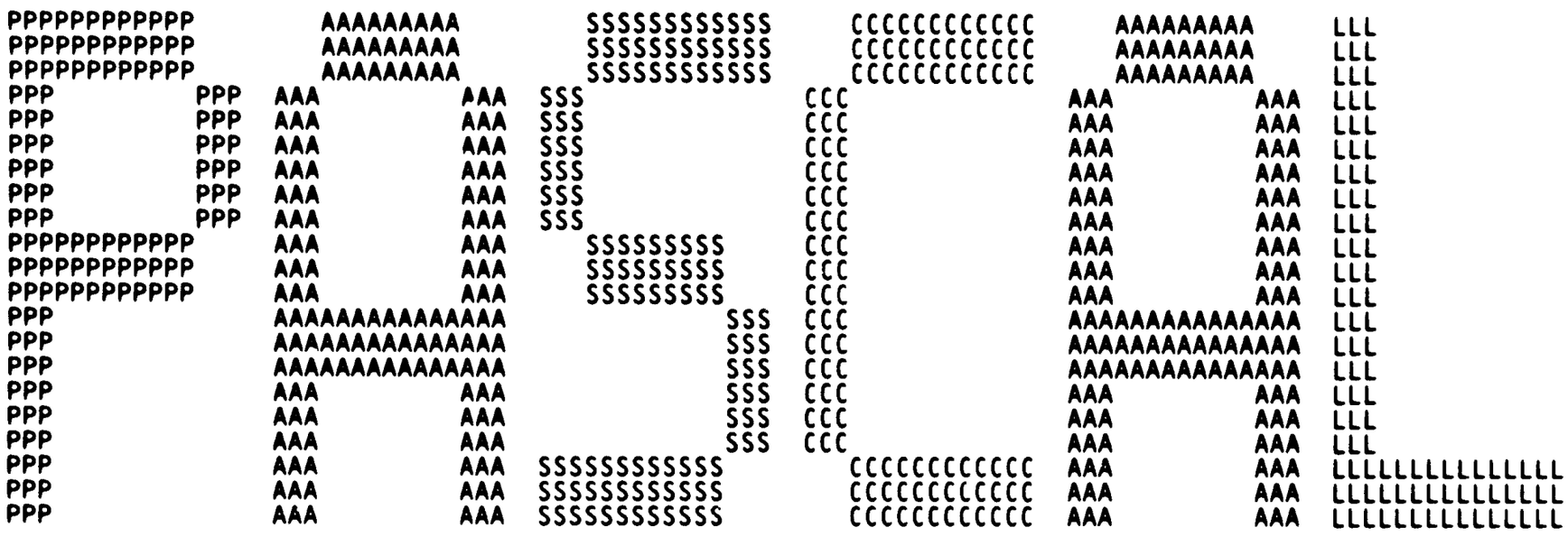
PPPPPPPPPPPP AAAAAAAAA SSSSSSSSSSS CCCCCCCCCCC AAAAAAAAA LLL
PPPPPPPPPPPP AAAAAAAAA SSSSSSSSSSS CCCCCCCCCCC AAAAAAAAA LLL
PPPPPPPPPPPP AAAAAAAAA SSSSSSSSSSS CCCCCCCCCCC AAAAAAAAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA AAA SSS CCC AAA AAA LLL
PPP PPP AAA 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

••FILE••ID••PAS101

```
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      000000       11
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      000000       11
PP      PP   AA      AA  SS              II      00      00   1111
PP      PP   AA      AA  SS              II      00      00   1111
PP      PP   AA      AA  SS              II      00      00     11
PP      PP   AA      AA  SS              II      00      00     11
PPPPPPPP     AA      AA    SSSSSS        II      00      00     11
PPPPPPPP     AA      AA    SSSSSS        II      00      00     11
PP           AAAAAAAAAA          SS      II      00      00     11
PP           AAAAAAAAAA          SS      II      00      00     11
PP           AA      AA          SS      II      00      00     11          ....
PP           AA      AA          SS      II      00      00     11          ....
PP           AA      AA  SSSSSSSS      IIIIII      000000     111111        ....
PP           AA      AA  SSSSSSSS      IIIIII      000000     111111        ....

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$IO_BASIC            ; PASCAL RMS Linkage
0000    26         .ident 'V04-000'
0000    27 ;
0000    28 ;*****************************************************************************
0000    29 ;*****************************************************************************
0000    30 ;**                                                                         **
0000    31 ;**             PASCAL RMS LINKAGE FOR VAX-11/780                           **
0000    32 ;**             =================================                           **
0000    33 ;**                                                                         **
0000    34 ;**                                                                         **
0000    35 ;**               VERSION V1.2 -- JANUARY 1981                              **
0000    36 ;**                                                                         **
0000    37 ;**    DEVELOPED BY:  COMPUTER SCIENCE DEPARTMENT                           **
0000    38 ;**                   UNIVERSITY OF WASHINGTON                              **
0000    39 ;**                   SEATTLE, WA 98195                                     **
0000    40 ;**                                                                         **
0000    41 ;**    AUTHORS:       MARK BAILEY, JOHN CHAN, HELLMUT GOLDE                 **
0000    42 ;**                                                                         **
0000    43 ;*****************************************************************************
0000    44 ;*****************************************************************************
0000    45 ;
0000    46 ;
0000    47 ; History :
0000    48 ;
0000    49 ;       1. Change PAS$IOERROR to signal all errors with LIB$STOP
0000    50 ;          instead of putting out messages directly
0000    51 ;          10-JUN-80 S. Azibert
0000    52 ;
0000    53 ;       2. Change routines PAS$INPUT and PAS$OUTPUT to use the filenames
0000    54 ;          SYS$INPUT and SYS$OUTPUT, if PAS$INPUT and PAS$OUTPUT are not defined.
0000    55 ;
0000    56 ;       3. Change the definition of PRN_CRLF so that on a terminal, a line
0000    57 ;          of output looks like:
```

```
0000     58 ;                <LF> <text> <CR>
0000     59 ;
0000     60 ;        4.  Fix a bug introduced into PAS$INPUT and PAS$OUTPUT.
0000     61 ;            $TRNLOG_S returns one of two successful values,
0000     62 ;            whereas the code was checking for an error return.
0000     63 ;
0000     64 ;        5.  Paul Hohensee       13-Jan-81
0000     65 ;            Change all tests of status returns from RMS to BLBC R0,label
0000     66 ;            or BLBS R0,label instead of CMPL R0,#RMS$NORMAL;BNEQ label, etc.
0000     67 ;
0000     68 ;        6.  Add a flag (PROMPT_FLAG) so that carriage control on
0000     69 ;            prompting can be done correctly.  S. Azibert  15-Jan-81
0000     70 ;
0000     71 ;        7.  Deallocate record buffer after a file is closed (PAS$CLOSE)
0000     72 ;            Record buffer is initially allocated by LIB$GET_VM, but the
0000     73 ;            space is never released. Ditto file name string (assuming
0000     74 ;            it was allocated by LIB$GET_VM (not in static storage).
0000     75 ;                                                    Paul Hohensee
0000     76 ;
0000     77 ;        8.  Eliminate call to PAS$FILENAME for PAS$INITFILES. It does not
0000     78 ;            seem to be necessary, since all file names passed to
0000     79 ;            PAS$INITFILES are allocated in static, read/only storage,
0000     80 ;            and therefore do not need PAS$FILENAME's services.
0000     81 ;            Also, since bugfix number 7 above deallocates the file name
0000     82 ;            string as well as the record buffer, multiple opens on the
0000     83 ;            same file in the same block would not work (they would
0000     84 ;            fail in RMS due to a bad file name) if space for the
0000     85 ;            file name string were allocated by LIB$GET_VM.
0000     86 ;                                                    Paul Hohensee 4/6/81
0000     87 ;
0000     88 ;        9.  Fix PAS$REWRITE to do a rewind on an empty file, rather than a truncate.
0000     89 ;                                                    Paul Hohensee 19-Jul-81
0000     90 ;
0000     91 ;        10. Fix PAS$OPEN to request read-only access to INCLUDE'd files.
0000     92 ;            Fix PAS$INPUT to request read-only access to INPUT.
0000     93 ;
0000     94 ;        11. Change references to external routines to general addressing.
0000     95 ;
0000     96 ;        12. Use NAM$C_BLN_V2 since compiler was built with VMS V2.
0000     97 ;            Steven Lionel 23-Oct-1981
0000     98 ;
0000     99 ;        13. Change PAS$OPEN so that it no longer scans leading and trailing blanks
0000    100 ;            from the filename.  V2 VMS does this for us.  We had been deallocating
0000    101 ;            less space than was originally allocated because of the blanks.
0000    102 ;                                                    Joyce Spencer 10-Oct-1981
0000    103 ;
0000    104 ;*****************************************************************
0000    105 ;*****************************************************************
0000    106 ;**                                                             **
0000    107 ;**                                                             **
0000    108 ;**                     SECTION 1                               **
0000    109 ;**                                                             **
0000    110 ;**                   BASIC PROCEDURES                          **
0000    111 ;**                                                             **
0000    112 ;**                                                             **
0000    113 ;*****************************************************************
0000    114 ;*****************************************************************
```

```
0000    116 ; For any file variable the following storage is assumed:
0000    117 ;
0000    118 ;                    ------------------
0000    119 ;           FSB:    |      POINTER     |
0000    120 ;                    ------------------
0000    121 ;                   |   STATUS WORD    |
0000    122 ;                    ------------------
0000    123 ;                   |       LAST       |
0000    124 ;                    ------------------
0000    125 ;                   |     LINELIMIT    |
0000    126 ;                    ------------------
0000    127 ;                   |     LINECOUNT    |
0000    128 ;                    ------------------
0000    129 ;                   |  RECORD NUMBER   |
0000    130 ;                    ------------------
0000    131 ;           RAB:    |                  |
0000    132 ;                     44(HEX) BYTES
0000    133 ;                   |        .         |
0000    134 ;                   |        .         |
0000    135 ;                   |        .         |
0000    136 ;                    ------------------
0000    137 ;           FAB:    |                  |
0000    138 ;                     50(HEX) BYTES
0000    139 ;                   |        .         |
0000    140 ;                   |        .         |
0000    141 ;                   |        .         |
0000    142 ;                    ------------------
0000    143 ;           NAM:    |                  |     NOTE: The NAM block is allocated
0000    144 ;                     38(HEX) BYTES            for the PASCAL logical files
0000    145 ;                   |        .         |     'INPUT' and 'OUTPUT' only.
0000    146 ;                   |        .         |
0000    147 ;                   |                  |
0000    148 ;                    ------------------
0000    149 ;
0000    150 ; Macro options
0000    151 ;
0000    152         .DSABL  GBL                     ; no undefined references
0000    153         .ENABL  FPT                     ; rounded arithmetic
0000    154 ;
0000    155 ; External references
0000    156 ;
0000    157         .EXTRN  LIB$GET_VM
0000    158         .EXTRN  LIB$FREE_VM
0000    159         .EXTRN  LIB$STOP                ; program abort
0000    160         .EXTRN  PAS$C_DFLTLINLI         ; default linelimit
0000    161         .EXTRN  PAS$_ERRACCFIL          ; PASCAL error message #8304
0000    162 ;
0000    163         .GLOBL  PAS$BLANK_R3
0000    164 ;
0000    165 ; Provide definitions of system values
0000    166 ;
0000    167         $DEVDEF                         ; device definitions
0000    168         $TRNLOGDEF
0000    169         $FABDEF
0000    170         $FORDEF                         ; FORTRAN error definitions
0000    171         $NAMDEF
0000    172         $RABDEF
```

```
                     0000   173          $RMSDEF                         ; for status code checking
                     0000   174          $stsdef                         ; status codes
                     0000   175          $SSDEF                          ; for system services return codes
                     0000   176 ;
                     0000   177 ; PASCAL compiler constants
                     0000   178 ;
                     0000   179 ; NOTE: The constants below with the names 'PAS$C_XXXXX' are
                     0000   180 ; used in the PASCAL compiler with the names 'XXXXX'. If the
                     0000   181 ; values in the compiler are altered then the below values
                     0000   182 ; must be altered accordingly.
                     0000   183 ;
00000101             0000   184          PAS$C_DFLTRECSI = 257           ; default buffer size
                     0000   185 ;        PAS$C_NIL = 0                   ; NIL pointer
00000001             0000   186          PAS$C_TRUE = 1                  ; TRUE
00000000             0000   187          PAS$C_FALSE = 0                 ; FALSE
00000000             0000   188          PAS$C_NOCARR = 0                ; no carriage control
                     0000   189 ;        PAS$C_CARRIAGE = 1              ; FORTRAN carriage control
00000002             0000   190          PAS$C_LIST = 2                  ; LIST carriage control
00000003             0000   191          PAS$C_PRN = 3                   ; PRN carriage control
                     0000   192 ;
                     0000   193 ; PRN carriage control constants
                     0000   194 ;
00008D01             0000   195          PRN_CRLF = ^X8D01               ; PRN carriage control constant
                     0000   196                                          ; for <LF> <text> <CR>
00000000             0000   197          PRN_NULL = ^X0000               ; PRN carriage control constant
                     0000   198                                          ; for no carriage control
00000001             0000   199          PRN_LF = ^X0001                 ; PRN carriage control constant
                     0000   200                                          ; for <LF> <prompt>
00008D00             0000   201          PRN_CR = ^X8D00                 ; PRN carriage control constant
                     0000   202                                          ; for <text> <CR>
                     0000   203
                     0000   204 ;
                     0000   205 ; File status block constants
                     0000   206 ;
00000018             0000   207          FSB$C_BLN = ^X18                ; FSB block length
00000005             0000   208          FSB$V_OPEN = 5
00000001             0000   209          FSB$V_EOF = 1
00000002             0000   210          FSB$V_EOLN = 2
00000003             0000   211          FSB$V_GET = 3
00000004             0000   212          FSB$V_TXT = 4                   ; textfile flag
00000000             0000   213          FSB$V_RDLN = 0                  ; last access was READLN
00000006             0000   214          FSB$V_DIR = 6                   ; direct access flag
00000007             0000   215          FSB$V_PUT = 7
00000008             0000   216          FSB$V_INT = 8                   ; internal flag
00000009             0000   217          FSB$V_PRMT = 9                  ; prompt flag
0000000A             0000   218          FSB$V_OUTPUT = 10               ; OUTPUT file flag
                     0000   219 ;        FSB$V_ACTIN = 11                ; actual input flag
0000000C             0000   220          FSB$V_INPUT = 12                ; INPUT file flag
0000000D             0000   221          FSB$V_PROMPT = 13               ; prompt flag
0000000E             0000   222          FSB$V_WRITPRMT = 14             ; WRITELN is being called to do prompting
0000001E             0000   223          FSB$V_DELZ = 30                 ; delete file if empty
0000001F             0000   224          FSB$V_INC = 31                  ; included file flag
00000006             0000   225          FSB$B_CC = 6                    ; carriage control byte offset
00000020             0000   226          FSB$M_OPEN = ^X0020
00000002             0000   227          FSB$M_EOF = ^X0002
00000004             0000   228          FSB$M_EOLN = ^X0004
00000008             0000   229          FSB$M_GET = ^X0008
```

```
                     0000   230 ;       FSB$M_PRMT = ^X0200
00000080             0000   231         FSB$M_PUT = ^X00000080
                     0000   232 ;       FSB$M_TXT = ^X0010
00000001             0000   233         FSB$M_RDLN = ^X0001
                     0000   234 ;       FSB$M_DIR = ^X00000040
                     0000   235 ;       FSB$M_INT = ^X00000100
00000400             0000   236         FSB$M_OUTPUT = ^X0400
00000800             0000   237         FSB$M_ACTIN = ^X0800
00001000             0000   238         FSB$M_INPUT = ^X1000
00002000             0000   239         FSB$M_PROMPT = ^X2000
00004000             0000   240         FSB$M_WRITPRMT = ^X4000
40000000             0000   241         FSB$M_DELZ = ^X40000000
80000000             0000   242         FSB$M_INC = ^X80000000
00000010             0000   243         FSB$L_CNT = 16                  ; line count (textfiles)
                     0000   244 ;       FSB$L_INC = 20                  ; INCLUDE block address
0000000C             0000   245         FSB$L_LIM = 12                  ; linelimit
00000008             0000   246         FSB$L_LST = 8                   ; last word offset
00000014             0000   247         FSB$L_PFSB = 20                 ; Related file FSB for prompting
                     0000   248                                         ; for INPUT, has address of OUTPUT FSB
                     0000   249                                         ; for OUTPUT, has address of INPUT FSB
                     0000   250                                         ; (shares storage with include address
                     0000   251                                         ; and direct access rec buf address)
00000014             0000   252         FSB$L_REC = 20                  ; record buffer address for
                     0000   253                                         ; direct access (shares storage
                     0000   254                                         ; with include address and related
                     0000   255                                         ; file FSB)
00000004             0000   256         FSB$L_STA = 4                   ; status word offset
                     0000   257 ;
                     0000   258 ; Character constants
                     0000   259 ;
00000009             0000   260         TAB = ^X09
00000020             0000   261         SPACE = ^X20
                     0000   262 ;       DOLLAR = ^X24
                     0000   263 ;       FORMFEED = ^XC
                     0000   264 ;       STAR = ^X2A
                     0000   265 ;       PLUS = ^X2B
                     0000   266 ;       MINUS = ^X2D
                     0000   267 ;       POINT = ^X2E
                     0000   268 ;       ZERO = ^X30
                     0000   269 ;       ONE = ^X31
                     0000   270 ;       NINE = ^X39
                     0000   271 ;       AA    ^^41
                     0000   272 ;       DD      44
                     0000   273 ;       EE    ,5
                     0000   274 ;       ZZ = ^X5A
                     0000   275 ;       UNDERSCORE = ^X5F
00000061             0000   276         AA_SMALL = ^X61
0000007A             0000   277         ZZ_SMALL = ^X7A
                     0000   278 ;
                     0000   279 ;
                 00000000   280         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                     0000   281 ;
                     0000   282 ;       **********************
                     0000   283 ;       *                    *
                     0000   284 ;       *    PAS$READCK      *
                     0000   285 ;       *                    *
                     0000   286 ;       **********************
```

```
                        0000   287 ;
                        0000   288 ; Argument offsets
                        0000   289 ;
                        0000   290 ;       AP                                  ; number of arguments (1)
                00000004 0000   291        FSB_DISP = 04                       ; address of FSB
                        0000   292 ;
                    0040 0000   293        .ENTRY  PAS$READOK,^M<R6>
      56    04 AC   D0   0002   294        MOVL    FSB_DISP(AP),R6             ; R6 = address of FSB
   0B 04 A6    03   E1   0006   295        BBC     #FSB$V_GET,FSB$L_STA(R6),910$
                        000B   296                                            ; read access allowed?
00000921'EF     6C   FA   000B   297        CALLG   (AP),PAS$EOF
          16 50     E8   0012   298        BLBS    R0,920$                     ; read past EOF?
                    04   0015   299        RET
                        0016   300 ;
                        0016   301 ; Read access not allowed
                        0016   302 ;
                        0016   303  910$:
    7E    8334 8F   3C   0016   304        MOVZWL  #^X8334,-(SP)
    7E    0090 C6   9A   001B   305        MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R6),-(SP)
          0088 C6   DD   0020   306        PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R6)
000000E4'EF     03   FB   0024   307        CALLS   #3,PAS$IOERROR
                        002B   308 ;
                        002B   309 ; Read past end-of-file
                        002B   310 ;
                        002B   311  920$:
  0001827A 8F   DD   002B   312        PUSHL   #RMS$_EOF
    7E    0090 C6   9A   0031   313        MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R6),-(SP)
          0088 C6   DD   0036   314        PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R6)
000000E4'EF     03   FB   003A   315        CALLS   #3,PAS$IOERROR
                        0041   316 ;
                        0041   317 ;
                00000041 0041   318        .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                        0041   319 ;
                        0041   320 ;       ********************
                        0041   321 ;       *                  *
                        0041   322 ;       *   PAS$WRITEOK    *
                        0041   323 ;       *                  *
                        0041   324 ;       ********************
                        0041   325 ;
                        0041   326 ; Argument offsets
                        0041   327 ;
                        0041   328 ;       AP:                                 ; number of arguments (1)
                00000004 0041   329        FSB_DISP = 04                       ; FSB address
                        0041   330 ;
                    0040 0041   331        .ENTRY  PAS$WRITEOK,^M<R6>
      56    04 AC   D0   0043   332        MOVL    FSB_DISP(AP),R6             ; R6 = address of FSB
   01 04 A6    07   E1   0047   333        BBC     #FSB$V_PUT,FSB$L_STA(R6),910$
                        004C   334                                            ; WRITE access allowed?
                    04  004C   335        RET
                        004D   336 ;
                        004D   337 ; WRITE access not allowed
                        004D   338 ;
                        004D   339  910$:
    7E    8344 8F   3C   004D   340        MOVZWL  #^X8344,-(SP)
    7E    0090 C6   9A   0052   341        MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R6),-(SP)
          0088 C6   DD   0057   342        PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R6)
000000E4'EF     03   FB   005B   343        CALLS   #3,PAS$IOERROR
```

```
                        0062   344 ;
                        0062   345 ;
                 00000062   346         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                        0062   347 ;
                        0062   348 ;       ********************
                        0062   349 ;       *                  *
                        0062   350 ;       *  PAS$BUFFEROVER  *
                        0062   351 ;       *                  *
                        0062   352 ;       ********************
                        0062   353 ;
                        0062   354 ; Argument offsets
                        0062   355 ;
                        0062   356 ;       AP                                      ; number of arguments (1)
                 00000004   0062   357         FSB_DISP = 04                           ; FSB address
                        0062   358 ;
                    0040   0062   359         .ENTRY  PAS$BUFFEROVER,^M<R6>
              56    04 AC    D0   0064   360         MOVL    FSB_DISP(AP),R6         ; R6 = address of FSB
              7E    38 A6    3C   0068   361         MOVZWL  <FSB$C_BLN+RAB$W_USZ>(R6),-(SP)
                             006C   362                                         ; pass buffer size
           7E    8384 8F    3C   006C   363         MOVZWL  #^X8384, -(SP)          ; pass error number
           7E    0090 C6    9A   0071   364         MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R6),-(SP)
                 0088 C6    DD   0076   365         PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R6)
     000000E4'EF    04    FB   007A   366         CALLS   #4,PAS$IOERROR
                        0081   367 ;
                        0081   368 ;
                 00000081   369         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                        0081   370 ;
                        0081   371 ;       ********************
                        0081   372 ;       *                  *
                        0081   373 ;       *   PAS$FILENAME   *
                        0081   374 ;       *                  *
                        0081   375 ;       ********************
                        0081   376 ;
                        0081   377 ; Argument offsets
                        0081   378 ;
                        0081   379 ;       AP                                      ; number of arguments (2)
                 00000004   0081   380         LEN_DISP = 04                           ; address of string length
                 00000008   0081   381         STR_DISP = 08                           ; address of string
                        0081   382 ;
                    03BC   0081   383         .ENTRY  PAS$FILENAME,^M<R2,R3,R4,R5,R7,R8,R9>
           57    04 BC    9A   0083   384         MOVZBL  @LEN_DISP(AP),R7        ; R7 = string length
           58    08 BC    D0   0087   385         MOVL    @STR_DISP(AP),R8        ; R8 = string address
              5E    08    C2   008B   386         SUBL2   #8,SP                           ; make room for string address
                             008E   387                                         ; and string length
              59    5E    D0   008E   388         MOVL    SP,R9                   ; save address
                    59    DD   0091   389         PUSHL   R9
           04 AE    57    D0   0093   390         MOVL    R7,4(SP)
                 04 AE    DF   0097   391         PUSHAL  4(SP)
     00000000'GF    02    FB   009A   392         CALLS   #2,G^LIB$GET_VM
              59    69    D0   00A1   393         MOVL    (R9),R9
        69    68    57    28   00A4   394         MOVC3   R7,(R8),(R9)
              58    59    D0   00A8   395         MOVL    R9,R8
                             00AB   396
           04 BC    57    90   00AB   397         MOVB    R7,@LEN_DISP(AP)        ; store new length
           08 BC    58    D0   00AF   398         MOVL    R8,@STR_DISP(AP)        ; store new string address
                       04   00B3   399         RET
                            00B4   400 ;
```

```
                        00B4   401 ;
                        00B4   402         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                        00B4   403 ;
                        00B4   404 ;       ********************
                        00B4   405 ;       *                  *
                        00B4   406 ;       *  PAS$STATUSUPDAT *
                        00B4   407 ;       *                  *
                        00B4   408 ;       ********************
                        00B4   409 ;
                        00B4   410 ; Updates the FSB status word based upon the current position of the file
                        00B4   411 ; pointer. If the pointer is greater than the last position (FSB$L_LST)
                        00B4   412 ; then RDLN is set true and EOF is checked. If the pointer is equal to
                        00B4   413 ; last then EOLN is set true. Otherwise EOLN and RDLN are left false.
                        00B4   414 ;
                        00B4   415 ; Argument offsets
                        00B4   416 ;
                        00B4   417 ;       AP                                  ; number of arguments (1)
                00000004 00B4   418         FSB_DISP = 04                       ; FSB address
                        00B4   419 ;
                    0040 00B4   420         .ENTRY  PAS$STATUSUPDAT,^M<R6>
          56    04 AC   D0 00B6   421         MOVL    FSB_DISP(AP),R6             ; R6 = address of FSB
    04 A6 00000800 8F   CA 00BA   422         BICL2   #FSB$M_ACTIN,FSB$L_STA(R6)
                        00C2   423                                             ; clear actual input flag
     08 A6      66   D1 00C2   424         CMPL    (R6),FSB$L_LST(R6)
                   17   19 00C6   425         BLSS    130$                        ; middle of line
                   10   13 00C8   426         BEQL    120$                        ; end of line
                        00CA   427 ;
                        00CA   428 ; Passed end-of-line, clear EOLN and set RDLN
                        00CA   429 ;
     04 A6          04   CA 00CA   430         BICL2   #FSB$M_EOLN,FSB$L_STA(R6)
     04 A6          01   C8 00CE   431         BISL2   #FSB$M_RDLN,FSB$L_STA(R6)
02 04 A6            01   E1 00D2   432         BBC     #FSB$V_EOF,FSB$L_STA(R6),110$
                   66   D4 00D7   433         CLRL    (R6)                        ; EOF, clear pointer
                        00D9   434   110$:
                   04   00D9   435         RET
                        00DA   436 ;
                        00DA   437 ; End-of-line, set EOLN flag
                        00DA   438 ;
                        00DA   439   120$:
     04 A6          04   C8 00DA   440         BISL2   #FSB$M_EOLN,FSB$L_STA(R6)
                   04   00DE   441         RET
                        00DF   442 ;
                        00DF   443 ; Middle of line
                        00DF   444 ;
                        00DF   445   130$:
     04 A6          04   CA 00DF   446         BICL2   #FSB$M_EOLN,FSB$L_STA(R6)
                        00E3   447                                             ; make sure EOLN clear
                   04   00E3   448         RET
                        00E4   449 ;
                        00E4   450 ;
                        00E4   451         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                        00E4   452 ;
                        00E4   453 ;       ********************
                        00E4   454 ;       *                  *
                        00E4   455 ;       *   PAS$IOERROR    *
                        00E4   456 ;       *                  *
                        00E4   457 ;       ********************
```

```
                                00E4    458 ;
                                00E4    459 ; Argument offsets
                                00E4    460 ;
                                00E4    461 ;       AP                              ; number of arguments (variable)
                    00000004    00E4    462         FNM_DISP = 04                   ; file name string address
                    00000008    00E4    463         FNL_DISP = 08                   ; file name string length
                                00E4    464 ;       ERRT                            ; an indefinite number of error
                                00E4    465                                         ; code may follow
                                00E4    466 ;        .
                                00E4    467 ;        .
                                00E4    468 ;        .
                                00E4    469 ;
                          00FC  00E4    470         .ENTRY  PAS$IOERROR,^M<R2,R3,R4,R5,R6,R7>
                       55 D4    00E6    471         CLRL    R5
                    04 AC DD    00E8    472         PUSHL   FNM_DISP(AP)
                    08 AC DD    00EB    473         PUSHL   FNL_DISP(AP)
                56  6C 02 C3    00EE    474         SUBL3   #2,(AP),R6              ; R6 = number of error arguments
                57  5C 04 C1    00F2    475         ADDL3   #4,AP,R7                ; R7 = address of arguments
                   54  5E DO    00F6    476         MOVL    SP,R4                   ; save the top of the stack address
                               00F9    477 110$:
                   57  04  C0   00F9    478         ADDL2   #4,R7                   ; update address for special codes
                               00FC    479 111$:
                   57  04  C0   00FC    480         ADDL2   #4,R7                   ; update address for non-special codes
            00009000 8F 67 D1   00FF    481         CMPL    (R7),#^X9000            ; test if RMS error
                       43 18    0106    482         BGEQ    130$
            00008374 8F 67 D1   0108    483         CMPL    (R7),#^X8374            ; test for line limit exceeded
                       09 13    010F    484         BEQL    112$
            00008384 8F 67 D1   0111    485         CMPL    (R7),#^X8384            ; test for line length exceeded
                       1B 12    0118    486         BNEQ    115$
                               011A    487 112$:                                   ; buffer overflow and linelimit
7E  67  00210000 8F    C1   011A    488         ADDL3   #^X210000,(R7),-(SP)    ; store error number,
                       03 DD    0122    489         PUSHL   #3                      ; store count of FAO arguments
                    04 A7 DD    0124    490         PUSHL   4(R7)                   ; first FAO argument
                       00 DD    0127    491         PUSHL   #0                      ; second and third FAO arguments are 0
                       00 DD    0129    492         PUSHL   #0
                55     05 C0    012B    493         ADDL2   #5,R5                   ; count number of arguments pushed
                       56 D7    012E    494         DECL    R6
                  C6   56 F5    0130    495         SOBGTR  R6,110$                 ; loop if more arguments
                       20 11    0133    496         BRB     140$
                               0135    497 115$:
7E  67  00210000 8F    C1   0135    498         ADDL3   #^X210000,(R7),-(SP)    ; store error number for all other I/O error
                       03 DD    013D    499         PUSHL   #3                      ; push FAO count of arguments
                       00 DD    013F    500         PUSHL   #0                      ; store three null arguments
                       7E 7C    0141    501         CLRQ    -(SP)
                55     05 C0    0143    502         ADDL2   #5,R5                   ; count number of arguments stored
                  B3   56 F5    0146    503         SOBGTR  R6,111$                 ; loop if more arguments
                       0A 11    0149    504         BRB     140$
                               014B    505 130$:
                       67 DD    014B    506         PUSHL   (R7)                    ; store RMS  error number
                       00 DD    014D    507         PUSHL   #0                      ; store null argument
                55     02 C0    014F    508         ADDL2   #2,R5                   ; count items pushed on the stack
                  A7   56 F5    0152    509         SOBGTR  R6,111$                 ; loop if more arguments
                               0155    510 140$:
                               0155    511 ;
                               0155    512 ; This section of code reverses the order of the arguments to lib$stop
                               0155    513 ; that are already on the stack.  Then the error ERRACCFIL is pushed and LIB$STOP
                               0155    514 ; is called.
```

```
                                0155   515
                                0155   516 ;
                     53   5E   DO  0155   517                MOVL    SP,R3                   ; save the address of the top of the stack
                     52   74   DO  0158   518 1$:            MOVL    -(R4),R2                ; move the first item from bottom of the stack
                     53   54   D1  015B   519                CMPL    R4,R3                   ; have all the items been switched?
                          08   1F  015E   520                BLSSU   2$                      ; done : all items are switched
                     64   63   DO  0160   521                MOVL    (R3),(R4)               ; switch two items
                     83   52   DO  0163   522                MOVL    R2,(R3)+                ;
                          F0   11  0166   523                BRB     1$
                                0168   524 2$:
                          00   DD  0168   525                PUSHL   #0                      ; store FAO arguments for ERRACCFIL
                   7E     F8 AD  7D  016A   526                MOVQ    -8(FP),-(SP)            ; push name of file being accessed
                          03   DD  016E   527                PUSHL   #3                      ; store count of FAO arguments
           7E  00000000'8F     04  C1  0170   528                ADDL3   #4,#PAS$_ERRACCFIL,-(SP)        ; store error message number
                          55   05  C0  0178   529                ADDL2   #5,R5                   ; count number of arguments stored
                00000000'GF    55  FB  017B   530                CALLS   R5,G^LIB$STOP           ; signal errors and stop
                                0182   531 ;
                                0182   532 ;
                                0182   533                .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                0182   534 ;
                                0182   535 ;               ********************
                                0182   536 ;               *                  *
                                0182   537 ;               *   PAS$BLANK_R3   *
                                0182   538 ;               *                  *
                                0182   539 ;               ********************
                                0182   540 ;
                                0182   541 ; A JSB routine which skips leading blanks on a file. It requires
                                0182   542 ; that R2 contain the FSB address. THe following values are returned
                                0182   543 ; in the designated registers.
                                0182   544 ;
                                0182   545 ;               R0:     number of bytes in the line after the skip
                                0182   546 ;               R1:     address of the byte located
                                0182   547 ;               R2:     address of the FSB (input)
                                0182   548 ;
                                0182   549 PAS$BLANK_R3:
                                0182   550   110$:
                          52   DD  0182   551                PUSHL   R2
                FE77      CF 01  FB  0184   552                CALLS   #1,PAS$READOK           ; check read ok and EOF
                          51   62  DO  0189   553                MOVL    (R2),R1                 ; R1 = current buffer position
           50     08 A2     51  C3  018C   554                SUBL3   R1,FSB$L_LST(R2),R0     ; R0 = remaining line length
                                0191   555   120$:
                61   50   20   3B  0191   556                SKPC    #SPACE,R0,(R1)          ; skip blanks
                          50   D5  0195   557                TSTL    R0                      ; test for end-of-line
                          11   13  0197   558                BEQL    130$
                     53   50   DO  0199   559                MOVL    R0,R3
                61   50   09   3B  019C   560                SKPC    #TAB,R0,(R1)            ; skip tabs
                          50   D5  01A0   561                TSTL    R0                      ; test for end-of-line
                          06   13  01A2   562                BEQL    130$
                     53   50   D1  01A4   563                CMPL    R0,R3                   ; skipped any tabs?
                          E8   12  01A7   564                BNEQ    120$
                               05  01A9   565                RSB
                                01AA   566   130$:
                  04 A2    01   C8  01AA   567                BISL2   #FSB$M_RDLN,FSB$L_STA(R2)
                                01AE   568                                              ; force next line
                          D2   11  01AE   569                BRB     110$
                                01B0   570 ;
                                01B0   571 ;
```

```
                        000001B0   572          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                            01B0   573 ;
                            01B0   574 ;        ********************
                            01B0   575 ;        *                  *
                            01B0   576 ;        *     PAS$INPUT    *
                            01B0   577 ;        *                  *
                            01B0   578 ;        ********************
                            01B0   579 ;
                            01B0   580 ; Initializes, opens, and resets the standard file INPUT.
                            01B0   581 ;
                            01B0   582 ; Argument offsets
                            01B0   583 ;
                            01B0   584 ;        AP                                  ; number of arguments (1)
                  00000004  01B0   585          FSB_DISP = 04                       ; FSB address
                            01B0   586 ;
                            01B0   587 ; Constants
                            01B0   588 ;
                  00000009  01B0   589                  INPUTLEN = 9
54 55 50 4E 49 24 53 41 50  01B0   590 PASINPUT:       .ASCII  /PAS$INPUT/
54 55 50 4E 49 24 53 59 53  01B9   591 SYSINPUT:       .ASCII  /SYS$INPUT/
                  00000009  01C2   592 PASDESCR:       .LONG   INPUTLEN            ; create a descriptor for PAS$INPUT
                  000001B0' 01C6   593                  .LONG   PASINPUT
                            01CA   594 ;
                      003C  01CA   595          .ENTRY  PAS$INPUT,^M<R2,R3,R4,R5>
                            01CC   596 ;
                            01CC   597 ; Initialize
                            01CC   598 ;
                 5E   3F    C2  01CC   599          SUBL2   #63,SP                      ; clear 63 bytes on the stack
                      5E    DD  01CF   600          PUSHL   SP                          ; create a descriptor for RSLBUF
                      3F    DD  01D1   601          PUSHL   #63
                   54   5E    D0  01D3   602          MOVL    SP,R4                       ; save the address of the descriptor
              55   04   AC    D0  01D6   603          MOVL    FSB_DISP(AP),R5             ; R5 = address of FSB
            00000101   8F    DD  01DA   604          PUSHL   #PAS$C_DFLTRECSI            ; maximum buffer length
                      01    DD  01E0   605          PUSHL   #1                          ; textfile
                      00    DD  01E2   606          PUSHL   #0                          ; external file
                      09    DD  01E4   607          PUSHL   #INPUTLEN                   ; input name string length
                            01E6   608          $TRNLOG_S       LOGNAM=PASDESCR,RSLBUF=(R4)     ; try to translate PAS$INPUT
      00000629 8F   50    D1  01FA   609          CMPL    R0,#SS$_NOTRAN              ; on error,
                      05    13  0201   610          BEQL    1$                          ; use SYS$INPUT
                  AA  AF    DF  0203   611          PUSHAL  PASINPUT                    ; otherwise, use PAS$INPUT
                      03    11  0206   612          BRB     2$
                  AE  AF    DF  0208   613 1$:      PUSHAL  SYSINPUT
                      55    DD  020B   614 2$:      PUSHL   R5                          ; FSB address
      0000037C'EF   06    FB  020D   615          CALLS   #6,PAS$INITFILES
                            0214   616 ;
                            0214   617 ; Fix up RAB, FAB, and NAM blocks
                            0214   618 ;
              52   55    18  C1  0214   619          ADDL3   #FSB$C_BLN,R5,R2            ; R2 = address of RAB
53    52    00000044  8F    C1  0218   620          ADDL3   #RAB$C_BLN,R2,R3            ; R3 = address of FAB
54    53    00000050  8F    C1  0220   621          ADDL3   #FAB$C_BLN,R3,R4            ; R4 = address of NAM
            28  A3    54    D0  0228   622          MOVL    R4,FAB$L_NAM(R3)            ; NAM block address
                64    02    90  022C   623          MOVB    #NAM$C_BID,NAM$B_BID(R4)    ; block identification
             01 A4    38    90  022F   624          MOVB    #NAM$C_BLN_V2,NAM$B_BLN(R4) ; block length
                            0233   625 ;
                            0233   626 ; Open file
                            0233   627 ;
  04 A5   80000000  8F    C8  0233   628          BISL2   #FSB$M_INC,FSB$L_STA(R5); Fake INCLUDE'd file to get
```

```
                                023B   629                                     ; read-only access
                          00    DD  023B   630          PUSHL   #PAS$C_NOCARR              ; carriage control -- not used
                    00000101    8F  DD  023D   631          PUSHL   #PAS$C_DFLTRECSI
                                7E  7C  0243   632          CLRD    -(SP)
                                7E  7C  0245   633          CLRD    -(SP)
                                55  DD  0247   634          PUSHL   R5
                000003F5'EF     07  FB  0249   635          CALLS   #7,PAS$OPEN
            04 A5   80000000    8F  CA  0250   636          BICL2   #FSB$M_INC,FSB$L_STA(R5); Unfake INCLUDE'd file
                                    0258   637 ;
                                    0258   638 ; Reset file
                                    0258   639 ;
                                55  DD  0258   640          PUSHL   R5
                00000638'EF     01  FB  025A   641          CALLS   #1,PAS$RESET
            04 A5   00001000    8F  C8  0261   642          BISL2   #FSB$M_INPUT,FSB$L_STA(R5)
                                    0269   643                                     ; set input flag
                                04  0269   644          RET
                                    026A   645 ;
                                    026A   646 ;
                                0000026A   647          .PSECT  _PAS$CODE       PIC,EXE,SHR,NOWRT
                                    026A   648 ;
                                    026A   649 ;        ********************
                                    026A   650 ;        *                  *
                                    026A   651 ;        *    PAS$OUTPUT    *
                                    026A   652 ;        *                  *
                                    026A   653 ;        ********************
                                    026A   654 ;
                                    026A   655 ; Initializes, creates, and rewrites the standard file OUTPUT.
                                    026A   656 ;
                                    026A   657 ; Argument offsets
                                    026A   658 ;
                                    026A   659 ;        AP                              ; number of arguments (2)
                          00000004  026A   660          FSB_DISP = 04                   ; address of OUTPUT FSB
                          00000008  026A   661          INP_DISP = 08                   ; address of INPUT FSB
                                    026A   662 ;
                                    026A   663 ; Constants
                                    026A   664 ;
                          0000000A  026A   665                  OUTPUTLEN = 10
54 55 50 54 55 4F 24 53 59 53       026A   666 SYSOUTPUT:       .ASCII  /SYS$OUTPUT/            ; changed from PAS$OUTPUT for V1.2
54 55 50 54 55 4F 24 53 41 50       0274   667 PASOUTPUT:      .ASCII  /PAS$OUTPUT/
                          0000000A  027E   668 OUTDESCR:       .LONG   OUTPUTLEN
                          00000274' 0282   669                  .LONG   PASOUTPUT
                                    0286   670 ;
                              01FC  0286   671          .ENTRY  PAS$OUTPUT,^M<R2,R3,R4,R5,R6,R7,R8>
                                    0288   672 ;
                                    0288   673 ; Initialize file
                                    0288   674 ;
                            5E  3F  C2  0288   675          SUBL2   #63,SP                  ; put a 63 byte buffer on the stack
                                5E  DD  028B   676          PUSHL   SP                      ; create the descriptor for RSLBUF
                                3F  DD  028D   677          PUSHL   #63
                            52  5E  D0  028F   678          MOVL    SP,R2                   ; save the address of the descriptor
                         56   04  AC  D0  0292   679          MOVL    FSB_DISP(AP),R6         ; R6 = address of OUTPUT FSB
                57  56  0000005C  8F  C1  0296   680          ADDL3   #<FSB$C_BLN+RAB$C_BLN>,R6,R7
                                    029E   681                                      ; R7 = address of OUTPUT FAB
                58  57  00000050  8F  C1  029E   682          ADDL3   #FAB$C_BLN,R7,R8        ; R8 = NAM block address
                          00000101  8F  DD  02A6   683          PUSHL   #PAS$C_DFLTRECSI        ; maximum record size
                                01  DD  02AC   684          PUSHL   #1                      ; textfile
                                00  DD  02AE   685          PUSHL   #0                      ; external file
```

```
                         0A   DD   02B0   686          PUSHL   #OUTPUTLEN                      ; output name string length
                              02B2   687          $TRNLOG_S       LOGNAM=OUTDESCR,RSLBUF=(R2)     ; try to translate PAS$OUTPU
          00000629 8F    50   D1   02C6   688          CMPL    R0,#SS$_NOTRAN                  ; on error,
                         05   13   02CD   689          BEQL    1$                              ; use SYS$OUTPUT
                      A2 AF   DF   02CF   690          PUSHAL  PASOUTPUT                       ; otherwise, use PAS$OUTPUT
                         03   11   02D2   691          BRB     2$
                      93 AF   DF   02D4   692 1$:      PUSHAL  SYSOUTPUT                       ; output name string address
                         56   DD   02D7   693 2$:      PUSHL   R6                              ; FSB address
          0000037C'EF    06   FB   02D9   694          CALLS   #6,PAS$INITFILES
                              02E0   695 ;
                              02E0   696 ; Create file
                              02E0   697 ;
                   28 A7      58   D0   02E0   698          MOVL    R8,FAB$L_NAM(R7)        ; link NAM block
    68    38    00   68      00   2C   02E4   699          MOVC5   #0,(R8),#0,#NAM$C_BLN_V2,(R8)
                              02EA   700                                                   ; clear NAM block
                      68      02   90   02EA   701          MOVB    #NAM$C_BID,NAM$B_BID(R8)
                   01 A8      38   90   02ED   702          MOVB    #NAM$C_BLN_V2,NAM$B_BLN(R8)
                         02   DD   02F1   703          PUSHL   #PAS$C_LIST                     ; carriage control
                00000103 8F   DD   02F3   704          PUSHL   #PAS$C_DFLTRECSI+2              ; record length (allow 2 bytes for
                              02F9   705                                                   ; PRN carriage control buffer)
                         7E   7C   02F9   706          CLRD    -(SP)
                         7E   7C   02FB   707          CLRD    -(SP)
                         56   DD   02FD   708          PUSHL   R6                              ; FSB address
          000004B6'EF    07   FB   02FF   709          CALLS   #7,PAS$CREATE
                   3C A6 02   C0   0306   710          ADDL2   #2,FSB$C_BLN+RAB$L_UBF(R6)
                              030A   711                                                   ; reserve 2 bytes for PRN carriage
                              030A   712                                                   ; control
                              030A   713 ;
                              030A   714 ; Rewrite file
                              030A   715 ;
                         56   DD   030A   716          PUSHL   R6
             000006DF'EF    01   FB   030C   717          CALLS   #1,PAS$REWRITE
    04 A6    00000400 8F    C8   0313   718          BISL2   #FSB$M_OUTPUT,FSB$L_STA(R6)
                              031B   719                                                   ; set OUTPUT flag
                55    08 AC   D0   031B   720          MOVL    INP_DISP(AP),R5             ; R5 = address of INPUT FSB
                         56   13   031F   721          BEQL    10$                         ; done if no INPUT file
                         02   E1   0321   722          BBC     #DEV$V_TRM,-
                50 009C C5         0323   723                  FSB$C_BLN+RAB$C_BLN+FAB$L_DEV(R5),10$ ;
                              0327   724                                                   ; done if INPUT is not a terminal
             4B 40 A7    02   E1   0327   725          BBC     #DEV$V_TRM,FAB$L_DEV(R7),10$
                              032C   726                                                   ; done if OUTPUT is not a terminal
                              032C   727 ;
                              032C   728 ; INPUT and OUTPUT are going to terminals.  Reopen OUTPUT with PRN carriage
                              032C   729 ; control, and set FSB's to do prompting.
                              032C   730 ;
                              032C   731          $CLOSE  FAB=R7                          ; close OUTPUT
                1F A7    03   90   0335   732          MOVB    #FAB$C_VFC,FAB$B_RFM(R7); set fixed-length control format
                3F A7    02   90   0339   733          MOVB    #2,FAB$B_FSZ(R7)            ; set control field size to 2
                      1E A7 94   033D   734          CLRB    FAB$B_RAT(R7)               ; clear old carriage control
    1E A7    01   02    01   F0   0340   735          INSV    #1,#FAB$V_PRN,#1,FAB$B_RAT(R7)
                              0346   736                                                   ; set PRN carriage control
                              0346   737          $CREATE FAB=R7                          ; create new OUTPUT file
                40 A6    02   C3   034F   738          SUBL3   #2,FSB$C_BLN+RAB$L_RBF(R6),-
                      44 A6       0353   739                  FSB$C_BLN+RAB$L_RHB(R6)
                              0355   740                                                   ; set RAB header buffer address
                              0355   741                                                   ; (address of PRN carriage
                              0355   742                                                   ; control buffer)
```

```
                        0355   743          $CONNECT RAB=FSB$C_BLN(R6)      ; re-connect RAB
04 A5   01   09   01    F0  035F   744          INSV    #1,#FSB$V_PRMT,#1,FSB$L_STA(R5)
                        0365   745                                          ; set prompt bit of INPUT FSB
          14 A5   56    D0  0365   746          MOVL    R6,FSB$L_PFSB(R5)       ; set related file FSB of INPUT
          14 A6   55    D0  0369   747          MOVL    R5,FSB$L_PFSB(R6)       ; set related file FSB of OUTPUT
       44 B6   8D01 8F   B0  036D   748          MOVW    #PRN_CRLF,@FSB$C_BLN+RAB$L_RHB(R6)
                        0373   749                                          ; initialize carriage control
          06 A6   03    90  0373   750          MOVB    #PAS$C_PRN,FSB$B_CC(R6) ; set PRN carriage control in FSB
                  04    0377   751 10$:         RET
                        0378   752 ;
                        0378   753 ;
                 00000378   754          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                        0378   755 ;
                        0378   756 ;       ********************
                        0378   757 ;       *                  *
                        0378   758 ;       *   PAS$INITFILES  *
                        0378   759 ;       *                  *
                        0378   760 ;       ********************
                        0378   761 ;
                        0378   762 ; Called at procedure entry time to initialize the FSB, RAB and FAB to
                        0378   763 ; PASCAL default values after clearing them.
                        0378   764 ;
                        0378   765 ; Argument offsets
                        0378   766 ;
                        0378   767 ;       AP                              ; number of arguments (6 per file)
                        0378   768 ;
                00000004 0378   769          FSB_DISP = 04                   ; FSB address
                00000008 0378   770          NAM_DISP = 08                   ; name string address
                0000000C 0378   771          LEN_DISP = 12                   ; name string length
                00000010 0378   772          EXT_DISP = 16                   ; external/internal flag
                        0378   773                                          ;   0 = external
                        0378   774                                          ;   1 = internal (delete on close)
                00000014 0378   775          TXT_DISP = 20                   ; textfile flag
                        0378   776                                          ;   0 = non-textfile
                        0378   777                                          ;   1 = textfile
                00000018 0378   778          MRL_DISP = 24                   ; maximum record size
                        0378   779 ;
                        0378   780 ; Arguments above are repeated for each file
                        0378   781 ;
                        0378   782 ; Default file name
                        0378   783 ;
                  54 41 44 2E   0378   784 DFNAM: .ASCII  /.DAT/
                00000004 037C   785          DFLEN = 4
                        037C   786 ;
                    0FFC 037C   787          .ENTRY  PAS$INITFILES,^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
             5B   6C   06   C7  037E   788          DIVL3   #6,(AP),R11             ; R11 = # of files (N)
                  5A   5C   D0  0382   789          MOVL    AP,R10                  ; R10 = address of AP
                        0385   790                                          ; (simulates AP position in loop)
                        0385   791    10$:                                 ; loop until all files initialized
          56   04 AA   D0  0385   792          MOVL    FSB_DISP(R10),R6        ; R6 = address of FSB
       57   18   56   C1  0389   793          ADDL3   R6,#FSB$C_BLN,R7        ; R7 = address of RAB
    58   00000044 8F   57   C1  038D   794          ADDL3   R7,#RAB$C_BLN,R8        ; R8 = address of FAB
                        0395   795 ;
                        0395   796 ; Clear control blocks (in case on stack)
                        0395   797 ;
66   00AC 8F   00   66   00   2C  0395   798          MOVC5   #0,(R6),#0, -
                        039D   799                          #<FSB$C_BLN+RAB$C_BLN+FAB$C_BLN>,(R6)
```

```
                                    039D    800 ;
                                    039D    801 ; Initialize FSB
                                    039D    802 ;
04 A6   01   04   14 AA   F0        039D    803         INSV    TXT_DISP(R10),#FSB$V_TXT,#1,FSB$L_STA(R6)
                                    03A4    804                                         ; textfile flag
04 A6   01   08   10 AA   F0        03A4    805         INSV    EXT_DISP(R10),#FSB$V_INT,#1,FSB$L_STA(R6)
                                    03AB    806                                         ; internal flag
    0C A6   00000000'GF   D0        03AB    807         MOVL    G^PAS$C_DFLTLINLI,FSB$L_LIM(R6)
                                    03B3    808                                         ; linelimit
                                    03B3    809 ;
                                    03B3    810 ; Initialize RAB
                                    03B3    811 ;
            67   01   90            03B3    812         MOVB    #RAB$C_BID,RAB$B_BID(R7); block ID
         01 A7    44 8F   90        03B6    813         MOVB    #RAB$C_BLN,RAB$B_BLN(R7); block length
            3C A7   58   D0         03BB    814         MOVL    R8,RAB$L_FAB(R7)        ; FAB address
            1E A7   00   90         03BF    815         MOVB    #RAB$C_SEQ,RAB$B_RAC(R7); sequential record access
         20 A7    18 AA   B0        03C3    816         MOVW    MRL_DISP(R10),RAB$W_USZ(R7)
                                    03C8    817                                         ; set buffer size
                                    03C8    818 ;
                                    03C8    819 ; Initialize FAB
                                    03C8    820 ;
            68   03   90            03C8    821         MOVB    #FAB$C_BID,FAB$B_BID(R8); block ID
         01 A8    50 8F   90        03CB    822         MOVB    #FAB$C_BLN,FAB$B_BLN(R8); block length
         2C A8    08 AA   D0        03D0    823         MOVL    NAM_DISP(R10),FAB$L_FNA(R8)
                                    03D5    824                                         ; file specification address
         34 A8    0C AA   90        03D5    825         MOVB    LEN_DISP(R10),FAB$B_FNS(R8)
                                    03DA    826                                         ; file specification size
         30 A8    9B AF   DE        03DA    827         MOVAL   DFNAM,FAB$L_DNA(R8)     ; default file name
            35 A8   04   90         03DF    828         MOVB    #DFLEN,FAB$B_DNS(R8)    ; default file name length
                                    03E3    829
                                    03E3    830 ;       The call to PAS$FILENAME was removed under the assumption that
                                    03E3    831 ;       any name passed to PAS$INITFILES is in read/only static storage
                                    03E3    832 ;       and does not need space for it allocated by LIB$GET_VM, nor
                                    03E3    833 ;       does it need leading blanks stripped from it.
                                    03E3    834 ;
                                    03E3    835 ;       PUSHAL  FAB$L_FNA(R8)           ; file name string address
                                    03E3    836 ;       PUSHAB  FAB$B_FNS(R8)           ; file name string length address
                                    03E3    837 ;       CALLS   #2,PAS$FILENAME
                                    03E3    838
         1D A8   00   90            03E3    839         MOVB    #FAB$C_SEQ,FAB$B_ORG(R8); sequential files only
04 A8   01   04   10 AA   F0        03E7    840         INSV    EXT_DISP(R10),#FAB$V_TMD,#1,FAB$L_FOP(R8)
                                    03EE    841                                         ; temporary file
            5A   18   C0            03EE    842         ADDL2   #24,R10                 ; get next FSB address
              91 5B   F5            03F1    843         SOBGTR  R11,10$                 ; decrement # of files counter
                     04             03F4    844         RET
                                    03F5    845 ;
                                    03F5    846 ;
                                00003F5     847         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                    03F5    848 ;
                                    03F5    849 ;       ********************
                                    03F5    850 ;       *                  *
                                    03F5    851 ;       *    PAS$OPEN      *
                                    03F5    852 ;       *                  *
                                    03F5    853 ;       ********************
                                    03F5    854 ;
                                    03F5    855 ; Opens an existing file. The FSB, RAB, and FAB must have been
                                    03F5    856 ; initialized by a call to PAS$INIT before this routine is called.
```

```
                     03F5   857 ; Space for the record buffer is alocated by a call to 'GET_VM'.
                     03F5   858 ; Any error in opening or connecting the file causes a runtime error.
                     03F5   859 ;
                     03F5   860 ; Argument offsets
                     03F5   861 ;
                     03F5   862 ;       AP                              ; number of arguments (7)
            00000004 03F5   863        FSB_DISP = 04                   ; FSB address
            00000008 03F5   864        NAM_DISP = 08                   ; file name string
                     03F5   865                                        ;    0 = use default
            0000000C 03F5   866        LEN_DISP = 12                   ; file name string length
            00000010 03F5   867        ACC_DISP = 16                   ; RMS record access mode
                     03F5   868                                        ;    0 = sequential
                     03F5   869                                        ;    1 = direct
            00000014 03F5   870        FMT_DISP = 20                   ; RMS record format --- not used
                     03F5   871                                        ;    0 = variable
                     03F5   872                                        ;    1 = fixed
            00000018 03F5   873        MRL_DISP = 24                   ; maximum record length (buffer
                     03F5   874                                        ; size)
                     03F5   875                                        ;    negative = allocate zero
                     03F5   876                                        ;       but set USZ (used by
                     03F5   877                                        ;       compiler only)
                     03F5   878                                        ;    zero = not used
                     03F5   879                                        ;    positive = allocate as
                     03F5   880                                        ;       requested and check ok
            0000001C 03F5   881        CAR_DISP = 28                   ; carriage control flag --- not used
                     03F5   882                                        ;    0 = no carriage control
                     03F5   883                                        ;    1 = fortran carriage control
                     03F5   884                                        ;    2 = CR/LF carriage control (LIST)
                     03F5   885 ;
                 001C 03F5   886        .ENTRY  PAS$OPEN,^M<R2,R3,R4>
      000005CD'EF    16 03F7   887        JSB     CBINIT_R4               ; control block initialization
                     03FD   888                                        ; returns R2 = address of FSB
                     03FD   889                                        ;         R3 = address of RAB
                     03FD   890                                        ;         R4 = address of FAB
                     03FD   891 ;
                     03FD   892 ; Open file with correct privilages (read and/or write)
                     03FD   893 ;
                     03FD   894
                     03FD   895 ; INCLUDE'd files are opened Read-only
                     03FD   896 ;
   2C 04 A2    1F    E0 03FD   897        BBS     #FSB$V_INC,FSB$L_STA(R2),110$
                     0402   898 ;
                     0402   899 ; Read and write access
                     0402   900 ;
         16 A4       94 0402   901        CLRB    FAB$B_FAC(R4)
      16 A4    02     88 0405   902        BISB2   #FAB$M_GET,FAB$B_FAC(R4)
      16 A4    01     88 0409   903        BISB2   #FAB$M_PUT,FAB$B_FAC(R4)
      16 A4    10     88 040D   904        BISB2   #FAB$M_TRN,FAB$B_FAC(R4)
                     0411   905        $OPEN   FAB=R4
   03 04 A2    08     E1 041A   906        bbc     #fsb$v_int,fsb$l_sta(r2),101$   ; Better not be an internal file
            0083    31 041F   907        brw     920$
                     0422   908 101$:
         4A 50       E8 0422   909        BLBS    R0,120$                 ; branch if ok
0001829A 8F    50    D1 0425   910        CMPL    R0,#RMS$_PRV            ; check for privilege violation
            64    12 042C   911        BNEQ    900$
                     042E   912 ;
                     042E   913 ; Read access only
```

```
                                042E    914 ;
                                042E    915 110$:
                          16 A4 94      042E    916         CLRB    FAB$B_FAC(R4)
                       16 A4  02 88     0431    917         BISB2   #FAB$M_GET,FAB$B_FAC(R4)
                                0435    918         $OPEN   FAB=R4
                          2E 50 E8      043E    919         BLBS    R0,120$                 ; branch if ok
                    4C 04 A2    1F E0   0441    920         BBS     #FSB$V_INC,FSB$L_STA(R2),900$
                                0446    921                                             ; error if INCLUDE'd file and can't
                                0446    922                                             ; get read access
                0001829A 8F     50 D1   0446    923         CMPL    R0,#RMS$_PRV
                                43 12   044D    924         BNEQ    900$
                                044F    925 ;
                                044F    926 ; Write access only
                                044F    927 ;
                          16 A4 94      044F    928         CLRB    FAB$B_FAC(R4)
                       16 A4  01 88     0452    929         BISB2   #FAB$M_PUT,FAB$B_FAC(R4)
                       16 A4  10 C8     0456    930         BISL2   #FAB$M_TRN,FAB$B_FAC(R4)
                                045A    931         $OPEN   FAB=R4
                          09 50 E8      0463    932         BLBS    R0,120$                 ; branch if ok
                0001829A 8F     50 D1   0466    933         CMPL    R0,#RMS$_PRV
                                23 12   046D    934         BNEQ    900$
                                046F    935 ;
                                046F    936 ; Connect the file
                                046F    937 ;
                                046F    938 120$:
                                046F    939         $CONNECT RAB=R3
                00018009 8F     50 D1   0478    940         CMPL    R0,#RMS$_PENDING        ; check for completion
                                09 12   047F    941         BNEQ    121$
                                0481    942         $WAIT   RAB=R3
                                048A    943 121$:
                          05 50 E9      048A    944         BLBC    R0,900$                 ; branch if error
                       04 A2  20 C8     048D    945         BISL2   #FSB$M_OPEN,FSB$L_STA(R2)
                                0491    946                                             ; set open flag
                                04      0491    947         RET
                                0492    948 ;
                                0492    949 ; Open error, send error message and stop
                                0492    950 ;
                                0492    951 900$:
                                50 DD   0492    952         PUSHL   R0                      ; RMS error
                    7E  8314    8F 3C   0494    953         MOVZWL  #^X8314,-(SP)           ; PASCAL error
                      7E    34 A4 9A    0499    954         MOVZBL  FAB$B_FNS(R4),-(SP)     ; file name string length
                            2C A4 DD    049D    955         PUSHL   FAB$L_FNA(R4)           ; file name string
                FC3F CF     04 FB       04A0    956         CALLS   #4,PAS$IOERROR
                                04A5    957 920$:
                7E  8314    8F 3C       04A5    958         movzwl  #^x8314,-(sp)           ; PASCAL error
                  7E    34 A4 9A        04AA    959         movzbl  fab$b_fns(r4),-(sp)     ; file name string length
                            2C A4 DD    04AE    960         pushl   fab$l_fna(r4)           ; file name string
                FC2E CF     03 FB       04B1    961         calls   #3,pas$ioerror
                                04B6    962 ;
                                04B6    963 ;
                            00004B6     964         .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                04B6    965 ;
                                04B6    966 ;       ********************
                                04B6    967 ;       *                  *
                                04B6    968 ;       *    PAS$CREATE    *
                                04B6    969 ;       *                  *
                                04B6    970 ;       ********************
```

```
                        04B6   971 ;
                        04B6   972 ; Creates a new file. The FSB, RAB, and FAB must have been initialized
                        04B6   973 ; by a call to PAS$INIT before this routine is called. Any error in
                        04B6   974 ; creating or connecting the file causes a runtime error.
                        04B6   975 ;
                        04B6   976 ; Argument offsets
                        04B6   977 ;
                        04B6   978 ;       AP                              ; number of arguments (7)
00000004                04B6   979         FSB_DISP = 04                   ; FSB address
00000008                04B6   980         NAM_DISP = 08                   ; file name string
                        04B6   981                                         ;    0 = use default
0000000C                04B6   982         LEN_DISP = 12                   ; file name string length
00000010                04B6   983         ACC_DISP = 16                   ; RMS record access mode
                        04B6   984                                         ;    0 = sequential
                        04B6   985                                         ;    1 = direct
00000014                04B6   986         FMT_DISP = 20                   ; RMS record format
                        04B6   987                                         ;    0 = variable
                        04B6   988                                         ;    1 = fixed
00000018                04B6   989         MRL_DISP = 24                   ; maximum record length(buffer size)
                        04B6   990                                         ; (buffer size)
                        04B6   991                                         ;    negative or zero = error
                        04B6   992                                         ;        (not used)
                        04B6   993                                         ;    positive = allocate requested
                        04B6   994                                         ;        amount
0000001C                04B6   995         CAR_DISP = 28                   ; carriage control
                        04B6   996                                         ;    0 = no carriage control
                        04B6   997                                         ;    1 = FORTRAN carriage control
                        04B6   998                                         ;    2 = CR/LF carriage control
                        04B6   999                                         ;        (LIST)
                        04B6  1000 ;
                 007C   04B6  1001         .ENTRY  PAS$CREATE,^M<R2,R3,R4,R5,R6>
      000005CD'EF  16   04B8  1002         JSB     CBINIT_R4               ; control block initialization
                        04BE  1003                                         ; returns R2 = address of FSB
                        04BE  1004                                         ;         R3 = address of RAB
                        04BE  1005                                         ;         R4 = address of FAB
           55   D4      04BE  1006         CLRL    R5                      ; clear test register
      56   1C AC   D0    04C0  1007         MOVL    CAR_DISP(AP),R6         ; get carriage control into R6
   59 04 A2   0A   E1    04C4  1008         BBC     #FSB$V_OUTPUT,FSB$L_STA(R2),200$
                         04C9  1009                                         ; check for file OUTPUT
         10 A2   D5      04C9  1010         TSTL    FSB$L_CNT(R2)           ; check line count
                 2C   12 04CC  1011         BNEQ    270$
         08 AC   D5      04CE  1012         TSTL    NAM_DISP(AP)            ; check standard parameters
                 27   12 04D1  1013         BNEQ    270$
         10 AC   D5      04D3  1014         TSTL    ACC_DISP(AP)
                 22   12 04D6  1015         BNEQ    270$
         14 AC   D5      04D8  1016         TSTL    FMT_DISP(AP)
                 1D   12 04DB  1017         BNEQ    270$
00000101 8F      18 AC  D1    04DD  1018         CMPL    MRL_DISP(AP),#PAS$C_DFLTRECSI
                 13   12 04E5  1019         BNEQ    270$
                         04E7  1020 ;
      06 A2      56   91 04E7  1021         CMPB    R6,FSB$B_CC(R2)         ; check for existing carriage
                         04EB  1022                                         ; control
                 01   12 04EB  1023         BNEQ    280$                    ; return if new one is the same
                      04 04ED  1024         RET
                         04EE  1025 ;
                         04EE  1026 280$:
      06 A2      03   91 04EE  1027         CMPB    #PAS$C_PRN,FSB$B_CC(R2) ; return if old one is PRN and
```

```
                     13   12   04F2  1028          BNEQ    260$                            ; new one is LIST
                02   56   91   04F4  1029          CMPB    R6,#PAS$C_LIST
                     04   12   04F7  1030          BNEQ    275$
                          04   04F9  1031          RET
                               04FA  1032 ;
                               04FA  1033 270$:
                   00BF   31   04FA  1034          BRW     910$
                               04FD  1035 275$:
                               04FD  1036 ;
                               04FD  1037 ;  Old carriage control is PRN and new is not.  Disable prompting.
                               04FD  1038 ;
                50   14 A2   D0   04FD  1039          MOVL    FSB$L_PFSB(R2),R0               ; R0 = address of INPUT FSB
      04 A0   01   09   00   F0   0501  1040          INSV    #0,#FSB$V_PRMT,#1,FSB$L_STA(R0)
                               0507  1041                                                  ; clear PROMPT bit
                               0507  1042 260$:
                     55   D6   0507  1043          INCL    R5                              ; set OUTPUT flag
       04 A2   40000000 8F   C8   0509  1044          BISL2   #FSB$M_DELZ,FSB$L_STA(R2)
                               0511  1045                                                  ; set delete flag
                     52   DD   0511  1046          PUSHL   R2
          00000852'EF   01   FB   0513  1047          CALLS   #1,PAS$CLOSEINOUT
       04 A4   00008000 8F   CA   051A  1048          BICL2   #FAB$M_DLT,FAB$L_FOP(R4); clear delete flag
                               0522  1049 200$:
             1C 04 A2   04   E1   0522  1050          BBC     #FSB$V_TXT,FSB$L_STA(R2),216$
                               0527  1051                                                  ; skip if binary
                   1E A4   94   0527  1052          CLRB    FAB$B_RAT(R4)                   ; clear field
                06 A2   56   90   052A  1053          MOVB    R6,FSB$B_CC(R2)                 ; set carriage control field
                     01   56   D1   052E  1054          CMPL    R6,#1                           ; set carriage control
                     10   19   0531  1055          BLSS    216$
                     08   14   0533  1056          BGTR    212$
      1E A4   01   00   01   F0   0535  1057          INSV    #1,#FAB$V_FTN,#1,FAB$B_RAT(R4)
                               053B  1058                                                  ; FORTRAN carriage control
                     06   11   053B  1059          BRB     216$
                               053D  1060 212$:
      1E A4   01   01   01   F0   053D  1061          INSV    #1,#FAB$V_CR,#1,FAB$B_RAT(R4)
                               0543  1062                                                  ; CR/LF carriage control
                               0543  1063 216$:
                14 AC   D5   0543  1064          TSTL    FMT_DISP(AP)                    ; record format
                     06   12   0546  1065          BNEQ    220$
                1F A4   02   90   0548  1066          MOVB    #FAB$C_VAR,FAB$B_RFM(R4); variable
                     09   11   054C  1067          BRB     221$
                               054E  1068 220$:
                1F A4   01   90   054E  1069          MOVB    #FAB$C_FIX,FAB$B_RFM(R4); fixed
             36 A4   18 AC   B0   0552  1070          MOVW    MRL_DISP(AP),FAB$W_MRS(R4)
                               0557  1071                                                  ; set maximum record size
                               0557  1072 ;
                               0557  1073 ; Create file with read and write access
                               0557  1074 ;
                               0557  1075 221$:
                16 A4   94   0557  1076          CLRB    FAB$B_FAC(R4)
                16 A4   02   88   055A  1077          BISB2   #FAB$M_GET,FAB$B_FAC(R4)
                16 A4   01   88   055E  1078          BISB2   #FAB$M_PUT,FAB$B_FAC(R4)
                16 A4   10   88   0562  1079          BISB2   #FAB$M_TRN,FAB$B_FAC(R4)
                               0566  1080          $CREATE FAB=R4
                   37 50   E9   056F  1081          BLBC    R0,900$                         ; Branch on error
                               0572  1082          $CONNECT RAB=R3
          00018009 8F   50   D1   057B  1083          CMPL    R0,#RMS$_PENDING                ; check for completion
                     09   12   0582  1084          BNEQ    131$
```

```
                                0584  1085          $WAIT   RAB=R3
                                058D  1086   131$:
                      19 50     E9  058D  1087          BLBC    R0,900$                 ; branch if error
                   04 A2  20    C8  0590  1088          BISL2   #FSB$M_OPEN,FSB$L_STA(R2)
                                0594  1089                                              ; set open flag
                         55     D5  0594  1090          TSTL    R5                      ; check for file OUTPUT
                         10     13  0596  1091          BEQL    160$
          04 A2   00000400 8F   C8  0598  1092          BISL2   #FSB$M_OUTPUT,FSB$L_STA(R2)
                                05A0  1093                                              ; set OUTPUT flag
          04 A2   40000000 8F   CA  05A0  1094          BICL2   #FSB$M_DELZ,FSB$L_STA(R2)
                                05A8  1095                                              ; clear delete flag
                                05A8  1096   160$:
                                04  05A8  1097          RET
                                05A9  1098 ;
                                05A9  1099 ; Create error, send error message and abort
                                05A9  1100 ;
                                05A9  1101   900$:
                         50     DD  05A9  1102          PUSHL   R0                      ; RMS error
                   7E    8314 8F   3C  05AB  1103          MOVZWL  #^X8314,-(SP)           ; PASCAL error
                     7E    34 A4   9A  05B0  1104          MOVZBL  FAB$B_FNS(R4),-(SP)     ; file name string length
                        2C A4      DD  05B4  1105          PUSHL   FAB$L_FNA(R4)           ; file name string
                   FB28 CF    04   FB  05B7  1106          CALLS   #4,PAS$IOERROR
                                05BC  1107 ;
                                05BC  1108 ; File OUTPUT erroneously opened
                                05BC  1109 ;
                                05BC  1110   910$:
                   7E    83F4 8F   3C  05BC  1111          MOVZWL  #^X83F4,-(SP)           ; error code
                     7E    34 A4   9A  05C1  1112          MOVZBL  FAB$B_FNS(R4),-(SP)     ; file name string length
                        2C A4      DD  05C5  1113          PUSHL   FAB$L_FNA(R4)           ; file name string
                   FB17 CF    03   FB  05C8  1114          CALLS   #3,PAS$IOERROR
                                05CD  1115 ;
                                05CD  1116 ;
                              000005CD  1117          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                05CD  1118 ;
                                05CD  1119 ;       ********************
                                05CD  1120 ;       *                  *
                                05CD  1121 ;       *     CBINIT_R4    *
                                05CD  1122 ;       *                  *
                                05CD  1123 ;       ********************
                                05CD  1124 ;
                                05CD  1125 ; This JSB routine initializes the RAB and FAB control blocks during
                                05CD  1126 ; an OPEN/CREATE request. The addresses of the FSB, RAB, and FAB are
                                05CD  1127 ; returned in registers R2, R3, and R4 respectively. Space for the user
                                05CD  1128 ; buffer is also allocated.
                                05CD  1129 ;
                                05CD  1130 CBINIT_R4:
                      52  04 AC     D0  05CD  1131          MOVL    FSB_DISP(AP),R2         ; R2 = address of FSB
                   53    18   52     C1  05D1  1132          ADDL3   R2,#FSB$C_BLN,R3        ; R3 = address of RAB
          54   00000044 8F    53     C1  05D5  1133          ADDL3   R3,#RAB$C_BLN,R4        ; R4 = address of FAB
                                05DD  1134 ;
                                05DD  1135 ; Allocate buffer space
                                05DD  1136 ;
                      18 AC     D5  05DD  1137          TSTL    MRL_DISP(AP)
                         28     15  05E0  1138          BLEQ    120$
                   2A 04 A2     0A  E0  05E2  1139          BBS     #FSB$V_OUTPUT,FSB$L_STA(R2),121$
                            24 A3     DF  05E7  1140          PUSHAL  RAB$L_UBF(R3)
                            18 AC     DF  05EA  1141          PUSHAL  MRL_DISP(AP)
```

```
                00000000'GF    02    FB  05ED  1142          CALLS   #2,G^LIB$GET_VM
                         1A 50 E8    05F4  1143          BLBS    R0,121$
                            50 DD    05F7  1144          PUSHL   R0
                      7E  8324 BF    3C  05F9  1145          MOVZWL  #^X8324,-(SP)
                        7E   34 A4    9A  05FE  1146          MOVZBL  FAB$B_FNS(R4),-(SP)
                             2C A4    DD  0602  1147          PUSHL   FAB$L_FNA(R4)
                       FADA CF    04    FB  0605  1148          CALLS   #4,PAS$IOERROR
                                        060A  1149  120$:
                                05    13  060A  1150          BEQL    121$
                          18 AC   18 AC    CE  060C  1151          MNEGL   MRL_DISP(AP),MRL_DISP(AP)
                                        0611  1152  121$:
                04 A2   01   06    10 AC    F0  0611  1153          INSV    ACC_DISP(AP),#FSB$V_DIR,#1,FSB$L_STA(R2)
                                        0618  1154                                  ; direct flag
                          20 A3   18 AC    B0  0618  1155          MOVW    MRL_DISP(AP),RAB$W_USZ(R3)
                                        061D  1156                                  ; user record area size
                                08 AC    D5  061D  1157          TSTL    NAM_DISP(AP)            ; check for file name
                                   15    13  0620  1158          BEQL    910$                    ; branch if no file name
                          2C A4   08 AC    D0  0622  1159          MOVL    NAM_DISP(AP),FAB$L_FNA(R4)
                                        0627  1160                                  ; file name string address
                          34 A4   0C AC    90  0627  1161          MOVB    LEN_DISP(AP),FAB$B_FNS(R4)
                                        062C  1162                                  ; file name string length
                                2C A4    DF  062C  1163          PUSHAL  FAB$L_FNA(R4)           ; file name string address
                                34 A4    9F  062F  1164          PUSHAB  FAB$B_FNS(R4)           ; file name string length address
                         FA4A CF    02    FB  0632  1165          CALLS   #2,PAS$FILENAME         ; translate file name
                                        0637  1166  910$:
                                05      0637  1167          RSB
                                        0638  1168 ;
                                        0638  1169 ;
                               00000638 1170          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                        0638  1171 ;
                                        0638  1172 ;       ********************
                                        0638  1173 ;       *                  *
                                        0638  1174 ;       *    PAS$RESET     *
                                        0638  1175 ;       *                  *
                                        0638  1176 ;       ********************
                                        0638  1177 ;
                                        0638  1178 ; Rewinds a file to the beginning of information and sets/clears the
                                        0638  1179 ; appropriate flags in the status word of the FSB. If the file is
                                        0638  1180 ; not already opened (the open bit is clear) then an existing file
                                        0638  1181 ; is opened by a call to PAS$OPEN. The buffer is NOT filled.
                                        0638  1182 ;
                                        0638  1183 ; Argument offsets
                                        0638  1184 ;
                                        0638  1185 ;       AP                              ; number of arguments (1)
                                00000004 0638  1186          FSB_DISP = 04                   ; FSB address
                                        0638  1187 ;
                                   001C 0638  1188          .ENTRY  PAS$RESET,^M<R2,R3,R4>
                              52   04 AC    D0  063A  1189          MOVL    FSB_DISP(AP),R2         ; R2 = address of FSB
                            53   18   52    C1  063E  1190          ADDL3   R2,#FSB$C_BLN,R3        ; R3 = address of RAB
                54   00000044 8F     53    C1  0642  1191          ADDL3   R3,#RAB$C_BLN,R4        ; R4 = address of FAB
                           2B 04 A2    05    E0  064A  1192          BBS     #FSB$V_OPEN,FSB$L_STA(R2),110$
                                        064F  1193                                  ; branch if open
                           14 04 A2    08    E1  064F  1194          BBC     #FSB$V_INT,FSB$L_STA(R2),105$
                                        0654  1195                                  ; internal file?
                                        0654  1196 ;
                                        0654  1197 ; Error if unopened internal file
                                        0654  1198 ;
```

```
                00018292 8F    DD   0654  1199          PUSHL   #RMS$_FNF                   ; pass 'file not found' error
             7E    0090 C2    9A   065A  1200          MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R2),-(SP)
                   0088 C2    DD   065F  1201          PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R2)
             FA7C  CF    03    FB   0663  1202          CALLS   #3,PAS$IOERROR
                                    0668  1203 ;
                                    0668  1204 ; Open an unopened but existing file
                                    0668  1205 ;
                                    0668  1206   105$:
                         00    DD   0668  1207          PUSHL   #PAS$C_NOCARR               ; carriage control -- not used
                      7E   20 A3    3C   066A  1208          MOVZWL  RAB$W_USZ(R3),-(SP)         ; record length
                         7E    7C   066E  1209          CLRD    -(SP)
                         7E    7C   0670  1210          CLRD    -(SP)
                            04 AC    DD   0672  1211          PUSHL   FSB_DISP(AP)
                      FD7B  CF    07    FB   0675  1212          CALLS   #7,PAS$OPEN
                                    067A  1213 ;
                                    067A  1214 ; Rewind the file if applicable
                                    067A  1215 ; Flush the buffer if necessary
                                    067A  1216 ;
                                    067A  1217   110$:
                   47 40 A4    00    E0   067A  1218          BBS     #DEV$V_REC,FAB$L_DEV(R4),120$
                                    067F  1219                                              ; can't rewind unit record  device
                   0D 04 A2    07    E1   067F  1220          BBC     #FSB$V_PUT,FSB$L_STA(R2),115$
                                    0684  1221                                              ; last operation write?
                      28 A3    62    D1   0684  1222          CMPL    (R2),RAB$L_RBF(R3)          ; buffer empty
                               07    13   0688  1223          BEQL    115$
                00000A12'EF    6C    FA   068A  1224          CALLG   (AP),PAS$WRITELN            ; flush buffer
                                    0691  1225   115$:
                      1E A3    00    90   0691  1226          MOVB    #RAB$C_SEQ,RAB$B_RAC(R3); make sure sequential
                                    0695  1227                                              ; (for binary files)
                                    0695  1228          $REWIND RAB=R3
                00018009 8F    50    D1   069E  1229          CMPL    R0,#RMS$_PENDING            ; check for completion
                               09    12   06A5  1230          BNEQ    118$
                                    06A7  1231          $WAIT   RAB=R3
                                    06B0  1232   118$:
                            13 50    E8   06B0  1233          BLBS    R0,120$                     ; branch if ok
                               50    DD   06B3  1234          PUSHL   R0
                         7E    8354 8F    3C   06B5  1235          MOVZWL  #^X8354,-(SP)
                            7E    34 A4    9A   06BA  1236          MOVZBL  FAB$B_FNS(R4),-(SP)
                               2C A4    DD   06BE  1237          PUSHL   FAB$L_FNA(R4)
                         FA1E  CF    04    FB   06C1  1238          CALLS   #4,PAS$IOERROR
                                    06C6  1239 ;
                                    06C6  1240 ; Reset status word
                                    06C6  1241 ;
                                    06C6  1242   120$:
                      04 A2    02    CA   06C6  1243          BICL    #FSB$M_EOF,FSB$L_STA(R2)
                      04 A2    04    CA   06CA  1244          BICL    #FSB$M_EOLN,FSB$L_STA(R2)
                      04 A2    08    C8   06CE  1245          BISL2   #FSB$M_GET,FSB$L_STA(R2); set read flag
04 A2              00000080 8F    CA   06D2  1246          BICL2   #FSB$M_PUT,FSB$L_STA(R2); clear write flag
                      04 A2    01    C8   06DA  1247          BISL    #FSB$M_RDLN,FSB$L_STA(R2)
                                    06DE  1248                                              ; set READLN flag
                                 04 06DE  1249          RET
                                    06DF  1250 ;
                                    06DF  1251 ;
                                 000006DF  1252          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                    06DF  1253 ;
                                    06DF  1254 ;         ********************
                                    06DF  1255 ;         *                  *
```

```
                        06DF   1255 ;+
                        06DF   1256 ;       *    PAS$REWRITE    *
                        06DF   1257 ;       *                   *
                        06DF   1258 ;       *********************
                        06DF   1259 ;
                        06DF   1260 ; Closes and deletes the existing file (if one exists) and creates a
                        06DF   1261 ; new file, setting/clearing the appropriate flags in the status word
                        06DF   1262 ; of the FSB
                        06DF   1263 ;
                        06DF   1264 ; Argument offsets
                        06DF   1265 ;
                        06DF   1266 ;       AP                                  ; number of arguments (1)
                00000004 06DF   1267        FSB_DISP = 04                       ; FSB address
                        06DF   1268 ;
                   003C 06DF   1269        .ENTRY  PAS$REWRITE,^M<R2,R3,R4,R5>
          52   04 AC   D0   06E1   1270        MOVL    FSB_DISP(AP),R2             ; R2 = address of FSB
        53   18   52   C1   06E5   1271        ADDL3   R2,#FSB$C_BLN,R3            ; R3 = address of RAB
  54 00000044 8F   53   C1   06E9   1272        ADDL3   R3,#RAB$C_BLN,R4            ; R4 = address of FAB
     15 04 A2   05   E0   06F1   1273        BBS     #FSB$V_OPEN,FSB$L_STA(R2),110$
                        06F6   1274                                            ; branch if existing file
                        06F6   1275 ;
                        06F6   1276 ; Not yet opened, create a new file
                        06F6   1277 ;
                   02   DD   06F6   1278        PUSHL   #PAS$C_LIST                 ; default carriage control
        7E   20 A3   3C   06F8   1279        MOVZWL  RAB$W_USZ(R3),-(SP)         ; record size
                7E   7C   06FC   1280        CLRD    -(SP)
                7E   7C   06FE   1281        CLRD    -(SP)
             04 AC   DD   0700   1282        PUSHL   FSB_DISP(AP)
     FDAE CF   07   FB   0703   1283        CALLS   #7,PAS$CREATE
              0096   31   0708   1284        BRW     180$
                        070B   1285 ;
                        070B   1286 ; Truncate the file from the 1st record on
                        070B   1287 ;
                        070B   1288   110$:
  03 40 A4   00   E1   070B   1289        BBC     #DEV$V_REC,FAB$L_DEV(R4),111$
              008E   31   0710   1290        BRW     180$                        ; skip for unit record device
     1E A3   00   90   0713   1291 111$:   MOVB    #RAB$C_SEQ,RAB$B_RAC(R3)    ; make sure sequential
                        0717   1292                                            ; (for binary files)
                        0717   1293        $REWIND RAB=R3
00018009 8F   50   D1   0720   1294        CMPL    R0,#RMS$_PENDING
                09   12   0727   1295        BNEQ    120$
                        0729   1296        $WAIT   RAB=R3
                        0732   1297   120$:
           03 50   E8   0732   1298        BLBS    R0,125$                     ; branch if ok
            009D   31   0735   1299        BRW     900$
                        0738   1300   125$:
                        0738   1301        $GET    RAB=R3                      ; get first for truncate
00018009 8F   50   D1   0741   1302        CMPL    R0,#RMS$_PENDING
                09   12   0748   1303        BNEQ    130$
                        074A   1304        $WAIT   RAB=R3
                        0753   1305   130$:
0001827A 8F   50   D1   0753   1306        CMPL    R0,#RMS$_EOF                ; check if empty file
                24   12   075A   1307        BNEQ    137$
                        075C   1308
                        075C   1309        $REWIND RAB=R3                      ; if empty, rewind and set TPT bit
00018009 8F   50   D1   0765   1310        CMPL    R0,#RMS$_PENDING
                09   12   076C   1311        BNEQ    135$
                        076E   1312        $WAIT   RAB=R3
```

```
                5B 50    E9  0777  1313 135$:    BLBC    R0,900$
          04 A3    02    C8  077A  1314          BISL2   #RAB$M_TPT,RAB$L_ROP(R3)
                   21    11  077E  1315          BRB     180$
                         0780  1316
                52 50    E9  0780  1317 137$:    BLBC    R0,900$                 ; branch if error
                         0783  1318              $TRUNCATE RAB=R3                ; truncate the (empty or non-empty) file
 00018009 8F     50    D1  078C  1319          CMPL    R0,#RMS$_PENDING
                   09    12  0793  1320          BNEQ    140$
                         0795  1321          $WAIT   RAB=R3
                         079E  1322    140$:
                34 50    E9  079E  1323          BLBC    R0,900$                 ; branch if error
                         07A1  1324 ;
                         07A1  1325 ; Set the FSB and record address
                         07A1  1326 ;
                         07A1  1327    180$:
       28 A3    24 A3    D0  07A1  1328          MOVL    RAB$L_UBF(R3),RAB$L_RBF(R3)
                         07A6  1329                                              ; set write buffer address
       22 A3    20 A3    B0  07A6  1330          MOVW    RAB$W_USZ(R3),RAB$W_RSZ(R3)
                         07AB  1331                                              ; set write buffer size
          04 A2    01    CA  07AB  1332          BICL2   #FSB$M_RDLN,FSB$L_STA(R2)
                         07AF  1333                                              ; clear RDLN flag
          04 A2    02    C8  07AF  1334          BISL    #FSB$M_EOF,FSB$L_STA(R2); set EOF
          04 A2    04    C8  07B3  1335          BISL    #FSB$M_EOLN,FSB$L_STA(R2)
                         07B7  1336                                              ; set EOLN
          04 A2    08    CA  07B7  1337          BICL2   #FSB$M_GET,FSB$L_STA(R2); clear read flag
    04 A2    00000080 8F    C8  07BB  1338          BISL2   #FSB$M_PUT,FSB$L_STA(R2); set write flag
                10 A2    D4  07C3  1339          CLRL    FSB$L_CNT(R2)           ; clear write record count
          62    28 A3    D0  07C6  1340          MOVL    RAB$L_RBF(R3),(R2)      ; initialize pointer to first
          50    20 A3    32  07CA  1341          CVTWL   RAB$W_USZ(R3),R0
    08 A2 50    28 A3    C1  07CE  1342          ADDL3   RAB$L_RBF(R3),R0,FSB$L_LST(R2)
                         07D4  1343                                              ; set last
                   04    07D4  1344          RET
                         07D5  1345 ;
                         07D5  1346 ; Error detected during rewrite
                         07D5  1347 ;
                         07D5  1348    900$:
                50    DD  07D5  1349          PUSHL   R0
    7E    8364 8F    3C  07D7  1350          MOVZWL  #^X8364,-(SP)
      7E    34 A4    9A  07DC  1351          MOVZBL  FAB$B_FNS(R4),-(SP)
            2C A4    DD  07E0  1352          PUSHL   FAB$L_FNA(R4)
    F8FC CF    04    FB  07E3  1353          CALLS   #4,PAS$IOERROR
                         07E8  1354 ;
                         07E8  1355 ;
                   0000 07E8  1356          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                         07E8  1357 ;
                         07E8  1358 ;       ********************
                         07E8  1359 ;       *                  *
                         07E8  1360 ;       *     PAS$FIND     *
                         07E8  1361 ;       *                  *
                         07E8  1362 ;       ********************
                         07E8  1363 ;
                         07E8  1364 ; Sets access by key field and sets key value for the next read.
                         07E8  1365 ; The access mode is returned to sequential at the end
                         07E8  1366 ; of the next read (PAS$GETBIN).
                         07E8  1367 ;
                         07E8  1368 ; Argument offsets
                         07E8  1369 ;       AP                                  ; number of arguments (2)
```

```
                    00000004  07E8  1370          FSB_DISP = 04                   ; FSB address
                    00000008  07E8  1371          REC_DISP = 08                   ; relative record number (by value)
                              07E8  1372 ;
                        03C0  07E8  1373          .ENTRY  PAS$FIND,^M<R6,R7,R8,R9>
           56   04 AC    D0  07EA  1374          MOVL    FSB_DISP(AP),R6         ; R6 = address of FSB
      57   56    18    C1  07EE  1375          ADDL3   #FSB$C_BLN,R6,R7        ; R7 = address of RAB
58    57   00000044 8F   C1  07F2  1376          ADDL3   #RAB$C_BLN,R7,R8        ; R8 = address of FAB
                              07FA  1377 ;
                              07FA  1378 ; Check if RESET called
                              07FA  1379 ;
      42 04 A6    03    E1  07FA  1380          BBC     #FSB$V_GET,FSB$L_STA(R6),930$
                              07FF  1381                                          ; read access?
                              07FF  1382 ;
                              07FF  1383 ; Check for valid file type and set access to key
                              07FF  1384 ;           (1) must be binary file
                              07FF  1385 ;           (2) sequential file with fixed length records
                              07FF  1386 ;
      2C 04 A6    04    E0  07FF  1387          BBS     #FSB$V_TXT,FSB$L_STA(R6),910$
                              0804  1388                                          ; must be binary file
         1D A8    00    91  0804  1389          CMPB    #FAB$C_SEQ,FAB$B_ORG(R8); sequential file
                        26  12  0808  1390          BNEQ    910$
         1F A8    01    91  080A  1391          CMPB    #FAB$C_FIX,FAB$B_RFM(R8); fixed length records
                        20  12  080E  1392          BNEQ    910$
         1E A7    01    90  0810  1393          MOVB    #RAB$C_KEY,RAB$B_RAC(R7); set key access
         34 A7    04    90  0814  1394          MOVB    #4,RAB$B_KSZ(R7)        ; set key size
         04 A6    02    CA  0818  1395          bicl2   #fsb$m_eof,fsb$l_sta(r6); clear eof flag
      30 A7    14 A6    DE  081C  1396          MOVAL   FSB$L_REC(R6),RAB$L_KBF(R7)
                              0821  1397                                          ; set key buffer address
      14 A6    08 AC    D0  0821  1398          MOVL    REC_DISP(AP),FSB$L_REC(R6)
                              0826  1399                                          ; set key
      04 04 A6    03    E1  0826  1400          BBC     #FSB$V_GET,FSB$L_STA(R6),115$
         04 A6    01    C8  082B  1401          BISL2   #FSB$M_RDLN,FSB$L_STA(R6)
                              082F  1402                                          ; set RDLN flag
                              082F  1403  115$:
                        04    082F  1404          RET
                              0830  1405 ;
                              0830  1406 ; Error, file not of appropriate type
                              0830  1407 ;
                              0830  1408  910$:
   7E    83C4 8F    3C    0830  1409          MOVZWL  #^X83C4,-(SP)
      7E    34 A8    9A  0835  1410          MOVZBL  FAB$B_FNS(R8),-(SP)
            2C A8    DD  0839  1411          PUSHL   FAB$L_FNA(R8)
F8A3 CF       03    FB  083C  1412          CALLS   #3,PA$$IOERROR
                              0841  1413 ;
                              0841  1414 ; Error, file not reset or rewritten
                              0841  1415 ;
                              0841  1416  930$:
   7E    83D4 8F    3C    0841  1417          MOVZWL  #^X83D4,-(SP)
      7E    34 A8    9A  0846  1418          MOVZBL  FAB$B_FNS(R8),-(SP)
            2C A8    DD  084A  1419          PUSHL   FAB$L_FNA(R8)
F892 CF       03    FB  084D  1420          CALLS   #3,PA$$IOERROR
                              0852  1421 ;
                              0852  1422 ;
                        00000852  1423          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                              0852  1424 ;
                              0852  1425 ;       ********************
                              0852  1426 ;       *                  *
```

```
                                0852  1427 ;       *    PAS$CLOSE     *
                                0852  1428 ;       *  PAS$CLOSEINOUT  *
                                0852  1429 ;       *                  *
                                0852  1430 ;       ********************
                                0852  1431 ;
                                0852  1432 ; Closes N files (N > 0). The pointer is set to nil and the open
                                0852  1433 ; flag is cleared. Any error in closing the file causes a runtime error.
                                0852  1434 ;
                                0852  1435 ; Argument offsets
                                0852  1436 ;
                                0852  1437 ;       AP                              ; number of arguments (n)
                                0852  1438 ;       AP+4                            ; FSB address of file #1
                                0852  1439 ;         .
                                0852  1440 ;         .
                                0852  1441 ;         .
                                0852  1442 ;       AP+N                            ; FSB address of file #n
                                0852  1443 ;
                        00FC    0852  1444          .ENTRY  PAS$CLOSEINOUT,^M<R2,R3,R4,R5,R6,R7>
                57      01      D0    0854  1445          MOVL    #1,R7                   ; set flag for CLOSEINOUT
                        04      11    0857  1446          BRB     CLOSEENT
                                0859  1447 ;
                        00FC    0859  1448          .ENTRY  PAS$CLOSE,^M<R2,R3,R4,R5,R6,R7>
                57      D4      085B  1449          CLRL    R7                      ; set flag for CLOSE
                                085D  1450  CLOSEENT:
                5E      7E      DE    085D  1451          MOVAL   -(SP),SP                ; make room for parameter
                                0860  1452                                          ; to LIB$FREE_VM
                52      6C      D0    0860  1453          MOVL    (AP),R2                 ; R2 = number of arguments
          53    04      5C      C1    0863  1454          ADDL3   AP,#4,R3                ; R3 = address of 1st FSB address
                                0867  1455    10$:                                  ; loop until all files closed
                        54      63    D0    0867  1456          MOVL    (R3),R4                 ; R4 = address of FSB
                55      18      54    C1    086A  1457          ADDL3   R4,#FSB$C_BLN,R5        ; R5 = address of RAB
56        55    00000044      8F    C1    086E  1458          ADDL3   #RAB$C_BLN,R5,R6        ; R6 = address of FAB
             77 04  A4          05    E1    0876  1459          BBC     #FSB$V_OPEN, FSB$L_STA(R4),120$
                                087B  1460                                          ; branch if file already closed
                0A 57           00    E0    087B  1461          BBS     #0,R7,15$               ; branch if call from CLOSEINOUT
             79 04  A4          0A    E0    087F  1462          BBS     #FSB$V_OUTPUT,FSB$L_STA(R4),130$
                                0884  1463                                          ; branch if file OUTPUT or INPUT
             74 04  A4          0C    E0    0884  1464          BBS     #FSB$V_INPUT,FSB$L STA(R4),130$
                                0889  1465  15$:
             14 04  A4          03    E0    0889  1466          BBS     #FSB$V_GET,FSB$L_STA(R4),110$
                                088E  1467                                          ; branch if get access
             0F 04  A4          04    E1    088E  1468          BBC     #FSB$V_TXT,FSB$L_STA(R4),110$
                                0893  1469                                          ; branch if not textfile
                64      28 A5      D1    0893  1470          CMPL    RAB$L_RBF(R5),(R4)
                                09      13    0897  1471          BEQL    110$
                                54      DD    0899  1472          PUSHL   R4
          00000A12'EF           01      FB    089B  1473          CALLS   #1,PAS$WRITELN
                                08A2  1474  110$:
             0D 04  A4          1E    E1    08A2  1475          BBC     #FSB$V_DELZ,FSB$L_STA(R4),105$
                                08A7  1476                                          ; branch if not delete
                        10 A4      D5    08A7  1477          TSTL    FSB$L_CNT(R4)           ; check line count
                                08      12    08AA  1478          BNEQ    105$
       04 A6    00008000      8F    C8    08AC  1479          BISL2   #FAB$M_DLT,FAB$L_FOP(R6); set delete flag
                                08B4  1480  105$:
                                08B4  1481          $CLOSE  FAB=R6                  ; close the file
                        09 50      E8    08BD  1482          BLBS    R0,115$                 ; branch if ok
          0001C032 8F           50      D1    08C0  1483          CMPL    R0,#RMS$_MKD            ; check for no deletion error
```

```
                    45   12  08C7  1484          BNEQ    135$                    ; branch if that's not it
                                08C9  1485  115$:
                 22 57   E8  08C9  1486          BLBS    R7,117$                 ; branch if file INPUT or OUTPUT
                                08CC  1487
           6E    34 A6   9A  08CC  1488          MOVZBL  FAB$B_FNS(R6),(SP)      ; deallocate file name string
                 2C A6   DF  08D0  1489          PUSHAL  FAB$L_FNA(R6)
                 04 AE   DF  08D3  1490          PUSHAL  4(SP)
00000000'GF      02   FB  08D6  1491          CALLS   #2,G^LIB$FREE_VM        ; ignore errors
                                08DD  1492
           6E    20 A5   3C  08DD  1493          MOVZWL  RAB$W_USZ(R5),(SP)      ; deallocate file buffer
                 24 A5   DF  08E1  1494          PUSHAL  RAB$L_UBF(R5)
                 04 AE   DF  08E4  1495          PUSHAL  4(SP)
00000000'GF      02   FB  08E7  1496          CALLS   #2,G^LIB$FREE_VM        ; ignore errors
                                08EE  1497  117$:
        04 A4    20   CA  08EE  1498          BICL2   #FSB$M_OPEN,FSB$L_STA(R4)
                                08F2  1499                                  ; clear OPEN flag
                                08F2  1500  120$:
              53    04   C0  08F2  1501          ADDL2   #4,R3
                    52   D7  08F5  1502          DECL    R2                  ; loop if more files
                    03   15  08F7  1503          BLEQ    125$
                 FF6B    31  08F9  1504          BRW     10$
                         04  08FC  1505  125$:   RET
                             08FD  1506 ;
                             08FD  1507 ;
                             08FD  1508 ; Error: file OUTPUT cannot be closed
                             08FD  1509 ;
                             08FD  1510 130$:
        7E    83E4 8F   3C  08FD  1511          MOVZWL  #^X83E4,-(SP)           ; PASCAL error code
          7E    34 A6   9A  0902  1512          MOVZBL  FAB$B_FNS(R6),-(SP)     ; file name string length
                2C A6   DD  0906  1513          PUSHL   FAB$L_FNA(R6)           ; file name
     F7D6 CF    03   FB  0909  1514          CALLS   #3,PAS$IOERROR
                             090E  1515  135$:
                    50   DD  090E  1516          PUSHL   R0
     7E    83B4 8F   3C  0910  1517          MOVZWL  #^X83B4,-(SP)
       7E    34 A6   9A  0915  1518          MOVZBL  FAB$B_FNS(R6),-(SP)
             2C A6   DD  0919  1519          PUSHL   FAB$L_FNA(R6)
     F7C3 CF    04   FB  091C  1520          CALLS   #4,PAS$IOERROR
                             0921  1521 ;
                             0921  1522 ;
                             0921  1523 ;
                        00000921  1524          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                             0921  1525 ;
                             0921  1526 ;       ********************
                             0921  1527 ;       *                  *
                             0921  1528 ;       *      PAS$EOF     *
                             0921  1529 ;       *                  *
                             0921  1530 ;       ********************
                             0921  1531 ;
                             0921  1532 ; Checks for end-of-file. If the RDLN bit is set the next record
                             0921  1533 ; is retrieved.
                             0921  1534 ;
                             0921  1535 ; Argument offsets
                             0921  1536 ;
                             0921  1537 ;       AP                              ; number of arguments (1)
                 00000004    0921  1538          FSB_DISP = 04                   ; FSB address
                             0921  1539 ;
                     0040    0921  1540          .ENTRY  PAS$EOF,^M<R6>          ; end of file
```

```
                56   04 AC    D0  0923  1541          MOVL    FSB_DISP(AP),R6         ; R6 = address of variable
      07 04 A6      00    E1  0927  1542          BBC     #FSB$V_RDLN,FSB$L_STA(R6),10$
                                0920  1543                                          ; need next record?
 0000095D'EF        6C    FA  092C  1544          CALLG   (AP),PAS$ACTUALGET      ; yes
                                0933  1545    10$:
                50   00    D0  0933  1546          MOVL    #PAS$C_FALSE,R0         ; set function return to FALSE
      03 04 A6      01    E1  0936  1547          BBC     #FSB$V_EOF,FSB$L_STA(R6),99$
                                093B  1548                                          ; branch if not EOF
                50   01    D0  093B  1549          MOVL    #PAS$C_TRUE,R0          ; set function return to TRUE
                                093E  1550    99$:
                      04    093E  1551          RET
                                093F  1552 ;
                                093F  1553 ;
                          0000093F  1554          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                093F  1555 ;
                                093F  1556 ;       ********************
                                093F  1557 ;       *                  *
                                093F  1558 ;       *     PAS$EOLN     *
                                093F  1559 ;       *                  *
                                093F  1560 ;       ********************
                                093F  1561 ;
                                093F  1562 ; Checks for end-of-line. If the RDLN bit is set the next record
                                093F  1563 ; is retrieved.
                                093F  1564 ;
                                093F  1565 ; Argument offsets
                                093F  1566 ;
                                093F  1567 ;       AP                              ; number of arguments (1)
                    00000004    093F  1568          FSB_DISP = 04                   ; FSB address
                                093F  1569 ;
                        0040    093F  1570          .ENTRY  PAS$EOLN,^MR6           ; end of line
                56   04 AC      0941  1571          MOVL    FSB_DISP(AP),R6         ; R6 = address of pointer
      07 04 A6      00    E1  0945  1572          BBC     #FSB$V_RDLN,FSB$L_STA(R6),110$
                                094A  1573                                          ; need next record
 0000095D'EF        6C    FA  094A  1574          CALLG   (AP),PAS$ACTUALGET
                                0951  1575    110$:
                50   00    D0  0951  1576          MOVL    #PAS$C_FALSE,R0         ; set function return to FALSE
      03 04 A6      02    E1  0954  1577          BBC     #FSB$V_EOLN,FSB$L_STA(R6),199$
                                0959  1578                                          ; branch if not eoln
                50   01    D0  0959  1579          MOVL    #PAS$C_TRUE,R0          ; set function return to TRUE
                                095C  1580    199$:
                      04    095C  1581          RET
                                095D  1582 ;
                                095D  1583 ;
                          0000095D  1584          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                                095D  1585 ;
                                095D  1586 ;       ********************
                                095D  1587 ;       *                  *
                                095D  1588 ;       *  PAS$ACTUALGET   *
                                095D  1589 ;       *                  *
                                095D  1590 ;       ********************
                                095D  1591 ;
                                095D  1592 ; Does the actual file access for text and binary files. PAS$ACTUALGET
                                095D  1593 ; is called from the compiler if the RDLN flag is set, from other
                                095D  1594 ; input routines in the I-O interface, or from PAS$EOF and PAS$EOLN
                                095D  1595 ; if the RDLN flag is set. The access codes should be checked before
                                095D  1596 ; calling this procedure. The RDLN flag being set implies read access
                                095D  1597 ; is permitted.
```

```
                                095D  1598 ;
                                095D  1599 ; Argument offsets
                                095D  1600 ;
                                095D  1601 ;       AP                                      ; number of arguments
                   00000004     095D  1602         FSB_DISP = 04                           ; FSB address
                                095D  1603 ;
                        01C0    095D  1604         .ENTRY  PAS$ACTUALGET,^M<R6,R7,R8>
                56   04 AC   D0 095F  1605         MOVL    FSB_DISP(AP),R6                 ; R6 = address of FSB
                04 A6   01   CA 0963  1606         BICL2   #FSB$M_RDLN,FSB$L_STA(R6)
                                0967  1607                                                 ; clear RDLN flag
             57   18   56   C1 0967  1608         ADDL3   R6,#FSB$C_BLN,R7                ; R7 = address of RAB
            40 04 A6   09   E1 096B  1609         BBC     #FSB$V_PRMT,FSB$L_STA(R6),10$
                                0970  1610                                                 ; branch if not prompting
                                0970  1611 ;
                                0970  1612 ; Prompting is performed on INPUT/OUTPUT.  Check if any characters in
                                0970  1613 ; OUTPUT buffer.
                                0970  1614 ;
                58   14 A6   D0 0970  1615         MOVL    FSB$L_PFSB(R6),R8               ; R8 = OUTPUT FSB address
                40 A8   68   D1 0974  1616         CMPL    (R8),FSB$C_BLN+RAB$L_RBF(R8)
                                0978  1617                                                 ; any characters in buffer?
                        36   13 0978  1618         BEQL    10$                             ; no--continue
                                097A  1619 ;
                                097A  1620 ; Characters are present in OUTPUT buffer.  Write these characters as a
                                097A  1621 ; prompt for the current GET.
                                097A  1622 ;
      04 A8   00004000 8F   C8 097A  1623         BISL2   #FSB$M_WRITPRMT,FSB$L_STA(R8)           ; set flag to call writeln
            06 04 A8   0D   E0 0982  1624         BBS     #FSB$V_PROMPT,FSB$L_STA(R8),1$          ; was a prompt emmitted
                                0987  1625                                                       ; on the previous line?
               44 B8   01   B0 0987  1626         MOVW    #PRN_LF,@FSB$C_BLN+RAB$L_RHB(R8)        ; no, use <LF> <prompt>
                        04   11 098B  1627         BRB     2$
               44 B8   00   B0 098D  1628 1$:     MOVW    #PRN_NULL,@FSB$C_BLN+RAB$L_RHB(R8)
                                0991  1629                                                       ; set null carriage control
                        58   DD 0991  1630 2$:     PUSHL   R8                              ; argument is OUTPUT FSB address
          00000A12'EF   01   FB 0993  1631         CALLS   #1,PAS$WRITELN                  ; write the prompt line
            44 B8   8D01 8F   B0 099A  1632         MOVW    #PRN_CRLF,@FSB$C_BLN+RAB$L_RHB(R8)
                                09A0  1633                                                       ; reset normal carriage control
      04 A8   00004000 8F   CA 09A0  1634         BICL2   #FSB$M_WRITPRMT,FSB$L_STA(R8)   ; clear the flag which affects
                                09A8  1635                                                       ; carriage control of a call to WRIT
      04 A8   00002000 8F   C8 09A8  1636         BISL2   #FSB$M_PROMPT,FSB$L_STA(R8)     ; set the prompt flag
                                09B0  1637 10$:     $GET    RAB=R7
          00018009 8F   50   D1 09B9  1638         CMPL    R0,#RMS$_PENDING
                        09   12 09C0  1639         BNEQ    105$
                                09C2  1640         $WAIT   RAB=R7
                                09CB  1641 105$:
          0001827A 8F   50   D1 09CB  1642         CMPL    R0,#RMS$_EOF                    ; check for eof
                        06   12 09D2  1643         BNEQ    110$
                04 A6   02   C8 09D4  1644         BISL2   #FSB$M_EOF,FSB$L_STA(R6); set EOF flag
                        11   11 09D8  1645         BRB     111$
                                09DA  1646 110$:
                     0E 50   E8 09DA  1647         BLBS    R0,111$                         ; branch if ok
                        50   DD 09DD  1648         PUSHL   R0
                   7E   78 A7   9A 09DF  1649         MOVZBL  <RAB$C_BLN+FAB$B_FNS>(R7),-(SP)
                        70 A7   DD 09E3  1650         PUSHL   <RAB$C_BLN+FAB$L_FNA>(R7)
                F6F9 CF   03   FB 09E6  1651         CALLS   #3,PAS$IOERROR
                                09EB  1652 111$:
                   66   24 A7   D0 09EB  1653         MOVL    RAB$L_UBF(R7),(R6)              ; set pointer to first
                15 04 A6   04   E1 09EF  1654         BBC     #FSB$V_TXT,FSB$L_STA(R6),199$
```

```
                        09F4  1655                                          ; done if binary file
                        09F4  1656 ;
                        09F4  1657 ; Set textfile parameters
                        09F4  1658 ;
          51    22 A7   32  09F4  1659          CVTWL   RAB$W_RSZ(R7),R1
    08 A6     51    66   C1  09F8  1660          ADDL3   (R6),R1,FSB$L_LST(R6)   ; set last to last+1
        08 B6     20   90  09FD  1661          MOVB    #SPACE,@FSB$L_LST(R6)   ; store EOLN blank
                  51   D5  0A01  1662          TSTL    R1                      ; check for EOLN
                  04   12  0A03  1663          BNEQ    199$
        04 A6     04   C8  0A05  1664          BISL2   #FSB$M_EOLN,FSB$L_STA(R6)
                           0A09  1665                                          ; set EOLN flag
                           0A09  1666   199$:
04 A6   00000800 8F   C8  0A09  1667          BISL2   #FSB$M_ACTIN,FSB$L_STA(R6)
                           0A11  1668                                          ; set actual input flag
                        04 0A11  1669          RET
                           0A12  1670 ;
                           0A12  1671 ;
                        00000A12  1672          .PSECT  _PAS$CODE,      PIC,EXE,SHR,NOWRT
                           0A12  1673 ;
                           0A12  1674 ;        ********************
                           0A12  1675 ;        *                  *
                           0A12  1676 ;        *   PAS$WRITELN    *
                           0A12  1677 ;        *                  *
                           0A12  1678 ;        ********************
                           0A12  1679 ;
                           0A12  1680 ; Writes a record (line) to the file.
                           0A12  1681 ;
                           0A12  1682 ; Argument offsets
                           0A12  1683 ;
                           0A12  1684 ;        AP                              ; number of arguments (1)
                  00000004 0A12  1685          FSB_DISP = 04                   ; FSB address
                           0A12  1686 ;
                      000C 0A12  1687          .ENTRY  PAS$WRITELN,^M<R2,R3>
      F628 CF     6C   FA  0A14  1688          CALLG   (AP),PAS$WRITEOK
        52    04 AC   D0  0A19  1689          MOVL    FSB_DISP(AP),R2         ; R2 = address of FSB
     53    18   52   C1  0A1D  1690          ADDL3   R2,#FSB$C_BLN,R3        ; R3 = address of RAB
50    62    28 A3   C3  0A21  1691          SUBL3   RAB$L_RBF(R3),(R2),R0
        22 A3   50   F7  0A26  1692          CVTLW   R0,RAB$W_RSZ(R3)
     23 04 A2   0E   E0  0A2A  1693          BBS     #FSB$V_WRITPRMT,FSB$L_STA(R2),10$       ; do a WRITELN: <text> <CR>
                           0A2F  1694                                          ; if following a prompt
                           0A2F  1695
     1E 04 A2   0D   E1  0A2F  1696          BBC     #FSB$V_PROMPT,FSB$L_STA(R2),10$
   44 B2    8D00 8F   B0  0A34  1697          MOVW    #PRN_CR,@FSB$C_BLN+RAB$L_RHB(R2)
                        0A3A  1698          $PUT    RAB=R3
        04 A3   02   CA  0A43  1699          BICL2   #RAB$M_TPT,RAB$L_ROP(R3)        ; clear TPT bit
              2F 50   E9  0A47  1700          BLBC    R0,910$
   44 B2    8D01 8F   B0  0A4A  1701          MOVW    #PRN_CRLF,@FSB$C_BLN+RAB$L_RHB(R2)
                  10   11  0A50  1702          BRB     105$
                           0A52  1703 10$:     $PUT    RAB=R3
        04 A3   02   CA  0A5B  1704          BICL2   #RAB$M_TPT,RAB$L_ROP(R3)        ; clear TPT bit
              17 50   E9  0A5F  1705          BLBC    R0,910$                 ; branch if error
                           0A62  1706 105$:
04 A2   00002000 8F   CA  0A62  1707          BICL2   #FSB$M_PROMPT,FSB$L_STA(R2)      ; clear the prompt flag
              10 A2   D6  0A6A  1708          INCL    FSB$L_CNT(R2)           ; increment line count
     0C A2    10 A2   D1  0A6D  1709          CMPL    FSB$L_CNT(R2),FSB$L_LIM(R2)
                           0A72  1710                                          ; check linelimit
                  13   14  0A72  1711          BGTR    920$                    ; abort if exceeded
```

```
                62    28 A3    D0  0A74  1712          MOVL    RAB$L_RBF(R3),(R2)      ; set pointer to first element
                                04  0A78  1713          RET
                                    0A79  1714 ;
                                    0A79  1715 ; Write error
                                    0A79  1716 ;
                                    0A79  1717 910$:
                      50    DD    0A79  1718          PUSHL   R0
                   7E   78 A3    9A  0A7B  1719          MOVZBL  <RAB$C_BLN+FAB$B_FNS>(R3),-(SP)
                      70 A3    DD  0A7F  1720          PUSHL   <RAB$C_BLN+FAB$L_FNA>(R3)
              F65D CF    03    FB  0A82  1721          CALLS   #3,PAS$IOERROR
                                    0A87  1722 ;
                                    0A87  1723 ; Error, linelimit exceeded
                                    0A87  1724 ;
                                    0A87  1725 920$:
                      0C A2    DD  0A87  1726          PUSHL   FSB$L_LIM(R2)           ; pass linelimit
                7E  8374 8F    3C  0A8A  1727          MOVZWL  #^X8374,-(SP)
                   7E   78 A3    9A  0A8F  1728          MOVZBL  <RAB$C_BLN+FAB$B_FNS>(R3),-(SP)
                      70 A3    DD  0A93  1729          PUSHL   <RAB$C_BLN+FAB$L_FNA>(R3)
              F649 CF    04    FB  0A96  1730          CALLS   #4,PAS$IOERROR
                                    0A9B  1731 ;
                                    0A9B  1732 ;
                                    0A9B  1733 ;
                                    0A9B  1734          .END
```

```
$$.TMP1                 = 00000001
$$.TMP2                 = 00000053
$$ARGS                  = 00000006
$$T1                    = 0000001C
AA_SMALL                = 00000061
ACC_DISP                = 00000010
CAR_DISP                = 0000001C
CBINIT_R4                 000005CD R      02
CLOSEENT                  0000085D R      02
DEV$V_REC               = 00000000
DEV$V_TRM               = 00000002
DFLEN                   = 00000004
DFNAM                     00000378 R      02
EXT_DISP                = 00000010
FAB$B_BID               = 00000000
FAB$B_BLN               = 00000001
FAB$B_DNS               = 00000035
FAB$B_FAC               = 00000016
FAB$B_FNS               = 00000034
FAB$B_FSZ               = 0000003F
FAB$B_ORG               = 0000001D
FAB$B_RAT               = 0000001E
FAB$B_RFM               = 0000001F
FAB$C_BID               = 00000003
FAB$C_BLN               = 00000050
FAB$C_FIX               = 00000001
FAB$C_SEQ               = 00000000
FAB$C_VAR               = 00000002
FAB$C_VFC               = 00000003
FAB$L_DEV               = 00000040
FAB$L_DNA               = 00000030
FAB$L_FNA               = 0000002C
FAB$L_FOP               = 00000004
FAB$L_NAM               = 00000028
FAB$M_DLT               = 00008000
FAB$M_GET               = 00000002
FAB$M_PUT               = 00000001
FAB$M_TRN               = 00000010
FAB$V_CR                = 00000001
FAB$V_FTN               = 00000000
FAB$V_PRN               = 00000002
FAB$V_TMD               = 00000004
FAB$W_MRS               = 00000036
FMT_DISP                = 00000014
FNL_DISP                = 00000008
FNM_DISP                = 00000004
FSB$B_CC                = 00000006
FSB$C_BLN               = 00000018
FSB$L_CNT               = 00000010
FSB$L_LIM               = 0000000C
FSB$L_LST               = 00000008
FSB$L_PFSB              = 00000014
FSB$L_REC               = 00000014
FSB$L_STA               = 00000004
FSB$M_ACTIN             = 00000800
FSB$M_DELZ              = 40000000
FSB$M_EOF               = 00000002
FSB$M_EOLN              = 00000004
FSB$M_GET               = 00000008
FSB$M_INC               = 80000C00
FSB$M_INPUT             = 00001000
FSB$M_OPEN              = 00000020
FSB$M_OUTPUT            = 00000400
FSB$M_PROMPT            = 00002000
FSB$M_PUT               = 00000080
FSB$M_RDLN              = 00000001
FSB$M_WRITPRMT          = 00004000
FSB$V_DELZ              = 0000001E
FSB$V_DIR               = 00000006
FSB$V_EOF               = 00000001
FSB$V_EOLN              = 00000002
FSB$V_GET               = 00000003
FSB$V_INC               = 0000001F
FSB$V_INPUT             = 0000000C
FSB$V_INT               = 00000008
FSB$V_OPEN              = 00000005
FSB$V_OUTPUT            = 0000000A
FSB$V_PRMT              = 00000009
FSB$V_PROMPT            = 0000000D
FSB$V_PUT               = 00000007
FSB$V_RDLN              = 00000000
FSB$V_TXT               = 00000004
FSB$V_WRITPRMT          = 0000000E
FSB_DISP                = 00000004
INPUTLEN                = 00000009
INP_DISP                = 00000008
LEN_DISP                = 0000000C
LIB$FREE_VM               ********    X   00
LIB$GET_VM                ********    X   00
LIB$STOP                  ********    X   00
MRL_DISP                = 00000018
NAM$B_BID               = 00000000
NAM$B_BLN               = 00000001
NAM$C_BID               = 00000002
NAM$C_BLN_V2            = 00000038
NAM_DISP                = 00000008
OUTDESCR                  0000027E R      02
OUTPUTLEN               = 0000000A
PAS$ACTUALGET             0000095D RG     02
PAS$BLANK_R3              00000182 RG     02
PAS$BUFFEROVER            00000062 RG     02
PAS$CLOSE                 00000859 RG     02
PAS$CLOSEINOUT            00000852 RG     02
PAS$CREATE                000004B6 RG     02
PAS$C_DFLTLINLI           ********    X   00
PAS$C_DFLTRECSI         = 00000101
PAS$C_FALSE             = 00000000
PAS$C_LIST              = 00000002
PAS$C_NOCARR            = 00000000
PAS$C_PRN               = 00000003
PAS$C_TRUE              = 00000001
PAS$EOF                   00000921 RG     02
PAS$EOLN                  0000093F RG     02
PAS$FILENAME              00000081 RG     02
```

```
PAS$FIND                        000007E8 RG     02
PAS$INITFILES                   0000037C RG     02
PAS$INPUT                       000001CA RG     02
PAS$IOERROR                     000000E4 RG     02
PAS$OPEN                        000003F5 RG     02
PAS$OUTPUT                      00000286 RG     02
PAS$READOK                      00000000 RG     02
PAS$RESET                       00000638 RG     02
PAS$REWRITE                     000006DF RG     02
PAS$STATUSUPDAT                 000000B4 RG     02
PAS$WRITELN                     00000A12 RG     02
PAS$WRITEOK                     00000041 RG     02
PAS$_ERRACCFIL                  ********   X    00
PASDESCR                        000001C2 R      02
PASINPUT                        000001B0 R      02
PASOUTPUT                       00000274 R      02
PRN_CR                        = 00008D00
PRN_CRLF                      = 00008D01
PRN_LF                        = 00000001
PRN_NULL                      = 00000000
RAB$B_BID                     = 00000000
RAB$B_BLN                     = 00000001
RAB$B_KSZ                     = 00000034
RAB$B_RAC                     = 0000001E
RAB$C_BID                     = 00000001
RAB$C_BLN                     = 00000044
RAB$C_KEY                     = 00000001
RAB$C_SEQ                     = 00000000
RAB$L_FAB                     = 0000003C
RAB$L_KBF                     = 00000030
RAB$L_RSF                     = 00000028
RAB$L_RHB                     = 0000002C
RAB$L_ROP                     = 00000004
RAB$L_UBF                     = 00000024
RAB$M_TPT                     = 00000002
RAB$W_RSZ                     = 00000022
RAB$W_USZ                     = 00000020
REC_DISP                      = 00000008
RMS$_EOF                      = 0001827A
RMS$_FNF                      = 00018292
RMS$_MKD                      = 0001C032
RMS$_PENDING                  = 00018009
RMS$_PRV                      = 0001829A
SPACE                         = 00000020
SS$_NOTRAN                    = 00000629
STR_DISP                      = 00000008
SYS$CLOSE                       ********   G    02
SYS$CONNECT                     ********   G    02
SYS$CREATE                      ********   G    02
SYS$GET                         ********   G    02
SYS$OPEN                        ********   G    02
SYS$PUT                         ********   G    02
SYS$REWIND                      ********   G    02
SYS$TRNLOG                      ********   G    02
SYS$TRUNCATE                    ********   G    02
SYS$WAIT                        ********   G    02
SYSINPUT                        000001B9 R      02
SYSOUTPUT                       0000026A R      02
TAB                           = 00000009
TRNLOG$_ACMODE                = 00000014
TRNLOG$_DSBMSK                = 00000018
TRNLOG$_LOGNAM                = 00000004
TRNLOG$_NARGS                 = 00000006
TRNLOG$_RSLBUF                = 0000000C
TRNLOG$_RSLLEN                = 00000008
TRNLOG$_TABLE                 = 00000010
TXT_DISP                      = 00000014
ZZ_SMALL                      = 0000007A
```

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	00000A9B (2715.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	34	00:00:00.09	00:00:00.61
Command processing	139	00:00:00.46	00:00:03.78
Pass 1	418	00:00:16.71	00:00:38.13
Symbol table sort	0	00:00:01.86	00:00:02.64
Pass 2	291	00:00:04.96	00:00:12.75
Symbol table output	22	00:00:00.17	00:00:00.25
Psect synopsis output	3	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	910	00:00:24.28	00:00:58.19

```
The working set limit was 1800 pages.
100189 bytes (196 pages) of virtual memory were used to buffer the intermediate code.
There were 80 pages of symbol table space allocated to hold 1383 non-local and 88 local symbols.
1734 source lines were read in Pass 1, producing 70 object records in Pass 2.
30 pages of virtual memory were used to define 28 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SYSLIB]STARLET.MLB;2	25

```
1470 GETS were required to define 25 macros.

There were no errors, warnings or information messages.

MACRO/DISABLE=TRACE/LIS=LIS$:PASIO1/OBJ=OBJ$:PASIO1 MSRC$:PASIO1/UPDATE=(ENH$:PASIO1)
```

PASDEF
LIS

PASI02
LIS

STATUS
LIS

UNKNOWN
LIS

SHUTDOWN
LIS

SHARE
LIS

PASI01
LIS

PASI03
LIS

PASCAL

PASCLINE
LIS

TIMESTAMP
LIS