

[illegible]

```
PPPPPPPP  LL      IIIIII  PPPPPPPP  UU      UU  TTTTTTTTTT  FFFFFFFFFF  IIIIII  LL
PPPPPPPP  LL      IIIIII  PPPPPPPP  UU      UU  TTTTTTTTTT  FFFFFFFFFF  IIIIII  LL
PP      PP  LL      II      PP      PP  UU      UU  TT      FF      II      LL
PP      PP  LL      II      PP      PP  UU      UU  TT      FF      II      LL
PP      PP  LL      II      PP      PP  UU      UU  TT      FF      II      LL
PPPPPPPP  LL      II      PPPPPPPP  UU      UU  TT      FFFFFFFF  II      LL
PPPPPPPP  LL      II      PPPPPPPP  UU      UU  TT      FFFFFFFF  II      LL
PP      LL      II      PP      UU      UU  TT      FF      II      LL
PP      LL      II      PP      UU      UU  TT      FF      II      LL
PP      LL      II      PP      UU      UU  TT      FF      II      LL
PP      LL      II      PP      UU      UU  TT      FF      II      LL
PP      LL      IIIIII  IIIIII  PP      UU      UU      UU      IIIIII  LL
PP      LLLLLLLLLL  IIIIII  PP      UU      UU      UU      LLLLLLLLLL  ....
PP      LLLLLLLLLL  IIIIII  PP      UU      UU      UU      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
LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
```



```
0000 1      .title pli$putfile
0000 2      .ident /1-003/
0000 3
0000 4
0000 5
0000 6
0000 7
0000 8      ; Edit CGN1003
0000 9      ; Edit WHM1002
0000 10
0000 11 *****
0000 12
0000 13
0000 14
0000 15
0000 16
0000 17
0000 18
0000 19
0000 20
0000 21
0000 22
0000 23
0000 24
0000 25
0000 26
0000 27
0000 28
0000 29
0000 30
0000 31
0000 32
0000 33
0000 34
0000 35
0000 36
0000 37
0000 38
0000 39
0000 40
0000 41
0000 42
0000 43
0000 44
0000 45
0000 46
0000 47
0000 48
0000 49
0000 50
0000 51
0000 52
0000 53
0000 54
0000 55
0000 56
0000 57

      .title pli$putfile
      .ident /1-003/

      ; Edit CGN1003
      ; Edit WHM1002

*****
COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
ALL RIGHTS RESERVED.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
CORPORATION.

DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.

*****

++
facility:
      VAX/VMS PL1 runtime library

abstract:
      This module contains the routines to initialize the runtime system
      for putting elements to a pl1 stream file.

author: c. spitz 28-nov-79

modified:

      1-002  Bill Matthews  29-September-1982
      Invoke macros $defdat and rtshare instead of $defopr and share.

      1-003  Chip Nylander  08-August-1983
      Initialize the parent pointer with FP when stream block
      allocated.

--
;
```

```
0000 58 : external definitions
0000 59 :
0000 60
0000 61      $deffcb      ;define file control block
0000 62      $defdat      ;define operand node data types
0000 63      $defstr      ;define stream block offsets
0000 64      $defputopt    ;define put options block
0000 65      $rabdef      ;define rms rab offsets
0000 66      $fabdef      ;define rms fab offsets
0000 67      $rmsdef      ;define rms error codes
0000 68
0000 69 :
0000 70 : local data
0000 71 :
0000 72
0000 73      rtshare      ;sharable
0000 74
0000 75
0000 76 :++
0000 77 : pli$putfile_r6
0000 78
0000 79 : functional description:
0000 80
0000 81 : This routine initializes the runtime system to put elements to a stream
0000 82 : file. The syntax of the PUT LIST statement is:
0000 83 :     PUT [FILE(fileref)] [SKIP(expr)] [LINE(expr)] [PAGE]
0000 84 :         LIST(out1 [,out2[,out3[...]]) [OPTIONS(option_list)];
0000 85 :         where: out is an expression or [DO iter_var =
0000 86 :             expr TO expr [BY expr]]
0000 87 :             and option_list is a list of:
0000 88 :                 CANCEL_CONTROL_0
0000 89
0000 90 : The syntax of the PUT EDIT statement is the same except that LIST is
0000 91 : replaced by EDIT and a format list follows the element list.
0000 92
0000 93 : inputs:
0000 94 :     ap - address of file control block. must be preserved by inline code
0000 95 :     r0 - line option
0000 96 :     r1 - page option
0000 97 :     r2 - address of skip option
0000 98 :     r3 - cancel control o (0=off)
0000 99 :     r4 - address of the compiled format (for put edit) or 0 (for put list)
0000 100 :     (sp) - address of inline code for 1st element to put
0000 101 : outputs:
0000 102 :     ap - address of the file control block. must be preserved by inline
0000 103 :         code
0000 104 :     r11 - address of stream block. must be preserved by inline code
0000 105 :     r0-r10 are available for use by inline code
0000 106 : side effects:
0000 107 :     r0-r6 are destroyed. the file is opened with the default attributes
0000 108 :     stream and output.
0000 109
0000 110 :--
0000 111
0000 112 pli$putfile_r6::
0000 113 :
0000 114 : allocate a stream block
```



```

      0000 115 :
5E 00000C08 8F 8ED0 0000 116 :      popl    r6                ;save return address
      5B 5E D0 0003 117 :      subl    #str_c_len,sp        ;allocate space for stream block
14 AB 0C04 CB 9E 000A 118 :      movl    sp,r11             ;set address of stream block
      22 AB 50 7D 0013 119 :      movab   str_l_stack(r11),str_l_fld_end(r11) ;set end of field
      2A AB 52 7D 0017 120 :      movq    r0,str_b_field+10(r11) ;save line and page
      32 AB 56 D0 001B 121 :      movq    r2,str_b_field+18(r11) ;save skip and cancel control o
      001F 122 :      movl    r6,str_b_field+26(r11) ;save return address
      001F 123 :
      001F 124 :      initialize format stack and pointer
      001F 125 :
      08 AB 5D D0 001F 126 :      movl    fp,str_l_parent(r11) ;set default parent pointer
      0C AB 02 CA 0023 127 :      bicl    #str_m_edit,str_l_fs(r11) ;assume list directed
      04 AB 54 D0 0027 128 :      movl    r4,str_l_fp(r11)      ;set address of format pointer
      0C AB 04 13 002B 129 :      beql    5$                  ;if eql, list directed
      55 0C04 CB 9E 0031 130 :      bisl    #str_m_edit,str_l_fs(r11) ;set edit directed
      65 54 D0 0036 131 5$:      movab   str_l_stack(r11),r5      ;get address of stack pointer
      6B 55 D0 0039 132 :      movl    r4,r5              ;copy fp to format stack
      003C 133 :      movl    r5,str_l_sp(r11)    ;store sp
      003C 134 :
      003C 135 :      open file if needed
      003C 136 :
1E 0C AC 01 E0 003C 137 :      bbs     #atr_v_opened,fcbl_attr(ap),10$ ;if file open, cont
      00000820 8F DD 0041 138 :      pushl   #<atr_m_stream+atr_m_output> ;request stream and output
      5C DD 0047 139 :      pushl   ap                ;set fcb
00000000'GF 02 FB 0049 140 :      calls   #2,g^plisopen      ;open the file
      0A 0C AC 01 E0 0050 141 :      bbs     #atr_v_opened,fcbl_attr(ap),10$ ;if file open, cont
50 00000000'8F D0 0055 142 :      movl    #plis_open,r0      ;set open failure
      00B6 31 005C 143 :      brw     fail              ;and fail
      005F 144 :
      005F 145 :      check file attributes
      005F 146 :
      0A 0C AC 03 E3 005F 147 10$:      bbcs     #atr_v_recur,fcbl_attr(ap),15$ ;if recursive i/o
50 00000000'8F D0 0064 148 :      movl    #plis_recurio,r0    ;set illegal recursive i/o
      00A7 31 006B 149 :      brw     fail              ;and fail
      6D 01AE CC 9E 006E 150 15$:      movab   fcb_l_condit(ap),(fp) ;set condition handler address
      0A 0C AC 05 E0 0073 151 :      bbs     #atr_v_output,fcbl_attr(ap),20$ ;if output, cont
50 00000000'8F D0 0078 152 :      movl    #plis_notout,r0     ;set not output file
      0093 31 007F 153 :      brw     fail              ;and fail
      0A 0C AC 0B E0 0082 154 20$:      bbs     #atr_v_stream,fcbl_attr(ap),30$ ;if stream, cont
50 00000000'8F D0 0087 155 :      movl    #plis_notstream,r0 ;set not stream file
      0084 31 008E 156 :      brw     fail              ;and fail
      0091 157 :
      0091 158 :      allocate buffer if necessary
      0091 159 :
      14 AC D5 0091 160 30$:      tstl     fcb_l_buf(ap)          ;buffer already allocated?
      37 12 0094 161 :      bneq    50$              ;if neq, then yes, cont
      7E 2A AC 3C 0096 162 :      movzwl   fcb_w_linesize(ap),-(sp) ;push size to allocate
      0084 CC 6E B0 009A 163 :      movw     (sp),2fcb_b_rab+rab$w_rsz>(ap) ;set it in rab
      0080 CC 00 90 009F 164 :      movb     #rab$c_seq,2fcb_b_rab+rab$b_rac>(ap) ;set for seq puts in rab
      5E DD 00A4 165 :      pushl    sp                ;push address of temp
      04 AE DF 00A6 166 :      pushal   4(sp)            ;push address of size
00000000'GF 02 FB 00A9 167 :      calls   #2,g^lib$get_vm     ;get buffer
      0A 50 E8 00B0 168 :      blbs     r0,40$           ;if lbs, continue
50 00000000'8F D0 00B3 169 :      movl    #plis_novirmem,r0   ;set no virt. mem.
      0058 31 00BA 170 :      brw     fail              ;and fail
      51 8ED0 00BD 171 40$:      popl     r1                ;get addr of buf
```

```

      14 AC 51 D0 00C0 172      movl r1,fcbl_buf(ap)      ;store addr in fcb
      1C AC 51 D0 00C4 173      movl r1,fcbl_buf_pt(ap)   ;init buf pt
008A CC 51 D0 00C8 174      movl r1,<fcbl_buf+rab$l_rbf>(ap) ;store in rab
      00CD 175      ;
      00CD 176      ; process page, line, and skip options
      00CD 177      ;
66 AC 01 1F 2E AB F0 00CD 178 50$: insv <str_b_field+22>(r11),#rab$vc_co, -;set cco in rop
      00D4 179      #1,<fcbl_buf+rab$l_rop>(ap) ;
      26 AB D5 00D4 180      tstl <str_b_field+14>(r11) ;was page specified?
      15 13 00D7 181      beql 60$ ;if egl, then no, cont
      0A OC AC 07 E0 00D9 182      bbs #atr_v_print,fcbl_attr(ap),55$ ;if print ok
50 00000000'8F D0 00DE 183 53$: movl #pli$_notprint,r0 ;set not print file
      002D 31 00E5 184      brw fail ;and fail
      00000000'GF 16 00E8 185 55$: jsb g^pli$$putpage_r6 ;do a put page
      51 22 AB D0 00EE 186 60$: movl str_b_field+10(r11),r1 ;was line specified?
      0B 13 00F2 187      beql 70$ ;if egl, then no
      E5 OC AC 07 E1 00F4 188      bbc #atr_v_print,fcbl_attr(ap),53$ ;if not print, error
      00000000'GF 16 00F9 189      jsb g^pli$$putline_r6 ;process the line option
      52 2A AB D0 00FF 190 70$: movl <str_b_field+18>(r11),r2 ;was skip specified?
      09 13 0103 191      beql 100$ ;if egl, no, cont
      52 62 3C 0105 192      movzwl (r2),r2 ;get number to skip
      00000000'GF 16 0108 193      jsb g^pli$$putskip_r2 ;process the skips
      010E 194      ;
      010E 195      ; return to inline code
      010E 196      ;
      OC AC 08 CA 010E 197 100$: bicl #atr_m_recur,fcbl_attr(ap) ;clr recursion flag
      32 BB 17 0112 198      jmp @str_b_field+26(r11) ;'return'
      0115 199      ;
      OC AC 08 AA 0115 200 fail: bicw #1@atr_v_recur,fcbl_attr(ap);reset recursion bit
      6D D4 0119 201      clrl (fp) ;remove any handler
      08 AC 50 D0 011B 202      movl r0,fcbl_error(ap) ;set error in fcb
      5C DD 011F 203      pushl ap ;set fcb address
      50 DD 0121 204      pushl r0 ;set error code
      00000000'8F DD 0123 205      pushl #pli$_error ;set error condition
00000000'GF 03 FL 0129 206      calls #3,g^pli$io_error ;signal error condition
      04 0130 207      ret ;return
      0131 208
      0131 209
      0131 210      .end
```


PLISPUTFILE
Symbol table

J 12

16-SEP-1984 02:23:50 VAX/VMS Macro V04-00 Page 5
6-SEP-1984 11:39:28 [PLIRTL.SRC]PLIPUTFIL.MAR;1 (1)

ATR_M_OUTPUT	=	00000020		
ATR_M_RECUR	=	00000008		
ATR_M_STREAM	=	00000800		
ATR_V_OPENED	=	00000001		
ATR_V_OUTPUT	=	00000005		
ATR_V_PRINT	=	00000007		
ATR_V_RECUR	=	00000003		
ATR_V_STREAM	=	0000000B		
FAIC	=	00000115	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
PLISS\$PUTLINE_R6	*****		X	02
PLISS\$PUTPAGE_R6	*****		X	02
PLISS\$PUTSKIP_R2	*****		X	02
PLIS\$IO_ERROR	*****		X	02
PLIS\$OPEN	*****		X	02
PLIS\$PUTFILE_R6	00000000	RG		02
PLIS\$ERROR	*****		X	02
PLIS\$NOTOUT	*****		X	02
PLIS\$NOTPRINT	*****		X	02
PLIS\$NOTSTREAM	*****		X	02
PLIS\$NOVIRMEM	*****		X	02
PLIS\$OPEN	*****		X	02
PLIS\$RECURSIO	*****		X	02
PUTOPT_B_BITS	00000004			
PUTOPT_C_LEN	00000005			
PUTOPT_L_FXDCTL	00000000			

RAB\$B_RAC	=	0000001E
RAB\$C_SEQ	=	00000000
RAB\$L_RBF	=	00000028
RAB\$L_ROP	=	00000004
RAB\$V_CCO	=	0000001F
RAB\$W_RSZ	=	00000022
SIZ...	=	00000001
STR_B_FIELD	=	00000018
STR_C_LEN	=	000000C8
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	=	000000C4
STR_L_STACK_END	=	00000408
STR_M_EDIT	=	00000002

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

PSECT name	Allocation	PSECT No.	Attributes														
. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE				
\$ABSS	00000C08 (3080.)	01 (1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE				
_PLI\$CODE	00000131 (305.)	02 (2.)	PIC	USR	CON	REL	LCL	SHR	EXE	RD	NOWRT	NOVEC	LONG				

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

Phase	Page faults	CPU Time	Elapsed Time
Initialization	10	00:00:00.06	00:00:00.49
Command processing	72	00:00:00.53	00:00:02.05
Pass 1	197	00:00:07.00	00:00:14.17
Symbol table sort	0	00:00:00.76	00:00:01.62
Pass 2	43	00:00:01.24	00:00:02.86
Symbol table output	8	00:00:00.08	00:00:00.23
Psect synopsis output	2	00:00:00.02	00:00:00.04
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	332	00:00:09.70	00:00:21.47

The working set limit was 900 pages.
38281 bytes (75 pages) of virtual memory were used to buffer the intermediate code.
There were 40 pages of symbol table space allocated to hold 709 non-local and 12 local symbols.
210 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

741 GETS were required to define 12 macros.

There were no errors, warnings or information messages.

MACRO/ENABLE=SUPPRESSION/DISABLE=TRACEBACK/LIS=LISS\$:PLIPUTFIL/OBJ=OBJ\$:PLIPUTFIL MSRC\$:PLIPUTFIL/UPDATE=(ENH\$:PLIPUTFIL)+LIB\$:PLIRTM

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PLIFORMAT
LIS

PLIGETBUF
LIS

PLMSGTXT
LIS

PLIGETDI
LIS

PLIHEEP
LIS

PLIPUTFIL
LIS

PLIRMSBIS
LIS

PLIRECOPT
LIS

PLIOPEN
LIS

PLIREAD
LIS

PLIPROTEC
LIS

PLIREWRT
LIS

PLIGETLIS
LIS

PLIPUTDI
LIS

PLIPKDIU
LIS

PLIPUTLIS
LIS

PLMSGPTR
LIS

PLIPKDIU
LIS

PLIPUTBUF
LIS

PLIGETFIL
LIS