```
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**OUTXHR

```
  OOOOOO    UU      UU  TTTTTTTTTT  XX      XX  HH      HH  RRRRRRRR
  OOOOOO    UU      UU  TTTTTTTTTT  XX      XX  HH      HH  RRRRRRRR
OO      OO  UU      UU      TT      XX      XX  HH      HH  RR      RR
OO      OO  UU      UU      TT      XX      XX  HH      HH  RR      RR
OO      OO  UU      UU      TT        XX  XX    HH      HH  RR      RR
OO      OO  UU      UU      TT        XX  XX    HH      HH  RR      RR
OO      OO  UU      UU      TT          XX      HHHHHHHHHH  RRRRRRRR
OO      OO  UU      UU      TT          XX      HHHHHHHHHH  RRRRRRRR
OO      OO  UU      UU      TT        XX  XX    HH      HH  RR  RR
OO      OO  UU      UU      TT        XX  XX    HH      HH  RR  RR
OO      OO  UU      UU      TT      XX      XX  HH      HH  RR    RR        ....
OO      OO  UU      UU      TT      XX      XX  HH      HH  RR    RR        ....
  OOOOOO    UUUUUUUUUU      TT      XX      XX  HH      HH  RR      RR      ....
  OOOOOO    UUUUUUUUUU      TT      XX      XX  HH      HH  RR      RR      ....
```

```
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 %TITLE 'Enters characters into index flag buffer'
;  2          0002  0 MODULE OUTXHR ( IDENT = 'V04-000'
;  3        P 0003  0 %BLISS32[,
;  4        P 0004  0         ADDRESSING_MODE(EXTERNAL=LONG_RELATIVE,NONEXTERNAL=LONG_RELATIVE)
;  5          0005  0         ]
;  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  OTHE    *
; 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 !++
; 33          0033  1 ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
; 34          0034  1 !
; 35          0035  1 ! ABSTRACT: Enter a single character into the Index flag buffer.
; 36          0036  1 !
; 37          0037  1 !
; 38          0038  1 ! ENVIRONMENT: Transportable
; 39          0039  1 !
; 40          0040  1 ! AUTHOR: R.W.Friday     CREATION DATE: December, 1978
; 41          0041  1 !
```

```
;    43          0042  1 %SBTTL 'Revision History'
;    44          0043  1 !
;    45          0044  1 ! MODIFIED BY:
;    46          0045  1 !
;    47          0046  1 !       010     REM00010        Ray Marshall    17-November-1983
;    48          0047  1 !               Modified the way we look at the character output.  We strip
;    49          0048  1 !                 off the high order bit so we can still check in the same
;    50          0049  1 !                 way if is printable.
;    51          0050  1 !               (I don't know what happened in IDENTs 7 through 9.  When I
;    52          0051  1 !                 started this edit, the IDENT was at 009.)
;    53          0052  1 !
;    54          0053  1 !       006     KAD00006        Keith Dawson    07-Mar-1983
;    55          0054  1 !               Global edit of all modules. Updated module names, idents,
;    56          0055  1 !               copyright dates. Changed require files to BLISS library.
;    57          0056  1 !
;    58          0057  1 !--
```

```
;    60          0058  1 %SBTTL 'Module Level Declarations'
;    61          0059  1
;    62          0060  1 !
;    63          0061  1 ! TABLE OF CONTENTS:
;    64          0062  1 !
;    65          0063  1 FORWARD ROUTINE
;    66          0064  1     OUTXPH : NOVALUE;
;    67          0065  1 !
;    68          0066  1 ! INCLUDE FILES:
;    69          0067  1 !
;    70          0068  1
;    71          0069  1 LIBRARY 'NXPORT:XPORT';                 ! XPORT Library
;    72          0070  1 REQUIRE 'REQ:RNODEF';                   ! RUNOFF variant definitions
;    73          0201  1
;    74        U 0202  1 %IF DSRPLUS %THEN
;    75        U 0203  1 LIBRARY 'REQ:DPLLIB';                 ! DSRPLUS BLISS Library
;    76          0204  1 %ELSE
;    77          0205  1 LIBRARY 'REQ:DSRLIB';                 ! DSR BLISS Library
;    78          0206  1 %FI
;    79          0207  1
;    80          0208  1 !
;    81          0209  1 ! MACROS:
;    82          0210  1 !
;    83          0211  1 !
;    84          0212  1 ! EQUATED SYMBOLS:
;    85          0213  1 !
;    86          0214  1
;    87          0215  1 EXTERNAL LITERAL
;    88          0216  1     RINTES : UNSIGNED (8);
;    89          0217  1
;    90          0218  1 EXTERNAL LITERAL                       !Error messages
;    91          0219  1     RNFITC;
;    92          0220  1
;    93          0221  1 !
;    94          0222  1 ! OWN STORAGE:
;    95          0223  1 !
;    96          0224  1 !
;    97          0225  1 ! EXTERNAL REFERENCES:
;    98          0226  1 !
;    99          0227  1
;   100          0228  1 EXTERNAL
;   101          0229  1     GCA : GCA_DEFINITION,
;   102          0230  1     SCA : SCA_DEFINITION,
;   103          0231  1     TSF : TSF_DEFINITION,
;   104          0232  1     MRA : REF FIXED_STRING,
;   105          0233  1     XMRA : FIXED_STRING,
;   106          0234  1     XTSF : VECTOR;
;   107          0235  1
;   108          0236  1 EXTERNAL ROUTINE
;   109          0237  1         ERML,
;   110          0238  1         OUTLIN;
;   111          0239  1
```

```
;   113     0240  1 GLOBAL ROUTINE OUTXHR (KHAR) : NOVALUE =
;   114     0241  1
;   115     0242  1 !++
;   116     0243  1 ! FUNCTIONAL DESCRIPTION:
;   117     0244  1 !
;   118     0245  1 !       OUTXHR takes KHAR and puts it, along with underlining codes,
;   119     0246  1 !       into the buffer where things marked with the Index flag
;   120     0247  1 !       are saved.
;   121     0248  1 !
;   122     0249  1 ! FORMAL PARAMETERS:
;   123     0250  1 !
;   124     0251  1 !       KHAR is the character to be output.
;   125     0252  1 !
;   126     0253  1 ! IMPLICIT INPUTS:
;   127     0254  1 !
;   128     0255  1 !       This code will continue to work correctly only if the only
;   129     0256  1 !       possible values that GCA_XCASE can have is ONE_CAP or LEAVE_CASE.
;   130     0257  1 !
;   131     0258  1 ! IMPLICIT OUTPUTS:     None
;   132     0259  1 !
;   133     0260  1 ! ROUTINE VALUE:
;   134     0261  1 ! COMPLETION CODES:     None
;   135     0262  1 !
;   136     0263  1 ! SIDE EFFECTS: None
;   137     0264  1 !
;   138     0265  1 !--
;   139     0266  1
;   140     0267  2     BEGIN
;   141     0268  2
;   142     0269  2     LOCAL
;   143     0270  2         HOLD_TSF,
;   144     0271  2         XCHAR;                  !Actual character to put in index buffer
;   145     0272  2
;   146     0273  2     !Assume character needs no translation.
;   147     0274  2     XCHAR = .KHAR;
;   148     0275  2     !Now see if it needs translation.
;   149     0276  2
;   150     0277  3     IF NOT (.SCA_FRC_CHR
;   151     0278  3               OR .SCA_WORD_SET
;   152     0279  3               OR .SCA_FRC_CASE
;   153     0280  3               OR (.GCA_XCASE EQL LEAVE_CASE))
;   154     0281  2     THEN
;   155     0282  2     !Character needs to be translated.
;   156     0283  2
;   157     0284  3         IF LETTER (.XCHAR)
;   158     0285  2         THEN
;   159     0286  2         !Only letters get translated, and this is one.
;   160     0287  2
;   161     0288  2             IF .FS_LENGTH (XMRA) EQL 0
;   162     0289  2             THEN
;   163     0290  2             !The first character of the word, if it's a letter,
;   164     0291  2             !gets translated to upper case.
;   165     0292  2             !NOTE:  Implicit here is that GCA_XCASE is specifying ONE_CAP
;   166     0293  2
;   167     0294  3                 IF LOWER_LETTER (.XCHAR)
;   168     0295  2                 THEN
;   169     0296  3                     XCHAR = UPPER_CASE (.XCHAR)
```

```
;  170          0297  2                  ELSE
;  171          0298  3                      (0)
;  172          0299  3
;  173          0300  2              ELSE
;  174          0301  2              !Other letters in the word get translated
;  175          0302  2              !to lower case.
;  176          0303  2
;  177          0304  3                  IF UPPER_LETTER (.XCHAR)
;  178          0305  2                  THEN
;  179          0306  2                      XCHAR = LOWER_CASE (.XCHAR);
;  180          0307  2
;  181          0308  2      !Switch to Index flag work area.
;  182          0309  2      HOLD_TSF = .TSF;
;  183          0310  2      TSF = XTSF;
;  184          0311  2
;  185          0312  2      !First, be sure there's room enough to save the entry.
;  186          0313  3      BEGIN
;  187          0314  3      LOCAL
;  188          0315  3          NEEDED;
;  189          0316  3
;  190          0317  3      NEEDED = 0;
;  191          0318  3
;  192          0319  3      IF .SCA_WRD_AC_UND
;  193          0320  3      THEN
;  194          0321  3          NEEDED = 3;
;  195          0322  3
;  196          0323  3      IF .SCA_WRD_AC_BLD
;  197          0324  3      THEN
;  198          0325  3          NEEDED = .NEEDED + 3;
;  199          0326  3
;  200          0327  3      NEEDED = .NEEDED + 1;
;  201          0328  3
;  202          0329  3      IF (.TSF_INT_HL + .NEEDED) GTR .FS_MAXSIZE (XMRA)
;  203          0330  3      THEN
;  204          0331  3      !Won't fit.  Tell user and terminate in the middle of this phrase.
;  205          0332  4          BEGIN
;  206          0333  4          TSF = .HOLD_TSF;                                !Restore "real" TSF.
;  207          0334  4          ERML(RNFITC);
;  208          0335  4          OUTXPH ();
;  209          0336  4          RETURN;
;  210          0337  3          END;
;  211          0338  2      END;
;  212          0339  2
;  213          0340  2      !Now, put the character into the Index flag buffer.
;  214          0341  2      !First, check for underlining.  Note that only underlining on a "per
;  215          0342  2      !character" basis is sticky enough to get into the index.  I.e., ^&A
;  216          0343  2      !gets indexed as "A", but &A really does get underlined.
;  217          0344  2      IF .SCA_WRD_AC_UND
;  218          0345  2      THEN
;  219          0346  2      !Underlining was forced for this character.
;  220          0347  3          BEGIN
;  221          0348  3          FS_WCHAR (XMRA, RINTES);
;  222          0349  3          FS_WCHAR (XMRA, %C'U');
;  223          0350  3          FS_WCHAR (XMRA, %C' ');
;  224          0351  3          TSF_INT_HL = .TSF_INT_HL + 3;
;  225          0352  2          END;
;  226          0353  2
```

```
;  227          0354  2     !Now put the character itself in.
;  228          0355  2     FS_WCHAR (XMRA, .XCHAR);
;  229          0356  2     TSF_INT_HL = .TSF_INT_HL + 1;
;  230          0357  2
;  231          0358  2     IF (.XCHAR AND %X'7F') GEQ %C' '
;  232          0359  2         AND (.XCHAR AND %X'7F') LSS %O'177'
;  233          0360  2     THEN
;  234          0361  2         TSF_EXT_HL = .TSF_EXT_HL + 1
;  235          0362  2     ELSE
;  236          0363  2         IF .XCHAR EQL %O'010'                   !Backspace???
;  237          0364  2         THEN
;  238          0365  2         !Back up for backspace.
;  239          0366  2             TSF_EXT_HL = .TSF_EXT_HL - 1;
;  240          0367  2
;  241          0368  2     !Switch back to primary buffer
;  242          0369  2     TSF = .HOLD_TSF;
;  243          0370  1     END;                                        !End of OUTXHR

                                                                .TITLE  OUTXHR Enters characters into index flag buffer
                                                                .IDENT  \V04-000\

                                                                .EXTRN  RINTES, RNFITC, GCA
                                                                .EXTRN  SCA, TSF, MRA, XMRA
                                                                .EXTRN  XTSF, ERML, OUTLIN

                                                                .PSECT  $CODE$,NOWRT,2

                                                 007C 00000     .ENTRY  OUTXHR, Save R2,R3,R4,R5,R6                     ; 0240
                                        56 00000000G   EF  9E 00002     MOVAB   SCA+200, R6
                                        55 00000000G   EF  9E 00009     MOVAB   TSF, R5
                                        54 00000000G   EF  9E 00010     MOVAB   XMRA+4, R4
                                        52       04    AC  D0 00017     MOVL    KHAR, XCHAR                             ; 0274
                                        63       10    A6  E8 0001B     BLBS    SCA+216, 4$                             ; 0277
                                        5F       98    A6  E8 0001F     BLBS    SCA+96, 4$                              ; 0278
                                        5B       D8    A6  E8 00023     BLBS    SCA+160, 4$                             ; 0279
                                           00000000G   FF  D5 00027     TSTL    @GCA+84                                 ; 0280
                                                       53  13 0002D     BEQL    4$
                                                       50  D4 0002F     CLRL    R0                                      ; 0284
                                00000041   8F           52  D1 00031     CMPL    XCHAR, #65
                                                       0B  19 00038     BLSS    1$
                                                       50  D6 0003A     INCL    R0
                                0000005A   8F           52  D1 0003C     CMPL    XCHAR, #90
                                                       12  15 00043     BLEQ    2$
                                00000061   8F           52  D1 00045 1$: CMPL    XCHAR, #97
                                                       34  19 0004C     BLSS    4$
                                0000007A   8F           52  D1 0004E     CMPL    XCHAR, #122
                                                       2B  14 00055     BGTR    4$
                                                 08    A4  D5 00057 2$: TSTL    XMRA+12                                 ; 0288
                                                       17  12 0005A     BNEQ    3$
                                00000061   8F           52  D1 0005C     CMPL    XCHAR, #97                              ; 0294
                                                       1D  19 00063     BLSS    4$
                                0000007A   8F           52  D1 00065     CMPL    XCHAR, #122
                                                       14  14 0006C     BGTR    4$
                                                   52  20  C2 0006E     SUBL2   #32, XCHAR                              ; 0296
                                                       0F  11 00071     BRB     4$                                      ; 0294
                                                   0C  50  E9 00073 3$: BLBC    R0, 4$                                  ; 0304
```

```
                0000005A  8F              52  D1 00076          CMPL    XCHAR, #90                              ;
                                          03  14 0007D          BGTR    4$                                      ;
                          52              20  C0 0007F          ADDL2   #32, XCHAR                              ; 0306
                          53              65  D0 00082 4$:      MOVL    TSF, HOLD_TSF                           ; 0309
                          65 00000000G    EF  9E 00085          MOVAB   XTSF, TSF                               ; 0310
                                          50  D4 0008C          CLRL    NEEDED                                  ; 0317
                    03            66      01  E1 0008E          BBC     #1, SCA+200, 5$                         ; 0319
                          50              03  D0 00092          MOVL    #3, NEEDED                              ; 0321
                          03              66  E9 00095 5$:      BLBC    SCA+200, 6$                             ; 0323
                          50              03  C0 00098          ADDL2   #3, NEEDED                              ; 0325
                                          50  D6 0009B 6$:      INCL    NEEDED                                  ; 0327
                          50        00    B5  C0 0009D          ADDL2   @TSF, R0                                ; 0329
                    04    A4              50  D1 000A1          CMPL    R0, XMRA+8                              ;
                                          18  15 000A5          BLEQ    7$                                      ;
                          65              53  D0 000A7          MOVL    HOLD_TSF, TSF                           ; 0333
                                00000000G 8F  DD 000AA          PUSHL   #RNFITC                                 ; 0334
                00000000G EF              01  FB 000B0          CALLS   #1, ERML                                ;
                00000000V EF              00  FB 000B7          CALLS   #0, OUTXPH                              ; 0335
                                          04  000BE          RET                                             ; 0342
                21            66          01  E1 000BF 7$:      BBC     #1, SCA+200, 8$                         ; 0344
                    00        B4    00G   8F  90 000C3          MOVB    #RINTES, @XMRA+4                        ; 0348
                                          64  D6 000C8          INCL    XMRA+4                                  ;
                                    08    A4  D6 000CA          INCL    XMRA+12                                 ;
                    00        B4    55    8F  90 000CD          MOVB    #85, @XMRA+4                            ; 0349
                                          64  D6 000D2          INCL    XMRA+4                                  ;
                                    08    A4  D6 000D4          INCL    XMRA+12                                 ;
                    00        B4          20  90 000D7          MOVB    #32, @XMRA+4                            ; 0350
                                          64  D6 000DB          INCL    XMRA+4                                  ;
                                    08    A4  D6 000DD          INCL    XMRA+12                                 ;
                    00        B5          03  C0 000E0          ADDL2   #3, @TSF                                ; 0351
                    00        B4          52  90 000E4 8$:      MOVB    XCHAR, @XMRA+4                          ; 0355
                                          64  D6 000E8          INCL    XMRA+4                                  ;
                                    08    A4  D6 000EA          INCL    XMRA+12                                 ;
                          50              65  D0 000ED          MOVL    TSF, R0                                 ;
                                          60  D6 000F0          INCL    (R0)                                    ; 0356
                20                52    07  00  ED 000F2          CMPZV   #0, #7, XCHAR, #32                      ; 0358
                                          10  19 000F7          BLSS    9$                                      ;
0000007F        8F                52    07  00  ED 000F9          CMPZV   #0, #7, XCHAR, #127                     ; 0359
                                          05  18 00102          BGEQ    9$                                      ;
                                    04    A0  D6 00104          INCL    4(R0)                                   ; 0361
                                          08  11 00107          BRB     10$                                     ;
                          08              52  D1 00109 9$:      CMPL    XCHAR, #8                               ; 0363
                                          03  12 0010C          BNEQ    10$                                     ;
                                    04    A0  D7 0010E          DECL    4(R0)                                   ; 0366
                          65              53  D0 00111 10$:     MOVL    HOLD_TSF, TSF                           ; 0369
                                          04  00114          RET                                             ; 0370

; Routine Size:  277 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  245     0371  1 GLOBAL ROUTINE OUTXPH : NOVALUE =
;  246     0372  1
;  247     0373  1 !++
;  248     0374  1 ! FUNCTIONAL DESCRIPTION:
;  249     0375  1 !
;  250     0376  1 !       OUTXPH is called to output a word collected by the action of
;  251     0377  1 !       the Index flag.  At the same time, it turns off the index flag.
;  252     0378  1 !       Note that this routine is currently NOT called when a .SUBINDEX,
;  253     0379  1 !       .INDEX, or .ENTRY command is being processed.  In later enhancements
;  254     0380  1 !       it could be made possible to allow the Index flag to be active during
;  255     0381  1 !       the processing of one of these commands.  However that is not the
;  256     0382  1 !       case at this time.
;  257     0383  1 !
;  258     0384  1 ! FORMAL PARAMETERS:     None
;  259     0385  1 !
;  260     0386  1 ! IMPLICIT INPUTS:       None
;  261     0387  1 !
;  262     0388  1 ! IMPLICIT OUTPUTS:      None
;  263     0389  1 !
;  264     0390  1 ! ROUTINE VALUE:
;  265     0391  1 ! COMPLETION CODES:      None
;  266     0392  1 !
;  267     0393  1 ! SIDE EFFECTS: None
;  268     0394  1 !
;  269     0395  1 !--
;  270     0396  1
;  271     0397  2     BEGIN
;  272     0398  2
;  273     0399  2     LOCAL
;  274     0400  2         TSF_HOLD,
;  275     0401  2         MRA_HOLD,
;  276     0402  2         XTN;
;  277     0403  2
;  278     0404  2     !Preserve current buffer status.
;  279     0405  2     TSF_HOLD = .TSF;
;  280     0406  2     MRA_HOLD = .MRA;
;  281     0407  2     !Set up buffers where Index flag is doing its work.
;  282     0408  2     TSF = XTSF;
;  283     0409  2     MRA = XMRA;
;  284     0410  2     !Output the collected word only if there is one.
;  285     0411  2
;  286     0412  2     IF .TSF_INT_HL NEQ 0
;  287     0413  2     THEN
;  288     0414  3         BEGIN
;  289     0415  3         !Allocate a transaction number for this entry.
;  290     0416  3
;  291     0417  3         XTN = .GCA_NORMAL_XTN;
;  292     0418  3         GCA_NORMAL_XTN = .GCA_NORMAL_XTN + 1;    !Bump for next index entry
;  293     0419  3
;  294     0420  3         !Now associate the transaction number with the entry.
;  295     0421  3         TSF_FIRST_XTN = .XTN;
;  296     0422  3         TSF_LAST_XTN = .XTN;
;  297     0423  3         !And finally, attach the same transaction number to the
;  298     0424  3         !word in the text to which this applies.
;  299     0425  3
;  300     0426  3         IF .SCA_WRD_F_XTN EQL 0
;  301     0427  3         THEN
```

```
;  302     0428  3              SCA_WRD_F_XTN = .XTN;
;  303     0429  3
;  304     0430  3          SCA_WRD_L_XTN = .XTN;
;  305     0431  3          !Make this TSF something that goes into the index.
;  306     0432  3          TSF_INDEX = TRUE;
;  307     0433  3          !And now go output this line.
;  308     0434  3          OUTLIN (FALSE);
;  309     0435  2          END;
;  310     0436  2
;  311     0437  2      !Retrieve previous buffer status.
;  312     0438  2      TSF = .TSF_HOLD;
;  313     0439  2      MRA = .MRA_HOLD;
;  314     0440  2      !Turn off the index flag now, and that's it until next time.
;  315     0441  2      !Note that SCA_X_FLAG is used only when the Index flag is
;  316     0442  2      !active.  Failure to turn it off will cause the program to
;  317     0443  2      !lose coordination with Index flag processing.  Typical
;  318     0444  2      !symptoms of this problem include the index table getting too
;  319     0445  2      !large, or, even more obscurely, normal text being rejected
;  320     0446  2      !because it's too complicated.
;  321     0447  2      SCA_X_FLAG = FALSE;
;  322     0448  2      RETURN;
;  323     0449  1      END;                                    !End of OUTXPH


                                00FC 00000          .ENTRY  OUTXPH, Save R2,R3,R4,R5,R6,R7          ; 0371
                    57 00000000G  EF   9E 00002     MOVAB   GCA+168, R7                             ;
                    56 00000000G  EF   9E 00009     MOVAB   MRA, R6                                 ;
                    55 00000000G  EF   9E 00010     MOVAB   TSF, R5                                 ;
                    54 00000000G  EF   9E 00017     MOVAB   SCA+296, R4                             ;
                    53            65   D0 0001E     MOVL    TSF, TSF_HOLD                           ; 0405
                    52            66   D0 00021     MOVL    MRA, MRA_HOLD                           ; 0406
                    65 00000000G  EF   9E 00024     MOVAB   XTSF, TSF                               ; 0408
                    66 00000000G  EF   9E 0002B     MOVAB   XMRA, MRA                               ; 0409
                    50            65   D0 00032     MOVL    TSF, R0                                 ; 0412
                                  60   D5 00035     TSTL    (R0)                                    ;
                                  25   13 00037     BEQL    2$                                      ;
                    51            67   D0 00039     MOVL    GCA+168, XTN                            ; 0417
                                  67   D6 0003C     INCL    GCA+168                                 ; 0418
              38    A0            51   D0 0003E     MOVL    XTN, 56(R0)                             ; 0421
              3C    A0            51   D0 00042     MOVL    XTN, 60(R0)                             ; 0422
                                  64   D5 00046     TSTL    SCA+296                                 ; 0426
                                  03   12 00048     BNEQ    1$                                      ;
                    64            51   D0 0004A     MOVL    XTN, SCA+296                            ; 0428
              04    A4            51   D0 0004D 1$: MOVL    XTN, SCA+300                            ; 0430
              14    A0            01   D0 00051     MOVL    #1, 20(R0)                              ; 0432
                                  7E   D4 00055     CLRL    -(SP)                                   ; 0434
     00000000G    EF            01   FB 00057     CALLS   #1, OUTLIN                              ;
                    65            53   D0 0005E 2$: MOVL    TSF_HOLD, TSF                           ; 0438
                    66            52   D0 00061     MOVL    MRA_HOLD, MRA                           ; 0439
                          FF74    C4   D4 00064     CLRL    SCA+156                                 ; 0447
                                       04 00068     RET                                             ; 0449

; Routine Size:  105 bytes,    Routine Base:  $CODE$ + 0115
```

```
;  324          0450  1
;  325          0451  1 END                                   .End of module
;  326          0452  0 ELUDOM
```

```
;                                  PSECT SUMMARY

;       Name                    Bytes                           Attributes

;  $CODE$                         382  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         0         0         252       00:00.1
;  _$255$DUA28:[RUNOFF.SRC]DSRLIB.L32;1            1248        40         3          86       00:00.3
```

```
;                                     COMMAND QUALIFIERS

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

; Size:          382 code + 0 data bytes
; Run Time:          00:08.5
; Elapsed Time:      00:19.5
; Lines/CPU Min:      3209
; Lexemes/CPU-Min:  17112
; Memory Used:   91 pages
; Compilation Complete
```

NEWPAG
LIS

NODOPX
LIS

OFT
LIS

OUTTXT
LIS

NDXVRS
LIS

NOTE
LIS

OUTLIN
LIS

NM
LIS

PACK
LIS

OUTXHR
LIS

NDXXTN
LIS

OUTCHA
LIS

OUTHDR
LIS