```
RRRRRRRRRRR      UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN         NNN     000000000     FFFFFFFFFFFFFFF   FFFFFFFFFFFFFFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNN         NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRR          RRR UUU          UUU  NNNNNN      NNN  000          000 FFF               FFF
RRRRRRRRRRR      UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRRRRRRRRRRR     UUU          UUU  NNN   NNN   NNN  000          000 FFFFFFFFFFF       FFFFFFFFFFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR   RRR        UUU          UUU  NNN       NNNNNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR      RRR     UUU          UUU  NNN          NNN 000          000 FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
RRR         RRR  UUUUUUUUUUUUUUU  NNN          NNN    000000000     FFF               FFF
```

```
_$2

Sym
---
NDX
NDX
NUM
NUM
OPE
OUT
PAC
PAC
PAC
PAC
PAC
PAC
PAC
PAD
PAG
PAG
PAG
PAG
PAG
PAG
PAG
PER
PUT
RCO
RIN

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR

STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

FILEID**PUTNDX

```
PPPPPPPP    UU      UU  TTTTTTTTTT  NN      NN  DDDDDDDD    XX      XX
PPPPPPPP    UU      UU  TTTTTTTTTT  NN      NN  DDDDDDDD    XX      XX
PP      PP  UU      UU      TT      NN      NN  DD      DD  XX      XX
PP      PP  UU      UU      TT      NN      NN  DD      DD  XX      XX
PP      PP  UU      UU      TT      NNNN    NN  DD      DD    XX  XX
PP      PP  UU      UU      TT      NNNN    NN  DD      DD    XX  XX
PPPPPPPP    UU      UU      TT      NN  NN  NN  DD      DD      XX
PPPPPPPP    UU      UU      TT      NN  NN  NN  DD      DD      XX
PP          UU      UU      TT      NN    NNNN  DD      DD    XX  XX
PP          UU      UU      TT      NN    NNNN  DD      DD    XX  XX
PP          UU      UU      TT      NN      NN  DD      DD  XX      XX      ....
PP          UU      UU      TT      NN      NN  DD      DD  XX      XX      ....
PP          UUUUUUUUUU      TT      NN      NN  DDDDDDDD    XX      XX      ....
PP          UUUUUUUUUU      TT      NN      NN  DDDDDDDD    XX      XX      ....

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 MODULE PUTNDX ( IDENT = 'V04-000'
;     2        P 0002  0                 %BLISS32 [ , ADDRESSING_MODE ( EXTERNAL =     LONG_RELATIVE,
;     3          0003  0                                                NONEXTERNAL =  LONG_RELATIVE)]
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    18          0018  1 !*  TRANSFERRED.                                                             *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    22          0022  1 !*  CORPORATION.                                                             *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT: Routines to save index information in a file.
;    34          0034  1 !
;    35          0035  1 !
;    36          0036  1 ! ENVIRONMENT: Transportable
;    37          0037  1 !
;    38          0038  1 ! AUTHOR: R.W.Friday     CREATION DATE: February, 1979
;    39          0039  1 !
```

```
;    41          0040  1 %SBTTL 'Revision History'
;    42          0041  1 !
;    43          0042  1 ! MODIFIED BY:
;    44          0043  1 !
;    45          0044  1 !       015     KFA00015        Ken Alden       19-Jul-1983
;    46          0045  1 !               Conditionalized the internal logic errors to check for
;    47          0046  1 !                 /AUTO conditions.
;    48          0047  1 !
;    49          0048  1 !       014     KAD00014        Keith Dawson    18-May-1983
;    50          0049  1 !               Restored lost information written to the .BFL file for FLIP
;    51          0050  1 !                 output.
;    52          0051  1 !
;    53          0052  1 !       013     KAD00013        Keith Dawson    11-April-1983
;    54          0053  1 !               Added support for new termination error messages for
;    55          0054  1 !                 information written to .BRN file. This involved adding
;    56          0055  1 !                 another formal to the RGH routine; this third formal is
;    57          0056  1 !                 TRUE or FALSE, as the caller determines whether or not to
;    58          0057  1 !                 increment the count of information written to the .BRN file.
;    59          0058  1 !
;    60          0059  1 !       012     KAD00012        Keith Dawson    23-March-1983
;    61          0060  1 !               Changed GCA_FLIP bit to (.gca_op_dev EQL op_dev_flip).
;    62          0061  1 !
;    63          0062  1 !       011     KAD00011        Keith Dawson    20-Mar-1983
;    64          0063  1 !               Removed LN01 conditionals and all references to .BIX
;    65          0064  1 !                 and .BTC files.
;    66          0065  1 !
;    67          0066  1 !       010     KAD00010        Keith Dawson    07-Mar-1983
;    68          0067  1 !               Global edit of all modules. Updated module names, idents,
;    69          0068  1 !               copyright dates. Changed require files to BLISS library.
;    70          0069  1 !
;    71          0070  1 !--
```

```
;    73          0071  1 %SBTTL 'Module Level Declarations'
;    74          0072  1 !
;    75          0073  1 ! TABLE OF CONTENTS:
;    76          0074  1 !
;    77          0075  1 REQUIRE 'REQ:RNODEF.REQ';       ! RUNOFF definition
;    78          0206  1
;    79          0207  1 FORWARD ROUTINE
;    80          0208  1     PUTNDY : NOVALUE,
;    81          0209  1     PUTXTN : NOVALUE,
;    82          0210  1     RGH : NOVALUE;
;    83          0211  1 !
;    84          0212  1 ! INCLUDE FILES:
;    85          0213  1 !
;    86          0214  1
;    87          0215  1 LIBRARY 'NXPORT:XPORT';                ! XPORT Library
;    88          0216  1
;    89        U 0217  1 %IF DSRPLUS %THEN
;    90        U 0218  1 LIBRARY 'REQ:DPLLIB';                 ! DSRPLUS BLISS Library
;    91          0219  1 %ELSE
;    92          0220  1 LIBRARY 'REQ:DSRLIB';                 ! DSR BLISS Library
;    93          0221  1 %FI
;    94          0222  1
;    95          0223  1 !
;    96          0224  1 ! MACROS:
;    97          0225  1 !
;    98          0226  1 MACRO
;    99        M 0227  1     check_open =
;   100        M 0228  1 %IF FLIP %THEN
;   101        M 0229  1           (IF (.gca_op_dev EQL op_dev_flip)
;   102        M 0230  1           THEN
;   103        M 0231  1               .rnoiob [IOB$V_OPEN]
;   104        M 0232  1           ELSE
;   105        M 0233  1               .brnoob [IOB$V_OPEN])
;   106        M 0234  1 %ELSE
;   107        M 0235  1           (.brnoob [IOB$V_OPEN])
;   108        M 0236  1 %FI
;   109          0237  1           %;
;   110          0238  1
;   111          0239  1 !
;   112          0240  1 ! EQUATED SYMBOLS:
;   113          0241  1 !
;   114          0242  1 !
;   115          0243  1 ! OWN STORAGE:
;   116          0244  1 !
;   117          0245  1 !
;   118          0246  1 ! EXTERNAL REFERENCES:
;   119          0247  1 !
;   120          0248  1
;   121          0249  1 EXTERNAL
;   122          0250  1     RNOIOB : REF $XPO_IOB (),
;   123          0251  1     BRNOOB : $XPO_IOB ();
;   124          0252  1
;   125          0253  1 EXTERNAL
;   126          0254  1     gca : gca_definition,
;   127          0255  1     tsf : tsf_definition;
;   128          0256  1
;   129          0257  1 EXTERNAL LITERAL                        !Error messages
```

```
;  130          0258  1      RNFILE;
;  131          0259  1
;  132          0260  1 EXTERNAL ROUTINE
;  133          0261  1      ERMS;
```

```
;  135          0262  1 GLOBAL ROUTINE PUTNDY (ENTRY_LENGTH, ENTRY_ADDRESS, XTN, BAR_FLAG) : NOVALUE =  !
;  136          0263  1
;  137          0264  1 !++
;  138          0265  1 ! FUNCTIONAL DESCRIPTION:
;  139          0266  1 !
;  140          0267  1 !       PUTNDY writes an index entry to the index file.
;  141          0268  1 !
;  142          0269  1 ! FORMAL PARAMETERS:
;  143          0270  1 !
;  144          0271  1 !       ENTRY_LENGTH is the number of bytes representing the entry.
;  145          0272  1 !       ENTRY_ADDRESS is the address of the entry.
;  146          0273  1 !       XTN is the associated transaction number.
;  147          0274  1 !       BAR_FLAG is the change-bar flag.
;  148          0275  1 !
;  149          0276  1 ! IMPLICIT INPUTS:      None
;  150          0277  1 !
;  151          0278  1 ! IMPLICIT OUTPUTS:     None
;  152          0279  1 !
;  153          0280  1 ! ROUTINE VALUE:
;  154          0281  1 ! COMPLETION CODES:     None
;  155          0282  1 !
;  156          0283  1 ! SIDE EFFECTS:         None
;  157          0284  1 !
;  158          0285  1 !--
;  159          0286  1
;  160          0287  2     BEGIN
;  161          0288  2
;  162          0289  2     LOCAL
;  163          0290  2         temp_record : VECTOR [3];
;  164        U 0291  2 %IF FLIP %THEN
;  165        U 0292  2     LOCAL
;  166        U 0293  2         flip_record : $flip_tcxtxt;
;  167          0294  2 %FI
;  168          0295  2
;  169          0296  2     !Cursory check to make sure the file is opened.
;  170          0297  3     IF NOT check_open
;  171          0298  2     THEN
;  172          0299  3         BEGIN
;  173          0300  3         IF NOT .gca_black_box
;  174          0301  3         THEN
;  175          0302  3             erms (rnfile, CH$PTR (UPLIT ('putndy')), 6);
;  176          0303  3         RETURN;
;  177          0304  2         END;
;  178          0305  2
;  179          0306  2     !If there is no index entry to write, just return.  Note that this
;  180          0307  2     !can happen if the user says, for example, ".X;", which is a valid
;  181          0308  2     !RUNOFF command but causes nothing to happen in the index.
;  182          0309  2     IF .entry_length LEQ 0
;  183          0310  2     THEN
;  184          0311  2         RETURN;
;  185          0312  2
;  186        U 0313  2 %IF FLIP %THEN
;  187        U 0314  2     IF (.gca_op_dev EQL op_dev_flip)
;  188        U 0315  2     THEN
;  189        U 0316  2         BEGIN
;  190        U 0317  2         flip_record [tcxtxt_code] = flip$k_tcxtxt;
;  191        U 0318  2         flip_record [tcxtxt_length] = .entry_length;
```

```
;  192          U 0319  2          flip_record [tcxtxt_xtn] = .xtn;
;  193          U 0320  2          flip_record [tcxtxt_bar] = .bar_flag;
;  194          U 0321  2
;  195          U 0322  2          ch$move (.entry_length, CH$PTR (.entry_address),
;  196          U 0323  2              CH$PTR (flip_record[tcxtxt_text]));
;  197          U 0324  2
;  198          U 0325  2          $XPO_PUT ( IOB=.rnoiob, STRING= (flip$k_tcxtxt_basesiz+.entry_length,
;  199          U 0326  2              CH$PTR (flip_record)));
;  200          U 0327  2
;  201          U 0328  2          RETURN
;  202          U 0329  2          END;
;  203            0330  2 %FI
;  204            0331  2
;  205            0332  2      ! Write the Record Group Header for this NDX record. The length of the
;  206            0333  2      ! described record is 4 plus the allocation of the text.
;  207            0334  2      ! Do count this record in the .BRN count.
;  208            0335  2      rgh (brn_index, 4 + ch$allocation (.entry_length), true);
;  209            0336  2
;  210            0337  2      !Create a one-word descriptor record.
;  211            0338  2      TEMP_RECORD [0] = INDEX_ENTRY;      !This says what comes next.
;  212            0339  2
;  213            0340  2      !Write a descriptor identifying what is about to come as an index entry.
;  214            0341  2      $XPO_PUT (IOB = BRNOOB, BINARY_DATA = (1, TEMP_RECORD));
;  215            0342  2
;  216            0343  2      !Generate a descriptor record for this entry.
;  217            0344  2      TEMP_RECORD [0] = .ENTRY_LENGTH;
;  218            0345  2      TEMP_RECORD [1] = .XTN;
;  219            0346  2      TEMP_RECORD [2] = .BAR_FLAG;
;  220            0347  2
;  221            0348  2      !Write out the descriptor record for this entry.
;  222            0349  2      $XPO_PUT (IOB = BRNOOB, BINARY_DATA = (3, TEMP_RECORD));
;  223            0350  2
;  224            0351  2      !Write out the index entry.
;  225            0352  2      $XPO_PUT (IOB = BRNOOB, BINARY_DATA = (CH$ALLOCATION (.ENTRY_LENGTH), .ENTRY_ADDRESS));
;  226            0353  2
;  227            0354  1      END;                                    !End of PUTNDY


                                                    .TITLE  PUTNDX
                                                    .IDENT  \V04-000\

                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                    00  00  79  64  6E  74  75  70  00000 P.AAA:    .ASCII  \putndy\<0><0>                          ;

                                                    .EXTRN  RNOIOB, BRNOOB, GCA
                                                    .EXTRN  TSF, RNFILE, ERMS
                                                    .EXTRN  XPO$PUT, XPO$FAILURE

                                                    .PSECT  $CODE$,NOWRT,2

                                        007C 00000          .ENTRY  PUTNDY, Save R2,R3,R4,R5,R6                 ; 0262
                        56 00000000G    EF   9E 00002       MOVAB   XPO$PUT, R6                                 ;
                        55 00000000G    EF   9E 00009       MOVAB   XPO$FAILURE, R5                             ;
                        54 00000000G    EF   9E 00010       MOVAB   IOB$+68, R4                                 ;
                        5E              14   C2 00017       SUBL2   #20, SP                                     ;
                        1E          EE  A4   E8 0001A       BLBS    BRNOOB+50, 2$                               ; 0297
```

```
                 01 00000000G   EF   E9 0001E             BLBC    GCA+228, 1$                                     ; 0300
                                    04 00025             RET
                          06    DD   00026 1$:           PUSHL   #6                                              ; 0302
                    00000000'   EF   9F 00028            PUSHAB  P.AAA
                    00000000G   8F   DD 0002E            PUSHL   #RNFILE
          00000000G EF          03   FB 00034            CALLS   #3, ERMS
                                    04 0003B             RET                                                     ; 0299
                 53       04   AC   D0 0003C 2$:        MOVL    ENTRY_LENGTH, R3                                ; 0309
                          01   14   00040                BGTR    3$
                                    04 00042             RET
                          01   DD   00043 3$:           PUSHL   #1                                              ; 0335
                 52       03   A3   9E 00045            MOVAB   3(R3), R2
                 52       04   C6   00049            DIVL2   #4, R2
                          04   A2   9F 0004C            PUSHAB  4(R2)
                          01   DD   0004F            PUSHL   #1
          00000000V EF          03   FB 00051            CALLS   #3, RGH
                08    AE          03   D0 00058            MOVL    #3, TEMP_RECORD                                 ; 0338
                6E              04   B0 0005C            MOVW    #4, $IOB$OUTPUT                                 ; 0341
                02    AE          02   90 0005F            MOVB    #2, $IOB$OUTPUT+2
                03    AE          01   90 00063            MOVB    #1, $IOB$OUTPUT+3
                04    AE    08    AE   9E 00067            MOVAB   TEMP_RECORD, $IOB$OUTPUT+4
                64              6E   9E 0006C            MOVAB   $IOB$OUTPUT, IOB$+68
                E8    A4          07   90 0006F            MOVB    #7, IOB$+44
                                    55   DD 00073         PUSHL   R5
                                    7E   D4 00075         CLRL    -(SP)
                          BC    A4   9F 00077            PUSHAB  IOB$
                          66        03   FB 0007A            CALLS   #3, XPO$PUT
                08    AE          53   D0 0007D            MOVL    R3, TEMP_RECORD                                 ; 0344
                0C    AE    0C    AC   7D 00081            MOVQ    XTN, TEMP_RECORD+4                              ; 0345
                6E              0C   B0 00086            MOVW    #12, $IOB$OUTPUT                                ; 0349
                02    AE          02   90 00089            MOVB    #2, $IOB$OUTPUT+2
                03    AE          01   90 0008D            MOVB    #1, $IOB$OUTPUT+3
                04    AE    08    AE   9E 00091            MOVAB   TEMP_RECORD, $IOB$OUTPUT+4
                64              6E   9E 00096            MOVAB   $IOB$OUTPUT, IOB$+68
                E8    A4          07   90 00099            MOVB    #7, IOB$+44
                                    55   DD 0009D         PUSHL   R5
                                    7E   D4 0009F         CLRL    -(SP)
                          BC    A4   9F 000A1            PUSHAB  IOB$
                          66        03   FB 000A4            CALLS   #3, XPO$PUT
          6E                52        04   A5 000A7            MULW3   #4, R2, $IOB$OUTPUT                     ; 0352
                02    AE          02   90 000AB            MOVB    #2, $IOB$OUTPUT+2
                03    AE          01   90 000AF            MOVB    #1, $IOB$OUTPUT+3
                04    AE    08    AC   D0 000B3            MOVL    ENTRY_ADDRESS, $IOB$OUTPUT+4
                64              6E   9E 000B8            MOVAB   $IOB$OUTPUT, IOB$+68
                E8    A4          07   90 000BB            MOVB    #7, IOB$+44
                                    55   DD 000BF         PUSHL   R5
                                    7E   D4 000C1         CLRL    -(SP)
                          BC    A4   9F 000C3            PUSHAB  IOB$
                          66        03   FB 000C6            CALLS   #3, XPO$PUT
                                    04   000C9            RET                                                     ; 0354

; Routine Size:  202 bytes,    Routine Base:  $CODE$ + 0000

;  228          0355  1
```

```
;  230          0356  1 GLOBAL ROUTINE PUTXTN (PAGE_REF, XTN) : NOVALUE =       !
;  231          0357  1
;  232          0358  1 !++
;  233          0359  1 ! FUNCTIONAL DESCRIPTION:
;  234          0360  1 !
;  235          0361  1 !       PUTXTN writes a transaction-number/page-reference pair
;  236          0362  1 !       to the index file.
;  237          0363  1 !
;  238          0364  1 ! FORMAL PARAMETERS:    None
;  239          0365  1 !
;  240          0366  1 ! IMPLICIT INPUTS:      None
;  241          0367  1 !
;  242          0368  1 ! IMPLICIT OUTPUTS:     None
;  243          0369  1 !
;  244          0370  1 ! ROUTINE VALUE:
;  245          0371  1 ! COMPLETION CODES:     None
;  246          0372  1 !
;  247          0373  1 ! SIDE EFFECTS:         None
;  248          0374  1 !
;  249          0375  1 !--
;  250          0376  1
;  251          0377  2     BEGIN
;  252          0378  2
;  253          0379  2     MAP
;  254          0380  2         PAGE_REF : REF VECTOR [PAGE_SCT_SIZE];
;  255          0381  2
;  256          0382  2     LOCAL
;  257          0383  2         TEMP_RECORD : VECTOR [1 + PAGE_SCT_SIZE];
;  258        U 0384  2 %IF FLIP %THEN
;  259        U 0385  2     LOCAL
;  260        U 0386  2         FLIP_RECORD : $FLIP_TCXXTN;
;  261          0387  2 %FI
;  262          0388  2
;  263          0389  2     !Cursory check to make sure the file is opened.
;  264          0390  3     IF NOT CHECK_OPEN
;  265          0391  2     THEN
;  266          0392  3         BEGIN
;  267          0393  3         IF NOT .gca_black_box
;  268          0394  3         THEN
;  269          0395  3             ERMS (RNFILE, CH$PTR (UPLIT ('PUTXTN')), 6);
;  270          0396  3         RETURN;
;  271          0397  2         END;
;  272          0398  2
;  273        U 0399  2 %IF FLIP %THEN
;  274        U 0400  2     IF (.gca_op_dev EQL op_dev_flip)
;  275        U 0401  2     THEN
;  276        U 0402  2         BEGIN
;  277        U 0403  2         !Create a one-word descriptor record.
;  278        U 0404  2         FLIP_RECORD[TCXXTN_CODE] = INDEX_XTN;
;  279        U 0405  2         !Put transaction number into the temporary record.
;  280        U 0406  2         FLIP_RECORD[TCXXTN_XTN] = .XTN;
;  281        U 0407  2         !Copy the page number in too.
;  282        U 0408  2         INCR I FROM 0 TO PAGE_SCT_SIZE-1 DO
;  283        U 0409  2             VECTOR[FLIP_RECORD[TCXXTN_SCT],.I] = .PAGE_REF [.I];
;  284        U 0410  2
;  285        U 0411  2         FLIP_RECORD[TCXXTN_CODE] = FLIP$K_TCXXTN;
;  286        U 0412  2         $XPO_PUT (IOB=.RNOIOB, STRING=(FLIP$K_TCXXTN_SIZE,
```

```
;  287          U 0413  2                    CH$PTR(FLIP_RECORD)));
;  288          U 0414  2            RETURN
;  289          U 0415  2            END;
;  290             0416  2 %FI
;  291             0417  2
;  292             0418  2      ! Write the Record Group Header for this index record. The length of the
;  293             0419  2      ! described record is 2 plus PAGE_SCT_SIZE.
;  294             0420  2      ! Do not count this record in the .BRN count.
;  295             0421  2      rgh (brn_index, 2 + page_sct_size, false);
;  296             0422  2
;  297             0423  2      !Create a one-word descriptor record.
;  298             0424  2      TEMP_RECORD [0] = INDEX_XTN;        !This indicates what comes next
;  299             0425  2
;  300             0426  2      !Write the descriptor record.
;  301             0427  2      $XPO_PUT (IOB = BRNOOB, BINARY_DATA = (1, TEMP_RECORD));
;  302             0428  2
;  303             0429  2      !Put transaction number into the temporary record.
;  304             0430  2      TEMP_RECORD [0] = .XTN;
;  305             0431  2
;  306             0432  2      !Copy the page number in too.
;  307             0433  2      INCR I FROM 1 TO PAGE_SCT_SIZE DO
;  308             0434  2          TEMP_RECORD [.I] = .PAGE_REF [.I - 1];
;  309             0435  2
;  310             0436  2      $XPO_PUT (IOB = BRNOOB, BINARY_DATA = (1 + PAGE_SCT_SIZE, TEMP_RECORD));
;  311             0437  2
;  312             0438  1      END;                                    !End of PUTXTN


                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

00  00  4E  54  58  54  55  50  00008 P.AAB:        .ASCII  \PUTXTN\<0><0>                                  ;


                                                    .PSECT  $CODE$,NOWRT,2

                                001C 00000          .ENTRY  PUTXTN, Save R2,R3,R4                           ; 0356
                    54 00000000G  EF   9E 00002     MOVAB   XPO$PUT, R4                                     ;
                    53 00000000G  EF   9E 00009     MOVAB   XPO$FAILURE, R3                                 ;
                    52 00000000G  EF   9E 00010     MOVAB   IOB$+68, R2                                     ;
                    5E            1C   C2 00017     SUBL2   #28, SP                                         ;
                    1D        EE  A2   E8 0001A     BLBS    BRNOOB+50, 1$                                   ; 0390
                    7F 00000000G  EF   E8 0001E     BLBS    GCA+228, 3$                                     ; 0393
                                  06   DD 00025     PUSHL   #6                                              ; 0395
                       00000000'  EF   9F 00027     PUSHAB  P.AAB                                           ;
                       00000000G  8F   DD 0002D     PUSHL   #RNFILE                                         ;
00000000G  EF                    03   FB 00033     CALLS   #3, ERMS                                        ;
                                 04   0003A         RET                                                     ; 0392
                    7E           06   7D 0003B 1$:  MOVQ    #6, -(SP)                                       ; 0421
                                 01   DD 0003E     PUSHL   #1                                              ;
00000000V  EF                    03   FB 00040     CALLS   #3, RGH                                         ;
        08  AE                   02   D0 00047     MOVL    #2, TEMP_RECORD                                 ; 0424
            6E                   04   B0 0004B     MOVW    #4, $IOB$OUTPUT                                 ; 0427
        02  AE                   02   90 0004E     MOVB    #2, $IOB$OUTPUT+2                               ;
        03  AE                   01   90 00052     MOVB    #1, $IOB$OUTPUT+3                               ;
        04  AE           08      AE   9E 00056     MOVAB   TEMP_RECORD, $IOB$OUTPUT+4                      ;
```

```
                     62              6E  9E 0005B         MOVAB   $IOB$OUTPUT, IOB$+68                ;
                 E8  A2              07  90 0005E         MOVB    #7, IOB$+44                         ;
                                     53  DD 00062         PUSHL   R3                                  ;
                                     7E  D4 00064         CLRL    -(SP)                               ;
                                 BC  A2  9F 00066         PUSHAB  IOB$                                ;
                     64              03  FB 00069         CALLS   #3, XPO$PUT                         ;
                 08  AE          08  AC  D0 0006C         MOVL    XTN, TEMP_RECORD                    ; 0430
                     51              01  D0 00071         MOVL    #1, I                               ; 0434
                     50          04 BC41  DE 00074 2$:     MOVAL   @PAGE_REF[I], R0                    ;
                 08 AE41         FC  A0  D0 00079         MOVL    -4(R0), TEMP_RECORD[I]              ;
                F1               51      04  F3 0007F     AOBLEQ  #4, I, 2$                           ;
                                 6E      14  B0 00083     MOVW    #20, $IOB$OUTPUT                    ; 0436
                 02              AE      02  90 00086     MOVB    #2, $IOB$OUTPUT+2                   ;
                 03              AE      01  90 0008A     MOVB    #1, $IOB$OUTPUT+3                   ;
                 04              AE  08  AE  9E 0008E     MOVAB   TEMP_RECORD, $IOB$OUTPUT+4          ;
                                 62      6E  9E 00093     MOVAB   $IOB$OUTPUT, IOB$+68                ;
                 E8              A2      07  90 00096     MOVB    #7, IOB$+44                         ;
                                         53  DD 0009A     PUSHL   R3                                  ;
                                         7E  D4 0009C     CLRL    -(SP)                               ;
                                     BC  A2  9F 0009E     PUSHAB  IOB$                                ;
                                 64          03  FB 000A1 CALLS   #3, XPO$PUT                         ;
                                             04  000A4 3$:  RET                                       ; 0438

; Routine Size:  165 bytes,    Routine Base:  $CODE$ + 00CA
```

```
;  314     0439  1 %SBTTL 'RGH -- write a Record Group Header to the binary file.'
;  315     0440  1 GLOBAL ROUTINE RGH (HEADER, COUNT, INCREMENT_BRN_COUNT) : NOVALUE =
;  316     0441  1
;  317     0442  1 !++
;  318     0443  1 ! FUNCTIONAL DESCRIPTION:
;  319     0444  1 !
;  320     0445  1 !       RGH writes a .BRN "envelope" -- a Record Group Header -- to the
;  321     0446  1 !       binary file. This header describes the following binary record
;  322     0447  1 !       so that the utilities can skip over it if they do not need to
;  323     0448  1 !       interpret it. For example, the INDEX utility would skip over binary
;  324     0449  1 !       data that the header identified as contents-related.
;  325     0450  1 !
;  326     0451  1 !       RGH also increments the GCA counts of the information written to
;  327     0452  1 !       the .BRN file so that it can be reported, if requested, by RTERM.
;  328     0453  1 !
;  329     0454  1 ! FORMAL PARAMETERS:
;  330     0455  1 !
;  331     0456  1 !       HEADER  describes what kind of record follows. It is either
;  332     0457  1 !               BRN_INDEX (=1), BRN_CONTENTS (=2), or BRN_CROSSREF (=3).
;  333     0458  1 !
;  334     0459  1 !       COUNT   is the length of the following binary record group.
;  335     0460  1 !
;  336     0461  1 !       INCREMENT_BRN_COUNT  is TRUE if we should bump the count of items
;  337     0462  1 !               written to the .BRN file, and FALSE if not. Thus, it is the
;  338     0463  1 !               responsibility of the caller to determine whether this item
;  339     0464  1 !               written to the .BRN should be counted.
;  340     0465  1 !
;  341     0466  1 ! IMPLICIT INPUTS:      None
;  342     0467  1 !
;  343     0468  1 ! IMPLICIT OUTPUTS:     None
;  344     0469  1 !
;  345     0470  1 ! ROUTINE VALUE:
;  346     0471  1 ! COMPLETION CODES:     None
;  347     0472  1 !
;  348     0473  1 ! SIDE EFFECTS:         None
;  349     0474  1 !
;  350     0475  1 !--
;  351     0476  2     BEGIN
;  352     0477  2
;  353     0478  2     LOCAL
;  354     0479  2         temp_record : VECTOR [2];
;  355     0480  2
;  356     0481  2     ! Cursory check to make sure the file is opened.
;  357     0482  3     IF NOT check_open
;  358     0483  2     THEN
;  359     0484  3         BEGIN
;  360     0485  3         IF NOT .gca_black_box
;  361     0486  3         THEN
;  362     0487  3             erms (rnfile, CH$PTR (UPLIT ('RGH')), 3);
;  363     0488  3         RETURN;
;  364     0489  2         END;
;  365     0490  2
;  366     0491  2     ! Create a two-word descriptor record.
;  367     0492  2     temp_record [0] = .header;
;  368     0493  2     temp_record [1] = .count;
;  369     0494  2
;  370     0495  2     ! Now really write the Record Group Header.
```

```
;  371          0496  2     $XPO_PUT (IOB = brnoob, BINARY_DATA = (2, temp_record));
;  372          0497  2
;  373          0498  2     ! If requested by the caller, increment the count of the information
;  374          0499  2     ! written to the .BRN file.
;  375          0500  2     !
;  376          0501  2     IF .increment_brn_count
;  377          0502  2     THEN
;  378          0503  2         SELECTONE .header OF
;  379          0504  2             SET
;  380          0505  2             [brn_index] :       gca_index_cnt = .gca_index_cnt + 1;
;  381          0506  2             [brn_contents] :    gca_contents_cnt = .gca_contents_cnt + 1;
;  382        U 0507  2 %IF DSRPLUS %THEN
;  383        U 0508  2             [brn_crossref] :    gca_new_cref_cnt = .gca_new_cref_cnt + 1;
;  384          0509  2 %FI
;  385          0510  2             TES;
;  386          0511  2
;  387          0512  2     RETURN;
;  388          0513  1     END;                                    !End of RGH


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2
                                    00  48  47  52  00010 P.AAC:  .ASCII  \RGH\<0>                                ;

                                                        .PSECT  $CODE$,NOWRT,2
                                                001C 00000      .ENTRY  RGH, Save R2,R3,R4                      ; 0440
                                          54 00000000G  EF  9E 00002      MOVAB   GCA+228, R4                     ;
                                          53 00000000G  EF  9E 00009      MOVAB   BRNOOB+48, R3                   ;
                                          5E            10  C2 00010      SUBL2   #16, SP                         ;
                                          19        02  A3  E8 00013      BLBS    BRNOOB+50, 1$                   ; 0482
                                          64            64  E8 00017      BLBS    GCA+228, 3$                     ; 0485
                                                        03  DD 0001A      PUSHL   #3                              ; 0487
                                             00000000'  EF  9F 0001C      PUSHAB  P.AAC                           ;
                                             00000000G  8F  DD 00022      PUSHL   #RNFILE                         ;
                                00000000G    EF                03  FB 00028      CALLS   #3, ERMS                        ;
                                                        04  0002F      RET                                     ; 0484
                                          52        04  AC  D0 00030 1$:  MOVL    HEADER, R2                      ; 0492
                                    08    AE            52  D0 00034      MOVL    R2, TEMP_RECORD                 ;
                                    0C    AE        08  AC  D0 00038      MOVL    COUNT, TEMP_RECORD+4            ; 0493
                                          6E            08  B0 0003D      MOVW    #8, $IOB$OUTPUT                 ; 0496
                                    02    AE            02  90 00040      MOVB    #2, $IOB$OUTPUT+2               ;
                                    03    AE            01  90 00044      MOVB    #1, $IOB$OUTPUT+3               ;
                                    04    AE        08  AE  9E 00048      MOVAB   TEMP_RECORD, $IOB$OUTPUT+4      ;
                                    14    A3            6E  9E 0004D      MOVAB   $IOB$OUTPUT, IOB$+68            ;
                                    FC    A3            07  90 00051      MOVB    #7, IOB$+44                     ;
                                             00000000G  EF  9F 00055      PUSHAB  XPO$FAILURE                     ;
                                                        7E  D4 0005B      CLRL    -(SP)                           ;
                                                    D0  A3  9F 0005D      PUSHAB  IOB$                            ;
                                00000000G    EF                03  FB 00060      CALLS   #3, XPO$PUT             ;
                                          13        0C  AC  E9 00067      BLBC    INCREMENT_BRN_COUNT, 3$         ; 0501
                                          01            52  D1 0006B      CMPL    R2, #1                          ; 0505
                                                        05  12 0006E      BNEQ    2$                              ;
                                                FF7C    C4  D6 00070      INCL    GCA+96                          ;
```

```
                                        04 00074        RET                                          ;
                      02           52   D1 00075 2$:    CMPL    R2, #2                               ; 0506
                                   04   12 00078        BNEQ    3$                                   ;
                          FF78     C4   D6 0007A        INCL    GCA+92                               ;
                                        04 0007E 3$:    RET                                          ; 0513

; Routine Size:  127 bytes,    Routine Base:  $CODE$ + 016F


;  389          0514  1
;  390          0515  1 END                                          !End of module
;  391          0516  0 ELUDOM
```

```
;                             PSECT SUMMARY
;
;      Name                       Bytes                       Attributes
;
;   $PLIT$                           20  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $CODE$                          494  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;                        Library Statistics
;
;                                          -------- Symbols --------      Pages      Processing
;      File                                Total   Loaded   Percent      Mapped     Time
;
;   _$255$DUA28:[SYSLIB]XPORT.L32;1          590       99        16        252       00:00.1
;   _$255$DUA28:[RUNOFF.SRC]DSRLIB.L32;1    1248       16         1         86       00:00.3
```

```
;                             COMMAND QUALIFIERS

;      BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:PUTNDX/OBJ=OBJ$:PUTNDX MSRC$:PUTNDX/UPDATE=(ENH$:PUTNDX)

; Size:          494 code + 20 data bytes
; Run Time:         00:13.3
; Elapsed Time:     00:32.1
; Lines/CPU Min:     2324
; Lexemes/CPU-Min: 47094
; Memory Used:   110 pages
; Compilation Complete
```

PAGE
LIS

PERMUTE
LIS

PAGCMB
LIS

PANPA
LIS

RCS
LIS

POOL
LIS

RINIT
LIS

PERIOD
LIS

PUS
LIS

PUTNDX
LIS

PARSEP
LIS

PUTTOC
LIS

REQUIR
LIS

PAGMRG
LIS

PARAG
LIS

REPEAT
LIS