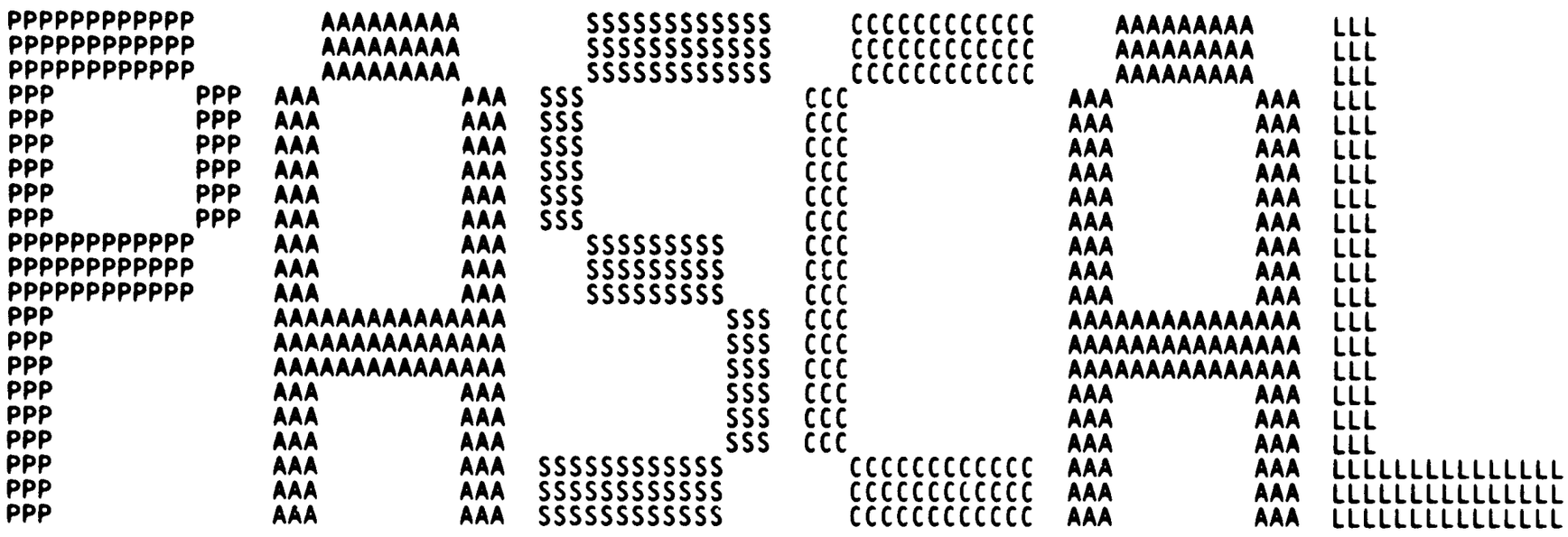
PPPPPPPPPPPP AAAAAAAAA SSSSSSSSSSS CCCCCCCCCCC AAAAAAAAA LLL
PPPPPPPPPPPP AAAAAAAAA SSSSSSSSSSS CCCCCCCCCCC AAAAAAAAA LLL
PPPPPPPPPPPP AAAAAAAAA SSSSSSSSSSS CCCCCCCCCCC AAAAAAAAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
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      000000    44    44
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      000000    44    44
PP      PP   AA      AA   SS            II      00      00  44    44
PP      PP   AA      AA   SS            II      00      00  44    44
PP      PP   AA      AA   SS            II      00      00  44    44
PP      PP   AA      AA   SS            II      00      00  44    44
PPPPPPPP     AA      AA     SSSSSS      II      00      00  4444444444
PPPPPPPP     AA      AA     SSSSSS      II      00      00  4444444444
PP           AAAAAAAAAA           SS    II      00      00        44
PP           AAAAAAAAAA           SS    II      00      00        44
PP           AA      AA           SS    II      00      00        44        ....
PP           AA      AA           SS    II      00      00        44        ....
PP           AA      AA   SSSSSSSS    IIIIII      000000          44        ....
PP           AA      AA   SSSSSSSS    IIIIII      000000          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     1 ;
0000     2 ;*****************************************************************************
0000     3 ;*                                                                           *
0000     4 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     5 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     6 ;*  ALL RIGHTS RESERVED.                                                     *
0000     7 ;*                                                                           *
0000     8 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000     9 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    10 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    11 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    12 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    13 ;*  TRANSFERRED.                                                             *
0000    14 ;*                                                                           *
0000    15 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    16 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    17 ;*  CORPORATION.                                                             *
0000    18 ;*                                                                           *
0000    19 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    20 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    21 ;*                                                                           *
0000    22 ;*                                                                           *
0000    23 ;*****************************************************************************
0000    24 ;
0000    25         .TITLE  PAS$IO_COMP             ; PASCAL RMS linkage
0000    26         .IDENT  'V04-000'
0000    27 ;
0000    28 ;*****************************************************************************
0000    29 ;*****************************************************************************
0000    30 ;**                                                                         **
0000    31 ;**             PASCAL RMS LINKAGE FOR VAX-11/780                           **
0000    32 ;**             =================================                           **
0000    33 ;**                                                                         **
0000    34 ;**                                                                         **
0000    35 ;**                VERSION V1.2 -- JANUARY 1981                             **
0000    36 ;**                                                                         **
0000    37 ;**   DEVELOPED BY:  COMPUTER SCIENCE DEPARTMENT                            **
0000    38 ;**                  UNIVERSITY OF WASHINGTON                               **
0000    39 ;**                  SEATTLE, WA 98195                                      **
0000    40 ;**                                                                         **
0000    41 ;**   AUTHORS:       MARK BAILEY, JOHN CHAN, HELLMUT GOLDE                  **
0000    42 ;**                                                                         **
0000    43 ;*****************************************************************************
0000    44 ;*****************************************************************************
0000    45 ;
0000    46 ; Modified: 16Oct80 Change the value of PRN_CRLF so that a line on the
0000    47 ;                    terminal is printed as <LF> <text> <CR> Susan Azibert
0000    48 ;
0000    49 ;           13Jan81 Change all tests of status from FMS to BLBC R0,label
0000    50 ;                   and BLBS R0,label from CMPL R0,#%RMS$_NORMAL;BNEQ label,etc.
0000    51 ;
0000    52 ;       3. Fix PAS$CONCAT to request read-only access instead of read-write access
0000    53 ;
0000    54 ;       1-005 11-Aug-81 Paul Hohensee   Changed to general addressing of external ro
0000    55 ;
0000    56 ;*****************************************************************************
0000    57 ;*****************************************************************************
```

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

```
0000     68 ;For any file variable the following storage is assumed:
0000     69 ;
0000     70 ;                    ------------------
0000     71 ;           FSB:     |     POINTER    |
0000     72 ;                    ------------------
0000     73 ;                    |   STATUS WORD  |
0000     74 ;                    ------------------
0000     75 ;                    |      LAST      |
0000     76 ;                    ------------------
0000     77 ;                    |    LINELIMIT   |
0000     78 ;                    ------------------
0000     79 ;                    |    LINECOUNT   |
0000     80 ;                    ------------------
0000     81 ;                    |  RECORD NUMBER |
0000     82 ;                    ------------------
0000     83 ;           RAB:     |                |
0000     84 ;                    |  44(HEX) BYTES |
0000     85 ;                    |       .        |
0000     86 ;                    |       .        |
0000     87 ;                    |       .        |
0000     88 ;                    ------------------
0000     89 ;           FAB:     |                |
0000     90 ;                    |  50(HEX) BYTES |
0000     91 ;                    |       .        |
0000     92 ;                    |       .        |
0000     93 ;                    |       .        |
0000     94 ;                    ------------------
0000     95 ;           NAM:     |                |    NOTE: The NAM block is allocated
0000     96 ;                    |  38(HEX) BYTES |    for the PASCAL logical files
0000     97 ;                    |       .        |    'INPUT' and 'OUTPUT' only.
0000     98 ;                    |        .       |
0000     99 ;                    |                |
0000    100 ;                    ------------------
0000    101 ;
0000    102 ; Macro options
0000    103 ;
0000    104         .DSABL  GBL                     ; no undefined references
0000    105         .ENABL  FPT                     ; rounded arithmetic
0000    106 ;
0000    107 ; External references
0000    108 ;
0000    109         .EXTRN  PAS$IOERROR
0000    110         .EXTRN  PAS$WRITELN
0000    111         .EXTRN  PAS$OPEN
0000    112         .EXTRN  PAS$CLOSE
0000    113         .EXTRN PAS$CLOSEINOUT
0000    114         .EXTRN  PAS$RESET
0000    115 ;
0000    116         .EXTRN  SYS$PUTMSG              ; messages to terminal
0000    117         .EXTRN  SYS$TRNLOG
0000    118         .EXTRN  LIB$GET_VM
0000    119 ;
0000    120 ; Provide definitions of system values
0000    121 ;
0000    122         $FABDEF
0000    123         $NAMDEF
0000    124         $RABDEF
```

```
                     0000   125          $RMSDEF                         ; for status code checking
                     0000   126          $SSDEF                          ; system definitions
                     0000   127 ;
                     0000   128 ; PASCAL compiler constants
                     0000   129 ;
                     0000   130 ; NOTE: The constants below with the names 'PAS$C_XXXXX' are
                     0000   131 ; used in the PASCAL compiler with the names 'XXXXX'. If the
                     0000   132 ; values in the compiler are altered then the values below
                     0000   133 ; must be altered accordingly.
                     0000   134 ;
00000101             0000   135          PAS$C_DFLTRECSI = 257;          ; default buffer size
                     0000   136 ;        PAS$C_NIL = 0                   ; NIL pointer
                     0000   137 ;        PAS$C_TRUE = 1                  ; TRUE
                     0000   138 ;        PAS$C_FALSE = 0                 ; FALSE
00000000             0000   139          PAS$C_NOCARR = 0                ; no carriage control
                     0000   140 ;        PAS$C_CARRIAGE = 1              ; FORTRAN carriage control
                     0000   141 ;        PAS$C_LIST = 2                  ; LIST carriage control
                     0000   142 ;        PAS$C_PRN = 3                   ; PRN carriage control
                     0000   143 ;
                     0000   144 ; PRN carriage control constants
                     0000   145 ;
                     0000   146 ;        PRN_CRLF = ^X8D01               ; PRN carriage control constant
                     0000   147                                          ; for <LF> <text> <CR>
                     0000   148 ;        PRN_NULL = ^X0000               ; PRN carriage control constant
                     0000   149                                          ; for no carriage control
                     0000   150 ;
                     0000   151 ; File status block constants
                     0000   152 ;
00000018             0000   153          FSB$C_BLN = ^X18                ; FSB block length
                     0000   154 ;        FSB$V_OPEN = 5
                     0000   155 ;        FSB$V_EOF = 1
                     0000   156 ;        FSB$V_EOLN = 2
                     0000   157 ;        FSB$V_GET = 3
                     0000   158 ;        FSB$V_TXT = 4                   ; textfile flag
                     0000   159 ;        FSB$V_RDLN = 0                  ; last access READLN
                     0000   160 ;        FSB$V_DIR = 6                   ; direct access flag
                     0000   161 ;        FSB$V_PUT = 7
                     0000   162 ;        FSB$V_INT = 8                   ; internal flag
                     0000   163 ;        FSB$V_PRMT = 9                  ; prompt flag
                     0000   164 ;        FSB$V_OUTPUT = 10               ; OUTPUT file flag
                     0000   165 ;        FSB$V_ACTIN = 11                ; actual input flag
                     0000   166 ;        FSB$V_DELZ = 30                 ; delete file if empty
0000001F             0000   167          FSB$V_INC = 31                  ; included file flag
                     0000   168 ;        FSB$B_CC = 6                    ; carriage control byte offset
                     0000   169 ;        FSB$M_OPEN = ^X0020
00000002             0000   170          FSB$M_EOF = ^X0002
                     0000   171 ;        FSB$M_EOLN = ^X0004
                     0000   172 ;        FSB$M_GET = ^X0008
                     0000   173 ;        FSB$M_PRMT = ^X0200
                     0000   174 ;        FSB$M_PUT = ^X00000080
00000010             0000   175          FSB$M_TXT = ^X0010
00000001             0000   176          FSB$M_RDLN = ^X0001
                     0000   177 ;        FSB$M_DIR = ^X00000040
                     0000   178 ;        FSB$M_INT = ^X00000100
                     0000   179 ;        FSB$M_OUTPUT = ^X0400
                     0000   180 ;        FSB$M_ACTIN = ^X0800
40000000             0000   181          FSB$M_DELZ = ^X40000000
```

```
80000000  0000   182          FSB$M_INC = ^X80000000
          0000   183 ;         FSB$L_CNT = 16                  ; line count (textfiles)
00000014  0000   184          FSB$L_INC = 20                  ; %INCLUDE block address
          0000   185 ;         FSB$L_LIM = 12                  ; LINELIMIT
          0000   186 ;         FSB$L_LST = 8                   ; LAST WORD OFFSET
          0000   187 ;         FSP$L_PFSB = 20                 ; Related file FSB for prompting
          0000   188                                           ; for INPUT, has address of OUTPUT FSB
          0000   189                                           ; for OUTPUT, has address of INPUT FSB
          0000   190                                           ; (shares storage with include address
          0000   191                                           ; and direct access record
          0000   192                                           ; buffer address)
          0000   193 ;         FSB$L_REC = 20                  ; record buffer address for
          0000   194                                           ; direct access (shares storage
          0000   195                                           ; with include address and related
          0000   196                                           ; file FSB)
00000004  0000   197          FSB$L_STA = 4                   ; status word offset
          0000   198 ;
          0000   199 ; Character constants
          0000   200 ;
          0000   201 ;         TAB = ^X09
00000020  0000   202          SPACE = ^X20
          0000   203 ;         DOLLAR = ^X24
          0000   204 ;         FORMFEED = ^XC
          0000   205 ;         STAR = ^X2A
          0000   206 ;         PLUS = ^X2B
          0000   207 ;         MINUS = ^X2D
          0000   208 ;         POINT = ^X2E
          0000   209 ;         ZERO = ^X30
          0000   210 ;         ONE = ^X31
          0000   211 ;         NINE = ^X39
          0000   212 ;         AA = ^X41
          0000   213 ;         DD = ^X44
          0000   214 ;         EE = ^X45
          0000   215 ;         ZZ = ^X5A
          0000   216 ;         UNDERSCORE = ^X5F
          0000   217 ;         AA_SMALL = ^X61
          0000   218 ;         ZZ_SMALL = ^X7A
          0000   219 ;
          0000   220 ;
00000000  0000   221          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
          0000   222 ;
          0000   223 ;         ********************
          0000   224 ;         *                  *
          0000   225 ;         *   PAS$COMPINIT   *
          0000   226 ;         *                  *
          0000   227 ;         ********************
          0000   228 ;
          0000   229 ; Initializes the file control blocks for certain files used by the
          0000   230 ; compiler.  Specifically, it
          0000   231 ;
          0000   232 ;         (1) clears the RDLN flag for the input file so that PAS$EOF and
          0000   233 ;             PAS$EOLN will never retrieve the next record
          0000   234 ;
          0000   235 ;         (2) sets a flag to delete the output file if it is empty
          0000   236 ;
          0000   237 ;         (3) clears the textfile flag for the object file
          0000   238 ;
```

```
                                0000   239 ; In addition, determines if the listing is going to the terminal.  If
                                0000   240 ; the listing is going to the terminal, the fourth (return) argument is
                                0000   241 ; set to 1, otherwise it is set to 0.
                                0000   242 ;
                                0000   243 ; Argument offsets
                                0000   244 ;
                                0000   245 ;       AP                              ; number of arguments (4)
                   00000004     0000   246         INP_DISP = 04                   ; input file FSB
                   00000008     0000   247         OUT_DISP = 08                   ; output file FSB
                   0000000C     0000   248         OBJ_DISP = 12                   ; object file FSB
                   00000010     0000   249         TRM_DISP = 16                   ; list_to_terminal flag (by reference)
                                0000   250 ;
                                0000   251 ; Other constants
                                0000   252 ;
                                0000   253 PASOUT:
54 55 50 54 55 4F 24 53 41 50   0000   254         .ASCII  /PAS$OUTPUT/            ; listing file logical name
                                000A   255 SYSOUT:
                           000A'000A   256         .WORD   20$-10$                 ; length of string
54 55 50 54 55 4F 24 53 59 53   000C   257 10$:    .ASCII  /SYS$OUTPUT/            ; standard output logical name
                                0016   258 20$:
                                0016   259 SYSERR:
                           0009'0016   260         .WORD   40$-30$                 ; length of string
   52 4F 52 52 45 24 53 59 53   0018   261 30$:    .ASCII  /SYS$ERROR/             ; error output logical name
                                0021   262 40$:
                           00FC 0021   263         .ENTRY  PAS$COMPINIT,^M<R2,R3,R4,R5,R6,R7>
        5E   00000088 8F   C2 0023   264         SUBL2   #136,SP                 ; make room for translated names
             56   04 AC   D0 002A   265         MOVL    INP_DISP(AP),R6
             04 A6   01   CA 002E   266         BICL2   #FSB$M_RDLN,FSB$L_STA(R6); clear input RDLN flag
             56   0C AC   D0 0032   267         MOVL    OBJ_DISP(AP),R6
             04 A6   10   CA 0036   268         BICL2   #FSB$M_TXT,FSB$L_STA(R6) ; clear textfile flag
             56   08 AC   D0 003A   269         MOVL    OUT_DISP(AP),R6
     04 A6   40000000 8F   C8 003E   270         BISL2   #FSB$M_DELZ,FSB$L_STA(R6); set to delete if empty
                                0046   271 ;
                                0046   272 ; Now determine if listing is going to terminal.
                                0046   273 ;
  BC AD   B6 AF   0A   28       0046   274         MOVC3   #10,PASOUT,-68(FP)      ; move listing logical name
          FC AD   0A   D0       004C   275         MOVL    #10,-4(FP)              ; set length of logical name
                  00   DD       0050   276         PUSHL   #0                      ; arg3 (process-permanent flag)
                                0052   277                                         ; is set to false
              FC AD   DF       0052   278         PUSHAL  -4(FP)                  ; arg2 (address of length)
              BC AD   DF       0055   279         PUSHAL  -68(FP)                 ; arg1 (address of string)
 000000A3'EF   03   FB         0058   280         CALLS   #3,PAS$TRANS            ; translate logical name
                                005F   281 ;
                                005F   282 ;  Loop through names SYS$OUTPUT and SYS$ERROR, translating them and
                                005F   283 ;  comparing the result to the translation of PAS$OUTPUT.  If either
                                005F   284 ;  is equal, return TRUE; otherwise return FALSE.
                                005F   285 ;
          56   02   D0         005F   286         MOVL    #2,R6                   ; loop index
      57   A5   AF   DE         0062   287         MOVAL   SYSOUT,R7               ; address of logical name string
                                0066   288 COMPLOOP:                             ; compare translated names
FF78 CD   02 A7   67   28       0066   289         MOVC3   (R7),2(R7),-136(FP)     ; move logical name
          B8 AD   67   3C       006D   290         MOVZWL  (R7),-72(FP)            ; set length of logical name
                  01   DD       0071   291         PUSHL   #1                      ; arg3 (process-permanent flag)
                                0073   292                                         ; is set to TRUE
              B8 AD   DF       0073   293         PUSHAL  -72(FP)                 ; arg2 (address of length)
          FF78 CD   DF         0076   294         PUSHAL  -136(FP)                ; arg1 (address of string)
 000000A3'GF   03   FB         007A   295         CALLS   #3,G^PAS$TPANS          ; translate logical name
```

```
                B8 AD    FC AD    D1  0081   296          CMPL    -4(FP),-72(FP)           ; are lengths equal?
                                   0F  12 0086   297          BNEQ    TRYNXT                   ; no--try next logical name
          FF78 CD    BC AD    FC AD    29 0088   298          CMPC3   -4(FP),-68(FP),-136(FP) ; else, are strings equal?
                                   05  12 0090   299          BNEQ    TRYNXT                   ; no--try next logical name
                      10 BC    01       D0 0092   300          MOVL    #1,@TRM_DISP(AP)         ; yes--set return value TRUE
                                        04 0096   301          RET                              ; return TRUE to caller
                                           0097   302 TRYNXT:                                 ; try next logical name
                    57    FF7B CF       DE 0097   303          MOVAL   SYSERR,R7                ; address of next logical name
                             C7 56       F5 009C   304          SOBGTR  R6,COMPLOOP              ; loop through all logical names
                             10 BC       D4 009F   305          CLRL    @TRM_DISP(AP)            ; set return value FALSE
                                         04 00A2   306          RET                              ; return FALSE to caller
                                            00A3   307 ;
                                            00A3   308 ;  Routine to translate logical names, used by PAS$COMPINIT above.
                                            00A3   309 ;
                                            00A3   310 PAS$TRANS:                               ; routine to translate names
                                    0700    00A3   311          .WORD   ^M<R8,R9,R10>
                          5E    10  C2    00A5   312          SUBL2   #16,SP                   ; make room for $TRNLOG args
                FC AD     04 AC  D0    00A8   313          MOVL    4(AP),-4(FP)             ; address of string
                F8 AD     08 BC  D0    00AD   314          MOVL    @8(AP),-8(FP)            ; length of logical name
                F4 AD     04 AC  D0    00B2   315          MOVL    4(AP),-12(FP)            ; address of result
                F0 AD     50 8F  9A    00B7   316          MOVZBL  #^0,-16(FP)              ; length of result
                             58  D4    00BC   317          CLRL    R8                       ; clear colon flag
                                         00BE   318 TRANS:   $TRNLOG_S LOGNAM=-8(FP),-       ; translate the name
                                         00BE   319                    RSLLEN=@8(AP),-
                                         00BE   320                    RSLBUF=-16(FP)
                          1B  58  E9     00D4   321          BLBC    R8,CHKCOL                ; branch if not trying without colon
                00000629 8F     50  D1   00D7   322          CMPL    R0,#SS$_NOTRAN           ; was translation successful:
                                0E  12   00DE   323          BNEQ    10$                      ; yes--continue
                             59    08 BC  D0    00E0   324          MOVL    @8(AP),R9                ; else put colon back and quit
                          04 BC49     3A  90    00E4   325          MOVB    #^A/:/,@4(AP)[R9]        ; put colon back
                       08 BC    59     01  C1    00E9   326          ADDL3   #1,R9,@8(AP)             ; set return length
                                58     D4  00EE   327 10$:     CLRL    R8                       ; not trying without colon any more
                                23     11  00F0   328          BRB     CHKHDR                   ; go check for 4-byte header
                                           00F2   329 CHKCOL:                                  ; check if translate failed due to colon
                             1F 0C AC     E8  00F2   330          BLBS    12(AP),CHKHDR            ; if process-perm name, continue
                00000629 8F     50  D1   00F6   331          CMPL    R0,#SS$_NOTRAN           ; translation successful?
                                16  12   00FD   332          BNEQ    CHKHDR                   ; yes--continue
                             59    08 BC  D0    00FF   333          MOVL    @8(AP),R9                ; translated length
                                59     D7  0103   334          DECL    R9                       ; index of last character
                          3A    04 BC49     91  0105   335          CMPB    @4(AP)[R9],#^A/:/        ; last character a colon?
                                09     12  010A   336          BNEQ    CHKHDR                   ; no--continue
                                   08 BC  D7    010C   337          DECL    @8(AP)                   ; else remove colon and try again
                                58     01  D0  010F   338          MOVL    #1,R8                    ; set flag to show trying again
                                50     01  D0  0112   339          MOVL    #1,R0                    ; set status to continue
                                           0115   340 CHKHDR:                                  ; check for 4-byte header
                             1B    04 BC     91  0115   341          CMPB    @4(AP),#^X1B             ; translated name begins with esc?
                                   2F     12  0119   342          BNEQ    CHKPRM                   ; no--make sure not process-perm
                                59     04  D0  011B   343          MOVL    #4,R9                    ; initialize number of bytes to remove
                5F 8F     04 BC49     91  011E   344          CMPB    @4(AP)[R9],#^A/_/        ; translated name begins with '_'?
                                02     12  0124   345          BNEQ    10$                      ; no--remove just first 4 bytes
                                59     D6  0126   346          INCL    R9                       ; else remove 5 bytes
                5A    08 BC     59     C3  0128   347 10$:     SUBL3   R9,@8(AP),R10            ; r10 <- new string length
          04 BC    04 BC49     5A     28  012D   348          MOVC3   R10,@4(AP)[R9],@4(AP)    ; move remaining chars to front
                                5A     D7  0134   349          DECL    R10                      ; r10 <- index of last character
                          3A    04 BC4A     91  0136   350          CMPB    @4(AP)[R10],#^A/:/       ; name ends with colon?
                                07     13  013B   351          BEQL    20$                      ; yes--continue
                                5A     D6  013D   352          INCL    R10                      ; else add colon at end
```

```
                04 BC4A    3A   90  013F    353          MOVB    #^A/:/,@4(AP)[R10]
             08 BC   5A    01   C1  0144    354 20$:     ADDL3   #1,R10,@8(AP)           ; update length
                           04   0149    355          RET                             ; return
                                014A    356 CHKPRM:                                  ; check if process permanent name
                     11 0C AC   E8  014A    357          BLBS    12(AP),RET              ; return if process permanent name
           00000629 8F    50   D1  014E    358          CMPL    R0,#SS$_NOTRAN          ; translation was successful?
                          08   13  0155    359          BEQL    RET                     ; no--return
              F8 AD   08 BC   D0  0157    360          MOVL    @8(AP),-8(FP)           ; update length of $TRNLOG arg
                        FF5F   31  015C    361          BRW     TRANS                   ; translate recursively
                           04   C15F    362 RET:     RET                             ; return to caller
                                0160    363 ;
                                0160    364 ;
                           00000160    365          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
                                0160    366 ;
                                0160    367 ;       ********************
                                0160    368 ;       *                  *
                                0160    369 ;       *  PAS$READINPUT   *
                                0160    370 ;       *                  *
                                0160    371 ;       ********************
                                0160    372 ;
                                0160    373 ; Reads the next record into the user supplieed buffer returning the
                                0160    374 ; length read. The end-of-file flag is altered as need be.
                                0160    375 ;
                                0160    376 ; Argument offsets
                                0160    377 ;
                                0160    378 ;       AP                              ; number of arguments (4)
                     00000004   0160    379          FSB_DISP = 04                   ; FSB address
                     00000008   0160    380          BUF_DISP = 08                   ; buffer address
                     0000000C   0160    381          LEN_DISP = 12                   ; length of buffer (by reference)
                     00000010   0160    382          SOS_DISP = 16                   ; SOS line buffer (by reference)
                     00000014   0160    383          incl_eof_disp = 20              ; eof on INCLUDE'd file flag
                                0160    384                                          ; (byte by ref)
                                0160    385                                          ;    0 = no eof
                                0160    386                                          ;    1 = eof hit
                                0160    387 ;
                                003C    0160    388          .ENTRY  PAS$READINPUT,^M<R2,R3,R4,R5>
                          14 BC   94    0162    389          clrb    @incl_eof_disp(ap)      ; assume no eof
                     54   04 AC   D0    0165    390          MOVL    FSB_DISP(AP),R4         ; R4 = address of FSB
                52   18   54   C1    0169    391          ADDL3   R4,#FSB$C_BLN,R2        ; R2 = address of RAB
55         00000044 8F   52   C1    016D    392          ADDL3   R2,#RAB$C_BLN,R5        ; R5 = address of FAB
                     53   24 A2   D0    0175    393          MOVL    RAB$L_UBF(R2),R3        ; save buffer address
                24 A2   08 AC   D0    0179    394          MOVL    BUF_DISP(AP),RAB$L_UBF(R2); set new buffer address
                     02   3F A5   91    017E    395          CMPB    FAB$B_FSZ(R5),#2        ; check for SOS line numbers
                                07   12  0182    396          BNEQ    120$
                2C A2   10 AC   D0    0184    397          MOVL    SOS_DISP(AP),RAB$L_RHB(R2)
                                03   11  0189    398          BRB     130$
                                018B    399 120$:
                          2C A2   D4    018B    400          CLRL    RAB$L_RHB(R2)
                                018E    401 130$:
                                018E    402          $GET    RAB=R2
                00018278 8F   50   D1    0197    403          CMPL    R0,#RMS$_EOF
                                18   12  019E    404          BNEQ    110$
                   0D 04 A4   1F   E1    01A0    405          BBC     #FSB$V_INC_FSB$L_STA(R4),100$
                00000379'EF   6C   FA    01A5    406          CALLG   (AP),PAS$EXCLUDE
                     14 BC   01   90    01AC    407          movb    #1,@incl_eof_disp(ap)   ; set eof flag
                                09   11  01B0    408          BRB     111$
                                01B2    409 100$:
```

```
                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$L_UBF(R2)        ; restore buffer address
                                  01C4   417          RET
                                  01C5   418 ;
                                  01C5   419 ; Input error
                                  01C5   420 ;
                                  01C5   421  910$:
                         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$IOERROR
                                  01D5   426 ;
                                  01D5   427 ;
                                  01D5   428          .PSECT  _PAS$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$WRITELN
                                  04  01EE   453          RET
                                  01EF   454 ;
                                  01EF   455 ;
                                  01EF   456          .PSECT  _PAS$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$WRITEOUTPUT, buffer overflow
                                01EF   468 ; is NOT checked.
                                01EF   469 ;
                                01EF   470 ; Argument list
                                01EF   471 ;
                                01EF   472 ;       AP                                  ; number of arguments (4)
                  00000004      01EF   473         FSB_DISP = 04                       ; FSB address
                  00000008      01EF   474         BUF_DISP = 08                       ; buffer address
                  0000000C      01EF   475         LEN_DISP = 12                       ; buffer length (by reference)
                  00000010      01EF   476         RST_DISP = 16                       ; reset flag (by value)
                                01EF   477                                             ; 0: do not reset line to be empty
                                01EF   478                                             ; *. reset line after sending to terminal
                                01EF   479 ;
                        007C    01EF   480         .ENTRY  PAS$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 A7   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$:     RET                                 ; all done
                                0241   503 ;
                                0241   504 ;
                           00000241    505         .PSECT  _PAS$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 ;         33222222222211111111111
                                        026E   551 ;         10987654321098765432109876543210
                                        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    581          INC$X_DVI = 20
                     00000024  026E    582          INC$T_FID = 36
                               026E    583 ;
                               026E    584 ; Argument offsets
                               026E    585 ;
                               026E    586 ;        AP                              ; number of arguments (7)
                     00000004  026E    587          FSB_DISP = 04                   ; FSB address
                     00000008  026E    588          BUF_DISP = 08                   ; character buffer address
                     0000000C  026E    589          POS_DISP = 12                   ; current buffer position (by value)
                     00000010  026E    590          LST_DISP = 16                   ; last buffer position (by reference)
                     00000014  026E    591          spec_disp = 20                  ; include specification string length
                               026E    592                                          ;       (by value)
                     00000018  026E    593          LEN_DISP = 24                   ; file name string length (by value)
                     0000001C  026E    594          INC_DISP = 28                   ; recursive include flag
                               026E    595                                          ;    0 => ok
                               026E    596                                          ;    1 => recursive
                               026E    597 ;
                               026E    598 ; Constants
                               026E    599 ;
                     0000002A  026E    600 INCLBLN: .LONG  INC$C_BLN
                               0272    601 ;
                         07FC  0272    602          .ENTRY  PAS$INCLUDE,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
                               0274    603
              56   04 AC   D0  0274    604          MOVL    FSB_DISP(AP),R6         ; R6 = address of FSB
           57   18   56   C1  0278    605          ADDL3   R6,#FSB$C_BLN,R7        ; R7 = address of RAB
     58 00000044 8F   57   C1  027C    606          ADDL3   R7,#RAB$C_BLN,R8        ; R8 = address of FAB
     59 00000050 8F   58   C1  0284    607          ADDL3   R8,#FAB$C_BLN,R9        ; R9 = address of NAM
                               028C    608 ;
                               028C    609 ; Allocate memory for include block
                               028C    610 ;
                  5E   04   C2  028C    611          SUBL2   #4,SP
                  5A   5E   D0  028F    612          MOVL    SP,R10                  ; R10 = address of memory address
                       5A   DD  0292    613          PUSHL   R10
                     D7 AF   DF  0294    614          PUSHAL  INCLBLN
           00000000'GF   02   FB  0297    615          CALLS   #2,G^LIB$GET_VM
                    03 50   E8  029E    616          BLBS    R0,101$
                     0091   31  02A1    617          BRW     910$
                               02A4    618 ;
                               02A4    619 ; Save include information
                               02A4    620 ;
                               02A4    621   101$:
                  5A   6A   D0  02A4    622          MOVL    (R10),R10               ; R10 = address of include block
              6A   14 A6   D0  02A7    623          MOVL    FSB$L_INC(R6),INC$L_NXT(R10)
           04 AA   0C AC   D0  02AB    624          MOVL    POS_DISP(AP),INC$L_POS(R10)
     08 AA   10 A7   06  28  02B0    625          MOVC3   #06,RAB$W_RFA(R7),INC$T_RFA(R10)
     0E AA   2A A9   06  28  02B6    626          MOVC3   #06,NAM$W_DID(R9),INC$T_DID(R10)
     14 AA   14 A9   10  28  02BC    627          MOVC3   #16,NAM$T_DVI(R9),INC$X_DVI(R10)
     24 AA   24 A9   06  28  02C2    628          MOVC3   #06,NAM$W_FID(R9),INC$T_FID(R10)
                               02C8    629 ;
                               02C8    630 ; Blank out remaining part of line
                               02C8    631 ;
        51   0C AC   01   C3  02C8    632          SUBL3   #1,POS_DISP(AP),R1      ; R1 = substring offset
     50   10 BC   0C AC   C3  02CD    633          SUBL3   POS_DISP(AP),@LST_DISP(AP),R0
                          50   D6  02D3    634          INCL    R0                      ; R0 = substring length
08 BC41   50   20   66   00   2C  02D5    635          MOVC5   #0,(R6),#SPACE,R0,@BUF_DISP(AP)[R1]; blank out substring
                               02DD    636 ;
                               02DD    637 ; 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   695          NAM_DISP = 04                   ; current input file name block
                     00000008  034A   696          LNK_DISP = 08                   ; head of included file list
                               034A   697 ;
                         00C0  034A   698          .ENTRY  PAS$INCHECK,^M<R6,R7>
                 56  04 AC    D0  034C   699          MOVL    NAM_DISP(AP),R6         ; R6 = address of NAM block
                 57  08 AC    D0  0350   700          MOVL    LNK_DISP(AP),R7         ; R7 = address of first include block
                       03    11  0354   701          BRB     130$
                               0356   702  120$:
                     57  67    D0  0356   703          MOVL    INC$L_NXT(R7),R7        ; get next include block
                               0359   704  130$:
                         50    D4  0359   705          CLRL    R0                      ; set R0 to FALSE
                         57    D5  035B   706          TSTL    R7                      ; any more?
                         1B    13  035D   707          BEQL    899$
          14 A7      14 A6    10  029  035F   708          CMPC    #16,NAM$T_DVI(R6),INC$X_DVI(R7); check device
                         EF    12  0365   709          BNEQ    120$
          0E A7      2A A6    06  029  0367   710          CMPC    #06,NAM$W_DID(R6),INC$T_DID(R7); check directory
                         E7    12  036D   711          BNEQ    120$
          24 A7      24 A6    06  029  036F   712          CMPC    #06,NAM$W_FID(R6),INC$T_FID(R7); check file id
                         DF    12  0375   713          BNEQ    120$
                     50  01    D0  0377   714          MOVL    #1,R0                   ; set R0 to TRUE
                               037A   715  899$:
                         04    037A   716          RET
                               037B   717 ;
                               037B   718 ;
                     0000037B   719          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
                               037B   720 ;
                               037B   721 ;        ********************
                               037B   722 ;        *                  *
                               037B   723 ;        *   PAS$EXCLUDE    *
                               037B   724 ;        *                  *
                               037B   725 ;        ********************
                               037B   726 ;
                               037B   727 ; Called when an end-of-file is encountered on an included file. The
                               037B   728 ; current file is closed and the previous file is opened and set to
                               037B   729 ; the last record access on the previous file.
                               037B   730 ;
                               037B   731 ; Argument offsets
                               037B   732 ;
                               037B   733 ;         AP                                 ; number of arguments (3)
                     00000004  037B   734          FSB_DISP = 04                   ; FSB address
                     00000008  037B   735          BUF_DISP = 08                   ; character buffer
                     0000000C  037B   736          LST_DISP = 12                   ; last buffer character
                               037B   737 ;
                         07FC  037B   738          .ENTRY  PAS$EXCLUDE,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10>
                 56  04 AC    D0  037D   739          MOVL    FSB_DISP(AP),R6
             57  18  56    C1  0381   740          ADDL3   R6,#FSB$C_BLN,R7        ; R7 = address of RAB
58  00000044 8F  57    C1  0385   741          ADDL3   R7,#RAB$C_BLN,R8        ; R8 = address of FAB
59  00000050 8F  58    C1  038D   742          ADDL3   R8,#FAB$C_BLN,R9        ; R9 = address of NAM
                               0395   743 ;
                               0395   744 ; Close current file
                               0395   745 ;
                         56    DD  0395   746          PUSHL   R6
          00000000'GF    01    FB  0397   747          CALLS   #1,G^PAS$CLOSEINOUT
                               039E   748 ;
                               039E   749 ; Restore NAM block and open previous file by NAM block
                               039E   750 ;
                 5A  14 A6    D0  039F   751          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   809          FSB_DISP = 04                   ; FAB address
                     00000008  0422   810          BUF_DISP = 08                   ; character buffer
                     0000000C  0422   811          POS_DISP = 12                   ; last character position
                     00000010  0422   812          LEN_DISP = 16                   ; length of name
                               0422   813 ;
                         00FC  0422   814          .ENTRY  PAS$CONCAT,^M<R2,R3,R4,R5,R6,R7>
                               0424   815 ;
               56   04 AC   D0  0424   816          MOVL    FSB_DISP(AP),R6
                               0428   817 ;
                               0428   818 ; Close current file
                               0428   819 ;
                        56     DD  0428   820          PUSHL   R6
          00000000'GF   01     FB  042A   821          CALLS   #1,G^PAS$CLOSEINOUT
                               0431   822 ;
                               0431   823 ; Open concatenated file
                               0431   824 ;
04 A6      80000000 8F   C8  0431   825          BISL2   #FSB$M_INC,FSB$L_STA(R6); Fake INCLUDE'd file to get
                               0439   826                                          ; read-only access
                        00     DD  0439   827          PUSHL   #PAS$C_NOCARR           ; carriage control -- not used
                FFFFFEFF 8F    DD  043B   828          PUSHL   #-PAS$C_DFLTRECSI
                        7E     7C  0441   829          CLRD    -(SP)
                     10 AC     DD  0443   830          PUSHL   LEN_DISP(AP)
                     0C AC     D7  0446   831          DECL    POS_DISP(AP)
 52    0C AC        10 AC     C3  0449   832          SUBL3   LEN_DISP(AP),POS_DISP(AP),R2
                        52     D7  044F   833          DECL    R2
                  08 BC42      9F  0451   834          PUSHAB  @BUF_DISP(AP)[R2]
                        56     DD  0455   835          PUSHL   R6
          00000000'GF   07     FB  0457   836          CALLS   #7,G^PAS$OPEN
04 A6      80000000 8F   CA  045E   837          BICL2   #FSB$M_INC,FSB$L_STA(R6); Unfake INCLUDE'd file
                               0466   838 ;
                               0466   839 ; Reset file
                               0466   840 ;
                        56     DD  0466   841          PUSHL   R6
          00000000'GF   01     FB  0468   842          CALLS   #1,G^PAS$RESET
                04 A6   01     CA  046F   843          BICL2   #FSB$M_RDLN,FSB$L_STA(R6)
                               0473   844 ;
                        04     0473   845          RET
                               0474   846 ;
                               0474   847 ;
                               0474   848 ;
                               0474   849          .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               = 0000000C
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
_255DUA28:[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