MOVL    NAM_DISP(AP),R6          ; R6 = address of NAM block
          57   08 AC    D0   0350          MOVL    LNK_DISP(AP),R7          ; R7 = address of first include block
               03       11   0354          BRB     130$
                             0356  120$:
             57   67    D0   0356          MOVL    INC$L_NXT(R7),R7         ; get next include block
                             0359  130$:
               50       D4   0359          CLRL    R0                       ; set R0 to FALSE
               57       D5   035B          TSTL    R7                       ; any more?
               1B       13   035D          BEQL    899$
   14 A7   14 A6       10   035F          CMPC    #16,NAM$T_DVI(R6),INC$X_DVI(R7); check device
               EF       12   0365          BNEQ    120$
   0E A7   2A A6       06   0367          CMPC    #06,NAM$W_DID(R6),INC$T_DID(R7); check directory
               E7       12   036D          BNEQ    120$
   24 A7   24 A6       06   036F          CMPC    #06,NAM$W_FID(R6),INC$T_FID(R7); check file id
               DF       12   0375          BNEQ    120$
             50   01    D0   0377          MOVL    #1,R0                    ; set R0 to TRUE
                             037A  899$:
                        04   037A          RET
                             037B  ;
                             037B  ;
                  0000037B   037B          .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                             037B  ;
                             037B  ;       ********************
                             037B  ;       *                  *
                             037B  ;       *   PAS$EXCLUDE    *
                             037B  ;       *                  *
                             037B  ;       ********************
                             037B  ;
                             037B  ; Called when an end-of-file is encountered on an included file. The
                             037B  ; current file is closed and the previous file is opened and set to
                             037B  ; the last record access on the previous file.
                             037B  ;
                             037B  ; Argument offsets
                             037B  ;
                             037B  ;       AP                               ; number of arguments (3)
                    00000004 037B          FSB_DISP = 04                    ; FSB address
                    00000008 037B          BUF_DISP = 08                    ; character buffer
                    0000000C 037B          LST_DISP = 12                    ; last buffer character
                             037B  ;
                        07FC 037B          .ENTRY  PAS$EXCLUDE,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
          56   04 AC    D0   037D          MOVL    FSB_DISP(AP),R6
     57   18   56       C1   0381          ADDL3   R6,#FSB$C_BLN,R7         ; R7 = address of RAB
58   00000044 8F   57       C1   0385      ADDL3   R7,#RAB$C_BLN,R8         ; R8 = address of FAB
59   00000050 8F   58       C1   038D      ADDL3   R8,#FAB$C_BLN,R9         ; R9 = address of NAM
                             0395  ;
                             0395  ; Close current file
                             0395  ;
               56       DD   0395          PUSHL   R6
     00000000'GF   01       FB   0397      CALLS   #1,G^PAS$CLOSEINOUT
                             039E  ;
                             039E  ; Restore NAM block and open previous file by NAM block
                             039E  ;
          5A   14 A6    D0   039E          MOVL    FSB$L_INC(R6),R10        ; R10 = address of include block
```

```
                14 A6   6A   D0  03A2   752          MOVL    INC$L_NXT(R10),FSB$L_INC(R6); restore next pointer
         2A A9  0E AA   06   28  03A6   753          MOVC3   #6,INC$T_DID(R10),NAM$W_DID(R9); directory id
         14 A9  14 AA   10   28  03AC   754          MOVC3   #16,INC$X_DVI(R10),NAM$T_DVI(R9); device id
         24 A9  24 AA   06   28  03B2   755          MOVC3   #06,INC$T_FID(R10),NAM$W_FID(R9); file id
      04 A8   01000000  8F   C8  03B8   756          BISL2   #FAB$M_NAM,FAB$L_FOP(R8); open via NAM block
                        00   DD  03C0   757          PUSHL   #PAS$C_NOCARR           ; carriage control -- not used
              FFFFFEFF  8F   DD  03C2   758          PUSHL   #-PAS$C_DFLTRECSI       ; no buffer allocated
                        7E   7C  03C8   759          CLRD    -(SP)
                        7E   7C  03CA   760          CLRD    -(SP)
                        56   DD  03CC   761          PUSHL   R6
         00000000'GF    07   FB  03CE   762          CALLS   #7,G^PAS$OPEN
      04 A8   01000000  8F   CA  03D5   763          BICL2   #FAB$M_NAM,FAB$L_FOP(R8); cancel NAM block access
                                 03DD   764 ;
                                 03DD   765 ; Test if still an included file
                                 03DD   766 ;
                   14 A6   D5  03DD   767          TSTL    FSB$L_INC(R6)
                        08   12  03E0   768          BNEQ    120$
      04 A6   80000000  8F   CA  03E2   769          BICL2   #FSB$M_INC,FSB$L_STA(R6); not included, clear flag
                                 03EA   770   120$:
                                 03EA   771 ;
                                 03EA   772 ; Restore last record read via random access by RFA
                                 03EA   773 ;
                1E A7   02   90  03EA   774          MOVB    #RAB$C_RFA,RAB$B_RAC(R7)
      10 A7     08 AA   06   28  03EE   775          MOVC3   #06,INC$T_RFA(R10),RAB$W_RFA(R7); record file address
                                 03F4   776          $GET    RAB=R7
                   12 50   E9  03FD   777          BLBC    R0,910$                 ; branch if error
                1E A7   00   90  0400   778          MOVB    #RAB$C_SEQ,RAB$B_RAC(R7); restore sequential access
                                 0404   779 ;
                                 0404   780 ; Blank out first part of record already read
                                 0404   781 ;
         50     04 AA   01   C3  0404   782          SUBL3   #1,INC$L_POS(R10),R0    ; R0 = length to blank
  08 BC  50     20 08 BC   00   2C  0409   783          MOVC5   #0,@BUF_DISP(AP),#SPACE,R0,@BUF_DISP(AP)
                                 0411   784
                             04  0411   785          RET
                                 0412   786 ;
                                 0412   787 ; Read error
                                 0412   788 ;
                                 0412   789   910$:
                        50   DD  0412   790          PUSHL   R0
                7E   34 A8   9A  0414   791          MOVZBL  FAB$B_FNS(R8),-(SP)
                     2C A8   DD  0418   792          PUSHL   FAB$L_FNA(R8)
         00000000'GF    03   FB  041B   793          CALLS   #3,G^PAS$IOERROR
                                 0422   794 ;
                                 0422   795 ;
                         00000422   796          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
                                 0422   797 ;
                                 0422   798 ;        ********************
                                 0422   799 ;        *                  *
                                 0422   800 ;        *     PAS$CONCAT   *
                                 0422   801 ;        *                  *
                                 0422   802 ;        ********************
                                 0422   803 ;
                                 0422   804 ; Called when concatenated input files existed
                                 0422   805 ;
                                 0422   806 ; Argument offsets
                                 0422   807 ;
                                 0422   808 ;        AP                              ; number of arguments (4)
```

```
                00000004  0422          FSB_DISP = 04                    ; FAB address
                00000008  0422          BUF_DISP = 08                    ; character buffer
                0000000C  0422          POS_DISP = 12                    ; last character position
                00000010  0422          LEN_DISP = 16                    ; length of name
                          0422   ;
                    00FC  0422          .ENTRY  PAS$CONCAT,^M<R2,R3,R4,R5,R6,R7>
                          0424   ;
          56   04 AC   D0 0424          MOVL    FSB_DISP(AP),R6
                          0428   ;
                          0428   ; Close current file
                          0428   ;
                     56   DD 0428        PUSHL   R6
   00000000'GF       01   FB 042A        CALLS   #1,G^PAS$CLOSEINOUT
                          0431   ;
                          0431   ; Open concatenated file
                          0431   ;
04 A6   80000000 8F   C8 0431          BISL2   #FSB$M_INC,FSB$L_STA(R6); Fake INCLUDE'd file to get
                          0439                                           ; read-only access
                     00   DD 0439        PUSHL   #PAS$C_NOCARR            ; carriage control -- not used
        FFFFFEFF 8F   DD 043B          PUSHL   #-PAS$C_DFLTRECSI
                     7E   7C 0441        CLRD    -(SP)
                 10 AC   DD 0443        PUSHL   LEN_DISP(AP)
                 0C AC   D7 0446        DECL    POS_DISP(AP)
 52   0C AC      10 AC   C3 0449        SUBL3   LEN_DISP(AP),POS_DISP(AP),R2
                     52   D7 044F        DECL    R2
             08 BC42      9F 0451        PUSHAB  @BUF_DISP(AP)[R2]
                     56   DD 0455        PUSHL   R6
   00000000'GF       07   FB 0457        CALLS   #7,G^PAS$OPEN
04 A6   80000000 8F   CA 045E          BICL2   #FSB$M_INC,FSB$L_STA(R6); Unfake INCLUDE'd file
                          0466   ;
                          0466   ; Reset file
                          0466   ;
                     56   DD 0466        PUSHL   R6
   00000000'GF       01   FB 0468        CALLS   #1,G^PAS$RESET
         04 A6       01   CA 046F        BICL2   #FSB$M_RDLN,FSB$L_STA(R6)
                          0473   ;
                     04   0473          RET
                          0474   ;
                          0474   ;
                          0474   ;
                          0474          .END
```

```
$$.TMP1                 = 00000001
$$.TMP2                 = 00000057
$420                      0000026D R     02
BUF_DISP                = 00000008
CHKCOL                    000000F2 R     02
CHKHDR                    00000115 R     02
CHKPRM                    0000014A R     02
COMPLOOP                  00000066 R     02
FAB$B_FNS               = 00000034
FAB$B_FSZ               = 0000003F
FAB$C_BLN               = 00000050
FAB$L_FNA               = 0000002C
FAB$L_FOP               = 00000004
FAB$M_NAM               = 01000000
FSB$C_BLN               = 00000018
FSB$L_INC               = 00000014
FSB$L_STA               = 00000004
FSB$M_DELZ              = 40000000
FSB$M_EOF               = 00000002
FSB$M_INC               = 80000000
FSB$M_RDLN              = 00000001
FSB$M_TXT               = 00000010
FSB$V_INC               = 0000001F
FSB_DISP                = 00000004
INC$C_BLN               = 0000002A
INC$L_NXT               = 00000000
INC$L_POS               = 00000004
INC$T_DID               = 0000000E
INC$T_FID               = 00000024
INC$T_RFA               = 00000008
INC$X_DVI               = 00000014
INCLBLN                   0000026E R     02
INCL_EOF_DISP           = 00000014
INC_DISP                = 0000001C
INP_DISP                = 00000004
LEN_DISP                = 00000010
LIB$GET_VM                ********   X   00
LNK_DISP                = 00000008
LST_DISP                = 0000000C
NAM$T_DVI               = 00000014
NAM$W_DID               = 0000002A
NAM$W_FID               = 00000024
NAM_DISP                = 00000004
OBJ_DISP                = 0000000C
OUT_DISP                = 00000008
PAS$CLOSE                 ********   X   00
PAS$CLOSEINOUT            ********   X   00
PAS$COMPINIT              00000021 RG    02
PAS$CONCAT                00000422 RG    02
PAS$C_DFLTRECSI         = 00000101
PAS$C_NOCARR            = 00000000
PAS$EXCLUDE               0000037B RG    02
PAS$INCHECK               0000034A RG    02
PAS$INCLUDE               00000272 RG    02
PAS$IOERROR               ********   X   00
PAS$OPEN                  ********   X   00
PAS$PUTMSG                000001EF RG    02
PAS$READINPUT             00000160 RG    02
PAS$RESET                 ********   X   00
PAS$TAB                   00000241 RG    02
PAS$TRANS                 000000A3 R     02
PAS$WRITELN               ********   X   00
PAS$WRITEOBJ              000001D5 RG    02
PAS$WRITEOUTPUT           000001D5 RG    02
PASOUT                    00000000 R     02
POS_DISP                = 0000000C
RAB$B_RAC               = 0000001E
RAB$C_BLN               = 00000044
RAB$C_RFA               = 00000002
RAB$C_SEQ               = 00000000
RAB$L_RBF               = 00000028
RAB$L_RHB               = 0000002C
RAB$L_UBF               = 00000024
RAB$W_RFA               = 00000010
RAB$W_RSZ               = 00000022
RET                       0000015F R     02
RMS$_EOF                = 0001827A
RST_DISP                = 00000010
SOS_DISP                = 00000010
SPACE                   = 00000020
SPEC_DISP               = 00000014
SS$_NOTRAN              = 00000629
SYS$GET                   ********  G    02
SYS$PUTMSG                ********   X   00
SYS$TRNLOG                ********  GX   00
SYSERR                    00000016 R     02
SYSOUT                    0000000A R     02
TAB_DISP                = 00000008
TRANS                     000000BE R     02
TRM_DISP                = 00000010
TRYNXT                    00000097 R     02
```

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

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (    0.)  00 (  0.)  NOPIC   USR   CON   ABS   LCL NOSHR NOEXE NORD  NOWRT NOVEC BYTE
$ABS$                           00000000  (    0.)  01 (  1.)  NOPIC   USR   CON   ABS   LCL NOSHR   EXE   RD    WRT NOVEC BYTE
_PAS$CODE                       00000474  ( 1140.)  02 (  2.)    PIC   USR   CON   REL   LCL   SHR   EXE   RD  NOWRT NOVEC BYTE

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

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     31    00:00:00.07    00:00:00.72
Command processing                106    00:00:00.55    00:00:02.26
Pass 1                            320    00:00:11.13    00:00:17.85
Symbol table sort                   0    00:00:01.49    00:00:02.49
Pass 2                            157    00:00:02.74    00:00:04.34
Symbol table output                11    00:00:00.09    00:00:00.09
Psect synopsis output               3    00:00:00.02    00:00:00.02
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              631    00:00:16.09    00:00:27.77

The working set limit was 1350 pages.
65039 bytes (128 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1155 non-local and 21 local symbols.
849 source lines were read in Pass 1, producing 37 object records in Pass 2.
17 pages of virtual memory were used to define 15 macros.

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

Macro library name                       Macros defined
------------------                       --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                   12

1231 GETS were required to define 12 macros.

There were no errors, warnings or information messages.

MACRO/DISABLE=TRACE/LIS=LIS$:PASIO4/OBJ=OBJ$:PASIO4 MSRC$:PASIO4/UPDATE=(ENH$:PASIO4)
```

PASRT1
LIS

PASMACROS
REQ

PASFCB
SDL

PASPROLOG
REQ

PASOPEDEF
REQ

PASABSL
LIS

PASCARD2
LIS

PASRTL

PASBUGCOD
REQ

PASCLOSE2
LIS

PASRT2
LIS

PASRT3
LIS

PASRTL
MAP

PASIO4
LIS

PASEXTERN
REQ

PASKDB
REQ

PASPFD
REQ

PASCONVER
LIS

PASLIB
REQ

PASPFV
REQ

PASBIN
LIS

PASCLOCK2
LIS

PASRT4
LIS