```
PPPPPPPPPPPP      LLL                IIIIIIIII    RRRRRRRRRRRR      TTTTTTTTTTTTTTT   LLL
PPPPPPPPPPPPP     LLL                IIIIIIIII    RRRRRRRRRRRRR     TTTTTTTTTTTTTTT   LLL
PPPPPPPPPPPPP     LLL                IIIIIIIII    RRRRRRRRRRRRR     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
PPPPPPPPPPPPP     LLL                   III       RRRRRRRRRRRRR         TTT           LLL
PPPPPPPPPPPPP     LLL                   III       RRRRRRRRRRRRR         TTT           LLL
PPPPPPPPPPPPP     LLL                   III       RRRRRRRRRRRRR         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    BBBBBBBB    YY      YY  TTTTTTTTTT    SSSSSSSS    IIIIII    ZZZZZZZZZZ
PPPPPPPP    LL          IIIIII    BBBBBBBB    YY      YY  TTTTTTTTTT    SSSSSSSS    IIIIII    ZZZZZZZZZZ
PP      PP  LL            II      BB      BB  YY      YY      TT        SS            II            ZZ
PP      PP  LL            II      BB      BB  YY      YY      TT        SS            II            ZZ
PP      PP  LL            II      BB      BB    YY  YY        TT        SS            II          ZZ
PP      PP  LL            II      BB      BB    YY  YY        TT        SS            II          ZZ
PPPPPPPP    LL            II      BBBBBBBB        YY          TT          SSSSSS      II        ZZ
PPPPPPPP    LL            II      BBBBBBBB        YY          TT          SSSSSS      II        ZZ
PP          LL            II      BB      BB      YY          TT                SS    II      ZZ
PP          LL            II      BB      BB      YY          TT                SS    II      ZZ
PP          LL            II      BB      BB      YY          TT                SS    II    ZZ            ....
PP          LL            II      BB      BB      YY          TT                SS    II    ZZ            ....
PP          LLLLLLLLLL  IIIIII    BBBBBBBB        YY          TT        SSSSSSSS    IIIIII  ZZZZZZZZZZ    ....
PP          LLLLLLLLLL  IIIIII    BBBBBBBB        YY          TT        SSSSSSSS    IIIIII  ZZZZZZZZZZ    ....

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$$bytesize
0000     2          .ident  /1-002/                                ; Edit WHM1002
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  ;
0000    35  ; abstract:
0000    36  ;
0000    37  ;       This module contains the pl1 runtime routine that determines the
0000    38  ;       byte size of an operand to be read, written or rewritten.
0000    39  ;
0000    40  ; author:
0000    41  ;       c. spitz 10-sep-79
0000    42  ;
0000    43  ; modified:
0000    44  ;
0000    45  ;
0000    46  ;       1-002   Bill Matthews   29-September-1982
0000    47  ;
0000    48  ;               Invoke macros $defdat and rtshare instead of $defopr and share.
0000    49  ;
0000    50  ;--
0000    51
0000    52  ;
0000    53  ; external definitions
0000    54  ;
0000    55
0000    56          $defdat                         ;define pl1 operand node data types
0000    57
```

```
0000     58 ;
0000     59 ; local data
0000     60 ;
0000     61
0000     62          rtshare                         ;sharable
0000     63
```

```
                                0000     65 ;++
                                0000     66 ; pli$$bytesize -- determine byte size of an operand
                                0000     67 ;
                                0000     68 ; functional description:
                                0000     69 ;
                                0000     70 ;       This routine determines the size in bytes of an operand, based upon
                                0000     71 ;       the size field generated by the compiler and the operands data type.
                                0000     72 ;
                                0000     73 ; inputs:
                                0000     74 ;       (ap) - number of arguments (1)
                                0000     75 ;       4(ap) - word containing the size of the operand
                                0000     76 ;       6(ap) - word containing the data type of the operand
                                0000     77 ;
                                0000     78 ; outputs:
                                0000     79 ;       r0 - contains 1 or pli$_invdatyp&^c1
                                0000     80 ;       r1 - contains the length in bytes of the operand
                                0000     81 ;--
                                0000     82
                                0000     83
                         000C   0000     84          .entry  pli$$bytesize,^m<r2,r3>
              50    01    D0   0002     85          movl    #1,r0                   ;assume success
          52    04 AC    3C   0005     86          movzwl  4(ap),r2                ;get size of operand
          53    06 AC    3C   0009     87          movzwl  6(ap),r3                ;get data type of operand
                                000D     88          case    type=b,r3, -            ;case on the data type
                                000D     89                  <10$, -             ;undefined
                                000D     90                  90$, -              ;picture
                                000D     91                  20$, -              ;fixed binary
                                000D     92                  50$, -              ;float binary
                                000D     93                  80$, -              ;fixed decimal
                                000D     94                  110$, -             ;float decimal
                                000D     95                  10$, -              ;complex fixed binary
                                000D     96                  10$, -              ;complex float binary
                                000D     97                  10$, -              ;complex fixed decimal
                                000D     98                  10$, -              ;complex float decimal
                                000D     99                  90$, -              ;character
                                000D    100                  90$, -              ;character varying
                                000D    101                  10$, -              ;bit
                                000D    102                  10$, -              ;bit varying
                                000D    103                  100$, -             ;bit aligned
                                000D    104                  10$, -              ;fixed
                                000D    105                  10$, -              ;offset
                                000D    106                  90$, -              ;area
                                000D    107                  10$, -              ;file
                                000D    108                  10$, -              ;label
                                000D    109                  10$, -              ;entry
                                000D    110                  10$, -              ;format
                                000D    111                  10$, -              ;dope vector
                                000D    112                  90$, -              ;structure
                                000D    113                  10$, -              ;built in function
                                000D    114                  10$, -              ;condition
                                000D    115                  10$, -              ;generic
                                000D    116                  90$>                ;array
53  00000057 8F    D1   0049    117 10$:    cmpl    #<dat_k_structure+64>,r3 ;bit sized structure?
                 48    13   0050    118         beql    100$                    ;if eql, yes, cont
50  00000000'8F    D0   0052    119         movl    #pli$_invdatyp,r0       ;if invalid data type then set data
          50    07    CA   0059    120         bicl    #7,r0                   ;clr status
          50    04    C8   005C    121         bisl    #4,r0                   ;set fatal error
```

```
                      04  005F          ret                     ;type error and return
           51     01  D0  0060  20$:    movl    #1,r1           ;assume it fits in 1 byte
        04 52     04  E1  0063          bbc     #4,r2,40$       ;if precision > 15 then
           51     04  D0  0067          movl    #4,r1           ;set length of 4 bytes
                      04  006A  30$:    ret                     ;return
        FB 52     03  E1  006B  40$:    bbc     #3,r2,30$       ;if precision > 7  then
           51     02  D0  006F          movl    #2,r1           ;set length of 2 bytes
                      04  0072          ret                     ;return
           51     04  D0  0073  50$:    movl    #4,r1           ;for float bin, assume single precision
           35     52  D1  0076          cmpl    r2,#53          ;if precision > 53
                  04  15  0079          bleq    70$             ;then
           51     10  D0  007B          movl    #16,r1          ;set 16 bytes
                      04  007E  60$:    ret                     ;and return
           18     52  D1  007F  70$:    cmpl    r2,#24          ;if precision > 24
                  FA  15  0082          bleq    60$             ;then
           51     08  D0  0084          movl    #8,r1           ;set 8 bytes
                      04  0087          ret                     ;and return
           52     52  9A  0088  80$:    movzbl  r2,r2           ;ignore scale
 52     52    FF  8F  78  008B          ashl    #-1,r2,r2       ;divide number of decimal digits by 2
                  52  D6  0090          incl    r2              ;add 1
           51     52  D0  0092          movl    r2,r1           ;set result
                      04  0095          ret                     ;and return
           51     52  D0  0096  90$:    movl    r2,r1           ;set result
                      04  0099          ret                     ;return
           52     07  C0  009A  100$:   addl    #7,r2           ;add 7 to number of bits
 51     52    FD  8F  78  009D          ashl    #-3,r2,r1       ;divide by 8
                      04  00A2          ret                     ;and return
           51     04  D0  00A3  110$:   movl    #4,r1           ;for flt dec, assume single prec
           07     52  D1  00A6          cmpl    r2,#7           ;prec > 7?
                  01  14  00A9          bgtr    130$            ;if gtr, yes, cont
                      04  00AB  120$:   ret                     ;return
           51     08  D0  00AC  130$:   movl    #8,r1           ;assume double prec
           0F     52  D1  00AF          cmpl    r2,#15          ;prec > 15?
                  F7  15  00B2          bleq    120$            ;if leq, no, it's double
           51     10  D0  00B4          movl    #16,r1          ;set quad prec
                      04  00B7          ret                     ;return
                          00B8  140$:
                          00B8          .end
```

```
DAT_K_STRUCTURE= 00000017
PLI$$BYTESIZE    00000000 RG    01
PLI$_INVDATYP    ********   X   01
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
_PLI$CODE	000000B8 (184.)	01 (1.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	13	00:00:00.02	00:00:02.29
Command processing	69	00:00:00.55	00:00:04.76
Pass 1	65	00:00:00.80	00:00:03.92
Symbol table sort	0	00:00:00.03	00:00:00.18
Pass 2	32	00:00:00.35	00:00:01.24
Symbol table output	1	00:00:00.01	00:00:00.01
Psect synopsis output	1	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	181	00:00:01.78	00:00:12.41

```
The working set limit was 750 pages.
4105 bytes (9 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 29 non-local and 16 local symbols.
159 source lines were read in Pass 1, producing 11 object records in Pass 2.
4 pages of virtual memory were used to define 3 macros.
```

Macro library statistics

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

```
40 GETS were required to define 3 macros.

There were no errors, warnings or information messages.

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

PLICASERR
LIS

PLICONDIT
LIS

DATDEF
SDL

FILEDEF
IN

PLIRTDEF
MAR

PLICHRSTR
LIS

PLICHAR
LIS

PLICLOSE
LIS

PLIRTL

PLIFILDSP
IN

PLIBYTSIZ
LIS

PLIRTL
MAP

ENVCODES
IN

PLIBIT
LIS