```
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA       SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBB          BBB   AAA         AAA  SSS          RRR         RRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBBBBBBBBBBB       AAA         AAA     SSSSSSSSS    RRRRRRRRRRR       TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR         TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBB          BBB   AAA         AAA           SSS  RRR      RRR      TTT        LLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB       AAA         AAA  SSSSSSSSSSS  RRR         RRR    TTT        LLLLLLLLLLLLLLL
```

FILEID**BPASSDEAS

```
BBBBBBBB    PPPPPPPP      AAAAAA      SSSSSSSS    SSSSSSSS  DDDDDDDD    EEEEEEEEEE    AAAAAA      SSSSSSSS
BBBBBBBB    PPPPPPPP      AAAAAA      SSSSSSSS    SSSSSSSS  DDDDDDDD    EEEEEEEEEE    AAAAAA      SSSSSSSS
BB      BB  PP      PP   AA      AA  SS          SS        DD      DD  EE            AA      AA  SS
BB      BB  PP      PP   AA      AA  SS          SS        DD      DD  EE            AA      AA  SS
BB      BB  PP      PP   AA      AA  SS          SS        DD      DD  EE            AA      AA  SS
BB      BB  PP      PP   AA      AA  SS          SS        DD      DD  EE            AA      AA  SS
BBBBBBBB    PPPPPPPP     AA      AA    SSSSSS      SSSSSS  DD      DD  EEEEEEEE      AA      AA    SSSSSS
BBBBBBBB    PPPPPPPP     AA      AA    SSSSSS      SSSSSS  DD      DD  EEEEEEEE      AA      AA    SSSSSS
BB      BB  PP           AAAAAAAAAA          SS          SS  DD      DD  EE            AAAAAAAAAA          SS
BB      BB  PP           AAAAAAAAAA          SS          SS  DD      DD  EE            AAAAAAAAAA          SS
BB      BB  PP           AA      AA          SS          SS  DD      DD  EE            AA      AA          SS    ....
BB      BB  PP           AA      AA          SS          SS  DD      DD  EE            AA      AA          SS    ....
BBBBBBBB    PP           AA      AA  SSSSSSSS    SSSSSSSS    DDDDDDDD    EEEEEEEEEE    AA      AA  SSSSSSSS      ....
BBBBBBBB    PP           AA      AA  SSSSSSSS    SSSSSSSS    DDDDDDDD    EEEEEEEEEE    AA      AA  SSSSSSSS      ....

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
```

```
;  1          0001  0 !
;  2          0002  0 !<blf/width:80>
;  3          0003  0 !
;  4          0004  0 MODULE bpa$assdeas (                    ! Assign and deassign monitor calls
;  5          0005  0                     IDENT = '1-328'             ! File: BPASSDEAS.B32  Edit: SBL1328
;  6          0006  0                     ) =
;  7          0007  1 BEGIN
;  8          0008  1 !
;  9          0009  1 !*****************************************************************************
; 10          0010  1 !*                                                                           *
; 11          0011  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
; 12          0012  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
; 13          0013  1 !*  ALL RIGHTS RESERVED.                                                      *
; 14          0014  1 !*                                                                           *
; 15          0015  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 16          0016  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 17          0017  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 18          0018  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
; 19          0019  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 20          0020  1 !*  TRANSFERRED.                                                              *
; 21          0021  1 !*                                                                           *
; 22          0022  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 23          0023  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 24          0024  1 !*  CORPORATION.                                                              *
; 25          0025  1 !*                                                                           *
; 26          0026  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 27          0027  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
; 28          0028  1 !*                                                                           *
; 29          0029  1 !*                                                                           *
; 30          0030  1 !*****************************************************************************
; 31          0031  1 !
; 32          0032  1 !<blf/uppercase_key>
; 33          0033  1 !<blf/lowercase_user>
; 34          0034  1 !
; 35          0035  1
; 36          0036  1 !++
; 37          0037  1 ! FACILITY: PDP-11 BASIC-PLUS/VAX
; 38          0038  1 !
; 39          0039  1 ! ABSTRACT:
; 40          0040  1 !
; 41          0041  1 !       This module contains the routines to support ASSIGN, DEASSIGN
; 42          0042  1 !       and DEASSIGN ALL functions.
; 43          0043  1 !
; 44          0044  1 ! ENVIRONMENT: Native mode VAX processor, User mode.
; 45          0045  1 !
; 46          0046  1 ! AUTHOR: Jim Ibbett, CREATION DATE: 18-May-79.
; 47          0047  1 !
; 48          0048  1 ! MODIFIED BY:
; 49          0049  1 !
; 50          0050  1 !       VERSION X01
; 51          0051  1 !
; 52          0052  1 !       2-Jul-79, Jim Ibbett
; 53          0053  1 ! 278   - Put logical names in process table (were in group table
; 54          0054  1 !         during testing).
; 55          0055  1 !
; 56          0056  1 !       5-Jul-79, Jim Ibbett
; 57          0057  1 ! 245   - Fix bug in CNV_DEVNAM.
```

```
;    58          0058  1 !
;    59          0059  1 !       5-Sep-79, V. Eriksson
;    60          0060  1 ! 309   - Modifications to comply with VAX RTL standards.
;    61          0061  1 !
;    62          0062  1 !       10-Sep-79, V.Eriksson
;    63          0063  1 ! 319   - Modifications to comply with VAX RTL standards.
;    64          0064  1 !
;    65          0065  1 !       17-Sep-79, Jim Ibbett
;    66          0066  1 ! 325   - Fix bpa$ascii so null chars are ignored
;    67          0067  1 ! 1-326 - Change require files around.  JBS 03-OCT-1979
;    68          0068  1 ! 1-327 - Make PIC.  JBS 16-OCT-1979
;    69          0069  1 ! 1-328 - Replace signal of BPA$_INTCONCHK with OTS$_FATINTERR.  SBL 16-Mar-1982
;    70          0070  1 !--
;    71          0071  1
;    72          0072  1 !
;    73          0073  1 !<blf/page>
```

```
;    75          0074  1 !
;    76          0075  1 ! SWITCHES:
;    77          0076  1 !
;    78          0077  1
;    79          0078  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    80          0079  1
;    81          0080  1 !
;    82          0081  1 ! TABLE OF CONTENTS:
;    83          0082  1 !
;    84          0083  1
;    85          0084  1 FORWARD ROUTINE
;    86          0085  1     bpa$assign,
;    87          0086  1     bpa$deassign,
;    88          0087  1     bpa$deass_all,
;    89          0088  1     bpa$find_dab : NOVALUE,                    ! M 319
;    90          0089  1     bpa$make_dab,
;    91          0090  1     bpa$release_dab,
;    92          0091  1     bpa$ascii,
;    93          0092  1     cnv_devnam : NOVALUE,                      ! M 319
;    94          0093  1     cnv_ppn,
;    95          0094  1     exit_handler : NOVALUE;
;    96          0095  1
;    97          0096  1 !
;    98          0097  1 ! INCLUDE FILES:
;    99          0098  1 !
;   100          0099  1
;   101          0100  1 REQUIRE 'RTLIN:RTLPSECT';
;   102          0195  1
;   103          0196  1 REQUIRE 'RTLIN:BPASTRUCT';
;   104          0287  1
;   105          0288  1 REQUIRE 'RTLIN:BPAFQBDEF';
;   106          0412  1
;   107          0413  1 REQUIRE 'RTLIN:BPADABDEF';
;   108          0463  1
;   109          0464  1 REQUIRE 'RTLIN:BPAERRDEF';
;   110          0660  1
;   111          0661  1 LIBRARY 'RTLSTARLE';
;   112          0662  1
;   113          0663  1 !
;   114          0664  1 ! MACROS:
;   115          0665  1 !
;   116          0666  1 !       NONE
;   117          0667  1 !
;   118          0668  1 !
;   119          0669  1 ! EQUATED SYMBOLS:
;   120          0670  1 !
;   121          0671  1 !       NONE
;   122          0672  1 !
;   123          0673  1 ! PSECTS:
;   124          0674  1 !
;   125          0675  1 declare_psects (bpa);
;   126          0676  1 !
;   127          0677  1 ! OWN STORAGE:
;   128          0678  1 !
;   129          0679  1
;   130          0680  1 OWN
;   131          0681  1     exit_reason,
```

```
;  132          0682  1      exit_block : VECTOR [4] INITIAL (0, 0, 0, 0),
;  133          0683  1      queue_initted : INITIAL (0),
;  134          0684  1      bpa$al_dabhead : VECTOR [2, LONG];
;  135          0685  1
;  136          0686  1  !
;  137          0687  1  ! EXTERNAL REFERENCES:
;  138          0688  1  !
;  139          0689  1
;  140          0690  1  EXTERNAL ROUTINE
;  141          0691  1      bpa$get_block,
;  142          0692  1      bpa$free_block;
;  143          0693  1
;  144          0694  1  BUILTIN
;  145          0695  1      INSQUE,
;  146          0696  1      REMQUE;
;  147          0697  1
;  148          0698  1  EXTERNAL
;  149          0699  1      bpa$gb_usr_prot : BYTE,
;  150          0700  1      bpa$gb_usr_real : BYTE,
;  151          0701  1      bpa$al_usrppn : VECTOR [, LONG];
;  152          0702  1
;  153          0703  1  EXTERNAL LITERAL
;  154          0704  1      OTS$_FATINTERR;
```

```
;  156          0705  1 GLOBAL ROUTINE bpa$assign (firqb) =                ! M 319
;  157          0706  1
;  158          0707  1 !++
;  159          0708  1 ! FUNCTIONAL DESCRIPTION:
;  160          0709  1 !
;  161          0710  1 !       Routine provides support for RSTS assign command
;  162          0711  1 !       which has various flavours, see below...
;  163          0712  1 !
;  164          0713  1 ! FORMAL PARAMETERS:
;  165          0714  1 !
;  166          0715  1 !       firqb = Pointer to firqb                                ! M 319
;  167          0716  1 !
;  168          0717  1 ! IMPLICIT INPUTS:
;  169          0718  1 !
;  170          0719  1 !       FIRQB contains either:-
;  171          0720  1 !               a) logical device name & real device name,
;  172          0721  1 !               b) ppn,
;  173          0722  1 !               c) protection code,
;  174          0723  1 !       or      d) device name.
;  175          0724  1 !
;  176          0725  1 ! IMPLICIT OUTPUTS:
;  177          0726  1 !
;  178          0727  1 !       NONE
;  179          0728  1 !
;  180          0729  1 ! ROUTINE VALUE:
;  181          0730  1 !
;  182          0731  1 !       Returns TRUE or signals fatal errors.
;  183          0732  1 !
;  184          0733  1 ! SIDE EFFECTS:
;  185          0734  1 !
;  186          0735  1 !       Dependant upon function (a,b,c,d above) :-
;  187          0736  1 !       a) logical name entered into process logical name table,
;  188          0737  1 !       b) default user ppn established,
;  189          0738  1 !       c) default user protection code established,
;  190          0739  1 !   or  d) specified device is allocated to user.
;  191          0740  1 !
;  192          0741  1 !--
;  193          0742  1
;  194          0743  2     BEGIN
;  195          0744  2
;  196          0745  2     MAP                                             ! A 319
;  197          0746  2         firqb : REF $fqb_def;                       ! Defines firqb             ! A 319
;  198          0747  2
;  199          0748  2     LOCAL
;  200          0749  2         ppn : VECTOR [10, BYTE],                    ! ppn string
;  201          0750  2         len : BYTE,                                 ! length of string(s)
;  202          0751  2         sts,                                ! return status from subr calls
;  203          0752  2         dot,                                        ! string pointer
;  204          0753  2         descrip1 : BLOCK [8, BYTE],
;  205          0754  2         descrip2 : BLOCK [8, BYTE],
;  206          0755  2         asc1 : BLOCK [6, BYTE],                     ! buffer for device name
;  207          0756  2         asc2 : BLOCK [6, BYTE];                     ! buffer for device name
;  208          0757  2
;  209          0758  2     descrip2 [dsc$w_length] = 0;
;  210          0759  2     descrip2 [dsc$b_dtype] = dsc$k_dtype_t;
;  211          0760  2     descrip2 [dsc$b_class] = dsc$k_class_s;
;  212          0761  2     descrip2 [dsc$a_pointer] = asc2;
```

```
;  213          0762  2
;  214          0763  2      IF .firqb [fqb$w_fqnam1] NEQU 0
;  215          0764  2      THEN
;  216          0765  2          BEGIN
;  217          0766  2          cnv_devnam (asc2, descrip2 [dsc$w_length], .firqb);     ! M 319
;  218          0767  2          bpa$ascii (.firqb [fqb$w_fqnam1], len, asc1);
;  219          0768  2          descrip1 [dsc$w_length] = 0;
;  220          0769  2          descrip1 [dsc$b_dtype] = dsc$k_dtype_t;
;  221          0770  2          descrip1 [dsc$b_class] = dsc$k_class_s;
;  222          0771  2          descrip1 [dsc$a_pointer] = asc1;
;  223          0772  2          bpa$ascii (.firqb [fqb$w_fqnam2], descrip1 [dsc$w_length], asc1 + 3);
;  224          0773  2          descrip1 [dsc$w_length] = .descrip1 [dsc$w_length] + .len;
;  225          0774  2          sts = $crelog (tblflg = 2, lognam = descrip1, eqlnam = descrip2);
;  226          0775  2                                                  ! M278
;  227          0776  2
;  228          0777  2          IF NOT .sts THEN RETURN SIGNAL (badfuo, 0, .sts);
;  229          0778  2
;  230          0779  2          END
;  231          0780  2      ELSE
;  232          0781  2
;  233          0782  2          IF .firqb [fqb$w_ppn] NEQU 0
;  234          0783  2          THEN
;  235          0784  2              BEGIN
;  236          0785  2              dot = 0;
;  237          0786  2              ppn [.dot] = '[';
;  238          0787  2              dot = .dot + 1;
;  239          0788  2              cnv_ppn (.firqb [fqb$b_proj], len, ppn [.dot]);
;  240          0789  2              dot = .dot + .len;
;  241          0790  2              ppn [.dot] = ',';
;  242          0791  2              dot = .dot + 1;
;  243          0792  2              cnv_ppn (.firqb [fqb$b_prog], len, ppn [.dot]);
;  244          0793  2              dot = .dot + .len;
;  245          0794  2              ppn [.dot] = ']';
;  246          0795  2              dot = .dot + 1;
;  247          0796  2              CH$MOVE (.dot, ppn, .bpa$al_usrppn [1]);
;  248          0797  2              bpa$al_usrppn [0] = .dot;
;  249          0798  2              END
;  250          0799  2          ELSE
;  251          0800  2
;  252          0801  2              IF .firqb [fqb$b_prot_real] NEQU 0
;  253          0802  2              THEN
;  254          0803  2                  BEGIN
;  255          0804  2                  bpa$gb_usr_prot = .firqb [fqb$b_prot_code];
;  256          0805  2                  bpa$gb_usr_real = 1;
;  257          0806  2                  END
;  258          0807  2              ELSE
;  259          0808  2                  BEGIN
;  260          0809  2                  cnv_devnam (asc2, descrip2 [dsc$w_length], .firqb);
;  261          0810  2                                                  ! M 319
;  262          0811  2                  bpa$find_dab (.descrip2 [dsc$w_length],
;  263          0812  2                      .descrip2 [dsc$a_pointer], sts);    ! M 319
;  264          0813  2
;  265          0814  2                  IF .sts EQLU 0
;  266          0815  3                  THEN
;  267          0816  4                      BEGIN
;  268          0817  4                      sts = $alloc (devnam = descrip2);
;  269          0818  4
```

```
;  270          0819  4                  IF NOT .sts
;  271          0820  4                  THEN
;  272          0821  4                      RETURN SIGNAL (badfuo, 0, .sts)
;  273          0822  4                  ELSE
;  274          0823  4                      bpa$make_dab (.descrip2 [dsc$w_length],
;  275          0824  4                          .descrip2 [dsc$a_pointer]);
;  276          0825  4
;  277          0826  3                  END;
;  278          0827  3
;  279          0828  2              END;
;  280          0829  2
;  281          0830  2      RETURN 1;
;  282          0831  1      END;                                    !End of bpa$assign


                                                        .TITLE  BPA$ASSDEAS
                                                        .IDENT  \1-328\

                                                        .PSECT  _BPA$DATA,NOEXE,  PIC,2

                                                00000 EXIT_REASON:
                                                        .BLKB   4
                00000000  00000000  00000000  00000000  00004 EXIT_BLOCK:
                                                        .LONG   0, 0, 0, 0                      ;
                                                00000000  00014 QUEUE_INITTED:
                                                        .LONG   0                               ;
                                                00018 BPA$AL_DABHEAD:
                                                        .BLKB   8

                                                        .EXTRN  BPA$GET_BLOCK, BPA$FREE_BLOCK
                                                        .EXTRN  BPA$GB_OSR_PROT
                                                        .EXTRN  BPA$GB_USR_REAL
                                                        .EXTRN  BPA$AL_USRPPN, OTS$_FATINTERR
                                                        .EXTRN  SYS$CRELOG, SYS$ALLOC

                                                        .PSECT  _BPA$CODE,NOWRT,  SHR,  PIC,2

                                                007C 00000      .ENTRY  BPA$ASSIGN, Save R2,R3,R4,5,R6          ; 0705
                    5E                          34   C2 00002      SUBL2   #52, SP                         ;
          18    AE  010E0000                    8F   D0 00005      MOVL    #17694720, DESCRIP2             ; 0758
          1C    AE          08                  AE   9E 0000D      MOVAB   ASC2, DESCRIP2+4                ; 0761
                    52        04                AC   D0 00012      MOVL    FIRQB, R2                       ; 0763
                              08                A2   B5 00016      TSTW    8(R2)                           ;
                                                5B   13 00019      BEQL    1$                              ;
                                                52   DD 0001B      PUSHL   R2                              ; 0766
                              1C                AE   9F 0001D      PUSHAB  DESCRIP2                        ;
                              10                AE   9F 00020      PUSHAB  ASC2                            ;
          0000V CF                              03   FB 00023      CALLS   #3, CNV_DEVNAM                  ;
                              10                AE   9F 00028      PUSHAB  ASC1                            ; 0767
                              04                AE   9F 0002B      PUSHAB  LEN                             ;
                    7E        08                A2   3C 0002E      MOVZWL  8(R2), -(SP)                    ;
          0000V CF                              03   FB 00032      CALLS   #3, BPA$ASCII                   ;
            20  AE  010E0000                    8F   D0 00037      MOVL    #17694720, DESCRIP1             ; 0768
            24  AE          10                  AE   9E 0003F      MOVAB   ASC1, DESCRIP1+4                ; 0771
                              13                AE   9F 00044      PUSHAB  ASC1+3                          ; 0772
                              24                AE   9F 00047      PUSHAB  DESCRIP1                        ;
                    7E        0A                A2   3C 0004A      MOVZWL  10(R2), -(SP)                   ;
```

```
                0000V  CF              03  FB 0004E          CALLS   #3, BPA$ASCII                                    ;
                       50              6E  9A 00053          MOVZBL  LEN, R0                                          ; 0773
                    20 AE              50  A0 00056          ADDW2   R0, DESCRIP1                                     ;
                       7E              D4  0005A          CLRL    -(SP)                                            ; 0774
                                1C     AE  9F 0005C          PUSHAB  DESCRIP2                                         ;
                                28     AE  9F 0005F          PUSHAB  DESCRIP1                                         ;
                                           02  DD 00062          PUSHL   #2                                               ;
                00000000G       00         04  FB 00064          CALLS   #4, SYS$CRELOG                                   ;
                          04    AE         50  D0 0006B          MOVL    R0, STS                                          ;
                                76     04  AE  E8 0006F          BLBS    STS, 3$                                          ; 0777
                                       00AC  31 00073          BRW     5$                                               ;
                                06     A2  B5 00076  1$:       TSTW    6(R2)                                            ; 0782
                                           5A  13 00079          BEQL    2$                                               ;
                                           56  D4 0007B          CLRL    DOT                                              ; 0785
                             28 AE46       5B  8F 0007D          MOVB    #91, PPN[DOT]                                    ; 0786
                                           56  D6 00083          INCL    DOT                                              ; 0787
                                28 AE46    9F 00085          PUSHAB  PPN[DOT]                                         ; 0788
                                04     AE  9F 00089          PUSHAB  LEN                                              ;
                                07     A2  9A 0008C          MOVZBL  7(R2), -(SP)                                     ;
                0000V  CF              03  FB 00090          CALLS   #3, CNV_PPN                                      ;
                       50              6E  9A 00095          MOVZBL  LEN, R0                                          ; 0789
                       56              50  C0 00098          ADDL2   R0, DOT                                          ;
                             28 AE46       2C  90 0009B          MOVB    #44, PPN[DOT]                                    ; 0790
                                           56  D6 000A0          INCL    DOT                                              ; 0791
                                28 AE46    9F 000A2          PUSHAB  PPN[DOT]                                         ; 0792
                                04     AE  9F 000A6          PUSHAB  LEN                                              ;
                                06     A2  9A 000A9          MOVZBL  6(R2), -(SP)                                     ;
                0000V  CF              03  FB 000AD          CALLS   #3, CNV_PPN                                      ;
                       50              6E  9A 000B2          MOVZBL  LEN, R0                                          ; 0793
                       56              50  C0 000B5          ADDL2   R0, DOT                                          ;
                             28 AE46       5D  8F 000B8          MOVB    #93, PPN[DOT]                                    ; 0794
                                           56  D6 000BE          INCL    DOT                                              ; 0795
                                50 00000000G   00  D0 000C0          MOVL    BPA$AL_USRPPN+4, R0                              ; 0796
                60        28   AE             56  28 000C7          MOVC3   DOT, PPN, (R0)                                   ;
                00000000G       00             56  D0 000CC          MOVL    DOT, BPA$AL_USRPPN                               ; 0797
                                               6C  11 000D3          BRB     7$                                               ; 0782
                                16     A2  95 000D5  2$:       TSTB    22(R2)                                           ; 0801
                                           11  13 000D8          BEQL    4$                                               ;
                00000000G       00     17  A2 90 000DA          MOVB    23(R2), BPA$GB_USR_PROT                          ; 0804
                00000000G       00         01  90 000E2          MOVB    #1, BPA$GB_USR_REA[                              ; 0805
                                           56  11 000E9  3$:       BRB     7$                                               ; 0801
                                           52  DD 000EB  4$:       PUSHL   R2                                               ; 0809
                                1C     AE  9F 000ED          PUSHAB  DESCRIP2                                         ;
                                10     AE  9F 000F0          PUSHAB  ASC2                                             ;
                     0000V  CF             03  FB 000F3          CALLS   #3, CNV_DEVNAM                                   ;
                                04     AE  9F 000F8          PUSHAB  STS                                              ; 0811
                                20     AE  DD 000FB          PUSHL   DESCRIP2+4                                       ; 0812
                          7E   20     AE  3C 000FE          MOVZWL  DESCRIP2, -(SP)                                  ; 0811
                0000V  CF              03  FB 00102          CALLS   #3, BPA$FIND_DAB                                 ;
                                04     AE  D5 00107          TSTL    STS                                              ; 0814
                                           35  12 0010A          BNEQ    7$                                               ;
                                           7E  7C 0010C          CLRQ    -(SP)                                            ; 0817
                                           7E  7C 0010E          CLRQ    -(SP)                                            ;
                                28     AE  9F 00110          PUSHAB  DESCRIP2                                         ;
                00000000G       00         05  FB 00113          CALLS   #5, SYS$ALLOC                                    ;
                          04    AE         50  D0 0011A          MOVL    R0, STS                                          ;
                                13     04  AE  E8 0011E          BLBS    STS, 6$                                          ; 0819
```

```
                                    04    AE  DD 00122 5$:     PUSHL   STS                                    ; 0821
                                          7E  D4 00125         CLRL    -(SP)                                  ;
                                001A8090  8F  DD 00127         PUSHL   #1736848                               ;
                    00000000G   00        03  FB 0012D         CALLS   #3, LIB$SIGNAL                         ;
                                          04  00134         RET                                            ;
                                    1C    AE  DD 00135 6$:     PUSHL   DESCRIP2+4                             ; 0824
                              7E    1C    AE  3C 00138         MOVZWL  DESCRIP2, -(SP)                        ; 0823
                        0000V CF          02  FB 0013C         CALLS   #2, BPA$MAKE_DAB                       ;
                              50          01  D0 00141 7$:     MOVL    #1, R0                                 ; 0830
                                          04  00144         RET                                            ; 0831

; Routine Size:  325 bytes,    Routine Base:  _BPA$CODE + 0000

;  283          0832  1
```

```
;  285     0833  1 GLOBAL ROUTINE bpa$deassign (firqb) =          ! M 319
;  286     0834  1
;  287     0835  1 !++
;  288     0836  1 ! FUNCTIONAL DESCRIPTION:
;  289     0837  1 !
;  290     0838  1 !       Routine provides support for the RSTS deassign command
;  291     0839  1 !       which has several flavours, see below...
;  292     0840  1 !
;  293     0841  1 ! FORMAL PARAMETERS:
;  294     0842  1 !
;  295     0843  1 !       firqb = Pointer to firqb                                  ! M 319
;  296     0844  1 !
;  297     0845  1 ! IMPLICIT INPUTS:
;  298     0846  1 !
;  299     0847  1 !       FIRQB contains either:-
;  300     0848  1 !               a) logical device name,
;  301     0849  1 !               b) ppn,
;  302     0850  1 !               c) protection code,
;  303     0851  1 !       or      d) device name.
;  304     0852  1 !
;  305     0853  1 ! IMPLICIT OUTPUTS:
;  306     0854  1 !
;  307     0855  1 !       NONE
;  308     0856  1 !
;  309     0857  1 ! ROUTINE VALUE:
;  310     0858  1 !
;  311     0859  1 !       Always returns TRUE.
;  312     0860  1 !
;  313     0861  1 ! SIDE EFFECTS:
;  314     0862  1 !
;  315     0863  1 !       Dependant upon function (a,b,c,d above):-
;  316     0864  1 !       a) logical name removed from process logical name table,
;  317     0865  1 !       b) default user ppn cleared,
;  318     0866  1 !       c) default protection cleared,
;  319     0867  1 !   or  d) device is deallocated.
;  320     0868  1 !
;  321     0869  1 !--
;  322     0870  1
;  323     0871  2     BEGIN
;  324     0872  2
;  325     0873  2     MAP                                         ! A 319
;  326     0874  2         firqb : REF $fqb_def;                   ! Defines firqb     ! A 319
;  327     0875  2
;  328     0876  2     LOCAL
;  329     0877  2         sts,
;  330     0878  2         descrip1 : BLOCK [8, BYTE],
;  331     0879  2         descrip2 : BLOCK [8, BYTE],
;  332     0880  2         asc1 : VECTOR [6, BYTE],
;  333     0881  2         asc2 : VECTOR [6, BYTE],
;  334     0882  2         length;
;  335     0883  2
;  336     0884  2     IF .firqb [fqb$w_fqnam1] NEQU 0
;  337     0885  2     THEN
;  338     0886  3         BEGIN
;  339     0887  3         bpa$ascii (.firqb [fqb$w_fqnam1], sts, asc1);
;  340     0888  3         bpa$ascii (.firqb [fqb$w_fqnam2], length, asc1 + 3);
;  341     0889  3         length = .length + .sts;
```

```
;  342          0890  3          descrip1 [dsc$w_length] = .length;
;  343          0891             descrip1 [dsc$b_dtype] = dsc$k_dtype_t;
;  344          0892             descrip1 [dsc$b_class] = dsc$k_class_s;
;  345          0893             descrip1 [dsc$a_pointer] = asc1;
;  346          0894             $dellog (tblflg = 2, lognam = descrip1);        ! M278
;  347          0895             END
;  348          0896         ELSE
;  349          0897
;  350          0898             IF .firqb [fqb$w_ppn] NEQU 0
;  351          0899             THEN
;  352          0900                 bpa$al_usrppn [0] = 0
;  353          0901             ELSE
;  354          0902
;  355          0903                 IF .firqb [fqb$b_prot_real] NEQU 0
;  356          0904                 THEN
;  357          0905                     bpa$gb_usr_real = 0
;  358          0906                 ELSE
;  359          0907                     BEGIN
;  360          0908                     descrip2 [dsc$w_length] = 0;
;  361          0909                     descrip2 [dsc$b_dtype] = dsc$k_dtype_t;
;  362          0910                     descrip2 [dsc$b_class] = dsc$k_class_s;
;  363          0911                     descrip2 [dsc$a_pointer] = asc2;
;  364          0912                     cnv_devnam (asc2, descrip2 [dsc$w_length], .firqb);
;  365          0913                                                     ! M 319
;  366          0914                     bpa$find_dab (.descrip2 [dsc$w_length],
;  367          0915                         .descrip2 [dsc$a_pointer], sts);    ! M 319
;  368          0916
;  369          0917                     IF .sts NEQU 0 THEN bpa$release_dab (.sts);
;  370          0918
;  371          0919                     END;
;  372          0920
;  373          0921  2      RETURN 1;
;  374          0922  1      END;                                    !End of bpa$deassign


                                                .EXTRN  SYS$DELLOG

                                        0004 00000      .ENTRY  BPA$DEASSIGN, Save R2                   ; 0833
                                     5E         28  C2 00002  SUBL2   #40, SP                          ;
                                     52     04  AC  D0 00005  MOVL    FIRQB, R2                       ; 0884
                                         08  A2  B5 00009  TSTW    8(R2)                           ;
                                            41  13 0000C  BEQL    1$                              ;
                                         10  AE  9F 0000E  PUSHAB  ASC1                            ; 0887
                                         08  AE  9F 00011  PUSHAB  STS                             ;
                                     7E      08  A2  3C 00014  MOVZWL  8(R2), -(SP)                 ;
                                0000V CF         03  FB 00018  CALLS   #3, BPA$ASCII                ;
                                         13  AE  9F 0001D  PUSHAB  ASC1+3                          ; 0888
                                         04  AE  9F 00020  PUSHAB  LENGTH                          ;
                                     7E      0A  A2  3C 00023  MOVZWL  10(R2), -(SP)                ;
                                0000V CF         03  FB 00027  CALLS   #3, BPA$ASCII                ;
                                     6E      04  AE  C0 0002C  ADDL2   STS, LENGTH                  ; 0889
                                   20    AE      6E  B0 00030  MOVW    LENGTH, DESCRIP1              ; 0890
                                   22    AE  010E  8F  B0 00034  MOVW    #270, DESCRIP1+2              ; 0891
                                   24    AE      10  AE  9E 0003A  MOVAB   ASC1, DESCRIP1+4              ; 0893
                                            7E  D4 0003F  CLRL    -(SP)                           ; 0894
                                         24  AE  9F 00041  PUSHAB  DESCRIP1                        ;
```

```
                                02  DD 00044        PUSHL   #2                          ;
        00000000G  00           03  FB 00046        CALLS   #3, SYS$DELLOG              ;
                                50  11 0004D        BRB     4$                          ; 0884
                        06      A2  B5 0004F 1$:    TSTW    6(R2)                       ; 0898
                                08  13 00052        BEQL    2$                          ;
               00000000G        00  D4 00054        CLRL    BPA$AL_USRPPN               ; 0900
                                43  11 0005A        BRB     4$                          ;
                        16      A2  95 0005C 2$:    TSTB    22(R2)                      ; 0903
                                08  13 0005F        BEQL    3$                          ;
               00000000G        00  94 00061        CLRB    BPA$GB_USR_REAL             ; 0905
                                36  11 00067        BRB     4$                          ;
           18  AE 010E0000      8F  D0 00069 3$:    MOVL    #17694720, DESCRIP2         ; 0908
           1C  AE        08     AE  9E 00071        MOVAB   ASC2, DESCRIP2+4            ; 0911
                                52  DD 00076        PUSHL   R2                          ; 0912
                        1C      AE  9F 00078        PUSHAB  DESCRIP2                    ;
                        10      AE  9F 0007B        PUSHAB  ASC2                        ;
            0000V  CF           03  FB 0007E        CALLS   #3, CNV_DEVNAM              ;
                        04      AE  9F 00083        PUSHAB  STS                         ; 0914
                        20      AE  DD 00086        PUSHL   DESCRIP2+4                  ; 0915
                   7E   20      AE  3C 00089        MOVZWL  DESCRIP2, -(SP)             ; 0914
            0000V  CF           03  FB 0008D        CALLS   #3, BPA$FIND_DAB            ;
                        04      AE  D5 00092        TSTL    STS                         ; 0917
                                08  13 00095        BEQL    4$                          ;
                        04      AE  DD 00097        PUSHL   STS                         ;
            0000V  CF           01  FB 0009A        CALLS   #1, BPA$RELEASE_DAB         ;
                   50           01  D0 0009F 4$:    MOVL    #1, R0                      ; 0921
                                    04 000A2        RET                                 ; 0922

; Routine Size:  163 bytes,    Routine Base:  _BPA$CODE + 0145

;  375         0923  1
```

```
;  377          0924  1 GLOBAL ROUTINE bpa$deass_all =
;  378          0925  1
;  379          0926  1 !++
;  380          0927  1 ! FUNCTIONAL DESCRIPTION:
;  381          0928  1 !
;  382          0929  1 !       Routine provides support for the RSTS deassign_all
;  383          0930  1 !       command. All allocated devices are deallocated.
;  384          0931  1 !
;  385          0932  1 ! FORMAL PARAMETERS:
;  386          0933  1 !
;  387          0934  1 !       NONE
;  388          0935  1 !
;  389          0936  1 ! IMPLICIT INPUTS:
;  390          0937  1 !
;  391          0938  1 !       bpa$al_dabhead is a double longword header of the DAB deque.
;  392          0939  1 !
;  393          0940  1 ! IMPLICIT OUTPUTS:
;  394          0941  1 !
;  395          0942  1 !       NONE
;  396          0943  1 !
;  397          0944  1 ! ROUTINE VALUE:
;  398          0945  1 !
;  399          0946  1 !       Always returns TRUE.
;  400          0947  1 !
;  401          0948  1 ! SIDE EFFECTS:
;  402          0949  1 !
;  403          0950  1 !       Removes all entries from the DAB deque.
;  404          0951  1 !
;  405          0952  1 !--
;  406          0953  1
;  407          0954  2     BEGIN
;  408          0955  2
;  409          0956  2     UNTIL .bpa$al_dabhead EQLU bpa$al_dabhead DO
;  410          0957  2         bpa$release_dab (.bpa$al_dabhead);
;  411          0958  2
;  412          0959  2     RETURN 1;
;  413          0960  1     END;                                    !End of bpa$deass_all


                                    0004 00000          .ENTRY  BPA$DEASS_ALL, Save R2                  ; 0924
                         52 00000000'  EF   9E 00002    MOVAB   BPA$AL_DABHEAD, R2                      ;
                         50            62   9E 00009 1$: MOVAB   BPA$AL_DABHEAD, R0                      ; 0956
                         50            62   D1 0000C    CMPL    BPA$AL_DABHEAD, R0                      ;
                                       09   13 0000F    BEQL    2$                                      ;
                                       62   DD 00011    PUSHL   BPA$AL_DABHEAD                          ; 0957
                               0000V   CF   01   FB 00013    CALLS   #1, BPA$RELEASE_DAB                 ;
                                       EF   11 00018    BRB     1$                                      ;
                         50            01   D0 0001A 2$: MOVL    #1, R0                                  ; 0959
                                            04 0001D    RET                                             ; 0960

; Routine Size:  30 bytes,    Routine Base:  _BPA$CODE + 01E8

;  414          0961  1
```

```
;  416     0962  1 ROUTINE bpa$find_dab (length, addr, answer) : NOVALUE =          ! M 319
;  417     0963  1
;  418     0964  1 !++
;  419     0965  1 ! FUNCTIONAL DESCRIPTION:
;  420     0966  1 !
;  421     0967  1 !       Routine scans the dab list to find an entry containing
;  422     0968  1 !       an identical device name string to that passed to it.
;  423     0969  1 !
;  424     0970  1 ! FORMAL PARAMETERS:
;  425     0971  1 !
;  426     0972  1 !       length = length of string
;  427     0973  1 !       addr   = address of string
;  428     0974  1 !       answer = pointer to a longword to receive the address of the    ! A 319
;  429     0975  1 !                dab entry if a match is found  ( = 0 if not found).    ! A 319
;  430     0976  1 !
;  431     0977  1 ! IMPLICIT INPUTS:
;  432     0978  1 !
;  433     0979  1 !       NONE
;  434     0980  1 !
;  435     0981  1 ! IMPLICIT OUTPUTS:
;  436     0982  1 !
;  437     0983  1 !       NONE
;  438     0984  1 !
;  439     0985  1 ! ROUTINE VALUE:
;  440     0986  1 !
;  441     0987  1 !       NONE                                                        ! M 319
;  442     0988  1 !
;  443     0989  1 ! SIDE EFFECTS:
;  444     0990  1 !
;  445     0991  1 !       NONE
;  446     0992  1 !
;  447     0993  1 !--
;  448     0994  1
;  449     0995  2     BEGIN
;  450     0996  2
;  451     0997  2     MAP
;  452     0998  2         length : REF VECTOR [, WORD],
;  453     0999  2         addr : REF VECTOR [, LONG],
;  454     1000  2         answer : REF VECTOR [1, LONG];          ! A 319
;  455     1001  2
;  456     1002  2     LOCAL
;  457     1003  2         sts,
;  458     1004  2         next,
;  459     1005  2         dab1 : REF $dab_def,
;  460     1006  2         dab2 : REF $dab_def;
;  461     1007  2
;  462     1008  2     next = 0;
;  463     1009  2     answer [0] = 0;                         ! M 319
;  464     1010  2 !+
;  465     1011  2 ! Initialize the queue.
;  466     1012  2 !-
;  467     1013  2
;  468     1014  3     IF ( NOT .queue_initted)
;  469     1015  2     THEN
;  470     1016  3         BEGIN
;  471     1017  3
;  472     1018  3         LOCAL
```

```
;  473          1019  3          ast_status,
;  474          1020  3          dclexh_status;
;  475          1021  3
;  476          1022  3      ast_status = $setast (enbflg = 0);
;  477          1023  3
;  478          1024  4      IF ( NOT .queue_initted)
;  479          1025  3      THEN
;  480          1026  4          BEGIN
;  481          1027  4          bpa$al_dabhead [0] = bpa$al_dabhead [1] = bpa$al_dabhead [0];
;  482          1028  4          exit_block [1] = exit_handler;
;  483          1029  4          exit_block [2] = 1;
;  484          1030  4          exit_block [3] = exit_reason;
;  485          1031  4          dclexh_status = $dclexh (desblk = exit_block);
;  486          1032  4          queue_initted = 1;
;  487          1033  4          END
;  488          1034  3      ELSE
;  489          1035  3          dclexh_status = 1;
;  490          1036  3
;  491          1037  3      IF (.ast_status EQL ss$_wasset) THEN $setast (enbflg = 1);
;  492          1038  3
;  493          1039  3      IF ( NOT .dclexh_status) THEN SIGNAL (badfuo, 0, .dclexh_status);
;  494          1040  3
;  495          1041  2      END;
;  496          1042  2
;  497          1043  2  IF .bpa$al_dabhead EQLU bpa$al_dabhead THEN RETURN; ! M 319
;  498          1044  2
;  499          1045  2  dab2 = .bpa$al_dabhead;
;  500          1046  2
;  501          1047  2  DO
;  502          1048  3      BEGIN
;  503          1049  3
;  504          1050  3      IF .dab2 [dab$b_length] EQLU length [0]
;  505          1051  3      THEN
;  506          1052  4          BEGIN
;  507          1053  4          sts = CH$COMPARE (length [0], addr [0], length [0],
;  508          1054  4              dab2 [dab$a_name]);
;  509          1055  4
;  510          1056  4          IF .sts EQL 0
;  511          1057  4          THEN
;  512          1058  5              BEGIN
;  513          1059  5              answer [0] = .dab2;              ! M 319
;  514          1060  5              EXITLOOP;
;  515          1061  4              END;
;  516          1062  4
;  517          1063  3          END;
;  518          1064  3
;  519          1065  3      next = .dab2 [dab$l_next];
;  520          1066  3      dab2 = .next;
;  521          1067  3      END
;  522          1068  2  UNTIL .next EQLU bpa$al_dabhead;
;  523          1069  2
;  524          1070  1  END;                                         ! End of bpa$find_dab


                                                    .EXTRN  SYS$SETAST, SYS$DCLEXH
```

```
                                 03FC 00000 BPA$FIND_DAB:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9             ; 0962
                 59 00000000G  00  9E 00002        MOVAB   SYS$SETAST, R9                           ;
                 58 00000000'  EF  9E 00009        MOVAB   BPA$AL_DABHEAD, R8                       ;
                 57                D4 00010        CLRL    NEXT                                     ; 1008
                           0C  3C  D4 00012        CLRL    @ANSWER                                  ; 1009
                 59        FC  A8  E8 00015        BLBS    QUEUE_INITTED, 4$                        ; 1014
                               7E  D4 00019        CLRL    -(SP)                                    ; 1022
                 69            01  FB 0001B        CALLS   #1, SYS$SETAST                           ;
                 53            50  D0 0001E        MOVL    R0, AST_STATUS                           ;
                 2C        FC  A8  E8 00021        BLBS    QUEUE_INITTED, 1$                        ; 1024
                 50            68  9E 00025        MOVAB   BPA$AL_DABHEAD, R0                       ; 1027
           04    A8            50  D0 00028        MOVL    R0, BPA$AL_DABHEAD+4                     ;
                 68            50  D0 0002C        MOVL    R0, BPA$AL_DABHEAD                       ;
           F0    A8      0000V CF  9E 0002F        MOVAB   EXIT_HANDLER, EXIT_BLOCK+4               ; 1028
           F4    A8            01  D0 00035        MOVL    #1, EXIT_BLOCK+8                         ; 1029
           F8    A8        E8  A8  9E 00039        MOVAB   EXIT_REASON, EXIT_BLOCK+12               ; 1030
                           EC  A8  9F 0003E        PUSHAB  EXIT_BLOCK                               ; 1031
    00000000G    00            01  FB 00041        CALLS   #1, SYS$DCLEXH                           ;
                 52            50  D0 00048        MOVL    R0, DCLEXH_STATUS                        ;
           FC    A8            01  D0 0004B        MOVL    #1, QUEUE_INITTED                        ; 1032
                               03  11 0004F        BRB     2$                                       ; 1024
                 52            01  D0 00051 1$:    MOVL    #1, DCLEXH_STATUS                        ; 1035
                 09            53  D1 00054 2$:    CMPL    AST_STATUS, #9                           ; 1037
                               05  12 00057        BNEQ    3$                                       ;
                               01  DD 00059        PUSHL   #1                                       ;
                 69            01  FB 0005B        CALLS   #1, SYS$SETAST                           ;
                 11            52  E8 0005E 3$:    BLBS    DCLEXH_STATUS, 4$                        ; 1039
                               52  DD 00061        PUSHL   DCLEXH_STATUS                            ;
                               7E  D4 00063        CLRL    -(SP)                                    ;
                   001A8090    8F  DD 00065        PUSHL   #1736848                                 ;
    00000000G    00            03  FB 0006B        CALLS   #3, LIB$SIGNAL                           ;
                 50            68  9E 00072 4$:    MOVAB   BPA$AL_DABHEAD, R0                       ; 1043
                 50            68  D1 00075        CMPL    BPA$AL_DABHEAD, R0                       ;
                               33  13 00078        BEQL    8$                                       ;
                 54            68  D0 0007A        MOVL    BPA$AL_DABHEAD, DAB2                     ; 1045
04   AC     0A   A4   08       00  ED 0007D 5$:    CMPZV   #0, #8, 10(DAB2), LENGTH                 ; 1050
                               19  12 00084        BNEQ    7$                                       ;
                 55            01  D0 00086        MOVL    #1, R5                                   ; 1054
            0B   A4   08   BC  04  AC  29 00089    CMPC3   LENGTH, @ADDR, 11(DAB2)                  ;
                               03  1A 00090        BGTRU   6$                                       ;
                 55            01  D9 00092        SBWC    #1, R5                                   ;
                 56            55  D0 00095 6$:    MOVL    R5, STS                                  ;
                               05  12 00098        BNEQ    7$                                       ; 1056
           0C    BC            54  D0 0009A        MOVL    DAB2, @ANSWER                            ; 1059
                                   04 0009E        RET                                              ; 1058
                 57            64  D0 0009F 7$:    MOVL    (DAB2), NEXT                             ; 1065
                 54            57  D0 000A2        MOVL    NEXT, DAB2                               ; 1066
                 50            68  9E 000A5        MOVAB   BPA$AL_DABHEAD, R0                       ; 1068
                 50            57  D1 000A8        CMPL    NEXT, R0                                 ;
                               D0  12 000AB        BNEQ    5$                                       ;
                                   04 000AD 8$:    RET                                              ; 1070

; Routine Size:  174 bytes,    Routine Base:  _BPA$CODE + 0206
```

```
;  526          1071  1 ROUTINE bpa$make_dab (length, addr) =
;  527          1072  1
;  528          1073  1 !++
;  529          1074  1 ! FUNCTIONAL DESCRIPTION:
;  530          1075  1 !
;  531          1076  1 !       Routine inserts a dab into the dab list.
;  532          1077  1 !
;  533          1078  1 ! FORMAL PARAMETERS:
;  534          1079  1 !
;  535          1080  1 !       length = length of device name string
;  536          1081  1 !       addr   = address of string
;  537          1082  1 !
;  538          1083  1 ! IMPLICIT INPUTS:
;  539          1084  1 !
;  540          1085  1 !       NONE
;  541          1086  1 !
;  542          1087  1 ! IMPLICIT OUTPUTS:
;  543          1088  1 !
;  544          1089  1 !       NONE
;  545          1090  1 !
;  546          1091  1 ! ROUTINE VALUE:
;  547          1092  1 !
;  548          1093  1 !       Always returns TRUE.
;  549          1094  1 !
;  550          1095  1 ! SIDE EFFECTS:
;  551          1096  1 !
;  552          1097  1 !       The required # of bytes are got from free core.
;  553          1098  1 !
;  554          1099  1 !--
;  555          1100  1
;  556          1101  2     BEGIN
;  557          1102  2
;  558          1103  2     MAP
;  559          1104  2         length : REF VECTOR [, WORD],
;  560          1105  2         addr : REF VECTOR [, LONG];
;  561          1106  2
;  562          1107  2     LOCAL                                          ! A 309
;  563          1108  2         sts,                                       ! Status returned from calls    ! A 309
;  564          1109  2         dab2 : REF $dab_def;
;  565          1110  2
;  566          1111  3     IF NOT (sts = bpa$get_block (dab$k_length_f + length [0],   ! M 309
;  567          1112  3             dab2))                                 ! A 309
;  568          1113  2     THEN                                           ! A 309
;  569          1114  2         RETURN SIGNAL (badfuo, 0, .sts);           ! A 309
;  570          1115  2
;  571          1116  2     CH$MOVE (length [0], addr [0], dab2 [dab$a_name]);
;  572          1117  2     dab2 [dab$b_length] = length [0];
;  573          1118  2     dab2 [dab$w_mode] = 0;
;  574          1119  2     INSQUE (.dab2, bpa$al_dabhead);
;  575          1120  2     RETURN 1;
;  576          1121  1     END;                                           ! End of bpa$make_dab


                                                007C 00000 BPA$MAKE_DAB:
```

```
                                                            .WORD   Save R2,R3,R4,R5,R6                 ; 1071
                      5E          04  C2 00002              SUBL2   #4, SP                              ;
                                  5E  DD 00005              PUSHL   SP                                  ; 1111
         7E       04  AC          0B  C1 00007              ADDL3   #11, LENGTH, -(SP)                  ;
           00000000G  00          02  FB 0000C              CALLS   #2, BPA$GET_BLOCK                   ;
                      12          50  E8 00013              BLBS    STS, 1$                             ;
                                  50  DD 00016              PUSHL   STS                                 ; 1114
                                  7E  D4 00018              CLRL    -(SP)                               ;
                        001A8090  8F  DD 0001A              PUSHL   #1736848                            ;
           00000000G  00          03  FB 00020              CALLS   #3, LIB$SIGNAL                      ;
                                      04 00027              RET                                         ;
                      56          6E  D0 00028 1$:          MOVL    DAB2, R6                            ; 1116
    0B   A6       08  BC      04  AC  28 0002B              MOVC3   LENGTH, @ADDR, 11(R6)               ;
                  0A  A6      04  AC  90 00032              MOVB    LENGTH, 10(R6)                      ; 1117
                              08  A6  B4 00037              CLRW    8(R6)                               ; 1118
           00000000'  EF          66  0E 0003A              INSQUE  (R6), BPA$AL_DABHEAD                ; 1119
                      50          01  D0 00041              MOVL    #1, R0                              ; 1120
                                      04 00044              RET                                         ; 1121

; Routine Size:  69 bytes,    Routine Base:  _BPA$CODE + 02B4
```

```
;   578          1122  1 ROUTINE bpa$release_dab (dab_addr) =
;   579          1123  1
;   580          1124  1 !++
;   581          1125  1 ! FUNCTIONAL DESCRIPTION:
;   582          1126  1 !
;   583          1127  1 !       Routine removes the specified dab entry from the list
;   584          1128  1 !       and deallocates the device described by the entry.
;   585          1129  1 !
;   586          1130  1 ! FORMAL PARAMETERS:
;   587          1131  1 !
;   588          1132  1 !       dab_addr = address of dab entry in list
;   589          1133  1 !
;   590          1134  1 ! IMPLICIT INPUTS:
;   591          1135  1 !
;   592          1136  1 !       NONE
;   593          1137  1 !
;   594          1138  1 ! IMPLICIT OUTPUTS:
;   595          1139  1 !
;   596          1140  1 !       NONE
;   597          1141  1 !
;   598          1142  1 ! ROUTINE VALUE:
;   599          1143  1 !
;   600          1144  1 !       Always returns TRUE.
;   601          1145  1 !
;   602          1146  1 ! SIDE EFFECTS:
;   603          1147  1 !
;   604          1148  1 !       The unlinked entry is returned to free core.
;   605          1149  1 !
;   606          1150  1 !--
;   607          1151  1
;   608          1152  2     BEGIN
;   609          1153  2
;   610          1154  2     MAP
;   611          1155  2         dab_addr : REF VECTOR [, LONG];
;   612          1156  2
;   613          1157  2     LOCAL
;   614          1158  2         sts,                                ! Status returned from calls    ! A 309
;   615          1159  2         dab : REF $dab_def,
;   616          1160  2         addr,
;   617          1161  2         descrip : BLOCK [8, BYTE];
;   618          1162  2
;   619          1163  2     dab = dab_addr [0];
;   620          1164  2     descrip [dsc$w_length] = .dab [dab$b_length];
;   621          1165  2     descrip [dsc$b_dtype] = dsc$k_dtype_t;
;   622          1166  2     descrip [dsc$b_class] = dsc$k_class_s;
;   623          1167  2     descrip [dsc$a_pointer] = dab [dab$a_name];
;   624          1168  2     $dalloc (devnam = descrip);
;   625          1169  2     REMQUE (dab_addr [0], addr);
;   626          1170  2
;   627          1171  3     IF NOT (sts = bpa$free_block (dab_addr [0],          ! M 309
;   628          1172  3             dab$k_length_f + .dab [dab$b_length]))       ! A 309
;   629          1173  2     THEN                                                ! A 309
;   630          1174  2         RETURN SIGNAL (badfuo, 0, .sts);                ! A 309
;   631          1175  2
;   632          1176  2     RETURN 1;
;   633          1177  1     END;                                                ! End of bpa$release_dab
```

```
                                                      .EXTRN  SYS$DALLOC

                                  0004 00000 BPA$RELEASE_DAB:
                                                      .WORD   Save R2                                 ; 1122
                 5E          08   C2 00002            SUBL2   #8, SP                                  ;
                 52     04   AC   D0 00005            MOVL    DAB_ADDR, DAB                           ; 1163
                 6E     0A   A2   9B 00009            MOVZBW  10(DAB), DESCRIP                        ; 1164
          02     AE   010E   8F   B0 0000D            MOVW    #270, DESCRIP+2                         ; 1165
          04     AE     0B   A2   9E 00013            MOVAB   11(R2), DESCRIP+4                       ; 1167
                             7E   D4 00018            CLRL    -(SP)                                   ; 1168
                        04   AE   9F 0001A            PUSHAB  DESCRIP                                 ;
   00000000G     00          02   FB 0001D            CALLS   #2, SYS$DALLOC                          ;
                 50     04   BC   0F 00024            REMQUE  @DAB_ADDR, ADDR                         ; 1169
                 7E     0A   A2   9A 00028            MOVZBL  10(DAB), -(SP)                          ; 1172
                 6E          0B   C0 0002C            ADDL2   #11, (SP)                               ;
                        04   AC   DD 0002F            PUSHL   DAB_ADDR                                ; 1171
   00000000G     00          02   FB 00032            CALLS   #2, BPA$FREE_BLOCK                      ;
                 12          50   E8 00039            BLBS    STS, 1$                                 ;
                             50   DD 0003C            PUSHL   STS                                     ; 1174
                             7E   D4 0003E            CLRL    -(SP)                                   ;
                  001A8090   8F   DD 00040            PUSHL   #1736848                                ;
   00000000G     00          03   FB 00046            CALLS   #3, LIB$SIGNAL                          ;
                                  04 0004D            RET                                             ;
                 50          01   D0 0004E 1$:        MOVL    #1, R0                                  ; 1176
                                  04 00051            RET                                             ; 1177

; Routine Size:  82 bytes,    Routine Base:  _BPA$CODE + 02F9


;  634          1178  1 !
;  635          1179  1
```

```
;  637          1180  1 GLOBAL ROUTINE bpa$ascii (addr, length, asc) =
;  638          1181  1
;  639          1182  1 !++
;  640          1183  1 ! FUNCTIONAL DESCRIPTION:
;  641          1184  1 !
;  642          1185  1 !       Routine converts a word containing a rad-50 representation
;  643          1186  1 !       of upto 3 characters into an ascii string.
;  644          1187  1 !
;  645          1188  1 ! NOTE:- Imbedded spaces will be ignored !
;  646          1189  1 !       i.e. A<space>B or <space>AB will become AB<null>
;  647          1190  1 !       (These are illegal in filenames anyway!)
;  648          1191  1 !
;  649          1192  1 !
;  650          1193  1 ! FORMAL PARAMETERS:
;  651          1194  1 !
;  652          1195  1 !       On input : addr = word containing rad-50
;  653          1196  1 !
;  654          1197  1 !       On output : asc is 3 byte vector containing upto 3 ascii chars
;  655          1198  1 !                   length contains # of chars in string
;  656          1199  1 !
;  657          1200  1 ! IMPLICIT INPUTS:
;  658          1201  1 !
;  659          1202  1 !       NONE
;  660          1203  1 !
;  661          1204  1 ! IMPLICIT OUTPUTS:
;  662          1205  1 !
;  663          1206  1 !       NONE
;  664          1207  1 !
;  665          1208  1 ! ROUTINE VALUE:
;  666          1209  1 !
;  667          1210  1 !       Always returns TRUE.
;  668          1211  1 !
;  669          1212  1 ! SIDE EFFECTS:
;  670          1213  1 !
;  671          1214  1 !       NONE
;  672          1215  1 !
;  673          1216  1 !--
;  674          1217  1
;  675          1218  2     BEGIN
;  676          1219  2
;  677          1220  2     MAP
;  678          1221  2         length : REF VECTOR [, WORD],
;  679          1222  2         asc : REF VECTOR [, BYTE];
;  680          1223  2
;  681          1224  2     LOCAL
;  682          1225  2         rad,
;  683          1226  2         rcode;
;  684          1227  2
;  685          1228  2     length [0] = 0;
;  686          1229  2     rad = .addr;
;  687          1230  2
;  688          1231  2     DECR x FROM 2 TO 0 DO
;  689          1232  3         BEGIN
;  690          1233  3         rcode = .rad MOD %O'50';
;  691          1234  3         rad = .rad - .rcode;
;  692          1235  3         rad = .rad/%O'50';
;  693          1236  3
```

```
;  694        1237  3           CASE .rcode FROM %O'0' TO %O'47' OF
;  695        1238  3               SET
;  696        1239  3
;  697        1240  3               [%O'1' TO %O'32'] :
;  698        1241  3                   rcode = .rcode + %O'100';
;  699        1242  3
;  700        1243  3               [%O'36' TO %O'47'] :
;  701        1244  3                   rcode = .rcode + %O'22';
;  702        1245  3
;  703        1246  3               [%O'33'] :
;  704        1247  3                   rcode = %C'$';
;  705        1248  3
;  706        1249  3               [%O'34'] :
;  707        1250  3                   rcode = %C'.';
;  708        1251  3
;  709        1252  3               [INRANGE] :
;  710        1253  3                   rcode = %O'0';                  ! Ignore 0 & 35                  ! M325
;  711        1254  3
;  712        1255  3               [OUTRANGE] :
;  713        1256  3                   RETURN SIGNAL (OTS$_FATINTERR);
;  714        1257  3               TES;
;  715        1258  3
;  716        1259  3           asc [.x] = .rcode;
;  717        1260  3
;  718        1261  3           IF (.rcode NEQU 0) THEN length [0] = .length [0] + 1;
;  719        1262  3
;  720        1263  2           END;
;  721        1264  2
;  722        1265  2       RETURN 1;
;  723        1266  1       END;                                !End of bpa$ascii


                                       001C 00000          .ENTRY  BPA$ASCII, Save R2,R3,R4            ; 1180
                               08   BC   B4 00002          CLRW    @LENGTH                             ; 1228
                     54        04   AC   D0 00005          MOVL    ADDR, RAD                           ; 1229
                     53             02   D0 00009          MOVL    #2, X                               ; 1259
      7E        00   54             01   7A 0000C 1$:      EMUL    #1, RAD, #0, -(SP)                  ; 1233
      52        52   8E             28   7B 00011          EDIV    #40, (SP)+, RCODE, RCODE            ;
                     54             52   C2 00016          SUBL2   RCODE, RAD                          ; 1234
                     54             28   C6 00019          DIVL2   #40, RAD                            ; 1235
                27   00             52   CF 0001C          CASEL   RCODE, #0, #39                      ; 1237
    005E      005E   005E         0073      00020 2$:      .WORD   7$-2$,-                             ;
    005E      005E   005E         005E      00028                  3$-2$,-                             ;
    005E      005E   005E         005E      00030                  3$-2$,-                             ;
    005E      005E   005E         005E      00038                  3$-2$,-                             ;
    005E      005E   005E         005E      00040                  3$-2$,-                             ;
    005E      005E   005E         005E      00048                  3$-2$,-                             ;
    0069      005E   005E         005E      00050                  3$-2$,-                             ;
    0064      0064   0073         006E      00058                  3$-2$,-                             ;
    0064      0064   0064         0064      00060                  3$-2$,-                             ;
    0064      0064   0064         0064      00068                  3$-2$,-                             ;
                                                                   3$-2$,-                             ;
                                                                   3$-2$,-                             ;
                                                                   3$-2$,-                             ;
```

```
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          3$-2$,-                          ;
                                                          5$-2$,-                          ;
                                                          6$-2$,-                          ;
                                                          7$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$,-                          ;
                                                          4$-2$                            ;
                 00000000G   8F  DD 00070       PUSHL     #OTS$_FATINTERR                  ; 1256
     00000000G  00           01  FB 00076       CALLS     #1, LIB$SIGNAL                   ;
                                 04 0007D       RET                                        ;
            52        40     A2  9E 0007E 3$:   MOVAB     64(R2), RCODE                    ; 1241
                             11  11 00082       BRB       8$                               ;
            52               12  C0 00084 4$:   ADDL2     #18, RCODE                       ; 1244
                             0C  11 00087       BRB       8$                               ;
            52               24  D0 00089 5$:   MOVL      #36, RCODE                       ; 1247
                             07  11 0008C       BRB       8$                               ;
            52               2E  D0 0008E 6$:   MOVL      #46, RCODE                       ; 1250
                             02  11 00091       BRB       8$                               ;
                             52  D4 00093 7$:   CLRL      RCODE                            ; 1253
       0C BC43               52  90 00095 8$:   MOVB      RCODE, @ASC[X]                   ; 1259
                             52  D5 0009A       TSTL      RCODE                            ; 1261
                             03  13 0009C       BEQL      9$                               ;
                      08     BC  B6 0009E       INCW      @LENGTH                          ;
            02               53  F4 000A1 9$:   SOBGEQ    X, 10$                           ; 1231
                             03  11 000A4       BRB       11$                              ;
                           FF63  31 000A6 10$:  BRW       1$                               ;
            50               01  D0 000A9 11$:  MOVL      #1, R0                           ; 1265
                                 04 000AC       RET                                        ; 1266

; Routine Size:  173 bytes,    Routine Base:  _BPA$CODE + 034B

;  724          1267  1
```

```
;  726          1268  1 ROUTINE cnv_ppn (addr, length, ppn) =
;  727          1269  1
;  728          1270  1 !++
;  729          1271  1 ! FUNCTIONAL DESCRIPTION:
;  730          1272  1 !
;  731          1273  1 !       Routine converts a decimal ppn # (max 377) to an
;  732          1274  1 !       ASCII string.
;  733          1275  1 !
;  734          1276  1 ! FORMAL PARAMETERS:
;  735          1277  1 !
;  736          1278  1 !       On input : addr = ppn #
;  737          1279  1 !
;  738          1280  1 !       On output : ppn is a 3 byte vector containing the ascii string
;  739          1281  1 !                   length contains the # of chars.
;  740          1282  1 !
;  741          1283  1 ! IMPLICIT INPUTS:
;  742          1284  1 !
;  743          1285  1 !       NONE
;  744          1286  1 !
;  745          1287  1 ! IMPLICIT OUTPUTS:
;  746          1288  1 !
;  747          1289  1 !       NONE
;  748          1290  1 !
;  749          1291  1 ! ROUTINE VALUE:
;  750          1292  1 !
;  751          1293  1 !       Always returns TRUE.
;  752          1294  1 !
;  753          1295  1 ! SIDE EFFECTS:
;  754          1296  1 !
;  755          1297  1 !       NONE
;  756          1298  1 !
;  757          1299  1 !--
;  758          1300  1
;  759          1301  2     BEGIN
;  760          1302  2
;  761          1303  2     MAP
;  762          1304  2         length : REF VECTOR [, WORD],
;  763          1305  2         ppn : REF VECTOR [, BYTE];
;  764          1306  2
;  765          1307  2     LOCAL
;  766          1308  2         c : VECTOR [3, BYTE],
;  767          1309  2         val;
;  768          1310  2
;  769          1311  2     length [0] = 0;
;  770          1312  2     val = .addr;
;  771          1313  2     c [0] = .val/100;
;  772          1314  2     c [1] = (.val - .c [0]*100)/10;
;  773          1315  2     c [2] = .val - (.c [1]*10 + .c [0]*100);
;  774          1316  2
;  775          1317  2     INCR x FROM 0 TO 2 DO
;  776          1318  3         BEGIN
;  777          1319  3         ppn [.x] = .c [.x] + %C'0';
;  778          1320  3         length [0] = .length [0] + 1;
;  779          1321  2         END;
;  780          1322  2
;  781          1323  2     RETURN 1;
;  782          1324  1     END;                                    !End of cnv_ppn
```

```
                                 0004 00000 CNV_PPN:.WORD   Save R2                                  ; 1268
                          5E           04   C2 00002        SUBL2   #4, SP                         ;
                                08     BC   B4 00005        CLRW    @LENGTH                        ; 1311
                          50    04     AC   D0 00008        MOVL    ADDR, VAL                      ; 1312
                    51    50 00000064  8F   C7 0000C        DIVL3   #100, VAL, R1                  ; 1313
                          6E           51   90 00014        MOVB    R1, C                          ;
                          51           6E   9A 00017        MOVZBL  C, R1                          ; 1314
                          51 00000064  8F   C4 0001A        MULL2   #100, R1                       ;
                    51    50           51   C3 00021        SUBL3   R1, VAL, R1                    ;
                    52    51           0A   C7 00025        DIVL3   #10, R1, R2                    ;
                       01 AE           52   90 00029        MOVB    R2, C+1                        ;
                          51     01 AE  9A   0002D        MOVZBL  C+1, R1                        ; 1315
                          51           0A   C4 00031        MULL2   #10, R1                        ;
                          52           6E   9A 00034        MOVZBL  C, R2                          ;
                          52 00000064  8F   C4 00037        MULL2   #100, R2                       ;
                          51           52   C0 0003E        ADDL2   R2, R1                         ;
                02 AE     50           51   83 00041        SUBB3   R1, VAL, C+2                   ;
                                       50   D4 00046        CLRL    X                              ; 1319
                0C BC40        6E40    30   81 00048 1$:     ADDB3   #48, C[X], @PPN[X]             ;
                                08     BC   B6 0004F        INCW    @LENGTH                        ; 1320
                    F2         50     02   F3 00052        AOBLEQ  #2, X, 1$                      ; 1317
                               50     01   D0 00056        MOVL    #1, R0                         ; 1323
                                       04   00059        RET                                    ; 1324

; Routine Size:  90 bytes,    Routine Base:  _BPA$CODE + 03F8
```

```
;  784          1325  1 ROUTINE cnv_devnam (asc, dot, firqb) : NOVALUE =        ! M 319
;  785          1326  1
;  786          1327  1 !++
;  787          1328  1 ! FUNCTIONAL DESCRIPTION:
;  788          1329  1 !
;  789          1330  1 !       Routine converts the RSTS-type device name described
;  790          1331  1 !       in the FIRQB into a VAX-type ascii string.
;  791          1332  1 !       e.g. KB64 -> TTE0:
;  792          1333  1 !
;  793          1334  1 ! FORMAL PARAMETERS:
;  794          1335  1 !
;  795          1336  1 !       asc = address of 6-byte vector to store string.
;  796          1337  1 !       dot = Pointer to a longword to receive the length of the string ! A 319
;  797          1338  1 !       firqb = Pointer to firqb                                       ! A 319
;  798          1339  1 !
;  799          1340  1 ! IMPLICIT INPUTS:
;  800          1341  1 !
;  801          1342  1 !       firqb [fqb$w_devnam]  contains the device name as 2 ascii characters,
;  802          1343  1 !       firqb [fqb$b_devunit]  contains the device decimal unit #.
;  803          1344  1 !
;  804          1345  1 ! IMPLICIT OUTPUTS:
;  805          1346  1 !
;  806          1347  1 !       The asc vector contains the ascii string.
;  807          1348  1 !
;  808          1349  1 ! ROUTINE VALUE:
;  809          1350  1 !
;  810          1351  1 !       NONE                                                   ! M 319
;  811          1352  1 !
;  812          1353  1 ! SIDE EFFECTS:
;  813          1354  1 !
;  814          1355  1 !       NONE
;  815          1356  1 !
;  816          1357  1 !--
;  817          1358  1
;  818          1359  2     BEGIN
;  819          1360  2
;  820          1361  2     LOCAL
;  821          1362  2         num,
;  822          1363  2         un;
;  823          1364  2
;  824          1365  2     MAP
;  825          1366  2         firqb : REF $fqb_def,                  ! Defines firqb          ! A 319
;  826          1367  2         asc : REF VECTOR [, BYTE],
;  827          1368  2         dot : REF VECTOR [1, WORD];            ! A 319
;  828          1369  2
;  829          1370  2     dot [0] = 0;                               ! M 319
;  830          1371  2
;  831          1372  2     IF .firqb [fqb$w_devnam] EQLU %ASCII'KB'
;  832          1373  2     THEN
;  833          1374  2         CH$MOVE (2,
;  834          1375  2             UPLIT BYTE('TT'), asc [.dot [0]])  ! M 319
;  835          1376  2     ELSE
;  836          1377  2         CH$MOVE (2, firqb [fqb$w_devnam], asc [.dot [0]]);     ! M 319
;  837          1378  2
;  838          1379  2     dot [0] = .dot [0] + 2;                    ! M 319
;  839          1380  2     num = .firqb [fqb$b_devunit];
;  840          1381  2     un = (.num^(-4)) + %A';
```

```
;  841          1382  2     num = .num AND %O'17';
;  842          1383  2     asc [.dot [0]] = .un;                    ! M 319
;  843          1384  2     dot [0] = .dot [0] + 1;                  ! M 319
;  844          1385  2
;  845          1386  2     IF .num GTRU 9
;  846          1387  2     THEN
;  847          1388  3         BEGIN
;  848          1389  3         asc [.dot [0]] = '1';                ! M 319
;  849          1390  3         dot [0] = .dot [0] + 1;              ! M 319
;  850          1391  3         num = .num - 10;
;  851          1392  3         END;
;  852          1393  2
;  853          1394  2     asc [.dot [0]] = .num + '0';             ! M 319
;  854          1395  2     dot [0] = .dot [0] + 1;                  ! M 319
;  855          1396  2     asc [.dot [0]] = ':';                    ! M 319
;  856          1397  2     dot [0] = .dot [0] + 1;                  ! M 319
;  857          1398  1     END;                                     !End of cnv_devnam


                                    54  54  00452 P.AAA:  .ASCII  \TT\                                    ;


                                            001C 00000 CNV_DEVNAM:
                                                       .WORD   Save R2,R3,R4                                ; 1325
                                52          04   AC  7D 00002          MOVQ    ASC, R2                      ;  1375
                                            63   B4 00006          CLRW    (R3)                         ;  1370
                                50          0C   AC  D0 00008          MOVL    FIRQB, R0                    ;  1372
                           424B 8F          18   A0  B1 0000C          CMPW    24(R0), #16971               ;
                                                0C  12 00012          BNEQ    1$                           ;
                                51              63  3C 00014          MOVZWL  (R3), R1                     ;  1375
                                              6142  9F 00017          PUSHAB  (R1)[R2]                     ;
                                9E            E1    AF  B0 0001A          MOVW    P.AAA, @(SP)+                ;
                                                0A  11 0001E          BRB     2$                           ;
                                51              63  3C 00020 1$:      MOVZWL  (R3), R1                     ;  1377
                                              6142  9F 00023          PUSHAB  (R1)[R2]                     ;
                                9E            18    A0  B0 00026          MOVW    24(R0), @(SP)+               ;
                                63              02  A0 0002A 2$:      ADDW2   #2, (R3)                     ;  1379
                                50          1A    A0  9A 0002D          MOVZBL  26(R0), NUM                  ;  1380
                      51        50          FC    8F  78 00031          ASHL    #-4, NUM, R1                 ;  1381
                                51          41    A1  9E 00036          MOVAB   65(R1), UN                   ;
                50      50      04          00    EF  0003A          EXTZV   #0, #4, NUM, NUM             ;  1382
                                54              63  3C 0003F          MOVZWL  (R3), R4                     ;  1383
                              6442              51  90 00042          MOVB    UN, (R4)[R2]                 ;
                                              63  B6 00046          INCW    (R3)                         ;  1384
                                09              50  D1 00048          CMPL    NUM, #9                      ;  1386
                                                0C  1B 0004B          BLEQU   3$                           ;
                                51              63  3C 0004D          MOVZWL  (R3), R1                     ;  1389
                              6142              31  90 00050          MOVB    #49, (R1)[R2]                ;
                                              63  B6 00054          INCW    (R3)                         ;  1390
                                50              0A  C2 00056          SUBL2   #10, NUM                     ;  1391
                                51              63  3C 00059 3$:      MOVZWL  (R3), R1                     ;  1394
                      6142      50              30  81 0005C          ADDB3   #48, NUM, (R1)[R2]           ;
                                              63  B6 00061          INCW    (R3)                         ;  1395
                                50              63  3C 00063          MOVZWL  (R3), R0                     ;  1396
                              6042              3A  90 00066          MOVB    #58, (R0)[R2]                ;
```

```
                63  B6 0006A           INCW    (R3)                                    ; 1397
                    04 0006C           RET                                             ; 1398

; Routine Size:  109 bytes,    Routine Base:  _BPA$CODE + 0454
```

```
;  859          1399  1 ROUTINE exit_handler (                          ! Exit handler
;  860          1400  1         exit_reason                             ! reason
;  861          1401  1     ) : NOVALUE =
;  862          1402  1
;  863          1403  1 !++
;  864          1404  1 ! FUNCTIONAL DESCRIPTION:
;  865          1405  1 !
;  866          1406  1 !       This is the exit handler for assign/deassign.
;  867          1407  1 !       It deallocates all devices, in case any have not been
;  868          1408  1 !       deallocated already.
;  869          1409  1 !
;  870          1410  1 ! FORMAL PARAMETERS:
;  871          1411  1 !
;  872          1412  1 !       EXIT_REASON.rl.r        Not used
;  873          1413  1 !
;  874          1414  1 ! IMPLICIT INPUTS:
;  875          1415  1 !
;  876          1416  1 !       NONE
;  877          1417  1 !
;  878          1418  1 ! IMPLICIT OUTPUTS:
;  879          1419  1 !
;  880          1420  1 !       NONE
;  881          1421  1 !
;  882          1422  1 ! ROUTINE VALUE:
;  883          1423  1 !
;  884          1424  1 !       NONE
;  885          1425  1 !
;  886          1426  1 ! SIDE EFFECTS:
;  887          1427  1 !
;  888          1428  1 !       Deassigns the devices, if there are any.
;  889          1429  1 !
;  890          1430  1 !--
;  891          1431  1
;  892          1432  2     BEGIN
;  893          1433  2     bpa$deass_all ();
;  894          1434  2     RETURN;
;  895          1435  1     END;                                          !End of exit_handler


                                        0000 00000 EXIT_HANDLER:
                                                   .WORD   Save nothing                          ; 1399
                            FD20   CF   00  FB 00002       CALLS   #0, BPA$DEASS_ALL             ; 1433
                                        04 00007       RET                                       ; 1435

; Routine Size:  8 bytes,    Routine Base:  _BPA$CODE + 04C1

;  896          1436  1 END                                             !End of module bpa$assdeas
;  897          1437  1
;  898          1438  0 ELUDOM
```

```
                                .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;  _BPA$DATA                      32  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _BPA$CODE                    1225  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                           Library Statistics
;
;                                       -------- Symbols --------       Pages       Processing
;       File                            Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1     9776        15         0         581        00:01.1


;                                   COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BPASSDEAS/OBJ=OBJ$:BPASSDEAS MSRC$:BPASSDEAS/UPDATE=(ENH$:BPASSDEAS
;       )

; Size:         1223 code + 34 data bytes
; Run Time:       00:27.3
; Elapsed Time:   00:59.7
; Lines/CPU Min:   3163
; Lexemes/CPU-Min: 27196
; Memory Used:  157 pages
; Compilation Complete
```

BOOTBLOCK
MAP

STACONFIG
MAP

SYSBOOT
MAP

BPAMOVTUC
LIS

BOOTS

BPASSDEAS
LIS

BOOT58
MAP

STANDCONF
MAP

CONFIGURE
MAP

BPAWAKEUP
LIS

STASYSGEN
MAP

BPASETPRI
LIS