L 11

```
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSSS     CCCCCCCCCCCC     AAAAAAAAA     LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSSS     CCCCCCCCCCCC     AAAAAAAAA     LLL
PPPPPPPPPPPP        AAAAAAAAA        SSSSSSSSSSSS     CCCCCCCCCCCC     AAAAAAAAA     LLL
PPP         PPP  AAA         AAA  SSS              CCC              AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS              CCC              AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS              CCC              AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS              CCC              AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS              CCC              AAA         AAA  LLL
PPP         PPP  AAA         AAA  SSS              CCC              AAA         AAA  LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS     CCC              AAA         AAA  LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS     CCC              AAA         AAA  LLL
PPPPPPPPPPPP     AAA         AAA     SSSSSSSSS     CCC              AAA         AAA  LLL
PPP              AAAAAAAAAAAAAAA              SSS  CCC              AAAAAAAAAAAAAAA  LLL
PPP              AAAAAAAAAAAAAAA              SSS  CCC              AAAAAAAAAAAAAAA  LLL
PPP              AAAAAAAAAAAAAAA              SSS  CCC              AAAAAAAAAAAAAAA  LLL
PPP              AAA         AAA              SSS  CCC              AAA         AAA  LLL
PPP              AAA         AAA              SSS  CCC              AAA         AAA  LLL
PPP              AAA         AAA              SSS  CCC              AAA         AAA  LLL
PPP              AAA         AAA  SSSSSSSSSSSS        CCCCCCCCCCCC  AAA         AAA  LLLLLLLLLLLLLLL
PPP              AAA         AAA  SSSSSSSSSSSS        CCCCCCCCCCCC  AAA         AAA  LLLLLLLLLLLLLLL
PPP              AAA         AAA  SSSSSSSSSSSS        CCCCCCCCCCCC  AAA         AAA  LLLLLLLLLLLLLLL
```

FILEID**PASIO2

D 15

```
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      OOOOOO      222222
PPPPPPPP      AAAAAA      SSSSSSSS    IIIIII      OOOOOO      222222
PP      PP  AA      AA  SS              II      OO      OO  22      22
PP      PP  AA      AA  SS              II      OO      OO  22      22
PP      PP  AA      AA  SS              II      OO      OO          22
PP      PP  AA      AA  SS              II      OO      OO          22
PPPPPPPP    AA      AA    SSSSSS        II      OO      OO        22
PPPPPPPP    AA      AA    SSSSSS        II      OO      OO        22
PP          AAAAAAAAAA          SS      II      OO      OO      22
PP          AAAAAAAAAA          SS      II      OO      OO      22
PP          AA      AA          SS      II      OO      OO    22          ....
PP          AA      AA          SS      II      OO      OO    22          ....
PP          AA      AA  SSSSSSSS      IIIIII      OOOOOO    2222222222      ....
PP          AA      AA  SSSSSSSS      IIIIII      OOOOOO    2222222222      ....

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
```

PA
VO

```
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000            .TITLE  PAS$IO_INPUT                    ; PASCAL RMS Linkage
0000            .ident  'V04-000'
0000    ;
0000    ;************************************************************************
0000    ;************************************************************************
0000    ;**                                                                    **
0000    ;**                  PASCAL RMS LINKAGE FOR VAX-11/780                 **
0000    ;**                  =================================                 **
0000    ;**                                                                    **
0000    ;**                                                                    **
0000    ;**                   VERSION V1.0-1 -- OCTOBER 1979                   **
0000    ;**                                                                    **
0000    ;**   DEVELOPED BY:  COMPUTER SCIENCE DEPARTMENT                       **
0000    ;**                  UNIVERSITY OF WASHINGTON                          **
0000    ;**                  SEATTLE, WA 98195                                 **
0000    ;**                                                                    **
0000    ;**   AUTHORS:       MARK BAILEY, JOHN CHAN, HELLMUT GOLDE             **
0000    ;**                                                                    **
0000    ;************************************************************************
0000    ;************************************************************************
0000    ;
0000    ; Modified to allow input of 31 character scalar values.
0000    ;               Paul Hohensee  24Jan80
0000    ;
0000    ; Modified to check for overflow of integers during a READ
0000    ;               Susan Azibert 22May80
0000    ;
0000    ; Modified to check for overflow of real and double precision numbers
0000    ;       during a READ
0000    ;               Susan Azibert 22May80
0000    ;
0000    ; Modified to change the setting for PRN_CRLF to <LF> <text> <CR>
0000    ;               Susan Azibert 16Oct80
```

```
0000     ;
0000     ; Modified to force a READLN on read of a string if EOLN is true.
0000     ; Old behavior was to force a READLN only if both EOLN was true
0000     ; and last read was a string read. If the last read was not a string
0000     ; read, a null string (all blanks) was read and the file left
0000     ; positioned at EOLN.
0000     ;               Paul Hohensee 19Jan81
0000     ;
0000     ; Correct PAS$READSCAL so it accepts capital Z.
0000     ;               Paul Hohensee 20Feb81
0000     ;
0000     ; 7. Change PAS$CNV_IN_DEFG to an integer-valued function
0000     ;
0000     ; 8. 11-Aug-81 Paul Hohensee Change to general addressing of external routines
0000     ;
0000     ;*************************************************************************
0000     ;*************************************************************************
0000     ;**                                                                     **
0000     ;**                                                                     **
0000     ;**                             SECTION 2                               **
0000     ;**                                                                     **
0000     ;**                          INPUT PROCEDURES                           **
0000     ;**                                                                     **
0000     ;**                                                                     **
0000     ;*************************************************************************
0000     ;*************************************************************************
```

```
0000  ; For any file variable the following storage is assumed:
0000  ;
0000  ;                  ------------------
0000  ;          FSB:    !     POINTER     !
0000  ;                  ------------------
0000  ;                  !   STATUS WORD   !
0000  ;                  ------------------
0000  ;                  !      LAST       !
0000  ;                  ------------------
0000  ;                  !    LINELIMIT    !
0000  ;                  ------------------
0000  ;                  !    LINECOUNT    !
0000  ;                  ------------------
0000  ;                  !  RECORD NUMBER  !
0000  ;                  ------------------
0000  ;          RAB:    !                 !
0000  ;                     44(HEX) BYTES
0000  ;                  !        .        !
0000  ;                           .
0000  ;                  !        .        !
0000  ;                  ------------------
0000  ;          FAB:    !                 !
0000  ;                     50(HEX) BYTES
0000  ;                  !        .        !
0000  ;                           .
0000  ;                  !        .        !
0000  ;                  ------------------
0000  ;          NAM:    !                 !     NOTE: The NAM block is allocated
0000  ;                     38(HEX) BYTES         for the PASCAL logical files
0000  ;                  !        .        !      'INPUT' and 'OUTPUT' only.
0000  ;                           .
0000  ;                  !                 !
0000  ;                  ------------------
0000  ;
0000  ;          MACRO OPTIONS
0000  ;
0000           .DSABL  GBL                                ; No undefined references
0000           .ENABL  FPT                                ; Rounded arithmetic
0000  ;
0000  ; External references
0000  ;
0000           .EXTRN  PAS$READOK
0000           .EXTRN  PAS$ACTUALGET
0000           .EXTRN  PAS$STATUSUPDAT
0000           .EXTRN  PAS$BLANK_R3
0000           .EXTRN  PAS$IOERROR
0000  ;
0000           .EXTRN  FOR$CNV_IN_DEFG
0000           .EXTRN  OTS$CVT_TI_L
0000  ;
0000  ; Provide definitions of system values
0000  ;
0000           $DSCDEF                                    ; String descriptor definitions
0000           $FABDEF
0000           $RABDEF
0000  ;
0000  ; PASCAL compiler constants
```

```
          0000       ;
          0000       ; NOTE: The constants below with the names 'PAS$C_XXXXX' are
          0000       ;       used in the PASCAL compiler with the names 'XXXXX'. If the
          0000       ;       values in the compiler are altered then the values below
          0000       ;       must be altered accordingly.
          0000       ;
          0000       ;       PAS$C_DFLTRECSI = 257;                  ; default buffer size
          0000       ;       PAS$C_NIL = 0                           ; NIL pointer
          0000       ;       PAS$C_TRUE = 1                          ; TRUE
          0000       ;       PAS$C_FALSE = 0                         ; FALSE
          0000       ;       PAS$C_NOCARR = 0                        ; no carriage control
          0000       ;       PAS$C_CARRIAGE = 1                      ; FORTRAN carriage control
          0000       ;       PAS$C_LIST = 2                          ; LIST carriage control
          0000       ;       PAS$C_PRN = 3                           ; PRN carriage control
          0000       ;
          0000       ; PRN carriage control constants
          0000       ;
          0000       ;       PRN_CRLF = ^X8D01                       ; PRN carriage control constant
          0000                                                     ; for <LF> <text> <CR>
          0000       ;       PRN_NULL = ^X0000                       ; PRN carriage control constant
          0000                                                     ; for no carriage control
          0000       ;
          0000       ; File status block constants
          0000       ;
00000018  0000               FSB$C_BLN = ^X18                        ; FSB block length
          0000       ;       FSB$V_OPEN = 5
          0000       ;       FSB$V_EOF = 1
00000002  0000               FSB$V_EOLN = 2
          0000       ;       FSB$V_GET = 3
          0000       ;       FSB$V_TXT = 4                           ; textfile flag
          0000       ;       FSB$V_RDLN = 0                          ; last access READLN
          0000       ;       FSB$V_DIR = 6                           ; direct access flag
          0000       ;       FSB$V_PUT = 7
          0000       ;       FSB$V_INT = 8                           ; internal flag
          0000       ;       FSB$V_PRMT = 9                          ; prompt flag
          0000       ;       FSB$V_OUTPUT = 10                       ; OUTPUT file flag
0000000B  0000               FSB$V_ACTIN = 11                        ; actual input flag
          0000       ;       FSB$V_DELZ = 30                         ; delete file if empty
          0000       ;       FSB$V_INC = 31                          ; included file flag
          0000       ;       FSB$B_CC = 6                            ; carriage control byte offset
          0000       ;       FSB$M_OPEN = ^X0020
          0000       ;       FSB$M_EOF = ^X0002
          0000       ;       FSB$M_EOLN = ^X0004
          0000       ;       FSB$M_GET = ^X0008
          0000       ;       FSB$M_PRMT = ^X0200
          0000       ;       FSB$M_PUT = ^X00000080
          0000       ;       FSB$M_TXT = ^X0010
          0000       ;       FSB$M_RDLN = ^X0001
          0000       ;       FSB$M_DIR = ^X00000040
          0000       ;       FSB$M_INT = ^X00000100
          0000       ;       FSB$M_OUTPUT = ^X0400
          0000       ;       FSB$M_ACTIN = ^X0800
          0000       ;       FSB$M_DELZ = ^X40000000
          0000       ;       FSB$M_INC = ^X80000000
          0000       ;       FSB$L_CNT = 16                          ; line count (textfiles)
          0000       ;       FSB$L_INC = 20                          ; %INCLUDE block address
          0000       ;       FSB$L_LIM = 12                          ; linelimit
```

```
                    00000008  0000    FSB$L_LST = 8               ; last word offset
                              0000 ;  FSB$L_PFSB = 20             ; related file FSB for prompting
                              0000                                ; for INPUT, has address of OUTPUT FSB
                              0000                                ; for OUTPUT, has address of INPUT FSB
                              0000                                ; (shares storage with include address
                              0000                                ; and direct access record
                              0000                                ; buffer address
                              0000 ;  FSB$L_REC = 20              ; record buffer address for
                              0000                                ; direct access (shares storage
                              0000                                ; with include address and related
                              0000                                ; file FSB)
                    00000004  0000    FSB$L_STA = 4               ; status word offset
                              0000 ;
                              0000 ; Character constants
                              0000 ;
                              0000 ;  TAB = ^X09
                    00000020  0000    SPACE = ^X20
                    00000024  0000    DOLLAR = ^X24
                              0000 ;  FORMFEED = ^XC
                              0000 ;  STAR = ^X2A
                    0000002B  0000    PLUS = ^X2B
                    0000002D  0000    MINUS = ^X2D
                    0000002E  0000    POINT = ^X2E
                    00000030  0000    ZERO = ^X30
                              0000 ;  ONE = ^X31
                    00000038  0000    EIGHT = ^X38
                    00000039  0000    NINE = ^X39
                    00000041  0000    AA = ^X41
                    00000044  0000    DD = ^X44
                    00000045  0000    EE = ^X45
                    0000005A  0000    ZZ = ^X5A
                    0000005F  0000    UNDERSCORE = ^X5F
                    00000061  0000    AA_SMALL = ^X61
                    0000007A  0000    ZZ_SMALL = ^X7A
                              0000 ;
                    0CCCCCCC  0000    MAX10 = 214748364
                    80000000  0000    MAXNEG = ^X80000000
                              0000 ;
                      00000000  0000    .PSECT  _PAS$CODE,          PIC,EXE,SHR,NOWRT
                              0000 ;
                              0000 ;  ********************
                              0000 ;  *                  *
                              0000 ;  *    PAS$GETBIN    *
                              0000 ;  *                  *
                              0000 ;  ********************
                              0000 ;
                              0000 ; Gets the next record from a (binary) file
                              0000 ;
                              0000 ; Argument offsets
                              0000 ;
                              0000 ;  AP                          ; number of arguments (1)
                    00000004  0000    FSB_DISP = 04               ; FSB address
                              0000 ;
                          0080  0000    .ENTRY  PAS$GETBIN,^M<R7>
        57   18   04 AC   C1  0002    ADDL3   FSB_DISP(AP),#FSB$C_BLN,R7 ; R7 = address of RAB
   00000000'GF      6C    FA  0007    CALLG   (AP),G^PAS$READOK
         1E A7      00    90  000E    MOVB    #RAB$C_SEQ,RAB$B_RAC(R7); make sure sequential
```

```
00000000'GF    6C   FA  0012         CALLG   (AP),G^PAS$ACTUALGET     ; get for call to GET
                    04  0019         RET
                        001A  ;
                        001A  ;
                    0000001A         .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                        001A  ;
                        001A  ;      ********************
                        001A  ;      *                  *
                        001A  ;      *    PAS$GETTXT    *
                        001A  ;      *                  *
                        001A  ;      ********************
                        001A  ;
                        001A  ; Advances the file pointer and sets the status word as required
                        001A  ; for textfiles.
                        001A  ;
                        001A  ; Argument offsets
                        001A  ;
                        001A  ;      AP                               ; number of arguments (1)
            00000004    001A         FSB_DISP = 04                    ; FSB address
                        001A  ;
                0040    001A         .ENTRY  PAS$GETTXT,^M<R6>
00000000'GF    6C   FA  001C         CALLG   (AP),G^PAS$READOK
        56  04 AC   D0  0023         MOVL    FSB_DISP(AP),R6          ; R6 = address of FSB
               66   D6  0027         INCL    (R6)
00000000'GF    6C   FA  0029         CALLG   (AP),G^PAS$STATUSUPDAT   ; update status word
                    04  0030         RET
                        0031  ;
                        0031  ;
                    00000031         .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                        0031  ;
                        0031  ;      ********************
                        0031  ;      *                  *
                        0031  ;      *    PAS$READLN    *
                        0031  ;      *                  *
                        0031  ;      ********************
                        0031  ;
                        0031  ; Positions the pointer to the last character of the line, clears
                        0031  ; the EOLN flag, and sets the RDLN flag.
                        0031  ;
                        0031  ; Argument offsets
                        0031  ;
                        0031  ;      AP                               ; number of arguments (1)
            00000004    0031         FSB_DISP = 04                    ; FSB address
                        0031  ;
                0040    0031         .ENTRY  PAS$READLN,^M<R6>
        56  04 AC   D0  0033         MOVL    FSB_DISP(AP),R6          ; R6 = address of FSB
00000000'GF    6C   FA  0037         CALLG   (AP),G^PAS$READOK
        66  08 A6   D0  003E         MOVL    FSB$L_LST(R6),(R6)
               66   D6  0042         INCL    (R6)                     ; set pointer to LAST + 1
00000000'GF    6C   FA  0044         CALLG   (AP),G^PAS$STATUSUPDAT
                    04  004B         RET
                        004C  ;
                        004C  ;
                    0000004C         .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                        004C  ;
                        004C  ;      ********************
                        004C  ;      *                  *
```

```
                           004C   ;        *  PAS$READCHAR     *
                           004C   ;        *                   *
                           004C   ;        *********************
                           004C   ;
                           004C   ; Argument offsets
                           004C   ;
                           004C   ;        AP                             ; number of arguments (1)
            00000004       004C            FSB_DISP = 04                  ; FSB address
            00000008       004C            VAR_DISP = 08                  ; variable address
                           004C   ;
                    0040   004C            .ENTRY  PAS$READCHAR,^M<R6>
      56     04 AC  D0     004E            MOVL    FSB_DISP(AP),R6        ; R6 = address of FSB
                 56  DD    0052            PUSHL   R6
 00000000'GF     01  FB    0054            CALLS   #1,G^PAS$READOK
                           005B   ;
                           005B   ; Store the character and increment pointer
                           005B   ;
   08 BC    00 B6    90    005B            MOVB    @(R6),@VAR_DISP(AP)
                 66  D6    0060            INCL    (R6)
                 56  DD    0062            PUSHL   R6
 00000000'GF     01  FB    0064            CALLS   #1,G^PAS$STATUSUPDAT
                 04        006B            RET
                           006C   ;
                           006C   ;
               0000006C    006C            .PSECT  _PAS$CODE,             PIC,EXE,SHR,NOWRT
                           006C   ;
                           006C   ;        *********************
                           006C   ;        *                   *
                           006C   ;        *   PAS$READSTR     *
                           006C   ;        *                   *
                           006C   ;        *********************
                           006C   ;
                           006C   ; Argument offsets
                           006C   ;
                           006C   ;        AP                             ; number of arguments (3)
            00000004       006C            FSB_DISP = 04                  ; FSB address
            00000008       006C            STR_DISP = 08                  ; string address
            0000000C       006C            LEN_DISP = 12                  ; string length (by value)
                           006C   ;
                    007C   006C            .ENTRY  PAS$READSTR,^M<R2,R3,R4,R5,R6>
      56     04 AC  D0     006E            MOVL    FSB_DISP(AP),R6        ; R6 = address of FSB
   09 04 A6     02  E1     0072            BBC     #FSB$V_EOLN,FSB$L_STA(R6),100$; if EOLN = TRUE,
                 56  DD    0077            PUSHL   R6                     ; go to next line
 00000031'GF     01  FB    0079            CALLS   #1,G^PAS$READLN
                           0080   100$:
                 56  DD    0080            PUSHL   R6
 00000000'GF     01  FB    0082            CALLS   #1,G^PAS$READOK
   50    08 A6  66   C3    0089            SUBL3   (R6),FSB$L_LST(R6),R0  ; R0 = remaining length
OC AC  20    00 B6  50  2C 008E            MOVC5   R0,@(R6),#SPACE,LEN_DISP(AP),@STR_DISP(AP)
          08 BC            0095
      66        51  D0     0097            MOVL    R1,(R6)                ; store new pointer
                56  DD     009A            PUSHL   R6
 00000000'GF    01  FB     009C            CALLS   #1,G^PAS$STATUSUPDAT
                04         00A3            RET
                           00A4   ;
                           00A4   ;
               000000A4    00A4            .PSECT  _PAS$CODE,             PIC,EXE,SHR,NOWRT
```

```
                                    00A4  ;
                                    00A4  ;               *********************
                                    00A4  ;               *                   *
                                    00A4  ;               *   PAS$READSCAL    *
                                    00A4  ;               *                   *
                                    00A4  ;               *********************
                                    00A4  ;
                                    00A4  ; Reads a scalar value from the character file. Lower case letters are
                                    00A4  ; tranlated into upper case letters. If the name can not be found a
                                    00A4  ; runtime error occurs.
                                    00A4  ;
                                    00A4  ; Argument offsets
                                    00A4  ;
                                    00A4  ;               AP                                    ; number of arguments (4)
                          00000004  00A4                FSB_DISP = 04                         ; FSB address
                          00000008  00A4                SCA_DISP = 08                         ; scalar address
                          0000000C  00A4                NAM_DISP = 12                         ; name list address
                          00000010  00A4                MAX_DISP = 16                         ; maximum scalar value.
                                    00A4                                                      ; The name list is 'NAMELEN' times
                                    00A4                                                      ; the maximum scalar value bytes long
                                    00A4  ;
                                    00A4  ; Constants
                                    00A4  ;
                          0000001F  00A4                MAXNAM = 31                           ; maximum scalar name size (in bytes).
                                    00A4  ;                                                     this definition must match 'alfaleng'
                                    00A4  ;                                                     in the compiler.
                          00000020  00A4                NAMELEN = MAXNAM + 1                  ; size in bytes of entry in name list
                                    00A4  ;
                                    00A4  ; The scalar translation table
                                    00A4  ;
                                    00A4  SCALTRANSTABLE:
00'00'00'00'00'00'00'00'00'00'00'00' 00A4          .BYTE   0[^X23 - ^X0 + ^X1]
00'00'00'00'00'00'00'00'00'00'00'00' 00B0
00'00'00'00'00'00'00'00'00'00'00'00' 00BC
                                 24  00C8          .BYTE   DOLLAR
   00'00'00'00'00'00'00'00'00'00'00' 00C9          .BYTE   0[^X2F - ^X25 + ^X1]
      39 38 37 36 35 34 33 32 31 30  00D4          .BYTE   ^X30, ^X31, ^X32, ^X33, ^X34, ^X35, -
                                     00DE                  ^X36, ^X37, ^X38, ^X39
               00'00'00'00'00'00'00' 00DE          .BYTE   0[^X40 - ^X3A + ^X1]
4C 4B 4A 49 48 47 46 45 44 43 42 41  00E5          .BYTE   ^X41, ^X42, ^X43, ^X44, ^X45, -
58 57 56 55 54 53 52 51 50 4F 4E 4D  00F1
                              5A 59  00FD
                                     00FF                  ^X46, ^X47, ^X48, ^X49, ^X4A, -
                                     00FF                  ^X4B, ^X4C, ^X4D, ^X4E, ^X4F, -
                                     00FF                  ^X50, ^X51, ^X52, ^X53, ^X54, -
                                     00FF                  ^X55, ^X56, ^X57, ^X58, ^X59, -
                                     00FF                  ^X5A
                  00 5F 00 00 00 00  00FF          .BYTE   0,0,0,0,UNDERSCORE,0
4C 4B 4A 49 48 47 46 45 44 43 42 41  0105          .BYTE   ^X41, ^X42, ^X43, ^X44, ^X45, -
58 57 56 55 54 53 52 51 50 4F 4E 4D  0111
                              5A 59  011D
                                     011F                  ^X46, ^X47, ^X48, ^X49, ^X4A, -
                                     011F                  ^X4B, ^X4C, ^X4D, ^X4E, ^X4F, -
                                     011F                  ^X50, ^X51, ^X52, ^X53, ^X54, -
                                     011F                  ^X55, ^X56, ^X57, ^X58, ^X59, -
                                     011F                  ^X5A
                     00'00'00'00'00' 011F          .BYTE   0[^X7F - ^X7B + ^X1]
```

```
00'00'00'00'00'00'00'00'00'00'00'00'  0124        .BYTE   0[^X7F]
00'00'00'00'00'00'00'00'00'00'00'00'  0130
00'00'00'00'00'00'00'00'00'00'00'00'  013C
00'00'00'00'00'00'00'00'00'00'00'00'  0148
00'00'00'00'00'00'00'00'00'00'00'00'  0154
00'00'00'00'00'00'00'00'00'00'00'00'  0160
00'00'00'00'00'00'00'00'00'00'00'00'  016C
00'00'00'00'00'00'00'00'00'00'00'00'  0178
00'00'00'00'00'00'00'00'00'00'00'00'  0184
00'00'00'00'00'00'00'00'00'00'00'00'  0190
                  00'00'00'00'00'00'00'  019C
                                      01A3
                              01FC    01A3        .ENTRY  PAS$READSCAL,^M<R2,R3,R4,R5,R6,R7,R8>
                  56   04 AC    D0    01A5        MOVL    FSB_DISP(AP),R6           ; R6 = address of FSB
                     52   56    D0    01A9        MOVL    R6,R2                     ; for PAS$BLANK_R3
                 00000000'GF    16    01AC        JSB     G^PAS$BLANK_R3            ; skip leading blanks
                                      01B2                                          ; returns next address in R1
                                      01B2  ;
                                      01B2  ; Check if first character is a letter
                                      01B2  ;
                  41 8F   61    91    01B2        CMPB    (R1),#AA
                          7A    19    01B6        BLSS    900$                      ; error
                  5A 8F   61    91    01B8        CMPB    (R1),#ZZ
                          0C    15    01BC        BLEQ    110$                      ; ok
                  7A 8F   61    91    01BE        CMPB    (R1),#ZZ_SMALL
                          6E    14    01C2        BGTR    900$                      ; error
                  61 8F   61    91    01C4        CMPB    (R1),#AA_SMALL
                          68    19    01C8        BLSS    900$                      ; error
                                      01CA  ;
                                      01CA  ; Ok, lets read and translate the string
                                      01CA  ;
                                      01CA  110$:
             50   08 A6   51    C3    01CA        SUBL3   R1,FSB$L_LST(R6),R0
                          50    D6    01CF        INCL    R0                        ; R0 = # of characters left in line
                     5E   50    C2    01D1        SUBL2   R0,SP                     ; make room for translated string
                                      01D4                                          ; on stack
  50    FEC9 CF   00   61   50    2F    01D4        MOVTUC  R0,(R1),#0,SCALTRANSTABLE,R0,(SP)
                          6E          01DC
                                      01DD  ;
                                      01DD  ; Update the FSB
                                      01DD  ;
                     66   51    D0    01DD        MOVL    R1,(R6)                   ; update pointer
                          56    DD    01E0        PUSHL   R6
                 00000000'GF   01    FB    01E2        CALLS   #1,G^PAS$STATUSUPDAT
                                      01E9  ;
                                      01E9  ; Try to find a match and store the value
                                      01E9  ;
                     55   5E    C2    01E9        SUBL2   SP,R5                     ; R5 = # of characters translated
                     1F   55    D1    01EC        CMPL    R5,#MAXNAM                ; compare only first maxnam bytes
                          03    15    01EF        BLEQ    115$
                     55   1F    D0    01F1        MOVL    #MAXNAM,R5
                                      01F4  115$:
             54   10 AC   20    C5    01F4        MULL3   #NAMELEN,MAX_DISP(AP),R4; R4 = table offset of current string
                          58    D4    01F9        CLRL    R8                        ; R8 will equal 1 if unique inital substring
                                      01FB  120$:
   6E   55   20   0C BC44   20    2D    01FB        CMPC5   #NAMELEN,@NAM_DISP(AP)[R4],#SPACE,R5,(SP)
                          15    13    0203        BEQL    199$                      ; exit, found an exact  match, search no fur
                  52    B5  0205          TSTW    R2                      ; did we match full input string?
                  05    12  0207          BNEQ    125$
                  58    D6  0209          INCL    R8                      ; R8 := 1 if first initial substring match
             57   54    D0  020B          MOVL    R4,R7                   ; preserve offset
                            020E  125$:
             54   20    C2  020E          SUBL2   #NAMELEN,R4             ; R4 = offset of next string to try
                  E8    18  0211          BGEQ    120$
                  58    D7  0213          DECL    R8                      ; no exact match, was there a unique initial
                  1B    12  0215          BNEQ    900$                    ;    NEQ: no, error
             54   57    D0  0217          MOVL    R7,R4                   ; yes, set up table offset
                            021A  ;
                            021A  ; Store value and exit
                            021A  ;
                            021A  199$:
             54   20    C6  021A          DIVL2   #NAMELEN,R4             ; convert offset to index
  00000100 8F    10 AC  D1  021D          CMPL    MAX_DISP(AP),#256       ; store byte or word?
                  06    14  0225          BGTR    201$
          08 BC   54    90  0227          MOVB    R4,@SCA_DISP(AP)        ; store byte
                  04    11  022B          BRB     202$
                            022D  201$:
          08 BC   54    B0  022D          MOVW    R4,@SCA_DISP(AP)        ; store word
                            0231  202$:
                        04  0231          RET
                            0232  ;
                            0232  ; No match found, input conversion error
                            0232  ;
                            0232  900$:
        7E    8394 8F   3C  0232          MOVZWL  #^X8394,-(SP)
        7E    0090 C6   9A  0237          MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R6),-(SP)
              0088 C6   DD  023C          PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R6)
  00000000'GF     03    FB  0240          CALLS   #3,G^PAS$IOERROR
                            0247  ;
                            0247  ;
                        00000247          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
                            0247  ;
                            0247  ;       ********************
                            0247  ;       *                  *
                            0247  ;       *   PAS$READINT    *
                            0247  ;       *                  *
                            0247  ;       ********************
                            0247  ;
                            0247  ; Argument offsets
                            0247  ;
                            0247  ;       AP                              ; number of arguments
               00000004     0247          FSB_DISP = 04                   ; FSB address
               00000008     0247          VAR_DISP = 08                   ; variable address
                            0247  ; Descriptor offsets
               FFFFFFF4     0247          RESULT = -12                    ; offset of result
               FFFFFFF8     0247          LENGTH = -8                     ; offset of length
               FFFFFFFA     0247          CLASS = -6                      ; offset of class and type
               FFFFFFFC     0247          ADDR = -4                       ; offset of address
                            0247  ;
                      005C  0247          .ENTRY  PAS$READINT,^M<R2,R3,R4,R6>
        56    04 AC     D0  0249          MOVL    FSB_DISP(AP),R6         ; R6 = address of FSB
             52   56    D0  024D          MOVL    R6,R2                   ; R2 needed for PAS$BLANK_R3
  00000000'GF     16        0250          JSB     G^PAS$BLANK_R3          ; skip leading blanks
                            0256                                          ; returns R1 as address of byte
```

```
                    50    D4  0256              CLRL    R0                    ; set counter
                    52    D4  0258              CLRL    R2                    ; set sum register
                    54    D4  025A              CLRL    R4                    ; clear extract register
              53    01    D0  025C              MOVL    #1,R3                 ; set sign flag
                    51    DD  025F              PUSHL   R1                    ; store address of string in descriptor
              2B    61    91  0261              CMPB    (R1),#PLUS            ; plus?
                    06    12  0264              BNEQU   100$
                    50    D6  0266              INCL    R0
                    51    D6  0268              INCL    R1
                    10    11  026A              BRB     110$
                              026C       100$:                                ; minus?
              2D    61    91  026C              CMPB    (R1),#MINUS
                    0B    12  026F              BNEQU   110$
        53   FFFFFFFF 8F  D0  0271              MOVL    #-1,R3                ; set sign flag
                    50    D6  0278              INCL    R0
                    51    D6  027A              INCL    R1
                              027C       110$:                                ; process integer
         54    61   30    83  027C              SUBB3   #ZERO,(R1),R4         ; R4 = integer value of digit
                    30    19  0280              BLSS    120$
              39    61    91  0282              CMPB    (R1),#NINE
                    2B    14  0285              BGTR    120$                  ; branch if not digit
        0CCCCCCC 8F 52    D1  0287              CMPL    R2,#MAX10             ; check for out of range
                    16    19  028E              BLSS    111$
                    4D    14  0290              BGTR    900$
              38    61    91  0292              CMPB    (R1),#EIGHT
                    48    14  0295              BGTR    900$
                    0D    19  0297              BLSS    111$
                    53    D5  0299              TSTL    R3                    ; check for largest negative
                    42    18  029B              BGEQ    900$
        52   80000000 8F  D0  029D              MOVL    #MAXNEG,R2
                    06    11  02A4              BRB     112$
                              02A6      111$:
              52    0A    C4  02A6              MULL2   #10,R2
              52    54    C0  02A9              ADDL2   R4,R2                 ; R2 = new sum
                              02AC      112$:
                    50    D6  02AC              INCL    R0                    ; increment counter
                    51    D6  02AE              INCL    R1                    ; increment address
                    CA    11  02B0              BRB     110$                  ; loop if more digits
                              02B2      120$:                                 ; read until not digit
                    50    D5  02B2              TSTL    R0                    ; test for no digits read
                    29    13  02B4              BEQL    900$                  ; conversion error
              0A    50    D1  02B6              CMPL    R0,#10                ; check for excess digits
                    24    14  02B9              BGTR    900$
                    50    DD  02BB              PUSHL   R0                    ; store length of descriptor
                    7E    D4  02BD              CLRL    -(SP)                 ; clear a longword for the result
                    5E    DD  02BF              PUSHL   SP                    ; pass the address to store the result
               F8   AD    9F  02C1              PUSHAB  LENGTH(FP)            ; pass the address of the descriptor
        00000000'GF 02    FB  02C4              CALLS   #2,G^OTS$CVT_TI_L     ; call conversion routine
                11  50    E9  02CB              BLBC    R0,900$               ; if error, output message
             08 BC  8E    D0  02CE              MOVL    (SP)+,@VAR_DISP(AP)   ; store integer
              66    51    D0  02D2              MOVL    R1,(R6)               ; restore pointer address
                    56    DD  02D5              PUSHL   R6
        00000000'GF 01    FB  02D7              CALLS   #1,G^PAS$STATUSUPDAT  ; update status block
                          04  02DE              RET
                              02DF      ;
                              02DF      ; No match found, input conversion error
                              02DF      ;
```

```
                                 02DF    900$:
           7E   8394 8F    3C    02DF          MOVZWL  #^X8394,-(SP)
           7E   0090 C6    9A    02E4          MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R6),-(SP)
                0088 C6    DD    02E9          PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R6)
     00000000'GF      03    FB    02ED          CALLS   #3,G^PAS$IOERROR
                                 02F4    ;
                                 02F4    ;
                             000002F4          .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                                 02F4    ;
                                 02F4    ;     ********************
                                 02F4    ;     *                  *
                                 02F4    ;     *   PAS$READREAL   *
                                 02F4    ;     *                  *
                                 02F4    ;     ********************
                                 02F4    ;
                                 02F4    ; Argument offsets
                                 02F4    ;
                                 02F4    ;     AP                                ; number of arguments (2)
                     00000004    02F4          FSB_DISP = 04                     ; FSB address
                     00000008    02F4          VAR_DISP = 08                     ; variable address
                                 02F4    ;
                         001C    02F4          .ENTRY  PAS$READREAL,^M<R2,R3,R4>
           52     04 AC    D0    02F6          MOVL    FSB_DISP(AP),R2           ; R2 = address of FSB
      00000000'GF      16    02FA          JSB     G^PAS$BLANK_R3            ; skip leading blanks
                                 0300                                            ; returns located byte in R1
           50        51    D0    0300          MOVL    R1,R0                     ; save starting address
                                 0303                                            ; check for plus
           2B        61    91    0303          CMPB    (R1),#PLUS
                     04    12    0306          BNEQ    210$
                     51    D6    0308          INCL    R1
                     07    11    030A          BRB     220$
                                 030C    210$:                                   ; check for minus if not plus
           2D        61    91    030C          CMPB    (R1),#MINUS
                     02    12    030F          BNEQ    220$
                     51    D6    0311          INCL    R1
                                 0313    220$:                                   ; count integer part
           30        61    91    0313          CMPB    (R1),#ZERO
                     09    19    0316          BLSS    230$
           39        61    91    0318          CMPB    (R1),#NINE
                     04    14    031B          BGTR    230$
                     51    D6    031D          INCL    R1
                     F2    11    031F          BRB     220$                      ; loop
                                 0321    230$:                                   ; count decimal point
           2E        61    91    0321          CMPB    (R1),#POINT
                     10    12    0324          BNEQ    250$
                     51    D6    0326          INCL    R1
                                 0328    240$:                                   ; count decimal part
           30        61    91    0328          CMPB    (R1),#ZERO
                     09    19    032B          BLSS    250$
           39        61    91    032D          CMPB    (R1),#NINE
                     04    14    0330          BGTR    250$
                     51    D6    0332          INCL    R1
                     F2    11    0334          BRB     240$                      ; loop
                                 0336    250$:                                   ; check for 'E'
        45 8F        61    91    0336          CMPB    (R1),#EE
                     06    13    033A          beql    251$
        65 8F        61    91    033C          cmpb    (r1),#^a'e'
```

```
                 2A   12  0340          BNEQ    280$                       ; done if no exponent
                          0342  251$:                                      ; found exponent
                 51   D6  0342          INCL    R1
                          0344                                             ; check sign
           2B    61   91  0344          CMPB    (R1),#PLUS
                 04   12  0347          BNEQ    260$
                 51   D6  0349          INCL    R1
                 07   11  034B          BRB     270$
                          034D  260$:                                      ; check minus if not plus
           2D    61   91  034D          CMPB    (R1),#MINUS
                 02   12  0350          BNEQ    270$
                 51   D6  0352          INCL    R1
                          0354  270$:                                      ; two digit exponent
           30    61   91  0354          CMPB    (R1),#ZERO
                 13   19  0357          BLSS    280$
           39    61   91  0359          CMPB    (R1),#NINE
                 0E   14  035C          BGTR    280$
                 51   D6  035E          INCL    R1
           30    61   91  0360          CMPB    (R1),#ZERO
                 07   19  0363          BLSS    280$
           39    61   91  0365          CMPB    (R1),#NINE
                 02   14  0368          BGTR    280$
                 51   D6  036A          INCL    R1
                          036C  280$:                                      ; finished with number
      53   51    50   C3  036C          SUBL3   R0,R1,R3                   ; R3 = length
        04 BC    51   D0  0370          MOVL    R1,@FSB_DISP(AP)           ; update file pointer
                 53   D5  0374          TSTL    R3
                 25   13  0376          BEQL    900$                       ; branch if conversion error
                          0378  ;
                          0378  ; Make room for value on stack and convert input
                          0378  ;
           5E    08   C2  0378          SUBL2   #8,SP
           54    5E   D0  037B          MOVL    SP,R4                      ; R4 = address of double result
                 53   DD  037E          PUSHL   R3                         ; length
                 54   DD  0380          PUSHL   R4                         ; value address
                 50   DD  0382          PUSHL   R0                         ; string address
  00000473'GF    03   FB  0384          CALLS   #3,G^PAS$CNV_IN_DEFG
              0F 50   E9  038B          BLBC    R0,900$                    ; branch if error
        08 BC    64   76  038E          CVTDF   (R4),@VAR_DISP(AP)         ; store read number
              04 AC   DD  0392          PUSHL   FSB_DISP(AP)
  00000000'GF    01   FB  0395          CALLS   #1,G^PAS$STATUSUPDAT       ; update status block
                      04  039C          RET
                          039D  ;
                          039D  ; Input conversion error
                          039D  ;
                          039D  900$:
      7E   8394  8F   3C  039D          MOVZWL  #^X8394,-(SP)
      7E   0090  C2   9A  03A2          MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R2),-(SP)
           0088  C2   DD  03A7          PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R2)
  00000000'GF    03   FB  03AB          CALLS   #3,G^PAS$IOERROR
                          03B2  ;
                          03B2  ;
                      000003B2          .PSECT  _PAS$CODE,               PIC,EXE,SHR,NOWRT
                          03B2  ;
                          03B2  ;       ********************
                          03B2  ;       *                  *
                          03B2  ;       *   PAS$READDOUB   *
```

```
                         03B2  ;          *                   *
                         03B2  ;          ********************
                         03B2  ;
                         03B2  ; Argument offsets
                         03B2  ;
                         03B2  ;          AP                               ; number of arguments (2)
              00000004   03B2             FSB_DISP = 04                    ; FSB address
              00000008   03B2             VAR_DISP = 08                    ; variable address
                         03B2  ;
                  000C   03B2             .ENTRY  PAS$READDOUB,^M<R2,R3>
  52    04 AC      D0    03B4             MOVL    FSB_DISP(AP),R2          ; R2 = address of FSB
00000000'GF        16    03B8             JSB     G^PAS$BLANK_R3           ; skip leading blanks
                         03BE                                              ; R1 = located address
       50   51     D0    03BE             MOVL    R1,R0                    ; save starting address
                         03C1  ;
                         03C1  ; Check for plus
                         03C1  ;
       2B   61     91    03C1             CMPB    (R1),#PLUS
            04     12    03C4             BNEQ    210$
            51     D6    03C6             INCL    R1
            07     11    03C8             BRB     220$
                         03CA  210$:                                       ; check for minus if not plus
       2D   61     91    03CA             CMPB    (R1),#MINUS
            02     12    03CD             BNEQ    220$
            51     D6    03CF             INCL    R1
                         03D1  220$:                                       ; count integer part
       30   61     91    03D1             CMPB    (R1),#ZERO
            09     19    03D4             BLSS    230$
       39   61     91    03D6             CMPB    (R1),#NINE
            04     14    03D9             BGTR    230$
            51     D6    03DB             INCL    R1
            F2     11    03DD             BRB     220$                     ; loop
                         03DF  230$:                                       ; count decimal point
       2E   61     91    03DF             CMPB    (R1),#POINT
            10     12    03E2             BNEQ    250$
            51     D6    03E4             INCL    R1
                         03E6  240$:                                       ; count decimal part
       30   61     91    03E6             CMPB    (R1),#ZERO
            09     19    03E9             BLSS    250$
       39   61     91    03EB             CMPB    (R1),#NINE
            04     14    03EE             BGTR    250$
            51     D6    03F0             INCL    R1
            F2     11    03F2             BRB     240$                     ; loop
                         03F4  250$:                                       ; check for 'D' or 'E'
    44 8F   61     91    03F4             CMPB    (R1),#DD
            12     13    03F8             BEQL    251$
    64 8F   61     91    03FA             cmpb    (r1),#^a'd'
            0C     13    03FE             beql    251$
    45 8F   61     91    0400             CMPB    (R1),#EE
            06     13    0404             beql    251$
    65 8F   61     91    0406             cmpb    (r1),#^a'e'
            2A     12    040A             BNEQ    280$                     ; done if no exponent
                         040C  251$:                                       ; found exponent
            51     D6    040C             INCL    R1
                         040E                                              ; check sign
       2B   61     91    040E             CMPB    (R1),#PLUS
            04     12    0411             BNEQ    260$
```

```
                   51   D6 0413          INCL    R1
                   07   11 0415          BRB     270$
                           0417  260$:                                   ; check minus if not plus
          2D       61   91 0417          CMPB    (R1),#MINUS
                   02   12 041A          BNEQ    270$
                   51   D5 041C          INCL    R1
                           041E  270$:                                   ; two digit exponent
          30       61   91 041E          CMPB    (R1),#ZERO
                   13   19 0421          BLSS    280$
          39       61   91 0423          CMPB    (R1),#NINE
                   0E   14 0426          BGTR    280$
                   51   D6 0428          INCL    R1
          30       61   91 042A          CMPB    (R1),#ZERO
                   07   19 042D          BLSS    280$
          39       61   91 042F          CMPB    (R1),#NINE
                   02   14 0432          BGTR    280$
                   51   D6 0434          INCL    R1
                           0436  280$:                                   ; finished with number
       53     51   50   C3 0436          SUBL3   R0,R1,R3                ; R3 = length
          04  BC   51   D0 043A          MOVL    R1,@FSB_DISP(AP)        ; update file pointer
                   53   D5 043E          TSTL    R3
                   1C   13 0440          BEQL    900$                    ; branch if conversion error
                           0442        ;
                           0442        ; Convert input
                           0442        ;
                   53   DD 0442          PUSHL   R3                      ; length
              08   AC   DD 0444          PUSHL   VAR_DISP(AP)            ; variable address
                   50   DD 0447          PUSHL   R0                      ; string address
    00000473'GF    03   FB 0449          CALLS   #3,G^PAS$CNV_IN_DEFG
              0B   50   E9 0450          BLBC    R0,900$                 ; branch if error
              04   AC   DD 0453          PUSHL   FSB_DISP(AP)
    00000000'GF    01   FB 0456          CALLS   #1,G^PAS$STATUSUPDAT    ; update status block
                   04      045D          RET
                           045E        ;
                           045E        ; Input conversion error
                           045E        ;
                           045E  900$:
       7E     8394 8F   3C 045E          MOVZWL  #^X8394,-(SP)
       7E     0090 C2   9A 0463          MOVZBL  <FSB$C_BLN+RAB$C_BLN+FAB$B_FNS>(R2),-(SP)
              0088 C2   DD 0468          PUSHL   <FSB$C_BLN+RAB$C_BLN+FAB$L_FNA>(R2)
    00000000'GF    03   FB 046C          CALLS   #3,G^PAS$IOERROR
                           0473        ;
                           0473        ;
                       00000473          .PSECT  _PAS$CODE,              PIC,EXE,SHR,NOWRT
                           0473        ;
                           0473        ;         ********************
                           0473        ;         *                  *
                           0473        ;         *  PAS$CNV_IN_DEFG *
                           0473        ;         *                  *
                           0473        ;         ********************
                           0473        ;
                           0473        ; Converts a character string representing a real or double precission
                           0473        ; value into a double precission value
                           0473        ;
                           0473        ; Argument offsets
                           0473        ;
                           0473        ;         AP                             ; number of arguments (3)
```

```
                    00000004  0473          BUF_DISP = 04                  ; buffer address
                    00000008  0473          VAR_DISP = 08                  ; variable address (of quadword)
                    0000000C  0473          LEN_DISP = 12                  ; string length (by value)
                              0473     ;
                        0000  0473          .ENTRY  PAS$CNV_IN_DEFG,^M<>
                              0475     ;
                              0475     ; Make room for descriptor on stack
                              0475     ;
          5E     08     C2    0475          SUBL2   #DSC$C_S_BLN,SP
          51     5E     D0    0478          MOVL    SP,R1                  ; R1 = address of descriptor
     04 A1    04 AC     D0    047B          MOVL    BUF_DISP(AP),DSC$A_POINTER(R1); string address
        61    0C AC     B0    0480          MOVW    LEN_DISP(AP),DSC$W_LENGTH(R1); string length
                              0484     ;
                              0484     ; Convert the value
                              0484     ;
                 00     DD    0484          PUSHL   #0                     ; zero digits in fraction
              08 AC     DD    0486          PUSHL   VAR_DISP(AP)           ; variable address
                 51     DD    0489          PUSHL   R1                     ; descriptor address
  00000000'GF    03     FB    048B          CALLS   #3,G^FOR$CNV_IN_DEFG
                        04    0492          RET
                              0493     ;
                              0493     ;
                              0493     ;
                              0493          .END
```

```
                                                            H 16
PAS$IO_INPUT                      ; PASCAL RMS linkage               16-SEP-1984 02:07:22  VAX/VMS Macro V04-00        Page  17
Symbol table                                                          5-SEP-1984 02:32:14  [PASCAL.SRC]PASIO2.MAR;1          (1)

AA                                = 00000041                 UNDERSCORE                    = 0000005F
AA_SMALL                          = 00000061                 VAR_DISP                      = 00000008
ADDR                              = FFFFFFFC                 ZERO                          = 00000030
BUF_DISP                          = 00000004                 ZZ                            = 0000005A
CLASS                             = FFFFFFFA                 ZZ_SMALL                      = 0000007A
DD                                = 00000044
DOLLAR                            = 00000024
DSC$A_POINTER                     = 00000004
DSC$C_S_BLN                       = 00000008
DSC$W_LENGTH                      = 00C00000
EE                                = 00000045
EIGHT                             = 00000038
FAB$B_FNS                         = 00000034
FAB$L_FNA                         = 0000002C
FOR$CNV_IN_DEFG                     ********    X    00
FSB$C_BLN                         = 00000018
FSB$L_LST                         = 00000008
FSB$L_STA                         = 00000004
FSB$V_ACTIN                       = 0000000B
FSB$V_EOLN                        = 00000002
FSB_DISP                          = 00000004
LENGTH                            = FFFFFFF8
LEN_DISP                          = 0000000C
MAXTO                             = 0CCCCCCC
MAXNAM                            = 0000001F
MAXNEG                            = 80000000
MAX_DISP                          = 00000010
MINUS                             = 0000002D
NAMELEN                           = 00000020
NAM_DISP                          = 0000000C
NINE                              = 00000039
OTS$CVT_TI_L                        ********    X    00
PAS$ACTUALGET                       ********    X    00
PAS$BLANK_R3                        ********    X    00
PAS$CNV_IN_DEFG                     00000473 RG      02
PAS$GETBIN                          00000000 RG      02
PAS$GETTXT                          0000001A RG      02
PAS$IOERROR                         ********    X    00
PAS$READCHAR                        0000004C RG      02
PAS$READDOUB                        000003B2 RG      02
PAS$READINT                         00000247 RG      02
PAS$READLN                          00000031 RG      02
PAS$READOK                          ********    X    00
PAS$READREAL                        000002F4 RG      02
PAS$READSCAL                        000001A3 RG      02
PAS$READSTR                         0000006C RG      02
PAS$STATUSUPDAT                     ********    X    00
PLUS                              = 0000002B
POINT                             = 0000002E
RAB$B_RAC                         = 0000001E
RAB$C_BLN                         = 00000044
RAB$C_SEQ                         = 00000000
RESULT                            = FFFFFFF4
SCALTRANSTABLE                      000000A4 R       02
SCA_DISP                          = 00000008
SPACE                             = 00000020
STR_DISP                          = 00000008
```

```
                                            +----------------+
                                            ! 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                       00000493  ( 1171.)  02 (  2.)    PIC   USR   CON   REL   LCL   SHR   EXE   RD  NOWRT NOVEC BYTE

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

Phase                     Page faults    CPU Time       Elapsed Time
-----                     -----------    --------       ------------
Initialization                     32    00:00:00.08    00:00:00.64
Command processing                106    00:00:00.47    00:00:02.21
Pass 1                            206    00:00:05.52    00:00:11.91
Symbol table sort                   0    00:00:00.60    00:00:00.61
Pass 2                            153    00:00:02.10    00:00:04.85
Symbol table output                 8    00:00:00.07    00:00:00.09
Psect synopsis output               2    00:00:00.03    00:00:00.03
Cross-reference output              0    00:00:00.00    00:00:00.00
Assembler run totals              510    00:00:08.90    00:00:20.36

The working set limit was 1200 pages.
34227 bytes (67 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 484 non-local and 35 local symbols.
831 source lines were read in Pass 1, producing 40 object records in Pass 2.
10 pages of virtual memory were used to define 9 macros.

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

Macro library name                        Macros defined
------------------                        --------------
_$255$DUA28:[SYSLIB]STARLET.MLB;2                     6

497 GETS were required to define 6 macros.

There were no errors, warnings or information messages.

MACRO/DISABLE=TRACE/LIS=LIS$:PASIO2/OBJ=OBJ$:PASIO2 MSRC$:PASIO2/UPDATE=(ENH$:PASIO2)
```

0292 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY