```
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
```

FILEID**PLIREWRIT

```
PPPPPPPP    LL            IIIIII    RRRRRRRR    EEEEEEEEEE  WW      WW  RRRRRRRR      IIIIII    TTTTTTTTTT
PPPPPPPP    LL            IIIIII    RRRRRRRR    EEEEEEEEEE  WW      WW  RRRRRRRR      IIIIII    TTTTTTTTTT
PP      PP  LL              II      RR      RR  EE          WW      WW  RR      RR      II          TT
PP      PP  LL              II      RR      RR  EE          WW      WW  RR      RR      II          TT
PP      PP  LL              II      RR      RR  EE          WW      WW  RR      RR      II          TT
PP      PP  LL              II      RR      RR  EE          WW      WW  RR      RR      II          TT
PPPPPPPP    LL              II      RRRRRRRR    EEEEEEEE    WW      WW  RRRRRRRR        II          TT
PPPPPPPP    LL              II      RRRRRRRR    EEEEEEEE    WW      WW  RRRRRRRR        II          TT
PP          LL              II      RR  RR      EE          WW  WW  WW  RR  RR          II          TT
PP          LL              II      RR  RR      EE          WW  WW  WW  RR  RR          II          TT
PP          LL              II      RR    RR    EE          WWWW  WWWW  RR    RR        II          TT          ....
PP          LL              II      RR    RR    EE          WWWW  WWWW  RR    RR        II          TT          ....
PP          LLLLLLLLLL    IIIIII    RR      RR  EEEEEEEEEE  WW      WW  RR      RR    IIIIII        TT          ....
PP          LLLLLLLLLL    IIIIII    RR      RR  EEEEEEEEEE  WW      WW  RR      RR    IIIIII        TT          ....

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

```
0000     1         .title  pli$rewrite - pl1 runtime rewrite file
0000     2         .ident  /1-003/                                 ; Edit DSB1003
0000     3
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 ;++
0000    31 ; facility:
0000    32 ;
0000    33 ;       VAX/VMS PL1 runtime library.
0000    34 ; abstract:
0000    35 ;
0000    36 ;       This module contains the pl1 runtime routine for rewriting a record to
0000    37 ;       a file.
0000    38 ;
0000    39 ; author: c. spitz 18-jul-79
0000    40 ;
0000    41 ; modified:
0000    42 ;
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   Dave Blickstein 21-February-1984
0000    49 ;
0000    50 ;               Do a $FIND if RECORD_ID was specified.  RECORD_ID was
0000    51 ;               being "ignored" because RMS does not allow it for $UPDATE.
0000    52 ;
0000    53 ;--
0000    54
0000    55
0000    56 ;+
0000    57 ; external definitions
```

```
0000     58 ;-
0000     59
0000     60          $deffcb                         ;define file control block offsets
0000     61          $defdat                         ;define operand node data types
0000     62          $defpl1rtcons                   ;define pl1 runtime constants
0000     63          $rabdef                         ;define rab offsets
0000     64          $fabdef                         ;define fab offsets
0000     65          $rmsdef                         ;define rms error codes
0000     66
0000     67 ;+
0000     68 ; local definitions
0000     69 ;-
0000     70 $offset 4,positive,<-                    ;define arguments
0000     71          <fcbaddr,4>,-                   ;addr of fcb
0000     72          <fromaddr,4>,-                  ;addr of from
0000     73          <fromlen,2>,-                   ;length of from
0000     74          <fromtyp,2>,-                   ;data type of from
0000     75          <keyaddr,4>,-                   ;addr of key
0000     76          <keylen,4>,-                    ;length of key
0000     77          <keytyp,4>,-                    ;data type of key
0000     78          <keynum,4>,-                    ;addr of key number
0000     79          <matchgtr,4>,-                  ;addr of match greater
0000     80          <matchgeq,4>,-                  ;addr of match greater equal
0000     81          <recidfrom,4>,-                 ;addr of record id from
0000     82          <recidto,4>,-                   ;addr of record id to
0000     83          <fxcaddr,4>,-                   ;addr of fixed control
0000     84          <fxclen,2>,-                    ;length of fixed control
0000     85          <fxctyp,2>,-                    ;data type of fixed control
0000     86          >                               ;
0000     87
0000     88          rtshare                         ;sharable
0000     89
```

```
                     0000     91
                     0000     92 ;++
                     0000     93 ; pli$rewrite -- rewrite a record to a file
                     0000     94 ;
                     0000     95 ; functional description:
                     0000     96 ;
                     0000     97 ; This routine rewrites a record to a pl1 file.
                     0000     98 ;
                     0000     99 ; inputs:
                     0000    100 ;       (ap) - number of arguments
                     0000    101 ;                1 if no from, no key and no options
                     0000    102 ;                3 if from, no key and no options
                     0000    103 ;                6 if from, key, and no options
                     0000    104 ;                13 if any options
                     0000    105 ;       4(ap) - addr of fcb
                     0000    106 ;       8(ap) - addr of from
                     0000    107 ;       12(ap) - word length of from
                     0000    108 ;        14(ap) - word data type of from
                     0000    109 ;       16(ap) - addr of key
                     0000    110 ;       20(ap) - size/prec of key
                     0000    111 ;       24(ap) - data type of key
                     0000    112 ;       28(ap) - addr of key number
                     0000    113 ;       32(ap) - addr of match greater
                     0000    114 ;       36(ap) - addr of match greater
                     0000    115 ;       40(ap) - addr of record id from
                     0000    116 ;       44(ap) - addr of record id to
                     0000    117 ;       48(ap) - addr of fixed control
                     0000    118 ;       52(ap) - length of fixed control
                     0000    119 ;        54(ap) - data type of fixed control
                     0000    120 ;
                     0000    121 ; outputs:
                     0000    122 ;       fcb_l_attr
                     0000    123 ;               atr_m_bfall, atr_m_delete, atr_m_virgin and atr_m_write are set
                     0000    124 ;                       to false
                     0000    125 ;               atr_m_currec is set true
                     0000    126 ;       fcb_q_rfa is set to the rfa of the record that was rewritten
                     0000    127 ;
                     0000    128 ; side effects:
                     0000    129 ;       the record is rewritten in the file.
                     0000    130 ;
                     0000    131 ;--
                     0000    132
             007C    0000    133         .entry  pli$rewrite,^m<r2,r3,r4,r5,r6>
                     0002    134 ;
                     0002    135 ; check arguments
                     0002    136 ;
   01    6C    D1    0002    137         cmpl    (ap),#1                 ;enough arguments?
         14    13    0005    138         beql    10$                     ;if eql, yes
   03    6C    D1    0007    139         cmpl    (ap),#3                 ;correct number of args?
         0F    13    000A    140         beql    10$                     ;if eql, no
   06    6C    D1    000C    141         cmpl    (ap),#6                 ;correct number?
         0A    13    000F    142         beql    10$                     ;if eql, yes, cont
   0D    6C    D1    0011    143         cmpl    (ap),#13                ;correct number?
         05    13    0014    144         beql    10$                     ;if neq, no
         50    D4    0016    145         clrl    r0                      ;indicate not enough parms
      01CC    31    0018    146         brw     fail                    ;and fail
52 04 AC    D0    001B    147 10$:    movl    fcbaddr(ap),r2          ;get address of fcb
```

```
                53   0C A2    D0  001F   148          movl    fcb_l_attr(r2),r3       ;get attributes
                                  0023   149 ;
                                  0023   150 ; open the file if necessary. it will be opened with record and update.
                                  0023   151 ; if the file is still closed after calling open, the error condition is
                                  0023   152 ; signaled.
                                  0023   153 ;
              21 53    01    E0  0023   154          bbs     #atr_v_opened,r3,30$    ;if file opened, continue
          00001010 8F          DD  0027   155          pushl   #atr_m_record!atr_m_update ;file must have record and update
                      52       DD  002D   156 20$:     pushl   r2                      ;push address of fcb
    00000000'GF       02       FB  002F   157          calls   #2,g^pli$open           ;open the file
              53   0C A2        D0  0036   158          movl    fcb_l_attr(r2),r3       ;get attributes
              0A 53    01       E0  003A   159          bbs     #atr_v_opened,r3,30$    ;if file opened, continue
    50    00000000'8F          D0  003E   160          movl    #pli$_open,r0           ;set open failure
                    019?       31  0045   161          brw     fail                    ;and fail
                                  0048   162 ;
                                  0048   163 ; check statndard attributes of file. record and update must be present.
                                  0048   164 ; delete must be absent.
                                  0048   165 ;
              0A 53    0C       E0  0048   166 30$:     bbs     #atr_v_record,r3,40$    ;if file has record, continue
    50    00000000'8F          D0  004C   167          movl    #pli$_notrec,r0         ;set not record file
                    0191       31  0053   168          brw     fail                    ;and fail
              0A 53    04       E0  0056   169 40$:     bbs     #atr_v_update,r3,5000$  ;if file doesn't have update
    50    00000000'8F          D0  005A   170          movl    #pli$_notupdate,r0      ;set not update file
                    0183       31  0061   171          brw     fail                    ;and fail
                                  0064   172 ;
                                  0064   173 ; process options
                                  0064   174 ;
              54   62 A2        9E  0064   175 5000$:   movab   fcb_b_rab(r2),r4       ;get address of rab
                   2C A4        D4  0068   176          clrl    rab$l_rhb(r4)           ;assume no fixed control from
04 A4     00600000 8F          CA  006B   177          bicl    #<rab$m_kge!rab$m_kgt>,rab$l_rop(r4) ;clear kge and kgt
                 0D    6C      D1  0073   178          cmpl    (ap),#13                ;options passed?
                       40      19  0076   179          blss    60$                     ;if lss, no
                                  0078   180 ; fixed control
              50   30 AC        9E  0078   181          movab   fxcaddr(ap),r0          ;get addr of fixed control
          00000000'GF          16  007C   182          jsb     g^pli$$fxdctlfrom       ;process fixed control
                                  0082   183 ; key number
              50   1C AC        D0  0082   184          movl    keynum(ap),r0           ;get addr of key num
              51   10 AC        D0  0086   185          movl    keyaddr(ap),r1          ;get addr of key
          00000000'GF          16  008A   186          jsb     g^pli$$keynum           ;process keynumber
                                  0090   187 ; match greater
              50   20 AC        D0  0090   188          movl    matchgtr(ap),r0         ;get addr of match greater
          00000000'GF          16  0094   189          jsb     g^pli$$matchgtr         ;process match greater
                                  009A   190 ; match greater equal
              50   24 AC        D0  009A   191          movl    matchgeq(ap),r0         ;get addr of match greater equal
          00000000'GF          16  009E   192          jsb     g^pli$$matchgeq         ;process match greater equal
                                  00A4   193 ; record id to
              50   2C AC        D0  00A4   194          movl    recidto(ap),r0          ;get addr of record id to
          00000000'GF          16  00A8   195          jsb     g^pli$$valrecidto       ;validate record id to
                                  00AE   196 ; record id from
              50   28 AC        D0  00AE   197          movl    recidfrom(ap),r0        ;get addr of record id from
          00000000'GF          16  00B2   198          jsb     g^pli$$recidfrom        ;process record id from
                                  00B8   199 ;
                                  00B8   200 ; if from option is not present, try to use allocated buffer.
                                  00B8   201 ;
                 01    6C      D1  00B8   202 60$:     cmpl    (ap),#1                 ;from specified?
                       24      14  00BB   203          bgtr    90$                     ;if gtr, then maybe
              0A 53    11       E0  00BD   204 70$:     bbs     #atr_v_bfall,r3,80$     ;if buffer not allocated
```

```
                50   00000000'8F    D0  00C1   205 75$:    movl    #pli$_nofrom,r0         ;set no from specified
                           011C    31  00C8   206           brw     fail                    ;and fail
                        06   6C    D1  00CB   207 80$:     cmpl    (ap),#6                 ;key specified?
                             05    19  00CE   208           blss    85$                     ;if lss, no, cont
                          10 AC    D5  00D0   209           tstl    keyaddr(ap)             ;key specified?
                             EC    12  00D3   210           bneq    75$                     ;if neq, yes, fail, it requires from
                   28 A4    14 A2   D0  00D5   211 85$:     movl    fcb_l_buf(r2),rab$l_rbf(r4) ;copy address of buffer
                   22 A4    1C A2   B0  00DA   212           movw    fcb_l_buf_pt(r2),rab$w_rsz(r4) ;copy size of buffer to rab
                             65    11  00DF   213           brb     120$                    ;cont
                                         00E1   214 ;
                                         00E1   215 ; process from option.
                                         00E1   216 ;
                   28 A4    08 AC   D0  00E1   217 90$:     movl    fromaddr(ap),rab$l_rbf(r4) ;copy address of buffer
                             D5    13  00E6   218           beql    70$                     ;if eql, then not specified
                          0C AC    DD  00E8   219           pushl   fromlen(ap)             ;push size and data type of from ref
                00000000'GF    01    FB  00EB   220           calls   #1,g^pli$$bytesize      ;get byte size
                          03 50    E8  00F2   221           blbs    r0,95$                  ;if invalid data type
                           00E1    31  00F5   222           brw     fail                    ;then fail
                      22 A4    51    B0  00F8   223 95$:     movw    r1,rab$w_rsz(r4)        ;set size in rab
                      0E AC    0B    B1  00FC   224           cmpw    #dat_k_char_var,fromtyp(ap) ;if from is char var
                             13    12  0100   225           bneq    100$                    ;then
                      0B 53    0D    E0  0102   226           bbs     #atr_v_scalvar,r3,96$   ;if scalar varying set, write it all
                   22 A4    08 BC   B0  0106   227           movw    @fromaddr(ap),rab$w_rsz(r4) ;set current size
                      28 A4    02    C0  010B   228           addl    #2,rab$l_rbf(r4)        ;bump address so we don't write length
                             04    11  010F   229           brb     100$                    ;cont
                      22 A4    02    A0  0111   230 96$:     addw    #2,rab$w_rsz(r4)        ;include size of vcha
                                         0115   231 ;
                                         0115   232 ; process key option.
                                         0115   233 ;
                        06   6C    D1  0115   234 100$:    cmpl    (ap),#6                 ;key specified?
                             2C    19  0118   235           blss    120$                    ;if lss, then no, continue
                      50    10 AC    D0  011A   236           movl    keyaddr(ap),r0          ;copy address of key
                             26    13  011E   237           beql    120$                    ;if eql, then no key
                      0A 53    08    E0  0120   238           bbs     #atr_v_keyed,r3,110$    ;if file is keyed, continue
                50   00000000'8F    D0  0124   239           movl    #pli$_notkeyd,r0        ;set not keyed file
                           00B9    31  012B   240           brw     fail                    ;and fail
                      50    10 AC    9E  012E   241 110$:    movab   keyaddr(ap),r0          ;point to key descr
                   00000000'GF    16  0132   242           jsb     g^pli$$readkey_r6       ;process key
                 5A 00BC C2       05  E0  0138   243           bbs     #fab$v_bio,<fcb_b_fab+fab$b_fac>(r2),160$ ;if blockio, cont
                   00000000'GF    16  013E   244           jsb     g^pli$$smallget         ;find the record
                             52    11  0144   245           brb     160$                    ;cont
                                         0146   246 ;
                                         0146   247 ; set up for sequential rewrite. if current record not correct, find the
                                         0146   248 ; correct one.
                                         0146   249 ;
                        0D   6C    D1  0146   250 120$:    cmpl    (ap),#13                ;options passed?
                             05    19  0149   251           blss    125$                    ;if lss, no, cont
                          28 AC    D5  014B   252           tstl    recidfrom(ap)           ;record id from specified?
                             38    12  014E   253           bneq    140$                    ;if neq, yes, cont
                      0A 53    0A    E0  0150   254 125$:    bbs     #atr_v_seql,r3,130$     ;if seql file, continue
                50   00000000'8F    D0  0154   255           movl    #pli$_notsql,r0         ;set not sequential file
                           0089    31  015B   256           brw     fail                    ;and fail
                 05 00BC C2       05  E1  015E   257 130$:    bbc     #fab$v_bio,<fcb_b_fab+fab$b_fac>(r2),135$ ;if block io
                          38 A2    D4  0164   258           clrl    rab$l_bkt(r2)           ;set for seql write
                             2F    11  0167   259           brb     160$                    ;cont;
                      0A 53    13    E1  0169   260 135$:    bbc     #atr_v_delete,r3,137$   ;if file has delete
                50   00000000'8F    D0  016D   261 136$:    movl    #pli$_nocurrec,r0       ;set no current record
```

```
                0070   31  0174   262             brw     fail                            ;and fail
          19 53   12   E1  0177   263 137$:       bbc     #atr_v_currec,r3,150$           ;if current record not correct
          1E A4   02   90  017B   264             movb    #rab$c_rfa,rab$b_rac(r4)        ;set for rfa access
       10 A4   20 A2   7D  017F   265             movq    fcb_q_rfa(r2) rab$w_rfa(r4)     ;set rfa in rab
          E5 53   19   E0  0184   266             bbs     #atr_v_virgin,r3,136$           ;if file just opened, fail
                       0188   267 140$:       $find   r4                              ;find the pl1 current record
             21 50     E9  0191   268             blbc    r0,170$                         ;if find failed, then fail
          1E A4   00   90  0194   269 150$       movb    #rab$c_seq,rab$b_rac(r4)        ;set sequential access in rab
                       0198   270 ;
                       0198   271 ; Update record.
                       0198   272 ;
       0B 60BC C2   05   E1  0198   273 160$:       bbc     #fab$v_bio,<fcb_b_fab+fab$b_fac>(r2),161$ ;if block io
                       019E   274             $write  r4                              ;do a write
                09     11  01A7   275             brb     165$                            ;cont
                       01A9   276 161$:       $update r4                              ;update the record
             0A 50     E8  01B2   277 165$:       blbs    r0,180$                         ;if lbs, continue
    50   00000000'8F   D0  01B5   278 170$:       movl    #pli$_rmsr,r0                   ;set error code
                0028   31  01BC   279             brw     fail                            ;and fail
 0C A2   021A0000 8F   CA  01BF   280 180$:       bicl    #<atr_m_bfall!atr_m_delete! - ;'deallocate' buffer, clear del,
                       01C7   281                     atr_m_virgin!atr_m_write>,fcb_l_attr(r2) ;virgin and write
 0C A2   00040000 8F   C8  01C7   282             bisl    #atr_m_currec,fcb_l_attr(r2)    ;set current record not correct
       20 A2   10 A4   7D  01CF   283             movq    rab$w_rfa(r4),fcb_q_rfa(r2)     ;save correct current record's rfa
             0D   6C   D1  01D4   284             cmpl    (ap),#13                        ;options passed?
                0A   19  01D7   285             blss    185$                            ;if lss, no
          50   2C AC   D0  01D9   286             movl    recidto(ap),r0                  ;get addr of recid to
                04   13  01DD   287             beql    185$                            ;if eql, not specified
          60   10 A4   7D  01DF   288             movq    rab$w_rfa(r4),,(r0)             ;set recid to
             50   01   D0  01E3   289 185$:       movl    #1,r0                           ;set success
                       04  01E6   290             ret                                     ;return
                       01E7   291
```

```
                                 01E7   293
                          12     01E7   294 fail:   bneq    10$                     ;if neq, then enough parms, continue
     50   00000000'8F     D0     01E9   295         movl    #pli$_parm,r0           ;set not enough parms
                  52     D4     01F0   296         clrl    r2                      ;assume no fcb specified
                     01   6C     D1     01F2   297         cmpl    (ap),#1         ;was there a fcb specified?
                          32     19     01F5   298         blss    40$             ;if lss, then no
                  52   04 AC     D0     01F7   299         movl    fcbaddr(ap),r2  ;get address of fcb
     50   00000000'8F     D1     01FB   300 10$:     cmpl    #pli$_rmsr,r0           ;rms rab error code?
                          25     12     0202   301         bneq    40$             ;if neq, then no, raise error
  08 A4   0001827A 8F     D1     0204   302         cmpl    #rms$_eof,rab$l_sts(r4) ;end of file?
                          0C     12     0209   303         bneq    20$             ;if neq, then no
                          52     DD     020E   304         pushl   r2              ;set fcb addr
                          50     DD     0210   305         pushl   r0              ;set error code
                  00000000'8F     DD     0212   306         pushl   #pli$_endfile   ;raise endfile condition
                          19     11     0218   307         brb     50$             ;continue
                     06   6C     D1     021A   308 20$:     cmpl    (ap),#6         ;key specified?
                          0A     19     021D   309         blss    40$             ;if lss, no, cont
                  53   10 AC     9E     021F   310         movab   keyaddr(ap),r3  ;set addr of key descr for onkey
                  00000000'GF     16     0223   311         jsb     g^pli$$chk_keycnd       ;check for key condition
                          52     DD     0229   312 40$:     pushl   r2              ;set fcb addr
                          50     DD     022B   313         pushl   r0              ;set error code
                  00000000'8F     DD     022D   314         pushl   #pli$_error     ;set error condition
     00000000'GF          03     FB     0233   315 50$:     calls   #3,g^pli$io_error       ;signal the condition
                          04     023A   316         ret                             ;return
                                 023B   317
                                 023B   318
                                 023B   319         .end
```

```
$$.TMP1                 = 00000001
$$.TMP2                 = 00000054
ATR_M_BFALL             = 00020000
ATR_M_CURREC            = 00040000
ATR_M_DELETE            = 00080000
ATR_M_RECORD            = 00001000
ATR_M_UPDATE            = 00000010
ATR_M_VIRGIN            = 02000000
ATR_M_WRITE             = 00100000
ATR_V_BFALL             = 00000011
ATR_V_CURREC            = 00000012
ATR_V_DELETE            = 00000013
ATR_V_KEYED             = 00000008
ATR_V_OPENED            = 00000001
ATR_V_RECORD            = 0000000C
ATR_V_SCALVAR           = 0000000D
ATR_V_SEQL              = 0000000A
ATR_V_UPDATE            = 00000004
ATR_V_VIRGIN            = 00000019
DAT_K_CHAR_VAR          = 0000000B
DIR...                  = 00000001
FAB$B_FAC               = 00000016
FAB$V_BIO               = 00000005
FAIL                      000001E7 R      02
FCBADDR                   00000004
FCB_B_ENVIR               000001C2
FCB_B_ESA                 0000012E
FCB_B_EXTRA               0000003D
FCB_B_FAB                 000000A6
FCB_B_IDENT               00000040
FCB_B_IDENT_NAM           00000042
FCB_B_NAM                 000000F6
FCB_B_NUMKCBS             0000003C
FCB_B_RAB                 00000062
FCB_C_LEN                 000001C2
FCB_C_STRLEN              00000034
FCB_L_ATTR                0000000C
FCB_L_BUF                 00000014
FCB_L_BUF_END             00000018
FCB_L_BUF_PT              0000001C
FCB_L_CNDADDR             000001B2
FCB_L_CONDIT              000001AE
FCB_L_DTTR                00000010
FCB_L_ERROR               00000008
FCB_L_KCB                 00000038
FCB_L_NEXT                00000000
FCB_L_PREVIOUS            00000004
FCB_L_PRN                 00000034
FCB_Q_RFA                 00000020
FCB_W_COLUMN              0000002E
FCB_W_IDENT_LEN           00000040
FCB_W_LINE                00000030
FCB_W_LINESIZE            0000002A
FCB_W_PAGE                00000032
FCB_W_PAGESIZE            0000002C
FCB_W_REVISION            00000028
FROMADDR                  00000008
FROMLEN                   0000000C
FROMTYP                   0000000E
FXCADDR                   00000030
FXCLEN                    00000034
FXCTYP                    00000036
KEYADDR                   00000010
KEYLEN                    00000014
KEYNUM                    0000001C
KEYTYP                    00000018
MATCHGEQ                  00000024
MATCHGTR                  00000020
PLI$$BYTESIZE             ********    X   02
PLI$$CHK_KEYCND           ********    X   02
PLI$$FXDCTLFROM           ********    X   02
PLI$$KEYNUM               ********    X   02
PLI$$MATCHGEQ             ********    X   02
PLI$$MATCHGTR             ********    X   02
PLI$$READKEY_R6           ********    X   02
PLI$$RECIDFROM            ********    X   02
PLI$$SMALLGET             ********    X   02
PLI$$VALRECIDTO           ********    X   02
PLI$IO_ERROR              ********    X   02
PLI$OPEN                  ********    X   02
PLI$REWRITE               00000000 RG     02
PLI$_ENDFILE              ********      X 02
PLI$_ERROR                ********      X 02
PLI$_NOCURREC             ********      X 02
PLI$_NOFROM               ********      X 02
PLI$_NOTKEYD              ********      X 02
PLI$_NOTREC               ********      X 02
PLI$_NOTSQL               ********      X 02
PLI$_NOTUPDATE            ********      X 02
PLI$_OPEN                 ********      X 02
PLI$_PARM                 ********      X 02
PLI$_RMSR                 ********      X 02
RAB$B_RAC               = 0000001E
RAB$C_RFA               = 0C000002
RAB$C_SEQ               = 00000000
RAB$L_BKT               = 00000038
RAB$L_RBF               = 00000028
RAB$L_RHB               = 0000002C
RAB$L_ROP               = 00000004
RAB$L_STS               = 00000008
RAB$M_KGE               = 00200000
RAB$M_KGT               = 00400000
RAB$W_RFA               = 00000010
RAB$W_RSZ               = 00000022
RECIDFROM                 00000028
RECIDTO                   0000002C
RMS$_EOF                = 0001827A
SIZ...                  = 00000001
SYS$FIND                  ********  GX  02
SYS$UPDATE                ********  GX  02
SYS$WRITE                 ********  GX  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	000001C2 (450.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_PLI$CODE	0000023B (571.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	21	00:00:00.08	00:00:00.51
Command processing	86	00:00:00.56	00:00:01.73
Pass 1	204	00:00:07.75	00:00:16.69
Symbol table sort	0	00:00:00.81	00:00:01.39
Pass 2	63	00:00:01.48	00:00:02.96
Symbol table output	12	00:00:00.09	00:00:00.29
Psect synopsis output	2	00:00:00.01	00:00:00.22
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	388	00:00:10.79	00:00:23.80

The working set limit was 1050 pages.
41924 bytes (82 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 710 non-local and 33 local symbols.
319 source lines were read in Pass 1, producing 14 object records in Pass 2.
23 pages of virtual memory were used to define 20 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[PLIRTL.OBJ]PLIRTMAC.MLB;1	6
_255DUA28:[SYSLIB]STARLET.MLB;2	11
TOTALS (all libraries)	17

805 GETS were required to define 17 macros.

There were no errors, warnings or information messages.

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