```
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
```

```
NN       NN  DDDDDDDD    XX       XX  MM       MM    SSSSSSSS    GGGGGGGG
NN       NN  DDDDDDDD    XX       XX  MM       MM    SSSSSSSS    GGGGGGGG
NN       NN  DD      DD  XX       XX  MMMM   MMMM  SS          GG
NN       NN  DD      DD  XX       XX  MMMM   MMMM  SS          GG
NNNN     NN  DD      DD    XX   XX    MM  MM  MM  SS          GG
NNNN     NN  DD      DD    XX   XX    MM  MM  MM  SS          GG
NN  NN   NN  DD      DD      XX       MM      MM    SSSSSS    GG
NN  NN   NN  DD      DD      XX       MM      MM    SSSSSS    GG
NN    NNNN  DD      DD    XX   XX    MM      MM          SS  GG  GGGGGG
NN    NNNN  DD      DD    XX   XX    MM      MM          SS  GG  GGGGGG
NN       NN  DD      DD  XX       XX  MM      MM          SS  GG      GG      ....
NN       NN  DD      DD  XX       XX  MM      MM          SS  GG      GG      ....
NN       NN  DDDDDDDD    XX       XX  MM      MM  SSSSSSSS      GGGGGG       ....
NN       NN  DDDDDDDD    XX       XX  MM      MM  SSSSSSSS      GGGGGG       ....


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 'NDXMSG -- Dump entry as message'
;     2        0002  0 MODULE NDXMSG (IDENT = 'V04-000'
;     3        0003  0                %BLISS32 [, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE)]
;     4        0004  0                ) =
;     5        0005  1 BEGIN
;     6        0006  1
;     7        0007  1 !
;     8        0008  1 !*****************************************************************************
;     9        0009  1 !*                                                                           *
;    10        0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11        0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12        0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13        0013  1 !*                                                                           *
;    14        0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND CO.IED    *
;    15        0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;    16        0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17        0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18        0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19        0019  1 !*  TRANSFERRED.                                                              *
;    20        0020  1 !*                                                                           *
;    21        0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22        0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23        0023  1 !*  CORPORATION.                                                              *
;    24        0024  1 !*                                                                           *
;    25        0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26        0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27        0027  1 !*                                                                           *
;    28        0028  1 !*                                                                           *
;    29        0029  1 !*****************************************************************************
;    30        0030  1 !
;    31        0031  1
;    32        0032  1 !++
;    33        0033  1 ! FACILITY:
;    34        0034  1 !   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
;    35        0035  1 !
;    36        0036  1 ! ABSTRACT:
;    37        0037  1 !   This module contains code to dump an index entry in internal format
;    38        0038  1 !
;    39        0039  1 ! ENVIRONMENT:   Transportable
;    40        0040  1 !
;    41        0041  1 ! AUTHOR:        JPK
;    42        0042  1 !
;    43        0043  1 ! CREATION DATE: January 1982
;    44        0044  1 !
;    45        0045  1 ! MODIFIED BY:
;    46        0046  1 !
;    47        0047  1 !       005     JPK00015        04-Feb-1983
;    48        0048  1 !               Cleaned up module names, modified revision history to
;    49        0049  1 !               conform with established standards. Updated copyright dates.
;    50        0050  1 !
;    51        0051  1 !       004     JPK00012        24-Jan-1983
;    52        0052  1 !               Modified NDXVMSMSG.MSG to define error messages for both
;    53        0053  1 !               DSRINDEX and INDEX.
;    54        0054  1 !               Added require of NDXVMSREQ.R32 to NDXOUT, NDXFMT, NDXDAT,
;    55        0055  1 !               INDEX, NDXMSG, NDXXTN, NDXTMS, NDXVMS and NDXPAG for BLISS32.
;    56        0056  1 !               Since this file defines the error message literals,
;    57        0057  1 !               the EXTERNAL REFERENCEs for the error message literals
```

```
;    58          0058  1 !                 have been removed.
;    59          0059  1 !
;    60          0060  1 !          003      JPK00010        24-Jan-1983
;    61          0061  1 !                   Removed routines GETDAT and UPDDAT from NDXDAT - they
;    62          0062  1 !                   performed no useful function. Removed references to these
;    63          0063  1 !                   routines from NDXOUT, NDXFMT, and NDXMSG.
;    64          0064  1 !                   Removed reference to XPOOL in NDXOUT - not used.
;    65          0065  1 !
;    66          0066  1 !          002      JPK00004        24-Sep-1982
;    67          0067  1 !                   Modified NDXOUT, NDXMSG, NDXFMT, and NDXDAT for TOPS-20.
;    68          0068  1 !                   Strings stored in the index pool use the first fullword
;    69          0069  1 !                   for their length. References to these strings were incorrect.
;    70          0070  1 !
;    71          0071  1 !--
```

```
;    73          0072  1 !
;    74          0073  1 ! TABLE OF CONTENTS:
;    75          0074  1 !
;    76          0075  1 FORWARD ROUTINE
;    77          0076  1     ENTMSG : NOVALUE,                ! Dump current entry
;    78          0077  1     DMPENT : NOVALUE;                ! Dump entry
;    79          0078  1 !
;    80          0079  1 ! INCLUDE FILES:
;    81          0080  1 !
;    82          0081  1
;    83          0082  1 LIBRARY 'NXPORT:XPORT';
;    84          0083  1
;    85          0084  1 REQUIRE 'REQ:NDXPOL';
;    86          0302  1
;    87        L 0303  1 %IF %BLISS (BLISS32)
;    88          0304  1 %THEN
;    89          0305  1
;    90          0306  1 REQUIRE 'REQ:NDXVMSREQ';
;    91          0587  1
;    92          0588  1 %FI
;    93          0589  1
;    94          0590  1 !
;    95          0591  1 ! MACROS:
;    96          0592  1 !
;    97          0593  1 !
;    98          0594  1 ! EQUATED SYMBOLS:
;    99          0595  1 !
;   100          0596  1 !
;   101          0597  1 ! OWN STORAGE:
;   102          0598  1 !
;   103          0599  1 !
;   104          0600  1 ! EXTERNAL REFERENCES:
;   105          0601  1 !
;   106          0602  1 EXTERNAL LITERAL
;   107          0603  1     RINTES : UNSIGNED (8),           ! RUNOFF escape sequence character
;   108          0604  1     MAXLST;                          ! Maximum subindex depth
;   109          0605  1
;   110          0606  1 EXTERNAL
;   111          0607  1     LSTPTR : REF $XE_BLOCK,          ! Pointer to current entry
;   112          0608  1     INDLVL,                          ! Current subindex level
;   113          0609  1     LSTSTK : VECTOR;                 ! Entry stack
```

```
;  115     0610  1 %SBTTL 'ENTMSG - Dump current index entry'
;  116     0611  1 GLOBAL ROUTINE ENTMSG : NOVALUE =
;  117     0612  1 !++
;  118     0613  1 !
;  119     0614  1 ! FUNCTIONAL DESCRIPTION:
;  120     0615  1 !
;  121     0616  1 !       This routine builds a string which represents the current index
;  122     0617  1 !       entry from the index stack. The string is dumped via DMPENT.
;  123     0618  1 !
;  124     0619  1 ! FORMAL PARAMETERS:
;  125     0620  1 !
;  126     0621  1 !       None
;  127     0622  1 !
;  128     0623  1 ! IMPLICIT INPUTS:
;  129     0624  1 !
;  130     0625  1 !       LSTPTR  - Pointer to current entry
;  131     0626  1 !       INDLVL  - Subindex level
;  132     0627  1 !       LSTSTK  - Index entry stack
;  133     0628  1 !
;  134     0629  1 ! IMPLICIT OUTPUTS:
;  135     0630  1 !
;  136     0631  1 !       None
;  137     0632  1 !
;  138     0633  1 ! ROUTINE VALUE:
;  139     0634  1 ! COMPLETION CODES:
;  140     0635  1 !
;  141     0636  1 !       None
;  142     0637  1 !
;  143     0638  1 ! SIDE EFFECTS:
;  144     0639  1 !
;  145     0640  1 !       None
;  146     0641  1 !--
;  147     0642  2     BEGIN
;  148     0643  2     LOCAL
;  149     0644  2         LINE : VECTOR [CH$ALLOCATION (1200)],
;  150     0645  2         LEN,
;  151     0646  2         PTR;
;  152     0647  2
;  153     0648  2     !
;  154     0649  2     ! Build a string containing the whole entry
;  155     0650  2     !
;  156     0651  2     LEN = 0;
;  157     0652  2     PTR = CH$PTR (LINE);
;  158     0653  2
;  159     0654  2     LSTSTK [.INDLVL] = .LSTPTR; ! Save pointer to current entry
;  160     0655  2
;  161     0656  2     INCR I FROM 0 TO .INDLVL DO
;  162     0657  3         BEGIN
;  163     0658  3         LOCAL
;  164     0659  3             XEPTR : REF $XE_BLOCK,
;  165     0660  3             FROM_VEC : REF VECTOR,
;  166     0661  3             FROM_LEN,
;  167     0662  3             FROM_PTR;
;  168     0663  3
;  169     0664  3         !
;  170     0665  3         ! Point to entry
;  171     0666  3         !
```

```
;  172          0667  3          X'PTR = .LSTSTK [.I];
;  173          0668  3
;  174          0669  3          !
;  175          0670  3          ! P..nt to text
;  176          0671  3          !
;  177          0672  3          FROM_VEC = .XEPTR [XE$A_TEXT];
;  178          0673  3
;  179          0674  3          !
;  180          0675  3          ! Get the length. It's stored as the first fullword of the string
;  181          0676  3          !
;  182          0677  3          FROM_LEN = .FROM_VEC [0];
;  183          0678  3          FROM_PTR = CH$PTR (FROM_VEC [1]);
;  184          0679  3
;  185          0680  3          !
;  186          0681  3          ! Append to line
;  187          0682  3          !
;  188          0683  3          CH$MOVE (.FROM_LEN, .FROM_PTR, .PTR);
;  189          0684  3          PTR = CH$PLUS (.PTR, .FROM_LEN);
;  190          0685  3          LEN = .LEN + .FROM_LEN;
;  191          0686  3
;  192          0687  3          IF .I NEQ .INDLVL
;  193          0688  3          THEN
;  194          0689  4              BEGIN
;  195          0690  4              !
;  196          0691  4              ! Append subindex escape sequence
;  197          0692  4              !
;  198          0693  4              CH$WCHAR_A (RINTES, PTR);
;  199          0694  4              CH$WCHAR_A (%C'J', PTR);
;  200          0695  4              CH$WCHAR_A (0, PTR);
;  201          0696  4              LEN = .LEN + 3;
;  202          0697  3              END;
;  203          0698  2          END;
;  204          0699  2
;  205          0700  2      LSTSTK [.INDLVL] = 0;                ! Clean stack
;  206          0701  2
;  207          0702  2      DMPENT (.LEN, CH$PTR (LINE));        ! Dump entry
;  208          0703  1      END;


                                                                  .TITLE  NDXMSG NDXMSG -- Dump entry as message
                                                                  .IDENT  \V04-000\

                                                                  .EXTRN  DSRINDEX$_BADLOGIC
                                                                  .EXTRN  DSRINDEX$_BADVALUE
                                                                  .EXTRN  DSRINDEX$_INSVIRMEM
                                                                  .EXTRN  DSRINDEX$_LINELENG
                                                                  .EXTRN  DSRINDEX$_NOREF
                                                                  .EXTRN  DSRINDEX$_OPENIN
                                                                  .EXTRN  DSRINDEX$_OPENOUT
                                                                  .EXTRN  DSRINDEX$_TOOMANY
                                                                  .EXTRN  DSRINDEX$_VALERR
                                                                  .EXTRN  DSRINDEX$_CANTBAL
                                                                  .EXTRN  DSRINDEX$_CLOSEQUOT
                                                                  .EXTRN  DSRINDEX$_CONFQUAL
                                                                  .EXTRN  DSRINDEX$_CTRLCHAR
                                                                  .EXTRN  DSRINDEX$_DOESNTFIT
                                                                  .EXTRN  DSRINDEX$_DUPBEGIN
```

```
                                        .EXTRN  DSRINDEX$_EMPTYIN
                                        .EXTRN  DSRINDEX$_IGNORED
                                        .EXTRN  DSRINDEX$_INVINPUT
                                        .EXTRN  DSRINDEX$_INVRECORD
                                        .EXTRN  DSRINDEX$_LASTCONT
                                        .EXTRN  DSRINDEX$_NOBEGIN
                                        .EXTRN  DSRINDEX$_NOEND
                                        .EXTRN  DSRINDEX$_NOINDEX
                                        .EXTRN  DSRINDEX$_NOLIST
                                        .EXTRN  DSRINDEX$_OVERSTRK
                                        .EXTRN  DSRINDEX$_SKIPPED
                                        .EXTRN  DSRINDEX$_SYNTAX
                                        .EXTRN  DSRINDEX$_TEXFILE
                                        .EXTRN  DSRINDEX$_TOODEEP
                                        .EXTRN  DSRINDEX$_TOOFEW
                                        .EXTRN  DSRINDEX$_TRUNCATED
                                        .EXTRN  DSRINDEX$_COMPLETE
                                        .EXTRN  DSRINDEX$_CREATED
                                        .EXTRN  DSRINDEX$_IDENT
                                        .EXTRN  DSRINDEX$_PROCFILE
                                        .EXTRN  DSRINDEX$_TEXT, DSRINDEX$_TEXTD
                                        .EXTRN  DSRINDEX$_TMS11
                                        .EXTRN  RINTES, MAXLST, LSTPTR
                                        .EXTRN  INDLVL, LSTSTK

                                        .PSECT  $CODE$,NOWRT,2

                          0FFC 00000    .ENTRY  ENTMSG, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-  ; 0611
                                                R11
                     5B 00000000G   EF  9E 00002        MOVAB   LSTSTK, R11
                     5E      FB50   CE  9E 00009        MOVAB   -1200(SP), SP
                                    5A  D4 0000E        CLRL    LEN                         ; 0651
                     59             6E  9E 00010        MOVAB   LINE, PTR                   ; 0652
                     56 00000000G   EF  D0 00013        MOVL    INDLVL, R6                  ; 0654
                   6B46 00000000G   EF  D0 0001A        MOVL    LSTPTR, LSTSTK[R6]
                     58             01  CE 00022        MNEGL   #1, I                       ; 0656
                                    25  11 00025        BRB     2$
                     50         6B48   D0 00027 1$:     MOVL    LSTSTK[I], XEPTR            ; 0667
                     50      10    A0  D0 0002B        MOVL    16(XEPTR), FROM_VEC         ; 0672
                     57            80  D0 0002F        MOVL    (FROM_VEC)+, FROM_LEN       ; 0677
                69   60            57  28 00032        MOVC3   FROM_LEN, (FROM_PTR), (PTR) ; 0683
                     59            57  C0 00036        ADDL2   FROM_LEN, PTR               ; 0684
                     5A            57  C0 00039        ADDL2   FROM_LEN, LEN               ; 0685
                     56            58  D1 0003C        CMPL    I, R6                       ; 0687
                                   0B  13 0003F        BEQL    2$
                     89      00G   8F  90 00041        MOVB    #RINTES, (PTR)+             ; 0693
                     89      4A    8F  9B 00045        MOVZBW  #74, (PTR)+                 ; 0694
                     5A            03  C0 00049        ADDL2   #3, LEN                     ; 0696
                D7   58            56  F3 0004C 2$:     AOBLEQ  R6, I, 1$                   ; 0656
                                 6B46  D4 00050        CLRL    LSTSTK[R6]                  ; 0700
                           4400   8F  BB 00053        PUSHR   #^M<R10,SP>                 ; 0702
                00000000V  EF        02  FB 00057        CALLS   #2, DMPENT
                                       04 0005E        RET                                 ; 0703

; Routine Size:  95 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  210     0704  1 GLOBAL ROUTINE DMPENT (LEN, STR_PTR) : NOVALUE =
;  211     0705  1
;  212     0706  1 !++
;  213     0707  1 ! FUNCTIONAL DESCRIPTION:
;  214     0708  1 !
;  215     0709  1 !       DMPENT untranslates text and dumps it
;  216     0710  1 !
;  217     0711  1 ! FORMAL PARAMETERS:
;  218     0712  1 !
;  219     0713  1 !       LEN     - is the number of bytes to be scanned.
;  220     0714  1 !       STR_PTR - is a CH$PTR to the index entry
;  221     0715  1 !
;  222     0716  1 ! IMPLICIT INPUTS:
;  223     0717  1 !
;  224     0718  1 !       NONE
;  225     0719  1 !
;  226     0720  1 ! IMPLICIT OUTPUTS:
;  227     0721  1 !
;  228     0722  1 !       NONE
;  229     0723  1 !
;  230     0724  1 ! ROUTINE VALUE:
;  231     0725  1 ! COMPLETION CODES:
;  232     0726  1 !
;  233     0727  1 !       NONE
;  234     0728  1 !
;  235     0729  1 ! SIDE EFFECTS:
;  236     0730  1 !
;  237     0731  1 !       NONE
;  238     0732  1 !
;  239     0733  1 !--
;  240     0734  1
;  241     0735  2     BEGIN
;  242     0736  2
;  243     0737  2     LOCAL
;  244     0738  2         COUNT,                                  ! Number of characters in output stream.
;  245     0739  2         XLP,                                    ! A CH$PTR to the next character location in XLINE.
;  246     0740  2         XLINE : VECTOR [CH$ALLOCATION (1200)],  ! Build up the dump line here.
;  247     0741  2         PTR;
;  248     0742  2
;  249     0743  2     IF .LEN EQL 0 THEN RETURN;                  ! Split if no text to display
;  250     0744  2
;  251     0745  2     XLP = CH$PTR (XLINE);                       ! Point to line buffer
;  252     0746  2     COUNT = 0;                                  ! Initialize length counter
;  253     0747  2
;  254     0748  2     !
;  255     0749  2     ! Scan the text, untranslating escape codes, etc back to RUNOFF flags.
;  256     0750  2     !
;  257     0751  2     PTR = .STR_PTR;                             ! Copy pointer to what's to be "unparsed"
;  258     0752  2
;  259     0753  2     INCR I FROM 1 TO .LEN DO
;  260     0754  3         BEGIN
;  261     0755  3
;  262     0756  3         LOCAL
;  263     0757  3             KHAR;
;  264     0758  3
;  265     0759  3         KHAR = CH$RCHAR_A (PTR);
;  266     0760  3
```

```
;  267     0761  3          IF .KHAR EQL RINTES
;  268     0762  3          THEN
;  269     0763  4              BEGIN
;  270     0764  4              !
;  271     0765  4              ! Untranslate special function
;  272     0766  4              !
;  273     0767  4
;  274     0768  4              LOCAL
;  275     0769  4                  FUNCTION_CODE,
;  276     0770  4                  OPERAND;
;  277     0771  4
;  278     0772  4              FUNCTION_CODE = CH$RCHAR_A (PTR);
;  279     0773  4              OPERAND = CH$RCHAR_A (PTR);
;  280     0774  4              I = .I + 2;
;  281     0775  4
;  282     0776  4              SELECTONE .FUNCTION_CODE OF
;  283     0777  4                  SET
;  284     0778  4
;  285     0779  4                  [%C'B'] :
;  286     0780  4                      !
;  287     0781  4                      ! Bolded character.
;  288     0782  4                      !
;  289     0783  4                      CH$WCHAR_A (%C'*', XLP);
;  290     0784  4
;  291     0785  4                  [%C'U'] :
;  292     0786  4                      !
;  293     0787  4                      ! Underlined character.
;  294     0788  4                      !
;  295     0789  4                      CH$WCHAR_A (%C'&', XLP);
;  296     0790  4
;  297     0791  4                  [%C'O'] :
;  298     0792  5                      BEGIN
;  299     0793  5                      !
;  300     0794  5                      ! An overstruck character.
;  301     0795  5                      ! NOTE: Order is the reverse of what user specified.
;  302     0796  5                      !
;  303     0797  5                      CH$WCHAR_A (.OPERAND, XLP);
;  304     0798  5                      CH$WCHAR_A (%C'%', XLP);
;  305     0799  5                      COUNT = .COUNT + 1;
;  306     0800  4                      END;
;  307     0801  4
;  308     0802  4                  [%C'J'] :
;  309     0803  4                      !
;  310     0804  4                      ! A word mark.  For indexing commands, this
;  311     0805  4                      ! starts a new sub-indexing level.
;  312     0806  4                      !
;  313     0807  4                      CH$WCHAR_A (%C'>', XLP);
;  314     0808  4
;  315     0809  4                  [%C'P'] :
;  316     0810  4                      !
;  317     0811  4                      ! No permute flag
;  318     0812  4                      !
;  319     0813  4                      CH$WCHAR_A (%C'~', XLP);
;  320     0814  4
;  321     0815  4                  [%C'N'] :
;  322     0816  4                      IF .OPERAND EQL %C'-'
;  323     0817  4                      THEN
```

```
;   324          0818  4                          CH$WCHAR_A (%C'=', XLP)          ! Hyphenate flag
;   325          0819  4                      ELSE
;   326          0820  4                          CH$WCHAR_A (%C'?', XLP);         ! Unknown sequence
;   327          0821  4
;   328          0822  4                  [OTHERWISE] :
;   329          0823  4                      !
;   330          0824  4                      ! Unknown/unsupported special function
;   331          0825  4                      !
;   332          0826  4                      CH$WCHAR_A (%C'?', XLP);
;   333          0827  4                  TES;
;   334          0828  4
;   335          0829  4              END
;   336          0830  3          ELSE
;   337          0831  4              BEGIN
;   338          0832  4              !
;   339          0833  4              ! Some normal character
;   340          0834  4              !
;   341          0835  4              ! Normal characters go out as themselves.  Control characters
;   342          0836  4              ! are translated to something else.
;   343          0837  4              !
;   344          0838  4
;   345          0839  5              IF (.KHAR GEQ %C' ') AND (.KHAR LEQ %O'176')
;   346          0840  4              THEN
;   347          0841  4                  !
;   348          0842  4                  ! Output a normal character
;   349          0843  4                  !
;   350          0844  4                  CH$WCHAR_A (.KHAR, XLP)
;   351          0845  4              ELSE
;   352          0846  5                  BEGIN
;   353          0847  5                  !
;   354          0848  5                  ! Translate and output a control character.
;   355          0849  5                  ! NOTE: DEL (Octal 177) and NUL (Octal 0) have the same result.
;   356          0850  5                  !
;   357          0851  5                  CH$WCHAR_A (%C'^', XLP);
;   358          0852  5                  CH$WCHAR_A (.KHAR + %C'@', XLP);
;   359          0853  5                  COUNT = .COUNT + 1;
;   360          0854  4                  END;
;   361          0855  4
;   362          0856  3              END;
;   363          0857  3
;   364          0858  3          COUNT = .COUNT + 1;
;   365          0859  2          END;
;   366          0860  2
;   367          0861  2      !
;   368          0862  2      ! Now tell user what entry was
;   369          0863  2      !
;   370        L 0864  2 %IF %BLISS (BLISS32)
;   371          0865  2 %THEN                                               ! Signal messages in BLISS32
;   372          0866  2
;   373          0867  2      SIGNAL (INDEX$_TEXTD, 2, .COUNT, CH$PTR (XLINE));
;   374          0868  2
;   375        U 0869  2 %ELSE                                               ! Use $XPO_PUT_MSG otherwise
;   376        U 0870  2
;   377        U 0871  2      $XPO_PUT_MSG (SEVERITY = WARNING,
;   378        U 0872  2          STRING = $STR_CONCAT ('entry text: ''', (.COUNT, CH$PTR (XLINE)), ''''));
;   379        U 0873  2
;   380          0874  2 %FI
```

```
;  381     0875  2
;  382     0876  1     END;                                    !End of DMPENT


                                 007C 00000          .ENTRY  DMPENT, Save R2,R3,R4,R5,R6                     ; 0704
                          5E FB50  CE  9E 00002      MOVAB   -1200(SP), SP
                             04    AC  D5 00007      TSTL    LEN                                             ; 0743
                             01    12  12 0000A      BNEQ    1$
                                   04  0000C      RET
                          50       6E  9E 0000D 1$:  MOVAB   XLINE, XLP                                      ; 0745
                          56    08 AC  D0 00010      MOVL    STR_PTR, PTR                                    ; 0751
                                54 7C  00014        CLRQ    COUNT                                           ; 0746
                                009B 31  00016      BRW     13$                                             ; 0753
                          52       86  9A 00019 2$:  MOVZBL  (PTR)+, KHAR                                   ; 0759
                00000000G 8F       52  D1 0001C      CMPL    KHAR, #RINTES                                   ; 0761
                                   6D  12 00023      BNEQ    9$
                          51       86  9A 00025      MOVZBL  (PTR)+, FUNCTION_CODE                           ; 0772
                          53       86  9A 00028      MOVZBL  (PTR)+, OPERAND                                 ; 0773
                          55       02  C0 0002B      ADDL2   #2, I                                           ; 0774
                00000042  8F       51  D1 0002E      CMPL    FUNCTION_CODE, #66                              ; 0779
                                   05  12 00035      BNEQ    3$
                          60       2A  90 00037      MOVB    #42, (XLP)                                      ; 0783
                                   74  11 0003A      BRB     11$
                00000055  8F       51  D1 0003C 3$:  CMPL    FUNCTION_CODE, #85                              ; 0785
                                   05  12 00043      BNEQ    4$
                          60       26  90 00045      MOVB    #38, (XLP)                                      ; 0789
                                   66  11 00048      BRB     11$
                0000004F  8F       51  D1 0004A 4$:  CMPL    FUNCTION_CODE, #79                              ; 0791
                                   0A  12 00051      BNEQ    5$
                          80       53  90 00053      MOVB    OPERAND, (XLP)+                                 ; 0797
                          80       25  90 00056      MOVB    #37, (XLP)+                                     ; 0798
                                   54  D6 00059      INCL    COUNT                                           ; 0799
                                   55  11 0005B      BRB     12$                                             ; 0776
                0000004A  8F       51  D1 0005D 5$:  CMPL    FUNCTION_CODE, #74                              ; 0802
                                   05  12 00064      BNEQ    6$
                          60       3E  90 00066      MOVB    #62, (XLP)                                      ; 0807
                                   45  11 00069      BRB     11$
                00000050  8F       51  D1 0006B 6$:  CMPL    FUNCTION_CODE, #80                              ; 0809
                                   06  12 00072      BNEQ    7$
                          60    7E 8F  90 00074      MOVB    #126, (XLP)                                     ; 0813
                                   36  11 00078      BRB     11$
                0000004E  8F       51  D1 0007A 7$:  CMPL    FUNCTION_CODE, #78                              ; 0815
                                   0A  12 00081      BNEQ    8$
                          2D       53  D1 00083      CMPL    OPERAND, #45                                    ; 0816
                                   05  12 00086      BNEQ    8$
                          60       3D  90 00088      MOVB    #61, (XLP)                                      ; 0818
                                   23  11 0008B      BRB     11$                                             ; 0820
                          60       3F  90 0008D 8$:  MOVB    #63, (XLP)
                                   1E  11 00090      BRB     11$                                             ; 0818
                          20       52  D1 00092 9$:  CMPL    KHAR, #32                                       ; 0839
                                   0E  19 00095      BLSS    10$
                0000007E  8F       52  D1 00097      CMPL    KHAR, #126
                                   05  14 0009E      BGTR    10$
                          60       52  90 000A0      MOVB    KHAR, (XLP)                                     ; 0844
```

```
                                          0B  11 000A3        BRB     11$                          ;
                          80        5E    8F  90 000A5 10$:   MOVB    #94, (XLP)+                  ; 0851
              60          52        40    8F  81 000A9        ADDB3   #64, KHAR, (XLP)             ; 0852
                                          54  D6 000AE        INCL    COUNT                        ; 0853
                                          50  D6 000B0 11$:   INCL    XLP                          ; 0844
                                          54  D6 000B2 12$:   INCL    COUNT                        ; 0858
   FF5E       55          01        04    AC  F1 000B4 13$:   ACBL    LEN, #1, I, 2$               ; 0753
                                  4010    8F  BB 000BB        PUSHR   #^M<R4,SP>                   ; 0867
                                          02  DD 000BF        PUSHL   #2                           ;
                             00000000G    8F  DD 000C1        PUSHL   #DSRINDEX$_TEXTD             ;
                    00000000G  00         04  FB 000C7        CALLS   #4, LIB$SIGNAL               ;
                                              04 000CE        RET                                  ; 0876

; Routine Size:  207 bytes,    Routine Base:  $CODE$ + 005F


;   383          0877  1
;   384          0878  1 END                                  ! End of module
;   385          0879  0 ELUDOM



                                                              .EXTRN  LIB$SIGNAL

;                                  PSECT SUMMARY
;
;       Name                     Bytes                       Attributes
;
;   $CODE$                           302  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        29          4        252         00:00.1




;                                  COMMAND QUALIFIERS

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

; Size:          302 code + 0 data bytes
; Run Time:         00:13.8
; Elapsed Time:     00:28.6
; Lines/CPU Min:     3827
; Lexemes/CPU-Min: 51783
; Memory Used:  122 pages
```

```
; Compilation Complete
```

0344 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NDXMSG
LIS

NDXOUT
LIS

NDXINI
LIS