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
```

```
_$2

Sym
---
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI

PLI
PLI
PLI

PLI
PLI
PLI
PLI
PLI
PLI
PLI

PLI
PLI
PLI
PLI

PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
PLI
```

FILEID**PLIGETFIL

N 4

PL
1-

```
PPPPPPPP    LL            IIIIII      GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT  FFFFFFFFFF    IIIIII    LL
PPPPPPPP    LL            IIIIII      GGGGGGGG  EEEEEEEEEE  TTTTTTTTTT  FFFFFFFFFF    IIIIII    LL
PP      PP  LL              II      GG          EE              TT      FF              II      LL
PP      PP  LL              II      GG          EE              TT      FF              II      LL
PP      PP  LL              II      GG          EE              TT      FF              II      LL
PP      PP  LL              II      GG          EE              TT      FF              II      LL
PPPPPPPP    LL              II      GG          EEEEEEEE        TT      FFFFFFFF        II      LL
PPPPPPPP    LL              II      GG          EEEEEEEE        TT      FFFFFFFF        II      LL
PP          LL              II      GG  GGGGGG  EE              TT      FF              II      LL
PP          LL              II      GG  GGGGGG  EE              TT      FF              II      LL
PP          LL              II      GG      GG  EE              TT      FF              II      LL              ....
PP          LL              II      GG      GG  EE              TT      FF              II      LL              ....
PP          LLLLLLLLLL    IIIIII      GGGGGG    EEEEEEEEEE      TT      FF            IIIIII    LLLLLLLLLL      ....
PP          LLLLLLLLLL    IIIIII      GGGGGG    EEEEEEEEEE      TT      FF            IIIIII    LLLLLLLLLL      ....

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            .title  pli$getfile - pl1 runtime get file
0000            .ident  /1-003/                                   ; Edit CGN1003
0000                                                              ; Edit WHM1002
0000
0000      ;
0000      ;*****************************************************************************
0000      ;*                                                                           *
0000      ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000      ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000      ;*  ALL RIGHTS RESERVED.                                                     *
0000      ;*                                                                           *
0000      ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000      ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
0000      ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000      ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000      ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000      ;*  TRANSFERRED.                                                             *
0000      ;*                                                                           *
0000      ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000      ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
0000      ;*  CORPORATION.                                                             *
0000      ;*                                                                           *
0000      ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000      ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000      ;*                                                                           *
0000      ;*                                                                           *
0000      ;*****************************************************************************
0000      ;
0000
0000
0000      ;++
0000      ; facility:
0000      ;
0000      ;       VAX/VMS PL1 runtime library.
0000      ;
0000      ; abstract:
0000      ;
0000      ;       This module contains the pl1 runtime routine for initializing
0000      ;       the runtime system to perform a get from a pl1 stream file.
0000      ;
0000      ; author: c. spitz 4-oct-79
0000      ;
0000      ; modified:
0000      ;
0000      ;
0000      ;       1-002   Bill Matthews   29-September-1982
0000      ;
0000      ;               Invoke macros $defdat and rtshare instead of $defopr and share.
0000      ;
0000      ;       1-003   Chip Nylander   08-August-1983
0000      ;
0000      ;               Initialize the parent pointer with FP when stream block
0000      ;               allocated.
0000      ;
0000      ;--
0000
0000      ;
```

```
0000    ; external definitions
0000    ;
0000            $deffcb                                 ;define file control block
0000            $defstk                                 ;define stack frame offsets
0000            $defstr                                 ;define stream block offsets
0000            $defdat                                 ;define operand node data types
0000            $rabdef                                 ;define rms rab offsets
0000            $fabdef                                 ;define rms fab offsets
0000            $rmsdef                                 ;define rms error codes
0000
0000    ;
0000    ; local definitions
0000    ;
0000
0000            rtshare                                 ;sharable
```

```
                              0000  ;++
                              0000  ; pli$getfile_r6 -- get elements from a stream file
                              0000  ;
                              0000  ; functional description:
                              0000  ;
                              0000  ; This routine initializes the runtime system to perform a GET statement.
                              0000  ; the get statement is compiled into code that sets up the parameters to
                              0000  ; this routine, and then jsb's to it. this routine opens the file if necessary,
                              0000  ; allocates a buffer for the file, allocates and initializes a stream block for
                              0000  ; the GET statement, processes any options specified in the GET statement and
                              0000  ; returns to the inline code.
                              0000  ; following the jsb in the inline code, the compiler has generated code
                              0000  ; that stores the address and size of the next element in r0 and r1. the
                              0000  ; inline code then jsb's to the routine in the pli$getedit or pli$getlist
                              0000  ; modules that processes elements of that data type.
                              0000  ;
                              0000  ; inputs:
                              0000  ;       ap - address of file control block
                              0000  ;       (sp) - address of first element inline code
                              0000  ;       r0 - address of skip option (number of skips)
                              0000  ;       r1 - address of the compiled format (getedit) or 0 (getlist)
                              0000  ;       r2 - no_echo 0=off
                              0000  ;       r3 - no_filter 0=off
                              0000  ;       r4 - address of prompt
                              0000  ;       r5 - purge_type_ahead 0=off
                              0000  ; outputs:
                              0000  ;       r11 - address of the stream block for this get statement
                              0000  ; side effects:
                              0000  ;       ap is preserved
                              0000  ;       r0-r6 are destroyed
                              0000  ;--
                              0000          .enabl  lsb
                              0000
                              0000  pli$getfile_r6::
                              0000  ;
                              0000  ; open file if nec
                              0000  ;
           7E     50     7D   0000          movq    r0,-(sp)                ;save skip and format
     1E 0C AC     01     E0   0003          bbs     #atr_v_opened,fcb_l_attr(ap),10$ ;if file open, cont
      00000840 8F        DD   0008          pushl   #<atr_m_stream+atr_m_input> ;request stream and input
                  5C     DD   000E          pushl   ap                      ;set fcb
    00000000'GF   02     FB   0010          calls   #2,g^pli$open           ;open the file
     0A 0C AC     01     E0   0017          bbs     #atr_v_opened,fcb_l_attr(ap),10$ ;if file opened, cont
 50   00000000'8F        D0   001C          movl    #pli$_open,r0           ;set open failure
            0191         31   0023          brw     fail                    ;and fail
                              0026  ;
                              0026  ; check file attributes
                              0026  ;
     0A 0C AC     03     E3   0026  10$:    bbcs    #atr_v_recur,fcb_l_attr(ap),20$ ;if illegal recursion
 50   00000000'8F        D0   002B          movl    #pli$_recursio,r0       ;set recursive i/o
            0182         31   0032          brw     fail                    ;and fail
      6D   01AE CC       9E   0035  20$:    movab   fcb_l_condit(ap),(fp)   ;set condition handler address
     0A 0C AC     06     E0   003A          bbs     #atr_v_input,fcb_l_attr(ap),40$ ;if input, cont
 50   00000000'8F        D0   003F  30$:    movl    #pli$_notinput,r0       ;set not input file
            016E         31   0046          brw     fail                    ;and fail
     0A 0C AC     0B     E0   0049  40$:    bbs     #atr_v_stream,fcb_l_attr(ap),50$ ;if stream, cont
 50   00000000'8F        D0   004E          movl    #pli$_notstream,r0      ;set not stream file
```

```
                   015F    31  0055          brw     fail                      ;and fail
                               0058      ;
                               0058      ; allocate buffer if needed
                               0058      ;
                 14 AC     D5  0058  50$:    tstl    fcb_l_buf(ap)             ;buffer already allocated?
                    3E     12  005B          bneq    80$                       ;if neq, yes, cont
           03 0C AC    15  E1  005D          bbc     #atr_v_app_comma,fcb_l_attr(ap),60$ ;if append comma
                 2A AC     B6  0062          incw    fcb_w_linesize(ap)        ;add 1 to buffer size
            7E   2A AC     3C  0065  60$:    movzwl  fcb_w_linesize(ap),-(sp) ;get size of a line
          0082 CC   6E     B0  0069          movw    (sp),<fcb_b_rab+rab$w_usz>(ap) ;set it in rab
                    5E     DD  006E          pushl   sp                        ;push addr of temp
                 04 AE     DF  0070          pushal  4(sp)                     ;push addr of size
       00000000'GF   02    FB  0073          calls   #2,g^lib$get_vm           ;get buffer
                 0A 50     E8  007A          blbs    r0,70$                    ;if lbs, then cont
   50      00000000'8F     D0  007D          movl    #pli$_novirmem,r0         ;set no virt mem
                  0130     31  0084          brw     fail                      ;and fail
                    50 8ED0    0087  70$:    popl    r0                        ;get buffer addr
             14 AC  50     D0  008A          movl    r0,fcb_l_buf(ap)          ;store addr of buffer in fcb
             1C AC  50     D0  008E          movl    r0,fcb_l_buf_pt(ap)       ;init buf pointer
             18 AC  50     D0  0092          movl    r0,fcb_l_buf_end(ap)      ;init buf end
          0086 CC   50     D0  0096          movl    r0,<fcb_b_rab+rab$l_ubf>(ap) ;store in rab too.
                               009B      ;
                               009B      ; set locate mode if not app_comma
                               009B      ;
   66 AC     00010000 8F   CA  009B  80$:    bicl    #rab$m_loc,<fcb_b_rab+rab$l_rop>(ap) ;clear locate mode
         08 0C AC   15     E1  00A3          bbc     #atr_v_app_comma,fcb_l_attr(ap),90$ ;if not app_comma, cont
   66 AC     00010000 8F   C8  00A8          bisl    #rab$m_loc,<fcb_b_rab+rab$l_rop>(ap) ;set locate mode
                               00B0      ;
                               00B0      ;allocate a stream block
                               00B0      ;
               50   8E     7D  00B0  90$:    movq    (sp)+,r0                  ;restore skip and format
                    56 8ED0    00B3          popl    r6                        ;save return address
     5E      00000C08 8F   C2  00B6          subl    #str_c_len,sp             ;alloc space for stream block
               5B   5E     D0  00BD          movl    sp,r11                    ;set address of stream block
        14 AB    0408 CB   9E  00C0          movab   str_l_stack_end(r11),str_l_fld_end(r11) ;set end of field
                               00C6      ;
                               00C6      ; initialize format stack
                               00C6      ;
             08 AB  5D     D0  00C6          movl    fp,str_l_parent(r11)      ;set default parent pointer
             04 AB  51     D0  00CA          movl    r1,str_l_fp(r11)          ;set address of format pointer
                    04     13  00CE          beql    95$                       ;if eql, cont
             0C AC  04     CA  00D0          bicl    #atr_m_comma_exp,fcb_l_attr(ap) ;clear comma expected if edit
          0C04 CB   51     D0  00D4  95$:    movl    r1,str_l_stack(r11)       ;copy format pointer to stack
        6B    0C04 CB      9E  00D9          movab   str_l_stack(r11),str_l_sp(r11) ;store format stack pointer
                 0C AB     D4  00DE          clrl    str_l_fs(r11)             ;init status
                               00E1      ;
                               00E1      ; set all options except prompt and skip
                               00E1      ;
           51   66 AC      9E  00E1          movab   <fcb_b_rab+rab$l_rop>(ap),r1 ;get addr of rop
   61      16FFF8FF 8F     CA  00E5          bicl    #^c<rab$m_cco!rab$m_eof! - ;clear unused bits in rop
                               00EC                  rab$m_pmt!rab$m_pta!rab$m_rah! - ;
                               00EC                  rab$m_rne!rab$m_rnf!rab$m_wbh>,(r1) ;
  61    01    18    52     F0  00EC          insv    r2,#rab$v_rne,#1,(r1)     ;set read no echo
  61    01    1B    53     F0  00F1          insv    r3,#rab$v_rnf,#1,(r1)     ;set read no filter
  61    01    1D    55     F0  00F6          insv    r5,#rab$v_pta,#1,(r1)     ;set purge type ahead
                               00FB      ;
                               00FB      ; process skip and prompt options. skips turn into a read with prompt where
```

```
                                00FB     ; the prompt is a <cr><lf>. if a positive number, n, skips was specified, we
                                00FB     ; will do n-1 skips before we do the user specified prompt (if any). this re-
                                00FB     ; quires us to append the users prompt to the <cr><lf> for the last skip. if
                                00FB     ; no skip was specified, but a prompt was specified, we will use the users
                                00FB     ; prompt as is. note that prompting always forces a get, so that whatever is
                                00FB     ; left in the buffer from the last get statement is lost.
                                00FB     ;
                                00FB     ;
 18 AB    00000A0D 8F     D0    00FB     100$:    movl    #^x0a0d,str_b_field(r11) ;set to prompt with <cr><lf>
         0096 CC   02     90    0103              movb    #2,<fcb_b_rab+rab$b_psz>(ap) ;set size in rab
      0092 CC    18 AB    9E    0108              movab   str_b_field(r11),<fcb_b_rab+ - ;set prompt addr
                                010E                      rab$l_pbf>(ap)          ;in rab
 66 AC    40000000 8F     C8    010E              bisl    #rab$m_pmt,<fcb_b_rab+rab$l_rop>(ap) ;set to prompt in rab
                52   50   D0    0116              movl    r0,r2                   ;copy skip addr
                     16   13    0119              beql    115$                    ;if eql, then no skip
                52   62   32    011B              cvtwl   (r2),r2                 ;get number to skip
                     0A   14    011E              bgtr    110$                    ;if gtr, cont
   50    00000000'8F      D0    0120              movl    #pli$_invskip,r0        ;set invalid skip
                   008D   31    0127              brw     fail                    ;and fail
      02 0C AC     19     E1    012A     110$:    bbc     #atr_v_virgin,fcb_l_attr(ap),115$ ;if virgin
                     52   D6    012F              incl    r2                      ;do an extra skip to get first record
 0C AC    00040000 8F     CA    0131     115$:    bicl    #atr_m_currec,fcb_l_attr(ap) ;clear currec in fcb to do skips
                     54   D5    0139              tstl    r4                      ;prompt specified?
                     55   13    013B              beql    160$                    ;if eql, then not there, cont
                51   64   3C    013D              movzwl  (r4),r1                 ;get length
                     50   13    0140              beql    160$                    ;if eql, then ignore it
    000000FD 8F      51   D1    0142              cmpl    r1,#253                 ;is it too big?
                     0A   15    0149              bleq    120$                    ;if leq, no, cont
   50    00000000'8F      D0    014B              movl    #pli$_promptobig,r0     ;set prompt too big
                   0062   31    0152              brw     fail                    ;and fail
                     52   DD    0155     120$:    pushl   r2                      ;save number of skips
          10 AB      51   D0    0157              movl    r1,str_l_fld_pt(r11)    ;save size of prompt
    1A AB    64      84   28    015B              movc3   (r4)+,(r4),str_b_field+2(r11) ;copy prompt to field
                     52 8ED0    0160              popl    r2                      ;restore number of skips
                     19   14    0163              bgtr    140$                    ;if gtr, do n-1 with our prompt
      0092 CC    1A AB    9E    0165              movab   str_b_field+2(r11),<fcb_b_rab+ - ;no skips, so set his prompt
                                016B                      rab$l_pbf>(ap)          ;addr in rab
      0096 CC    10 AB    90    016B              movb    str_l_fld_pt(r11),<fcb_b_rab+ - ;set size of users prompt
                                0171                      rab$b_psz>(ap)          ;in  rab
                     14   11    0171              brb     150$                    ;do the prompt
      33 0C AC     00     E0    0173     130$:    bbs     #atr_v_eof,fcb_l_attr(ap),170$ ;if eof, signal endfile
        00000000'GF       16    0178              jsb     g^pli$$get_rec          ;skip a record
                  F2 52   F5    017E     140$:    sobgtr  r2,130$                 ;if skips left, do them
      0096 CC    10 AB    80    0181              addb    str_l_fld_pt(r11),<fcb_b_rab+ - ;add size of users prompt
                                0187                      rab$b_psz>(ap)          ;to prompt size in rab
      1F 0C AC     00     E0    0187     150$:    bbs     #atr_v_eof,fcb_l_attr(ap),170$ ;if eof, signal endfile
        00000000'GF       16    018C              jsb     g^pli$$get_rec          ;skip a record
                  F2 52   F4    0192     160$:    sobgeq  r2,150$                 ;process skips
 66 AC    40000000 8F     CA    0195              bicl    #rab$m_pmt,<fcb_b_rab+rab$l_rop>(ap) ;clear prompt in rab
 0C AC    00040000 8F     C8    019D              bisl    #atr_m_currec,fcb_l_attr(ap) ;set currec to turn off skipping
        0C AC      08     CA    01A5              bicl    #atr_m_recur,fcb_l_attr(ap) ;clr recursive flag
                                01A9     ;
                                01A9     ; 'return' jump back to main line code
                                01A9     ;
                     66   17    01A9              jmp     (r6)                    ;return to user
                                01AB
                     5C   DD    01AB     170$:    pushl   ap                      ;set fcb addr
```

```
              00   DD  01AD             pushl   #0                      ;set no error code
       00000000'8F   DD  01AF             pushl   #pli$_endfile           ;set endfile condition
              0A   11  01B5             brb     180$                    ;signal the condition
                       01B7       ;
                       01B7       ;failure routine
                       01B7       ;
                       01B7
              5C   DD  01B7       fail:   pushl   ap                      ;set fcb addr
              50   DD  01B9             pushl   r0                      ;set error code
       00000000'8F   DD  01BB             pushl   #pli$_error             ;set error condition
              6D   D4  01C1       180$:   clrl    (fp)                    ;remove the handler in this frame
      0C AC   08   AA  01C3             bicw    #1@atr_v_recur,fcb_l_attr(ap);reset
  00000000'GF   03   FB  01C7             calls   #3,g^pli$io_error       ;signal the error
                   04  01CE             ret                             ;return
                       01CF
                       01CF             .dsabl  lsb
                       01CF             .end
```

```
ATR_M_COMMA_EXP                 = 00000004
ATR_M_CURREC                    = 00040000
ATR_M_INPUT                     = 00000040
ATR_M_RECUR                     = 00000008
ATR_M_STREAM                    = 00000800
ATR_V_APP_COMMA                 = 00000015
ATR_V_EOF                       = 00000000
ATR_V_INPUT                     = 00000006
ATR_V_OPENED                    = 00000001
ATR_V_RECUR                     = 00000003
ATR_V_STREAM                    = 0000000B
ATR_V_VIRGIN                    = 00000019
FAIC                              000001B7 R      02
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
LIB$GET_VM                        ********    X   02
PLI$$GET_REC                      ********    X   02
PLI$GETFILE_R6                    00000000 RG     02
PLI$IO_ERROR                      ********    X   02
PLI$OPEN                          ********    X   02
PLI$_ENDFILE                      ********    X   02
PLI$_ERROR                        ********    X   02
PLI$_INVSKIP                      ********    X   02
PLI$_NOTINPUT                     ********    X   02
PLI$_NOTSTREAM                    ********    X   02
PLI$_NOVIRMEM                     ********    X   02
PLI$_OPEN                         ********    X   02
PLI$_PROMPTOBIG                   ********    X   02

PLI$_RECURSIO                     ********    X   02
RAB$B_PSZ                       = 00000034
RAB$L_PBF                       = 00000030
RAB$L_ROP                       = 00000004
RAB$L_UBF                       = 00000024
RAB$M_CCO                       = 80000000
RAB$M_EOF                       = 00000100
RAB$M_LOC                       = 00010000
RAB$M_PMT                       = 40000000
RAB$M_PTA                       = 20000000
RAB$M_RAH                       = 00000200
RAB$M_RNE                       = 01000000
RAB$M_RNF                       = 08000000
RAB$M_WBH                       = 00000400
RAB$V_PTA                       = 0000001D
RAB$V_RNE                       = 00000018
RAB$V_RNF                       = 0000001B
RAB$W_USZ                       = 00000020
SIZ...                          = 00000001
STK_L_AP                          00000008
STK_L_ARG_LIST                    FFFFFFF8
STK_L_CND_HND                     00000000
STK_L_CND_LST                     FFFFFFF4
STK_L_DISPLAY                     FFFFFFFC
STK_L_FP                          0000000C
STK_L_PC                          00000010
STK_L_PSL                         00000004
STK_L_REGS                        00000014
STR_B_FIELD                       00000018
STR_C_LEN                         00000C08
STR_L_FLD_END                     00000014
STR_L_FLD_PT                      00000010
STR_L_FP                          00000004
STR_L_FS                          0000000C
STR_L_PARENT                      00000008
STR_L_SP                          00000000
STR_L_STACK                       00000C04
STR_L_STACK_END                   00000408
```

```
                                                +----------------+
                                                ! Psect synopsis !
                                                +----------------+

PSECT name                      Allocation          PSECT No.  Attributes
----------                      ----------          ---------  ----------
.  ABS  .                       00000000  (     0.)  00 (  0.)  NOPIC     USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                           FFFFFFFC  (     0.)  01 (  1.)  NOPIC     USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
_PLI$CODE                       000001CF  (   463.)  02 (  2.)    PIC     USR    CON    REL    LCL   SHR   EXE   RD   NOWRT NOVEC LONG

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

Phase                      Page faults    CPU Time       Elapsed Time
-----                      -----------    --------       ------------
Initialization                       9    00:00:00.04    00:00:00.82
Command processing                  67    00:00:00.52    00:00:03.00
Pass 1                             214    00:00:07.26    00:00:21.15
Symbol table sort                    0    00:00:00.82    00:00:02.14
Pass 2                              57    00:00:01.36    00:00:04.71
Symbol table output                 11    00:00:00.09    00:00:00.11
Psect synopsis output                2    00:00:00.03    00:00:00.03
Cross-reference output               0    00:00:00.00    00:00:00.00
Assembler run totals               360    00:00:10.13    00:00:31.96

The working set limit was 1050 pages.
39656 bytes (78 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 712 non-local and 20 local symbols.
261 source lines were read in Pass 1, producing 11 object records in Pass 2.
17 pages of virtual memory were used to define 15 macros.

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

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[PLIRTL.OBJ]PLIRTMAC.MLB;1                    5
_$255$DUA28:[SYSLIB]STARLET.MLB;2                         7
TOTALS (all libraries)                                   12

739 GETS were required to define 12 macros.

There were no errors, warnings or information messages.

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

0308 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY