```
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    PPPPPPPP    KK      KK  DDDDDDDD      IIIIII    VV      VV  LL
PPPPPPPP    LL            IIIIII    PPPPPPPP    KK      KK  DDDDDDDD      IIIIII    VV      VV  LL
PP      PP  LL              II      PP      PP  KK      KK  DD      DD      II      VV      VV  LL
PP      PP  LL              II      PP      PP  KK      KK  DD      DD      II      VV      VV  LL
PP      PP  LL              II      PP      PP  KK    KK    DD      DD      II      VV      VV  LL
PP      PP  LL              II      PP      PP  KK    KK    DD      DD      II      VV      VV  LL
PPPPPPPP    LL              II      PPPPPPPP    KKKKKK      DD      DD      II      VV      VV  LL
PPPPPPPP    LL              II      PPPPPPPP    KKKKKK      DD      DD      II      VV      VV  LL
PP          LL              II      PP          KK    KK    DD      DD      II      VV      VV  LL
PP          LL              II      PP          KK    KK    DD      DD      II      VV      VV  LL
PP          LL              II      PP          KK      KK  DD      DD      II        VV  VV    LL              ....
PP          LL              II      PP          KK      KK  DD      DD      II        VV  VV    LL              ....
PP          LLLLLLLLLL    IIIIII    PP          KK      KK  DDDDDDDD      IIIIII      VV      LLLLLLLLLL      ....
PP          LLLLLLLLLL    IIIIII    PP          KK      KK  DDDDDDDD      IIIIII      VV      LLLLLLLLLL      ....


LL              IIIIII      SSSSSSSS
LL              IIIIII    SSSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0000     1          .title  pli$div_pk_long
0000     2          .ident  /1-002/                                ; Edit WHM1002
0000     3  ;
0000     4  ;*****************************************************************************
0000     5  ;*                                                                           *
0000     6  ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     7  ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     8  ;*  ALL RIGHTS RESERVED.                                                     *
0000     9  ;*                                                                           *
0000    10  ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
0000    11  ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000    12  ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
0000    13  ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
0000    14  ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
0000    15  ;*  TRANSFERRED.                                                             *
0000    16  ;*                                                                           *
0000    17  ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
0000    18  ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000    19  ;*  CORPORATION.                                                             *
0000    20  ;*                                                                           *
0000    21  ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
0000    22  ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    23  ;*                                                                           *
0000    24  ;*                                                                           *
0000    25  ;*****************************************************************************
0000    26  ;
0000    27  ;++
0000    28  ;
0000    29  ;
0000    30  ; routine:
0000    31  ;
0000    32  ;       PLI$DIV_PK_LONG
0000    33  ;
0000    34  ;
0000    35  ; facility:
0000    36  ;
0000    37  ;       VAX/VMS PL1 runtime library.
0000    38  ;
0000    39  ; abstract:
0000    40  ;
0000    41  ;       Runtime routine performs fixed decimal (packed decimal) division.
0000    42  ;       The routine is called  when precision and scale  requirements for
0000    43  ;       the quotient imply multiple  precision  division.  The routine is
0000    44  ;       only called when such multiple precision division is required and
0000    45  ;       when the divisor has a precision of  30 or 31 decimal digits.
0000    46  ;       (Call   pli$div_pkshort   if   multiple   precision  division  is
0000    47  ;       required and the divisor has precision less than 30 decimal digits).
0000    48  ;
0000    49  ; author: Peter Baum  30-jun-1980
0000    50  ;
0000    51  ; modifications:
0000    52  ;
0000    53  ;
0000    54  ;       1-002   Bill Matthews   29-September-1982
0000    55  ;
0000    56  ;               Invoke macros $defdat and rtshare instead of $defopr and share.
0000    57  ;
```

```
0000     58 ;
0000     59 ;
0000     60 ;
0000     61 ;
0000     62 ; documentation file: [pl1.doc.codegen]THEORY.MEM
0000     63 ;
0000     64 ; functional description:
0000     65 ;
0000     66 ;       This routine calculates:
0000     67 ;
0000     68 ;       z = x / y
0000     69 ;
0000     70 ;       let a = scale(z) + scale(y) - scale(x)  - 31 + prec(x)
0000     71 ;           b = scale(z) + scale(y) - scale(x) + prec(x)
0000     72 ;           c = 31 - prec(x)
0000     73 ;           d = 31 - prec(y)
0000     74 ;
0000     75 ;       this routine is called if b > 31 and d < 2
0000     76 ;
0000     77 ;       Prior to the call:
0000     78 ;               if c not 0 then shift x left by c.
0000     79 ;               Thus x is a 31 digit packed decimal.
0000     80 ;
0000     81 ;
0000     82 ;
0000     83 ;
0000     84 ;       input:
0000     85 ;               0(ap)   # of arguments
0000     86 ;               4(ap)   address of dividend (shifted left by c)
0000     87 ;               8(ap)   address of divisor
0000     88 ;               12(ap)  precision of divisor (high order bytes zeroed)
0000     89 ;               16(ap)  address of quotient
0000     90 ;               20(ap)  precision of quotient (high order bytes zeroed)
0000     91 ;               24(ap)  a as defined above(high order bytes zeroed)
0000     92 ;
0000     93 ;
0000     94 ;       output:
0000     95 ;               quotient returned at address specified by 16(ap)
0000     96 ;
0000     97 ;
0000     98 ; optimization notes:
0000     99 ;
0000    100 ;       1) Optimized for speed, not space.
0000    101 ;       2) Optimized for y > 0.
0000    102 ;       3) Assumes speed for register to register operations are the same
0000    103 ;          for byte operations and longword operations.
0000    104 ;       4) Many  packed  instruction  sequences were timed. Do not change
0000    105 ;          unless  actual  tests  are  made  to determine relative speed.
0000    106 ;          Tests were made on 11/780 and Comet.
0000    107 ;
0000    108 ;
0000    109 ;
0000    110 ; possible optimizations:
0000    111 ;
0000    112 ;       1) currently  we  always calculate the next 15 digits each
0000    113 ;       iteration  and then truncate the last iteration as part of
0000    114 ;       the  final  step.   We  might be able to go through making
```

```
0000   115 ;       calculations with fewer digits on this last pass.
0000   116 ;
0000   117 ;       2) the classical trade-off involving flow paths which
0000   118 ;       could be merged (sometimes resulting in things like
0000   119 ;       adding 0) or merged with switches or left alone (resulting
0000   120 ;       in space-speed tradeoff in favor of space). Since we are
0000   121 ;       running on a virtual machine, even the speed factor is not
0000   122 ;       obvious. This came up when remainder was 0, when borrow
0000   123 ;       was not needed, when less than 15 digits where required
0000   124 ;       for the last part of the quotient, and when the new
0000   125 ;       algorithm had found precisely what the next 15 digits
0000   126 ;       were but did not have a value yet for the remainder (the
0000   127 ;       other algorithms always have a remainder when the quotient
0000   128 ;       is known).
0000   129 ;
0000   130 ;       3) New algorithm - special case y2=0
0000   131 ;
0000   132 ;       4) New algorithm - optimize branches according to probable
0000   133 ;       sign of R(H).
0000   134 ;
0000   135 ;
0000   136 ;
0000   137 ;
0000   138 ; variable use:
0000   139 ;
0000   140 ;                size
0000   141 ;                in
0000   142 ;     variable   digits          use
0000   143 ;     ---------  ------   -------------------------
0000   144 ;        y1        15     High order digits of divisor.
0000   145 ;        y2        16     Low order digits of divisor.
0000   146 ;        x         31     Initially dividend, thereafter
0000   147 ;                         remainders of successive divide
0000   148 ;                         operations
0000   149 ;        z       @y(ap)   Quotient.
0000   150 ;        z2        31     Temporarily holds trial low
0000   151 ;                         order digits of quotient.
0000   152 ;        t1        31     High order digits of the remainder.
0000   153 ;        t2        31     Holds the 15 low order digits of the
0000   154 ;                         46 digit remainder. 31 digits for
0000   155 ;                         possible later changes.
0000   156 ;        t3        16     Holds the low order digits of the
0000   157 ;                         remainder.
0000   158 ;        t4        31     Temporary used because packed instructions can
0000   159 ;                         not overlap their operands.
0000   160 ;
0000   161 ;
0000   162 ;
0000   163 ;
0000   164 ;
0000   165 ;
0000   166 ; register usage:
0000   167 ;
0000   168 ;    register                  use
0000   169 ;    --------   ------------------------------------------------
0000   170 ;        r6      a = additional digits of precision required
0000   171 ;        r7      stky(sp) which holds a copy of divisor
```

```
                        ;       r8      py(ap) = precision of y
                        ;       r9      r = number of additional digits of the quotient
                        ;               that are to be found for next step
                        ;       r10     z(ap)
                        ;       r11     pz(ap) = precision of quotient
                        ;
                        ;--
                        ;
                        ; stack offsets for work area
                        ;
                                $offset 0,,<-                   ;
                                <,16>,-                         ;x, 31 digits
                                <stky1,8>,-                     ;y1 15 digits
                                <stky2,9>,-                     ;y2 16 digits
                                <stkz2,16>,-                    ;z2 31 digits
                                <stkt1,16>,-                    ;t1 31 digits
                                <stkt2,16>,-                    ;t2 31 digits (15 digits used)
                                <stky,16>,-                     ;y  31 digits
                                <stkt3,9>,-                     ;t3 16 digits
                                <stkt4,16>,-                    ;t4 31 digits
                                <stksign,1>,-                   ;sign of quotient, 2 bits
                                <stklen,0>,-                    ;length of work area
                                >                               ;
                        ;
                        ; parameter offsets
                        ;
                                $offset 4,,<-                   ;
                                <x>,-                           ;x = dividend by reference
                                <y>,-                           ;y = divisor by reference
                                <py>,-                          ;prec(y) by value
                                <z>,-                           ;z = quotient by reference
                                <pz>,-                          ;prec(z) by value
                                <consta>,-                      ;a as defined above
                                >                               ;
                        ;
                        ; constant data area
                        ; (plidata.mar not used because they cause longword displacements to be used)
                        ;
                                rtshare
                    1C  one:    .packed +1                      ;
                    0C  zero:   .packed +0                      ;
                        ;**warning** the following two data definitions must be contiguous
                    09  nines:  .byte   9                       ;10**16 - 1 (must be followed by 10**15-1)
          9C 99 99 99 99 99 99 99  nine15: .packed +999999999999999  ;10**15 - 1
99 99 99 99 09 00 00 00 00 00 00 00  bignine:.packed +00000000000000099999999999999999 ;10**16 - 1
                   9C 99 99 99
          0C 00 00 00 00 00 00 00 01  ten15:  .packed +1000000000000000  ;10**15
          9D 99 99 99 99 99 99 99 09  neg9:   .packed -9999999999999999  ;-(10**16 - 1)
                        ;
                        ; local symbol definitions
                        ;
                0000000F bytes_to_sign=15                       ;bytes to sign for fixed decimal 31
                        ;
                        ; run time routine pli$div_pk_long
                        ;
                    CFFC        .entry  pli$div_pk_long,^M<iv,dv,r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
```

```
                    5E    85 AE    9E   002F    228          movab   -stklen(sp),sp          ;make room for temporaries
                    56    18 AC    D0   0033    229          movl    consta(ap),r6           ;get value of a
                    57    51 AE    9E   0037    230          movab   stky(sp),r7             ;address of copy of divisor
                    58    0C AC    D0   003B    231          movl    py(ap),r8               ;precision of divisor
                    5A    10 AC    D0   003F    232          movl    z(ap),r10               ;save address of quotient
                    5B    14 AC    D0   0043    233          movl    pz(ap),r11              ;precision of quotient
                          7A AE    94   0047    234          clrb    stksign(sp)             ;clear sign flag
                6E    04 BC    1F   34   004A    235          movp    #31,@x(ap),(sp)         ;move x, set cond. code
                               24   14   004F    236          bgtr    50$                     ;branch if x > 0
                               1C   19   0051    237          blss    40$                     ;branch if x < 0
                                         0053    238 ;
                                         0053    239 ;x = 0
                                         0053    240 ;
        08 BC    58    AA AF    00   37   0053    241          cmpp4   #0,zero,r8,@y(ap)       ;divisor zero?
                               09   13   005A    242          beql    30$                     ;branch if divide by 0
6A   5B    00    A0 AF    00    00   F8   005C    243          ashp    #0,#0,zero,#0,r11,(r10) ;z = 0
                               04   0064    244          ret
  5B   95 AF    00    98 AF    00   27   0065    245 30$:     divp    #0,zero,#0,zero,r11,(r10) ;cause divide by zero
                               6A   006D
                               04   006E    246          ret
                                         006F    247 ;
                                         006F    248 ;x not 0
                                         006F    249 ;
                                         006F    250 40$:                                ;
                          7A AE    96   006F    251          incb    stksign(sp)             ;set low order bit
                          0F AE    97   0072    252          decb    bytes_to_sign(sp)       ;x < 0 so make it positive
                                         0075    253 ;
                                         0075    254 ;determine sign of y
                                         0075    255 ;y may be 0 at this point
                                         0075    256 ;optimized for y>0
                                         0075    257 ;
                                         0075    258 50$:                                ;
                67    08 BC    58   34   0075    259          movp    r8,@y(ap),(r7)          ;move y into temporary
                               0D   18   007A    260          bgeq    60$                     ;branch if so
                          7A AE    96   007C    261          incb    stksign(sp)             ;set neg indicator
58   FF7A CF    0J    08 BC    58   23   007F    262          subp6   r8,@y(ap),#0,zero,r8,(r7) ;conve t to positive
                               67             0088
                                         0089    263 ;
                                         0089    264 ;get y1 and y2
                                         0089    265 ;
                                         0089    266 60$:                                ;
    0F    00    67    58    F0 8F    F8   0089    267          ashp    #-16,r8,(r7),#0,#15,stky1(sp) ;high order 15 digits of y
                               10 AE        0090
    1F    00    10 AE    0F    10   F8   0092    268          ashp    #16,#15,stky1(sp),#0,#31,stkt1(sp) ;y with 16 low order zeroed
                               31 AE        0099
    10    67    58    31 AE    1F   23   009B    269          subp6   #31,stkt1(sp),r8,(r7),#16,stky2(sp) ;16 low order digits y
                               18 AE        00A2
                                         00A4    270 ;
                                         00A4    271 ;prec(y) is large enough for y1 to possibly not be 0
                                         00A4    272 ;
    10 AE    0F    FF58 CF    00   37   00A4    273          cmpp4   #0,zero,#15,stky1(sp)   ;y1=0?
                               03   13   00AC    274          beql    80$                     ;branch if y1 is zero
                          0098 31   31   00AE    275          brw     200$                    ;branch if y1 is not zero
                                         00B1    276 ;
                                         00B1    277 ; y2(16) holds all of y
                                         00B1    278 ;
                                         00B1    279 80$:                                ;
```

```
                18 AE    10    6E    1F    37   00B1    280          cmpp4   #31,(sp),#16,stky2(sp)  ;x<y ?
                                     13    19   00B7    281          blss    95$                     ;branch if x<y; shift x by 15 is ok
6A      5B      6E     1F     18 AE    10    27   00B9    282          divp    #16,stky2(sp),#31,(sp),r11,(r10) ;z=x/y2
        1F      6A     5B     18 AE    10    25   00C1    283          mulp    #16,stky2(sp),r11,(r10),#31,stkt1(sp) ;t1=(x/y2)*y2
                                        31 AE          00C8
                                     15    11   00CA    284          brb     110$                    ;
    5B      00      FF2F CF      00    00    F8   00CC    285 95$:     ashp    #0,#0,zero,#0,r11,(r10) ;clear quotient
                                     6A          00D4
                                     12    11   00D5    286          brb     115$
  1F      21 AE     0F     18 AE    10    25   00D7    287 100$:    mulp    #16,stky2(sp),#15,stkz2(sp),#31,stkt1(sp) ;t1=y2*z2
                                        31 AE          00DF
          6E      1F     31 AE    1F    22   00E1    288 110$:    subp4   #31,stkt1(sp),#31,(sp)  ;x=x-t1
                                     3E    13   00E7    289          beql    150$                    ;branch if remainder = 0
                                           C0E9    290 ;
                                           00E9    291 ;determine r, the number of the next low order digits to obtain
                                           00E9    292 ;
                        59     0F    D0   00E9    293 115$:    movl    #15,r9                  ,r=15
                        59     56    D1   00EC    294          cmpl    r6,r9                   ;a>15?
                                     03    18   00EF    295          bgeq    130$                    ;branch if larger
                        59     56    D0   00F1    296          movl    r6,r9                   ;r=a
31 AE     1F     00     6E     1F    59   F8   00F4    297 130$:    ashp    r9,#31,(sp),#0,#31,stkt1(sp) ;shift x left by r
                6E      31 AE    1F    34   00FC    298          movp    #31,stkt1(sp),(sp)      ;copy back into x
31 AE     5B     00     6A     5B    59   F8   0101    299          ashp    r9,r11,(r10),#0,r11,stkt1(sp) ;shift z left by r
                6A      31 AE    5B    34   0109    300          movp    r11,stkt1(sp),(r10)     ;copy back into z
        0F      6E     1F     18 AE    10    27   010E    301          divp    #16,stky2(sp),#31,(sp),#15,stkz2(sp) ;z2(15)=x/y2
                                        21 AE          0115
          6A      5B     21 AE    0F    20   0117    302          addp4   #15,stkz2(sp),r11,(r10) ;z=z+z2
                        56     59    C2   011D    303          subl2   r9,r6                   ;a=a-r
                                     B5    12   0120    304          bneq    100$                    ;branch if more
                           13 7A AE    E8   0122    305 140$:    blbs    stksign(sp),155$        ;branch if quotient <0
                                     04    0126    306          ret                             ;
                                           0127    307 ;
                                           0127    308 ;remainder = 0
                                           0127    309 ;
                                           0127    310 150$:                                    ;remainder = 0
31 AE     5B     00     6A     5B    56   F8   0127    311          ashp    r6,r11,(r10),#0,r11,stkt1(sp) ;account for scale
                                  0B 7A AE    E8   012F    312          blbs    stksign(sp),160$        ;branch if quotient is negative
                6A      31 AE    5B    34   0133    313          movp    r11,stkt1(sp),(r10)     ;copy back into quotient
                                     04    0138    314          ret                             ;
                31 AE     6A     5B    34   0139    315 155$:    movp    r11,(r10),stkt1(sp)     ;copy quotient into temporary
                                           013E    316 160$:                                    ;enter if t1 holds quotient
5B      FEBB CF      00     31 AE    5B    23   013E    317          subp6   r11,stkt1(sp),#0,zero,r11,(r10) ;make z negative
                                     6A          0147
                                     04    0148    318          ret
                                           0149    319 ;
                                           0149    320 ;New division algorithm
                                           0149    321 ;
                                           0149    322 200$:                                    ;
                                           0149    323 ;
                                           0149    324 ;insure that we have x<y
                                           0149    325 ;
                67     5B     6E    1F    37   0149    326          cmpp4   #31,(sp),r8,(r7)         ;x<y?
                                     19    19   014E    327          blss    220$                    ;branch if x < y
    6A      5B     6E     1F     67    58    27   0150    328          divp    r8,(r7),#31,(sp),r11,(r10) ;z=x/y
31 AE     1F     67     58     6A    5B    25   0157    329          mulp    r11,(r10),r8,(r7),#31,stkt1(sp);t1=y*z
          6E      1F     31 AE    1F    22   015F    330          subp4   #31,stkt1(sp),#31,(sp)  ;x=x-t1
                                     C0    13   0165    331          beql    150$                    ;branch if remainder is zero
```

```
                                09   11  0167   332          brb     230$                    ;
   5B   00    FE92 CF    00     00   F8  0169   333 220$:    ashp    #0,#0,zero,#0,r11,(r10) ;clear quotient
                                6A       0171
                                         0172   334 ;
                                         0172   335 ;Assumes z is defined
                                         0172   336 ;       0 < x < y
                                         0172   337 ;       and r6=a gives the number of additional digits required.
                                         0172   338 ;Determine r = # of digits for next part of quotient.
                                         0172   339 ;
                           59    56  D0  0172   340 230$:    movl    r6,r9                   ;r=a
                           0F    56  D1  0175   341          cmpl    r6,#15                  ;a>15?
                                 03  15  0178   342          bleq    240$                    ;branch if larger
                           59    0F  D0  017A   343          movl    #15,r9                  ;r=15
                           56    59  C2  017D   344 240$:    subl2   r9,r6                   ;update r6 = a
                                         0180   345 ;
                                         0180   346 ;calculate z2 = min(B-1,[x/y1])
                                         0180   347 ;
                                         0180   348 250$:                                   ;
    1F   6E    1F    10 AE     0F   27  0180   349          divp    #15,stky1(sp),#31,(sp),#31,stkt1(sp);t1=x/y1
                                31 AE    0187
  FE71 CF    10    31 AE     1F   37  0189   350          cmpp4   #31,stkt1(sp),#16,nines ;t1>10**16-1?
                                 09  15  0191   351          bleq    310$                    ;branch if not
        21 AE    FE6B CF     0F   34  0193   352          movp    #15,nine15,stkz2(sp)    ;move in base-1
                                 0A  11  019A   353          brb     320$                    ;skip ashp, you have 15 digits
                                         019C   354 ;
                                         019C   355 ;calculate y2*z2
                                         019C   356 ;       t3 = high order 16 digits of y2*z2
                                         019C   357 ;       t2 =  low order 15 digits of y2*z2
                                         019C   358 ;
                                         019C   359 310$:
  0F   00    31 AE     1F    FF 8F     F8  019C   360          ashp    #-1,#31,stkt1(sp),#0,#15,stkz2(sp) ;we only get 15 digits
                                21 AE    01A4
  1F   18 AE    10    21 AE     0F   25  01A6   361 320$:    mulp    #15,stkz2(sp),#16,stky2(sp),#31,stkt1(sp) ;t1=y2*z2
                                31 AE    01AE
  10   00    31 AE     1F    F1 8F     F8  01B0   362          ashp    #-15,#31,stkt1(sp),#0,#16,stkt3(sp) ;t3(16)=t1 shifted right 15
                                61 AE    01B8
     1F   00    61 AE     10    0F     F8  01BA   363          ashp    #15,#16,stkt3(sp),#0,#31,stkt4(sp) ;t4(31) = t3 shifted left 15
                                6A AE    01C1
  0F   31 AE    1F    6A AE     1F   23  01C3   364          subp6   #31,stkt4(sp),#31,stkt1(sp),#15,stkt2(sp) ;t2(15)=t1-t4
                                41 AE    01CB
                                 4C  13  01CD   365          beql    330$                    ;branch if no borrow required
                                         01CF   366 ;
                                         01CF   367 ;borrow is -1, t2 not 0
                                         01CF   368 ;calculate R(H) =
                                         01CF   369 ;       t1(31) = 31 high order digits of x(46) - y*z2
                                         01CF   370 ;       t2(15) = 15  low order digits of x(46) - y*z2
                                         01CF   371 ;
  1E   10 AE    0F    21 AE     0F   25  01CF   372          mulp    #15,stkz2(sp),#15,stky1(sp),#30,stkt4(sp) ;t4(30)=y1*z2
                                6A AE    01D7
     1F   00    6A AE     1E    01 F8     01D9   373          ashp    #1,#30,stkt4(sp),#0,#31,stkt1(sp);31 high order of 46
                                31 AE    01E0
       31 AE    1F    61 AE     10   20  01E2   374          addp4   #16,stkt3(sp),#31,stkt1(sp) ;t1(31)=t1(31)+t3(16)
     1F   6E    1F    31 AE     1F   23  01E9   375          subp6   #31,stkt1(sp),#31,(sp),#31,stkt4(sp) ;t4=x-t1
                                6A AE    01F0
                                 1E  15  01F2   376          bleq    325$                    ;branch if R(H) negative (t4 < = 0)
1F   6A AE    1F    FE07 CF     01   23  01F4   377          subp6   #1,one,#31,stkt4(sp),#31,stkt1(sp) ;t1=t4-1 (borrow 1)
                                31 AE    01FD
```

```
0F  FE14 CF   10   41 AE   0F   23  01FF            subp6   #15,stkt2(sp),#16,ten15,#15,stkt4(sp) ;t4=10**15-t2
                      6A AE             0208
          41 AE   6A AE   0F   34  020A            movp    #15,stkt4(sp),stkt2(sp) ;copy back into t2
                          63   11  0210            brb     370$                    ;
          31 AE   6A AE   1F   34  0212  325$:     movp    #31,stkt4(sp),stkt1(sp) ;make t1 hold high order R(H)
                    00AF       31  0218            brw     500$                    ;R(H) < 0 (can not be zero)
                                   021B  ;
                                   021B  ;no borrow, t2 = 0
                                   021B  ;calculate R(H) = t1(31) = 31 high order digits of x(46) - y*z2
                                   021B  ;                t2(15) = 0 = 15 low order digits of x(46) - y*z2
                                   021B  ;
                                   021B  330$:
  1E    10 AE    0F   21 AE   0F   25  021B            mulp    #15,stkz2(sp),#15,stky1(sp),#30,stkt1(sp) ;t1(30)=z2(15)*y1(15)
                      31 AE             0223
    1F    00    31 AE    1E   01   F8  0225            ashp    #1,#30,stkt1(sp),#0,#31,stkt4(sp) ;31 high order 46
                      6A AE             022C
    6A AE    1F   61 AE   10   20  022E            addp4   #16,stkt3(sp),#31,stkt4(sp) ;t4=t3(16)+t4
    1F    6E    1F   6A AE   1F   23  0235            subp6   #31,stkt4(sp),#31,(sp),#31,stkt1(sp) ;t1=x-t4
                      31 AE             023C
                          35   12  023E            bneq    370$                    ;branch if remainder not zero
                                   0240  ;
                                   0240  ;*********************************************************************
                                   0240  ;*                                                                   *
                                   0240  ;* R(H) = 0 , i.e.                                                  *
                                   0240  ;*remainder is zero                                                  *
                                   0240  ;*z2(15) holds last non-zero digits of the quotient                  *
                                   0240  ;*quotient has not been shifted yet to recieve z2                    *
                                   0240  ;*                                                                   *
                                   0240  ;* this is a local routine with no exit other than ret               *
                                   0240  ;*********************************************************************
                                   0240  ;
                                   0240  340$:
6A AE    5B    00    6A    5B   59   F8  0240            ashp    r9,r11,(r10),#0,r11,stkt4(sp) ;t4=z shifted left by r9=r
              6A    6A AE   5B   34  0248            movp    r11,stkt4(sp),(r10)     ;copy back quotient
                          59   0F  C2  024D            subl2   #15,r9                  ;shift needed to leave r9 digits
    0F    00    21 AE    0F   59   F8  0250            ashp    r9,#15,stkz2(sp),#0,#15,stkt1(sp);low order digits of z
                      31 AE             0257
       6A    5B    31 AE   0F   20  0259            addp4   #15,stkt1(sp),r11,(r10) ;z=z+z2
                          56   D5  025F            tstl    r6                      ;a = 0 ?
                          52   13  0261            beql    395$                    ;branch if a=0 (last iteration)
 31 AE    5B    00    6A    5B   56   F8  0263            ashp    r6,r11,(r10),#0,r11,stkt1(sp);adjust for scale
                  50 7A AE   E8  026B  350$:     blbs    stksign(sp),410$        ;branch if quotient < 0
          6A    31 AE   5B   34  026F            movp    r11,stkt1(sp),(r10)     ;back into quotient
                          04   0274            ret
                          53   19  0275  370$:     blss    500$                    ;branch if R(H)<0
                                   0277  ;
                                   0277  ;*********************************************************************
                                   0277  ;*R(H) > 0 ,  i.e.                                                   *
                                   0277  ;* t1 > 0                                                            *
                                   0277  ;*                                                                   *
                                   0277  ;*********************************************************************
                                   0277  ;
6A AE    5B    00    6A    5B   59   F8  0277  380$:     ashp    r9,r11,(r10),#0,r11,stkt4(sp) ;t4=z shifted left by r9=r
              6A    6A AE   5B   34  027F            movp    r11,stkt4(sp),(r10)     ;copy back quotient
                          56   D5  0284            tstl    r6                      ;a = 0 ?
                          1B   13  0286            beql    390$                    ;branch if a=0 (last iteration)
                                   0288  ;
```

```
                                                0288    430 ;a not 0
                                                0288    431 ;
      6A   5B    21 AE    0F    20              0288    432          addp4   #15,stkz2(sp),r11,(r10) ;z=z+z2
   1F   00   31 AE    1F    0F    F8              028E    433          ashp    #15,#31,stkt1(sp),#0,#31,stkt4(sp) ;t4=t1 shifted left 15
                             6A AE                0295
 1F    41 AE    0F    6A AE    1F    21          0297    434          addp6   #31,stkt4(sp),#15,stkt2(sp),#31,(sp) ;x=t4+t2
                                   6E             029F
                          FECF    31              02A0    435          brw     230$
                                                02A3    436 ;
                                                02A3    437 ;a = 0
                                                02A3    438 ;
                             59    0F    C2      02A3    439 390$:    subl2   #15,r9                      ;shift needed to leave r9 digits
   0F   00   21 AE    0F    59    F8              02A6    440          ashp    r9,#15,stkz2(sp),#0,#15,stkt1(sp);low order digits of z
                                31 AE             02AD
      6A   5B    31 AE    0F    20              02AF    441          addp4   #15,stkt1(sp),r11,(r10) ;z=z+z2
                      01 7A AE    E8              02B5    442 395$:    blbs    stksign(sp),400$            ;branch if quotient <0
                                04                02B9    443          ret                                 ;
                                                02BA    444 400$:                                        ;
      31 AE    6A    5B    34                    02BA    445          movp    r11,(r10),stkt1(sp)         ;make copy of quotient
5B  FD3A CF   00   31 AE    5B    23              02BF    446 410$:    subp6   r11,stkt1(sp),#0,zero,r11,(r10) ;make z negative
                                   6A             02C8
                                04                02C9    447          ret
                                                02CA    448 ;
                                                02CA    449 ;*****************************************************************************
                                                02CA    450 ;*                                                                           *
                                                02CA    451 ;*R(H) < 0                                                                    *
                                                02CA    452 ;* so check if R(H) + Y > = 0                                                 *
                                                02CA    453 ;*                                                                           *
                                                02CA    454 ;*****************************************************************************
                                                02CA    455 ;
                                                02CA    456 500$:
   FD52 CF    10    31 AE    1F    37              02CA    457          cmpp4   #31,stkt1(sp),#16,neg9  ;t1< -(10**16-1) ?
                                   66    19        02D2    458          blss    600$                    ;branch if R(H) + Y < 0
   1F   00   31 AE    1F    0F    F8              02D4    459          ashp    #15,#31,stkt1(sp),#0,#31,stkt4(sp) ;shift t1 left 15
                             6A AE                02DB
 1F    6A AE    1F    41 AE    0F    23          02DD    460          subp6   #15,stkt2(sp),#31,stkt4(sp),#31,stkt1(sp) ;t1=t1-t2
                                31 AE             02E5
      31 AE    1F    67    58    20              02E7    461          addp4   r8,(r7),#31,stkt1(sp)   ;t1=t1+y
                                   4B    19        02ED    462          blss    600$                    ;branch if R(H) + Y < 0
                                   3E    13        02EF    463          beql    530$                    ;branch if no remainder (first adjust z2)
   21 AE    0F    FD0A CF    01    22              02F1    464          subp4   #1,one,#15,stkz2(sp)     ;z2=z2-1
6A AE    5B    00    6A    5B    59              02F9    465          ashp    r9,r11,(r10),#0,r11,stkt4(sp) ;t4=z shifted left by r9=r
            6A    6A AE    5B    34              0301    466          movp    r11,stkt4(sp),(r10)     ;copy back quotient
                                   56    D5        0306    467          tstl    r6                      ;a = 0 ?
                                   0E    13        0308    468          beql    510$                    ;branch if a=0 (last iteration)
                                                030A    469 ;
                                                030A    470 ;a not 0
                                                030A    471 ;
      6A   5B    21 AE    0F    20              030A    472          addp4   #15,stkz2(sp),r11,(r10) ;z=z+z2
           6E    31 AE    1F    34              0310    473          movp    #31,stkt1(sp),(sp)      ;x=t1
                          FE5A    31              0315    474          brw     230$
                                                0318    475 ;
                                                0318    476 ;a = 0
                                                0318    477 ;
                             59    0F    C2      0318    478 510$:    subl2   #15,r9                      ;shift needed to leave r9 digits
   0F   00   21 AE    0F    59    F8              031B    479          ashp    r9,#15,stkz2(sp),#0,#15,stkt1(sp);low order digits of z
                                31 AE             0322
```

```
                6A   5B   31 AE    0F   20  0324   480          addp4   #15,stkt1(sp),r11,(r10) ;z=z+z2
                               8C 7A AE   E8  032A   481          blbs    stksign(sp),400$        ;branch if quotient <0
                                          04  032E   482          ret
                                              032F   483 ;
                                              032F   484 ;remainder is zero
                                              032F   485 ;
     21 AE   0F   FCCC CF    01   22  032F   486 530$:   subp4   #1,one,#15,stkz2(sp)    ;z2=z2-1
                              FF06   31  0337   487          brw     340$                    ;go add to quotient, then ret
                                              033A   488 ;
                                              033A   489 ;*****************************************************************************
                                              033A   490 ;*                                                                           *
                                              033A   491 ;*R(H) + Y < 0                                                                 *
                                              033A   492 ;*calculate L = min(B-1,[X/(y1+1)])                                            *
                                              033A   493 ;* by theorem 5, L=[X/(y1+1)]                                                 *
                                              033A   494 ;*                                                                           *
                                              033A   495 ;*****************************************************************************
                                              033A   496 ;
10   10 AE   0F   FCC1 CF    01   21  033A   497 600$:   addp6   #1,one,#15,stky1(sp),#16,stkt3(sp) ;t3=y1+1
                                61 AE         0343
     10   6E   1F   61 AE    10   27  0345   498          divp    #16,stkt3(sp),#31,(sp),#16,stkt1(sp) ;t1=x/t3
                                31 AE         034C
                                              034E   499 ;
                                              034E   500 ;split up y2*L
                                              034E   501 ;       t3 = high order 16 digits of y2*L
                                              034E   502 ;       t2 = low  order 15 digits of y2*L
  0F   00   31 AE   10   FF 8F   F8  034E   503          ashp    #-1,#16,stkt1(sp),#0,#15,stkz2(sp) ;we only get 15 digits
                                21 AE         0356
  1F   18 AE   10   21 AE    0F   25  0358   504          mulp    #15,stkz2(sp),#16,stky2(sp),#31,stkt1(sp) ;t1=y2*z2
                                31 AE         0360
  10   00   31 AE   1F   F1 8F   F8  0362   505          ashp    #-15,#31,stkt1(sp),#0,#16,stkt3(sp) ;t3(16)=t1 shifted right 15
                                61 AE         036A
     1F   00   61 AE   10   0F   F8  036C   506          ashp    #15,#16,stkt3(sp),#0,#31,stkt4(sp) ;t4(31) = t3 shifted left 15
                                6A AE         0373
  0F   31 AE   1F   6A AE    1F   23  0375   507          subp6   #31,stkt4(sp),#31,stkt1(sp),#15,stkt2(sp) ;t2(15)=t1-t4
                                41 AE         037D
                                              037F   508 ;
                                              037F   509 ;*****************************************************************************
                                              037F   510 ;*                                                                           *
                                              037F   511 ;* calculate R(L)                                                           *
                                              037F   512 ;*                                                                           *
                                              037F   513 ;*****************************************************************************
                                              037F   514 ;
                                    41   13   037F   515          beql    630$                    ;branch if no borrow required
                                              0381   516 ;
                                              0381   517 ;borrow is -1, t2 not 0
                                              0381   518 ;calculate R(L) =
                                              0381   519 ;       t1(31) = 31 high order digits of x(46) - y*z2
                                              0381   520 ;       t2(15) = 15  low order digits of x(46) - y*z2
                                              0381   521 ;note: it is always true that R(L) > = 0
                                              0381   522 ;
0F   FC92 CF   10   41 AE    0F   23  0381   523          subp6   #15,stkt2(sp),#16,ten15,#15,stkt4(sp) ;t4=10**15-t2
                                6A AE         038A
          41 AE   6A AE    0F   34  038C   524          movp    #15,stkt4(sp),stkt2(sp) ;copy back into t2
  1E   10 AE   0F   21 AE    0F   25  0392   525          mulp    #15,stkz2(sp),#15,stky1(sp),#30,stkt4(sp) ;t4(30)=y1*z2
                                6A AE         039A
     1F   00   6A AE   1E   01   F8  039C   526          ashp    #1,#30,stkt4(sp),#0,#31,stkt1(sp) ;high order 31 of 46
                                31 AE         03A3
```

```
      31 AE   1F   61 AE    10   20  03A5    527          addp4   #16,stkt3(sp),#31,stkt1(sp) ;t1(31)=t1(31)+t3(16)
   1F   6E   1F   31 AE    1F   23  03AC    528          subp6   #31,stkt1(sp),#31,(sp),#31,stkt4(sp) ;t4=x-t1
                        6A AE        03B3
1F   6A AE   1F   FC46 CF    01   23  03B5    529          subp6   #1,one,#31,stkt4(sp),#31,stkt1(sp) ;t1=t4-1 (borrow 1)
                           5 AE      03BE
                             23   11  03C0    530          brb     700$                    ;
                                      03C2    531 ;
                                      03C2    532 ;no borrow, t2 = 0
                                      03C2    533 ;calculate R(L) = t1(31) = x - y1*z2
                                      03C2    534 ;
                                      03C2    535 630$:
  1E   10 AE   0F   21 AE    0F   25  03C2    536          mulp    #15,stkz2(sp),#15,stky1(sp),#30,stkt1(sp) ;t1(30)=y1*z2
                           31 AE      03CA
   1F   00   31 AE   1E    01   F8  03CC    537          ashp    #1,#30,stkt1(sp),#0,#31,stkt4(sp) ;high order 31 of 46
                        6A AE        03D3
      6A AE   1F   61 AE    10   20  03D5    538          addp4   #16,stkt3(sp),#31,stkt4(sp) ;t4(31)=t4(31)+t3(16)
   1F   6E   1F   6A AE    1F   23  03DC    539          subp6   #31,stkt4(sp),#31,(sp),#31,stkt1(sp) ;t1=x-t4
                           31 AE      03E3
                                      03E5    540 ;
                                      03E5    541 ;*************************************************************************
                                      03E5    542 ;*                                                                       *
                                      03E5    543 ;* calculate Z2 = L + R(L)/Y                                            *
                                      03E5    544 ;*                                                                       *
                                      03E5    545 ;*************************************************************************
                                      03E5    546 ;
   1F   00   31 AE   1F    0F   F8  03E5    547 700$:    ashp    #15,#31,stkt1(sp),#0,#31,stkt4(sp) ;shift t1 left 15
                        6A AE        03EC
      6A AE   1F   41 AE    0F   20  03EE    548          addp4   #15,stkt2(sp),#31,stkt4(sp) ;t4=t4+t2
   0F   6A AE   1F   67    58   27  03F5    549          divp    r8,(r7),#31,stkt4(sp),#15,stkt1(sp) ;t1=t4/y
                           31 AE      03FC
      21 AE   0F   31 AE    0F   20  03FE    550          addp4   #15,stkt1(sp),#15,stkz2(sp) ;z2=z2+t1
                        FD9E   31   0405    551          brw     320$                    ;
                                      0408    552
                                      0408    553          .end
```

```
BIGNINE             0000000B R      02
BYTES_TO_SIGN   =   0000000F
CONSTA              00000018
DIR...          =   00000001
NEG9                00000024 R      02
NINE15              00000003 R      02
NINES               00000002 R      02
ONE                 00000000 R      02
PLI$DIV_PK_LONG     0000002D RG     02
PY                  0000000C
PZ                  00000014
STKLEN              0000007B
STKSIGN             0000007A
STKT1               00000031
STKT2               00000041
STKT3               00000061
STKT4               0000006A
STKY                00000051
STKY1               00000010
STKY2               00000018
STKZ2               00000021
TEN15               0000001B R      02
X                   00000004
Y                   00000008
Z                   00000010
ZERO                00000001 R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	0000007B (123.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_PLI$CODE	00000408 (1032.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	13	00:00:00.07	00:00:00.31
Command processing	74	00:00:00.57	00:00:02.13
Pass 1	92	00:00:02.36	00:00:04.43
Symbol table sort	0	00:00:00.05	00:00:00.05
Pass 2	105	00:00:01.19	00.00:02.95
Symbol table output	2	00:00:00.03	00:00:00.20
Psect synopsis output	1	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	287	00:00:04.29	00:00:10.09

The working set limit was 900 pages.
15119 bytes (30 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 26 non-local and 37 local symbols.
553 source lines were read in Pass 1, producing 14 object records in Pass 2.

3 pages of virtual memory were used to define 3 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[PLIRTL.OBJ]PLIRTMAC.MLB;1	3
_255DUA28:[SYSLIB]STARLET.MLB;2	0
TOTALS (all libraries)	3

44 GETS were required to define 3 macros.

There were no errors, warnings or information messages.

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

PLIFORMAT
LIS

PLIGETBUF
LIS

PLIMSGTXT
LIS

PLIGETEDI
LIS

PLIHEEP
LIS

PLIPUTFIL
LIS

PLIRMSBIS
LIS

PLIRECOPT
LIS

PLIOPEN
LIS

PLIREAD
LIS

PLIPROTEC
LIS

PLIREWRIT
LIS

PLIGETLIS
LIS

PLIPUTEDI
LIS

PLIPKDIVL
LIS

PLIPUTLIS
LIS

PLIMSGPTR
LIS

PLIPKDIVS
LIS

PLIPUTBUF
LIS

PLIGETFIL
LIS