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      000000    PPPPPPPP    EEEEEEEEEE  NN      NN
PPPPPPPP    LL            IIIIII      000000    PPPPPPPP    EEEEEEEEEE  NN      NN
PP      PP  LL              II      00      00  PP      PP  EE          NN      NN
PP      PP  LL              II      00      00  PP      PP  EE          NN      NN
PP      PP  LL              II      00      00  PP      PP  EE          NNNN    NN
PP      PP  LL              II      00      00  PP      PP  EE          NNNN    NN
PPPPPPPP    LL              II      00      00  PPPPPPPP    EEEEEEEE    NN  NN  NN
PPPPPPPP    LL              II      00      00  PPPPPPPP    EEEEEEEE    NN  NN  NN
PP          LL              II      00      00  PP          EE          NN    NNNN
PP          LL              II      00      00  PP          EE          NN    NNNN
PP          LL              II      00      00  PP          EE          NN      NN      ....
PP          LL              II      00      00  PP          EE          NN      NN      ....
PP          LLLLLLLLLL    IIIIII      000000    PP          EEEEEEEEEE  NN      NN      ....
PP          LLLLLLLLLL    IIIIII      000000    PP          EEEEEEEEEE  NN      NN      ....
```

```
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$open - pl1 runtime open file
0000     2          .ident  /1-007/                                 ; Edit BLS0294
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 opening a file.
0000    37  ;
0000    38  ;
0000    39  ;
0000    40  ; author: c. spitz 18-jul-79
0000    41  ;
0000    42  ; modified:
0000    43  ;
0000    44  ;       Bill Matthews   20-apr-1982
0000    45  ;       V1.4-02:
0000    46  ;               Retry $open if $display comes back with $rms_sup or $rms_support
0000    47  ;
0000    48  ;       1-003   Bill Matthews   29-September-1982
0000    49  ;
0000    50  ;               Invoke macros $defdat and rtshare instead of $defopr and share.
0000    51  ;
0000    52  ;       1-004   Debess Grabazs 31-Oct-1983
0000    53  ;               Use LNM$C_NAMLENGTH for maximum size of equivalence string.
0000    54  ;
0000    55  ;       1-005   Debess Grabazs 6-Dec-1983
0000    56  ;               Include $LNMDEF.
0000    57  ;
```

```
0000     58 ;       1-006   Dave Blickstein 9-Feburary-1984         SPR 11-64002
0000     59 ;
0000     60 ;               Corrected the code that determines the setting of FAB$_SQO.
0000     61 ;               An offset literal was being used instead of a mask literal.
0000     62 ;               This was causing SQO errors when doing RFA access on file
0000     63 ;               opened for READ.
0000     64 ;
0000     65 ;       1-007   Benn Schreiber   6-APR-1984
0000     66 ;               Zero XABs as part of initializing them.
0000     67 ;--
0000     68
0000     69
0000     70 ;
0000     71 ; library macro calls:
0000     72 ;
0000     73         $lnmdef                             ; LNM$_ symbols
0000     74
0000     75 ;
0000     76 ; external definitions
0000     77 ;
0000     78
0000     79         $deffcb                             ;define file control block offsets
0000     80         $defplirtcons                       ;define pl1 run time constants
0000     81         $defdat                             ;define operand node data types
0000     82         $defkcb                             ;define key control block offsets
0000     83         $defenv                             ;define environment block offsets
0000     84         $rabdef                             ;define rab offsets
0000     85         $fabdef                             ;define fab offsets
0000     86         $devdef                             ;define device characteristics
0000     87         $namdef                             ;define nam offsets
0000     88         $xabdef                             ;define common xab offsets
0000     89         $xabsumdef                          ;define summary xab offsets
0000     90         $xabkeydef                          ;define key xab offsets
0000     91         $xabprodef                          ;define protection xab offsets
0000     92         $xabdatdef                          ;define date and time xab offsets
0000     93         $xabfhcdef                          ;define file header char. xab offsets
0000     94
0000     95 ;
0000     96 ; local definitions
0000     97 ;
0000     98
00000004 0000     99 fcbarg          =       4           ;define fcb offset from ap
00000008 0000    100 attrarg         =       8           ;define attr offset from ap
0000000C 0000    101 linesizearg     =       12          ;define linesize offset from ap
00000010 0000    102 pagesizearg     =       16          ;define pagesize offset from ap
00000014 0000    103 titlearg        =       20          ;define title offset from ap
00000018 0000    104 envirarg        =       24          ;define environment block offset from ap
0000     105
0000     106 ;
0000     107 ; local data
0000     108 ;
0000     109
0000     110         rtshare                             ;sharable
0000     111
54 4E 49 52 50 53 59 53  0000    112 sysprint: .ascii        /SYSPRINT/
          4E 49 53 59 53 0008    113 sysin:  .ascii          /SYSIN/
54 55 50 4E 49 24 53 59 53 000D  114 sysinput: .ascii       /SYS$INPUT/
```

```
      54 55 50 54 55 4F 24 53 59 53  0016   115 sysoutput: .ascii       /SYS$OUTPUT/
                                     0020   116
                                     0020   117
```

```
                        0020   119
                        0020   120 ;++
                        0020   121 ; pli$open -- open a file
                        0020   122 ;
                        0020   123 ; functional description:
                        0020   124 ;
                        0020   125 ; This routine opens a pl1 file, with the attributes and options specified
                        0020   126 ; by the call and in the file declaration.  The FCB is updated to reflect
                        0020   127 ; the status of the file, and an RMS create or open is executed, followed
                        0020   128 ; by a connect.
                        0020   129 ;
                        0020   130 ; inputs:
                        0020   131 ;       (ap) = number of arguments (2-6)
                        0020   132 ;       4(ap) = address of the fcb for this file (required)
                        0020   133 ;       8(ap) = open time attributes of the file (required)
                        0020   134 ;       12(ap) = linesize (optional)
                        0020   135 ;       16(ap) = pagesize (optional)
                        0020   136 ;       20(ap) = address of title - char var (optional)
                        0020   137 ;       24(ap) = address of the open environment block (optional)
                        0020   138 ;
                        0020   139 ;       fcb input fields:
                        0020   140 ;               fcb_l_attr(atr_m_opened) = 1 if file opened already
                        0020   141 ;
                        0020   142 ;       NOTE: if the title argument is present, but the option is not wanted,
                        0020   143 ;       the length word or the address specified must be 0. the declared title
                        0020   144 ;       will be used if either is 0.
                        0020   145 ;
                        0020   146 ; outputs:
                        0020   147 ;       successful open
                        0020   148 ;       fcb is updated:
                        0020   149 ;               fcb_l_next = address of the next open fcb
                        0020   150 ;               fcb_l_previous = address of the last open fcb
                        0020   151 ;               fcb_w_linesize = the linesize argument, or the default for
                        0020   152 ;                                the device. (stream only)
                        0020   153 ;               fcb_w_pagesize = the pagesize argument, or the default for
                        0020   154 ;                                the device. (stream only)
                        0020   155 ;               fcb_w_column = 1 if output (stream only)
                        0020   156 ;               fcb_w_line = 1 (stream only)
                        0020   157 ;               fcb_w_page = 1 (stream only)
                        0020   158 ;               fcb_l_attr = the attributes argument or'ed with the declared
                        0020   159 ;                            attributes
                        0020   160 ;
                        0020   161 ;--
                   0FFC 0020   162           .entry  pli$open,^m<r2,r3,r4,r5,r6,r7,r8,r9,r10,r11>
                        0022   163
              01    6C   D1 0022   164           cmpl    (ap),#1                 ;if not enough parameters
                    05   14 0025   165           bgtr    10$                     ;then
                    50   D4 0027   166           clrl    r0                      ;signal not enough parameters
                  05CB   31 0029   167           brw     fail                    ;and fail
            56    04 AC   D0 002C   168 10$:     movl    fcbarg(ap),r6           ;get address of fcb
            03    28 A6   B1 0030   169           cmpw    fcb_w_revision(r6), -   ;check revision of fcb
                             0034   170                   #pli$c_version          ;
                    0A   13 0034   171           beql    20$                     ;if eql, cont
      50  00000000'8F   D0 0036   172           movl    #pli$_badrtl,r0         ;set bad version of file system
                  05B7   31 003D   173           brw     fail                    ;and fail
         01 0C A6   01   E1 0040   174 20$:     bbc     #atr_v_opened,fcb_l_attr(r6),30$ ;if file already opened,
                        04 0045   175           ret                             ;and just return
```

```
                58   08 AC    D0  0046   176 30$:    movl    attrarg(ap),r8          ;get open attributes
                58   10 A6    C8  004A   177           bisl    fcb_l_dttr(r6),r8       ;and in declared attributes
                                 004E   178 ;
                                 004E   179 ; add implied and default attributes as follows:
                                 004E   180 ;
                                 004E   181 ;       direct implies record and keyed
                                 004E   182 ;       keyed implies record
                                 004E   183 ;       print implies stream and output
                                 004E   184 ;       sequential implies record
                                 004E   185 ;       update implies record
                                 004E   186 ;       stream and output and pl1 identifier = "SYSPRINT" implies print
                                 004E   187 ;
                                 004E   188 ;       stream is the default for stream or record
                                 004E   189 ;       input is the default for input or output or update
                                 004E   190 ;       sequential is the default if record is specified
                                 004E   191 ;
                 07 58    09     E1  004E   192           bbc     #atr_v_direct,r8,40$    ;if direct
          58  00001100 8F     C8  0052   193           bisl    #<atr_m_record!atr_m_keyed>,r8 ;then imply record and keyed
          58  00001510 8F     D3  0059   194 40$:      bitl    #<atr_m_keyed! -        ;keyed or
                                 0060   195                   atr_m_record! -         ;record or
                                 0060   196                   atr_m_seql! -           ;sequential or
                                 0060   197                   atr_m_update>,r8        ;update
                     17   13    0060   198           beql    50$                     ;implies
          58  00001000 8F     C8  0062   199           bisl    #atr_m_record,r8        ;record
          58  00000600 8F     D3  0069   200           bitl    #<atr_m_seql!atr_m_direct>,r8 ;if neither seql or direct
                     07   12    0070   201           bneq    50$                     ;specified, then
          58  00000400 8F     C8  0072   202           bisl    #atr_m_seql,r8          ;default to sequential
                 07 58    07     E1  0079   203 50$:      bbc     #atr_v_print,r8,60$     ;if print
          58  00000820 8F     C8  007D   204           bisl    #<atr_m_stream!atr_m_output>,r8 ;then imply stream and output
          58  00001800 8F     D3  0084   205 60$:      bitl    #<atr_m_stream!atr_m_record>,r8 ;if neither stream or record
                     07   12    008B   206           bneq    70$                     ;specified, then
          58  00000800 8F     C8  008D   207           bisl    #atr_m_stream,r8        ;default to stream
          58  00000070 8F     D3  0094   208 70$:      bitl    #<atr_m_input! -        ;if neither input or
                                 009B   209                   atr_m_output! -         ;output or
                                 009B   210                   atr_m_update>,r8        ;update
                     07   12    009B   211           bneq    80$                     ;specified then
          58  00000040 8F     C8  009D   212           bisl    #atr_m_input,r8         ;default to input
                17 58    05     E1  00A4   213 80$:      bbc     #atr_v_output,r8,90$    ;if output and
                13 58    0B     E1  00A8   214           bbc     #atr_v_stream,r8,90$    ;stream specified
40 A6     20  FF4F CF     08     2D  00AC   215           cmpc5   #8,sysprint,#^x20,fcb_w_ident_len(r6),- ;and identifier =
                   42 A6         00B4   216                   fcb_b_ident_nam(r6)     ;SYSPRINT
                     07   12    00B6   217           bneq    90$                     ;then
          58  00000080 8F     C8  00B8   218           bisl    #atr_m_print,r8         ;imply print
                                 00BF   219 ;
                                 00BF   220 ; check consistency of attributes
                                 00BF   221 ;
                                 00BF   222 ; the attributes are inconsistent if:
                                 00BF   223 ;       both record and stream are specified, or
                                 00BF   224 ;       more than one of input, output, or update is specified, or
                                 00BF   225 ;       both direct and sequential are specified
                                 00BF   226 ;
              04 58    0C     E1  00BF   227 90$:      bbc     #atr_v_record,r8,100$   ;if record and
              1B 58    0B     E0  00C3   228           bbs     #atr_v_stream,r8,120$   ;stream both set, fail
              04 58    09     E1  00C7   229 100$:     bbc     #atr_v_direct,r8,110$   ;if direct and
              13 58    0A     E0  00CB   230           bbs     #atr_v_seql,r8,120$     ;sequential both set, fail
                                 00CF   231           assume  atr_v_input eq <atr_v_output+1> ;make sure these bits are
                                 00CF   232           assume  atr_v_input eq <atr_v_update+2> ;where we think they are
```

```
                55   58    03    04    EF   00CF    233 110$:   extzv   #atr_v_update,#3,r8,r5  ;get input, output and update bits
                                         00D4    234         case    type=b,r5,<120$,130$,130$,120$,130$> ;if inconsistent attributes
                 50    00000000'8F    D0   00E2    235 120$:   movl    #pli$_conattr,r0        ;then set status
                             050B    31   00E9    236         brw     fail                    ;and fail
                                     00EC    237 ;
                                     00EC    238 ; set up stream parameters in fcb
                                     00EC    239 ;
                                     00EC    240 ; the line size and page size options are validated and stored in the fcb.
                                     00EC    241 ; the column position is set to 1.  the page number and line number are both
                                     00EC    242 ; set to 1.
                                     00EC    243 ;
                                     00EC    244         assume  <fcb_w_linesize+2> eq fcb_w_pagesize
                                     00EC    245         assume  <fcb_w_linesize+4> eq fcb_w_column
                                     00EC    246         assume  <fcb_w_column+2> eq fcb_w_line
                                     00EC    247         assume  <fcb_w_column+4> eq fcb_w_page
                       2A A6    B4   00EC    248 130$:   clrw    fcb_w_linesize(r6)      ;clr linesize
                          02    6C   D1   00EF    249         cmpl    (ap),#2                 ;linesize passed?
                                28   15   00F2    250         bleq    170$                    ;if leq, then no
                          50    0C AC   D0   00F4    251         movl    linesizearg(ap),r0      ;get linesize arg
                                22   13   00F8    252         beql    170$                    ;if eql, ignore it
                                0A   14   00FA    253         bgtr    150$                    ;if gtr, cont
                 50    00000000'8F    D0   00FC    254 140$:   movl    #pli$_linesize,r0       ;set invalid linesize
                             04F1    31   0103    255         brw     fail                    ;and fail
                 58    00001040 8F    D3   0106    256 150$:   bitl    #<atr_m_record!atr_m_input>,r8 ;record or input?
                                E3   12   010D    257         bneq    140$                    ;if neq, yes, fail
                 00007FFF 8F    50   D1   010F    258         cmpl    r0,#32767               ;too big?
                                E4   14   0116    259         bgtr    140$                    ;if gtr, then yes, fail
                          2A A6 50   B0   0118    260         movw    r0,fcb_w_linesize(r6)   ;set linesize in fcb
                          03    6C   D1   011C    261 170$:   cmpl    (ap),#3                 ;pagesize passed?
                                2C   15   011F    262         bleq    220$                    ;if leq, then no
                          50    10 AC   D0   0121    263         movl    pagesizearg(ap),r0      ;get pagesize arg
                                26   13   0125    264         beql    220$                    ;if eql, ignore it
                                0A   14   0127    265         bgtr    190$                    ;if gtr, cont
                50    00000000'8F    D0   0129    266 180$:   movl    #pli$_pagesize,r0       ;set invalid pagesize
                             04C4    31   0130    267         brw     fail                    ;and fail
                58    00001040 8F    D3   0133    268 190$:   bitl    #<atr_m_record!atr_m_input>,r8 ;record or input?
                                ED   12   013A    269         bneq    180$                    ;if neq, yes, fail
                          E9 58 07   E1   013C    270         bbc     #atr_v_print,r8,180$    ;if not print, fail
                00007FFF 8F    50   D1   0140    271         cmpl    r0,#32767               ;too big
                                E0   14   0147    272         bgtr    180$                    ;if gtr, yes, fail
                          2C A6 50   B0   0149    273 210$:   movw    r0,fcb_w_pagesize(r6)   ;set pagesize
                          20 58 07   E1   014D    274 220$:   bbc     #atr_v_print,r8,230$    ;if not print, cont
                             50 01   B0   0151    275         movw    #1,r0                   ;get initial line, page and prn
                          30 A6 50   B0   0154    276         movw    r0,fcb_w_line(r6)       ;init line number
                          32 A6 50   B0   0158    277         movw    r0,fcb_w_page(r6)       ;init page number
                          34 A6 50   D0   015C    278         movl    r0,fcb_l_prn(r6)        ;init prn
                                2C A6 B5   0160    279         tstw    fcb_w_pagesize(r6)      ;pagesize specified?
                                   0C   12   0163    280         bneq    230$                    ;if neq, yes
                00000000'GF    00   FB   0165    281         calls   #0,g^lib$lp_lines       ;get system default pagesize
                 2C A6    50   06   A3   016C    282         subw3   #6,r0,fcb_w_pagesize(r6) ;set default pagesize (-3 top, -3 bot)
                                2E A6 B4   0171    283 230$:   clrw    fcb_w_column(r6)        ;init column
                                14 A6 D4   0174    284         clrl    fcb_l_buf(r6)           ;init buf addr
                                1C A6 D4   0177    285         clrl    fcb_l_buf_pt(r6)        ;init buf pointer
                                18 A6 D4   017A    286         clrl    fcb_l_buf_end(r6)       ;init buf end
                                           017D    287 ;
                                           017D    288 ;link the fab to the rab, the nam to the fab
                                           017D    289 ;
```

```
                5A    62 A6    9E  017D   290 240$:   movab   fcb_b_rab(r6),r10       ;get address of rab
             57    00A6 C6    9E  0181   291         movab   fcb_b_fab(r6),r7        ;get address of fab
             5B    00F6 C6    9E  0186   292         movab   fcb_b_nam(r6),r11       ;get address of nam
               3C AA    57    D0  018B   293         movl    r7,rab$l_fab(r10)       ;plug address of fab into rab
               28 A7    5B    D0  018F   294         movl    r11,fab$l_nam(r7)       ;plug address of nam into fab
               1F A7    02    90  0193   295         movb    #fab$c_var,fab$b_rfm(r7); set record format default
                  1E A7    94  0197   296         clrb    fab$b_rat(r7)           ;set default record attributes
            2C AA    34 A6    9E  019A   297         movab   fcb_l_prn(r6),rab$l_rhb(r10) ;set address of prn in rab
                                 019F   298 ;
                                 019F   299 ; validate title and ident, if title specified, use it as file name.
                                 019F   300 ; otherwise, use ident as file name.
                                 019F   301 ;
                59    40 A6    B0  019F   302         movw    fcb_w_ident_len(r6),r9  ;get length of ident
                      0A    14  01A3   303         bgtr    260$                    ;if <= 0
          50    00000000'8F    D0  01A5   304 250$:   movl    #pli$_fileident,r0      ;then set invalid file identifier
                   0448    31  01AC   305         brw     fail                    ;and fail
                   1F    59    B1  01AF   306 260$:   cmpw    r9,#31                  ;if length > 31
                         F1    14  01B2   307         bgtr    250$                    ;then fail
             2C A7    42 A6    9E  01B4   308         movab   fcb_b_ident_nam(r6),fab$l_fna(r7) ;default file name to ident
               34 A7    59    90  01B9   309         movb    r9,fab$b_fns(r7)        ;copy length to fab
          5E    000000FF 8F    C2  01BD   310         subl    #lnm$c_namlength,sp     ;get room for translate temp
                         5E    DD  01C4   311         pushl   sp                      ;set addr of dest
                000000FF 8F    DD  01C6   312         pushl   #lnm$c_namlength        ;set len of dest
                      42 A6    9F  01CC   313         pushab  fcb_b_ident_nam(r6)     ;set addr of src
                7E    40 A6    3C  01CF   314         movzwl  fcb_w_ident_len(r6),-(sp) ;set len of src
                   50    5E    D0  01D3   315         movl    sp,r0                   ;copy sp
                                 01D6   316         $trnlog_s (r0),,8(r0)           ;translate the ident
                      B8 50    E9  01EA   317         blbc    r0,250$                 ;if lbc, fail
          5E    00000050 8F    C0  01ED   318         addl    #80,sp                  ;clean stack
          00000000'8F    50    D1  01F4   319         cmpl    r0,#ss$_notran          ;did translation succeed?
                         2E    12  01FB   320         bneq    280$                    ;if neq, yes, use it
    40 A6    20    FE06 CF    05  2D  01FD   321         cmpc5   #5,sysin,#^x20,fcb_w_ident_len(r6), - ;identifier = sysin?
                      42 A6        0205
                                 0207   322                 fcb_b_ident_nam(r6)     ;
                         0C    12  0207   323         bneq    270$                    ;if neq, then no
             2C A7    FE00 CF    9E  0209   324         movab   sysinput,fab$l_fna(r7)  ;set filename to sys$input
                   34 A7    09    90  020F   325         movb    #9,fab$b_fns(r7)        ;set filename size to 9
                         16    11  0213   326         brb     280$                    ;continue
    40 A6    20    FDE6 CF    08  2D  0215   327 270$:   cmpc5   #8,sysprint,#^x20,fcb_w_ident_len(r6), -;identifier = sysprint?
                      42 A6        021D
                                 021F   328                 fcb_b_ident_nam(r6)     ;
                         0A    12  021F   329         bneq    280$                    ;if neq, then no
             2C A7    FDF1 CF    9E  0221   330         movab   sysoutput,fab$l_fna(r7) ;set filename to sys$output
                   34 A7    0A    90  0227   331         movb    #10,fab$b_fns(r7)       ;set filename size to 10
                      05    6C    D1  022B   332 280$:   cmpl    (ap),#5                 ;if title specified
                            24    19  022E   333         blss    320$                    ;then
                   52    14 AC    D0  0230   334         movl    titlearg(ap),r2         ;get address of title
                            1E    13  0234   335         beql    320$                    ;if eql, then ignore it
                      59    82    B0  0236   336         movw    (r2)+,r9                ;get size of title
                            0A    18  0239   337         bgeq    300$                    ;if size < 0
             50    00000000'8F    D0  023B   338 290$:   movl    #pli$_title,r0          ;then set invalid title
                      03B2    31  0242   339         brw     fail                    ;and fail
                0080 8F    59    B1  0245   340 300$:   cmpw    r9,#128                 ;if size > 128
                         EF    14  024A   341         bgtr    290$                    ;then set invalid title and fail
                   34 A7    59    90  024C   342         movb    r9,fab$b_fns(r7)        ;set size in fab
                   2C A7    52    D0  0250   343         movl    r2,fab$l_fna(r7)        ;set address in fab
                                 0254   344 ;
```

```
                                 0254   345 ; put address of extended string area and resultant string area
                                 0254   346 ; into name block. assume no xabs needed
                                 0254   347 ;
      0C AB    012E C6    9E   0254   348 320$:   movab   fcb_b_esa(r6),nam$l_esa(r11) ;plug address of esa in nam
      04 AB    012E C6    9F   025A   349         movab   fcb_b_esa(r6),nam$l_rsa(r11) ;plug address of rsa in nam
                  0B AB    94   0260   350         clrb    nam$b_esl(r11)              ;set no esl, yet
                  03 AB    94   0263   351         clrb    nam$b_rsl(r11)              ;set no rsl, yet
                  24 A7    D4   0266   352         clrl    fab$l_xab(r7)               ;set no xabs
                                 0269   353 ;
                                 0269   354 ; fill in file access field of fab
                                 0269   355 ;
               06 58    06    E1   0269   356         bbc     #atr_v_input,r8,330$    ;if input
               16 A7    02    90   026D   357         movb    #fab$m_get,fab$b_fac(r7) ;set get access in fac
                        0E    11   0271   358         brb     350$                    ;else
               06 58    05    E1   0273   359 330$:   bbc     #atr_v_output,r8,340$   ;if output
                        03    90   0277   360         movb    #<fab$m_get!fab$m_put>,- ;set get and put access
                     16 A7         0279   361                 fab$b_fac(r7)           ;in fac
                        04    11   027B   362         brb     350$                    ;else its update
               16 A7    0F    90   027D   363 340$:   movb    #<fab$m_get! -          ;set get
                                 0281   364                 fab$m_put! -            ;put,
                                 0281   365                 fab$m_del! -            ;delete, and
                                 0281   366                 fab$m_upd>,fab$b_fac(r7) ;update in fac
                                 0281   367 ;
                                 0281   368 ; process environment options
                                 0281   369 ;
                  5E    22    C2   0281   370 350$:   subl    #env_k_len,sp           ;get room for env block
                  59    5E    D0   0284   371         movl    sp,r9                   ;save addr of env block
      6E    22    00    6E    00   2C   0287   372         movc5   #0,(sp),#0,#env_k_len,(sp) ;zero it
                        5E    DD   028D   373         pushl   sp                      ;set addr of macro open block
                        00    DD   028F   374         pushl   #0                      ;assume no open environment
                     06    6C    D1   0291   375         cmpl    (ap),#6                 ;open environment?
                        04    19   0294   376         blss    360$                    ;if lss, no
                  6E    18 AC    D0   0296   377         movl    envirarg(ap),(sp)       ;set addr of open environment
                        56    DD   029A   378 360$:   pushl   r6                      ;set addr of fcb
                     08 AE    9F   029C   379         pushab  8(sp)                   ;set addr of addr
                     08 AE    9F   029F   380         pushab  8(sp)                   ;set addr of addr
                     08 AE    9F   02A2   381         pushab  8(sp)                   ;set addr of addr
               0C A6    58    D0   02A5   382         movl    r8,fcb_l_attr(r6)       ;set current file attributes
   00000000'GF       06    FB   02A9   383         calls   #6,g^pli$$envir         ;process environment
               58    0C A6    D0   02B0   384         movl    fcb_l_attr(r6),r8       ;get current file attributes
                  01 50    E8   02B4   385         blbs    r0,370$                 ;if lbs, cont
                        04    02B7   386         ret                             ;error already signaled, just return
                  5B    69    D0   02B8   387 370$:   movl    env_l_status(r9),r11    ;get env status
                        5C    13   02BB   388         beql    570$                    ;if eql, nothing to do, cont
                  5B    03    D3   02BD   389         bitl    #<env_m_create_dat!env_m_expire_dat>,r11 ;do we need a date xab?
                        23    13   02C0   390         beql    380$                    ;if eql, no, cont
                  5E    2C    C2   02C2   391         subl    #xab$k_datlen,sp        ;get room for date xab
      6E    2C    00    6E    00   2C   02C5   392         movc5   #0,(sp),#0,#xab$k_datlen,(sp) ;zero the xab
                        6E    12   90   02CB   393         movb    #xab$c_dat,xab$b_cod(sp) ;set xab type
                     01 AE    2C   90   02CE   394         movb    #xab$k_datlen,xab$b_bln(sp) ;set xab length
                  14 AE    04 A9    7D   02D2   395         movq    env_q_credate(r9),xab$q_cdt(sp) ;set creation date
                  1C AE    0C A9    7D   02D7   396         movq    env_q_expdate(r9),xab$q_edt(sp) ;set expiration date
                  04 AE    24 A7    D0   02DC   397         movl    fab$l_xab(r7),xab$l_nxt(sp) ;set pointer to next xab
                     24 A7    5E    D0   02E1   398         movl    sp,fab$l_xab(r7)        ;set addr of xab in fab
                  5B    30    D3   02E5   399 380$:   bitl    #<env_m_protect!env_m_uic>,r11 ;do we need a protection xab?
                        2F    13   02E8   400         beql    570$                    ;if eql, no, cont
      5E    00000058 8F    C2   02EA   401         subl    #xab$k_prolen,sp        ;get room for protection xab
```

```
                                0319   410 ;
                                0319   411 ; if its a keyed file allocate a summary xab on the stack, fill it in and
                                0319   412 ; plug its address into the fab.
                                0319   413 ;
    04 A7   00000040 8F   C8    0319   414 570$:   bisl    #fab$m_sqo,fab$l_fop(r7) ;set sqo
       58   00008310 8F   D3    0321   415         bitl    #<atr_m_update!atr_m_direct! - ;update or direct or
                                0328   416                 atr_m_keyed!atr_m_recidacc>,r8 ;keyed or record id access set?
                          08    13    0328   417         beql    575$                ;if eql, no, sqo is ok
    04 A7   00000040 8F   CA    032A   418         bicl    #fab$m_sqo,fab$l_fop(r7) ;clr sqo
             19 58    08   E1    0332   419 575$:   bbc     #atr_v_keyed,r8,580$    ;if not keyed, cont
                5E    0C   C2    0336   420         subl    #xab$k_sumlen,sp        ;allocate space for summary xab
6E  0C   00    6E    00   2C    0339   421         movc5   #0,(sp),#0,#xab$k_sumlen,(sp) ;zero it
                6E    16   90    033F   422         movb    #xab$c_sum,xab$b_cod(sp) ;set xab type
             01 AE    0C   90    0342   423         movb    #xab$c_sumlen,xab$b_bln(sp) ;set block length
          04 AE   24 A7   D0    0346   424         movl    fab$l_xab(r7),xab$l_nxt(sp) ;set pointer to next xab
             24 A7   5E   D0    034B   425         movl    sp,fab$l_xab(r7)        ;link summary xab to fab
                                034F   426 ;
                                034F   427 ; open the file if it is input or update, create it if it is output
                                034F   428 ;
             43 58    05   E0    034F   429 580$:   bbs     #atr_v_output,r8,610$   ;if input or update
                5E    2C   C2    0353   430 585$:   subl    #xab$k_fhclen,sp        ;get room for fhc xab
6E  2C   00    6E    00   2C    0356   431         movc5   #0,(sp),#0,#xab$k_fhclen,(sp) ;zero it
                6E    1D   90    035C   432         movb    #xab$c_fhc,xab$b_cod(sp) ;set xab type
             01 AE    2C   90    035F   433         movb    #xab$k_fhclen,xab$b_bln(sp) ;set block length
          04 AE   24 A7   D0    0363   434         movl    fab$l_xab(r7),xab$l_nxt(sp) ;set next xab pointer
             24 A7   5E   D0    0368   435         movl    sp,fab$l_xab(r7)        ;link fhc xab to fab
                                036C   436         $open   r7                      ;then open the file
                14 50    E9    0375   437         blbc    r0,590$                 ;if lbc, error on open
             51   0A AE   B0    0378   438         movw    xab$w_lrl(sp),r1        ;get length of longest record
             5E   2C AE   9E    037C   439         movab   xab$k_fhclen(sp),sp     ;clean stack
             18 A6   51   B1    0380   440         cmpw    r1,fcb_l_buf_end(r6)    ;lrl larger than user requested size?
                     1C   19    0384   441         blss    620$                    ;if lss, no, cont
             18 A6   51   3C    0386   442         movzwl  r1,fcb_l_buf_end(r6)    ;use lrl for buffer size
                     16   11    038A   443         brb     620$                    ;cont
    50   00000000'8F   D0    038C   444 590$:   movl    #pli$_rmsf,r0           ;then set rms fab error code
                0261   31    0393   445 600$:   brw     fail                    ;and fail
                                0396   446 610$:   $create r7                      ;if output, create the file
                EA 50    E9    039F   447         blbc    r0,590$                 ;if error on create, fail
                                03A2   448 ;
                                03A2   449 ; make sure file has desired attributes
                                03A2   450 ;
             03 58    08   E0    03A2   451 620$:   bbs     #atr_v_keyed,r8,640$    ;if not keyed
                   014F   31    03A6   452 630$:   brw     780$                    ;cont
             1D A7    20   91    03A9   453 640$:   cmpb    #fab$c_idx,fab$b_org(r7) ;is it indexed org?
                      F7   12    03AD   454         bneq    630$                    ;if neq, no, cont
             5B   09 AE   9A    03AF   455         movzbl  xab$b_nok(sp),r11       ;get number of keys
    58   00010000 8F   C8    03B3   456         bisl    #atr_m_indexed,r8       ;set indexed file
                5E   59    D0    03BA   457         movl    r9,sp                   ;clean stack, leaving env block
55  5B   0000004C 8F   C5    03BD   458         mull3   #xab$c_keylen,r11,r5    ;get size of key xabs required
```

```
                5E    55    C2  03C5    459          subl    r5,sp                           ;allocate room on the stack
        6E  55  00    6E    00  2C 03C8    460          movc5   #0,(sp),#0,r5,(sp)              ;zero it
                24 A7 5E    D0  03CE    461          movl    sp,fab$l_xab(r7)                ;set addr of xabs in fab
                      55    5E  D0 03D2    462          movl    sp,r5                           ;copy addr of first xab
                      50    D4  03D5    463          clrl    r0                              ;set index to 0
                      0B    11  03D7    464          brb     670$                            ;cont in loop
            55  0000004C 8F    C0  03D9    465 660$:    addl    #xab$c_keylen,r5                ;point to next key xab
                B8 A5     55    D0  03E0    466          movl    r5,<xab$l_nxt-xab$c_keylen>(r5) ;point last key xab to this one
                   65     15    90  03E4    467 670$:    movb    #xab$c_key,xab$b_cod(r5)        ;set xab type
                01 A5  4C 8F    90  03E7    468          movb    #xab$c_keylen,xab$b_bln(r5)     ;set xab length
                   17 A5  50    90  03EC    469          movb    r0,xab$b_ref(r5)                ;set key number
                   E5 50  5B    F2  03F0    470          aoblss  r11,r0,660$                     ;loop
                      04 A5     D4  03F4    471          clrl    xab$l_nxt(r5)                   ;terminate xab list
                                    03F7    472          $display r7                             ;get the key information
                      34 50     E8  0400    473          blbs    r0,675$                         ;if lbs, cont
            00000000'8F 50      D1  0403    474          cmpl    r0,#rms$_sup                    ;failure = not supported over net?
                         13     13  040A    475          beql    673$                            ;if eql, yes, cont
            00000000'8F 50      D1  040C    476          cmpl    r0,#rms$_support                ;failure = not supported over net?
                         0A     13  0413    477          beql    673$                            ;if eql, yes, cont
            50  00000000'8F      D0  0415    478 672$:    movl    #pli$_rmsf,r0                   ;set rms fab error
                      01D8      31  041C    479          brw     fail                            ;and fail
                                    041F    480 673$:    $close  r7                              ;close the file
                      EA 50     E9  0428    481          blbc    r0,672$                         ;if lbc, fail
                                    042B    482          $open   r7                              ;reopen file
                      DE 50     E9  0434    483          blbc    r0,672$                         ;if lbc, fail
                7E    5B    2C  C5  0437    484 675$:    mull3   #kcb_c_len,r11,-(sp)            ;get size of key control blocks
                            5E  DD  043B    485          pushl   sp                              ;set addr of temp
                      04 AE     9F  043D    486          pushab  4(sp)                           ;set addr of size
            00000000'GF 02      FB  0440    487          calls   #2,g^lib$get_vm                 ;get space for the kcbs
                      0A 50     E8  0447    488          blbs    r0,680$                         ;if lbs, cont
            50  00000000'8F      D0  044A    489          movl    #pli$_novirmem,r0               ;set no virtual memory
                      01A3      31  0451    490          brw     fail                            ;and fail
                         55 8ED0  0454    491 680$:    popl    r5                              ;get addr of kcbs
                38 A6    55     D0  0457    492          movl    r5,fcb_l_kcb(r6)                ;set addr of kcbs in fcb
                3C A6    5B     90  045B    493          movb    r11,fcb_b_numkcbs(r6)           ;set number of kcbs in fcb
                50    13 AE     90  045F    494 690$:    movb    xab$b_dtp(sp),r0                ;get data type of this key
                                    0463    495          assume  xab$c_stg eq 0                  ;
                                    0463    496          assume  xab$c_in2 eq 1                  ;
                                    0463    497          assume  xab$c_bn2 eq 2                  ;
                                    0463    498          assume  xab$c_in4 eq 3                  ;
                                    0463    499          assume  xab$c_bn4 eq 4                  ;
                                    0463    500          assume  xab$c_pac eq 5                  ;
                                    0463    501          case    type=b,r0,< -                   ;case on rms data type
                                    0463    502                  700$, -                        ;string
                                    0463    503                  720$, -                        ;int 2
                                    0463    504                  720$, -                        ;bin 2
                                    0463    505                  740$, -                        ;int 4
                                    0463    506                  740$, -                        ;bin 4
                                    0463    507                  750$>                          ;pac
        50  00000000'8F     D0  0473    508          movl    #pli$_inv_key,r0                ;set invalid key type
                      017A      31  047A    509          brw     fail                            ;and fail
                         50     D4  047D    510 700$:    clrl    r0                              ;clear total length
                   65    0A     D0  047F    511          movl    #dat_k_char,kcb_l_dtyp(r5)      ;set char data type
                51    08 A5     9E  0482    512          movab   kcb_w_pos0(r5),r1               ;get addr of pos0 in kcb
                52    1E AE     9E  0486    513          movab   xab$w_pos0(sp),r2               ;get addr of pos0 in xab
                54    2E AE     9E  048A    514          movab   xab$b_siz0(sp),r4               ;get addr of siz0 in xab
                   57    08     D0  048E    515          movl    #8,r7                           ;set max number of segments
```

```
                81   82   B0   0491    516 710$:   movw    (r2)+,(r1)+             ;set position in kcb
                5A   84   9A   0494    517         movzbl  (r4)+,r10               ;get size of segment
                50   5A   C0   0497    518         addl    r10,r0                  ;add size into total size
                81   5A   B0   049A    519         movw    r10,(r1)+               ;set size in kcb
                  F1 57   F5   049D    520         sobgtr  r7,710$                 ;go again
             04 A5   50   D0   04A0    521         movl    r0,kcb_l_prec(r5)       ;set precision in kcb
                28 A5   D4   04A4    522         clrl    kcb_l_zero(r5)          ;set no more segments
                    34   11   04A7    523         brb     760$                    ;cont
             04 A5   0F   D0   04A9    524 720$:   movl    #15,kcb_l_prec(r5)      ;set prec of 15
                65   02   D0   04AD    525 730$:   movl    #dat_k_fix_bin,kcb_l_dtyp(r5) ;set fixb data type
          08 A5   1E AE   B0   04B0    526         movw    xab$w_pos0(sp),kcb_w_pos0(r5) ;set position of key in kcb
          0A A5   2E AE   9B   04B5    527         movzbw  xab$b_siz0(sp),kcb_w_len0(r5) ;set byte size of key in kcb
                    1E   11   04BA    528         brb     755$                    ;cont
             04 A5   1F   D0   04BC    529 740$:   movl    #31,kcb_l_prec(r5)      ;set prec of 31
                    EB   11   04C0    530         brb     730$                    ;cont in common
                65   04   D0   04C2    531 750$:   movl    #dat_k_fix_dec,kcb_l_dtyp(r5) ;set fixd data type in kcb
       50   2E AE   02   85   04C5    532         mulb3   #2,xab$b_siz0(sp),r0    ;get prec of key
                50   97   04CA    533         decb    r0                      ;
             04 A5   50   9A   04CC    534         movzbl  r0,kcb_l_prec(r5)       ;set prec in kcb
          08 A5   1E AE   B0   04D0    535         movw    xab$w_pos0(sp),kcb_w_pos0(r5) ;set position of key in kcb
          0A A5   2E AE   9B   04D5    536         movzbw  xab$b_siz0(sp),kcb_w_len0(r5) ;set byte size of key in kcb
                0C A5   D4   04DA    537 755$:   clrl    kcb_w_pos1(r5)          ;set no more segments
    5E    0000004C 8F   C0   04DD    538 760$:   addl    #xab$c_keylen,sp        ;remove this xab
                55   2C   C0   04E4    539         addl    #kcb_c_len,r5           ;point to next kcb
                  0B 5B   F5   04E7    540         sobgtr  r11,770$                ;go again
             57   00A6 C6   9E   04EA    541         movab   fcb_b_fab(r6),r7        ;reset addr of fab
                5A   62 A6   9E   04EF    542         movab   fcb_b_rab(r6),r10       ;reset addr of rab
                    4D   11   04F3    543         brb     790$                    ;cont
                   FF67  31   04F5    544 770$:   brw     690$                    ;really go again
                         04F8    545
             10 58   10   E1   04F8    546 780$:   bbc     #atr_v_indexed,r8,782$  ;if indexed required
             1D A7   20   91   04FC    547         cmpb    #fab$c_idx,fab$b_org(r7) ;is it indexed org?
                    40   13   0500    548         beql    790$                    ;if eql, yes, cont
       50   00000000'8F   D0   0502    549         movl    #pli$_notindexed,r0     ;set not indexed file
                  00EB 31   0509    550         brw     fail                    ;and fail
             32 58   08   E1   050C    551 782$:   bbc     #atr_v_keyed,r8,790$    ;if not keyed, cont
             1D A7   10   91   0510    552         cmpb    #fab$c_rel,fab$b_org(r7) ;if keyed and file is relative
                    2C   13   0514    553         beql    790$                    ;then cont
          14 40 A7   1C   E1   0516    554         bbc     #dev$v_rnd,fab$l_dev(r7),785$ ;if device not random access, fail
          0F 40 A7   05   E0   051B    555         bbs     #dev$v_sqd,fab$l_dev(r7),785$ ;if device seql block mode, fail
             1F A7   01   91   0520    556         cmpb    #fab$c_fix,fab$b_rfm(r7) ;if fixed length records
                    1C   13   0524    557         beql    790$                    ;then continue
             18 58   0F   E0   0526    558         bbs     #atr_v_recidacc,r8,790$ ;if record id access, cont
          13 16 A7   05   E0   052A    559         bbs     #fab$v_bio,fab$b_fac(r7),790$ ;if block io, cont
                         052F    560 785$:   $close  r7                      ;close the file
       50   00000000'8F   D0   0538    561         movl    #pli$_invforgkey,r0     ;set invalid file organization
                  00B5 31   053F    562         brw     fail                    ;and fail
                         0542    563 ;
                         0542    564 ; connect to the file
                         0542    565 ;
                   24 A7   D4   0542    566 790$:   clrl    fab$l_xab(r7)           ;clear xab pointer
                         0545    567         $connect r10                    ;connect to the file
                   13 50   E8   054E    568         blbs    r0,800$                 ;if error on connect
                         0551    569         $close  r7                      ;then close the file
       50   00000000'8F   D0   055A    570         movl    #pli$_rmsr,r0           ;set rms rab error code
                  0093 31   0561    571         brw     fail                    ;and fail
                         0564    572 ;
```

```
                                0564   573 ; process fixed_control_size_to and file_id_to
                                0564   574 ;
          50   14 A9    D0     0564   575 800$:   movl    env_l_fileidto(r9),r0   ;get addr of file_id_to
                06    13     0568   576           beql    810$                    ;if eql, none
    60  010A C6   16    28     056A   577           movc3   #22,<fcb_b_nam+nam$t_dvi>(r6),(r0) ;set dvi and fid
          50   18 A9    D0     0570   578 810$:   movl    env_l_fxctlto(r9),r0    ;get addr of fixed_control_size_to
                04    13     0574   579           beql    820$                    ;if eql, none
          60   3F A7    9A     0576   580           movzbl  fab$b_fsz(r7),(r0)      ;set fixed_control_size_to
                                057A   581 ;
                                057A   582 ; set pagesize for terms and linesize/buffer size for all
                                057A   583 ;
       0E 40 A7    02    E1     057A   584 820$:   bbc     #dev$v_trm,fab$l_dev(r7),8205$ ;if its a terminal, then
             03    6C    D1     057F   585           cmpl    (ap),#3                 ;pagesize passed?
                06    15     0582   586           bleq    8202$                   ;if leq, then no
          50   10 AC    D0     0584   587           movl    pagesizearg(ap),r0      ;get pagesize arg
                03    12     0588   588           bneq    8205$                   ;if neq, then it was passed.
             2C A6    B4     058A   589 8202$:   clrw    fcb_w_pagesize(r6)      ;default pagesize tc 0 for terms
             36 A7    B5     058D   590 8205$:   tstw    fab$w_mrs(r7)           ;mrs specified?
                05    13     0590   591           beql    8210$                   ;if eql, no, use current buf size
       18 A6    36 A7    B0     0592   592           movw    fab$w_mrs(r7),fcb_l_buf_end(r6) ;use mrs as buf size
          20 58    0B    E1     0597   593 8210$:   bbc     #atr_v_stream,r8,822$   ;if not stream, cont
             2A A6    B5     059B   594           tstw    fcb_w_linesize(r6)      ;linesize specified?
                1B    12     059E   595           bneq    822$                    ;if neq, yes, cont
       07 40 A7    00    E1     05A0   596           bbc     #dev$v_rec,fab$l_dev(r7),821$ ;if its a record device,
       2A A6    3C A7    B0     05A5   597           movw    fab$w_bls(r7),fcb_w_linesize(r6) ;set actual linesize
                0F    12     05AA   598           bneq    822$                    ;cont
       2A A6    18 A6    B0     05AC   599 821$:   movw    fcb_l_buf_end(r6),fcb_w_linesize(r6) ;default to buffer size
          06 58    07    E1     05B1   600           bbc     #atr_v_print,r8,822$    ;if not print file, cont
    2A A6    0084 8F    B0     05B5   601           movw    #132,fcb_w_linesize(r6) ;set default of 132
                                05BB   602 ;
                                05BB   603 ; link the file to the opened file list, and set its status to opened
                                05BB   604 ;
             00000000'GF   16    05BB   605 822$:   jsb     g^pli$link_fcb          ;link fcb onto exit control list
    01AE C6  9F160006 8F   D0    05C1   606           movl    #^x9f160006,fcb_l_condit(r6) ;set up cnd hnd for i/o routines
    01B2 C6  00000000'GF   9E    05CA   607           movab   g^pli$$stream_hnd,fcb_l_cndaddr(r6) ;set stream handler address
             09 58    0B   E0    05D3   608           bbs     #atr_v_stream,r8,830$   ;if stream cont
    01B2 C6  00000000'GF   9E    05D7   609           movab   g^pli$$key_hnd,fcb_l_cndaddr(r6) ;set key handler address
          58  02040002 8F   C8    05E0   610 830$:   bisl    #<atr_m_currec!atr_m_opened!- ;set file opened, cur rec wrong
                                05E7   611                   atr_m_virgin>,r8        ; and in just opened stat
 0C A6    58  0018000D 8F   CB    05E7   612           bicl3   #<atr_m_delete!atr_m_recur! - ;clear delete, recursion flag,
                                05F0   613                   atr_m_comma_exp!atr_m_eof! - ;comma_expected and eof
                                05F0   614                   atr_m_write>,r8,fcb_l_attr(r6)  ;copy atr to fcb
                20 A6    7C    05F0   615           clrq    fcb_q_rfa(r6)           ;clear saved rfa
             50   01    D0     05F3   616           movl    #1,r0                   ;indicate success
                04    05F6   617           ret                                     ;return
                                05F7   618
```

```
                                05F7    620
                     12  12     05F7    621 fail:   bneq    10$                             ;if not enough parms
          50  00000000'8F  D0     05F9    622         movl    #pli$_parm,r0                   ;then set not enough parameters
                     51  D4     0600    623         clrl    r1                              ;assume no fcb specified
                  01  6C  D1     0602    624         cmpl    (ap),#1                         ;if fcb specified
                     18  19     0605    625         blss    20$                             ;then
                56  04  AC  D0     0607    626         movl    fcbarg(ap),r6                   ;get fcb
                08  A6  50  D0     060B    627 10$:    movl    r0,fcb_l_error(r6)              ;put error code in fcb
                   51  56  D0     060F    628         movl    r6,r1                           ;put fcb address in r1
                   0101  C1  95     0612    629         tstb    <fcb_b_nam+nam$b_esl>(r1) ;anything in esl?
                     07  12     0616    630         bneq    20$                             ;if neq, yes, cont
                   00F9  C1  90     0618    631         movb    <fcb_b_nam+nam$b_rsl>(r1),- ;copy rsl to esl
                   0101  C1        061C    632                 <fcb_b_nam+nam$b_esl>(r1) ;
                     51  DD     061F    633 20$:    pushl   r1                              ;set fcb addr
                     50  DD     0621    634         pushl   r0                              ;set error code
          00000000'8F  DD     0623    635         pushl   #pli$_undfile                   ;set undefinedfile condition
     00000000'GF  03  FB     0629    636         calls   #3,g^pli$io_error               ;signal the condition
                     04  0630    637         ret                                     ;return
                         0631    638
                         0631    639         .end
```

```
$$.TMP1                 =  00000001
$$.TMP2                 =  00000057
ATR_M_COMMA_EXP         =  00000004
ATR_M_CURREC            =  00040000
ATR_M_DELETE            =  00080000
ATR_M_DIRECT            =  00000200
ATR_M_EOF               =  00000001
ATR_M_INDEXED           =  00010000
ATR_M_INPUT             =  00000040
ATR_M_KEYED             =  00000100
ATR_M_OPENED            =  00000002
ATR_M_OUTPUT            =  00000020
ATR_M_PRINT             =  00000080
ATR_M_RECIDACC          =  00008000
ATR_M_RECORD            =  00001000
ATR_M_RECUR             =  00000008
ATR_M_SEQL              =  00000400
ATR_M_STREAM            =  00000800
ATR_M_UPDATE            =  00000010
ATR_M_VIRGIN            =  02000000
ATR_M_WRITE             =  00100000
ATR_V_DIRECT            =  00000009
ATR_V_INDEXED           =  00000010
ATR_V_INPUT             =  00000006
ATR_V_KEYED             =  00000008
ATR_V_OPENED            =  00000001
ATR_V_OUTPUT            =  00000005
ATR_V_PRINT             =  00000007
ATR_V_RECIDACC          =  0000000F
ATR_V_RECORD            =  0000000C
ATR_V_SEQL              =  0000000A
ATR_V_STREAM            =  0000000B
ATR_V_UPDATE            =  00000004
ATTRARG                 =  00000008
DAT_K_CHAR              =  0000000A
DAT_K_FIX_BIN           =  00000002
DAT_K_FIX_DEC           =  00000004
DEV$V_REC               =  00000000
DEV$V_RND               =  0000001C
DEV$V_SQD               =  00000005
DEV$V_TRM               =  00000002
ENVIRARG                =  00000018
ENV_K_LEN                  00000022
ENV_L_FILEIDTO             00000014
ENV_L_FACTLTO              00000018
ENV_L_STATUS               00000000
ENV_M_CREATE_DAT        =  00000001
ENV_M_EXPIRE_DAT        =  00000002
ENV_M_PROTECT           =  00000010
ENV_M_UIC               =  00000020
ENV_Q_CREDATE              00000004
ENV_Q_EXPDATE              0000000C
ENV_W_OWNGROUP             0000001E
ENV_W_OWNMEM               00000020
ENV_W_PROT                 0000001C
FAB$B_FAC               =  00000016
FAB$B_FNS               =  00000034
FAB$B_FSZ               =  0000003F
FAB$B_ORG               =  0000001D
FAB$B_RAT               =  0000001E
FAB$B_RFM               =  0000001F
FAB$C_FIX               =  00000001
FAB$C_IDX               =  00000020
FAB$C_REL               =  00000010
FAB$C_VAR               =  00000002
FAB$L_DEV               =  00000040
FAB$L_FNA               =  0000002C
FAB$L_FOP               =  00000004
FAB$L_NAM               =  00000028
FAB$L_XAB               =  00000024
FAB$M_DEL               =  00000004
FAB$M_GET               =  00000002
FAB$M_PUT               =  00000001
FAB$M_SQO               =  00000040
FAB$M_UPD               =  00000008
FAB$V_BIO               =  00000005
FAB$W_BLS               =  0000003C
FAB$W_MRS               =  00000036
FAIL                       000005F7 R      02
FCBARG                  =  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
KCB_C_LEN                  0000002C
KCB_L_DTYP                 00000000
KCB_L_PREC                 00000004
```

```
KCB_L_ZERO                      00000028
KCB_W_LEN0                      0000000A
KCB_W_LEN1                      0000000E
KCB_W_LEN2                      00000012
KCB_W_LEN3                      00000016
KCB_W_LEN4                      0000001A
KCB_W_LEN5                      0000001E
KCB_W_LEN6                      00000022
KCB_W_LEN7                      00000026
KCB_W_POS0                      00000008
KCB_W_POS1                      0000000C
KCB_W_POS2                      00000010
KCB_W_POS3                      00000014
KCB_W_POS4                      00000018
KCB_W_POS5                      0000001C
KCB_W_POS6                      00000020
KCB_W_POS7                      00000024
LIB$GET_VM                      ********    X   02
LIB$LP_LINES                    ********    X   02
LINESIZEARG                   = 0000000C
LNM$C_NAMLENGTH               = 000000FF
NAM$B_ESL                     = 0000000B
NAM$B_RSL                     = 00000003
NAM$L_ESA                     = 0000000C
NAM$L_RSA                     = 00000004
NAM$T_DVI                     = 00000014
PAGESIZEARG                   = 00000010
PLI$$ENVIR                      ********    X   02
PLI$$KEY_HND                    ********    X   02
PLI$$STREAM_HND                 ********    X   02
PLI$C_VERSION                 = 00000003
PLI$IO_ERROR                    ********    X   02
PLI$LINK_FCB                    ********    X   02
PLI$OPEN                        00000020 RG     02
PLI$_BADRTL                     ********      X 02
PLI$_CONATTR                    ********      X 02
PLI$_FILEIDENT                  ********      X 02
PLI$_INVFORGKEY                 ********      X 02
PLI$_INV_KEY                    ********      X 02
PLI$_LINESIZE                   ********      X 02
PLI$_NOTINDEXED                 ********      X 02
PLI$_NOVIRMEM                   ********      X 02
PLI$_PAGESIZE                   ********      X 02
PLI$_PARM                       ********      X 02
PLI$_RMSF                       ********      X 02
PLI$_RMSR                       ********      X 02
PLI$_TITLE                      ********      X 02
PLI$_UNDFILE                    ********      X 02
RAB$C_FAB                     = 0000003C
RAB$L_RHB                     = 0000002C
RMS$_SUP                        ********      X 02
RMS$_SUPPORT                    ********      X 02
SIZ...                        = 00000001
SS$_NOTRAN                      ********      X 02
SYS$CLOSE                       ********     GX 02
SYS$CONNECT                     ********     GX 02
SYS$CREATE                      ********     GX 02
SYS$DISPLAY                     ********     GX 02
SYS$OPEN                        ********     GX 02
SYS$TRNLOG                      ********     GX 02
SYSIN                           00000008 R      02
SYSINPUT                        0000000D R      02
SYSOUTPUT                       00000016 R      02
SYSPRINT                        00000000 R      02
TITLEARG                      = 00000014
XAB$B_BLN                     = 00000001
XAB$B_COD                     = 00000000
XAB$B_DTP                     = 00000013
XAB$B_NOK                     = 00000009
XAB$B_REF                     = 00000017
XAB$B_SIZ0                    = 0000002E
XAB$C_BN2                     = 00000002
XAB$C_BN4                     = 00000004
XAB$C_DAT                     = 00000012
XAB$C_FHC                     = 0000001D
XAB$C_IN2                     = 00000001
XAB$C_IN4                     = 00000003
XAB$C_KEY                     = 00000015
XAB$C_KEYLEN                  = 0000004C
XAB$C_PAC                     = 00000005
XAB$C_PRO                     = 00000013
XAB$C_STG                     = 00000000
XAB$C_SUM                     = 00000016
XAB$C_SUMLEN                  = 0000000C
XAB$K_DATLEN                  = 0000002C
XAB$K_FHCLEN                  = 0000002C
XAB$K_PROLEN                  = 00000058
XAB$K_SUMLEN                  = 0000000C
XAB$L_NXT                     = 00000004
XAB$Q_CDT                     = 00000014
XAB$Q_EDT                     = 0000001C
XAB$W_GRP                     = 0000000E
XAB$W_LRL                     = 0000000A
XAB$W_MBM                     = 0000000C
XAB$W_POS0                    = 0000001E
XAB$W_PRO                     = 00000008
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	000001C2 (450.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD NOWRT NOVEC BYTE
_PLI$CODE	00000631 (1585.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	10	00:00:00.10	00:00:00.54
Command processing	72	00:00:00.55	00:00:02.56
Pass 1	287	00:00:12.50	00:00:26.07
Symbol table sort	0	00:00:01.43	00:00:03.03
Pass 2	120	00:00:02.60	00:00:05.05
Symbol table output	24	00:00:00.17	00:00:00.37
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	514	00:00:17.39	00:00:37.64

The working set limit was 1200 pages.
69209 bytes (136 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1012 non-local and 80 local symbols.
639 source lines were read in Pass 1, producing 18 object records in Pass 2.
37 pages of virtual memory were used to define 34 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[PLIRTL.OBJ]PLIRTMAC.MLB;1	7
_255DUA28:[SYSLIB]STARLET.MLB;2	24
TOTALS (all libraries)	31

1139 GETS were required to define 31 macros.

There were no errors, warnings or information messages.

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

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