I 4

```
PPPPPPPPPPPP     LLL                  IIIIIIIII     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP     LLL                  IIIIIIIII     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPPPPPPPPPPP     LLL                  IIIIIIIII     RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
PPP         PPP  LLL                     III        RRR         RRR        TTT        LLL
PPP         PPP  LLL                     III        RRR         RRR        TTT        LLL
PPP         PPP  LLL                     III        RRR         RRR        TTT        LLL
PPP         PPP  LLL                     III        RRR         RRR        TTT        LLL
PPP         PPP  LLL                     III        RRR         RRR        TTT        LLL
PPP         PPP  LLL                     III        RRR         RRR        TTT        LLL
PPPPPPPPPPPP     LLL                     III        RRRRRRRRRRRR           TTT        LLL
PPPPPPPPPPPP     LLL                     III        RRRRRRRRRRRR           TTT        LLL
PPPPPPPPPPPP     LLL                     III        RRRRRRRRRRRR           TTT        LLL
PPP              LLL                     III        RRR   RRR              TTT        LLL
PPP              LLL                     III        RRR   RRR              TTT        LLL
PPP              LLL                     III        RRR   RRR              TTT        LLL
PPP              LLL                     III        RRR      RRR           TTT        LLL
PPP              LLL                     III        RRR      RRR           TTT        LLL
PPP              LLL                     III        RRR      RRR           TTT        LLL
PPP              LLLLLLLLLLLLLLL      IIIIIIIII     RRR         RRR        TTT        LLLLLLLLLLLLLLL
PPP              LLLLLLLLLLLLLLL      IIIIIIIII     RRR         RRR        TTT        LLLLLLLLLLLLLLL
PPP              LLLLLLLLLLLLLLL      IIIIIIIII     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

```
PPPPPPPP    LL            IIIIII     CCCCCCCC  VV      VV  TTTTTTTTTT  PPPPPPPP     IIIIII     CCCCCCCC
PPPPPPPP    LL            IIIIII     CCCCCCCC  VV      VV  TTTTTTTTTT  PPPPPPPP     IIIIII     CCCCCCCC
PP      PP  LL              II     CC          VV      VV      TT      PP      PP     II     CC
PP      PP  LL              II     CC          VV      VV      TT      PP      PP     II     CC
PP      PP  LL              II     CC          VV      VV      TT      PP      PP     II     CC
PP      PP  LL              II     CC          VV      VV      TT      PP      PP     II     CC
PPPPPPPP    LL              II     CC          VV      VV      TT      PPPPPPPP       II     CC
PPPPPPPP    LL              II     CC          VV      VV      TT      PPPPPPPP       II     CC
PP          LL              II     CC          VV      VV      TT      PP             II     CC
PP          LL              II     CC          VV      VV      TT      PP             II     CC
PP          LL              II     CC            VV  VV        TT      PP             II     CC
PP          LL              II     CC            VV  VV        TT      PP             II     CC
PP          LLLLLLLLLL    IIIIII     CCCCCCCC      VV          TT      PP           IIIIII     CCCCCCCC
PP          LLLLLLLLLL    IIIIII     CCCCCCCC      VV          TT      PP           IIIIII     CCCCCCCC


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
```

Table of contents

(1)	122	pli$cvt_to_pic - convert numeric to picture
(2)	251	edit interpret routines
(2)	429	pli$cvt_fr_pic - convert picture to numeric
(2)	620	pli$valid_pic - validate picture value

```
0000     1          .title pli$cvtpic - convert numeric and picture
0000     2          .ident  /1-003/                                         ; Edit CGN1003
0000     3                                                                  ; Edit WHM1002
0000     4 ;
0000     5 ;****************************************************************************
0000     6 ;*                                                                          *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000     9 ;*  ALL RIGHTS RESERVED.                                                    *
0000    10 ;*                                                                          *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    16 ;*  TRANSFERRED.                                                            *
0000    17 ;*                                                                          *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
0000    20 ;*  CORPORATION.                                                            *
0000    21 ;*                                                                          *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    24 ;*                                                                          *
0000    25 ;*                                                                          *
0000    26 ;****************************************************************************
0000    27 ;
0000    28 ;
0000    29 ;++
0000    30 ; facility:
0000    31 ;
0000    32 ;       VAX/VMS PL1 Run-Time library.
0000    33 ;
0000    34 ; abstract:
0000    35 ;
0000    36 ; This module contains routines to convert numeric to picture and picture
0000    37 ; to numeric.
0000    38 ;
0000    39 ; author: R. Heinen 21-JAN-1980
0000    40 ;
0000    41 ; modified on 13-Feb-1981 by R. Heinen
0000    42 ;       fixed problem with convert from overpunched sign character.
0000    43 ;
0000    44 ;       1-002   Bill Matthews   29-September-1982
0000    45 ;
0000    46 ;               Invoke macros $defdat and rtshare instead of $defopr and share.
0000    47 ;
0000    48 ;       1-003   Chip Nylander   04-April-1983
0000    49 ;
0000    50 ;               Fix conversion to picture with floating sign and no non-zero
0000    51 ;               digits to the left of the decimal point.
0000    52 ;
0000    53 ;               The fix is as follows:  when there is a floating sign, the
0000    54 ;               floating sign must be placed into the picture when
0000    55 ;               significance is established.  Significance is established
0000    56 ;               in three ways: any non-zero digit in the source, any
0000    57 ;               non-suppressed digit in the picture specification (e.g. '9'),
```

```
                     0000     58 ;               or an overt significance specifier ("V").
                     0000     59 ;
                     0000     60 ;               Move_zero_supress and move_digits take care of the first
                     0000     61 ;               two cases.  The third case was previously neglected;
                     0000     62 ;               set_significance now takes care of it.
                     0000     63 ;
                     0000     64 ;--
                     0000     65
                     0000     66 ;
                     0000     67 ; external definitions
                     0000     68 ;
                     0000     69         $defpic                         ; define picture constant
                     0000     70 ;
                     0000     71 ; local definitions
                     0000     72 ;
                     0000     73 ; define arguments for both routines
                     0000     74 ;
00000004             0000     75         picture_constant = 4            ;
00000008             0000     76         source_size      = 8            ;
0000000C             0000     77         source_address   = 12           ;
00000010             0000     78         target_size      = 16           ;
00000014             0000     79         target_address   = 20           ;
                     0000     80
                     0000     81 ;
                     0000     82 ; define stack for numeric to picture
                     0000     83 ;
FFFFFFFC             0000     84         sign             = -4           ; sign byte
FFFFFFFB             0000     85         float            = -5           ; float byte
FFFFFFFA             0000     86         significance     = -6           ; significance indicator
FFFFFFF9             0000     87         fil              = -7           ; fill character
FFFFFFF8             0000     88         zero_indic       = -8           ; zero indicator
00000009             0000     89         cvt_to_pic_stack= 9             ; size of stack
                     0000     90
                     0000     91 ;
                     0000     92 ; define picture to numeric stack
                     0000     93 ;
                FFFFFFFC     0000     94         found_sign       = -4           ; sign found
                FFFFFFD8     0000     95         inter_result     = -40          ; 31 bytes of storage for numeric value
                00000028     0000     96         cvt_fr_pic_stack= 40            ; stack size
                             0000     97
                             0000     98 ;
                             0000     99 ; local data
                             0000    100 ;
                             0000    101         rtshare
                             0000    102 ;
                             0000    103 ; conversion tables for over punch
                             0000    104 ;
                             0000    105 plus_over_punch:
      49 48 47 46 45 44 43 42 41 7B  0000    106         .byte   123,65,66,67,68,69,70,71,72,73
                             000A    107 minus_over_punch:
      52 51 50 4F 4E 4D 4C 4B 4A 7D  000A    108         .byte   125,74,75,76,77,78,79,80,81,82
                             0014    109 packed_zero:
                          0C 0014    110         .packed 0
                             0015    111 valid_char_table:
      39 38 37 36 35 34 33 32 31 30  0015    112         .ascii  /0123456789/
2A 20 64 44 63 43 24 2C 2E 2F 2D 2B  001F    113         .ascii  '+-/.,$CcDd *'
      49 48 47 46 45 44 43 42 41 7B  002B    114         .byte   123,65,66,67,68,69,70,71,72,73
```

```
      52 51 50 4F 4E 4D 4C 4B 4A 7D  0035   115          .byte   125,74,75,76,77,78,79,80,81,82
                                     0000002A 003F   116 valid_char_size = .-valid_char_table
                                              003F   117 over_punch_value:
   30 31 32 33 34 35 36 37 38 39 30  003F   118          .ascii  /09876543210/                   ; 0,R,Q,P,O,N,M,L,K,J,}
      30 31 32 33 34 35 36 37 38 39  004A   119          .ascii  /9876543210/                    ; I,H,G,F,E,D,C,B,A,{
                  33 30 34 30 30 30  0054   120          .ascii  /000403/                        ; pick up extra D anc C
```

```
                        005A   122          .sbttl  pli$cvt_to_pic - convert numeric to picture
                        005A   123 ;++
                        005A   124 ; pli$cvt_to_pic - convert numeric to picture
                        005A   125 ;
                        005A   126 ; functional description:
                        005A   127 ;
                        005A   128 ; This routine converts a packed decimal string described by source_size(ap)
                        005A   129 ; and source_address(ap) to a character string described by target_size(ap)
                        005A   130 ; and target_address(ap) based on the picture constant block addressed by
                        005A   131 ; picture_constant(ap).
                        005A   132 ;
                        005A   133 ; inputs:
                        005A   134 ;
                        005A   135 ;       0(ap) = 5
                        005A   136 ;       4(ap) = picture_constant address
                        005A   137 ;       8(ap) = source size
                        005A   138 ;       12(ap) = source address
                        005A   139 ;       16(ap) = target size
                        005A   140 ;       20(ap) = target address
                        005A   141 ;
                        005A   142 ; outputs:
                        005A   143 ;
                        005A   144 ;       target string is filled in.
                        005A   145 ;
                        005A   146 ; ERROR maybe signalled.
                        005A   147 ;
                        005A   148 ;--
                  CFFC  005A   149          .entry  pli$cvt_to_pic,^m<iv,dv,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
5B      04 AC     D0    005C   150          movl    picture_constant(ap),r11; address picture constant
        06 AB     95    0060   151          tstb    pic$b_language(r11)     ; type runtime?
           03     13    0063   152          beql    5$                      ; if eql then yes
                        0065   153 ;
                        0065   154 ; process editpc type
                        0065   155 ;
                  00A5  31    0065   156          brw     error                   ; temp---- error
                        0068   157 ;
                        0068   158 ; interpret subroutine at runtime
                        0068   159 ;
                  5E    09    C2    0068   160 5$:      subl    #cvt_to_pic_stack,sp    ; allocate stack space
                        006B   161 ;
                        006B   162 ; convert source string to internal buffer
                        006B   163 ;
              58    0C AC     D0    006B   164          movl    source_address(ap),r8   ; get address of the source string
              56    08 AC     9A    006F   165          movzbl  source_size(ap),r6      ; get source digit size
              6B    08 AC     B1    0073   166          cmpw    source_size(ap),pic$w_pq(r11); source same p,q as result?
                    1E        13    0077   167          beql    7$                      ; continue if yes
              50    09 AC     9A    0079   168          movzbl  source_size+1(ap),r0    ; get scale of source
                 59    6B     9A    007D   169          movzbl  pic$w_pq(r11),r9        ; get size of result
              52    01 AB     9A    0080   170          movzbl   ic$w_pq+1(r11),r2      ; get scale of result
                 52    50     C2    0084   171          subl    r0,r2                   ; source - result = shift
                 5E    59     C2    0087   172          subl    r9,sp                   ; allocate space for shift
6E    59    00    68  56    52     F8    008A   173          ashp    r2,r6,(r8),#0,r9,(sp)   ;
                 56    59     D0    009?   174          movl    r9,r6                   ; use new size
                 58    6E     9E    0094   175          movab   (sp),r8                 ; address it
                 5E    56     C2    0097   176 7$:      subl    r6,sp                   ; allocate room for result
                       5E     D7    009A   177          decl    sp                      ; allocate for sign
                    F8 AD     94    009C   178          clrb    zero_indic(fp)          ; assume zero
```

```
                6E   56   68    56   08  009F   179          cvtps   r6,(r8),r5,(sp)         ; convert to character
                                03   13  00A4   180          beql    10$                     ; if eql then zero
                                  F8 AD  96  00A6   181          incb    zero_indic(fp)          ; set non zero
                                         00A9   182 ;
                                         00A9   183 ; blank out the target field in case of error
                                         00A9   184 ;
          10 AC   20   14 BC    00    2C  00A9   185 10$:     movc5   #0,@target_address(ap),#32,target_size(ap),@target_address(ap);
                                14 BC          00B0
                                         00B2   186 ;
                                         00B2   187 ; initialize the assumed values
                                         00B2   188 ;
                                  FB AD  94  00B2   189          clrb    float(fp)               ; float is undefined
                                  FA AD  94  00B5   190          clrb    significance(fp)        ; significance is off
                                F9 AD    20  90  00B8   191          movb    #^a/ /,fill(fp)         ; fill begins as blank
                                   51    6E  9E  00BC   192          movab   (sp),r1                 ; address source string ( movc side effect)
                                FC AD    61  90  00BF   193          movb    (r1),sign(fp)           ; get sign
                                   2B    81  91  00C3   194          cmpb    (r1)+,#^a/+/            ; positive?
                                         05  13  00C6   195          beql    15$                     ; if eql then yes
                             40 05 AB    04  E1  00C8   196          bbc     #pic_v_has_sign,pic$b_flags(r11),error; else must have sign specifie
                                         00CD   197 ;
                                         00CD   198 ; allocate space for initial target string
                                         00CD   199 ;
                                50    04 AB  9A  00CD   200 15$:     movzbl  pic$b_byte_size(r11),r0 ; get max size of target
                                   5E    50  C2  00D1   201          subl    r0,sp                   ; allocate the space
                                   53    6E  9E  00D4   202          movab   (sp),r3                 ; address it ( movc side effect )
                                5A    08 AB  9E  00D7   203          movab   pic$b_program(r11),r10  ; address edit program
                                         00DB   204 ;
                                         00DB   205 ; main loop of interpreter
                                         00DB   206 ;
                                         00DB   207 fetch_next:
                                         00DB   208 ;
                                         00DB   209 ; interpret edit program
                                         00DB   210 ;
                52    8A   9A  00DB   211          movzbl  (r10)+,r2               ; get opcode
                50    8A   9A  00DE   212          movzbl  (r10)+,r0               ; get argument
                                         00E1   213          case    r2,<-
                                         00E1   214                  move_zero_supress,-
                                         00E1   215                  insert_character,-
                                         00E1   216                  set_fill_character,-
                                         00E1   217                  insert_significant,-
                                         00E1   218                  move_digits,-
                                         00E1   219                  insert_minus,-
                                         00E1   220                  insert_plus,-
                                         00E1   221                  insert_sign,-
                                         00E1   222                  set_float_character,-
                                         00E1   223                  set_float_minus,-
                                         00E1   224                  set_float_plus,-
                                         00E1   225                  set_float_sign,-
                                         00E1   226                  skip_if_zero,-
                                         00E1   227                  fill_field,-
                                         00E1   228                  set_significance,-
                                         00E1   229                  end_edit,-
                                         00E1   230                  supress_digit,-
                                         00E1   231                  move_digit_minus,-
                                         00E1   232                  move_digit_plus,-
                                         00E1   233                  move_digit_sign-
                                         00E1   234                  >
```

```
                                 010D   235
                                 010D   236 error:
                                 010D   237 ;
                00000000'8F   DD 010D   238         pushl   #pli$_cnverr            ; conversion error
                          7E  D4 0113   239         clrl    -(sp)                   ; signal error
                00000000'8F   DD 0115   240         pushl   #pli$_error             ;
            00000000'GF   03  FB 0118   241         calls   #3,g^lib$signal         ;
                              04 0122   242         ret
                                 0123   243 ;
                                 0123   244 ; end edit
                                 0123   245 ;
                                 0123   246 end_edit:
                        53    5E C2 0123   247         subl    sp,r3                   ; calc size of return string
 14 BC   10 AC   20     6E    53 2C 0126   248         movc5   r3,(sp),#32,target_size(ap),@target_address(ap); copy it
                              04 012E   249         ret                             ; done
```

```
                                012F   251          .sbttl  edit interpret routines
                                012F   252 ;
                                012F   253 ; zero_supress move
                                012F   254 ;
                                012F   255 move_zero_supress:
      1E FA AD      E8          012F   256          blbs    significance(fp),move_character; br if significance on
       30    61     91          0133   257 10$:     cmpb    (r1),#^a/0/             ; zero digit?
             0C     12          0136   258          bneq    15$                     ; if neq then insert it
             51     D6          0138   259          incl    r1                      ; pass zero digit
   83     F9 AD     90          013A   260          movb    fill(fp),(r3)+          ; insert fill character
          F2 50     F5          013E   261          sobgtr  r0,10$                  ; continue until done
           FF97     31          0141   262          brw     fetch_next              ;
   FA AD     01     88          0144   263 15$:     bisb    #1,significance(fp)     ; turn on significance
         FB AD     95          0148   264          tstb    float(fp)               ; float byte defined?
             04     13          014B   265          beql    move_character          ; br if no
   83    FB AD     90          014D   266          movb    float(fp),(r3)+         ; insert floab byte
                                0151   267 ;
                                0151   268 ; move characters
                                0151   269 ;
                                0151   270 move_character:
63    61    50     28          0151   271          movc3   r0,(r1),(r3)            ; move characters to output
            FF83    31          0155   272          brw     fetch_next              ;
                                0158   273 ;
                                0158   274 ; insert_character
                                0158   275 ;
                                0158   276 insert_character:
         83    50     90          0158   277          movb    r0,(r3)+                ; insert character
            FF7D    31          015B   278          brw     fetch_next              ;
                                015E   279 ;
                                015E   280 ; set_fill_character
                                015E   281 ;
                                015E   282 set_fill_character:
   F9 AD     50     90          015E   283          movb    r0,fill(fp)             ;
          FF76     31          0162   284          brw     fetch_next              ;
                                0165   285 ;
                                0165   286 ; significant_insert
                                0165   287 ;
                                0165   288 insert_significant:
     04 FA AD      E8          0165   289          blbs    significance(fp),10$    ; br if significance on
50     F9 AD     90          0169   290          movb    fill(fp),r0             ; get fill character
   83    50     90          016D   291 10$:     movb    r0,(r3)+                ; insert character
          FF68     31          0170   292          brw     fetch_next              ;
                                0173   293 ;
                                0173   294 ; move_digits
                                0173   295 ;
                                0173   296 move_digits:
     0C FA AD      E8          0173   297          blbs    significance(fp),10$    ; br if significance is on
52     FB AD     90          0177   298          movb    float(fp),r2            ; get float byte
             03     13          017B   299          beql    5$                      ; br if not defined
   83    52     90          017D   300          movb    r2,(r3)+                ; insert float byte
         FA AD     96          0180   301 5$:      incb    significance(fp)        ; set significance on
             CC     11          0183   302 10$:     brb     move_character          ; continue in common
                                0185   303 ;
                                0185   304 ; insert minus
                                0185   305 ;
                                0185   306 insert_minus:
2D     FC AD     91          0185   307          cmpb    sign(fp),#^a/-/         ; negative
```

```
          04   13 0189          beql    10$                     ; if eql then yes
    50 F9 AD   90 018B          movb    fill(fp),r0             ; insert blank or star
       83 50   90 018F 10$:     movb    r0,(r3)+                ; insert character if yes
        FF46   31 0192          brw     fetch_next              ; continue
                  0195 ;
                  0195 ; insert plus
                  0195 ;
                  0195 insert_plus:
    2B FC AD   91 0195          cmpb    sign(fp),#^a/+/         ; positive?
          04   13 0199          beql    10$                     ; if eql then yes
    50 F9 AD   90 019B          movb    fill(fp),r0             ; insert blankor fill
       83 50   90 019F 10$:     movb    r0,(r3)+                ; insert character if yes
        FF36   31 01A2          brw     fetch_next              ; continue
                  01A5 ;
                  01A5 ; insert sign
                  01A5 ;
                  01A5 insert_sign:
    83 FC AD   90 01A5          movb    sign(fp),(r3)+          ; insert sign byte
        FF2F   31 01A9          brw     fetch_next              ;
                  01AC ;
                  01AC ; set float character
                  01AC ;
                  01AC set_float_character:
    FB AD 50   90 01AC          movb    r0,float(fp)            ;
        FF28   31 01B0          brw     fetch_next              ;
                  01B3 ;
                  01B3 ; set float minus
                  01B3 ;
                  01B3 set_float_minus:
    2D FC AD   91 01B3          cmpb    sign(fp),#^a/-/         ; negative?
          07   12 01B7          bneq    10$                     ; if neq then no
    FB AD 50   90 01B9          movb    r0,float(fp)            ; set float if true
        FF1B   31 01BD          brw     fetch_next              ;
 FB AD F9 AD   90 01C0 10$:     movb    fill(fp),float(fp)      ; save as fill character
        FF13   31 01C5          brw     fetch_next
                  01C8 ;
                  01C8 ; set float plus
                  01C8 ;
                  01C8 set_float_plus:
    2B FC AD   91 01C8          cmpb    sign(fp),#^a/+/         ; positive
          07   12 01CC          bneq    10$                     ; if neq then no
    FB AD 50   90 01CE          movb    r0,float(fp)            ; set float if true
        FF06   31 01D2          brw     fetch_next              ;
 FB AD F9 AD   90 01D5 10$:     movb    fill(fp),float(fp)      ;
        FEFE   31 01DA          brw     fetch_next
                  01DD ;
                  01DD ; set float sign
                  01DD ;
                  01DD set_float_sign:
 FB AD FC AD   90 01DD          movb    sign(fp),float(fp)      ;
        FEF6   31 01E2          brw     fetch_next              ;
                  01E5 ;
                  01E5 ; skip_if_zero
                  01E5 ;
                  01E5 skip_if_zero:
       F8 AD   95 01E5          tstb    zero_indic(fp)          ; zero?
          04   12 01E8          bneq    10$                     ; if neq then not zero
```

```
                 5A   6A40   3E  01EA   365          movaw   (r10)[r0],r10           ; set new edit pc
                      FEEA   31  01EE   366 10$:     brw     fetch_next              ;
                             01F1   367 ;
                             01F1   368 ; fill_field
                             01F1   369 ;
                             01F1   370 fill_field:
    63   50   F9 AD   63   00   2C  01F1   371          movc5   #0,(r3),fill(fp),r0,(r3);
                      FEE0   31  01F8   372          brw     fetch_next              ;
                             01FB   373 ;
                             01FB   374 ; set_significance
                             01FB   375 ;
                             01FB   376 set_significance:
                   0D FA AD   E8  01FB   377          blbs    significance(fp),10$    ; br if significance is on
                52    FB AD   90  01FF   378          movb    float(fp),r2            ; get float byte
                         03   13  0203   379          beql    5$                      ; br if not defined
                      83   52   90  0205   380          movb    r2,(r3)+                ; insert float byte
                   FA AD   01   88  0208   381 5$:      bisb    #1,significance(fp)     ; turn on significance
                      FECC   31  020C   382 10$:     brw     fetch_next              ;
                             020F   383
                             020F   384 ;
                             020F   385 ; supress_digit
                             020F   386 ;
                             020F   387 supress_digit:
                      50   81   90  020F   388          movb    (r1)+,r0                ; get next source digit
                      30   50   91  0212   389          cmpb    r0,#^a/0/               ; zero?
                         03   12  0215   390          bneq    10$                     ; if neq then no
                      50   20   90  0217   391          movb    #^a/ /,r0               ; insert blank
                      83   50   90  021A   392 10$:     movb    r0,(r3)+                ; move character
                      FEBB   31  021D   393          brw     fetch_next              ;
                             0220   394 ;
                             0220   395 ; move_digit_minus
                             0220   396 ;
                             0220   397 move_digit_minus:
                      50   81   9A  0220   398          movzbl  (r1)+,r0                ; get next source digit
                   2D    FC AD   91  0223   399          cmpb    sign(fp),#^a/-/         ; negative source?
                         09   12  0227   400          bneq    10$                     ; if neq then no
                      50   30   82  0229   401          subb    #^a/0/,r0               ;
                50    FDD9 CF40   90  022C   402          movb    w^minus_over_punch[r0],r0; get new character
                      83   50   90  0232   403 10$:     movb    r0,(r3)+                ; insert character
                      FEA3   31  0235   404          brw     fetch_next              ;
                             0238   405 ;
                             0238   406 ; move_digit_plus
                             0238   407 ;
                             0238   408 move_digit_plus:
                      50   81   9A  0238   409          movzbl  (r1)+,r0                ; get next source character
                   2B    FC AD   91  023B   410          cmpb    sign(fp),#^a/+/         ; positive?
                         09   12  023F   411          bneq    10$                     ; if neq then no
                      50   30   82  0241   412          subb    #^a/0/,r0               ;
                50    FDB7 CF40   90  0244   413          movb    w^plus_over_punch[r0],r0; get new character
                      83   50   90  024A   414 10$:     movb    r0,(r3)+                ; insert new character
                      FE8B   31  024D   415          brw     fetch_next              ;
                             0250   416 ;
                             0250   417 ; move_digit_sign
                             0250   418 ;
                             0250   419 move_digit_sign:
                52    FDB6 CF   9E  0250   420          movab   w^minus_over_punch,r2   ; address minus set
                   2D    FC AD   91  0255   421          cmpb    sign(fp),#^a7-/         ; negative?
```

```
                05   13  0259   422          beql    10$                         ; if eql then yes
          52  FDA1 CF   9E  025B   423          movab   w^plus_over_punch,r2       ; address positive set
              50   81   9A  0260   424 10$:     movzbl  (r1)+,r0                    ; get source character
              50   30   82  0263   425          subb    #^a/0/,r0                   ;
           83   6240    90  0266   426          movb    (r2)[r0],(r3)+              ; insert new character
                FE6E    31  026A   427          brw     fetch_next                  ;
```

```
                                026D   429          .sbttl  pli$cvt_fr_pic - convert picture to numeric
                                026D   430 ;++
                                026D   431 ; pli$cvt_fr_pic - convert picture to numeric
                                026D   432 ;
                                026D   433 ; functional description:
                                026D   434 ;
                                026D   435 ; This routine converts a picture character string described by 8(ap) and 12(ap)
                                026D   436 ; to a numeric value described by 16(ap) and 20(ap) based on the picture constant
                                026D   437 ; addressed by picture_constant(ap).
                                026D   438 ;
                                026D   439 ; inputs:
                                026D   440 ;
                                026D   441 ;       0(ap) = 5
                                026D   442 ;       4(ap) = picture_constant address
                                026D   443 ;       8(ap) = source size
                                026D   444 ;       12(ap) = source address
                                026D   445 ;       16(ap) = target size
                                026D   446 ;       20(ap) = target address
                                026D   447 ;
                                026D   448 ; outputs:
                                026D   449 ;
                                026D   450 ;       target string is filled in.
                                026D   451 ;
                                026D   452 ; ERROR maybe signalled.
                                026D   453 ;
                                026D   454 ;--
                        CFFC    026D   455          .entry  pli$cvt_fr_pic,^m<iv,dv,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
                     5E   28    C2   026F   456          subl    #cvt_fr_pic_stack,sp    ; allocate stack space
                        FC AD    94   0272   457          clrb    found_sign(fp)          ; set no sign found
                     5B   04 AC    D0   0275   458          movl    picture_constant(ap),r11; address picture constant
                                0279   459 ;
                                0279   460 ; calc size of source string
                                0279   461 ;
                           56   04 AB    9A   0279   462          movzbl  pic$b_byte_size(r11),r6 ; get picture designate size
                           08 AC   56    B1   027D   463          cmpw    r6,source_size(ap)      ; less or greater than source?
                                   04    15   0281   464          bleq    10$                     ; if leq then use it
                           56   08 AC    3C   0283   465          movzwl  source_size(ap),r6      ; use smaller size
                           57   0C AC    D0   0287   466 10$:     movl    source_address(ap),r7   ; get source address
                              5E   56    C2   028B   467          subl    r6,sp                   ; allocate space for ascii text
                              58   6E    9E   028E   468          movab   (sp),r8                 ; copy address of space
                                0291   469 ;
                                0291   470 ; set result to zero
                                0291   471 ;
                           50   10 AC    9A   0291   472          movzbl  target_size(ap),r0      ; get target size p value
50   00    FD79 CF    01      00    F8   0295   473          ashp    #0,#1,w^packed_zero,#0,r0,@target_address(ap);
                                14 BC        029D
                                029F   474 ;
                                029F   475 ; loop through string, extracting digits and picking up sign
                                029F   476 ;
                                029F   477 locate_char:
                           53   87    9A   029F   478          movzbl  (r7)+,r3                ; get character
                    FD6D CF    2A    53    3A   02A2   479          locc    r3,#valid_char_size,w^valid_char_table; locate character in valid  t
                                   4A    12   02A8   480          bneq    select_action           ; if neq then valid character found
                                FE60     31   02AA   481          brw     error                   ; signal error
                                02AD   482 ;
                                02AD   483 ;
                                02AD   484 ; get next character
```

```
                                02AD    485 ;
                                02AD    486 next_character:
                     EF 56   F5 02AD    487         sobgtr  r6,locate_char          ; continue in more to scan
                                02B0    488 ;
                                02B0    489 ; converet number to numeric
                                02B0    490 ;
                                02B0    491 ; setup default sign based on presence of '+' or I format
                                02B0    492 ;
                        FC AD   95 02B0    493         tstb    found_sign(fp)          ; sign found?
                           0D   12 02B3    494         bneq    15$                     ; if neq then yes
                     FC AD    2B 90 02B5    495         movb    #^a/+/,found_sign(fp)   ; assume positive
                  04 05 AB    00 E1 02B9    496         bbc     #pic_v_minus,pic$b_flags(r11),15$; br if not negative default
                     FC AD    2D 90 02BE    497         movb    #^a/-/,found_sign(fp)   ;
                                02C2    498 15$:
                   50   58    5E C3 02C2    499         subl3   sp,r8,r0                ; get size of character string
                     7E   FC AD 90 02C6    500         movb    found_sign(fp),-(sp)    ; insert sign at front of buffer
                             27 13 02CA    501         beql    100$                    ; if eql then answer is zero
                        1F   50 D1 02CC    502         cmpl    r0,#31                  ; more than maximum?
                             03 15 02CF    503         bleq    10$                     ; if leq then ok
                        50   1F D0 02D1    504         movl    #31,r0                  ; convert maximum
                D8 AD   1F   6E 50 09 02D4    505 10$:    cvtsp   r0,(sp),#31,inter_result(fp); convert to packed
                                02DA    506 ;
                                02DA    507 ; scale intermediate result to requested precision
                                02DA    508 ;
                        50   11 AC 9A 02DA    509         movzbl  target_size+1(ap),r0    ;
                        51   01 AB 9A 02DE    510         movzbl  pic$w_pq+1(r11),r1      ;
                     51   50   51 C3 02E2    511         subl3   r1,r0,r1                ; calc shift count
                        50   10 AC 9A 02E6    512         movzbl  target_size(ap),r0      ; get prec of target
           50   00   D8 AD   1F   51 F8 02EA    513         ashp    r1,#31,inter_result(fp),#0,r0,@target_address(ap);
                           14 BC       02F1
                                04 02F3    514 100$:   ret                             ; done
                                02F4    515 ;
                                02F4    516 ; select action based on character type
                                02F4    517 ;
                                02F4    518 select_action:
                                02F4    519         case    r0,<-                   ; case on character location in table
                                02F4    520                 error,-                 ; zero is bad case
                                02F4    521                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    522                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    523                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    524                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    525                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    526                 pass_nega_digit,-       ; p  overpunched digit
                                02F4    527                 pass_nega_digit,-       ; p  overpunched digit
                                02F4    528                 pass_nega_digit,-       ; pa overpunched digit
                                02F4    529                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    530                 pass_nega_digit,-       ; pass overpunched digit
                                02F4    531                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    532                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    533                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    534                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    535                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    536                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    537                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    538                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    539                 pass_pos_digit,-        ; pass overpunched digit
                                02F4    540                 pass_pos_digit,-        ; pass overpunched digit
```

```
                     02F4   541                     next_character,-        ; skip star
                     02F4   542                     next_character,-        ; skip space
                     02F4   543                     db_test,-               ; test for db
                     02F4   544                     db_test,-               ; test for db
                     02F4   545                     cr_test,-               ; test for cr
                     02F4   546                     cr_test,-               ; test for cr
                     02F4   547                     next_character,-        ; skip $
                     02F4   548                     next_character,-        ; skip ,
                     02F4   549                     next_character,-        ; skip .
                     02F4   550                     next_character,-        ; skip /
                     02F4   551                     pass_sign,-             ; found -
                     02F4   552                     pass_sign,-             ; found +
                     02F4   553                     pass_digit,-            ; move normal digit
                     02F4   554                     pass_digit,-            ; move normal digit
                     02F4   555                     pass_digit,-            ; move normal digit
                     02F4   556                     pass_digit,-            ; move normal digit
                     02F4   557                     pass_digit,-            ; move normal digit
                     02F4   558                     pass_digit,-            ; move normal digit
                     02F4   559                     pass_digit,-            ; move normal digit
                     02F4   560                     pass_digit,-            ; move normal digit
                     02F4   561                     pass_digit,-            ; move normal digit
                     02F4   562                     pass_digit,-            ; move normal digit
                     02F4   563                     pass_digit>             ; move normal digit
                     0350   564 ;
                     0350   565 ; case subroutines
                     0350   566 ;
                     0350   567 pass_digit:
                88    53   90  0350   568             movb    r3,(r8)+                ; pass digit
                   FF57   31  0353   569             brw     next_character          ;
                     0356   570 pass_sign:
                  FC AD   95  0356   571             tstb    found_sign(fp)          ; sign found already?
                     07   12  0359   572             bneq    10$                     ; if neq then error
             FC AD   53   90  035B   573             movb    r3,found_sign(fp)       ; save sign character
                   FF4B   31  035F   574             brw     next_character          ;
                   FDA8   31  0362   575 10$:        brw     error                   ; signal error
                     0365   576
                     0365   577             .enabl  lsb
                     0365   578 db_test:
                56    02   D1  0365   579             cmpl    #2,r6                   ; one character left?
                     1F   12  0368   580             bneq    5$                      ; if neq then must be digit
             62 8F   67   91  036A   581             cmpb    (r7),#^a/b/             ; lower b?
                     1B   13  036E   582             beql    10$                     ; if eql then ok
             42 8F   67   91  0370   583             cmpb    (r7),#^a/B/             ; upper b?
                     15   13  0374   584             beql    10$                     ; if eql then yes
                     11   11  0376   585             brb     5$                      ; treat as positive digit
                     0378   586 cr_test:
                56    02   D1  0378   587             cmpl    #2,r6                   ; one character left?
                     0C   12  037B   588             bneq    5$                      ; if neq then must be digit
             72 8F   67   91  037D   589             cmpb    (r7),#^a/r/             ; lower r?
                     08   13  0381   590             beql    10$                     ; if eql then ok
             52 8F   67   91  0383   591             cmpb    (r7),#^a/R/             ; upper r?
                     02   13  0387   592             beql    10$                     ; if eql then ok
                     0E   11  0389   593 5$:         brb     pass_pos_digit          ; pass positive overpunch digit
                     57   D6  038B   594 10$:        incl    r7                      ; pass second character
                     56   D7  038D   595             decl    r6                      ; sount the character
             FC AD   2D   90  038F   596             movb    #^a/-/,found_sign(fp)   ; set sign
                   FF17   31  0393   597             brw     next_character          ; try next character
```

```
                FD74   31  0396   598 20$:    brw     error                   ; not valid string
                           0399   599
                           0399   600         .dsabl  lsb
                           0399   601 pass_pos_digit:
              FC AD    95  0399   602         tstb    found_sign(fp)          ; sign character seen?
                 13    12  039C   603         bneq    10$                     ; if neq then no
        05 AB    06    93  039E   604         bitb    #pic_m_t_format!pic_m_i_format,pic$b_flags(r11); legal in set?
                 0D    13  03A2   605         beql    10$                     ; if eql then no
        FC AD    2B    90  03A4   606         movb    #^a/+/,found_sign(fp)   ; set sign character
    88   FC92 CF40    90  03A8   607         movb    over_punch_value[r0],(r8)+; insert character based on table index
              FEFC    31  03AE   608         brw     next_character          ;
              FD59    31  03B1   609 10$:     brw     error                   ; signal error
                      03B4   610 pass_nega_digit:
              FC AD    95  03B4   611         tstb    found_sign(fp)          ; sign character seen?
                 13    12  03B7   612         bneq    10$                     ; if neq then no
        05 AB    0A    93  03B9   613         bitb    #pic_m_t_format!pic_m_r_format,pic$b_flags(r11); legal in set?
                 0D    13  03BD   614         beql    10$                     ; if eql then no
        FC AD    2D    90  03BF   615         movb    #^a/-/,found_sign(fp)   ; set sign character
    88   FC77 CF40    90  03C3   616         movb    over_punch_value[r0],(r8)+; insert character based on table index
              FEE1    31  03C9   617         brw     next_character          ;
              FD3E    31  03CC   618 10$:     brw     error                   ; signal error
```

```
                                03CF    620          .sbttl  pli$valid_pic - validate picture value
                                03CF    621 ;++
                                03CF    622 ; pli$valid_pic - validate picture value
                                03CF    623 ;
                                03CF    624 ; functional description:
                                03CF    625 ;
                                03CF    626 ; This routine is used by the valid-bif and EDIT I/O to validate picture
                                03CF    627 ; values.
                                03CF    628 ;
                                03CF    629 ; inputs:
                                03CF    630 ;
                                03CF    631 ;       0(ap) = 3
                                03CF    632 ;       4(ap) = picture constant address
                                03CF    633 ;       8(ap) = size of the test string
                                03CF    634 ;       12(ap) = address of the test string
                                03CF    635 ;
                                03CF    636 ; outputs:
                                03CF    637 ;
                                03CF    638 ;       r0 = validity indicator
                                03CF    639 ;
                                03CF    640 ; ERROR maybe signalled.
                                03CF    641 ;--
                          007C  03CF    642          .entry  pli$valid_pic,^m<r2,r3,r4,r5,r6>
                    56    04 AC   D0    03D1    643          movl    4(ap),r6                ; address picture constant
                       5E    1F   C2    03D5    644          subl    #31,sp                  ; allocate enough space for convert
                       55    5E   D0    03D8    645          movl    sp,r5                   ; copy address of target string
                             55   DD    03DB    646          pushl   r5                      ; convert to numeric
                       7E    66   3C    03DD    647          movzwl  pic$w_pq(r6),-(sp)      ; target p,q
                         0C AC   DD    03E0    648          pushl   12(ap)                  ; pass source address
                         08 AC   DD    03E3    649          pushl   8(ap)                   ; pass source size
                             56   DD    03E6    650          pushl   r6                      ; pass constant address
                    FE80 CF      05    FB    03E8    651          calls   #5,w^pli$cvt_fr_pic     ; convert to numeric
                      54    04   A6    9A    03ED    652          movzbl  pic$b_byte_size(r6),r4  ; get size of result
                          5E   54    C2    03F1    653          subl    r4,sp                   ; allocate space
                          53   5E    D0    03F4    654          movl    sp,r3                   ; copy result address
                               53    DD    03F7    655          pushl   r3                      ; convert to picture
                               54    DD    03F9    656          pushl   r4                      ;
                               55    DD    03FB    657          pushl   r5                      ;
                          52   66    3C    03FD    658          movzwl  pic$w_pq(r6),r2         ;
                               52    DD    0400    659          pushl   r2                      ;
                               56    DD    0402    660          pushl   r6                      ;
                    FC51 CF      05    FB    0404    661          calls   #5,w^pli$cvt_to_pic     ;
OC BC    08 AC      20   63      54    2D    0409    662          cmpc5   r4,(r3),#32,8(ap),a12(ap); compare strings
                               04    12    0411    663          bneq    10$                     ; if neq then continue
                          50   01    D0    0413    664          movl    #1,r0                   ; set success
                               04    0416    665          ret
                               50    D4    0417    666 10$:     clrl    r0                      ; set failure
                               04    0419    667          ret
                                     041A    668          .end
```

```
CR_TEST                         00000378 R      02
CVT_FR_PIC_STACK              = 00000028
CVT_TO_PIC_STACK              = 00000009
DB_TEST                         00000365 R      02
END_EDIT                        00000123 R      02
ERROR                           0000010D R      02
FETCH_NEXT                      000000DB R      02
FILL                          = FFFFFFF9
FILL_FIELD                      000001F1 R      02
FLOAT                         = FFFFFFFB
FOUND_SIGN                    = FFFFFFFC
INSERT_CHARACTER                00000158 R      02
INSERT_MINUS                    00000185 R      02
INSERT_PLUS                     00000195 R      02
INSERT_SIGN                     000001A5 R      02
INSERT_SIGNIFICANT              00000165 R      02
INTER_RESULT                  = FFFFFFD8
LIB$SIGNAL                      ********   X    02
LOCATE_CHAR                     0000029F R      02
MINUS_OVER_PUNCH                0000000A R      02
MOVE_CHARACTER                  00000151 R      02
MOVE_DIGITS                     00000173 R      02
MOVE_DIGIT_MINUS                00000220 R      02
MOVE_DIGIT_PLUS                 00000238 R      02
MOVE_DIGIT_SIGN                 00000250 R      02
MOVE_ZERO_SUPRESS               0000012F R      02
NEXT_CHARACTER                  000002AD R      02
OVER_PUNCH_VALUE                0000003F R      02
PACKED_ZERO                     00000014 R      02
PASS_DIGIT                      00000350 R      02
PASS_NEGA_DIGIT                 000003B4 R      02
PASS_POS_DIGIT                  00000399 R      02
PASS_SIGN                       00000356 R      02
PIC$B_BYTE_SIZE               = 00000004
PIC$B_FLAGS                   = 00000005
PIC$B_LANGUAGE                = 00000006
PIC$B_PROGRAM                 = 00000008
PIC$W_PQ                      = 00000000
PICTURE_CONSTANT              = 00000004
PIC_M_I_FORMAT                = 00000004
PIC_M_R_FORMAT                = 00000008
PIC_M_T_FORMAT                = 00000002
PIC_V_HAS_SIGN                = 00000004
PIC_V_MINUS                   = 00000000
PLI$CVT_FR_PIC                  0000026D RG     02
PLI$CVT_TO_PIC                  0000005A RG     02
PLI$VALID_PIC                   000003CF RG     02
PLI$_CNVERR                     ********   X    02
PLI$_ERROR                      ********   X    02
PLUS_OVER_PUNCH                 00000000 R      02
SELECT_ACTION                   000002F4 R      02
SET_FILL_CHARACTER              0000015E R      02
SET_FLOAT_CHARACTER             000001AC R      02
SET_FLOAT_MINUS                 000001B3 R      02
SET_FLOAT_PLUS                  000001C8 R      02
SET_FLOAT_SIGN                  000001DD R      02
SET_SIGNIFICANCE                000001FB R      02
SIGN                          = FFFFFFFC
SIGNIFICANCE                  = FFFFFFFA
SIZ...                        = 00000001
SKIP_IF_ZERO                    000001E5 R      02
SOURCE_ADDRESS                = 0000000C
SOURCE_SIZE                   = 00000008
SUPRESS_DIGIT                   0000020F R      02
TARGET_ADDRESS                = 00000014
TARGET_SIZE                   = 00000010
VALID_CHAR_SIZE               = 0000002A
VALID_CHAR_TABLE                00000015 R      02
ZERO_INDIC                    = FFFFFFF8
```

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
_PLI$CODE	0000041A (1050.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	10	00:00:00.09	00:00:01.10
Command processing	70	00:00:00.50	00:00:04.38
Pass 1	112	00:00:02.87	00:00:10.30
Symbol table sort	0	00:00:00.06	00:00:00.07
Pass 2	112	00:00:01.33	00:00:03.75
Symbol table output	9	00:00:00.08	00:00:00.08
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	315	00:00:04.96	00:00:19.70

The working set limit was 1050 pages.
16018 bytes (32 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 85 non-local and 35 local symbols.
668 source lines were read in Pass 1, producing 21 object records in Pass 2.
11 pages of virtual memory were used to define 10 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[PLIRTL.OBJ]PLIRTMAC.MLB;1	3
_255DUA28:[SYSLIB]STARLET.MLB;2	4
TOTALS (all libraries)	7

94 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

```
MACRO/ENABLE=SUPPRESSION/DISABLE=TRACEBACK/LIS=LIS$:PLICVTPIC/OBJ=OBJ$:PLICVTPIC MSRC$:PLICVTPIC/UPDATE=(ENH$:PLICVTPIC)+LIB$:PLIRTM
```

PLIDELETE
LIS

PLICONVRT
LIS

PLIDATA
LIS

PLICONTRL
LIS

PLIDATE
LIS

PLICVTPIC
LIS

PLIENVIR
LIS