```
RRRRRRRRRRRR   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
RRRRRRRRRRRR   UUU         UUU  NNN   NNN   NNN 000         000 FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR   UUU         UUU  NNN   NNN   NNN 000         000 FFFFFFFFFFFF     FFFFFFFFFFFF
RRRRRRRRRRRR   UUU         UUU  NNN   NNN   NNN 000         000 FFFFFFFFFFFF     FFFFFFFFFFFF
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
```

```
_$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
STK
STR
STR
STR
STR
STR
STR
STR
STR
STR
STR
```

FILEID**NDXFMT


```
NN      NN DDDDDDDD   XX      XX FFFFFFFFFF MM      MM TTTTTTTTTT
NN      NN DDDDDDDD   XX      XX FFFFFFFFFF MM      MM TTTTTTTTTT
NN      NN DD      DD XX      XX FF         MMMM  MMMM     TT
NN      NN DD      DD XX      XX FF         MMMM  MMMM     TT
NNNN    NN DD      DD   XX  XX   FF         MM  MM  MM     TT
NNNN    NN DD      DD   XX  XX   FF         MM  MM  MM     TT
NN  NN  NN DD      DD     XX     FFFFFFFF   MM      MM     TT
NN  NN  NN DD      DD     XX     FFFFFFFF   MM      MM     TT
NN    NNNN DD      DD   XX  XX   FF         MM      MM     TT
NN    NNNN DD      DD   XX  XX   FF         MM      MM     TT
NN      NN DD      DD XX      XX FF         MM      MM     TT       ....
NN      NN DD      DD XX      XX FF         MM      MM     TT       ....
NN      NN DDDDDDDD   XX      XX FF         MM      MM     TT       ....
NN      NN DDDDDDDD   XX      XX FF         MM      MM     TT       ....

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
```

```
0 %TITLE 'NDXFMT -- Binary index formatter'
0 MODULE NDXFMT (IDENT = 'V04-000'
0                   %BLISS32 [, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE)]
0               ) =
1 BEGIN
1 !
1 !
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:
1 !   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
1 !
1 ! ABSTRACT:
1 !       This module contains the code to generate the output index lines
1 !       from the internal binary index. The output lines are formatted
1 !       into pages by module NDXPAG. This module was adopted from the TCX
1 !       modules XPASSA and XPASSB.
1 !
1 ! ENVIRONMENT:   Transportable
1 !
1 ! AUTHOR:        JPK
1 !
1 ! CREATION DATE: January 1982
1 !
1 ! MODIFIED BY:
1 !
1 !       013     JPK00022        30-Mar-1983
1 !               Modified NDXVMS, NDXFMT, NDXPAG, NDXVMSMSG and NDXVMSREQ
1 !               to generate TEX output. Added module NDXTEX.
1 !
1 !       012     JPK00021        28-Mar-1983
1 !               Modified NDXT20 to include E2.0 functionality.
1 !               Modified NDXCLIDMP, NDXFMT, NDXPAG, NDXVRS to require RNODEF
```

```
!               for BLISS36 and to remove any conditional require based on
!               DSRPLUS_DEF.
!
!       011     JPK00020        17-Mar-1983
!               Modified NDXFMT to generate a ".NO PAGING" in the prologue
!               for /LAYOUT=GALLEY
!
!       010     JPK00018        09-Mar-1983
!               Modified INDEX to handle new BRN format.
!               Modified NDXOUT to handle specifyable levels on SORT= string.
!               Modified NDXFMT to output new RUNOFF prologue.
!               Modified NDXPAG to output new TMS prologue and RUNOFF epilogue.
!
!       009     JPK00017        23-Feb-1983
!               Modified NDXINI to initialize the zero'th entries of LLINES,
!               RLINES and TLINES which is where the telltale strings are
!               stored by NDXFMT.
!               Modified NDXFMT to write appropriate prologue for /TELLTALE,
!               save the appropriate lines for left and right telltales, and
!               to mark the end of every entry with a NULL.
!               Modified NDXPAG to change the NULL following each entry to a
!               space if LAYOUT is SEPARATE or to a comma otherwise and to
!               generate and output telltales.
!
!       008     JPK00015        04-Feb-1983
!               Cleaned up module names, modified revision history to
!               conform with established standards. Updated copyright dates.
!
!       007     JPK00012        24-Jan-1983
!               Modified NDXVMSMSG.MSG to define error messages for both
!               DSRINDEX and INDEX.
!               Added require of NDXVMSREQ.R32 to NDXOUT, NDXFMT, NDXDAT,
!               INDEX, NDXMSG, NDXXTN, NDXTMS, NDXVMS and NDXPAG for BLISS32.
!               Since this file defines the error message literals,
!               the EXTERNAL REFERENCEs for the error message literals
!               have been removed.
!
!       006     JPK00011        24-Jan-1983
!               Changed CMDBLK [NDX$G_LEVEL] to CMDBLK [NDX$H_LEVEL]
!               Changed CMDBLK [NDX$H_FORMAT] to CMDBLK [NDX$H_LAYOUT]
!               Changed CMDBLK [NDX$V_TMS11] and CMDBLK [NDX$V_TEX] to CMDBLK [NDX$H_FORMAT]
!               Changed comparisons of (.CHRSIZ EQLA CHRSZA) to
!               (.CMDBLK [NDX$H_FORMAT] EQL TMS11_A).
!               Definitions were changed in NDXCLI and references to the
!               effected fields were changed in NDXPAG, NDXFMT, INDEX, NDXVMS
!               and NDXCLIDMP.
!
!       005     JPK00010        24-Jan-1983
!               Removed routines GETDAT and UPDDAT from NDXDAT - they
!               performed no useful function. Removed references to these
!               routines from NDXOUT, NDXFMT, and NDXMSG.
!               Removed reference to XPOOL in NDXOUT - not used.
!
!       004     JPK00009        24-Jan-1983
!               Modified to enhance performance. The sort buckets have each
!               been divided into 27 sub-buckets; 1 for each letter and 1
!               for non-alphas. Removed reference to BUCKET from INDEX.
```

```
1 !                     Definition of the structure was added to NDXPOL. References
1 !                     to BUCKET were changed in modules NDXOUT, NDXINI, NDXFMT
1 !                     and NDXDAT.
1 !
1 !         003         JPK00006        24-Sep-1982
1 !                     Modified NDXFMT to ignore REQUIRE file if filename string
1 !                     is null. Added bold guide headings for RUNOFF output.
1 !
1 !         002         JPK00004        24-Sep-1982
1 !                     Modified NDXOUT, NDXMSG, NDXFMT, and NDXDAT for TOPS-20.
1 !                     Strings stored in the index pool use the first fullword
1 !                     for their length. References to these strings were incorrect.
1 !
1 !--
```

```
%SBTTL 'Module level declarations'
!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
    NDXFMT        : NOVALUE,              ! Generate output index
    EXAM_BUCKET   : NOVALUE,              ! Examine bucket
    FORMAT_ENTRY: NOVALUE,                ! Format an index entry
    INS_PAGE_NO : NOVALUE,                ! Insert page number(s) in print line
    BOOK_REF    : NOVALUE,                ! Insert book name in master index
    PAGE_REF    : NOVALUE,                ! Convert page reference to ASCII
    SAMEPG,                               ! Compare page references
    SAME_ATTR,                            ! Page references have same attributes
    INS_LINE    : NOVALUE,                ! Insert line into page
    SAVE_TELLTALE_HEADING : NOVALUE,      ! Save line as telltale heading
    LAST_LINE   : NOVALUE,                ! Process last line in column
    DO_CONTINUE : NOVALUE,                ! Generate continuation heading if necessary
    FILL_LINE   : NOVALUE,                ! Pack the print line
    INDENT_LINE : NOVALUE,                ! Indent a new line
    PADLIN      : NOVALUE;                ! Pad line to column width

!
! INCLUDE FILES:
!
LIBRARY 'NXPORT:XPORT';

SWITCHES LIST (REQUIRE);

REQUIRE 'REQ:NDXPOL';
```

```
!
! Version:      'V04-000'
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!

!++
! FACILITY:
!   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
!
! ABSTRACT:
!   This file contains literals and macros defining the data structures
!   found in the internal index pool
!
! ENVIRONMENT:  Transportable
!
! AUTHOR:       JPK
!
! CREATION DATE: January 1982
!
! MODIFIED BY:
!
!       003     JPK00015        04-Feb-1983
!               Cleaned up module names, modified revision history to
!               conform with established standards. Updated copyright dates.
!
!       002     JPK00009        24-Jan-1983
!               Modified to enhance performance. The sort buckets have each
!               been divided into 27 sub-buckets; 1 for each letter and 1
!               for non-alphas. Removed reference to BUCKET from INDEX.
!               Definition of the structure was added to NDXPOL. References
!               to BUCKET were changed in modules NDXOUT, NDXINI, NDXFMT
!               and NDXDAT.
!
!--
```

```
! Index entry

$FIELD XE_FIELDS =
    SET

    XE$A_PREV           = [$ADDRESS],           ! Link to previous item
    XE$A_NEXT           = [$ADDRESS],           ! Link to next item
    XE$A_SUBX           = [$ADDRESS],           ! Sub index pointer
    XE$A_REF            = [$ADDRESS],           ! Reference pointer
    XE$A_TEXT           = [$ADDRESS],           ! Pointer to text of index item
    XE$A_SORT_AS        = [$ADDRESS],           ! Pointer to SORT_AS string
    XE$H_SUBC           = [$SHORT_INTEGER],     ! Sub index level

    XE$V_FLAGS          = [$SHORT_INTEGER],     ! Entry flags

        $OVERLAY (XE$V_FLAGS)

        XE$V_BARS               = [$BIT],       ! Change bar flag

        $CONTINUE

    XE$A_BOOK_LIST      = [$ADDRESS]            ! Master index book name list

    $ALIGN (FULLWORD)

    TES;

LITERAL
    XE$K_LENGTH = $FIELD_SET_SIZE;

MACRO
    $XE_BLOCK = BLOCK [XE$K_LENGTH] FIELD (XE_FIELDS) %;

! End of Index entry


! Reference entry

$FIELD XX_FIELDS =
    SET

    XX$A_LINK           = [$ADDRESS],           ! Link to additional entries
    XX$A_APPEND         = [$ADDRESS],           ! APPEND text pointer
    XX$H_PAGE           = [$SHORT_INTEGER],     ! Transaction number

    XX$V_FLAGS          = [$SHORT_INTEGER],     ! Display attributes

        $OVERLAY (XX$V_FLAGS)

        XX$V_BOLD       = [$BIT],               ! Bold page reference
        XX$V_UNDERLINE  = [$BIT],               ! Underline page reference
        XX$V_BEGIN      = [$BIT],               ! Begin page range
        XX$V_END        = [$BIT],               ! End page range

        $CONTINUE

    XX$A_BOOK           = [$ADDRESS]            ! Master index book name
```

```
        $ALIGN (FULLWORD)

        TES;

    LITERAL
        XX$K_LENGTH = $FIELD_SET_SIZE;

    MACRO
        $XX_BLOCK = BLOCK [XX$K_LENGTH] FIELD (XX_FIELDS) %;

    ! End of Reference entry


    ! Master index book reference entry

    $FIELD XM_FIELDS =
        SET

        XM$A_LINK           = [$ADDRESS],           ! Link to additional entries
        XM$A_BOOK           = [$ADDRESS]            ! Pointer to book name

        TES;

    LITERAL
        XM$K_LENGTH = $FIELD_SET_SIZE;

    MACRO
        $XM_BLOCK = BLOCK [XM$K_LENGTH] FIELD (XM_FIELDS) %;

    ! End of Master index book reference entry


    ! Current Entry

    $FIELD C_FIELDS =
        SET

        C$A_CURR            = [$ADDRESS],           ! Pointer to current cell
        C$A_PREV            = [$ADDRESS],           ! Pointer to previous cell
        C$A_HEAD            = [$ADDRESS],           ! Pointer to head of chain

        $ALIGN (FULLWORD)

        C$V_FLAGS           = [$INTEGER],           ! Current cell flags

            $OVERLAY (C$V_FLAGS)

            C$V_IDNS        = [$BIT]                ! Identical string flag

            $CONTINUE

        TES;

    LITERAL
        C$K_LENGTH = $FIELD_SET_SIZE;
```

```
MACRO
    $C_BLOCK = BLOCK [C$K_LENGTH] FIELD (C_FIELDS) %;

! End of current entry


!
! Dummy datasets
!
LITERAL
    DS_X_ENTRY  = XE$K_LENGTH,
    DS_XX_ENTRY = XX$K_LENGTH,
    DS_XM_ENTRY = XM$K_LENGTH,
    DS_X_STRING = 0;

!
! Structure definition for bucket array.
!
!       Buckets are arranged so that each row represents the first letter of
!       the string and each column represents the second letter of the string.
!
!       This approach is used only for master indexes as no performance
!       improvement is realised until about 10 input files have been processed.
!
!       Indexes which are not master indexes use only the first element of
!       each row, i.e., [0, 0] ... [26, 0].
!
!       The only exception is for nonalphabetic characters which use only
!       element [0, 0]. Elements [0, 1] ... [0, 26] are not used since mapping
!       all nonalphabetics into one row loses the sort order of the first
!       character in the string. For nonalphabetics to work correctly in a two
!       dimensional bucket scheme, the array would have to be at least 127 x 127
!
!           0   1  .  .  .  26
!        0  **  not used  . . .
!        1  A?  AA          AZ
!        .                   .
!        .                   .
!        .                   .
!       26  Z?  ZA  .  .  .  ZZ
!
STRUCTURE
    $BUCKET_ARRAY [ROW_IDX, COL_IDX; M, N] =
        [M * N * %UPVAL] ($BUCKET_ARRAY + (ROW_IDX * N + COL_IDX) * %UPVAL);

!--     End of NDXPOL.REQ
```

```
;  160          0376  1
;  161          0377  1 REQUIRE 'REQ:NDXCLI';
```

```
!                               IDENT = 0V04-00004
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!
!
!++
! FACILITY:
!   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
!
! ABSTRACT:       INDEX command line definitions
!
! ENVIRONMENT:    Transportable
!
! AUTHOR:         JPK
!
! CREATION DATE: January 1982
!
! MODIFIED BY:
!
!       004     JPK00015        04-Feb-1983
!               Cleaned up module names, modified revision history to
!               conform with established standards. Updated copyright dates.
!
!       003     JPK00011        24-Jan-1983
!               Changed CMDBLK [NDX$G_LEVEL] to CMDBLK [NDX$H_LEVEL]
!               Changed CMDBLK [NDX$H_FORMAT] to CMDBLK [NDX$H_LAYOUT]
!               Changed CMDBLK [NDX$V_TMS11] and CMDBLK [NDX$V_TEX] to CMDBLK [NDX$H_FORMAT]
!               Changed comparisons of (.CHRSIZ EQLA CHRSZA) to
!               (.CMDBLK [NDX$H_FORMAT] EQL TMS11_A).
!               Definitions were changed in NDXCLI and references to the
!               effected fields were changed in NDXPAG, NDXFMT, INDEX, NDXVMS
!               and NDXCLIDMP.
!
!       002     RER00002        20-Jan-1983
!               Modified VMS command line interface module NDXVMS:
!                 - changed /FORMAT qualifier to /LAYOUT.
```

```
1       !               - changed use of /RESERVE and /REQUIRE for DSRPLUS.
1       !               - added code for new DSRPLUS qualifiers /FORMAT and
1       !                 /TELLTALE_HEADINGS.
1       !           Added fields to NDXCLI for new qualifiers: NDX$V_TELLTALE
1       !           and NDX$V_TEX.
1       !           Conditionalized output of NDX$V_PAGE_MERGE in NDXCLIDMP to
1       !           account for different DSR and DSRPLUS default values.
1       !
1       !--
1
```

```
!
!  NDXCMD_FIELDS
!
$FIELD ndxcmd_fields =
    SET

    NDX$V_OPTIONS        = [$INTEGER],          ! Command option indicators:

        $OVERLAY (NDX$V_OPTIONS)

        NDX$V_INPUT_CONCAT      = [$BIT],       ! Input file concatenated to previous
        NDX$V_OUTPUT            = [$BIT],       ! Generate output file
        NDX$V_REQUIRE           = [$BIT],       ! Require file specified
        NDX$V_PAGES             = [$BIT],       ! Include page references in index
        NDX$V_OVERRIDE          = [$BIT],       ! Override master index information
        NDX$V_STANDARD_PAGE     = [$BIT],       ! Generate standard page numbers
        NDX$V_CONTINUATION      = [$BIT],       ! Generate continuation headings
        NDX$V_GUIDE             = [$BIT],       ! Generate guide headings
        NDX$V_WORD_SORT         = [$BIT],       ! Sort entries word by word
        NDX$V_LOG               = [$BIT],       ! Generate /LOG message
        NDX$V_MASTER            = [$BIT],       ! Generate a master index
        NDX$V_PAGE_MERGE        = [$BIT],       ! Merge adjacent page references
        NDX$V_TELLTALE          = [$BIT],       ! Generate telltale headings

        $CONTINUE

    NDX$H_FORMAT        = [$SHORT_INTEGER],     ! Output format: DSR, TMS, TEX
    NDX$H_LAYOUT        = [$SHORT_INTEGER],     ! Output layout type
    NDX$H_NONALPHA      = [$SHORT_INTEGER],     ! Treatment of leading nonalphas during sort
    NDX$H_LEVEL         = [$SHORT_INTEGER],     ! Deepest level to include in index
    NDX$G_COLUMN_WID    = [$INTEGER],           ! Column width
    NDX$G_GUTTER_WID    = [$INTEGER],           ! Gutter width
    NDX$G_LINES_PAGE    = [$INTEGER],           ! Lines per page
    NDX$G_RESERVE_LINES = [$INTEGER],           ! Number of lines to reserve when requiring a file
    NDX$G_SEPARATE_WIDTH= [$INTEGER],           ! Width of reference portion of entry
    NDX$T_MASTER_BOOK   = [$DESCRIPTOR(DYNAMIC)], ! Book name descriptor for Master indexing
    NDX$T_INPUT_FILE    = [$DESCRIPTOR(DYNAMIC)], ! Input file name descriptor
    NDX$T_OUTPUT_FILE   = [$DESCRIPTOR(DYNAMIC)], ! Output file name descriptor
    NDX$T_REQUIRE_FILE  = [$DESCRIPTOR(DYNAMIC)], ! Require file name descriptor
    NDX$T_RELATED_FILE  = [$DESCRIPTOR(DYNAMIC)], ! Related file name descriptor is saved here
                                                  ! by NDXINP for later use by MAKNDX
    NDX$T_COMMAND_LINE  = [$DESCRIPTOR(DYNAMIC)]  ! Copy of entire command line

    TES;
!
!  End of NDXCMD_FIELDS
!

LITERAL
    NDXCMD$K_LENGTH = $FIELD_SET_SIZE;

MACRO
    $NDXCMD = BLOCK [NDXCMD$K_LENGTH] FIELD (NDXCMD_FIELDS) %;

$LITERAL                                ! Output formats (NDX$H_FORMAT)
    DSR             = $DISTINCT,        ! Runoff
    TMS11_A         = $DISTINCT,        ! TMS=A
```

```
;  R0502  1          TMS11_E              = $DISTINCT,      ! TMS=E
;  R0503  1          TEX                  = $DISTINCT;      ! TEX
;  R0504  1
;  R0505  1      $LITERAL                                   ! Output layouts (NDX$H_LAYOUT)
;  R0506  1          TWO_COLUMN           = $DISTINCT,      ! Normal two column format
;  R0507  1          ONE_COLUMN           = $DISTINCT,      ! Normal one column format
;  R0508  1          SEPARATE             = $DISTINCT,      ! Separate reference format
;  R0509  1          GALLEY               = $DISTINCT;      ! TMS11 Galley format
;  R0510  1
;  R0511  1      $LITERAL                                   ! Treatment of leading nonalphas during sort (NDX$H_NONALPHA)
;  R0512  1          BEFORE               = $DISTINCT,      ! Leading nonalphas sort before alphas
;  R0513  1          AFTER                = $DISTINCT,      ! Leading nonalphas sort after alphas
;  R0514  1          IGNORE               = $DISTINCT;      ! Leading nonalphas are ignored
;  R0515  1
;  R0516  1      !
;  R0517  1      !--        End of NDXCLI.REQ
```

```
1
1 REQUIRE 'REQ:NDXLIN';
```

```
!                                   IDENT = 0V04-00002
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!

!++
! FACILITY:
!   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
!
! ABSTRACT:
!   Contains output line type definitions.
!
! ENVIRONMENT:   Transportable
!
! AUTHOR:        JPK
!
! CREATION DATE: January 1982
!
! MODIFIED BY:
!
!       002     JPK00010        04-Feb-1983
!               Cleaned up module names, modified revision history to
!               conform with established standards. Updated copyright dates.
!
!--

LITERAL                                         !Output line types:
    BKT_E = 1,                                  !Bucket end blank line
    FILL = 2,                                   !Fill line
    GUIDE = 3,                                  !Guide heading
    GUIDE_FILL = 4,                             !Blank line after guide heading
    ENTRY_B = 5,                                !Beginning of top level entry
    ENTRY_W = 6,                                !Wrap line of top level entry
    ENTRY_E = 7,                                !Last line of top level entry
    SUB_B = 8,                                  !Beginning of subentry
    SUB_W = 9,                                  !Wrap line of subentry
```

```
        SUB_E = 10,                     !Last line of subentry
        CONT_HEAD = 11;                 !Continuation heading

    !
    !--     End of NDXLIN.REQ
```

```
1
1 REQUIRE 'REQ:PAGEN';
```

```
!
! Version:       'V04-000'
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!
!
!++
! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
!
! ABSTRACT:
!       A page number carries with it not only its current value, but also
!       codes as to how those values are to be displayed when they are finally
!       output. It was decided to do it this way rather than have a separate
!       table so that the program TCX would have less trouble.
!
! ENVIRONMENT:   Transportable BLISS
!
! AUTHOR:        Rich Friday
!
! CREATION DATE: 1978
!
! MODIFIED BY:
!
!       004     KAD00004        Keith Dawson    07-Mar-1983
!               Global edit of all modules. Updated module names, idents,
!               copyright dates. Changed require files to BLISS library.
!
!--

LITERAL
    page_sct_size = 4;

LITERAL                                         !Type of section:
    sct_chapt     = 1,                          ! Chapter section.
    sct_index     = 2,                          ! Index section.
    sct_append    = 3;                          ! Appendix section.
```

```
LITERAL
    sct_low       = 1,                               ! Lowest section type key.
    sct_high      = 3;                               ! Highest section type key.

MACRO
    sct_typ       = 0, 0,         4,         0 %,    ! Section Type (zero if none).
    sct_page_d    = 0, 4,         4,         0 %,    ! Display code for page number.
    sct_sub_page  = 0, %BPVAL/2, %BPVAL/2,   0 %,    ! Subpage, if any (zero if none).
    sct_number    = 1, 0,         %BPVAL,    0 %,    ! Type of section number.
    sct_page      = 2, 0,         %BPVAL,    0 %,    ! Page number.
    sct_subpg_d   = 3, 0,         4,         0 %,    ! Display code for subpages.
    sct_chapt_d   = 3, 4,         4,         0 %,    ! Display code for chapters.
    sct_appen_d   = 3, 8,         4,         0 %,    ! Display code for appendices.
    sct_index_d   = 3, 12,        4,         0 %;    ! Display code if indexes.

MACRO
    sct_run_page  = 3, %BPVAL/2, %BPVAL/2, 0 %; ! Running page number.

MACRO
    page_definition = BLOCK [page_sct_size] %;

!                        End of PAGEN.REQ
```

```
        1
L       1 %IF %BLISS (BLISS32)
        1 %THEN
        1
        1 REQUIRE 'REQ:NDXVMSREQ';
```

```
!
! Version:      'V04-000'
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!
!
!++
! FACILITY:
!   DSR (Digital Standard RUNOFF) /DSRPLUS DSRINDEX/INDEX Utility
!
! ABSTRACT:
!       This file contains external references to the error message numbers
!       for DSRINDEX/INDEX.
!
!       New messages must be defined in NDXVMSMSG.MSG and referenced here:
!       both in the MACRO section (for DSRINDEX) and the EXTERNAL LITERAL
!       section (for INDEX)
!
! ENVIRONMENT:   VAX/VMS User Mode
!
! AUTHOR:        JPK
!
! CREATION DATE: 01-Feb-1983
!
! MODIFIED BY:
!
!       004     JPK00022        30-Mar-1983
!               Modified NDXVMS, NDXFMT, NDXPAG, NDXVMSMSG and NDXVMSREQ
!               to generate TEX output. Added module NDXTEX.
!
!       003     JPK00021        28-Mar-1983
!               Modified NDXT20 to include E2.0 functionality.
!               Modified NDXCLIDMP, NDXFMT, NDXPAG, NDXVRS to require RNODEF
!               for BLISS36 and to remove any conditional require based on
!               DSRPLUS_DEF.
!
```

```
!         002     JPK00010        04-Feb-1983
!                 Cleaned up module names, modified revision history to
!                 conform with established standards. Updated copyright dates.
!
!--

REQUIRE 'REQ:RNODEF';
```

```
!
! Version:      'V04-000'
!
!****************************************************************************
!*                                                                          *
!*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
!*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
!*  ALL RIGHTS RESERVED.                                                    *
!*                                                                          *
!*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
!*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
!*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
!*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
!*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
!*  TRANSFERRED.                                                            *
!*                                                                          *
!*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
!*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
!*  CORPORATION.                                                            *
!*                                                                          *
!*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
!*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
!*                                                                          *
!*                                                                          *
!****************************************************************************
!
!

!++
! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
!
! ABSTRACT:
!       Converts BLISS/VARIANT values into useful names.
!
! ENVIRONMENT:   Transportable BLISS
!
! AUTHOR:        Rich Friday
!
! CREATION DATE: 1978
!
! MODIFIED BY:
!
!       016     KAD00016        Ray Marshall    19-Mar-1984
!               Added GERMAN, FRENCH, & ITALIAN.
!
!       015     KAD00015        Keith Dawson    18-Apr-1983
!               Made the LN01 conditional the default for vanilla DSR --
!                 its value is 0 (no variant supplied).
!
!       014     KAD00014        Keith Dawson    22-Mar-1983
!               Asserted the LN01 conditional when DSRPLUS is asserted.
!
!       013     KAD00013        Keith Dawson    20-Mar-1983
!               Removed all references to .BIX and .BTC files.
!
!       012     KAD00012        Keith Dawson    07-Mar-1983
!               Global edit of all modules. Updated module names, idents,
!               copyright dates. Changed require files to BLISS library.
```

```
!
!--

!++
        D E F I N I T I O N   O F  /VARIANT   B I T S
!
        The bit assignments are as follows:
!
        Bit  Weight   Meaning
!----------------------------------------------------------------------
        --      0     If no /VARIANT is supplied (as for vanilla DSR),
                      compile with LN01 support. LN01 support is also
                      implied by the DSRPLUS variant.
!
         0      1     CLEAR =   Unassigned
                      SET   =   Unassigned
!
         1      2     CLEAR =   Normal compile
                      SET   =   Compile for DSRPLUS
!
        4-6    16     CLEAR =   English (American) version
                      SET   =   16 = German (Austrian)
                                32 = French
                                48 = Italian
!--

!---------------------------------------------------------------------
!   This variable (LN01) controls whether or not to compile an LN01-flavored
!   DSR. It is asserted by default, and also whenever DSRPLUS is asserted.
!
!   Modules utilizing LN01 are:
!
!       DOOPTS  NOUT

COMPILETIME
    ln01 =
        ( (%VARIANT EQL 0) OR  %VARIANT/2 )
    ;

!---------------------------------------------------------------------
!   This variable (DSRPLUS) controls compilation for the DSRPLUS program.
!
!   All modules utilize DSRPLUS.

COMPILETIME
    dsrplus =
        ( %VARIANT/2 )
    ;

!---------------------------------------------------------------------
!   This variable (FLIP) controls compilation of FLIP features of DSRPLUS.
!   It assures that FLIP features are compiled only on VMS systems.
!
!   Modules utilizing FLIP are many and various.

COMPILETIME
    flip =
```

```
;  2          ( %VARIANT/2 AND %BLISS(BLISS32) )
;  1      ;
;  1
;  1  !--------------------------------------------------------------------------------
;  1  !        4-6    16      CLEAR =   English (American) version
;  1  !                       SET   =   16 = German (Austrian)
;  1  !                                 32 = French
;  1  !                                 48 = Italian
;  1  COMPILETIME
;  1      German  = ( %VARIANT/16 AND NOT %VARIANT/32 AND NOT %VARIANT/64 ) ;
;  1  COMPILETIME
;  1      French  = ( NOT %VARIANT/16 AND %VARIANT/32 AND NOT %VARIANT/64 ) ;
;  1  COMPILETIME
;  1      Italian = ( %VARIANT/16 AND %VARIANT/32 AND NOT %VARIANT/64 ) ;
;  1  !--------------------------------------------------------------------------------
;  1  !                End of RNODEF.REQ
```

```
%IF NOT DSRPLUS
%THEN

MACRO
     INDEX$_BADLOGIC  = DSRINDEX$_BADLOGIC  %,
     INDEX$_BADVALUE  = DSRINDEX$_BADVALUE  %,
     INDEX$_INSVIRMEM = DSRINDEX$_INSVIRMEM %,
     INDEX$_LINELENG  = DSRINDEX$_LINELENG  %,
     INDEX$_NOREF     = DSRINDEX$_NOREF     %,
     INDEX$_OPENIN    = DSRINDEX$_OPENIN    %,
     INDEX$_OPENOUT   = DSRINDEX$_OPENOUT   %,
     INDEX$_TOOMANY   = DSRINDEX$_TOOMANY   %,
     INDEX$_VALERR    = DSRINDEX$_VALERR    %,
     INDEX$_CANTBAL   = DSRINDEX$_CANTBAL   %,
     INDEX$_CLOSEQUOT = DSRINDEX$_CLOSEQUOT %,
     INDEX$_CONFQUAL  = DSRINDEX$_CONFQUAL  %,
     INDEX$_CTRLCHAR  = DSRINDEX$_CTRLCHAR  %,
     INDEX$_DOESNTFIT = DSRINDEX$_DOESNTFIT %,
     INDEX$_DUPBEGIN  = DSRINDEX$_DUPBEGIN  %,
     INDEX$_EMPTYIN   = DSRINDEX$_EMPTYIN   %,
     INDEX$_IGNORED   = DSRINDEX$_IGNORED   %,
     INDEX$_INVINPUT  = DSRINDEX$_INVINPUT  %,
     INDEX$_INVRECORD = DSRINDEX$_INVRECORD %,
     INDEX$_LASTCONT  = DSRINDEX$_LASTCONT  %,
     INDEX$_NOBEGIN   = DSRINDEX$_NOBEGIN   %,
     INDEX$_NOEND     = DSRINDEX$_NOEND     %,
     INDEX$_NOINDEX   = DSRINDEX$_NOINDEX   %,
     INDEX$_NOLIST    = DSRINDEX$_NOLIST    %,
     INDEX$_OVERSTRK  = DSRINDEX$_OVERSTRK  %,
     INDEX$_SKIPPED   = DSRINDEX$_SKIPPED   %,
     INDEX$_SYNTAX    = DSRINDEX$_SYNTAX    %,
     INDEX$_TEXFILE   = DSRINDEX$_TEXFILE   %,
     INDEX$_TOODEEP   = DSRINDEX$_TOODEEP   %,
     INDEX$_TOOFEW    = DSRINDEX$_TOOFEW    %,
     INDEX$_TRUNCATED = DSRINDEX$_TRUNCATED %,
     INDEX$_COMPLETE  = DSRINDEX$_COMPLETE  %,
     INDEX$_CREATED   = DSRINDEX$_CREATED   %,
     INDEX$_IDENT     = DSRINDEX$_IDENT     %,
     INDEX$_PROCFILE  = DSRINDEX$_PROCFILE  %,
     INDEX$_TEXT      = DSRINDEX$_TEXT      %,
     INDEX$_TEXTD     = DSRINDEX$_TEXTD     %,
     INDEX$_TMS11     = DSRINDEX$_TMS11     %;

%FI

EXTERNAL LITERAL
     INDEX$_BADLOGIC,      ! <internal logic error detected>
     INDEX$_BADVALUE,      ! <'!AS' is an invalid keyword value>
     INDEX$_INSVIRMEM,     ! <insufficient virtual memory>
     INDEX$_LINELENG,      ! <maximum line length is 120>
     INDEX$_NOREF,         ! <page reference not found>
     INDEX$_OPENIN,        ! <error opening '!AS' for input>
     INDEX$_OPENOUT,       ! <error opening '!AS' for output>
     INDEX$_TOOMANY,       ! <too many values supplied>
     INDEX$_VALERR,        ! <specified value is out of legal range>
     INDEX$_CANTBAL,       ! <can't balance last page>
```

```
1          INDEX$_CLOSEQUOT,      ! <missing close quote>
1          INDEX$_CONFQUAL,       ! <conflicting qualifiers>
1          INDEX$_CTRLCHAR,       ! <the following line contains control characters - ignored>
1          INDEX$_DOESNTFIT,      ! <'!AD' will not fit at the current indentation level>
1          INDEX$_DUPBEGIN,       ! <duplicate .XPLUS (BEGIN) - inserted as .XPLUS ()>
1          INDEX$_EMPTYIN,        ! <empty input file '!AS'>
1          INDEX$_IGNORED,        ! <'!AS' ignored>
1          INDEX$_INVINPUT,       ! <invalid input file format in file '!AS'>
1          INDEX$_INVRECORD,      ! <invalid record type in file '!AS'>
1          INDEX$_LASTCONT,       ! <can't generate continuation heading on last page>
1          INDEX$_NOBEGIN,        ! <.XPLUS (END) with no .XPLUS (BEGIN) - inserted as .XPLUS ()>
1          INDEX$_NOEND,          ! <.XPLUS (BEGIN) has no corresponding .XPLUS (END)>
1          INDEX$_NOINDEX,        ! <no index information in file '!AS'>
1          INDEX$_NOLIST,         ! <parameter list not allowed>
1          INDEX$_OVERSTRK,       ! <the following line contains an overstrike sequence>
1          INDEX$_SKIPPED,        ! <!UL reference!%S inside page range - ignored>
1          INDEX$_SYNTAX,         ! <error parsing '!AS'>
1          INDEX$_TEXFILE,        ! <error processing line !UL of TEX character file '!AS'>
1          INDEX$_TOODEEP,        ! <maximum subindex depth exceeded>
1          INDEX$_TOOFEW,         ! <not enough values supplied>
1          INDEX$_TRUNCATED,      ! <string too long - truncated>
1          INDEX$_COMPLETE,       ! <processing complete '!AS'>
1          INDEX$_CREATED,        ! <'!AS' created>
1          INDEX$_IDENT,          ! <INDEX version !AD>
1          INDEX$_PROCFILE,       ! <processing file '!AS'>
1          INDEX$_TEXT,           ! <!AS>
1          INDEX$_TEXTD,          ! <entry text: '!AD'>
1          INDEX$_TMS11;          ! <output file full - continuing with file '!AS'>
1
```

```
%FI

%IF %BLISS (BLISS36)
%THEN

REQUIRE 'REQ:RNODEF';

%FI

SWITCHES LIST (NOREQUIRE);

!
! MACROS:
!
MACRO
    !
    ! Don't quote the character, and it doesn't count visually.
    !
    NQCHAR_NC (X) =
        BEGIN
        CH$WCHAR_A(X, WORD_POINTER);
        INT_WORD_LENGTH = .INT_WORD_LENGTH + 1;
        END     %,

    !
    ! Don't quote the character, and it does count visually.
    !
    NQCHAR_C (X) =
        BEGIN
        NQCHAR_NC (X);
        EXT_WORD_LENGTH = .EXT_WORD_LENGTH + 1;
        TMS_WORD_LENGTH = .TMS_WORD_LENGTH + .CHRSIZ [X];
        END     %,

    !
    ! Quote the character; it doesn't count visually.
    !
    QCHAR_NC (X) =
        BEGIN
        NQCHAR_NC (%C'_');
        NQCHAR_NC (X);
        END     %,

    !
    ! Quote the character; it does count visually.
    !
    QCHAR_C (X) =
        BEGIN
        NQCHAR_NC (%C'_');
        NQCHAR_C (X);
        END     %,

    !
    ! Special macro to keep track of the backspace control character.
    !
    BACKSPACE (X) =
```

```
M 1            BEGIN
M 1            QCHAR_NC (X);
M 1            EXT_WORD_LENGTH = .EXT_WORD_LENGTH - 1;
  1            END      %,
  1
  1        !
  1        ! Clear the text lines being built up.
  1        !
M 1        CLR_RNO_LINE (_) =
M 1            BEGIN
M 1            LINE_POINTER = CH$PTR (RNO_LINE);
M 1            INT_LINE_LENGTH = 0;
M 1            EXT_LINE_LENGTH = 0;
M 1            TMS_LINE_LENGTH = 0;
  1            END      %,
  1
  1        !
  1        ! Clear the word being built
  1        !
M 1        CLR_RNO_WORD (_) =
M 1            BEGIN
M 1            WORD_POINTER = CH$PTR (RNO_WORD);
M 1            INT_WORD_LENGTH = 0;
M 1            EXT_WORD_LENGTH = 0;
M 1            TMS_WORD_LENGTH = 0;
M 1            HYPHENATE = FALSE;
  1            END      %,
  1
  1        !
  1        ! Write a line to the output file
  1        !
M 1        PUT_LINE (S) =
  1            $XPO_PUT (IOB = OUTIOB, STRING = S) %;
  1
  1    !
  1    ! EQUATED SYMBOLS:
  1    !
  1    LITERAL
  1        TRUE = 1,
  1        FALSE = 0;
  1
  1    LITERAL
  1        TMS_SAFETY_MARGIN = 4;                       ! Subtracted from # chars / line to avoid wrapping
  1
  1    !
  1    ! OWN STORAGE:
  1    !
  1    OWN
  1        BLANKS : INITIAL (CH$PTR (UPLIT ('    '))); ! Pointer to blanks
  1
  1    OWN
  1        LINE_POINTER,
  1        RNO_LINE : VECTOR [CH$ALLOCATION (1200)],
  1        INT_LINE_LENGTH,
  1        EXT_LINE_LENGTH,
  1        TMS_LINE_LENGTH,
  1        LINE_TYPE,
```

```
	PAGE_WIDTH;

OWN
	DOING_LAST;				! TRUE if processing last line

OWN
	SAVE_ENTRY : INITIAL (FALSE);		! TRUE if saw guide head at top of left page

!
! EXTERNAL REFERENCES:
!

EXTERNAL LITERAL
	TAB	: UNSIGNED (8),
	MAXLST,
	TMSSTD,					! Average TMS character size
	MSPACE,					! TMS 'em' space size
	RINTES	: UNSIGNED (8);

EXTERNAL
	CMDBLK : $NDXCMD,			! Command line information block
	NDXVRL,					! Length of version string
	NDXVRP,					! CH$PTR to version string
	OUTIOB : $XPO_IOB (),			! Output file IOB
	LSTPTR : REF $XE_BLOCK,
	INDLVL,					! Index level
	LSTSTK : VECTOR,			! Temporary stack for storing back links
	BUCKET : $BUCKET_ARRAY [27, 27],	! Hashing buckets
	CHRSIZ : REF VECTOR,			! TMS11 character size vector
	ALLOWD,					! Usuable lines per page
	LCOUNT,					! Number of lines in left column
	RCOUNT,					! Number of lines in right column
	TCOUNT,					! Number of lines in temp column
!
! NOTE: The vectors and blockvectors below have two extra entries allocated
!	to avoid needing to subtract 1 all the time (for entry zero),
!	and so that there will always be an available line at the end of the column
!
	LTYPE  : VECTOR,			! Left column line types
	LLINES : BLOCKVECTOR [, STR$K_D_BLN],	! Left column string descriptors
	RTYPE  : VECTOR,			! Right column line types
	RLINES : BLOCKVECTOR [, STR$K_D_BLN],	! Right column string descriptors
	TTYPE  : VECTOR,			! Temp column line types
	TLINES : BLOCKVECTOR [, STR$K_D_BLN];	! Right column for last page

EXTERNAL ROUTINE

%IF %BLISS (BLISS32)
%THEN

	OPEN_ERROR,				! File open error processor

%FI

	ENTMSG	: NOVALUE,
	GENTRY,
	LASTPG	: NOVALUE,
```

```
1      PACBAS,
1      PACPAG,
1      PAGEQL,
1      PAGMRG,
1      PUTPAG    : NOVALUE,
1      TMSINI    : NOVALUE,
1      TMSPUT    : NOVALUE,
1      XTNPAG;
```

```
%SBTTL 'NDXFMT -- Build output index from internal index'
GLOBAL ROUTINE NDXFMT : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This routine is the top-level control routine of the formatting
!       procedure.
!
!       If output is being generated for RUNOFF, a "standard" RUNOFF
!       environment is set up.
!
!       If a require file was specified, it is copied to the output file.
!
!       Each bucket of the binary index is then examined to form the
!       output index. The nonalpha bucket is examined first if either
!       /SORT=NONALPHA=BEFORE or /SORT=NONALPHA=IGNORE were specified
!       on the command line. Otherwise, the nonalpha bucket is examined last.
!
! FORMAL PARAMETERS:
!
!       None
!
! IMPLICIT INPUTS:
!
!       CMDBLK          - Command line information block
!
! IMPLICIT OUTPUTS:
!
!       INDLVL          - subindex level
!       PAGE_WIDTH      - width of output page
!       ALLOWD          - number of lines allowed on first page of index
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
! SIDE EFFECTS:
!
!       None
!
!--

    BEGIN
    LOCAL
        DSC_PTR : REF $STR_DESCRIPTOR (),
        CMD_LEN,
        CMD_PTR,
        PTR;

    !
    ! Initialization
    !
    LCOUNT = 0;
    RCOUNT = 0;
    DOING_LAST = FALSE;
```

```
    SELECTONE .CMDBLK [NDX$H_LAYOUT] OF
    SET

    [SEPARATE]:
        PAGE_WIDTH = .CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID] + .CMDBLK [NDX$G_SEPARATE_WIDTH

    [TWO_COLUMN]:
        PAGE_WIDTH = (2 * .CMDBLK [NDX$G_COLUMN_WID]) + .CMDBLK [NDX$G_GUTTER_WID];

    [OTHERWISE]:
        PAGE_WIDTH = .CMDBLK [NDX$G_COLUMN_WID];

    TES;

    !
    ! Reformat command line before outputting.
    !    Change '<' to '['
    !    Change '>' to ']'
    !    Change ';' to '.'
    !
    DSC_PTR = CMDBLK [NDX$T_COMMAND_LINE];
    CMD_LEN = .DSC_PTR [STR$H_LENGTH];
    CMD_PTR = .DSC_PTR [STR$A_POINTER];

    WHILE NOT CH$FAIL (PTR = CH$FIND_CH (.CMD_LEN, .CMD_PTR, %C'<')) DO
        CH$WCHAR (%C'[', .PTR);

    WHILE NOT CH$FAIL (PTR = CH$FIND_CH (.CMD_LEN, .CMD_PTR, %C'>')) DO
        CH$WCHAR (%C']', .PTR);

    WHILE NOT CH$FAIL (PTR = CH$FIND_CH (.CMD_LEN, .CMD_PTR, %C';')) DO
        CH$WCHAR (%C'.', .PTR);

    !
    ! Compute number of lines allowed on first page
    !
    IF .CMDBLK [NDX$G_RESERVE_LINES] NEQ 0
    THEN
        !
        ! Number of lines on first page is lines per page
        ! minus number of reserved lines.
        !
        ALLOWD = .CMDBLK [NDX$G_LINES_PAGE] - .CMDBLK [NDX$G_RESERVE_LINES]
    ELSE
        BEGIN

        IF .CMDBLK [NDX$H_FORMAT] EQL DSR
        THEN
            ALLOWD = .CMDBLK [NDX$G_LINES_PAGE] - 4      ! Default for RUNOFF
        ELSE
            ALLOWD = .CMDBLK [NDX$G_LINES_PAGE] - 22;    ! Default for TMS and TEX

        END;

    !
    ! Write output dependant prologue
```

```
    2       !
    2       SELECTONE .CMDBLK [NDX$H_FORMAT] OF
    2           SET
    2
    2           [DSR]:
    3               BEGIN
    3               !
    3               ! RUNOFF output
    3               !
    3               ! Generate a "standard" RUNOFF environment.
    3               !
L   3   %IF DSRPLUS
U   3   %THEN
U   3
U   3               PUT_LINE ($STR_CONCAT ('.! INDEX version ', (.NDXVRL, .NDXVRP)));
U   3
    3   %ELSE
    3
    3               PUT_LINE ($STR_CONCAT ('.! DSRINDEX version ', (.NDXVRL, .NDXVRP)));
    3
    3   %FI
    3
    3               PUT_LINE ($STR_CONCAT ('.! ', .DSC_PTR));
    3               PUT_LINE ('.SAVE');
    3               PUT_LINE ('.NO FLAGS ALL');
    3               PUT_LINE ('.NO FLAGS BREAK          .NO FLAGS CAPITALIZE    .NO FLAGS ENDFOOTNOTE');
    3               PUT_LINE ('.NO FLAGS HYPHENATE      .NO FLAGS INDEX         .NO FLAGS LOWERCASE');
    3               PUT_LINE ('.NO FLAGS PERIOD .NO FLAGS SPACE         .NO FLAGS SUBSTITUTE');
    3               PUT_LINE ('.NO FLAGS UPPERCASE');
    3               PUT_LINE ('.FLAGS ACCEPT            .FLAGS BOLD *           .FLAGS COMMENT !');
    3               PUT_LINE ('.FLAGS OVERSTRIKE %      .FLAGS UNDERLINE &');
    3               PUT_LINE ('.FLAGS ALL');
    3               PUT_LINE ('.NO FILL         .NO JUSTIFY');
    3
P   3               PUT_LINE (
P   3                   $STR_CONCAT (
P   3                       '.LEFT MARGIN 0             .RIGHT MARGIN ',
P   3                       $STR_ASCII (.PAGE_WIDTH),
P   3                       '       .PAGE SIZE , ',
P   3                       $STR_ASCII (.PAGE_WIDTH)
P   3                   )
    3               );
    3
    3               PUT_LINE ('.NUMBER INDEX            .NUMBER PAGE 1          .FIRST TITLE');
    3
    3               IF .CMDBLK [NDX$V_TELLTALE]
    3               THEN
    4                   BEGIN
    4                   !
    4                   ! Generate prologue to set up for /TELLTALE
    4                   !
    4                   PUT_LINE ('.LAYOUT 2,2');
    4                   PUT_LINE ('.TITLE');
    4                   PUT_LINE ('.SUBTITLE');
    3                   END;
    3
    3               IF .CMDBLK [NDX$H_LAYOUT] EQL GALLEY THEN PUT_LINE ('.NO PAGING');
```

```
                END;

            [TMS11_A, TMS11_E]:
                !
                ! Generate TMS11 heading
                !
                TMSINI ();

            [TEX]:
                BEGIN
                !
                ! Generate a heading for TEX
                !
                PUT_LINE ($STR_CONCAT ('% INDEX version ', (.NDXVRL, .NDXVRP)));
                PUT_LINE ($STR_CONCAT ('% ', .DSC_PTR));
                PUT_LINE ('\twocolindex');
                END;

            TES;

        DSC_PTR = CMDBLK [NDX$T_REQUIRE_FILE];

        IF .CMDBLK [NDX$V_REQUIRE] AND (.DSC_PTR [STR$H_LENGTH] GTR 0)
        THEN
            BEGIN
            !
            ! User specified a require file. Insert it here.
            !
            LOCAL
                REQUIRE_IOB : $XPO_IOB ();

            $XPO_IOB_INIT (IOB = REQUIRE_IOB);
P           $XPO_OPEN (IOB = REQUIRE_IOB, FILE_SPEC = .DSC_PTR
P               %IF %BLISS (BLISS32) %THEN , FAILURE = OPEN_ERROR %FI
                );

            WHILE $XPO_GET (IOB = REQUIRE_IOB) EQL XPO$_NORMAL DO
                BEGIN

                SELECTONE .CMDBLK [NDX$H_FORMAT] OF
                    SET

                    [TMS11_A, TMS11_E]:
                        TMSPUT (.REQUIRE_IOB [IOB$H_STRING], .REQUIRE_IOB [IOB$A_STRING], OUTIOB, FALSE);

                    [DSR, TEX]:
                        PUT_LINE (REQUIRE_IOB [IOB$T_STRING]);

                    TES;

                END;

            $XPO_CLOSE (IOB = REQUIRE_IOB);
            END
        ELSE
            IF .CMDBLK [NDX$H_FORMAT] EQL DSR
            THEN
```

```
3                   BEGIN
3                   !
3                   ! User didn't specify a require file and output is for RUNOFF.
3                   ! Therefore, insert default commands.
3                   !
3                   PUT_LINE ('.CENTER;INDEX');
3                   PUT_LINE ('.BLANK3');
2                   END;
2
2           IF .CMDBLK [NDX$H_NONALPHA] NEQ AFTER
2           THEN
2               !
2               ! Examine nonalpha bucket first if user specified
2               !
2               ! /SORT=NONALPHA=BEFORE or /SORT=NONALPHA=IGNORE
2               !
2               EXAM_BUCKET (0);
2
2           !
2           ! Examine all alpha buckets
2           !
2           INCR I FROM 1 TO 26 DO
2               EXAM_BUCKET (.I);
2
2           IF .CMDBLK [NDX$H_NONALPHA] EQL AFTER
2           THEN
2               !
2               ! Examine nonalpha characters last
2               !
2               EXAM_BUCKET (0);
2
2           !
2           ! Write last page
2           !
2           LASTPG ();
2
1           END;                            !End of NDXFMT
```

```
                                                                  .TITLE  NDXFMT NDXFMT -- Binary index formatter
                                                                  .IDENT  \V04-000\

                                                                  .PSECT  $PLIT$,NOWRT,NOEXE,2

                                       20 20 20 20 00000 P.AAA:   .ASCII  \    \
72 65 76 20 58 45 44 4E 49 52 53 44 20 21 2E 00004 P.AAC:   .ASCII  \.! DSRINDEX version \
                                    20 6E 6F 69 73 00013
                                          20 21 2E 00018 P.AAF:   .ASCII  \.! \
                                    45 56 41 53 2E 0001B P.AAH:   .ASCII  \.SAVE\
      4C 4C 41 20 53 47 41 4C 46 20 4F 4E 2E 00020 P.AAI:   .ASCII  \.NO FLAGS ALL\
4B 41 45 52 42 20 53 47 41 4C 46 20 4F 4E 2E 0002D P.AAJ:   .ASCII  \.NO FLAGS BREAK\<9><9>\.NO FLAGS CA\
   41 43 20 53 47 41 4C 46 20 4F 4E 2E 09 09 0003C
4C 46 20 4F 4E 2E 09 45 5A 49 4C 41 54 49 50 0004A          .ASCII  \PITALIZE\<9>\.NO FLAGS ENDFOOTNOTE\
45 54 4F 4E 54 4F 4F 46 44 4E 45 20 53 47 41 00059
45 48 50 59 48 20 53 47 41 4C 46 20 4F 4E 2E 00068 P.AAK:   .ASCII  \.NO FLAGS HYPHENATE\<9>\.NO FLAGS INDE\
20 53 47 41 4C 46 20 4F 4E 2E 09 45 54 41 4E 00077
                                    45 44 4E 49 00086
```

```
4F 4C 20 53 47 41 4C 46 20 4F 4E 2E 09 09 58  0008A          .ASCII  \X\<9><9>\.NO FLAGS LOWERCASE\                  ;
                        45 53 41 43 52 45 57  00099                                                                  ;
4F 49 52 45 50 20 53 47 41 4C 46 20 4F 4E 2E  000A0 P.AAL:   .ASCII  \.NO FLAGS PERIOD\<9>\.NO FLAGS SPACE\<9>       ;
41 50 53 20 53 47 41 4C 46 20 4F 4E 2E 09 44  000AF                                                                  ;
                                  2E 09 45 43  000BE                                                                  ;
53 42 55 53 20 53 47 41 4C 46 20 4F 4E 2E 09  000C1          .ASCII  <9>\.NO FLAGS SUBSTITUTE\                       ;
                           45 54 55 54 49 54  000D0                                                                  ;
52 45 50 50 55 20 53 47 41 4C 46 20 4F 4E 2E  000D6 P.AAM:   .ASCII  \.NO FLAGS UPPERCASE\                           ;
                              45 53 41 43  000E5                                                                      ;
5F 20 54 50 45 43 43 41 20 53 47 41 4C 46 2E  000E9 P.AAN:   .ASCII  \.FLAGS ACCEPT _\<9><9>\.FLAGS BOLD \           ;
   20 44 4C 4F 42 20 53 47 41 4C 46 2E 09 09  000F8                                                                  ;
45 4D 4D 4F 43 20 53 47 41 4C 46 2E 09 09 2A  00106          .ASCII  \*\<9><9>\.FLAGS COMMENT !\                     ;
                                 21 20 54 4E  00115                                                                  ;
49 52 54 53 52 45 56 4F 20 53 47 41 4C 46 2E  00119 P.AAO:   .ASCII  \.FLAGS OVERSTRIKE %\<9>\.FLAGS UNDERLI\        ;
44 4E 55 20 53 47 41 4C 46 2E 09 25 20 45 4B  00128                                                                  ;
                              49 4C 52 45  00137                                                                      ;
                              26 20 45 4E  0013B          .ASCII  \NE &\                                              ;
         4C 4C 41 20 53 47 41 4C 46 2E  0013F P.AAP:   .ASCII  \.FLAGS ALL\                                           ;
4A 20 4F 4E 2E 09 09 4C 4C 49 46 20 4F 4E 2E  00149 P.AAQ:   .ASCII  \.NO FILL\<9><9>\.NO JUSTIFY\                   ;
                  59 46 49 54 53 55  00158                                                                            ;
09 30 20 4E 49 47 52 41 4D 20 54 46 45 4C 2E  0015E P.AAT:   .ASCII  \.LEFT MARGIN 0\<9><9>\.RIGHT MARGIN\           ;
   4E 49 47 52 41 4D 20 54 48 47 49 52 2E 09  0016D                                                                  ;
                                          20  0017B          .ASCII  \ \                                              ;
   20 2C 20 45 5A 49 53 20 45 47 41 50 2E 09  0017C P.AAU:   .ASCII  <9>\.PAGE SIZE , \                              ;
09 09 58 45 44 4E 49 20 52 45 42 4D 55 4E 2E  0018A P.AAX:   .ASCII  \.NUMBER INDEX\<9><9>\.NUMBER PAGE 1\           ;
   31 20 45 47 41 50 20 52 45 42 4D 55 4E 2E  00199                                                                  ;
   45 4C 54 49 54 20 54 53 52 49 46 2E 09 09  001A7          .ASCII  <9><9>\.FIRST TITLE\                            ;
         32 2C 32 20 54 55 4F 59 41 4C 2E  001B5 P.AAY:   .ASCII  \.LAYOUT 2,2\                                       ;
                  45 4C 54 49 54 2E  001C0 P.AAZ:   .ASCII  \.TITLE\                                                  ;
      45 4C 54 49 54 42 55 53 2E  001C6 P.ABA:   .ASCII  \.SUBTITLE\                                                  ;
   47 4E 49 47 41 50 20 4F 4E 2E  001CF P.ABB:   .ASCII  \.NO PAGING\                                                 ;
6E 6F 69 73 72 65 76 20 58 45 44 4E 49 20 25  001D9 P.ABD:   .ASCII  \% INDEX version \                              ;
                                          20  001E8                                                                  ;
                                       20 25  001E9 P.ABG:   .ASCII  \% \                                            ;
      78 65 64 6E 69 6C 6F 63 6F 77 74 5C  001EB P.ABI:   .ASCII  <92>\twocolindex\                                   ;
   58 45 44 4E 49 3B 52 45 54 4E 45 43 2E  001F7 P.ABJ:   .ASCII  \.CENTER;INDEX\                                     ;
         33 4B 4E 41 4C 42 2E  00204 P.ABK:   .ASCII  \.BLANK3\                                                       ;

                                                     .PSECT  $OWN$,NOEXE,2

                               00000000'  00000 BLANKS: .ADDRESS P.AAA                                                ;
                                          00004 LINE_POINTER:
                                                         .BLKB   4
                                          00008 RNO_LINE:
                                                         .BLKB   1200
                                          004B8 INT_LINE_LENGTH:
                                                         .BLKB   4
                                          004BC EXT_LINE_LENGTH:
                                                         .BLKB   4
                                          004C0 TMS_LINE_LENGTH:
                                                         .BLKB   4
                                          004C4 LINE_TYPE:
                                                         .BLKB   4
                                          004C8 PAGE_WIDTH:
                                                         .BLKB   4
                                          004CC DOING_LAST:
                                                         .BLKB   4
```

```
        00000000  004D0 SAVE_ENTRY:
                                  .LONG    0                                    ;
            0014  004D4 $STR$STRING0:
                                  .WORD    20                                   ;
          01  0E  004D6           .BYTE    14, 1                                ;
       00000000'  004D8           .ADDRESS P.AAC                                ;
            0003  004DC $STR$STRING0:
                                  .WORD    3                                    ;
          01  0E  004DE           .BYTE    14, 1                                ;
       00000000'  004E0           .ADDRESS P.AAF                                ;
            0005  004E4 $IOB$OUTPUT:
                                  .WORD    5                                    ;
          01  0E  004E6           .BYTE    14, 1                                ;
       00000000'  004E8           .ADDRESS P.AAH                                ;
            000D  004EC $IOB$OUTPUT:
                                  .WORD    13                                   ;
          01  0E  004EE           .BYTE    14, 1                                ;
       00000000'  004F0           .ADDRESS P.AAI                                ;
            003B  004F4 $IOB$OUTPUT:
                                  .WORD    59                                   ;
          01  0E  004F6           .BYTE    14, 1                                ;
       00000000'  004F8           .ADDRESS P.AAJ                                ;
            0038  004FC $IOB$OUTPUT:
                                  .WORD    56                                   ;
          01  0E  004FE           .BYTE    14, 1                                ;
       00000000'  00500           .ADDRESS P.AAK                                ;
            0036  00504 $IOB$OUTPUT:
                                  .WORD    54                                   ;
          01  0E  00506           .BYTE    14, 1                                ;
       00000000'  00508           .ADDRESS P.AAL                                ;
            0013  0050C $IOB$OUTPUT:
                                  .WORD    19                                   ;
          01  0E  0050E           .BYTE    14, 1                                ;
       00000000'  00510           .ADDRESS P.AAM                                ;
            0030  00514 $IOB$OUTPUT:
                                  .WORD    48                                   ;
          01  0E  00516           .BYTE    14, 1                                ;
       00000000'  00518           .ADDRESS P.AAN                                ;
            0026  0051C $IOB$OUTPUT:
                                  .WORD    38                                   ;
          01  0E  0051E           .BYTE    14, 1                                ;
       00000000'  00520           .ADDRESS P.AAO                                ;
            000A  00524 $IOB$OUTPUT:
                                  .WORD    10                                   ;
          01  0E  00526           .BYTE    14, 1                                ;
       00000000'  00528           .ADDRESS P.AAP                                ;
            0015  0052C $IOB$OUTPUT:
                                  .WORD    21                                   ;
          01  0E  0052E           .BYTE    14, 1                                ;
       00000000'  00530           .ADDRESS P.AAQ                                ;
            001E  00534 $STR$STRING0:
                                  .WORD    30                                   ;
          01  0E  00536           .BYTE    14, 1                                ;
       00000000'  00538           .ADDRESS P.AAT                                ;
            000E  0053C $STR$STRING2:
                                  .WORD    14                                   ;
          01  0E  0053E           .BYTE    14, 1                                ;
```

```
00000000'  00540          .ADDRESS P.AAU                        ;
     002B  00544 $IOB$OUTPUT:
                          .WORD   43                            ;
   01  0E  00546          .BYTE   14, 1                         ;
00000000'  00548          .ADDRESS P.AAX                        ;
     000B  0054C $IOB$OUTPUT:
                          .WORD   11                            ;
   01  0E  0054E          .BYTE   14, 1                         ;
00000000'  00550          .ADDRESS P.AAY                        ;
     0006  00554 $IOB$OUTPUT:
                          .WORD   6                             ;
   01  0E  00556          .BYTE   14, 1                         ;
00000000'  00558          .ADDRESS P.AAZ                        ;
     0009  0055C $IOB$OUTPUT:
                          .WORD   9                             ;
   01  0E  0055E          .BYTE   14, 1                         ;
00000000'  00560          .ADDRESS P.ABA                        ;
     000A  00564 $IOB$OUTPUT:
                          .WORD   10                            ;
   01  0E  00566          .BYTE   14, 1                         ;
00000000'  00568          .ADDRESS P.ABB                        ;
     0010  0056C $STR$STRINGO:
                          .WORD   16                            ;
   01  0E  0056E          .BYTE   14, 1                         ;
00000000'  00570          .ADDRESS P.ABD                        ;
     0002  00574 $STR$STRINGO:
                          .WORD   2                             ;
   01  0E  00576          .BYTE   14, 1                         ;
00000000'  00578          .ADDRESS P.ABG                        ;
     000C  0057C $IOB$OUTPUT:
                          .WORD   12                            ;
   01  0E  0057E          .BYTE   14, 1                         ;
00000000'  00580          .ADDRESS P.ABI                        ;
     000D  00584 $IOB$OUTPUT:
                          .WORD   13                            ;
   01  0E  00586          .BYTE   14, 1                         ;
00000000'  00588          .ADDRESS P.ABJ                        ;
     0007  0058C $IOB$OUTPUT:
                          .WORD   7                             ;
   01  0E  0058E          .BYTE   14, 1                         ;
00000000'  00590          .ADDRESS P.ABK                        ;

                          .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$_TMS11
                                        .EXTRN  TAB, MAXLST, TMSSTD
                                        .EXTRN  MSPACE, RINTES, CMDBLK
                                        .EXTRN  NDXVRL, NDXVRP, OUTIOB
                                        .EXTRN  LSTPTR, INDLVL, LSTSTK
                                        .EXTRN  BUCKET, CHRSIZ, ALLOWD
                                        .EXTRN  LCOUNT, RCOUNT, TCOUNT
                                        .EXTRN  LTYPE, LLINES, RTYPE
                                        .EXTRN  RLINES, TTYPE, TLINES
                                        .EXTRN  OPEN_ERROR, ENTMSG
                                        .EXTRN  GENTRY, LASTPG, PACBAS
                                        .EXTRN  PACPAG, PAGEQL, PAGMRG
                                        .EXTRN  PUTPAG, TMSINI, TMSPUT
                                        .EXTRN  XTNPAG, XST$JOIN
                                        .EXTRN  XPO$PUT, XPO$FAILURE
                                        .EXTRN  XST$ASCII, XPO$OPEN
                                        .EXTRN  XPO$GET, XPO$CLOSE

                                        .PSECT  $CODE$,NOWRT,2

                          0FFC 00000    .ENTRY  NDXFMT, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,-  ; 1129
                                                R11                                          ;
          5B 00000000G  EF   9E 00002   MOVAB   CMDBLK+4, R11                                ;
          5A 00000000G  EF   9E 00009   MOVAB   XPO$PUT, R10                                 ;
          59 00000000G  EF   9E 00010   MOVAB   XPO$FAILURE, R9                              ;
          58 00000000'  EF   9E 00017   MOVAB   PAGE_WIDTH, R8                               ;
          57 00000000G  EF   9E 0001E   MOVAB   IOB$+68, R7                                  ;
          5E      FF0C  CE   9E 00025   MOVAB   -244(SP), SP                                 ;
             00000000G  EF   D4 0002A   CLRL    LCOUNT                                       ; 1182
             00000000G  EF   D4 00030   CLRL    RCOUNT                                       ; 1183
                    04  A8   D4 00036   CLRL    DOING_LAST                                   ; 1184
          50        02  AB   32 00039   CVTWL   CMDBLK+6, R0                                 ; 1186
          03             50  B1 0003D   CMPW    R0, #3                                       ; 1189
                         0D  12 00040   BNEQ    1$                                           ;
 50    08  AB       0C  AB   C1 00042   ADDL3   CMDBLK+16, CMDBLK+12, R0                     ; 1190
          68    18 BB40  9E 00048       MOVAB   @CMDBLK+28[R0], PAGE_WIDTH                   ;
```

```
                                14  11 0004D       BRB     3$                                   ;
                    01          50  B1 0004F 1$:   CMPW    R0, #1                               ; 1192
                                0B  12 00052       BNEQ    2$                                   ;
                    50      08  AB  D0 00054       MOVL    CMDBLK+12, R0                        ; 1193
                    68      0C BB40 3E 00058       MOVAW   @CMDBLK+16[R0], PAGE_WIDTH           ;
                                04  11 0005D       BRB     3$                                   ;
                    68      08  AB  D0 0005F 2$:   MOVL    CMDBLK+12, PAGE_WIDTH                ; 1196
                    56      44  AB  9E 00063 3$:   MOVAB   CMDBLK+72, DSC_PTR                   ; 1206
                    53          66  3C 00067       MOVZWL  (DSC_PTR), CMD_LEN                   ; 1207
                    52      04  A6  D0 0006A       MOVL    4(DSC_PTR), CMD_PTR                  ; 1208
               62   53          3C  3A 0006E 4$:   LOCC    #60, CMD_LEN, (CMD_PTR)              ; 1210
                                02  12 00072       BNEQ    5$                                   ;
                                51  D4 00074       CLRL    R1                                   ;
                                51  D5 00076 5$:   TSTL    PTR                                  ;
                                06  13 00078       BEQL    6$                                   ;
                    61      5B  8F  90 0007A       MOVB    #91, (PTR)                           ; 1211
                                EE  11 0007E       BRB     4$                                   ;
               62   53          3E  3A 00080 6$:   LOCC    #62, CMD_LEN, (CMD_PTR)              ; 1213
                                02  12 00084       BNEQ    7$                                   ;
                                51  D4 00086       CLRL    R1                                   ;
                                51  D5 00088 7$:   TSTL    PTR                                  ;
                                06  13 0008A       BEQL    8$                                   ;
                    61      5D  8F  90 0008C       MOVB    #93, (PTR)                           ; 1214
                                EE  11 00090       BRB     6$                                   ;
               62   53          3B  3A 00092 8$:   LOCC    #59, CMD_LEN, (CMD_PTR)              ; 1216
                                02  12 00096       BNEQ    9$                                   ;
                                51  D4 00098       CLRL    R1                                   ;
                                51  D5 0009A 9$:   TSTL    PTR                                  ;
                                05  13 0009C       BEQL    10$                                  ;
                    61          2E  90 0009E       MOVB    #46, (PTR)                           ; 1217
                                EF  11 000A1       BRB     8$                                   ;
                    51      10  AB  D0 000A3 10$:  MOVL    CMDBLK+20, R1                        ; 1228
                    50      14  AB  D0 000A7       MOVL    CMDBLK+24, R0                        ; 1222
                                0A  13 000AB       BEQL    11$                                  ;
     00000000G EF   51          50  C3 000AD       SUBL3   R0, R1, ALLOWD                       ; 1228
                                17  11 000B5       BRB     13$                                  ;
                    01          6B  B1 000B7 11$:  CMPW    CMDBLK+4, #1                         ; 1232
                                0A  12 000BA       BNEQ    12$                                  ;
          00000000G EF      FC  A1  9E 000BC       MOVAB   -4(R1), ALLOWD                       ; 1234
                                08  11 000C4       BRB     13$                                  ;
          00000000G EF      EA  A1  9E 000C6 12$:  MOVAB   -22(R1), ALLOWD                      ; 1236
                    50          6B  32 000CE 13$:  CVTWL   CMDBLK+4, R0                         ; 1243
                    01          50  B1 000D1       CMPW    R0, #1                               ; 1246
                                03  13 000D4       BEQL    14$                                  ;
                              01A9  31 000D6       BRW     16$                                  ;
              F8    AD 00000000G    EF  B0 000D9 14$:  MOVW    NDXVRL, $STR$STRING1             ; 1260
              FA    AD              0E  90 000E1       MOVB    #14, $STR$STRING1+2              ;
              FB    AD              01  90 000E5       MOVB    #1, $STR$STRING1+3               ;
              FC    AD 00000000G    EF  D0 000E9       MOVL    NDXVRP, $STR$STRING1+4           ;
                            F8  AD  9F 000F1       PUSHAB  $STR$STRING1                         ;
                            0C  A8  9F 000F4       PUSHAB  $STR$STRING0                         ;
          00000000G EF          02  FB 000F7       CALLS   #2, XST$JOIN                         ;
                    67          50  D0 000FE       MOVL    R0, IOB$+68                          ;
              E8    A7          07  90 00101       MOVB    #7, IOB$+44                          ;
                                59  DD 00105       PUSHL   R9                                   ;
                                7E  D4 00107       CLRL    -(SP)                                ;
                            BC  A7  9F 00109       PUSHAB  IOB$                                 ;
```

```
                    6A          03 FB 0010C       CALLS   #3, XPO$PUT                         ;
                                56 DD 0010F       PUSHL   DSC_PTR                             ; 1264
                           14   A8 9F 00111       PUSHAB  $STR$STRING0                        ;
          00000000G EF          02 FB 00114       CALLS   #2, XST$JOIN                        ;
                    67          50 D0 0011B       MOVL    R0, IOB$+68                         ;
               E8   A7          07 90 0011E       MOVB    #7, IOB$+44                         ;
                                59 DD 00122       PUSHL   R9                                  ;
                                7E D4 00124       CLRL    -(SP)                               ;
                           BC   A7 9F 00126       PUSHAB  IOB$                                ;
                    6A          03 FB 00129       CALLS   #3, XPO$PUT                         ;
                    67     1C   A8 9E 0012C       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1265
               E8   A7          07 90 00130       MOVB    #7, IOB$+44                         ;
                                59 DD 00134       PUSHL   R9                                  ;
                                7E D4 00136       CLRL    -(SP)                               ;
                           BC   A7 9F 00138       PUSHAB  IOB$                                ;
                    6A          03 FB 0013B       CALLS   #3, XPO$PUT                         ;
                    67     24   A8 9E 0013E       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1266
               E8   A7          07 90 00142       MOVB    #7, IOB$+44                         ;
                                59 DD 00146       PUSHL   R9                                  ;
                                7E D4 00148       CLRL    -(SP)                               ;
                           BC   A7 9F 0014A       PUSHAB  IOB$                                ;
                    6A          03 FB 0014D       CALLS   #3, XPO$PUT                         ;
                    67     2C   A8 9E 00150       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1267
               E8   A7          07 90 00154       MOVB    #7, IOB$+44                         ;
                                59 DD 00158       PUSHL   R9                                  ;
                                7E D4 0015A       CLRL    -(SP)                               ;
                           BC   A7 9F 0015C       PUSHAB  IOB$                                ;
                    6A          03 FB 0015F       CALLS   #3, XPO$PUT                         ;
                    67     34   A8 9E 00162       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1268
               E8   A7          07 90 00166       MOVB    #7, IOB$+44                         ;
                                59 DD 0016A       PUSHL   R9                                  ;
                                7E D4 0016C       CLRL    -(SP)                               ;
                           BC   A7 9F 0016E       PUSHAB  IOB$                                ;
                    6A          03 FB 00171       CALLS   #3, XPO$PUT                         ;
                    67     3C   A8 9E 00174       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1269
               E8   A7          07 90 00178       MOVB    #7, IOB$+44                         ;
                                59 DD 0017C       PUSHL   R9                                  ;
                                7E D4 0017E       CLRL    -(SP)                               ;
                           BC   A7 9F 00180       PUSHAB  IOB$                                ;
                    6A          03 FB 00183       CALLS   #3, XPO$PUT                         ;
                    67     44   A8 9E 00186       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1270
               E8   A7          07 90 0018A       MOVB    #7, IOB$+44                         ;
                                59 DD 0018E       PUSHL   R9                                  ;
                                7E D4 00190       CLRL    -(SP)                               ;
                           BC   A7 9F 00192       PUSHAB  IOB$                                ;
                    6A          03 FB 00195       CALLS   #3, XPO$PUT                         ;
                    67     4C   A8 9E 00198       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1271
               E8   A7          07 90 0019C       MOVB    #7, IOB$+44                         ;
                                59 DD 001A0       PUSHL   R9                                  ;
                                7E D4 001A2       CLRL    -(SP)                               ;
                           BC   A7 9F 001A4       PUSHAB  IOB$                                ;
                    6A          03 FB 001A7       CALLS   #3, XPO$PUT                         ;
                    67     54   A8 9E 001AA       MOVAB   $IOB$OUTPUT, IOB$+68                ; 1272
               E8   A7          07 90 001AE       MOVB    #7, IOB$+44                         ;
                                59 DD 001B2       PUSHL   R9                                  ;
                                7E D4 001B4       CLRL    -(SP)                               ;
                           BC   A7 9F 001B6       PUSHAB  IOB$                                ;
```

```
                  6A           03  FB 001B9          CALLS   #3, XPO$PUT                          ;
                  67      5C   A8  9E 001BC          MOVAB   $IOB$OUTPUT, IOB$+68                 ; 1273
             E8   A7           07  90 001C0          MOVB    #7, IOB$+44                          ;
                               59  DD 001C4          PUSHL   R9                                   ;
                               7E  D4 001C6          CLRL    -(SP)                                ;
                          BC   A7  9F 001C8          PUSHAB  IOB$                                 ;
                  6A           03  FB 001CB          CALLS   #3, XPO$PUT                          ;
                  67      64   A8  9E 001CE          MOVAB   $IOB$OUTPUT, IOB$+68                 ; 1274
             E8   A7           07  90 001D2          MOVB    #7, IOB$+44                          ;
                               59  DD 001D6          PUSHL   R9                                   ;
                               7E  D4 001D8          CLRL    -(SP)                                ;
                          BC   A7  9F 001DA          PUSHAB  IOB$                                 ;
                  6A           03  FB 001DD          CALLS   #3, XPO$PUT                          ;
                               7E  D4 001E0          CLRL    -(SP)                                ; 1283
                               68  DD 001E2          PUSHL   PAGE_WIDTH                           ;
                  7E    0903   8F  3C 001E4          MOVZWL  #2307, -(SP)                         ;
        00000000G EF           03  FB 001E9          CALLS   #3, XST$ASCII                        ;
                  52           50  D0 001F0          MOVL    R0, R2                               ;
                               7E  D4 001F3          CLRL    -(SP)                                ;
                               68  DD 001F5          PUSHL   PAGE_WIDTH                           ;
                  7E    0903   8F  3C 001F7          MOVZWL  #2307, -(SP)                         ;
        00000000G EF           03  FB 001FC          CALLS   #3, XST$ASCII                        ;
                               50  DD 00203          PUSHL   R0                                   ;
                          74   A8  9F 00205          PUSHAB  $STR$STRING2                         ;
                               52  DD 00208          PUSHL   R2                                   ;
                          6C   A8  9F 0020A          PUSHAB  $STR$STRING0                         ;
        00000000G EF           04  FB 0020D          CALLS   #4, XST$JOIN                         ;
                  67           50  D0 00214          MOVL    R0, IOB$+68                          ;
             E8   A7           07  90 00217          MOVB    #7, IOB$+44                          ;
                               59  DD 0021B          PUSHL   R9                                   ;
                               7E  D4 0021D          CLRL    -(SP)                                ;
                          BC   A7  9F 0021F          PUSHAB  IOB$                                 ;
                  6A           03  FB 00222          CALLS   #3, XPO$PUT                          ;
                  67      7C   A8  9E 00225          MOVAB   $IOB$OUTPUT, IOB$+68                 ; 1285
             E8   A7           07  90 00229          MOVB    #7, IOB$+44                          ;
                               59  DD 0022D          PUSHL   R9                                   ;
                               7E  D4 0022F          CLRL    -(SP)                                ;
                          BC   A7  9F 00231          PUSHAB  IOB$                                 ;
                  6A           03  FB 00234          CALLS   #3, XPO$PUT                          ;
    39       FD   AB           04  E1 00237          BBC     #4, CMDBLK+1, 15$                    ; 1287
                  67    0084   C8  9E 0023C          MOVAB   $IOB$OUTPUT, IOB$+68                 ; 1293
             E8   A7           07  90 00241          MOVB    #7, IOB$+44                          ;
                               59  DD 00245          PUSHL   R9                                   ;
                               7E  D4 00247          CLRL    -(SP)                                ;
                          BC   A7  9F 00249          PUSHAB  IOB$                                 ;
                  6A           03  FB 0024C          CALLS   #3, XPO$PUT                          ;
                  67    008C   C8  9E 0024F          MOVAB   $IOB$OUTPUT, IOB$+68                 ; 1294
             E8   A7           07  90 00254          MOVB    #7, IOB$+44                          ;
                               59  DD 00258          PUSHL   R9                                   ;
                               7E  D4 0025A          CLRL    -(SP)                                ;
                          BC   A7  9F 0025C          PUSHAB  IOB$                                 ;
                  6A           03  FB 0025F          CALLS   #3, XPO$PUT                          ;
                  67    0094   C8  9E 00262          MOVAB   $IOB$OUTPUT, IOB$+68                 ; 1295
             E8   A7           07  90 00267          MOVB    #7, IOB$+44                          ;
                               59  DD 0026B          PUSHL   R9                                   ;
                               7E  D4 0026D          CLRL    -(SP)                                ;
                          BC   A7  9F 0026F          PUSHAB  IOB$                                 ;
```

```
                        6A              03  FB 00272          CALLS   #3, XPO$PUT                        ;
                        04        02    AB  B1 00275 15$:     CMPW    CMDBLK+6, #4                       ; 1298
                                        1D  12 00279          BNEQ    18$                                ;
                        67      009C    C8  9E 0027B          MOVAB   $IOB$OUTPUT, IOB$+68               ;
                                        72  11 00280          BRB     19$                                ;
                        02              50  B1 00282 16$:     CMPW    R0, #2                             ; 1301
                                        0E  19 00285          BLSS    17$                                ;
                        03              50  B1 00287          CMPW    R0, #3                             ;
                                        09  14 0028A          BGTR    17$                                ;
               00000000G EF             00  FB 0028C          CALLS   #0, TMSINI                         ; 1305
                                        6D  11 00293          BRB     20$                                ;
                        04              50  B1 00295 17$:     CMPW    R0, #4                             ; 1307
                                        68  12 00298 18$:     BNEQ    20$                                ;
                   F8   AD  00000000G   EF  B0 0029A          MOVW    NDXVRL, $STR$STRING1               ; 1312
                   FA   AD              0E  90 002A2          MOVB    #14, $STR$STRING1+2                ;
                   FB   AD              01  90 002A6          MOVB    #1, $STR$STRING1+3                 ;
                   FC   AD  00000000G   EF  D0 002AA          MOVL    NDXVRP, $STR$STRING1+4             ;
                                  F8    AD  9F 002B2          PUSHAB  $STR$STRING1                       ;
                                00A4    C8  9F 002B5          PUSHAB  $STR$STRING0                       ;
               00000000G EF             02  FB 002B9          CALLS   #2, XST$JOIN                       ;
                        67              50  D0 002C0          MOVL    R0, IOB$+68                        ;
                   E8   A7              07  90 002C3          MOVB    #7, IOB$+44                        ;
                                        59  DD 002C7          PUSHL   R9                                 ;
                                        7E  D4 002C9          CLRL    -(SP)                              ;
                                  BC    A7  9F 002CB          PUSHAB  IOB$                               ;
                        6A              03  FB 002CE          CALLS   #3, XPO$PUT                        ;
                                        56  DD 002D1          PUSHL   DSC_PTR                            ; 1313
                                00AC    C8  9F 002D3          PUSHAB  $STR$STRING0                       ;
               00000000G EF             02  FB 002D7          CALLS   #2, XST$JOIN                       ;
                        67              50  D0 002DE          MOVL    R0, IOB$+68                        ;
                   E8   A7              07  90 002E1          MOVB    #7, IOB$+44                        ;
                                        59  DD 002E5          PUSHL   R9                                 ;
                                        7E  D4 002E7          CLRL    -(SP)                              ;
                                  BC    A7  9F 002E9          PUSHAB  IOB$                               ;
                        6A              03  FB 002EC          CALLS   #3, XPO$PUT                        ;
                        67      00B4    C8  9E 002EF          MOVAB   $IOB$OUTPUT, IOB$+68               ; 1314
                   E8   A7              07  90 002F4 19$:     MOVB    #7, IOB$+44                        ;
                                        59  DD 002F8          PUSHL   R9                                 ;
                                        7E  D4 002FA          CLRL    -(SP)                              ;
                                  BC    A7  9F 002FC          PUSHAB  IOB$                               ;
                        6A              03  FB 002FF          CALLS   #3, XPO$PUT                        ;
                        56        34    AB  9E 00302 20$:     MOVAB   CMDBLK+56, DSC_PTR                 ; 1319
           03      FC   AB              02  E0 00306          BBS     #2, CMDBLK, 22$                    ; 1321
                                      00A2  31 0030B 21$:     BRW     27$                                ;
                                        66  B5 0030E 22$:     TSTW    (DSC_PTR)                          ;
                                        F9  13 00310          BEQL    21$                                ;
 00F4    8F        00           6E      00  2C 00312          MOVC5   #0, (SP), #0, #244, IOB$           ; 1330
                                        6E     00319                                                     ;
                        6E  0301003D    8F  D0 0031A          MOVL    #50397245, IOB$                    ;
                   1E   AE      020E    8F  B0 00321          MOVW    #526, IOB$RESULTANT+2              ;
                   04   AE              56  D0 00327          MOVL    DSC_PTR, IOB$+4                    ; 1333
                   2C   AE              01  90 0032B          MOVB    #1, IOB$+44                        ;
                        00000000G       EF  9F 0032F          PUSHAB  OPEN_ERROR                         ;
                                        7E  D4 00335          CLRL    -(SP)                              ;
                                  08    AE  9F 00337          PUSHAB  IOB$                               ;
               00000000G EF             03  FB 0033A          CALLS   #3, XPO$OPEN                       ;
                   2C   AE              06  90 00341 23$:     MOVB    #6, IOB$+44                        ; 1335
```

```
                                 59   DD 00345          PUSHL   R9                                      :
                                 7E   D4 00347          CLRL    -(SP)                                   :
                         08      AE   9F 00349          PUSHAB  IOB$                                    :
          00000000G   EF         03   FB 0034C          CALLS   #3, XPO$GET                             :
          00208001    8F         50   D1 00353          CMPL    R0, #2129921                            :
                                 40   12 0035A          BNEQ    26$                                     :
                      50         6B   32 0035C          CVTWL   CMDBLK+4, R0                            : 1338
                      02         50   B1 0035F          CMPW    R0, #2                                  : 1341
                                 1A   19 00362          BLSS    24$                                     :
                      03         50   B1 00364          CMPW    R0, #3                                  :
                                 15   14 00367          BGTR    24$                                     :
                                 7E   D4 00369          CLRL    -(SP)                                   : 1342
                         BC      A7   9F 0036B          PUSHAB  OUTIOB                                  :
                         40      AE   DD 0036E          PUSHL   REQUIRE_IOB+56                          :
                      7E 40      AE   3C 00371          MOVZWL  REQUIRE_IOB+52, -(SP)                   :
          00000000G   EF         04   FB 00375          CALLS   #4, TMSPUT                              :
                                 C3   11 0037C          BRB     23$                                     :
                      01         50   B1 0037E 24$:     CMPW    R0, #1                                  : 1344
                                 05   13 00381          BEQL    25$                                     :
                      04         50   B1 00383          CMPW    R0, #4                                  :
                                 B9   12 00386          BNEQ    23$                                     :
                      67 34      AE   9E 00388 25$:     MOVAB   $IOB$OUTPUT, IOB$+68                    : 1345
                 E8   A7         07   90 0038C          MOVB    #7, IOB$+44                             :
                                 59   DD 00390          PUSHL   R9                                      :
                                 7E   D4 00392          CLRL    -(SP)                                   :
                         BC      A7   9F 00394          PUSHAB  IOB$                                    :
                      6A         03   FB 00397          CALLS   #3, XPO$PUT                             :
                                 A5   11 0039A          BRB     23$                                     : 1335
                 2C   AE         02   90 0039C 26$:     MOVB    #2, IOB$+44                             : 1351
                                 59   DD 003A0          PUSHL   R9                                      :
                                 7E   D4 003A2          CLRL    -(SP)                                   :
                         08      AE   9F 003A4          PUSHAB  IOB$                                    :
          00000000G   EF         03   FB 003A7          CALLS   #3, XPO$CLOSE                           :
                                 2B   11 003AE          BRB     28$                                     : 1321
                      01         6B   B1 003B0 27$:     CMPW    CMDBLK+4, #1                            : 1354
                                 26   12 003B3          BNEQ    28$                                     :
                      67 00BC    C8   9E 003B5          MOVAB   $IOB$OUTPUT, IOB$+68                    : 1361
                 E8   A7         07   90 003BA          MOVB    #7, IOB$+44                             :
                                 59   DD 003BE          PUSHL   R9                                      :
                                 7E   D4 003C0          CLRL    -(SP)                                   :
                         BC      A7   9F 003C2          PUSHAB  IOB$                                    :
                      6A         03   FB 003C5          CALLS   #3, XPO$PUT                             :
                      67 00C4    C8   9E 003C8          MOVAB   $IOB$OUTPUT, IOB$+68                    : 1362
                 E8   A7         07   90 003CD          MOVB    #7, IOB$+44                             :
                                 59   DD 003D1          PUSHL   R9                                      :
                                 7E   D4 003D3          CLRL    -(SP)                                   :
                         BC      A7   9F 003D5          PUSHAB  IOB$                                    :
                      6A         03   FB 003D8          CALLS   #3, XPO$PUT                             :
                      02 04      AB   B1 003DB 28$:     CMPW    CMDBLK+8, #2                            : 1365
                                 09   13 003DF          BEQL    29$                                     :
                                 7E   D4 003E1          CLRL    -(SP)                                   : 1372
          00000000V   EF         01   FB 003E3          CALLS   #1, EXAM_BUCKET                         :
                      52         01   D0 003EA 29$:     MOVL    #1, I                                   : 1377
                                 52   DD 003ED 30$:     PUSHL   I                                       : 1378
          00000000V   EF         01   FB 003EF          CALLS   #1, EXAM_BUCKET                         :
       F3             52         1A   F3 003F6          AOBLEQ  #26, I, 30$                             :
                      02 04      AB   B1 003FA          CMPW    CMDBLK+8, #2                            : 1380
```

```
                                 09  12 003FE          BNEQ    31$                          ;
                                 7E  D4 00400          CLRL    -(SP)                        ; 1385
              00000000V  EF      01  FB 00402          CALLS   #1, EXAM_BUCKET              ;
              00000000G  EF      00  FB 00409 31$:     CALLS   #0, LASTPG                   ; 1390
                                     04 00410          RET                                  ; 1392

; Routine Size:  1041 bytes,    Routine Base:  $CODE$ + 0000
```

```
%SBTTL 'EXAM_BUCKET -- Print contents of a bucket'
ROUTINE EXAM_BUCKET (BKT) : NOVALUE =
!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine is called to examine the contents of a bucket.
!
!       It generates a guide heading if one was requested via the
!       /GUIDE_HEADING qualifier.
!
! FORMAL PARAMETERS:
!
!       BKT - bucket number to examine
!
! IMPLICIT INPUTS:
!
!       CMDBLK - command line information block
!
! IMPLICIT OUTPUTS:
!
!       INDLVL  - subindex level is set to zero
!       LSTPTR - current entry pointer is zeroed
!       LINE_TYPE- is set if a guide heading is generated
!                  and at the end of the bucket
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
! SIDE EFFECTS:
!
!       None
!--
    BEGIN

    LOCAL
        LAST_SUB_BUCKET,
        TEXT_GENERATED;

    TEXT_GENERATED = FALSE;

    IF .BKT EQL 0
    THEN
        !
        ! Examining the nonalphabetic bucket (bucket 0).
        !
        ! The only sub-bucket used is zero because nonalphabetics do
        ! not sort correctly using the 'squared bucket' approach.
        !
        LAST_SUB_BUCKET = 0
    ELSE
        !
        ! Not examining the nonalphabetic bucket. Use sub-buckets 0-26.
        !
        LAST_SUB_BUCKET = 26;
```

```
INCR I FROM 0 TO .LAST_SUB_BUCKET DO
    BEGIN
    LSTPTR=0;                                   ! Set starting point for listing at head
    INDLVL=0;                                   ! Set starting indent level

    IF GENTRY (.BKT, .I, FALSE)
    THEN
        BEGIN
        !
        ! Non empty bucket
        !
        IF .CMDBLK [NDX$V_GUIDE] AND (NOT .TEXT_GENERATED)
        THEN
            BEGIN
            !
            ! Build guide heading
            !
            LINE_TYPE = GUIDE;

            IF .CMDBLK [NDX$H_FORMAT] EQL DSR
            THEN
                BEGIN
                !
                ! RUNOFF output
                !
                BIND
                    BUCKET_NAMES = UPLIT (
                        CH$PTR (UPLIT ('   ')),
                        CH$PTR (UPLIT ('-A-')),
                        CH$PTR (UPLIT ('-B-')),
                        CH$PTR (UPLIT ('-C-')),
                        CH$PTR (UPLIT ('-D-')),
                        CH$PTR (UPLIT ('-E-')),
                        CH$PTR (UPLIT ('-F-')),
                        CH$PTR (UPLIT ('-G-')),
                        CH$PTR (UPLIT ('-H-')),
                        CH$PTR (UPLIT ('-I-')),
                        CH$PTR (UPLIT ('-J-')),
                        CH$PTR (UPLIT ('-K-')),
                        CH$PTR (UPLIT ('-L-')),
                        CH$PTR (UPLIT ('-M-')),
                        CH$PTR (UPLIT ('-N-')),
                        CH$PTR (UPLIT ('-O-')),
                        CH$PTR (UPLIT ('-P-')),
                        CH$PTR (UPLIT ('-Q-')),
                        CH$PTR (UPLIT ('-R-')),
                        CH$PTR (UPLIT ('-S-')),
                        CH$PTR (UPLIT ('-T-')),
                        CH$PTR (UPLIT ('-U-')),
                        CH$PTR (UPLIT ('-V-')),
                        CH$PTR (UPLIT ('-W-')),
                        CH$PTR (UPLIT ('-X-')),
                        CH$PTR (UPLIT ('-Y-')),
                        CH$PTR (UPLIT ('-Z-'))
                    ) : VECTOR;
```

```
        LOCAL
            INDENT;

        INDENT = .CMDBLK [NDX$G_COLUMN_WID] / 3;       ! Number of leading blanks
        INDENT = ((.INDENT + 1) / 2) * 2;              ! Round to next even number

        INDENT_LINE (.INDENT);
        FILL_LINE (3, .BUCKET_NAMES [.BKT], .INDENT, .CMDBLK [NDX$G_COLUMN_WID], TRUE, FALSE, FA
        INS_LINE (.INT_LINE_LENGTH, CH$PTR (RNO_LINE), .EXT_LINE_LENGTH);
        END
    ELSE
        BEGIN
        !
        ! TMS11 or TEX output
        !
        BIND
            BUCKET_NAMES = UPLIT (
                CH$PTR (UPLIT (' ')),
                CH$PTR (UPLIT ('A')),
                CH$PTR (UPLIT ('B')),
                CH$PTR (UPLIT ('C')),
                CH$PTR (UPLIT ('D')),
                CH$PTR (UPLIT ('E')),
                CH$PTR (UPLIT ('F')),
                CH$PTR (UPLIT ('G')),
                CH$PTR (UPLIT ('H')),
                CH$PTR (UPLIT ('I')),
                CH$PTR (UPLIT ('J')),
                CH$PTR (UPLIT ('K')),
                CH$PTR (UPLIT ('L')),
                CH$PTR (UPLIT ('M')),
                CH$PTR (UPLIT ('N')),
                CH$PTR (UPLIT ('O')),
                CH$PTR (UPLIT ('P')),
                CH$PTR (UPLIT ('Q')),
                CH$PTR (UPLIT ('R')),
                CH$PTR (UPLIT ('S')),
                CH$PTR (UPLIT ('T')),
                CH$PTR (UPLIT ('U')),
                CH$PTR (UPLIT ('V')),
                CH$PTR (UPLIT ('W')),
                CH$PTR (UPLIT ('X')),
                CH$PTR (UPLIT ('Y')),
                CH$PTR (UPLIT ('Z'))
            ) : VECTOR;

        INS_LINE (1, .BUCKET_NAMES [.BKT], 1);
        END;

    LINE_TYPE = GUIDE_FILL;
    INS_LINE (4, .BLANKS, 4);
    END;

!
! Now process rest of bucket
!
DO FORMAT_ENTRY () UNTIL NOT GENTRY (.BKT, .I, FALSE);
```

```
 4
 4              TEXT_GENERATED = TRUE;
 3              END;
 3
 2          END;
 2
 2      IF .TEXT_GENERATED
 2      THEN
 3          BEGIN
 3          LINE_TYPE = BKT_E;
 3          INS_LINE (4, .BLANKS, 4);          ! Skip a line
 2          END;
 2
 1      END;


                                                                              .PSECT  $PLIT$,NOWRT,NOEXE,2

                                                             0020B            .BLKB   1
                                              00  20  20  20 0020C P.ABM:     .ASCII  \   \<0>
                                              00  2D  41  2D 00210 P.ABN:     .ASCII  \-A-\<0>
                                              00  2D  42  2D 00214 P.ABO:     .ASCII  \-B-\<0>
                                              00  2D  43  2D 00218 P.ABP:     .ASCII  \-C-\<0>
                                              00  2D  44  2D 0021C P.ABQ:     .ASCII  \-D-\<0>
                                              00  2D  45  2D 00220 P.ABR:     .ASCII  \-E-\<0>
                                              00  2D  46  2D 00224 P.ABS:     .ASCII  \-F-\<0>
                                              00  2D  47  2D 00228 P.ABT:     .ASCII  \-G-\<0>
                                              00  2D  48  2D 0022C P.ABU:     .ASCII  \-H-\<0>
                                              00  2D  49  2D 00230 P.ABV:     .ASCII  \-I-\<0>
                                              00  2D  4A  2D 00234 P.ABW:     .ASCII  \-J-\<0>
                                              00  2D  4B  2D 00238 P.ABX:     .ASCII  \-K-\<0>
                                              00  2D  4C  2D 0023C P.ABY:     .ASCII  \-L-\<0>
                                              00  2D  4D  2D 00240 P.ABZ:     .ASCII  \-M-\<0>
                                              00  2D  4E  2D 00244 P.ACA:     .ASCII  \-N-\<0>
                                              00  2D  4F  2D 00248 P.ACB:     .ASCII  \-O-\<0>
                                              00  2D  50  2D 0024C P.ACC:     .ASCII  \-P-\<0>
                                              00  2D  51  2D 00250 P.ACD:     .ASCII  \-Q-\<0>
                                              00  2D  52  2D 00254 P.ACE:     .ASCII  \-R-\<0>
                                              00  2D  53  2D 00258 P.ACF:     .ASCII  \-S-\<0>
                                              00  2D  54  2D 0025C P.ACG:     .ASCII  \-T-\<0>
                                              00  2D  55  2D 00260 P.ACH:     .ASCII  \-U-\<0>
                                              00  2D  56  2D 00264 P.ACI:     .ASCII  \-V-\<0>
                                              00  2D  57  2D 00268 P.ACJ:     .ASCII  \-W-\<0>
                                              00  2D  58  2D 0026C P.ACK:     .ASCII  \-X-\<0>
                                              00  2D  59  2D 00270 P.ACL:     .ASCII  \-Y-\<0>
                                              00  2D  5A  2D 00274 P.ACM:     .ASCII  \-Z-\<0>
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00278 P.ABL:     .ADDRESS P.ABM, P.ABN, P.ABO, P.ABP, P.ABQ, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00290                     P.ABR, P.ABS, P.ABT, P.ABU, P.ABV, P.ABW, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 002A8                     P.ABX, P.ABY, P.ABZ, P.ACA, P.ACB, P.ACC, -
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 002C0                     P.ACD, P.ACE, P.ACF, P.ACG, P.ACH, P.ACI, -
                              00000000' 00000000' 00000000' 002D8                     P.ACJ, P.ACK, P.ACL, P.ACM
                                              00  00  00  20 002E4 P.ACO:     .ASCII  \ \<0><0><0>
                                              00  00  00  41 002E8 P.ACP:     .ASCII  \A\<0><0><0>
                                              00  00  00  42 002EC P.ACQ:     .ASCII  \B\<0><0><0>
                                              00  00  00  43 002F0 P.ACR:     .ASCII  \C\<0><0><0>
                                              00  00  00  44 002F4 P.ACS:     .ASCII  \D\<0><0><0>
```

```
                                    00  00  00  45  002F8 P.ACT:   .ASCII  \E\<0><0><0>                                    ;
                                    00  00  00  46  002FC P.ACU:   .ASCII  \F\<0><0><0>                                    ;
                                    00  00  00  47  00300 P.ACV:   .ASCII  \G\<0><0><0>                                    ;
                                    00  00  00  48  00304 P.ACW:   .ASCII  \H\<0><0><0>                                    ;
                                    00  00  00  49  00308 P.ACX:   .ASCII  \I\<0><0><0>                                    ;
                                    00  00  00  4A  0030C P.ACY:   .ASCII  \J\<0><0><0>                                    ;
                                    00  00  00  4B  00310 P.ACZ:   .ASCII  \K\<0><0><0>                                    ;
                                    00  00  00  4C  00314 P.ADA:   .ASCII  \L\<0><0><0>                                    ;
                                    00  00  00  4D  00318 P.ADB:   .ASCII  \M\<0><0><0>                                    ;
                                    00  00  00  4E  0031C P.ADC:   .ASCII  \N\<0><0><0>                                    ;
                                    00  00  00  4F  00320 P.ADD:   .ASCII  \O\<0><0><0>                                    ;
                                    00  00  00  50  00324 P.ADE:   .ASCII  \P\<0><0><0>                                    ;
                                    00  00  00  51  00328 P.ADF:   .ASCII  \Q\<0><0><0>                                    ;
                                    00  00  00  52  0032C P.ADG:   .ASCII  \R\<0><0><0>                                    ;
                                    00  00  00  53  00330 P.ADH:   .ASCII  \S\<0><0><0>                                    ;
                                    00  00  00  54  00334 P.ADI:   .ASCII  \T\<0><0><0>                                    ;
                                    00  00  00  55  00338 P.ADJ:   .ASCII  \U\<0><0><0>                                    ;
                                    00  00  00  56  0033C P.ADK:   .ASCII  \V\<0><0><0>                                    ;
                                    00  00  00  57  00340 P.ADL:   .ASCII  \W\<0><0><0>                                    ;
                                    00  00  00  58  00344 P.ADM:   .ASCII  \X\<0><0><0>                                    ;
                                    00  00  00  59  00348 P.ADN:   .ASCII  \Y\<0><0><0>                                    ;
                                    00  00  00  5A  0034C P.ADO:   .ASCII  \Z\<0><0><0>                                    ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00350 P.ACN: .ADDRESS P.ACO, P.ACP, P.ACQ, P.ACR, P.ACS, -      ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00368               P.ACT, P.ACU, P.ACV, P.ACW, P.ACX, P.ACY, -  ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00380               P.ACZ, P.ADA, P.ADB, P.ADC, P.ADD, P.ADE, -  ;
00000000' 00000000' 00000000' 00000000' 00000000' 00000000' 00398               P.ADF, P.ADG, P.ADH, P.ADI, P.ADJ, P.ADK, -  ;
                              00000000' 00000000' 00000000' 003B0               P.ADL, P.ADM, P.ADN, P.ADO                   ;

                                                  BUCKET_NAMES=        P.ABL
                                                  BUCKET_NAMES=        P.ACN

                                                           .PSECT  $CODE$,NOWRT,2

                                        0FFC 00000 EXAM_BUCKET:
                                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11            ; 1394
                    5B 00000000G   EF   9E 00002           MOVAB   GENTRY, R11                                     ;
                    5A 00000000V   EF   9E 00009           MOVAB   INS_LINE, R10                                   ;
                    59 00000000G   EF   9E 00010           MOVAB   CMDBLK+12, R9                                   ;
                    58 00000000'   EF   9E 00017           MOVAB   LINE_TYPE, R8                                   ;
                                   56   D4 0001E           CLRL    TEXT_GENERATED                                  ; 1434
                    54        04   AC   D0 00020           MOVL    BKT, R4                                         ; 1436
                                   04   12 00024           BNEQ    1$                                              ;
                                   57   D4 00026           CLRL    LAST_SUB_BUCKET                                 ; 1444
                                   03   11 00028           BRB     2$                                              ;
                    57             1A   D0 0002A 1$:       MOVL    #26, LAST_SUB_BUCKET                            ; 1449
                    55             01   CE 0002D 2$:       MNEGL   #1, I                                           ; 1451
                                 009B   31 00030 3$:       BRW     8$                                              ;
                       00000000G   EF   D4 00033 4$:       CLRL    LSTPTR                                          ; 1453
                       00000000G   EF   D4 00039           CLRL    INDLVL                                          ; 1454
                                   7E   D4 0003F           CLRL    -(SP)                                           ; 1456
                                   30   BB 00041           PUSHR   #^M<R4,R5>                                      ;
                    6B             03   FB 00043           CALLS   #3, GENTRY                                      ;
                    E7             50   E9 00046           BLBC    R0, 3$                                          ;
                              F4   A9   95 00049           TSTB    CMDBLK                                          ; 1462
                                   6C   18 0004C           BGEQ    7$                                              ;
                    69             56   E8 0004E           BLBS    TEXT_GENERATED, 7$                              ;
```

```
                                68      03  D0 00051         MOVL    #3, LINE_TYPE                            ; 1468
              53                54      02  78 00054         ASHL    #2, R4, R3                               ; 1514
                                01  F8  A9  B1 00058         CMPW    CMDBLK+4, #1                             ; 1470
                                        3E  12 0005C         BNEQ    5$                                       ;
              52                69      03  C7 0005E         DIVL3   #3, CMDBLK+12, INDENT                    ; 1510
                                50  01  A2  9E 00062         MOVAB   1(R2), R0                                ; 1511
                                50      02  C6 00066         DIVL2   #2, R0                                   ;
              52                50      01  78 00069         ASHL    #1, R0, INDENT                           ;
                                        52  DD 0006D         PUSHL   INDENT                                   ; 1513
                    00000000V   EF      01  FB 0006F         CALLS   #1, INDENT_LINE                          ;
                                        7E  7C 00076         CLRQ    -(SP)                                    ; 1514
                                        01  DD 00078         PUSHL   #1                                       ;
                                        69  DD 0007A         PUSHL   CMDBLK+12                                ;
                                        52  DD 0007C         PUSHL   INDENT                                   ;
                          00000000'EF43 9F 0007E             PUSHAB  BUCKET_NAMES[R3]                         ;
                                        9E  DD 00085         PUSHL   @(SP)+                                   ;
                                        03  DD 00087         PUSHL   #3                                       ;
                    00000000V   EF      07  FB 00089         CALLS   #7, FILL_LINE                            ;
                                    F8  A8  DD 00090         PUSHL   EXT_LINE_LENGTH                          ; 1515
                                  FB44  C8  9F 00093         PUSHAB  RNO_LINE                                 ;
                                    F4  A8  DD 00097         PUSHL   INT_LINE_LENGTH                          ;
                                        0D  11 0009A         BRB     6$                                       ;
                                        01  DD 0009C 5$:     PUSHL   #1                                       ; 1553
                          00000000'EF43 9F 0009E             PUSHAB  BUCKET_NAMES[R3]                         ;
                                        9E  DD 000A5         PUSHL   @(SP)+                                   ;
                                        01  DD 000A7         PUSHL   #1                                       ;
                                6A      03  FB 000A9 6$:     CALLS   #3, INS_LINE                             ;
                                68      04  D0 000AC         MOVL    #4, LINE_TYPE                            ; 1556
                                        04  DD 000AF         PUSHL   #4                                       ; 1557
                                  FB3C  C8  DD 000B1         PUSHL   BLANKS                                   ;
                                        04  DD 000B5         PUSHL   #4                                       ;
                                6A      03  FB 000B7         CALLS   #3, INS_LINE                             ;
                    00000000V   EF      00  FB 000BA 7$:     CALLS   #0, FORMAT_ENTRY                         ; 1563
                                        7E  D4 000C1         CLRL    -(SP)                                    ;
                                        30  BB 000C3         PUSHR   #^M<R4,R5>                               ;
                                6B      03  FB 000C5         CALLS   #3, GENTRY                               ;
                                EF      50  E8 000C8         BLBS    R0, 7$                                   ;
                                56      01  D0 000CB         MOVL    #1, TEXT_GENERATED                       ; 1565
      FF5F      55              01      57  F1 000CE 8$:     ACBL    LAST_SUB_BUCKET, #1, I, 4$               ; 1451
                                0E      56  E9 000D4         BLBC    TEXT_GENERATED, 9$                       ; 1570
                                68      01  D0 000D7         MOVL    #1, LINE_TYPE                            ; 1573
                                        04  DD 000DA         PUSHL   #4                                       ; 1574
                                  FB3C  C8  DD 000DC         PUSHL   BLANKS                                   ;
                                        04  DD 000E0         PUSHL   #4                                       ;
                                6A      03  FB 000E2         CALLS   #3, INS_LINE                             ;
                                            04 000E5 9$:     RET                                              ; 1577

; Routine Size:  230 bytes,    Routine Base:  $CODE$ + 0411


;  802          1578  1
```

```
%SBTTL 'FORMAT_ENTRY -- collect & print index entry'
ROUTINE FORMAT_ENTRY : NOVALUE =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Collect and print the entire index line, including
!       name and all line number references.
!
!       No line is generated if doing a master index and the
!       subindex level is greater than the deepest level
!       allowed in the master index.
!
! FORMAL PARAMETERS:
!
!       None
!
! IMPLICIT INPUTS:
!
!       LSTPTR          - address of list item for index entry
!       INDLVL          - subindex level
!       LSTSTK          - subindex list stack
!       LINE_TYPE       - current line type
!       CMDBLK          - command line information block
!       CHRSIZ          - TMS character size vector
!
! IMPLICIT OUTPUTS:
!
!       LINE_TYPE       - current line type
!       LINE_POINTER    - pointer to next character position in output line
!       EXT_LINE_LENGTH - external line length
!       INT_LINE_LENGTH - internal line length
!       TMS_LINE_LENGTH - length of line in TMS relative units
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
! SIDE EFFECTS:
!
!       None
!
!--

    BEGIN

    LOCAL
        INDENT,
        STR_VEC : REF VECTOR,
        CHAR_COUNT,
        TEXT_PTR;

    IF .INDLVL GTR .CMDBLK [NDX$H_LEVEL] THEN RETURN;

    !
    ! Get pointer to text and length of text
```

```
        !
        STR_VEC = .LSTPTR [XE$A_TEXT];
        CHAR_COUNT = .STR_VEC [0];                          ! Length is 1st fullword of string
        TEXT_PTR = CH$PTR (STR_VEC [1]);

        IF .LINE_TYPE NEQ CONT_HEAD THEN LINE_TYPE = (IF .INDLVL EQL 0 THEN ENTRY_B ELSE SUB_B);

        !
        ! Indent the new line
        !
        INDENT = 2 * .INDLVL;
        INDENT_LINE (.INDENT);

        !
        ! Now format the entry
        !
        FILL_LINE (.CHAR_COUNT, .TEXT_PTR, .INDENT, .CMDBLK [NDX$G_COLUMN_WID], FALSE, FALSE, FALSE);

        IF .LINE_TYPE EQL CONT_HEAD
        THEN
            BEGIN
            !
            ! Doing a continuation heading
            !
            IF (.LSTSTK [.INDLVL + 1] EQL 0)
            OR (.INDLVL EQL .CMDBLK [NDX$H_LEVEL])
            THEN
                !
                ! Append ' (Cont.)' to line
                !
                FILL_LINE (7, CH$PTR (UPLIT ('(Cont.)')), .INDENT, .CMDBLK [NDX$G_COLUMN_WID], FALSE, FALSE, TRU
            END
        ELSE
            BEGIN
            !
            ! Doing a normal entry
            !
            IF .LSTPTR [XE$A_REF] NEQ 0
            THEN
                BEGIN
                !
                ! Have page references.
                !
                IF .CMDBLK [NDX$V_PAGES]                    ! Generating page references
                OR .CMDBLK [NDX$V_MASTER]                   ! or master index
                THEN
                    BEGIN
                    !
                    ! Insert a null following the entry so that we can detect the
                    ! start of the page references for building telltale headings
                    ! or continuation headings on the last page.
                    !
                    CH$WCHAR_A (0, LINE_POINTER);
                    EXT_LINE_LENGTH = .EXT_LINE_LENGTH + 1;
                    INT_LINE_LENGTH = .INT_LINE_LENGTH + 1;

                    TMS_LINE_LENGTH = .TMS_LINE_LENGTH
```

```
                + (IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE THEN .CHRSIZ [%C' '] ELSE .CHRSIZ [%C',']);

            INS_PAGE_NO (.LSTPTR [XE$A_REF]); ! Insert page numbers
            END;
        END
    ELSE
        BEGIN
        !
        ! No page references
        !
        IF  .CMDBLK [NDX$V_MASTER]
        AND (.INDLVL EQL .CMDBLK [NDX$H_LEVEL])
        AND (.LSTPTR [XE$A_SUBX] NEQ 0)
        THEN
            BEGIN
            !
            ! Doing master indexing, at deepest level specified,
            ! and not at the bottom of a .Y - generate book references
            !
            LOCAL
                M_INDENT,
                M_EXT,
                NEW_LINE,
                XMPTR : REF $XM_BLOCK;

            NEW_LINE = FALSE;
            XMPTR = .LSTPTR [XE$A_BOOK_LIST];

            IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE
            THEN
                BEGIN
                !
                ! Set indent and maximum external length for separate format
                !
                M_INDENT = .CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID];
                M_EXT = .PAGE_WIDTH;
                END
            ELSE
                BEGIN
                !
                ! Set indent and maximum external length for other formats.
                !
                M_INDENT = .INDENT;
                M_EXT = .CMDBLK [NDX$G_COLUMN_WID];
                END;

            !
            ! Insert a null following the entry so that we can detect the
            ! start of the page references for building telltale headings
            ! or continuation headings on the last page.
            !
            CH$WCHAR_A (0, LINE_POINTER);
            EXT_LINE_LENGTH = .EXT_LINE_LENGTH + 1;
            INT_LINE_LENGTH = .INT_LINE_LENGTH + 1;

            TMS_LINE_LENGTH = .TMS_LINE_LENGTH
                + (IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE THEN .CHRSIZ [%C' '] ELSE .CHRSIZ [%C',']);
```

```
5
5                 WHILE .XMPTR NEQ 0 DO
6                     BEGIN
6                     BOOK_REF (.XMPTR [XM$A_BOOK], .M_INDENT, .M_EXT, .NEW_LINE, FALSE);
6                     IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE THEN NEW_LINE = TRUE;
6
6                     IF .XMPTR [XM$A_LINK] NEQ 0
6                     THEN
6                         XMPTR = .XMPTR [XM$A_LINK]
6                     ELSE
6                         XMPTR = 0;
5                     END;
4                 END;
3             END;
2         END;
2     IF .LINE_TYPE NEQ CONT_HEAD THEN LINE_TYPE = (IF .INDLVL EQL 0 THEN ENTRY_E ELSE SUB_E);
2
2     INS_LINE (.INT_LINE_LENGTH, CH$PTR (RNO_LINE), .EXT_LINE_LENGTH);
2
1     END;                        !End of FORMAT_ENTRY


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

00  29  2E  74  6E  6F  43  28  003BC P.ADP:    .ASCII  \(Cont.)\<0>                          ;


                                                        .PSECT  $CODE$,NOWRT,2

                                 0FFC 00000 FORMAT_ENTRY:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11  ; 1580
                5B 00000000G  EF  9E 00002      MOVAB   CHRSIZ, R11                           ;
                5A 00000000V  EF  9E 00009      MOVAB   FILL_LINE, R10                        ;
                59 00000000G  EF  9E 00010      MOVAB   LSTPTR, R9                            ;
                58 00000000G  EF  9E 00017      MOVAB   INDLVL, R8                            ;
                57 00000000G  EF  9E 0001E      MOVAB   CMDBLK+12, R7                         ;
                56 00000000'  EF  9E 00025      MOVAB   LINE_TYPE, R6                         ;
                51                68  D0 0002C  MOVL    INDLVL, R1                            ; 1632
51    FE   A7   10                00  EC 0002F  CMPV    #0, #16, CMDBLK+10, R1                ;
                                  01  18 00035  BGEQ    1$                                    ;
                                      04 00037  RET                                           ;
                50                69  D0 00038 1$: MOVL LSTPTR, R0                            ; 1637
                50           10   A0  D0 0003B  MOVL    16(R0), STR_VEC                       ;
                54                60  D0 0003F  MOVL    (STR_VEC), CHAR_COUNT                 ; 1638
                52           04   A0  9E 00042  MOVAB   4(R0), TEXT_PTR                       ; 1639
                0B                66  D1 00046  CMPL    LINE_TYPE, #11                        ; 1641
                0F                13     00049  BEQL    4$                                    ;
                51                D5     0004B  TSTL    R1                                    ;
                05                12     0004D  BNEQ    2$                                    ;
                50                05  D0 0004F  MOVL    #5, R0                                ;
                03                11     00052  BRB     3$                                    ;
                50                08  D0 00054 2$: MOVL #8, R0                                ;
                66                50  D0 00057 3$: MOVL R0, LINE_TYPE                         ;
           53   51                01  78 0005A 4$: ASHL #1, R1, INDENT                        ; 1646
```

```
                                          53  DD 0005E          PUSHL   INDENT                                  ; 1647
                 00000000V  EF            01  FB 00060          CALLS   #1, INDENT_LINE                         ;
                                          7E  7C 00067          CLRQ    -(SP)                                   ; 1652
                                          7E  D4 00069          CLRL    -(SP)                                   ;
                                          67  DD 0006B          PUSHL   CMDBLK+12                               ;
                                          0C  BB 0006D          PUSHR   #^M<R2,R3>                              ;
                                          54  DD 0006F          PUSHL   CHAR_COUNT                              ;
                            6A            07  FB 00071          CALLS   #7, FILL_LINE                           ;
                            0B            66  D1 00074          CMPL    LINE_TYPE, #11                          ; 1654
                                          29  12 00077          BNEQ    6$                                      ;
                            50            68  D0 00079          MOVL    INDLVL, R0                              ; 1660
                            00000000GEF40     D5 0007C          TSTL    LSTSTK+4[R0]                            ;
                                          08  13 00083          BEQL    5$                                      ;
        68        FE    A7  10            00  EC 00085          CMPV    #0, #16, CMDBLK+10, INDLVL              ; 1661
                                          66  12 0008B          BNEQ    12$                                     ;
                                          01  DD 0008D  5$:     PUSHL   #1                                      ; 1666
                                          7E  7C 0008F          CLRQ    -(SP)                                   ;
                                          67  DD 00091          PUSHL   CMDBLK+12                               ;
                                          53  DD 00093          PUSHL   INDENT                                  ;
                            00000000'  EF 9F 00095          PUSHAB  P.ADP                                   ;
                                          07  DD 0009B          PUSHL   #7                                      ;
                            6A            07  FB 0009D          CALLS   #7, FILL_LINE                           ;
                                          43  11 000A0          BRB     10$                                     ; 1656
                            51            69  D0 000A2  6$:     MOVL    LSTPTR, R1                              ; 1673
                                  0C      A1  D5 000A5          TSTL    12(R1)                                  ;
                                          3E  13 000A8          BEQL    11$                                     ;
                05          F4  A7        03  E0 000AA          BBS     #3, CMDBLK, 7$                          ; 1679
                34          F5  A7        02  E1 000AF          BBC     #2, CMDBLK+1, 11$                       ; 1680
                                FB40      D6  94 000B4  7$:     CLRB    @LINE_POINTER                           ; 1688
                                FB40      C6  D6 000B8          INCL    LINE_POINTER                            ;
                                  F8      A6  D6 000BC          INCL    EXT_LINE_LENGTH                         ; 1689
                                  F4      A6  D6 000BF          INCL    INT_LINE_LENGTH                         ; 1690
                            50            6B  D0 000C2          MOVL    CHRSIZ, R0                              ; 1693
                            03    FA      A7  B1 000C5          CMPW    CMDBLK+6, #3                            ;
                                          07  12 000C9          BNEQ    8$                                      ;
                            50  0080      C0  D0 000CB          MOVL    128(R0), R0                             ;
                                          05  11 000D0          BRB     9$                                      ;
                            50  00B0      C0  D0 000D2  8$:     MOVL    176(R0), R0                             ;
                        FC  A6            50  C0 000D7  9$:     ADDL2   R0, TMS_LINE_LENGTH                     ;
                                  0C      A1  DD 000DB          PUSHL   12(R1)                                  ; 1695
                 00000000V  EF            01  FB 000DE          CALLS   #1, INS_PAGE_NO                         ;
                                    0081  31 000E5  10$:    BRW     20$                                     ; 1673
                7C          F5  A7        02  E1 000E8  11$:    BBC     #2, CMDBLK+1, 20$                       ; 1703
        68        FE    A7  10            00  EC 000ED          CMPV    #0, #16, CMDBLK+10, INDLVL              ; 1704
                                          74  12 000F3  12$:    BNEQ    20$                                     ;
                                  08      A1  D5 000F5          TSTL    8(R1)                                   ; 1705
                                          6F  13 000F8          BEQL    20$                                     ;
                                          54  D4 000FA          CLRL    NEW_LINE                                ; 1718
                            52    1C      A1  D0 000FC          MOVL    28(R1), XMPTR                           ; 1719
                                          51  D4 00100          CLRL    R1                                      ; 1721
                            03    FA      A7  B1 00102          CMPW    CMDBLK+6, #3                            ;
                                          0D  12 00106          BNEQ    13$                                     ;
                                          51  D6 00108          INCL    R1                                      ;
                55          67    04      A7  C1 0010A          ADDL3   CMDBLK+16, CMDBLK+12, M_INDENT          ; 1727
                            53    04      A6  D0 0010F          MOVL    PAGE_WIDTH, M_EXT                       ; 1728
                                          06  11 00113          BRB     14$                                     ; 1721
                            55            53  D0 00115  13$:    MOVL    INDENT, M_INDENT                        ; 1735
```

```
                53                 67  D0 00118        MOVL    CMDBLK+12, M_EXT                  ; 1736
                          FB40  D6  94 0011B 14$:   CLRB    @LINE_POINTER                     ; 1744
                          FB40  C6  D6 0011F        INCL    LINE_POINTER                      ;
                            F8  A6  D6 00123        INCL    EXT_LINE_LENGTH                   ; 1745
                            F4  A6  D6 00126        INCL    INT_LINE_LENGTH                   ; 1746
                50                 6B  D0 00129        MOVL    CHRSIZ, R0                        ; 1749
                07                 51  E9 0012C        BLBC    R1, 15$                           ;
                50        0080  C0  D0 0012F        MOVL    128(R0), R0                       ;
                                   05  11 00134        BRB     16$                               ;
                50        00B0  C0  D0 00136 15$:   MOVL    176(R0), R0                       ;
           FC   A6                 50  C0 0013B 16$:   ADDL2   R0, TMS_LINE_LENGTH               ;
                                   52  D5 0013F 17$:   TSTL    XMPTR                             ; 1751
                                   26  13 00141        BEQL    20$                               ;
                                   7E  D4 00143        CLRL    -(SP)                             ; 1753
                                   18  BB 00145        PUSHR   #^M<R3,R4>                        ;
                                   55  DD 00147        PUSHL   M_INDENT                          ;
                            04  A2  DD 00149        PUSHL   4(XMPTR)                          ;
     00000000V  EF                 05  FB 0014C        CALLS   #5, BOOK_REF                      ;
                03          FA  A7  B1 00153        CMPW    CMDBLK+6, #3                      ; 1754
                                   03  12 00157        BNEQ    18$                               ;
                54                 01  D0 00159        MOVL    #1, NEW_LINE                      ;
                                   62  D5 0015C 18$:   TSTL    (XMPTR)                           ; 1756
                                   05  13 0015E        BEQL    19$                               ;
                52                 62  D0 00160        MOVL    (XMPTR), XMPTR                    ; 1758
                                   DA  11 00163        BRB     17$                               ;
                                   52  D4 00165 19$:   CLRL    XMPTR                             ; 1760
                                   D6  11 00167        BRB     17$                               ; 1751
                0B                 66  D1 00169 20$:   CMPL    LINE_TYPE, #11                    ; 1766
                                   0F  13 0016C        BEQL    23$                               ;
                                   68  D5 0016E        TSTL    INDLVL                            ;
                                   05  12 00170        BNEQ    21$                               ;
                50                 07  D0 00172        MOVL    #7, R0                            ;
                                   03  11 00175        BRB     22$                               ;
                50                 0A  D0 00177 21$:   MOVL    #10, R0                           ;
                66                 50  D0 0017A 22$:   MOVL    R0, LINE_TYPE                     ;
                            F8  A6  DD 0017D 23$:   PUSHL   EXT_LINE_LENGTH                   ; 1768
                          FB44  C6  9F 00180        PUSHAB  RNO_LINE                          ;
                            F4  A6  DD 00184        PUSHL   INT_LINE_LENGTH                   ;
     00000000V  EF                 03  FB 00187        CALLS   #3, INS_LINE                      ;
                                       04 0018E        RET                                       ; 1770

; Routine Size:  399 bytes,    Routine Base:  $CODE$ + 04F7
```

```
1 %SBTTL 'INS_PAGE_NO -- generate page # list'
1 ROUTINE INS_PAGE_NO (LIST_HEAD) : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Generate the page number list for the index item
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       LIST_HEAD - address of page number list
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       CMDBLK          - command line information block
1 !       INDLVL          - subindex level
1 !       LSTSTK          - subindex list stack
1 !       PAGE_WIDTH      - size of output page
1 !       CHRSIZ          - TMS character size vector
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       LINE_POINTER    - pointer to next position in output line
1 !       EXT_LINE_LENGTH - external line length
1 !       INT_LINE_LENGTH - internal line length
1 !       TMS_LINE_LENGTH - TMS line length in relative units
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !
1 !       None
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         INDENT,
2         MAX_EXT_LENGTH,
2         REF_GENERATED,
2         MORE,
2         NEW_LINE,
2         CUR_BOOK,
2         XXPTR : REF $XX_BLOCK;
2
2     XXPTR = .LIST_HEAD;
2
2     NEW_LINE = FALSE;
2     CUR_BOOK = 0;
2
2     IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE
2     THEN
3         BEGIN
```

```
                !
                ! Doing an index with separate references
                !
                INDENT = .CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID];
                MAX_EXT_LENGTH = .PAGE_WIDTH;
                END
            ELSE
                BEGIN
                INDENT = 2 * .INDLVL;
                MAX_EXT_LENGTH = .CMDBLK [NDX$G_COLUMN_WID];
                END;

            DO
                BEGIN
                !
                ! Process all page references
                !
                REF_GENERATED = FALSE;

                IF .CUR_BOOK NEQ .XXPTR [XX$A_BOOK]
                THEN
                    BEGIN
                    !
                    ! Have encountered a new book name
                    !
                    ! NOTE: If you want to insert a comma after the book reference,
                    ! use .CMDBLK [NDX$V_PAGES] as the last argument to BOOK_REF
                    !
                    CUR_BOOK = .XXPTR [XX$A_BOOK];
                    BOOK_REF (.CUR_BOOK, .INDENT, .MAX_EXT_LENGTH, .NEW_LINE, FALSE);

                    IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE THEN NEW_LINE = TRUE;
                    REF_GENERATED = TRUE;
                    END;

                IF .CMDBLK [NDX$V_PAGES]
                THEN
                    BEGIN
                    !
                    ! Doing page references. Generate one.
                    !
                    SELECTONE TRUE OF
                    SET

                    [.XXPTR [XX$V_END]]:
                        !
                        ! END of page range.
                        ! These are picked up by BEGIN's so no processing is required
                        !
                        ;

                    [.XXPTR [XX$V_BEGIN]]:
                        BEGIN
                        !
                        ! BEGIN page range.
                        !
                        LOCAL
```

```
                SKIPPED_ENTRIES,
                XXEND : REF $XX_BLOCK;

            SKIPPED_ENTRIES = 0;
            PAGE_REF (.XXPTR, .INDENT, .MAX_EXT_LENGTH);    ! Write initial page
            REF_GENERATED = TRUE;

            !
            ! Find end of range and output
            !
            XXEND = .XXPTR;
            WHILE TRUE DO
                BEGIN

                IF .XXEND [XX$A_LINK] EQL 0
                THEN
                    BEGIN
                    !
                    ! No corresponding END.
                    ! Issue error message and exit loop.
                    !
%IF %BLISS (BLISS32)
%THEN                                                   ! Signal errors in BLISS32

                    SIGNAL (INDEX$_NOEND);

%ELSE                                                   ! Use $XPO_PUT_MSG otherwise

                    $XPO_PUT_MSG (SEVERITY = WARNING,
                        STRING = '.XPLUS (BEGIN) has no corresponding .XPLUS (END)');

%FI

                    ENTMSG ();

                    EXITLOOP;
                    END;

                !
                ! Point to next entry
                !
                XXEND = .XXEND [XX$A_LINK];

                IF .XXEND [XX$V_END]
                AND (.XXEND [XX$A_BOOK] EQL .XXPTR [XX$A_BOOK])
                THEN
                    BEGIN
                    !
                    ! Entry is an end range entry and the book identifiers
                    ! are the same; found the corresponding END
                    !
                    ! If BEGIN and END are on same page and have identical
                    ! display attributes, then don't generate a range.
                    !
                    XXPTR [XX$V_BEGIN] = FALSE;
                    XXEND [XX$V_END] = FALSE;
```

```
                        IF NOT SAMEPG (.XXPTR, .XXEND)
                        THEN
                            BEGIN
                            !
                            ! Not same page or display attributes not identical.
                            ! Generate page reference.
                            !
                            ! Append ' to '
                            !
                            FILL_LINE (2, CH$PTR (UPLIT ('to')), .INDENT,
                                .MAX_EXT_LENGTH, FALSE, FALSE, TRUE);

                            !
                            ! Insert end of range number
                            !
                            PAGE_REF (.XXEND, .INDENT, .MAX_EXT_LENGTH);
                            END;

                        !
                        ! Reset BEGIN and END attributes.
                        !
                        XXPTR [XX$V_BEGIN] = TRUE;
                        XXEND [XX$V_END] = TRUE;

                        EXITLOOP;                       ! Exit loop
                        END
                    ELSE
                        SKIPPED_ENTRIES = .SKIPPED_ENTRIES + 1;
                    END;

                IF .SKIPPED_ENTRIES NEQ 0
                THEN
                    BEGIN
                    !
                    ! There were entries inside a page range that were skipped.
                    ! Tell the user about it
                    !
%IF %BLISS (BLISS32)
%THEN                                               ! Signal errors in BLISS32

                    SIGNAL (INDEX$_SKIPPED, 1, .SKIPPED_ENTRIES);

%ELSE                                               ! Use $XPO_PUT_MSG otherwise

                    $XPO_PUT_MSG (SEVERITY = WARNING,
                        STRING = $STR_CONCAT ($STR_ASCII (.SKIPPED_ENTRIES),
                            ' reference(s) inside page range - ignored'));

%FI

                    ENTMSG ();                      ! Dump the entry too.
                    END;

                !
                ! NOTE: to avoid skipping entries inside a page range, remove
                ! the following line.
                !
```

```
                        XXPTR = .XXEND;
                        END;

                [OTHERWISE]:
                        BEGIN
                        !
                        ! Normal page ref
                        !
                        LOCAL
                            XXPEEK : REF $XX_BLOCK;

                        PAGE_REF (.XXPTR, .INDENT, .MAX_EXT_LENGTH); ! Insert reference into line
                        REF_GENERATED = TRUE;

                        XXPEEK = .XXPTR [XX$A_LINK];    ! Point to next entry
                        WHILE .XXPEEK NEQ 0 DO
                            BEGIN
                            !
                            ! Check for duplicate references
                            !
                            IF SAMEPG (.XXPTR, .XXPEEK)
                            THEN
                                !
                                ! Duplicate reference - skip it
                                !
                                XXPTR = .XXPEEK
                            ELSE
                                !
                                ! Not same, terminate scan
                                !
                                EXITLOOP;

                            XXPEEK = .XXPEEK [XX$A_LINK];
                            END;

                        IF .CMDBLK [NDX$V_PAGE_MERGE]
                        THEN
                            BEGIN
                            !
                            ! User selected old style page merging
                            ! (adjacent pages form a page range)
                            !
                            LOCAL
                                RANGE_SEEN;

                            RANGE_SEEN = FALSE;
                            XXPEEK = .XXPTR [XX$A_LINK];
                            WHILE .XXPEEK NEQ 0 DO
                                BEGIN

                                IF  PAGMRG (XTNPAG (.XXPTR [XX$H_PAGE]), XTNPAG (.XXPEEK [XX$H_PAGE]))
                                AND SAME_ATTR (.XXPTR, .XXPEEK)
                                THEN
                                    BEGIN
                                    !
                                    ! Found some adjacent pages with same display attributes
                                    !
```

```
8                               RANGE_SEEN = TRUE;
8                               XXPTR = .XXPEEK;
8                               END
7                           ELSE
7                               EXITLOOP;

7                           XXPEEK = .XXPEEK [XX$A_LINK];
6                           END;

6                       IF .RANGE_SEEN
6                       THEN
7                           BEGIN
7                           !
7                           ! We found a page range - write it out
7                           !
7                           FILL_LINE (2, CH$PTR (UPLIT ('to')), .INDENT, .MAX_EXT_LENGTH, FALSE, FALSE, TRUE);
7                           PAGE_REF (.XXPTR, .INDENT, .MAX_EXT_LENGTH);
6                           END;
5                       END;
4                   END;

4               TES;
3               END;

3           !
3           ! Check for more to do
3           !
3           MORE = FALSE;

3           XXPTR = .XXPTR [XX$A_LINK];
3           WHILE .XXPTR NEQ 0 DO
4               BEGIN

4               IF NOT .XXPTR [XX$V_END]
4               THEN
5                   BEGIN
5                   !
5                   ! Not a END range
5                   !
5                   MORE = TRUE;
5                   EXITLOOP;
4                   END;

4               XXPTR = .XXPTR [XX$A_LINK];
3               END;

3           IF .MORE                                            ! More to do
3           AND .CMDBLK [NDX$V_PAGES]                           ! Generating page references
3           AND .REF_GENERATED                                  ! Generated a reference
4           AND (
5               (.CMDBLK [NDX$H_LAYOUT] NEQ SEPARATE)           ! Not an index with separate references
5               OR (.CUR_BOOK EQL .XXPTR [XX$A_BOOK])           ! or next book is the same as the current one
4               )
3           THEN
4               BEGIN
4               !
4               ! Append a comma
```

```
; 1339      2113  4              !
; 1340      2114  4              CH$WCHAR_A (%C',', LINE_POINTER);
; 1341      2115  4              INT_LINE_LENGTH = .INT_LINE_LENGTH + 1;
; 1342      2116  4              EXT_LINE_LENGTH = .EXT_LINE_LENGTH + 1;
; 1343      2117  4              TMS_LINE_LENGTH = .TMS_LINE_LENGTH + .CHRSIZ [%C','];
; 1344      2118  3              END;
; 1345      2119  3
; 1346      2120  3          END
; 1347      2121  3
; 1348      2122  2      WHILE .MORE;                        ! Loop if more to do
; 1349      2123  2
; 1350      2124  1      END;                                !End of INS_PAGE_NO


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                        00  00  6F  74  003C4 P.ADQ:    .ASCII  \to\<0><0>                              ;
                        00  00  6F  74  003C8 P.ADR:    .ASCII  \to\<0><0>                              ;


                                                        .PSECT  $CODE$,NOWRT,2

                                    0FFC 00000 INS_PAGE_NO:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 1772
                 5B 00000000G   EF   9E 00002           MOVAB   CMDBLK+4, R11
                 53        04   AC   D0 00009           MOVL    LIST_HEAD, XXPTR                        ; 1820
                                59   D4 0000D           CLRL    NEW_LINE                                ; 1822
                                57   D4 0000F           CLRL    CUR_BOOK                                ; 1823
                 03        02   AB   B1 00011           CMPW    CMDBLK+6, #3                            ; 1825
                                0F   12 00015           BNEQ    1$                                      ;
       56       08   AB    0C   AB   C1 00017           ADDL3   CMDBLK+16, CMDBLK+12, INDENT            ; 1831
                 55 00000000'   EF   D0 0001D           MOVL    PAGE_WIDTH, MAX_EXT_LENGTH              ; 1832
                                0C   11 00024           BRB     2$                                      ; 1825
       56 00000000G  EF         01   78 00026 1$:       ASHL    #1, INDLVL, INDENT                      ; 1836
                 55        08   AB   D0 0002E           MOVL    CMDBLK+12, MAX_EXT_LENGTH               ; 1837
                                58   D4 00032 2$:       CLRL    REF_GENERATED                           ; 1845
             0C  A3             57   D1 00034           CMPL    CUR_BOOK, 12(XXPTR)                     ; 1847
                                21   13 00038           BEQL    4$                                      ;
                 57        0C   A3   D0 0003A           MOVL    12(XXPTR), CUR_BOOK                     ; 1856
                                7E   D4 0003E           CLRL    -(SP)                                   ; 1857
                         0220   8F   BB 00040           PUSHR   #^M<R5,R9>
                                56   DD 00044           PUSHL   INDENT
                                57   DD 00046           PUSHL   CUR_BOOK
            00000000V    EF     05   FB 00048           CALLS   #5, BOOK_REF
                 03        02   AB   B1 0004F           CMPW    CMDBLK+6, #3                            ; 1859
                                03   12 00053           BNEQ    3$                                      ;
                                59   01   D0 00055      MOVL    #1, NEW_LINE                            ;
                                58   01   D0 00058 3$:  MOVL    #1, REF_GENERATED                       ; 1860
       03       FC   AB         03   E0 0005B 4$:       BBS     #3, CMDBLK, 6$                          ; 1863
                         0150   31 00060 5$:            BRW     19$                                     ;
       F8       0A   A3         03   E0 00063 6$:       BBS     #3, 10(XXPTR), 5$                       ; 1872
       03       0A   A3         02   E0 00068           BBS     #2, 10(XXPTR), 7$                       ; 1879
                         00A8   31 0006D                BRW     14$                                     ;
                                54   D4 00070 7$:       CLRL    SKIPPED_ENTRIES                         ; 1888
                                55   DD 00072           PUSHL   MAX_EXT_LENGTH                          ; 1889
```

```
                      0048  8F   BB 00074          PUSHR   #^M<R3,R6>                          ;
        00000000V  EF       03   FB 00078          CALLS   #3, PAGE_REF                        ;
                   58       01   D0 0007F          MOVL    #1, REF_GENERATED                   ; 1890
                   52       53   D0 00082          MOVL    XXPTR, XXEND                        ; 1895
                            62   D5 00085 8$:      TSTL    (XXEND)                             ; 1899
                            16   12 00087          BNEQ    9$                                  ;
                 00000000G  8F   DD 00089          PUSHL   #DSRINDEX$_NOEND                    ; 1909
        00000000G  00       01   FB 0008F          CALLS   #1, LIB$SIGNAL                      ;
        00000000G  EF       00   FB 00096          CALLS   #0, ENTMSG                          ; 1918
                            57   11 0009D          BRB     12$                                 ; 1901
                   52       62   D0 0009F 9$:      MOVL    (XXEND), XXEND                      ; 1926
   4B       0A     A2       03   E1 000A2          BBC     #3, 10(XXEND), 11$                  ; 1928
            0C     A3  0C   A2   D1 000A7          CMPL    12(XXEND), 12(XXPTR)                ; 1929
                            44   12 000AC          BNEQ    11$                                 ;
            0A     A3       04   8A 000AE          BICB2   #4, 10(XXPTR)                       ; 1939
            0A     A2       08   8A 000B2          BICB2   #8, 10(XXEND)                       ; 1940
                            52   DD 000B6          PUSHL   XXEND                               ; 1942
                            53   DD 000B8          PUSHL   XXPTR                               ;
        00000000V  EF       02   FB 000BA          CALLS   #2, SAMEPG                          ;
                   24       50   E8 000C1          BLBS    R0, 10$                             ;
                            01   DD 000C4          PUSHL   #1                                  ; 1951
                            7E   7C 000C6          CLRQ    -(SP)                               ;
                            55   DD 000C8          PUSHL   MAX_EXT_LENGTH                      ; 1952
                            56   DD 000CA          PUSHL   INDENT                              ; 1951
                 00000000'  EF   9F 000CC          PUSHAB  P.ADQ                               ;
                            02   DD 000D2          PUSHL   #2                                  ;
        00000000V  EF       07   FB 000D4          CALLS   #7, FILL_LINE                       ;
                            55   DD 000DB          PUSHL   MAX_EXT_LENGTH                      ; 1957
                      0044  8F   BB 000DD          PUSHR   #^M<R2,R6>                          ;
        00000000V  EF       03   FB 000E1          CALLS   #3, PAGE_REF                        ;
            0A     A3       04   88 000E8 10$:     BISB2   #4, 10(XXPTR)                       ; 1963
            0A     A2       08   88 000EC          BISB2   #8, 10(XXEND)                       ; 1964
                            04   11 000F0          BRB     12$                                 ; 1931
                            54   D6 000F2 11$:     INCL    SKIPPED_ENTRIES                     ; 1969
                            8F   11 000F4          BRB     8$                                  ; 1896
                            54   D5 000F6 12$:     TSTL    SKIPPED_ENTRIES                     ; 1972
                            18   13 000F8          BEQL    13$                                 ;
                            54   DD 000FA          PUSHL   SKIPPED_ENTRIES                     ; 1982
                            01   DD 000FC          PUSHL   #1                                  ;
                 00000000G  8F   DD 000FE          PUSHL   #DSRINDEX$_SKIPPED                  ;
        00000000G  00       03   FB 00104          CALLS   #3, LIB$SIGNAL                      ;
        00000000G  EF       00   FB 0010B          CALLS   #0, ENTMSG                          ; 1992
                   53       52   D0 00112 13$:     MOVL    XXEND, XXPTR                        ; 1999
                          009B   31 00115          BRW     19$                                 ; 1869
                            55   DD 00118 14$:     PUSHL   MAX_EXT_LENGTH                      ; 2010
                      0048  8F   BB 0011A          PUSHR   #^M<R3,R6>                          ;
        00000000V  EF       03   FB 0011E          CALLS   #3, PAGE_REF                        ;
                   58       01   D0 00125          MOVL    #1, REF_GENERATED                   ; 2011
                   52       63   D0 00128          MOVL    (XXPTR), XXPEEK                     ; 2013
                            16   13 0012B 15$:     BEQL    16$                                 ; 2014
                            52   DD 0012D          PUSHL   XXPEEK                              ; 2019
                            53   DD 0012F          PUSHL   XXPTR                               ;
        00000000V  EF       02   FB 00131          CALLS   #2, SAMEPG                          ;
                   08       50   E9 00138          BLBC    R0, 16$                             ;
                   53       52   D0 0013B          MOVL    XXPEEK, XXPTR                       ; 2024
                   52       62   D0 0013E          MOVL    (XXPEEK), XXPEEK                    ; 2031
                            E8   11 00141          BRB     15$                                 ; 2014
```

```
     6B        FD    AB              03   E1 00143 16$:   BBC     #3, CMDBLK+1, 19$                ; 2034
                                     54   D4 00148        CLRL    RANGE_SEEN                       ; 2044
                     52              63   D0 0014A        MOVL    (XXPTR), XXPEEK                  ; 2045
                                     3D   13 0014D 17$:   BEQL    18$                              ; 2046
                     7E        08    A2   32 0014F        CVTWL   8(XXPEEK), -(SP)                 ; 2049
          00000000G  EF              01   FB 00153        CALLS   #1, XTNPAG                       ;
                                     50   DD 0015A        PUSHL   R0                               ;
                     7E        08    A3   32 0015C        CVTWL   8(XXPTR), -(SP)                  ;
          00000000G  EF              01   FB 00160        CALLS   #1, XTNPAG                       ;
                                     50   DD 00167        PUSHL   R0                               ;
          00000000G  EF              02   FB 00169        CALLS   #2, PAGMRG                       ;
                     19              50   E9 00170        BLBC    R0, 18$                          ;
                                     52   DD 00173        PUSHL   XXPEEK                           ; 2050
                                     53   DD 00175        PUSHL   XXPTR                            ;
          00000000V  EF              02   FB 00177        CALLS   #2, SAME_ATTR                    ;
                     0B              50   E9 0017E        BLBC    R0, 18$                          ;
                     54              01   D0 00181        MOVL    #1, RANGE_SEEN                   ; 2056
                     53              52   D0 00184        MOVL    XXPEEK, XXPTR                    ; 2057
                     52              62   D0 00187        MOVL    (XXPEEK), XXPEEK                 ; 2062
                                     C1   11 0018A        BRB     17$                              ; 2046
                     24              54   E9 0018C 18$:   BLBC    RANGE_SEEN, 19$                  ; 2065
                                     01   DD 0018F        PUSHL   #1                               ; 2071
                                     7E   7C 00191        CLRQ    -(SP)                            ;
                                     55   DD 00193        PUSHL   MAX_EXT_LENGTH                   ;
                                     56   DD 00195        PUSHL   INDENT                           ;
                          00000000'  EF   9F 00197        PUSHAB  P.ADR                            ;
                                     02   DD 0019D        PUSHL   #2                               ;
          00000000V  EF              07   FB 0019F        CALLS   #7, FILL_LINE                    ;
                                     55   DD 001A6        PUSHL   MAX_EXT_LENGTH                   ; 2072
                               0048  8F   BB 001A8        PUSHR   #^M<R3,R6>                       ;
          00000000V  EF              03   FB 001AC        CALLS   #3, PAGE_REF                     ;
                                     5A   D4 001B3 19$:   CLRL    MORE                             ; 2083
                     53              63   D0 001B5 20$:   MOVL    (XXPTR), XXPTR                   ; 2085
                                     08   13 001B8        BEQL    21$                              ; 2086
     F6        0A    A3              03   E0 001BA        BBS     #3, 10(XXPTR), 20$               ; 2089
                     5A              01   D0 001BF        MOVL    #1, MORE                         ; 2095
                     43              5A   E9 001C2 21$:   BLBC    MORE, 24$                        ; 2102
     38        FC    AB              03   E1 001C5        BBC     #3, CMDBLK, 23$                  ; 2103
                     35              58   E9 001CA        BLBC    REF_GENERATED, 23$               ; 2104
                     03        02    AB   B1 001CD        CMPW    CMDBLK+6, #3                     ; 2106
                                     06   12 001D1        BNEQ    22$                              ;
                     0C    A3        57   D1 001D3        CMPL    CUR_BOOK, 12(XXPTR)              ; 2107
                                     29   12 001D7        BNEQ    23$                              ;
          00000000'  FF              2C   90 001D9 22$:   MOVB    #44, @LINE_POINTER               ; 2114
                          00000000'  EF   D6 001E0        INCL    LINE_POINTER                     ;
                          00000000'  EF   D6 001E6        INCL    INT_LINE_LENGTH                  ; 2115
                          00000000'  EF   D6 001EC        INCL    EXT_LINE_LENGTH                  ; 2116
                     50 00000000G    EF   D0 001F2        MOVL    CHRSIZ, R0                       ; 2117
          00000000'  EF        00B0  C0   C0 001F9        ADDL2   176(R0), TMS_LINE_LENGTH         ;
                     03              5A   E9 00202 23$:   BLBC    MORE, 24$                        ; 2122
                               FE2A  31 00205             BRW     2$                               ;
                                     04 00208 24$:        RET                                      ; 2124

; Routine Size:  521 bytes,    Routine Base:  $CODE$ + 0686
```

```
%SBTTL 'BOOK_REF -- Generate a book reference for master index'
ROUTINE BOOK_REF (BOOK, INDENT, MAX_EXT, NEW_LINE, COMMA) : NOVALUE =
!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine inserts a book reference into RNO_LINE
!
! FORMAL PARAMETERS:
!
!       BOOK      - Address of book name string
!       INDENT    - Number of blanks to indent a new line
!       MAX_EXT   - Maximum external line length
!       NEW_LINE- Put book reference on a new line
!       COMMA     - Append a comma to the book name
!
! IMPLICIT INPUTS:
!
!       LINE_POINTER    - Pointer to RNO_LINE
!       EXT_LINE_LENGTH - External line length
!       INT_LINE_LENGTH - Internal line length
!       INDLVL          - Current index level
!       CMDBLK          - Command line information block
!       PAGE_WIDTH      - Output page width
!       TMSSTD          - Standard character size for TMS
!
! IMPLICIT OUTPUTS:
!
!       LINE_POINTER    - pointer to next position in output line
!       EXT_LINE_LENGTH - external line length
!       INT_LINE_LENGTH - internal line length
!       TMS_LINE_LENGTH - length of line in TMS relative units
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
! SIDE EFFECTS:
!
!       None
!--
    BEGIN
    LOCAL
        LEADING_BLANK,
        STR_VEC : REF VECTOR,
        PTR,
        LEN;

    LEADING_BLANK = TRUE;

    IF .NEW_LINE
    THEN
        BEGIN
        !
        ! Flush current line before inserting reference
        !
```

```
3            INS_LINE (.INT_LINE_LENGTH, CH$PTR (RNO_LINE), .EXT_LINE_LENGTH);
3            LINE_TYPE = (IF .INDLVL EQL 0 THEN ENTRY_W ELSE SUB_W);
3
3            INDENT_LINE (.INDENT);
3
3            LEADING_BLANK = FALSE;
3            END
2        ELSE
2            IF .CMDBLK [NDX$H_LAYOUT] EQL SEPARATE
2            THEN
3                BEGIN
3                !
3                ! Separate layout. Pad line to appropriate length.
3                ! NOTE: Separate layout is not allowed for TEX format.
3                !
3                LEN = .CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID] - .EXT_LINE_LENGTH;
3
3                IF .LEN GTR 0
3                THEN
4                    BEGIN
4                    !
4                    ! NOTE: if LEN LEQ 0 then line was already padded or overflowed.
4                    ! In any case we don't pad it cause we'll blow up with the
4                    ! negative count.
4                    !
4                    EXT_LINE_LENGTH = .EXT_LINE_LENGTH + .LEN;
4
5                    IF (.CMDBLK [NDX$H_FORMAT] EQL TMS11_A)
5                    OR (.CMDBLK [NDX$H_FORMAT] EQL TMS11_E)
4                    THEN
5                        BEGIN
5                        !
5                        ! TMS11 output (line is padded with a tabs)
5                        !
5                        CH$WCHAR_A (TAB, LINE_POINTER);
5                        CH$WCHAR_A (TAB, LINE_POINTER);
5                        INT_LINE_LENGTH = .INT_LINE_LENGTH + 2;
5                        TMS_LINE_LENGTH = (.CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID]) * TMSSTD;
5                        END
4                    ELSE
5                        BEGIN
5                        !
5                        ! RUNOFF output (pad line with spaces)
5                        !
5                        INT_LINE_LENGTH = .INT_LINE_LENGTH + .LEN;
5                        CH$FILL (%C' ', .LEN, .LINE_POINTER);
5                        LINE_POINTER = CH$PLUS (.LINE_POINTER, .LEN);
5                        END;
4                    END;
3
3                LEADING_BLANK = FALSE;
3                END;
2
2        !
2        ! Get pointer to book and length of book name
2        !
2        STR_VEC = .BOOK;
```

```
2       LEN = .STR_VEC [0];
2       PTR = CH$PTR (STR_VEC [1]);
2
2       !
2       ! Insert book name into line
2       !
2       FILL_LINE (.LEN, .PTR, .INDENT, .MAX_EXT, FALSE, TRUE, .LEADING_BLANK);
2
2       !
2       ! Append comma to end of reference if necessary
2       !
2       IF .COMMA
2       THEN
3           BEGIN
3           CH$WCHAR_A (%C',', LINE_POINTER);
3           INT_LINE_LENGTH = .INT_LINE_LENGTH + 1;
3           EXT_LINE_LENGTH = .EXT_LINE_LENGTH + 1;
3           TMS_LINE_LENGTH = .TMS_LINE_LENGTH + .CHRSIZ [%C','];
2           END;
2
1       END;
```

```
                                  03FC 00000 BOOK_REF:
                                                      .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9              ; 2126
             59 00000000G   EF  9E 00002              MOVAB   CMDBLK+4, R9                              ;
             58 00000000'   EF  9E 00009              MOVAB   LINE_POINTER, R8                          ;
             57                 01  D0 00010          MOVL    #1, LEADING_BLANK                         ; 2174
             33          10 AC  E9 00013              BLBC    NEW_LINE, 3$                              ; 2176
                       04B8 C8  DD 00017              PUSHL   EXT_LINE_LENGTH                           ; 2182
                         04 A8  9F 0001B              PUSHAB  RNO_LINE                                  ;
                       04B4 C8  DD 0001E              PUSHL   INT_LINE_LENGTH                           ;
        00000000V  EF       03  FB 00022              CALLS   #3, INS_LINE                              ;
                   00000000G EF D5 00029              TSTL    INDLVL                                    ; 2183
                            05  12 0002F              BNEQ    1$                                        ;
             50             06  D0 00031              MOVL    #6, R0                                    ;
                            03  11 00034              BRB     2$                                        ;
             50             09  D0 00036 1$:          MOVL    #9, R0                                    ;
          04C0 C8           50  D0 00039 2$:          MOVL    R0, LINE_TYPE                             ;
                         08 AC  DD 0003E              PUSHL   INDENT                                    ; 2185
        00000000V  EF       01  FB 00041              CALLS   #1, INDENT_LINE                           ;
                            51  11 00048              BRB     6$                                        ; 2187
          03        02 A9   B1 0004A 3$:              CMPW    CMDBLK+6, #3                              ; 2190
                            4D  12 0004E              BNEQ    7$                                        ;
  50      08 A9          0C A9  C1 00050              ADDL3   CMDBLK+16, CMDBLK+12, R0                  ; 2197
  56         50        04B8 C8  C3 00056              SUBL3   EXT_LINE_LENGTH, R0, LEN                  ;
                            3D  15 0005C              BLEQ    6$                                        ; 2199
          04B8 C8           56  C0 0005E              ADDL2   LEN, EXT_LINE_LENGTH                      ; 2207
             02             69  B1 00063              CMPW    CMDBLK+4, #2                              ; 2209
                            05  13 00066              BEQL    4$                                        ;
             03             69  B1 00068              CMPW    CMDBLK+4, #3                              ; 2210
                            1F  12 0006B              BNEQ    5$                                        ;
          00 B8         00G 8F  90 0006D 4$:          MOVB    #TAB, @LINE_POINTER                       ; 2216
                            68  D6 00072              INCL    LINE_POINTER                              ;
```

```
                 00   B8      00G   8F  90 00074        MOVB   #TAB, @LINE_POINTER                        ; 2217
                                    68  D6 00079        INCL   LINE_POINTER                               ;
               04B4   C8            02  C0 0007B        ADDL2  #2, INT_LINE_LENGTH                        ; 2218
       04BC   C8      50 00000000G  8F  C5 00080        MULL3  #TMSSTD, R0, TMS_LINE_LENGTH               ; 2219
                                    0F  11 0008A        BRB    6$                                         ; 2209
               04B4   C8            56  C0 0008C 5$:    ADDL2  LEN, INT_LINE_LENGTH                       ; 2226
   56          20                   6E  00 2C 00091     MOVC5  #0, (SP), #32, LEN, @LINE_POINTER          ; 2227
                              00    B8     00096
                      68            56  C0 00098        ADDL2  LEN, LINE_POINTER                          ; 2228
                                    57  D4 0009B 6$:    CLRL   LEADING_BLANK                              ; 2232
                      50      04    AC  D0 0009D 7$:    MOVL   BOOK, STR_VEC                              ; 2238
                      56            80  D0 000A1        MOVL   (STR_VEC)+, LEN                            ; 2239
                                    57  DD 000A4        PUSHL  LEADING_BLANK                              ; 2245
                                    01  DD 000A6        PUSHL  #1                                         ;
                                    7E  D4 000A8        CLRL   -(SP)                                      ;
                      7E      08    AC  7D 000AA        MOVQ   INDENT, -(SP)                              ;
                                    50  DD 000AE        PUSHL  PTR                                        ;
                                    56  DD 000B0        PUSHL  LEN                                        ;
          00000000V   EF            07  FB 000B2        CALLS  #7, FILL_LINE                              ;
                      1C      14    AC  E9 000B9        BLBC   COMMA, 8$                                  ; 2250
                 00   B8            2C  90 000BD        MOVB   #44, @LINE_POINTER                         ; 2253
                                    68  D6 000C1        INCL   LINE_POINTER                               ;
                            04B4    C8  D6 000C3        INCL   INT_LINE_LENGTH                            ; 2254
                            04B8    C8  D6 000C7        INCL   EXT_LINE_LENGTH                            ; 2255
                      50 00000000G  EF  D0 000CB        MOVL   CHRSIZ, R0                                 ; 2256
               04BC   C8    00B0    C0  C0 000D2        ADDL2  176(R0), TMS_LINE_LENGTH                   ;
                                    04 000D9 8$:        RET                                               ; 2259

; Routine Size:  218 bytes,    Routine Base:  $CODE$ + 088F
```

```
1 %SBTTL 'PAGE_REF -- Format a page reference'
1 ROUTINE PAGE_REF (XX_ENTRY, INDENT, MAX_EXT) : NOVALUE =
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine formats a page reference and inserts it into RNO_WORD
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       XX_ENTRY - Pointer to reference to be formatted
1 !       INDENT   - Number of blanks to indent new line if reference doesn't fit
1 !       MAX_EXT  - Maximum external length of line before wrapping
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       CMDBLK  - Command line information block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !
1 !       None
1 !--
2     BEGIN
2     MAP
2         XX_ENTRY : REF $XX_BLOCK;
2
2     LOCAL
2         REFBUF    : VECTOR [CH$ALLOCATION (100)],
2         PTR,
2         LEN,
2         PAGE : REF PAGE_DEFINITION;
2
2     PAGE = XTNPAG (.XX_ENTRY [XX$H_PAGE]);
2
2     PTR = CH$PTR (REFBUF);
2
2     IF .CMDBLK [NDX$V_STANDARD_PAGE]
2     THEN
2         !
2         ! User wants traditional page numbers
2         !
2         LEN = PACPAG (.PAGE, PTR)
2     ELSE
2         !
2         ! User wants running pages
2         !
2         LEN = PACBAS (.PAGE [SCT_RUN_PAGE], PTR, 10);
2
```

```
; 1545    2317  2     IF .XX_ENTRY [XX$A_APPEND] NEQ 0
; 1546    2318  2     THEN
; 1547    2319  3         BEGIN
; 1548    2320  3         !
; 1549    2321  3         ! Have appended text
; 1550    2322  3         !
; 1551    2323  3         LOCAL
; 1552    2324  3             A_VEC : REF VECTOR,
; 1553    2325  3             A_PTR,
; 1554    2326  3             A_LEN;
; 1555    2327  3
; 1556    2328  3         A_VEC = .XX_ENTRY [XX$A_APPEND];
; 1557    2329  3         A_LEN = .A_VEC [0];
; 1558    2330  3         A_PTR = CH$PTR (A_VEC [1]);
; 1559    2331  3
; 1560    2332  3         CH$MOVE (.A_LEN, .A_PTR, .PTR);
; 1561    2333  3         LEN = .LEN + .A_LEN;
; 1562    2334  2         END;
; 1563    2335  2
; 1564    2336  2     !
; 1565    2337  2     ! Insert reference into line
; 1566    2338  2     !
; 1567    2339  2     FILL_LINE (.LEN, CH$PTR (REFBUF), .INDENT, .MAX_EXT, .XX_ENTRY [XX$V_BOLD], .XX_ENTRY [XX$V_UNDERLINE],
; 1568    2340  2
; 1569    2341  1     END;


                                          01FC 00000 PAGE_REF:
                                                          .WORD   Save R2,R3,R4,R5,R6,R7,R8              ; 2261
                        5E       98   AE   9E 00002       MOVAB   -104(SP), SP                           :
                        56       04   AC   D0 00006       MOVL    XX_ENTRY, R6                           ; 2301
                        7E       08   A6   32 0000A       CVTWL   8(R6), -(SP)                           :
            00000000G   EF            01   FB 0000E       CALLS   #1, XTNPAG                             :
                        52            50   D0 00015       MOVL    R0, PAGE                               :
                        6E       04   AE   9E 00018       MOVAB   REFBUF, PTR                            ; 2303
         0D 00000000G   EF            05   E1 0001C       BBC     #5, CMDBLK, 1$                         ; 2305
                               4004   8F   BB 00024       PUSHR   #^M<R2,SP>                             ; 2310
            00000000G   EF            02   FB 00028       CALLS   #2, FACPAG                             :
                                      10   11 0002F       BRB     2$                                     :
                                      0A   DD 00031 1$:   PUSHL   #10                                    ; 2315
                                 04   AE   9F 00033       PUSHAB  PTR                                    :
                        7E       0E   A2   3C 00036       MOVZWL  14(PAGE), -(SP)                        :
            00000000G   EF            03   FB 0003A       CALLS   #3, PACBAS                             :
                        58            50   D0 00041 2$:   MOVL    R0, LEN                                :
                                 04   A6   D5 00044       TSTL    4(R6)                                  ; 2317
                                      0F   13 00047       BEQL    3$                                     :
                        50       04   A6   D0 00049       MOVL    4(R6), A_VEC                           ; 2328
                        57            80   D0 0004D       MOVL    (A_VEC)+, A_LEN                        ; 2329
             00   BE    60            57   28 00050       MOVC3   A_LEN, (A_PTR), @PTR                   ; 2332
                        58            57   C0 00055       ADDL2   A_LEN, LEN                             ; 2333
                                      01   DD 00058 3$:   PUSHL   #1                                     ; 2339
    7E       0A   A6    01            01   EF 0005A       EXTZV   #1, #1, 10(R6), -(SP)                  :
    7E       0A   A6    01            00   EF 00060       EXTZV   #0, #1, 10(R6), -(SP)                  :
                        7E       08   AC   7D 00066       MOVQ    INDENT, -(SP)                          :
```

```
                               18    AE  9F 0006A       PUSHAB  REFBUF                     ;
                                     58  DD 0006D       PUSHL   LEN                        ;
               00000000V  EF         07  FB 0006F       CALLS   #7, FILL_LINE              ;
                                         04 00076       RET                                ; 2341

; Routine Size:  119 bytes,    Routine Base:  $CODE$ + 0969
```

```
2342  1 %SBTTL 'SAMEPG -- Compare page references'
2343  1 ROUTINE SAMEPG (X1, X2) =
2344  1 !++
2345  1 !
2346  1 ! FUNCTIONAL DESCRIPTION:
2347  1 !
2348  1 !       This routine compares two page references to see if they are equal.
2349  1 !
2350  1 ! FORMAL PARAMETERS:
2351  1 !
2352  1 !       X1 - Pointer to first page reference
2353  1 !       X2 - Pointer to second page reference
2354  1 !
2355  1 ! IMPLICIT INPUTS:
2356  1 !
2357  1 !       None
2358  1 !
2359  1 ! IMPLICIT OUTPUTS:
2360  1 !
2361  1 !       None
2362  1 !
2363  1 ! ROUTINE VALUE:
2364  1 ! COMPLETION CODES:
2365  1 !
2366  1 !       Returns TRUE if references are equal, FALSE otherwise
2367  1 !
2368  1 ! SIDE EFFECTS:
2369  1 !
2370  1 !       None
2371  1 !--
2372  2     BEGIN
2373  2     MAP
2374  2         X1 : REF $XX_BLOCK,
2375  2         X2 : REF $XX_BLOCK;
2376  2
2377  2     IF PAGEQL (XTNPAG (.X1 [XX$H_PAGE]), XTNPAG (.X2 [XX$H_PAGE]), TRUE)
2378  2     THEN
2379  2         !
2380  2         ! Pages are identical in the RUNOFF sense.
2381  2         ! Compare reference attributes
2382  2         !
2383  2         RETURN SAME_ATTR (.X1, .X2)
2384  2     ELSE
2385  2         !
2386  2         ! Pages not equal in RUNOFF sense
2387  2         !
2388  2         RETURN FALSE;
2389  2
2390  1     END;


                         001C 00000 SAMEPG: .WORD    Save R2,R3,R4          ; 2343
            54 00000000G  EF   9E 00002         MOVAB    XTNPAG, R4             ;
                           01   DD 00009         PUSHL    #1                     ; 2377
```

```
           53    08  AC  D0 0000B        MOVL    X2, R3                    ;
           7E    08  A3  32 0000F        CVTWL   8(R3), -(SP)              ;
           64        01  FB 00013        CALLS   #1, XTNPAG                ;
                     50  DD 00016        PUSHL   R0                        ;
           52    04  AC  D0 00018        MOVL    X1, R2                    ;
           7E    08  A2  32 0001C        CVTWL   8(R2), -(SP)              ;
           64        01  FB 00020        CALLS   #1, XTNPAG                ;
                     50  DD 00023        PUSHL   R0                        ;
00000000G  EF        03  FB 00025        CALLS   #3, PAGEQL                ;
           0A        50  E9 0002C        BLBC    R0, 1$                    ;
                     0C  BB 0002F        PUSHR   #^M<R2,R3>                ; 2383
00000000V  EF        02  FB 00031        CALLS   #2, SAME_ATTR             ;
                         04 00038        RET                               ; 2388
                 50  D4 00039 1$:        CLRL    R0                        ;
                         04 0003B        RET                               ; 2390

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 09E0
```

```
1 %SBTTL 'SAME_ATTR - Check page references for same attributes'
1 ROUTINE SAME_ATTR (X1, X2) =
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine compares the display attributes of two page references
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       X1      - Pointer to first page reference
1 !       X2      - Pointer to second page reference
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       None
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       TRUE if references have the same display attributes, FALSE otherwise
1 !
1 ! SIDE EFFECTS:
1 !
1 !       None
1 !--
2     BEGIN
2     MAP
2         X1 : REF $XX_BLOCK,
2         X2 : REF $XX_BLOCK;
2
2     SELECTONE TRUE OF
2     SET
2
2     [(.X1 [XX$A_APPEND] NEQ 0) AND (.X2 [XX$A_APPEND] NEQ 0)]:
3         BEGIN
3         !
3         ! Both have appended text. Compare strings
3         !
3         LOCAL
3             STR_VEC : REF VECTOR,
3             L1,
3             P1,
3             L2,
3             P2;
3
3         STR_VEC = .X1 [XX$A_APPEND];
3         L1 = .STR_VEC [0];
3         P1 = CH$PTR (STR_VEC [1]);
3
3         STR_VEC = .X2 [XX$A_APPEND];
3         L2 = .STR_VEC [0];
3         P2 = CH$PTR (STR_VEC [1]);
```

```
            IF CH$NEQ (.L1, .P1, .L2, .P2) THEN RETURN FALSE;
            END;

        [(.X1 [XX$A_APPEND] EQL 0) AND (.X2 [XX$A_APPEND] EQL 0)]:
            !
            ! Neither have appended strings. Do nothing.
            !
            ;

        [OTHERWISE]:
            !
            ! One has appended text and the other doesn't.
            ! References are not identical
            !
            RETURN FALSE;

        TES;

        RETURN (
                (.X1 [XX$V_BOLD]          EQL .X2 [XX$V_BOLD])
            AND (.X1 [XX$V_UNDERLINE]     EQL .X2 [XX$V_UNDERLINE])
            AND (.X1 [XX$V_BEGIN]         EQL .X2 [XX$V_BEGIN])
            AND (.X1 [XX$V_END]           EQL .X2 [XX$V_END])
            AND (.X1 [XX$A_BOOK]          EQL .X2 [XX$A_BOOK])
            );

        END;
```

```
                                  007C 00000 SAME_ATTR:
                                                   .WORD   Save R2,R3,R4,R5,R6                    ; 2392
                      54     04  AC  D0 00002      MOVL    X1, R4                                 ; 2429
                      52     04  A4  D0 00006      MOVL    4(R4), R2
                                 53  D4 0000A      CLRL    R3
                                 52  D5 0000C      TSTL    R2
                                 02  13 0000E      BEQL    1$
                                 53  D6 00010      INCL    R3
                      50     08  AC  D0 00012 1$:  MOVL    X2, R0
                                 51  D4 00016      CLRL    R1
                             04  A0  D5 00018      TSTL    4(R0)
                                 02  13 0001B      BEQL    2$
                                 51  D6 0001D      INCL    R1
                      55         53  D2 0001F 2$:  MCOML   R3, R5
                      51         55  CA 00022      BICL2   R5, R1
                      01         51  D1 00025      CMPL    R1, #1
                                 1D  12 00028      BNEQ    3$
                      50         52  D0 0002A      MOVL    R2, STR_VEC                            ; 2441
                      53         60  D0 0002D      MOVL    (STR_VEC), L1                          ; 2442
                      51     04  A0  9E 00030      MOVAB   4(R0), P1                              ; 2443
                      52     08  AC  D0 00034      MOVL    X2, R2                                 ; 2445
                      50     04  A2  D0 00038      MOVL    4(R2), STR_VEC
                      52         80  D0 0003C      MOVL    (STR_VEC)+, L2                         ; 2446
  52        00        61         53  2D 0003F      CMPC5   L1, (P1), #0, L2, (P2)                 ; 2449
```

```
                                    60      00044
                                    1E   11 00045         BRB     6$
                                    53   D4 00047 3$:     CLRL    R3                              ; 2452
                                    52   D5 00049         TSTL    R2
                                    02   12 0004B         BNEQ    4$
                                    53   D6 0004D         INCL    R3
                   50         08    AC   D0 0004F 4$:     MOVL    X2, R0
                                    51   D4 00053         CLRL    R1
                              04    A0   D5 00055         TSTL    4(R0)
                                    02   12 00058         BNEQ    5$
                                    51   D6 0005A         INCL    R1
                   55               53   D2 0005C 5$:     MCOML   R3, R5
                   51               55   CA 0005F         BICL2   R5, R1
                   01               51   D1 00062         CMPL    R1, #1
                                    6A   12 00065 6$:     BNEQ    12$
                   50         08    AC   D0 00067         MOVL    X2, R0                          ; 2468
                                    51   D4 0006B         CLRL    R1
     52      0A    A0         0A    A4   8D 0006D         XORB3   10(R4), 10(R0), R2
                   04               52   E8 00073         BLBS    R2, 7$
                                    02   12 00076         BNEQ    7$
                                    51   D6 00078         INCL    R1
                                    52   D4 0007A 7$:     CLRL    R2                              ; 2469
     53      0A    A0         0A    A4   8D 0007C         XORB3   10(R4), 10(R0), R3
     04            53               01   E0 00082         BBS     #1, R3, 8$
                                    02   12 00086         BNEQ    8$
                                    52   D6 00088         INCL    R2
                   55               51   D2 0008A 8$:     MCOML   R1, R5
                   52               55   CA 0008D         BICL2   R5, R2
                                    51   D4 00090         CLRL    R1                              ; 2470
     53      0A    A0         0A    A4   8D 00092         XORB3   10(R4), 10(R0), R3
     04            53               02   E0 00098         BBS     #2, R3, 9$
                                    02   12 0009C         BNEQ    9$
                                    51   D6 0009E         INCL    R1
                   56               52   D2 000A0 9$:     MCOML   R2, R6
                   51               56   CA 000A3         BICL2   R6, R1
                                    52   D4 000A6         CLRL    R2                              ; 2471
     53      0A    A0         0A    A4   8D 000A8         XORB3   10(R4), 10(R0), R3
     04            53               03   E0 000AE         BBS     #3, R3, 10$
                                    02   12 000B2         BNEQ    10$
                                    52   D6 000B4         INCL    R2
                   55               51   D2 000B6 10$:    MCOML   R1, R5
                   52               55   CA 000B9         BICL2   R5, R2
                                    51   D4 000BC         CLRL    R1                              ; 2472
             0C    A0         0C    A4   D1 000BE         CMPL    12(R4), 12(R0)
                                    02   12 000C3         BNEQ    11$
                                    51   D6 000C5         INCL    R1
                   53               52   D2 000C7 11$:    MCOML   R2, R3
                   51               53   CA 000CA         BICL2   R3, R1
                   50               51   D0 000CD         MOVL    R1, R0                          ; 2467
                                         04 000D0         RET
                                    50   D4 000D1 12$:    CLRL    R0                              ; 2475
                                         04 000D3         RET

; Routine Size:  212 bytes,    Routine Base:  $CODE$ + 0A1C
```

```
1 %SBTTL 'INS_LINE -- Insert line into page'
1 ROUTINE INS_LINE (INT_LEN, PTR, EXT_LEN) : NOVALUE =
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine inserts a line into the print page
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       INT_LEN         - Internal line length
1 !       PTR             - CH$PTR to text
1 !       EXT_LEN         - External line length
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       ALLOWD          - Maximum number of lines allowed on page
1 !       LCOUNT          - Left column line count
1 !       RCOUNT          - Right column line count
1 !       CMDBLK          - Command line information block
1 !       DOING_LAST      - TRUE if LAST_LINE was called to process the last
1 !                         line in the column
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       LCOUNT          - Incremented if line inserted into left column
1 !       LLINES          - If line inserted into left column
1 !       LTYPE           - Type of line if inserted into left column
1 !
1 !       RCOUNT          - Incremented if line inserted into right column
1 !       RLINES          - If line inserted into right column
1 !       RTYPE           - Type of line if inserted into right column
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !
1 !       None
1 !--
2     BEGIN
2
2     SELECTONE .CMDBLK [NDX$H_LAYOUT] OF
2         SET
2
2         [TWO_COLUMN]:
2             !
2             ! Two column output
2             !
2             IF .LCOUNT EQL .ALLOWD
2             THEN
3                 BEGIN
3                 !
3                 ! Filling right column
3                 !
```

```
          IF (.RCOUNT EQL 0)
          THEN
              BEGIN
              !
              ! First line in column
              !
              IF CH$EQL (1, .BLANKS, .INT_LEN, .PTR, %C' ') THEN RETURN;

              IF .CMDBLK [NDX$V_CONTINUATION]
              THEN
                  BEGIN
                  !
                  ! Generate continuation heading
                  !
                  DO_CONTINUE (.INT_LEN, .PTR, .EXT_LEN);
                  RETURN;
                  END;
              END;

          IF (.RCOUNT EQL .ALLOWD - 1) AND NOT .DOING_LAST
          THEN
              !
              ! Insert last line in column
              !
              LAST_LINE (.INT_LEN, .PTR, .EXT_LEN)
          ELSE
              BEGIN
              !
              ! Insert normal line
              !
              RCOUNT = .RCOUNT + 1;
              RTYPE [.RCOUNT] = .LINE_TYPE;
              PADLIN (.INT_LEN, .PTR, .EXT_LEN, RLINES [.RCOUNT, 0,0,0,0]);
              SAVE_TELLTALE_HEADING (RLINES [0, 0,0,0,0], RTYPE [0], .RCOUNT);

              IF .RCOUNT EQL .ALLOWD THEN PUTPAG (FALSE);
              END;
          END
      ELSE
          BEGIN
          !
          ! Left column
          !
          IF (.LCOUNT EQL 0)
          THEN
              BEGIN
              !
              ! First line in left column
              !
              IF CH$EQL (1, .BLANKS, .INT_LEN, .PTR, %C' ') THEN RETURN;

              IF .CMDBLK [NDX$V_CONTINUATION]
              THEN
                  BEGIN
                  !
                  ! Generate continuation heading
                  !
```

```
                        DO CONTINUE (.INT_LEN, .PTR, .EXT_LEN);
                        RETURN;
                        END;
                    END;

                IF (.LCOUNT EQL .ALLOWD - 1) AND NOT .DOING_LAST
                THEN
                    !
                    ! Insert last line in left column
                    !
                    LAST_LINE (.INT_LEN, .PTR, .EXT_LEN)
                ELSE
                    BEGIN
                    !
                    ! Insert a normal line
                    !
                    LCOUNT = .LCOUNT + 1;
                    LTYPE [.LCOUNT] = .LINE_TYPE;
                    PADLIN (.INT_LEN, .PTR, .EXT_LEN, LLINES [.LCOUNT, 0,0,0,0]);
                    SAVE_TELLTALE_HEADING (LLINES [0, 0,0,0,0], LTYPE [0], .LCOUNT);
                    END;

                END;

        [OTHERWISE]:
            BEGIN
            !
            ! One column output
            !

            IF (.LCOUNT EQL 0)
            THEN
                BEGIN
                !
                ! First line in column
                !
                IF CH$EQL (1, .BLANKS, .INT_LEN, .PTR, %C' ') THEN RETURN;

                IF .CMDBLK [NDX$V_CONTINUATION]
                THEN
                    BEGIN
                    !
                    ! Generate continuation heading
                    !
                    DO CONTINUE (.INT_LEN, .PTR, .EXT_LEN);
                    RETURN;
                    END;
                END;

            IF (.LCOUNT EQL .ALLOWD - 1) AND NOT .DOING_LAST
            THEN
                !
                ! Insert last line in column
                !
                LAST_LINE (.INT_LEN, .PTR, .EXT_LEN)
            ELSE
                BEGIN
```

```
; 1878          2647 4                  !
; 1879          2648 4                  ! Insert a normal line
; 1880          2649 4                  !
; 1881          2650 4                  LCOUNT = .LCOUNT + 1;
; 1882          2651 4                  LTYPE [.LCOUNT] = .LINE_TYPE;
; 1883          2652 4                  PADLIN (.INT_LEN, .PTR, .EXT_LEN, LLINES [.LCOUNT, 0,0,0,0]);
; 1884          2653 4                  SAVE_TELLTALE_HEADING (LLINES [0, 0,0,0,0], LTYPE [0], .LCOUNT);
; 1885          2654 4
; 1886          2655 4                  IF .LCOUNT EQL .ALLOWD THEN PUTPAG (FALSE);
; 1887          2656 3                  END;
; 1888          2657 2              END;
; 1889          2658 2
; 1890          2659 2          TES;
; 1891          2660 1      END;



                                              0FFC 00000 INS_LINE:
                                                               .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 2477
                          5B 00000000V  EF  9E 00002           MOVAB   PADLIN, R11                             ;
                          5A 00000000G  EF  9E 00009           MOVAB   CMDBLK, R10                             ;
                          59 00000000G  EF  9E 00010           MOVAB   LLINES, R9                              ;
                          58 00000000G  EF  9E 00017           MOVAB   LTYPE, R8                               ;
                          57 00000000G  EF  9E 0001E           MOVAB   RCOUNT, R7                              ;
                          56 00000000G  EF  9E 00025           MOVAB   ALLOWD, R6                              ;
                          55 00000000'  EF  9E 0002C           MOVAB   BLANKS, R5                              ;
                          54 00000000G  EF  9E 00033           MOVAB   LCOUNT, R4                              ;
                          50        06  AA  32 0003A           CVTWL   CMDBLK+6, R0                            ; 2520
                          01            50  B1 0003E           CMPW    R0, #1                                  ; 2523
                                        03  13 00041           BEQL    1$                                      ;
                                      00A8  31 00043           BRW     10$                                     ;
                          66            64  D1 00046 1$:       CMPL    LCOUNT, ALLOWD                          ; 2527
                                        5C  12 00049           BNEQ    4$                                      ;
                                        67  D5 0004B           TSTL    RCOUNT                                  ; 2533
                                        0F  12 0004D           BNEQ    2$                                      ;
   04    AC          20       00  B5        01  2D 0004F           CMPC5   #1, @BLANKS, #32, INT_LEN, @PTR         ; 2539
                                  08    BC     00056                                                   ;
                                        5A  13 00058           BEQL    5$                                      ;
                     58           6A        06  E0 0005A           BBS     #6, CMDBLK, 6$                          ; 2541
                     50           66        01  C3 0005E 2$:       SUBL3   #1, ALLOWD, R0                          ; 2552
                                  50        67  D1 00062           CMPL    RCOUNT, R0                              ;
                                        05  12 00065           BNEQ    3$                                      ;
                                  57  04CC  C5  E9 00067           BLBC    DOING_LAST, 8$                          ;
                                        67  D6 0006C 3$:       INCL    RCOUNT                                  ; 2563
                                  50        67  D0 0006E           MOVL    RCOUNT, R0                              ; 2564
                     00000000GEF40  04C4  C5  D0 00071           MOVL    LINE_TYPE, RTYPE[R0]                    ;
                          00000000GEF40     7F 0007B           PUSHAQ  RLINES[R0]                              ; 2565
                                  7E    08  AC  7D 00082           MOVQ    PTR, -(SP)                              ;
                                        04  AC  DD 00086           PUSHL   INT_LEN                                 ;
                                  6B        04  FB 00089           CALLS   #4, PADLIN                              ;
                                            67  DD 0008C           PUSHL   RCOUNT                                  ; 2566
                          00000000G     EF  9F 0008E           PUSHAB  RTYPE                                   ;
                          00000000G     EF  9F 00094           PUSHAB  RLINES                                  ;
                     00000000V    EF        03  FB 0009A           CALLS   #3, SAVE_TELLTALE_HEADING               ;
                                  66        67  D1 000A1           CMPL    RCOUNT, ALLOWD                          ; 2568
```

```
                                              00AE  31 000A4       BRW     16$                                        ;
                                                64  D5 000A7 4$:   TSTL    LCOUNT                                     ; 2576
                                                0F  12 000A9       BNEQ    7$                                         ;
04   AC          20        00   B5              01  2D 000AB       CMPC5   #1, @BLANKS, #32, INT_LEN, @PTR            ; 2582
                                         08     BC     000B2                                                          ;
                                                45  13 000B4 5$:   BEQL    11$                                        ;
                 47             6A              06  E0 000B6 6$:   BBS     #6, CMDBLK, 12$                            ; 2584
                 50             66              01  C3 000BA 7$:   SUBL3   #1, ALLOWD, R0                             ; 2595
                                50              64  D1 000BE       CMPL    LCOUNT, R0                                 ;
                                                05  12 000C1       BNEQ    9$                                         ;
                                56       04CC   C5  E9 000C3 8$:   BLBC    DOING_LAST, 14$                            ;
                                                64  D6 000C8 9$:   INCL    LCOUNT                                     ; 2606
                                50              64  D0 000CA       MOVL    LCOUNT, R0                                 ; 2607
                              6840       04C4   C5  D0 000CD       MOVL    LINE_TYPE, LTYPE[R0]                       ;
                                              6940  7F 000D3       PUSHAQ  LLINES[R0]                                 ; 2608
                                7E         08   AC  7D 000D6       MOVQ    PTR, -(SP)                                 ;
                                           04   AC  DD 000DA       PUSHL   INT_LEN                                    ;
                                6B              04  FB 000DD       CALLS   #4, PADLIN                                 ;
                                                64  DD 000E0       PUSHL   LCOUNT                                     ; 2609
                                                58  DD 000E2       PUSHL   R8                                         ;
                                                59  DD 000E4       PUSHL   R9                                         ;
                         00000000V  EF          03  FB 000E6       CALLS   #3, SAVE_TELLTALE_HEADING                  ;
                                                    04 000ED       RET                                                ; 2527
                                                64  D5 000EE 10$:  TSTL    LCOUNT                                     ; 2620
                                                1E  12 000F0       BNEQ    13$                                        ;
04   AC          20        00   B5              01  2D 000F2       CMPC5   #1, @BLANKS, #32, INT_LEN, @PTR            ; 2626
                                         08     BC     000F9                                                          ;
                                                63  13 000FB 11$:  BEQL    17$                                        ;
                 0F             6A              06  E1 000FD       BBC     #6, CMDBLK, 13$                            ; 2628
                                7E         08   AC  7D 00101 12$:  MOVQ    PTR, -(SP)                                 ; 2634
                                           04   AC  DD 00105       PUSHL   INT_LEN                                    ;
                         00000000V  EF          03  FB 00108       CALLS   #3, DO_CONTINUE                            ;
                                                    04 0010F       RET                                                ; 2630
                 50             66              01  C3 00110 13$:  SUBL3   #1, ALLOWD, R0                             ; 2639
                                50              64  D1 00114       CMPL    LCOUNT, R0                                 ;
                                                14  12 00117       BNEQ    15$                                        ;
                                0F       04CC   C5  E8 00119       BLBS    DOING_LAST, 15$                            ;
                                7E         08   AC  7D 0011E 14$:  MOVQ    PTR, -(SP)                                 ; 2644
                                           04   AC  DD 00122       PUSHL   INT_LEN                                    ;
                         00000000V  EF          03  FB 00125       CALLS   #3, LAST_LINE                              ;
                                                    04 0012C       RET                                                ;
                                                64  D6 0012D 15$:  INCL    LCOUNT                                     ; 2650
                                50              64  D0 0012F       MOVL    LCOUNT, R0                                 ; 2651
                              6840       04C4   C5  D0 00132       MOVL    LINE_TYPE, LTYPE[R0]                       ;
                                              6940  7F 00138       PUSHAQ  LLINES[R0]                                 ; 2652
                                7E         08   AC  7D 0013B       MOVQ    PTR, -(SP)                                 ;
                                           04   AC  DD 0013F       PUSHL   INT_LEN                                    ;
                                6B              04  FB 00142       CALLS   #4, PADLIN                                 ;
                                                64  DD 00145       PUSHL   LCOUNT                                     ; 2653
                                                58  DD 00147       PUSHL   R8                                         ;
                                                59  DD 00149       PUSHL   R9                                         ;
                         00000000V  EF          03  FB 0014B       CALLS   #3, SAVE_TELLTALE_HEADING                  ;
                                66              64  D1 00152       CMPL    LCOUNT, ALLOWD                             ; 2655
                                                09  12 00155 16$:  BNEQ    17$                                        ;
                                                7E  D4 00157       CLRL    -(SP)                                      ;
                         00000000G  EF          01  FB 00159       CALLS   #1, PUTPAG                                 ;
                                                    04 00160 17$:  RET                                                ; 2660
```

```
; Routine Size:  353 bytes,    Routine Base:  $CODE$ + 0AF0
```

```
1 %SBTTL 'SAVE_TELLTALE_HEADING -- Save telltale heading string'
1 ROUTINE SAVE_TELLTALE_HEADING (
1     LINES : REF BLOCKVECTOR [, STR$K_D_BLN],
1     TYPES : REF VECTOR,
1     L_NUM
1     ) : NOVALUE =
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine is called by INS_LINE to save the current line as a
1 !       possible telltale heading.
1 !
1 !       The line is saved only if it is the first line of a continuation
1 !       heading, the first line of a multi-line top level entry or a
1 !       top-level entry which is only one line long.
1 !
1 !       If the line is saved, it is always saved as the right telltale string
1 !       which represents the last top level entry on the page.
1 !
1 !       The line is also saved as the left telltale if the right telltale
1 !       string is null which occurs after a right side page has been generated
1 !       and if the line is the first on the page.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       LINES           - Pointer to the blockvector of string descriptors
1 !                         for the current column.
1 !       TYPES           - Pointer to the vector of line types for the
1 !                         current column.
1 !       L_NUM           - Number of the line being generated.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       RLINES [0, ...] - The right telltale string
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       RLINES [0, ...] - The right telltale string is replaced if the line
1 !                         is saved.
1 !       RTYPES [0]      - The right telltale line type is replaced if the
1 !                         line is saved.
1 !       LLINES [0, ...] - The left telltale string is replaced if a right hand
1 !                         page was just generated.
1 !       LTYPES [0, ...] - The left telltale line type is replaced if the
1 !                         line is saved as a left telltale.
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !
1 !       None
1 !--
1
```

```
BEGIN

LOCAL
    LINE_NUMBER,
    SAVE_LINE;

LINE_NUMBER = .L_NUM;
SAVE_LINE = FALSE;

SELECTONE .TYPES [.LINE_NUMBER] OF
    SET

    [CONT_HEAD]:
        BEGIN
        !
        ! A continuation heading.
        !
        IF .LINE_NUMBER EQL 1
        THEN
            BEGIN
            !
            ! Only save the first line in a continuation heading
            !
            SAVE_LINE = TRUE;
            END;

        END;

    [GUIDE]:
        BEGIN
        !
        ! A guide heading.
        !
        BIND
            RHEAD = RLINES [0, 0,0,0,0] : $STR_DESCRIPTOR ();

        IF  (.LINE_NUMBER EQL 1)
        AND (.RHEAD [STR$H_LENGTH] EQL 0)
        THEN
            BEGIN
            !
            ! This guide heading is the first line on a left hand page.
            ! Set flag which indicates that the first top-level entry
            ! on the page should be saved as the left telltale.
            !
            SAVE_ENTRY = TRUE;
            END;

        END;

    [ENTRY_E]:
        BEGIN
        !
        ! The last line in a top level entry.
        !
        DECR I FROM .LINE_NUMBER - 1 TO 1 DO
            BEGIN
```

```
4                       !
4                       ! Find the first line in the entry.
4                       !
4                       SELECTONE .TYPES [.I] OF
4                           SET
4
4                           [ENTRY_B]:
5                               BEGIN
5                               !
5                               ! Found the first line in a multi-line entry.
5                               ! Save its line number and exit the loop.
5                               !
5                               LINE_NUMBER = .I;
5                               EXITLOOP;
4                               END;
4
4                           [ENTRY_W]:
4                               !
4                               ! Intermediate line of a multi-line entry.
4                               ! Continue search by doing nothing.
4                               !
4                               ;
4
4                           [OTHERWISE]:
4                               !
4                               ! Entry was not a multi-line entry. Exit the loop.
4                               !
4                               EXITLOOP;
4
4                           TES;
4
3                       END;
3
3                   SAVE_LINE = TRUE;
2                   END;
2
2               [OTHERWISE]:
2                   !
2                   ! Line should not be saved as a candidate for a telltale heading.
2                   !
2                   ;
2
2               TES;
2
2           IF .SAVE_LINE
2           THEN
3               BEGIN
3               !
3               ! We found a line which could be used as a telltale head.
3               !
3               BIND
3                   RHEAD = RLINES [0, 0,0,0,0] : $STR_DESCRIPTOR ();
3
4               IF  (.RHEAD [STR$H_LENGTH] EQL 0)
4               AND (
4                   (.LINE_NUMBER EQL 1) OR .SAVE_ENTRY
4                   )
```

```
        THEN
            BEGIN
            !
            ! The right telltale is null only after a right hand page has been
            ! generated. In this case, the line should be saved as the left
            ! telltale only if this is the first line on the page or if the
            ! first line on the page was a guide heading.
            !
            SAVE_ENTRY = FALSE;

            $STR_COPY (STRING = LINES [.LINE_NUMBER, 0,0,0,0],
                TARGET = LLINES [0, 0,0,0,0]);

            LTYPE [0] = .TYPES [.LINE_NUMBER];
            END;

        !
        ! Save the line as the right telltale.
        !
        $STR_COPY (STRING = LINES [.LINE_NUMBER, 0,0,0,0],
            TARGET = RLINES [0, 0,0,0,0]);

        RTYPE [0] = .TYPES [.LINE_NUMBER];
        END;

    END;


                                                    .EXTRN  XST$COPY, STR$FAILURE

                                  00FC 00000 SAVE_TELLTALE_HEADING:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7               ; 2662
         57 00000000G   EF   9E 00002               MOVAB   XST$COPY, R7                         ;
         56 00000000G   EF   9E 00009               MOVAB   STR$FAILURE, R6                      ;
         55 00000000G   EF   9E 00010               MOVAB   RHEAD, R5                            ;
         54 00000000'   EF   9E 00017               MOVAB   SAVE_ENTRY, R4                       ;
         52         0C  AC   D0 0001E               MOVL    L_NUM, LINE_NUMBER                   ; 2724
                        53   D4 00022               CLRL    SAVE_LINE                            ; 2725
         50      08 BC42     D0 00024               MOVL    @TYPES[LINE_NUMBER], R0              ; 2727
         0B             50   D1 00029               CMPL    R0, #11                              ; 2730
                        07   12 0002C               BNEQ    1$                                   ;
         01             52   D1 0002E               CMPL    LINE_NUMBER, #1                      ; 2735
                        39   12 00031               BNEQ    7$                                   ;
                        34   11 00033               BRB     6$                                   ; 2741
         03             50   D1 00035 1$:           CMPL    R0, #3                               ; 2746
                        0E   12 00038               BNEQ    2$                                   ;
         01             52   D1 0003A               CMPL    LINE_NUMBER, #1                      ; 2754
                        2D   12 0003D               BNEQ    7$                                   ;
                        65   B5 0003F               TSTW    RHEAD                                ; 2755
                        29   12 00041               BNEQ    7$                                   ;
         64             01   D0 00043               MOVL    #1, SAVE_ENTRY                       ; 2763
                        24   11 00046               BRB     7$                                   ; 2727
         07             50   D1 00048 2$:           CMPL    R0, #7                               ; 2768
                        1F   12 0004B               BNEQ    7$                                   ;
         50             52   D0 0004D               MOVL    LINE_NUMBER, I                       ; 2773
                        14   11 00050               BRB     5$                                   ;
```

```
                 51        08 BC40  D0 00052 3$:    MOVL    @TYPES[I], R1                            ; 2778
                 05             51  D1 00057        CMPL    R1, #5                                   ; 2781
                                05  12 0005A        BNEQ    4$                                       ;
                 52             50  D0 0005C        MOVL    I, LINE_NUMBER                           ; 2787
                                08  11 0005F        BRB     6$                                       ; 2782
                 06             51  D1 00061 4$:    CMPL    R1, #6                                   ; 2791
                                03  12 00064        BNEQ    6$                                       ;
                 E9             50  F5 00066 5$:    SOBGTR  I, 3$                                    ; 2773
                 53             01  D0 00069 6$:    MOVL    #1, SAVE_LINE                            ; 2808
                 42             53  E9 0006C 7$:    BLBC    SAVE_LINE, 10$                           ; 2819
                                65  B5 0006F        TSTW    RHEAD                                    ; 2828
                                26  12 00071        BNEQ    9$                                       ;
                 01             52  D1 00073        CMPL    LINE_NUMBER, #1                          ; 2830
                                03  13 00076        BEQL    8$                                       ;
                 1E             64  E9 00078        BLBC    SAVE_ENTRY, 9$                           ;
                                64  D4 0007B 8$:    CLRL    SAVE_ENTRY                               ; 2840
                                56  DD 0007D        PUSHL   R6                                       ; 2843
                                7E  D4 0007F        CLRL    -(SP)                                    ;
                       00000000G EF  9F 00081        PUSHAB  $STR$TARGET                              ;
                           04 BC42  7F 00087        PUSHAQ  @LINES[LINE_NUMBER]                      ;
                                7E  D4 0008B        CLRL    -(SP)                                    ;
                 67             05  FB 0008D        CALLS   #5, XST$COPY                             ;
       00000000G EF        08 BC42  D0 00090        MOVL    @TYPES[LINE_NUMBER], LTYPE               ; 2845
                                56  DD 00099 9$:    PUSHL   R6                                       ; 2852
                                7E  D4 0009B        CLRL    -(SP)                                    ;
                                55  DD 0009D        PUSHL   R5                                       ;
                           04 BC42  7F 0009F        PUSHAQ  @LINES[LINE_NUMBER]                      ;
                                7E  D4 000A3        CLRL    -(SP)                                    ;
                 67             05  FB 000A5        CALLS   #5, XST$COPY                             ;
       00000000G EF        08 BC42  D0 000A8        MOVL    @TYPES[LINE_NUMBER], RTYPE               ; 2854
                                04  000B1 10$:      RET                                              ; 2857

; Routine Size:  178 bytes,    Routine Base:  $CODE$ + 0C51
```

```
%SBTTL 'LAST_LINE -- Process last line in column'
ROUTINE LAST_LINE (I_LEN, PTR, E_LEN) : NOVALUE =
!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine processes the last line in the column.
!
!       If the current line is the beginning or middle of a subindex entry,
!       any preceeding lines which are also part of that subindex entry
!       will be moved to the next column. If the line preceeding the subindex
!       entry is the end of a top level entry, the top level entry will also
!       be moved. If the line preceeding the top level entry is part of a
!       guide head, the guide head will also be moved.
!
!       If the line is part of a top level entry, the entry will be moved if
!       either it is not the last line in the entry or if the entry has
!       subindex entries. If the entry is to be moved and the line preceeding
!       the heading is part of a guide heading, the guide heading is also moved.
!
!       If the line is part of a guide heading, the guide heading is moved
!       to the next column.
!
!       Lines are moved from the current column to the next by determining
!       the number of lines to be moved and by establishing a pointer to
!       the first line to be moved. The lines are then copied to the temporary
!       column TLINES and the rest of the current column are filled with
!       blank FILL lines. The copied lines are then inserted into the next
!       output column.
!
! FORMAL PARAMETERS:
!
!       I_LEN   - Internal line length
!       PTR     - CH$PTR to line
!       E_LEN   - External length
!
! IMPLICIT INPUTS:
!
!       ALLOWD          - number of lines allowed in column
!       RTYPE           - line types if doing right column
!       RLINES          - right column lines if doing right column
!       RCOUNT          - number of lines in column if doing right column
!       LTYPE           - line types if doing left column
!       LLINES          - left column lines if doing left column
!       LCOUNT          - number of lines in left column
!       LINE_TYPE       - type of current line
!       CMDBLK          - command line information block
!
! IMPLICIT OUTPUTS:
!
!       None
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
```

```
1 ! SIDE EFFECTS:
1 !
1 !       During the execution of this routine, DOING_LAST is set to TRUE
1 !       to prevent INS_LINE from calling this routine again which would
1 !       cause an infinite loop of calls.
1 !
1 !       TLINES AND TTYPE are used during the execution of this routine.
1 !       Any other routine which may be called during the execution of
1 !       this routine must use some other area to store lines and line types.
1 !--
2     BEGIN
2
2     LOCAL
2         TYPE,
2         COUNT_PTR,
2         TYPE_PTR          : REF VECTOR,
2         LINES_PTR         : REF BLOCKVECTOR [, STR$K_D_BLN],
2         COPY_LINES,
2         FILL_LINES,
2         S                 : REF $STR_DESCRIPTOR ();
2
2     DOING_LAST = TRUE;                                    ! Tell INS_LINE we're processing last line
2
2     COPY_LINES = 0;                                       ! Number of lines to copy to next column
2     FILL_LINES = 0;                                       ! Number of fill lines to insert in this column
2     TYPE = .LINE_TYPE;                                    ! Save current line type
2
2     IF .LCOUNT EQL .ALLOWD
2     THEN
3         BEGIN                                             ! Filling right column
3         COUNT_PTR = RCOUNT;                               ! Set up pointers to column variables
3         TYPE_PTR = RTYPE [0];
3         LINES_PTR = RLINES [0, 0,0,0,0];
3         END
2     ELSE
3         BEGIN                                             ! Filling left column
3         COUNT_PTR = LCOUNT;                               ! Set up pointers...
3         TYPE_PTR = LTYPE [0];
3         LINES_PTR = LLINES [0, 0,0,0,0];
2         END;
2
2     CASE .TYPE FROM BKT_E TO CONT_HEAD OF
2     SET
2
2     [SUB_B, SUB_W]:
3         BEGIN
3         !
3         ! Beginning or middle of subindex entry
3         !
3         DECR I FROM ..COUNT_PTR TO 1 DO
4             IF (.TYPE_PTR [.I] NEQ SUB_W) AND (.TYPE_PTR [.I] NEQ SUB_B)
3             THEN
3                 EXITLOOP
3             ELSE
4                 BEGIN
4                 COPY_LINES = .COPY_LINES + 1;     ! Copy this line to next column
4                 .COUNT_PTR = ..COUNT_PTR - 1;     ! One less line in column
```

```
                        END;

                IF (.TYPE_PTR [..COUNT_PTR] EQL ENTRY_E)
                AND .CMDBLK [NDX$V_CONTINUATION]
                THEN
                    BEGIN
                    !
                    ! Previous line is the end of an entry and we are
                    ! doing continuation headings. Move the entry too.
                    !
                    COPY_LINES = .COPY_LINES + 1;       ! Copy this line to next column
                    .COUNT_PTR = ..COUNT_PTR - 1;       ! One less line in column

                    DECR I FROM ..COUNT_PTR TO 1 DO
                        IF (.TYPE_PTR [.I] NEQ ENTRY_W) AND (.TYPE_PTR [.I] NEQ ENTRY_B)
                        THEN
                            EXITLOOP
                        ELSE
                            BEGIN
                            COPY_LINES = .COPY_LINES + 1;
                            .COUNT_PTR = ..COUNT_PTR - 1;
                            END;

                    IF .TYPE_PTR [..COUNT_PTR] EQL GUIDE_FILL
                    THEN
                        BEGIN                           ! Move the guide heading too
                        COPY_LINES = .COPY_LINES + 2;
                        .COUNT_PTR = ..COUNT_PTR - 2;
                        END;
                    END;

                FILL_LINES = .COPY_LINES + 1;
                END;

            [ENTRY_B TO ENTRY_E]:
                BEGIN
                !
                ! An index entry
                !
                IF (.TYPE NEQ ENTRY_E)
                OR (.CMDBLK [NDX$V_CONTINUATION] AND (.LSTPTR [XE$A_SUBX] NEQ 0))
                THEN
                    BEGIN
                    !
                    ! Line is not the end of an entry
                    !               OR
                    ! Doing continuation headings and have subindex entries.
                    !
                    ! Move it to next column.
                    !
                    DECR I FROM ..COUNT_PTR TO 1 DO
                        IF (.TYPE_PTR [.I] NEQ ENTRY_W) AND (.TYPE_PTR [.I] NEQ ENTRY_B)
                        THEN
                            EXITLOOP
                        ELSE
                            BEGIN
                            COPY_LINES = .COPY_LINES + 1;
```

```
                          .COUNT_PTR = ..COUNT_PTR - 1;
                          END;

                      IF .TYPE_PTR [..COUNT_PTR] EQL GUIDE_FILL
                      THEN
                          BEGIN                           ! Move the guide heading too
                          COPY_LINES = .COPY_LINES + 2;
                          .COUNT_PTR = ..COUNT_PTR - 2;
                          END;

                      FILL_LINES = .COPY_LINES + 1;
                      END;
                  END;

          [GUIDE_FILL]:
                  BEGIN
                  !
                  ! Line following guide heading.
                  !
                  COPY_LINES = 1;
                  .COUNT_PTR = ..COUNT_PTR - 1;
                  FILL_LINES = 2;
                  END;

          [GUIDE]:
                  !
                  ! Guide heading.
                  ! Start guide heading in next column.
                  !
                  FILL_LINES = 1;

          [INRANGE]:
                  !
                  ! Line is one of: BKT_E, FILL, SUB_E or CONT_HEAD. Just insert it.
                  !
                  ;

          TES;

          IF .COPY_LINES NEQ 0
          THEN
                  !
                  ! Copy lines to temp column
                  !
                  INCR I FROM 1 TO .COPY_LINES DO
                      BEGIN
                      $STR_COPY (STRING = LINES_PTR [..COUNT_PTR + .I, 0,0,0,0],
                          TARGET = TLINES [.I, 0,0,0,0]);
                      TTYPE [.I] = .TYPE_PTR [..COUNT_PTR + .I];
                      END;

          !
          ! Copy line we started with
          !
          COPY_LINES = .COPY_LINES + 1;
          TTYPE [.COPY_LINES] = .TYPE;
          PADLIN (.I_LEN, .PTR, .E_LEN, TLINES [.COPY_LINES, 0,0,0,0]);
```

```
2
2       IF .FILL_LINES NEQ 0
2       THEN
3           BEGIN
3           !
3           ! Insert FILL lines to fill out rest of column
3           !
3           LINE_TYPE = FILL;
3
3           INCR I FROM 1 TO .FILL_LINES DO INS_LINE (4, .BLANKS, 4);
2           END;
2
2       IF .COPY_LINES NEQ 0
2       THEN
2           !
2           ! Put copied lines in new column
2           !
2           INCR I FROM 1 TO .COPY_LINES DO
3               BEGIN
3               S = TLINES [.I, 0,0,0,0];
3               LINE_TYPE = .TTYPE [.I];
3               INS_LINE (.S [STR$H_LENGTH], .S [STR$A_POINTER], 0);
2               END;
2
2       DOING_LAST = FALSE;                     ! Not doing last line any more
2
1       END;
```

```
                                        0FFC 00000 LAST_LINE:
                                                       .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 2859
                    5B 00000000G  EF  9E 00002         MOVAB   TLINES, R11                                  ;
                    5A 00000000'  EF  9E 00009         MOVAB   LINE_TYPE, R10                               ;
              08    AA            01  D0 00010         MOVL    #1, DOING_LAST                               ; 2936
                                  55  D4 00014         CLRL    COPY_LINES                                   ; 2938
                                  57  D4 00016         CLRL    FILL_LINES                                   ; 2939
                    59            6A  D0 00018         MOVL    LINE_TYPE, TYPE                              ; 2940
       00000000G    EF 00000000G  EF  D1 0001B         CMPL    LCOUNT, ALLOWD                               ; 2942
                                  17  12 00026         BNEQ    1$                                           ;
                    52 00000000G  EF  9E 00028         MOVAB   RCOUNT, COUNT_PTR                            ; 2945
                    56 00000000G  EF  9E 0002F         MOVAB   RTYPE, TYPE_PTR                              ; 2946
                    58 00000000G  EF  9E 00036         MOVAB   RLINES, LINES_PTR                            ; 2947
                                  15  11 0003D         BRB     2$                                           ; 2942
                    52 00000000G  EF  9E 0003F 1$:     MOVAB   LCOUNT, COUNT_PTR                            ; 2951
                    56 00000000G  EF  9E 00046         MOVAB   LTYPE, TYPE_PTR                              ; 2952
                    58 00000000G  EF  9E 0004D         MOVAB   LLINES, LINES_PTR                            ; 2953
          0A        01                59  CF 00054 2$:     CASEL   TYPE, #1, #10                                ; 2956
  00AF    00B9      00BC          00BC       00058 3$:     .WORD   23$-3$,-                                     ;
  0016    0068      0068          0068       00060                 23$-3$,-                                     ;
          00BC      00BC          0016       00068                 22$-3$,-                                     ;
                                                                   21$-3$,-                                     ;
                                                                   13$-3$,-                                     ;
                                                                   13$-3$,-                                     ;
                                                                   13$-3$,-                                     ;
```

```
                                                           4$-3$,-                        ;
                                                           4$-3$,-                        ;
                                                           23$-3$,-                       ;
                                                           23$-3$                         ;
         50              62         01  C1 0006E 4$:   ADDL3   #1, (COUNT_PTR), I        ; 2965
                                    10  11 00072       BRB     7$                         ;
                         09       6640  D1 00074 5$:   CMPL    (TYPE_PTR)[I], #9          ;
                                    06  13 00078       BEQL    6$                         ;
                         08       6640  D1 0007A       CMPL    (TYPE_PTR)[I], #8          ;
                                    07  12 0007E       BNEQ    8$                         ;
                                    55  D6 00080 6$:   INCL    COPY_LINES                 ; 2970
                                    62  D7 00082       DECL    (COUNT_PTR)                ; 2971
                         ED         50  F5 00084 7$:   SOBGTR  I, 5$                      ; 2965
                         50         62  D0 00087 8$:   MOVL    (COUNT_PTR), R0            ; 2974
                         07       6640  D1 0008A       CMPL    (TYPE_PTR)[R0], #7         ;
                                    71  12 0008E       BNEQ    20$                        ;
         69 00000000G    EF         06  E1 00090       BBC     #6, CMDBLK, 20$            ; 2975
                                    55  D6 00098       INCL    COPY_LINES                 ; 2982
                                    62  D7 0009A       DECL    (COUNT_PTR)                ; 2983
         50              62         01  C1 0009C       ADDL3   #1, (COUNT_PTR), I         ; 2985
                                    10  11 000A0       BRB     11$                        ;
                         06       6640  D1 000A2 9$:   CMPL    (TYPE_PTR)[I], #6          ; 2986
                                    06  13 000A6       BEQL    10$                        ;
                         05       6640  D1 000A8       CMPL    (TYPE_PTR)[I], #5          ;
                                    07  12 000AC       BNEQ    12$                        ;
                                    55  D6 000AE 10$:  INCL    COPY_LINES                 ; 2991
                                    62  D7 000B0       DECL    (COUNT_PTR)                ; 2992
                         ED         50  F5 000B2 11$:  SOBGTR  I, 9$                      ; 2986
                         50         62  D0 000B5 12$:  MOVL    (COUNT_PTR), R0            ; 2995
                         04       6640  D1 000B8       CMPL    (TYPE_PTR)[R0], #4         ;
                                    3D  13 000BC       BEQL    19$                        ;
                                    41  11 000BE       BRB     20$                        ; 3003
                         07         59  D1 000C0 13$:  CMPL    TYPE, #7                   ; 3011
                                    14  12 000C3       BNEQ    14$                        ;
         47 00000000G    EF         06  E1 000C5       BBC     #6, CMDBLK, 23$            ; 3012
                         50 00000000G   EF  D0 000CD   MOVL    LSTPTR, R0                 ;
                                 08 A0  D5 000D4       TSTL    8(R0)                      ;
                                    3B  13 000D7       BEQL    23$                        ;
         50              62         01  C1 000D9 14$:  ADDL3   #1, (COUNT_PTR), I         ; 3023
                                    10  11 000DD       BRB     17$                        ;
                         06       6640  D1 000DF 15$:  CMPL    (TYPE_PTR)[I], #6          ;
                                    06  13 000E3       BEQL    16$                        ;
                         05       6640  D1 000E5       CMPL    (TYPE_PTR)[I], #5          ;
                                    07  12 000E9       BNEQ    18$                        ;
                                    55  D6 000EB 16$:  INCL    COPY_LINES                 ; 3028
                                    62  D7 000ED       DECL    (COUNT_PTR)                ; 3029
                         ED         50  F5 000EF 17$:  SOBGTR  I, 15$                     ; 3023
                         50         62  D0 000F2 18$:  MOVL    (COUNT_PTR), R0            ; 3032
                         04       6640  D1 000F5       CMPL    (TYPE_PTR)[R0], #4         ;
                                    06  12 000F9       BNEQ    20$                        ;
                         55         02  C0 000FB 19$:  ADDL2   #2, COPY_LINES             ; 3035
                         62         02  C2 000FE       SUBL2   #2, (COUNT_PTR)            ; 3036
                         57      01 A5  9E 00101 20$:  MOVAB   1(R5), FILL_LINES          ; 3039
                                    0D  11 00105       BRB     23$                        ; 2956
                         55         01  D0 00107 21$:  MOVL    #1, COPY_LINES             ; 3048
                                    62  D7 0010A       DECL    (COUNT_PTR)                ; 3049
                         57         02  D0 0010C       MOVL    #2, FILL_LINES             ; 3050
```

```
                                03  11 0010F        BRB     23$                              ; 2956
               57               01  D0 00111 22$:   MOVL    #1, FILL_LINES                   ; 3058
                                55  D5 00114 23$:   TSTL    COPY_LINES                       ; 3068
                                2C  13 00116        BEQL    26$                              ;
                                53  D4 00118        CLRL    I                                ; 3077
                                24  11 0011A        BRB     25$                              ;
     54        62               53  C1 0011C 24$:   ADDL3   I, (COUNT_PTR), R4               ; 3076
                    00000000G   EF  9F 00120        PUSHAB  STR$FAILURE                      ;
                                7E  D4 00126        CLRL    -(SP)                            ;
                              6B43  7F 00128        PUSHAQ  TLINES[I]                        ;
                              6844  7F 0012B        PUSHAQ  (LINES_PTR)[R4]                  ;
                                7E  D4 0012E        CLRL    -(SP)                            ;
        00000000G  EF           05  FB 00130        CALLS   #5, XST$COPY                     ;
        00000000GEF43         6644  D0 00137        MOVL    (TYPE_PTR)[R4], TTYPE[I]         ; 3077
     D8        53               55  F3 00140 25$:   AOBLEQ  COPY_LINES, I, 24$               ; 3073
                                55  D6 00144 26$:   INCL    COPY_LINES                       ; 3083
        00000000GEF45           59  D0 00146        MOVL    TYPE, TTYPE[COPY_LINES]          ; 3084
                              6B45  7F 0014E        PUSHAQ  TLINES[COPY_LINES]               ; 3085
               7E          08   AC  7D 00151        MOVQ    PTR, -(SP)                       ;
                           04   AC  DD 00155        PUSHL   I_LEN                            ;
        00000000V  EF           04  FB 00158        CALLS   #4, PADLIN                       ;
                                57  D5 0015F        TSTL    FILL_LINES                       ; 3087
                                18  13 00161        BEQL    29$                              ;
               6A               02  D0 00163        MOVL    #2, LINE_TYPE                    ; 3093
                                53  D4 00166        CLRL    I                                ; 3095
                                0D  11 00168        BRB     28$                              ;
                                04  DD 0016A 27$:   PUSHL   #4                               ;
                         FB3C   CA  DD 0016C        PUSHL   BLANKS                           ;
                                04  DD 00170        PUSHL   #4                               ;
           FC76    CF           03  FB 00172        CALLS   #3, INS_LINE                     ;
     EF            53           57  F3 00177 28$:   AOBLEQ  FILL_LINES, I, 27$               ;
                                55  D5 0017B 29$:   TSTL    COPY_LINES                       ; 3098
                                21  13 0017D        BEQL    32$                              ;
                                53  D4 0017F        CLRL    I                                ; 3103
                                19  11 00181        BRB     31$                              ;
               52             6B43  7E 00183 30$:   MOVAQ   TLINES[I], S                     ; 3105
               6A 00000000GEF43     D0 00187        MOVL    TTYPE[I], LINE_TYPE              ; 3106
                                7E  D4 0018F        CLRL    -(SP)                            ; 3107
                           04   A2  DD 00191        PUSHL   4(S)                             ;
               7E               62  3C 00194        MOVZWL  (S), -(SP)                       ;
           FC51    CF           03  FB 00197        CALLS   #3, INS_LINE                     ;
     E3            53           55  F3 0019C 31$:   AOBLEQ  COPY_LINES, I, 30$               ; 3103
                           08   AA  D4 001A0 32$:   CLRL    DOING_LAST                       ; 3110
                                    04 001A3        RET                                      ; 3112

; Routine Size:  420 bytes,    Routine Base:  $CODE$ + 0D03
```

```
1 %SBTTL 'DO_CONTINUE -- Generate continuation heading if necessary'
1 ROUTINE DO_CONTINUE (I_LEN, PTR, E_LEN) : NOVALUE =
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine is called to generate a continuation heading if necessary
1 !
1 !       The input line is copied to local storage since it may reside in
1 !       RNO_LINE which is used to build the continuation heading. Note
1 !       that the input line cannot be copied to TLINES since this routine
1 !       may be called during the execution of LAST_LINE which uses TLINES
1 !       to temporarily store one to many lines of text.
1 !
1 !       The continuation heading is generated by setting LINE_TYPE to
1 !       CONT_HEAD and by scanning the subindex list stack from bottom
1 !       to top, calling FORMAT_ENTRY for each level.
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       I_LEN   - Internal length of line being processed
1 !       PTR     - CH$PTR to line
1 !       E_LEN   - External length of line being processed
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       CMDBLK  - Command line information block
1 !       LSTSTK  - Subindex list stack
1 !       INDLVL  - Subindex level
1 !       LINE_TYPE - Type of current line
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !
1 !       Sets CMDBLK [NDX$V_CONTINUATION] to false while processing
1 !       the input line. This prevents INS_LINE from calling the routine
1 !       again which would cause an infinite loop.
1 !
1 !       Sets LINE_TYPE to CONT_HEAD so that FORMAT_ENTRY will generate
1 !       a continuation heading instead of normal index entries.
1 !--
2    BEGIN
2    LOCAL
2       LINE : VECTOR [CH$ALLOCATION(1200)],
2       INT,
2       EXT,
2       TYP;
2
2    CMDBLK [NDX$V_CONTINUATION] = FALSE;        ! Prevent infinite loop of calls
```

```
2
2       !
2       ! Copy input parameters and input line
2       !
2       INT = .I_LEN;
2       EXT = .E_LEN;
2       TYP = .LINE_TYPE;
2       CH$MOVE (.INT, .PTR, CH$PTR (LINE));
2       CLR_RNO_LINE ();
2
3       IF (.TYP EQL SUB_B)                                 ! First line of subindex entry
3       OR (.TYP EQL SUB_E)                                 ! Last line of subindex entry
2       THEN                                                ! NOTE: can't get SUB_W as first
3           BEGIN                                           ! line in column.
3           LOCAL
3               PTR,
3               SAV_INDLVL;
3
3           PTR = .LSTPTR;                                  ! Save current entry pointer
3           SAV_INDLVL = .INDLVL;                           ! Save current index level
3           LSTSTK [.INDLVL] = 0;                           ! Make sure top of stack is zero
3
3           INCR I FROM 0 TO .SAV_INDLVL - 1 DO
4               BEGIN                                       ! Generate continuation heading
4               INDLVL = .I;
4               LSTPTR = .LSTSTK [.I];
4               LINE_TYPE = CONT_HEAD;
4               FORMAT_ENTRY ();
3               END;
3
3           INDLVL = .SAV_INDLVL;                           ! Restore current index level
3           LSTPTR = .PTR;                                  ! and entry pointer
3           END;
2
2       LINE_TYPE = .TYP;
2       INS_LINE (.INT, CH$PTR (LINE), .EXT);               ! Insert original line
2       CMDBLK [NDX$V_CONTINUATION] = TRUE;                 ! Reset continuation switch
1       END;



                                        0FFC 00000 DO_CONTINUE:
                                                   .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 3114
                 5B 00000000G   EF   9E 00002      MOVAB    INDLVL, R11                                   ;
                 5A 00000000G   EF   9E 00009      MOVAB    LSTPTR, R10                                   ;
                 59 00000000'   EF   9E 00010      MOVAB    LINE_TYPE, R9                                 ;
                 5E       FB50  CE   9E 00017      MOVAB    -1200(SP), SP                                 ;
        00000000G EF        40  8F   8A 0001C      BICB2    #64, CMDBLK                                   ; 3169
                 57         04  AC   D0 00024      MOVL     I_LEN, INT                                    ; 3174
                 58         0C  AC   D0 00028      MOVL     E_LEN, EXT                                    ; 3175
                 56             69   D0 0002C      MOVL     LINE_TYPE, TYP                                ; 3176
     6E       08 BC             57   28 0002F      MOVC3    INT, @PTR, LINE                               ; 3177
            FB40 C9       FB44  C9   9E 00034      MOVAB    RNO_LINE, LINE_POINTER                        ; 3178
                            F4  A9   7C 0003B      CLRQ     INT_LINE_LENGTH                               ;
                            FC  A9   D4 0003E      CLRL     TMS_LINE_LENGTH                               ;
```

```
                    08           56  D1 00041       CMPL    TYP, #8                        ; 3180
                                 05  13 00044       BEQL    1$                             ;
                    0A           56  D1 00046       CMPL    TYP, #10                       ; 3181
                                 32  12 00049       BNEQ    4$                             ;
                    54           6A  D0 0004B 1$:   MOVL    LSTPTR, PTR                    ; 3188
                    50           6B  D0 0004E       MOVL    INDLVL, R0                     ; 3189
                    53           50  D0 00051       MOVL    R0, SAV_INDLVL                 ;
                      00000000GEF40  D4 00054       CLRL    LSTSTK[R0]                     ; 3190
                    52           01  CE 0005B       MNEGL   #1, I                          ; 3192
                                 13  11 0005E       BRB     3$                             ;
                    6B           52  D0 00060 2$:   MOVL    I, INDLVL                      ; 3194
                    6A 00000000GEF42 D0 00063       MOVL    LSTSTK[I], LSTPTR              ; 3195
                    69           0B  D0 0006B       MOVL    #11, LINE_TYPE                 ; 3196
              F5DD  CF           00  FB 0006E       CALLS   #0, FORMAT_ENTRY               ; 3197
        E9          52           53  F2 00073 3$:   AOBLSS  SAV_INDLVL, I, 2$              ; 3192
                    6B           53  D0 00077       MOVL    SAV_INDLVL, INDLVL             ; 3200
                    6A           54  D0 0007A       MOVL    PTR, LSTPTR                    ; 3201
                    69           56  D0 0007D 4$:   MOVL    TYP, LINE_TYPE                 ; 3204
                                 58  DD 00080       PUSHL   EXT                            ; 3205
                          04     AE  9F 00082       PUSHAB  LINE                           ;
                                 57  DD 00085       PUSHL   INT                            ;
              FBBD  CF           03  FB 00087       CALLS   #3, INS_LINE                   ;
         00000000G  EF     40    8F  88 0008C       BISB2   #64, CMDBLK                    ; 3206
                                     04 00094       RET                                    ; 3207

; Routine Size:  149 bytes,    Routine Base:  $CODE$ + 0EA7
```

```
%SBTTL 'FILL_LINE -- Format an output line'
ROUTINE FILL_LINE (I_LEN, I_PTR, INDENT, MAX_EXT, BOLD, UND, L_BLANK) : NOVALUE =
!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine copies and formats the string specified by
!       I_LEN and I_PTR into RNO_LINE. If the external length
!       of the line exceeds MAX_EXT, the line is forced out.
!
! FORMAL PARAMETERS:
!
!       I_LEN   - Length of input string
!       I_PTR   - Pointer to input string
!       INDENT  - Number of spaces to indent line
!       MAX_EXT - Maximum external line length allowed.
!       BOLD    - TRUE if string is to be bolded
!       UND     - TRUE if string is to be underlined
!       L_BLANK - TRUE if a blank should be inserted before the string
!
! IMPLICIT INPUTS:
!
!       LINE_POINTER    - pointer to next character in output line
!       EXT_LINE_LENGTH - external line length
!       INT_LINE_LENGTH - internal line length
!       CMDBLK          - command line information block
!       TMSSTD          - standard character size for TMS
!       TMS_LINE_LENGTH - length of line in TMS relative units
!       CHRSIZ          - TMS character size vector
!
! IMPLICIT OUTPUTS:
!
!       LINE_POINTER    - pointer to next character in output line
!       EXT_LINE_LENGTH - external line length
!       INT_LINE_LENGTH - internal line length
!       TMS_LINE_LENGTH - length of line in TMS relative units
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
! SIDE EFFECTS:
!
!       None
!--
    BEGIN
    LOCAL
        WORD_POINTER,
        RNO_WORD : VECTOR [CH$ALLOCATION(1200)],
        INT_WORD_LENGTH,
        EXT_WORD_LENGTH,
        TMS_WORD_LENGTH,
        MAX_LEN,
        HYPHENATE,
        HYPH,
        INSERT_BLANK,
```

```
	    CHAR_COUNT,
	    TEXT_PTR;

	IF .CMDBLK [NDX$H_FORMAT] EQL DSR
	THEN
	    !
	    ! RUNOFF output.
	    !
	    MAX_LEN = .MAX_EXT
	ELSE
	    !
	    ! For TMS11 or TEX output, we don't count characters.
	    !
	    ! The line length is expressed in units and each character is
	    ! assigned a length in units.
	    !
	    ! The line length is computed as follows:
	    !	    (line length in characters - safety margin in characters)
	    !	    * average character size
	    !
	    ! The safety margin is present to allow for slightly larger
	    ! italicized or bolded characters.
	    !
	    MAX_LEN = (.MAX_EXT - TMS_SAFETY_MARGIN) * TMSSTD;

	CHAR_COUNT = .I_LEN;
	TEXT_PTR = .I_PTR;

	HYPH = FALSE;
	INSERT_BLANK = .L_BLANK;

	WHILE .CHAR_COUNT GTR 0 DO
	    BEGIN
	    LOCAL
		L_PTR,
		L_LEN;

	    CLR_RNO_WORD ();				! Initialize word variables

	    !
	    ! Get next word.
	    !
	    L_PTR = .TEXT_PTR;
	    L_LEN = .CHAR_COUNT;
	    INCR I FROM 1 TO .CHAR_COUNT DO
		BEGIN
		!
		! Skip leading whitespace if any
		!
		LOCAL
		    CH;

		CH = CH$RCHAR_A (L_PTR);
		IF (.CH EQL %C' ') OR (.CH EQL TAB)
		THEN
		    L_LEN = .L_LEN - 1			! One less input character
		ELSE
```

```
                    BEGIN
                    !
                    ! End of leading whitespace
                    !
                    L_PTR = CH$PLUS (.L_PTR, -1);   ! Back up to account for overshoot
                    EXITLOOP;
                    END;
                END;

        TEXT_PTR = .L_PTR;                          ! Point past leading whitespace
        CHAR_COUNT = .L_LEN;                        ! Update length of string

        INCR I FROM 1 TO .CHAR_COUNT DO
            BEGIN
            !
            ! Now copy the word to the word buffer
            !
            LOCAL
                CH;

            CH = CH$RCHAR_A (L_PTR);

            IF .CH EQL RINTES
            THEN
                BEGIN
                !
                ! RUNOFF escape sequence
                !
                LOCAL
                    OPERAND;

                CH = CH$RCHAR_A (L_PTR);            ! Get function
                OPERAND = CH$RCHAR_A (L_PTR);       ! And the operand
                I = .I + 2;                         ! Update number of characters processed
                L_LEN = .L_LEN - 3;                 ! Update remaining character count

                SELECTONE .CH OF
                    SET

                    [%C'B']:                        ! Bold
                        NQCHAR_NC (%C'*');

                    [%C'O']:                        ! Overstrike
                        BEGIN
                        NQCHAR_NC (.OPERAND);
                        NQCHAR_NC (%C'%');
                        END;

                    [%C'U']:                        ! Underline
                        NQCHAR_NC (%C'&');

                    [%C'N']:
                        IF .OPERAND EQL %C'-'
                        THEN
                            BEGIN                   ! Hyphenate
                            HYPHENATE = TRUE;
                            EXITLOOP;
```

```
                            END;

                        TES;
                    END
                ELSE
                    BEGIN
                    !
                    ! Not a RUNOFF escape sequence, process character
                    !
                    L_LEN = .L_LEN - 1;                 ! One less character

                    IF (.CH EQL %C' ') OR (.CH EQL TAB)
                    THEN
                        EXITLOOP;                       ! Whitespace - end of word

                    !
                    ! Bold and underline character if requested
                    !
                    IF .BOLD THEN NQCHAR_NC (%C'*');
                    IF .UND  THEN NQCHAR_NC (%C'&');

                    IF (.CH LSS %C' ') OR (.CH GTR %O'176')
                    THEN
                        !
                        ! Control character
                        !
                        IF .CH NEQ %O'10'
                        THEN
                            QCHAR_C (.CH)               ! Normal control character
                        ELSE
                            BACKSPACE (.CH)             ! Process backspace
                    ELSE
                        !
                        ! Not a control character
                        !
                        IF (.CH EQL %C'&')
                        OR (.CH EQL %C'*')
                        OR (.CH EQL %C'_')
                        OR (.CH EQL %C'-')
                        OR (.CH EQL %C'%')
                        THEN
                            QCHAR_C (.CH)               ! RUNOFF flag so quote it
                        ELSE
                            NQCHAR_C (.CH);             ! Ordinary character
                    END;
                END;

            TEXT_PTR = .L_PTR;                          ! Point to next character
            CHAR_COUNT = .L_LEN;                        ! Update length of input string

            !
            ! Compute space needed if word is to be added to current line.
            !
            IF .CMDBLK [NDX$H_FORMAT] EQL DSR
            THEN
                BEGIN
                !
```

```
                ! For RUNOFF, total = line length + word length
                !   + 1 for comma in case this is the last word in the entry
                !   + 1 if previous word was not terminated by a hyphenate sequence
                !   + 1 if current word terminated with a hyphenate sequence
                !
                L_LEN = .EXT_LINE_LENGTH + .EXT_WORD_LENGTH + 1
                    + (IF NOT .HYPH THEN 1 ELSE 0) + (IF .HYPHENATE THEN 1 ELSE 0);
                END
            ELSE
                BEGIN
                !
                ! For TMS and TEX, total length = line length + word length
                !   + length of blank in case this is the last word in the entry
                !   + length of blank if previous word was not terminated by a hyphenate sequence
                !   + length of hyphen if current word terminated with a hyphenate sequence
                !
                L_LEN = .TMS_LINE_LENGTH + .TMS_WORD_LENGTH + .CHRSIZ [%C' ']
                    + (IF NOT .HYPH THEN .CHRSIZ [%C' '] ELSE 0)
                    + (IF .HYPHENATE THEN .CHRSIZ [%C'-'] ELSE 0);
                END;

            IF .L_LEN GTR .MAX_LEN
            THEN
                BEGIN
                !
                ! Word will not fit on line
                !
                IF .HYPH
                THEN
                    BEGIN
                    !
                    ! Previous word terminated with a hyphenate sequence.
                    ! Add trailing hyphen.
                    !
                    CH$WCHAR_A (%C'-', LINE_POINTER);
                    INT_LINE_LENGTH = .INT_LINE_LENGTH + 1;
                    EXT_LINE_LENGTH = .EXT_LINE_LENGTH + 1;
                    TMS_LINE_LENGTH = .TMS_LINE_LENGTH + .CHRSIZ [%C'-'];
                    END;

                INS_LINE (.INT_LINE_LENGTH, CH$PTR (RNO_LINE), .EXT_LINE_LENGTH);

                IF .LINE_TYPE NEQ CONT_HEAD THEN LINE_TYPE = (IF .INDLVL EQL 0 THEN ENTRY_W ELSE SUB_W);

                !
                ! Indent new line
                !
                INDENT_LINE (.INDENT + 4);

                !
                ! Verify word will fit on new line
                !
                IF .CMDBLK [NDX$H_FORMAT] EQL DSR
                THEN
                    BEGIN
                    !
                    ! For RUNOFF...
```

```
                        !
                        L_LEN = .EXT_LINE_LENGTH + .EXT_WORD_LENGTH + 1
                            + (IF .HYPHENATE THEN 1 ELSE 0);
                        END
                    ELSE
                        BEGIN
                        !
                        ! For TMS and TEX...
                        !
                        L_LEN = .TMS_LINE_LENGTH + .TMS_WORD_LENGTH + .CHRSIZ [%C' ']
                            + (IF .HYPHENATE THEN .CHRSIZ [%C'-'] ELSE 0);
                        END;

                    IF .L_LEN GTR .MAX_LEN
                    THEN

%IF %BLISS (BLISS32)
%THEN                                                   ! Signal errors in BLISS32

                        SIGNAL (INDEX$_DOESNTFIT, 2, .INT_WORD_LENGTH, CH$PTR (RNO_WORD));

%ELSE                                                   ! Use $XPO_PUT_MSG otherwise

                        $XPO_PUT_MSG (SEVERITY = WARNING,
                            STRING = $STR_CONCAT ('the word ''', (.INT_WORD_LENGTH, CH$PTR (RNO_WORD)),
                                ''' will not fit at current indentation level'));

%FI

                    END
                ELSE
                    BEGIN
                    !
                    ! Word will fit on line.
                    !
                    IF CH$NEQ (1, .BLANKS, .INT_LINE_LENGTH, CH$PTR (RNO_LINE), %C' ')
                    AND (NOT .HYPH)
                    AND .INSERT_BLANK
                    THEN
                        BEGIN
                        !
                        ! Line is non-blank, previous word did not terminate
                        ! with a hyphenate sequence, and we are supposed to insert a blank.
                        !
                        ! Add a blank to the line before adding the word.
                        !
                        CH$WCHAR_A (%C' ', LINE_POINTER);
                        INT_LINE_LENGTH = .INT_LINE_LENGTH + 1;
                        EXT_LINE_LENGTH = .EXT_LINE_LENGTH + 1;
                        TMS_LINE_LENGTH = .TMS_LINE_LENGTH + .CHRSIZ [%C' '];
                        END;
                    END;

                !
                ! Append word to line
                !
                CH$MOVE (.INT_WORD_LENGTH, CH$PTR (RNO_WORD), .LINE_POINTER);
```

```
; 2785      3550  3         LINE_POINTER = CH$PLUS (.LINE_POINTER, .INT_WORD_LENGTH);
; 2786      3551  3         INT_LINE_LENGTH = .INT_LINE_LENGTH + .INT_WORD_LENGTH;
; 2787      3552  3         EXT_LINE_LENGTH = .EXT_LINE_LENGTH + .EXT_WORD_LENGTH;
; 2788      3553  3         TMS_LINE_LENGTH = .TMS_LINE_LENGTH + .TMS_WORD_LENGTH;
; 2789      3554  3
; 2790      3555  3         HYPH = .HYPHENATE;                    ! Save hyphenation status of current word
; 2791      3556  3         INSERT_BLANK = TRUE;
; 2792      3557  2         END;
; 2793      3558  2
; 2794      3559  1     END;



                                          OFFC 00000 FILL_LINE:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11           ; 3209
                 5E        FB40   CE   9E 00002            MOVAB   -1216(SP), SP
                 01   00000000G   EF   B1 00007            CMPW    CMDBLK+4, #1                                    ; 3268
                                  06   12 0000E            BNEQ    1$
                 6E          10   AC   D0 00010            MOVL    MAX_EXT, MAX_LEN                                ; 3273
                                  0D   11 00014            BRB     2$
       50        10          AC   04   C3 00016 1$:        SUBL3   #4, MAX_EXT, R0                                 ; 3288
       6E                    50 00000000G 8F   C5 0001B    MULL3   #TMSSTD, R0, MAX_LEN
                              59      04   AC   D0 00023 2$: MOVL   I_LEN, CHAR_COUNT                               ; 3290
                 04          AE      08   AC   D0 00027    MOVL    I_PTR, TEXT_PTR                                 ; 3291
                                  56   D4 0002C            CLRL    HYPH                                            ; 3293
                 08          AE      1C   AC   D0 0002E    MOVL    L_BLANK, INSERT_BLANK                           ; 3294
                                  59   D5 00033 3$:        TSTL    CHAR_COUNT                                      ; 3296
                                  01   14 00035            BGTR    4$
                                     04 00037              RET
                 57          10   AE   9E 00038 4$:        MOVAB   RNO_WORD, WORD_POINTER                          ; 3302
                                  58   D4 0003C            CLRL    INT_WORD_LENGTH
                                  5A   7C 0003E            CLRQ    TMS_WORD_LENGTH
                             0C   AE   D4 00040            CLRL    HYPHENATE
                 51          04   AE   D0 00043            MOVL    TEXT_PTR, L_PTR                                 ; 3307
                 53               59   D0 00047            MOVL    CHAR_COUNT, L_LEN                               ; 3308
                                  50   D4 0004A            CLRL    I                                               ; 3309
                                  19   11 0004C            BRB     8$
                 52               81   9A 0004E 5$:        MOVZBL  (L_PTR)+, CH                                    ; 3317
                 20               52   D1 00051            CMPL    CH, #32                                         ; 3318
                                  09   13 00054            BEQL    6$
       00000000G 8F               52   D1 00056            CMPL    CH, #TAB
                                  04   12 0005D            BNEQ    7$
                                  53   D7 0005F 6$:        DECL    L_LEN                                           ; 3320
                                  04   11 00061            BRB     8$
                                  51   D7 00063 7$:        DECL    L_PTR                                           ; 3326
                                  04   11 00065            BRB     9$                                              ; 3322
       E3        50               59   F3 00067 8$:        AOBLEQ  CHAR_COUNT, I, 5$                               ; 3309
                 04          AE   51   D0 0006B 9$:        MOVL    L_PTR, TEXT_PTR                                 ; 3331
                 59               53   D0 0006F            MOVL    L_LEN, CHAR_COUNT                               ; 3332
                                  54   D4 00072            CLRL    I                                               ; 3334
                           00E7   31 00074 10$:            BRW     27$
                 50               81   9A 00077 11$:       MOVZBL  (L_PTR)+, CH                                    ; 3342
       00000000G 8F               50   D1 0007A            CMPL    CH, #RINTES                                     ; 3344
                                  50   12 00081            BNEQ    16$
                 50               81   9A 00083            MOVZBL  (L_PTR)+, CH                                    ; 3353
```

```
                    52            81  9A 00086          MOVZBL  (L_PTR)+, OPERAND                        ; 3354
                    54            02  C0 00089          ADDL2   #2, I                                    ; 3355
                    53            03  C2 0008C          SUBL2   #3, L_LEN                                ; 3356
        00000042    8F            50  D1 0008F          CMPL    CH, #66                                  ; 3361
                                  05  12 00096          BNEQ    12$
                    67            2A  90 00098          MOVB    #42, (WORD_POINTER)                      ; 3362
                                  7E  11 0009B          BRB     20$
        0000004F    8F            50  D1 0009D  12$:    CMPL    CH, #79                                  ; 3364
                                  0A  12 000A4          BNEQ    13$
                    87            52  90 000A6          MOVB    OPERAND, (WORD_POINTER)+                 ; 3366
                                  58  D6 000A9          INCL    INT_WORD_LENGTH
                    67            25  90 000AB          MOVB    #37, (WORD_POINTER)                      ; 3367
                                  78  11 000AE          BRB     22$
        00000055    8F            50  D1 000B0  13$:    CMPL    CH, #85                                  ; 3370
                                  05  12 000B7          BNEQ    14$
                    67            26  90 000B9          MOVB    #38, (WORD_POINTER)                      ; 3371
                                  6A  11 000BC          BRB     22$
        0000004E    8F            50  D1 000BE  14$:    CMPL    CH, #78                                  ; 3373
                                  AD  12 000C5          BNEQ    10$
                    2D            52  D1 000C7          CMPL    OPERAND, #45                             ; 3374
                                  A8  12 000CA          BNEQ    10$
              0C    AE            01  D0 000CC          MOVL    #1, HYPHENATE                            ; 3377
                                0091  31 000D0  15$:    BRW     28$                                      ; 3376
                                  53  D7 000D3  16$:    DECL    L_LEN                                    ; 3388
                    20            50  D1 000D5          CMPL    CH, #32                                  ; 3390
                                  F6  13 000D8          BEQL    15$
       00000000G    8F            50  D1 000DA          CMPL    CH, #TAB
                                  ED  13 000E1          BEQL    15$
                    05      14    AC  E9 000E3          BLBC    BOLD, 17$                                ; 3397
                    87            2A  90 000E7          MOVB    #42, (WORD_POINTER)+
                                  58  D6 000EA          INCL    INT_WORD_LENGTH
                    05      18    AC  E9 000EC  17$:    BLBC    UND, 18$                                 ; 3398
                    87            26  90 000F0          MOVB    #38, (WORD_POINTER)+
                                  58  D6 000F3          INCL    INT_WORD_LENGTH
                    20            50  D1 000F5  18$:    CMPL    CH, #32                                  ; 3400
                                  09  19 000F8          BLSS    19$
        0000007E    8F            50  D1 000FA          CMPL    CH, #126
                                  27  15 00101          BLEQ    23$
                    08            50  D1 00103  19$:    CMPL    CH, #8                                   ; 3405
                                  15  13 00106          BEQL    21$
                    87      5F    8F  90 00108          MOVB    #95, (WORD_POINTER)+                     ; 3407
                                  58  D6 0010C          INCL    INT_WORD_LENGTH
                    67            50  90 0010E          MOVB    CH, (WORD_POINTER)
                                  5B  D6 00111          INCL    EXT_WORD_LENGTH
                    5A 00000000GFF40  C0 00113          ADDL2   @CHRSIZ[CH], TMS_WORD_LENGTH
                                  3D  11 0011B  20$:    BRB     26$                                      ; 3405
                    87      5F    8F  90 0011D  21$:    MOVB    #95, (WORD_POINTER)+                     ; 3409
                                  58  D6 00121          INCL    INT_WORD_LENGTH
                    67            50  90 00123          MOVB    CH, (WORD_POINTER)
                                  5B  D7 00126          DECL    EXT_WORD_LENGTH
                                  30  11 00128  22$:    BRB     26$                                      ; 3405
                    26            50  D1 0012A  23$:    CMPL    CH, #38                                  ; 3414
                                  18  13 0012D          BEQL    24$
                    2A            50  D1 0012F          CMPL    CH, #42                                  ; 3415
                                  13  13 00132          BEQL    24$
        0000005F    8F            50  D1 00134          CMPL    CH, #95                                  ; 3416
                                  0A  13 0013B          BEQL    24$
```

```
                          2E           50 D1 0013D          CMPL    CH, #46                                  ; 3417
                                       05 13 00140          BEQL    24$                                      ;
                          25           50 D1 00142          CMPL    CH, #37                                  ; 3418
                                       06 12 00145          BNEQ    25$                                      ;
                          87        5F 8F 90 00147   24$:   MOVB    #95, (WORD_POINTER)+                     ; 3420
                                       58 D6 0014B          INCL    INT_WORD_LENGTH                          ;
                          67           50 90 0014D   25$:   MOVB    CH, (WORD_POINTER)                       ; 3422
                          5A 00000000GFF40 C0 00150         ADDL2   @CHRSIZ[CH], TMS_WORD_LENGTH             ;
                                       5B D6 00158          INCL    EXT_WORD_LENGTH                          ; 3420
                                       57 D6 0015A   26$:   INCL    WORD_POINTER                             ; 3407
                                       58 D6 0015C          INCL    INT_WORD_LENGTH                          ;
      FF13       54        01          59 F1 0015E   27$:   ACBL    CHAR_COUNT, #1, I, 11$                   ; 3334
                       04  AE          51 D0 00164   28$:   MOVL    L_PTR, TEXT_PTR                          ; 3426
                           59          53 D0 00168          MOVL    L_LEN, CHAR_COUNT                        ; 3427
                           54          56 D2 0016B          MCOML   HYPH, R4                                 ; 3442
                           01 00000000G EF B1 0016E         CMPW    CMDBLK+4, #1                             ; 3432
                                       27 12 00175          BNEQ    33$                                      ;
                 50        5B 00000000' EF C1 00177         ADDL3   EXT_LINE_LENGTH, EXT_WORD_LENGTH, R0     ; 3441
                           05          54 E9 0017F          BLBC    R4, 29$                                  ; 3442
                           51          01 D0 00182          MOVL    #1, R1                                   ;
                                       02 11 00185          BRB     30$                                      ;
                                       51 D4 00187   29$:   CLRL    R1                                       ;
                           50          51 C0 00189   30$:   ADDL2   R1, R0                                   ;
                           05      0C  AE E9 0018C          BLBC    HYPHENATE, 31$                           ;
                           51          01 D0 00190          MOVL    #1, R1                                   ;
                                       02 11 00193          BRB     32$                                      ;
                                       51 D4 00195   31$:   CLRL    R1                                       ;
                           53      01 A140 9E 00197  32$:   MOVAB   1(R1)[R0], L_LEN                         ;
                                       34 11 0019C          BRB     38$                                      ; 3432
                 51        5A 00000000' EF C1 0019E  33$:   ADDL3   TMS_LINE_LENGTH, TMS_WORD_LENGTH, R1     ; 3452
                           50 00000000G EF D0 001A6         MOVL    CHRSIZ, R0                               ;
                           51    0080  C0 C0 001AD          ADDL2   128(R0), R1                              ;
                           07          54 E9 001B2          BLBC    R4, 34$                                  ; 3453
                           52    0080  C0 D0 001B5          MOVL    128(R0), R2                              ;
                                       02 11 001BA          BRB     35$                                      ;
                                       52 D4 001BC   34$:   CLRL    R2                                       ;
                           51          52 C0 001BE   35$:   ADDL2   R2, R1                                   ;
                           07      0C  AE E9 001C1          BLBC    HYPHENATE, 36$                           ; 3454
                           50    00B4  C0 D0 001C5          MOVL    180(R0), R0                              ;
                                       02 11 001CA          BRB     37$                                      ;
                                       50 D4 001CC   36$:   CLRL    R0                                       ;
                 53        51          50 C1 001CE   37$:   ADDL3   R0, R1, L_LEN                            ;
                           6E          53 D1 001D2   38$:   CMPL    L_LEN, MAX_LEN                           ; 3457
                                       03 14 001D5          BGTR    39$                                      ;
                                  00D2 31 001D7             BRW     50$                                      ;
                           29          56 E9 001DA   39$:   BLBC    HYPH, 40$                                ; 3463
         00000000'         FF          2D 90 001DD          MOVB    #45, @LINE_POINTER                       ; 3470
                      00000000'        EF D6 001E4          INCL    LINE_POINTER                             ;
                      00000000'        EF D6 001EA          INCL    INT_LINE_LENGTH                          ; 3471
                      00000000'        EF D6 001F0          INCL    EXT_LINE_LENGTH                          ; 3472
                           50 00000000G EF D0 001F6         MOVL    CHRSIZ, R0                               ; 3473
         00000000'         EF    00B4  C0 C0 001FD          ADDL2   180(R0), TMS_LINE_LENGTH                 ;
                      00000000'        EF DD 00206   40$:   PUSHL   EXT_LINE_LENGTH                          ; 3476
                      00000000'        EF 9F 0020C          PUSHAB  RNO_LINE                                 ;
                      00000000'        EF DD 00212          PUSHL   INT_LINE_LENGTH                          ;
            F997           CF          03 FB 00218          CALLS   #3, INS_LINE                             ;
                           0B 00000000' EF D1 0021D         CMPL    LINE_TYPE, #11                           ; 3478
```

```
                                        17 13 00224       BEQL    43$                                          ;
                          00000000G     EF D5 00226       TSTL    INDLVL                                       ;
                                        05 12 0022C       BNEQ    41$                                          ;
                     50                 06 D0 0022E       MOVL    #6, R0                                       ;
                                        03 11 00231       BRB     42$                                          ;
                     50                 09 D0 00233 41$:  MOVL    #9, R0                                       ;
                00000000'               EF 50 D0 00236 42$:  MOVL    R0, LINE_TYPE                             ;
           7E          0C               AC 04 C1 0023D 43$:  ADDL3   #4, INDENT, -(SP)                         ; 3483
                00000000V               EF 01 FB 00242       CALLS   #1, INDENT_LINE                           ;
                     01 00000000G       EF B1 00249       CMPW    CMDBLK+4, #1                                 ; 3488
                                        1A 12 00250       BNEQ    46$                                          ;
           50        5B 00000000'       EF C1 00252       ADDL3   EXT_LINE_LENGTH, EXT_WORD_LENGTH, R0          ; 3494
                     05         0C      AE E9 0025A       BLBC    HYPHENATE, 44$                               ; 3495
                     51                 01 D0 0025E       MOVL    #1, R1                                       ;
                                        02 11 00261       BRB     45$                                          ;
                                        51 D4 00263 44$:  CLRL    R1                                           ;
           53                01 A140    9E 00265 45$:  MOVAB   1(R1)[R0], L_LEN                                ;
                                        25 11 0026A       BRB     49$                                          ; 3488
           51        5A 00000000'       EF C1 0026C 46$:  ADDL3   TMS_LINE_LENGTH, TMS_WORD_LENGTH, R1          ; 3502
                     50 00000000G       EF D0 00274       MOVL    CHRSIZ, R0                                   ;
                     51       0080      C0 C0 0027B       ADDL2   128(R0), R1                                  ;
                     07         0C      AE E9 00280       BLBC    HYPHENATE, 47$                               ; 3503
                     50       00B4      C0 D0 00284       MOVL    180(R0), R0                                  ;
                                        02 11 00289       BRB     48$                                          ;
                                        50 D4 0028B 47$:  CLRL    R0                                           ;
           53        51                 50 C1 0028D 48$:  ADDL3   R0, R1, L_LEN                                ;
                     6E                 53 D1 00291 49$:  CMPL    L_LEN, MAX_LEN                               ; 3506
                                        5A 15 00294       BLEQ    51$                                          ;
                                10      AE 9F 00296       PUSHAB  RNO_WORD                                     ; 3512
                                        58 DD 00299       PUSHL   INT_WORD_LENGTH                              ;
                                        02 DD 0029B       PUSHL   #2                                           ;
                          00000000G     8F DD 0029D       PUSHL   #DSRINDEX$_DOESNTFIT                         ;
                00000000G   00          04 FB 002A3       CALLS   #4, LIB$SIGNAL                               ;
                                        44 11 002AA       BRB     51$                                          ; 3457
00000000'  EF          20 00000000'  FF 01 2D 002AC 50$:  CMPC5   #1, @BLANKS, #32, INT_LINE_LENGTH, RNO_LINE  ; 3528
                          00000000'     EF    002B9                                                    ;
                                        30 13 002BE       BEQL    51$                                          ;
                     2D                 54 E9 002C0       BLBC    R4, 51$                                      ; 3529
                     29         08      AE E9 002C3       BLBC    INSERT_BLANK, 51$                            ; 3530
                00000000'               FF 20 90 002C7       MOVB    #32, @LINE_POINTER                        ; 3539
                          00000000'     EF D6 002CE       INCL    LINE_POINTER                                 ;
                          00000000'     EF D6 002D4       INCL    INT_LINE_LENGTH                              ; 3540
                          00000000'     EF D6 002DA       INCL    EXT_LINE_LENGTH                              ; 3541
                     50 00000000G       EF D0 002E0       MOVL    CHRSIZ, R0                                   ; 3542
                00000000'     EF 0080   C0 C0 002E7       ADDL2   128(R0), TMS_LINE_LENGTH                     ;
          00000000'  FF        10       AE 58 28 002F0 51$:  MOVC3   INT_WORD_LENGTH, RNO_WORD, @LINE_POINTER  ; 3549
                00000000'               EF 58 C0 002F9       ADDL2   INT_WORD_LENGTH, LINE_POINTER             ; 3550
                00000000'               EF 58 C0 00300       ADDL2   INT_WORD_LENGTH, INT_LINE_LENGTH          ; 3551
                00000000'               EF 5B C0 00307       ADDL2   EXT_WORD_LENGTH, EXT_LINE_LENGTH          ; 3552
                00000000'               EF 5A C0 0030E       ADDL2   TMS_WORD_LENGTH, TMS_LINE_LENGTH          ; 3553
                     56         0C      AE D0 00315       MOVL    HYPHENATE, HYPH                              ; 3555
                08   AE                 01 D0 00319       MOVL    #1, INSERT_BLANK                             ; 3556
                                FD13    31 0031D          BRW     3$                                           ; 3296
                                        04 00320          RET                                                  ; 3559

; Routine Size:  801 bytes,    Routine Base:  $CODE$ + 0F3C
```

```
%SBTTL 'INDENT_LINE -- Indent new line'
ROUTINE INDENT_LINE (INDENT) : NOVALUE =
!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine clears out RNO_LINE and fills it with the number
!       of blanks specified by INDENT.
!
! FORMAL PARAMETERS:
!
!       INDENT - Number of blanks to indent line
!
! IMPLICIT INPUTS:
!
!       CMDBLK          - Command line information block
!       TMSSTD          - Standard TMS character size
!       MSPACE          - Size of /m sequence
!
! IMPLICIT OUTPUTS:
!
!       EXT_LINE_LENGTH = .INDENT
!       INT_LINE_LENGTH = .INDENT
!       LINE_POINTER    = Pointer to next character position in RNO_LINE
!       TMS_LINE_LENGTH = Length of line in TMS relative units
!
! ROUTINE VALUE:
! COMPLETION CODES:
!
!       None
!
! SIDE EFFECTS:
!
!       None
!--
    BEGIN
    LOCAL
        IND;

    IND = .INDENT;
    CLR_RNO_LINE ();
    EXT_LINE_LENGTH = .IND;

    IF  (.IND GEQ .CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID])
    AND ((.CMDBLK [NDX$H_FORMAT] EQL TMS11_A) OR (.CMDBLK [NDX$H_FORMAT] EQL TMS11_E))
    THEN
        BEGIN
        !
        ! TMS11 output and padding out to second column
        ! for /MASTER/LAYOUT=SEPARATE
        !
        CH$WCHAR_A (TAB, LINE_POINTER);         ! Use tabs to get to second column
        CH$WCHAR_A (TAB, LINE_POINTER);
        INT_LINE_LENGTH = .INT_LINE_LENGTH + 2;

        IND = .IND - .CMDBLK [NDX$G_COLUMN_WID] - .CMDBLK [NDX$G_GUTTER_WID];
```

```
; 2853      3617  3             TMS_LINE_LENGTH = (.CMDBLK [NDX$G_COLUMN_WID] + .CMDBLK [NDX$G_GUTTER_WID]) * TMSSTD
; 2854      3618  3                 + (.IND / 2) * MSPACE;
; 2855      3619  3             END
; 2856      3620  2         ELSE
; 2857      3621  2             TMS_LINE_LENGTH = (.IND / 2) * MSPACE;
; 2858      3622  2
; 2859      3623  2         CH$FILL (%C' ', .IND, .LINE_POINTER);
; 2860      3624  2         INT_LINE_LENGTH = .INT_LINE_LENGTH + .IND;
; 2861      3625  2         LINE_POINTER = CH$PLUS (CH$PTR (RNO_LINE), .INT_LINE_LENGTH);
; 2862      3626  1         END;


                                             03FC 00000 INDENT_LINE:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9                     ; 3561
                     59 00000000G    8F    D0 00002     MOVL    #MSPACE, R9                                      ;
                     58 00000000G    EF    9E 00009     MOVAB   CMDBLK+16, R8                                    ;
                     57 00000000'    EF    9E 00010     MOVAB   LINE_POINTER, R7                                 ;
                     56        04    AC    D0 00017     MOVL    INDENT, IND                                      ; 3599
                     67        04    A7    9E 0001B     MOVAB   RNO_LINE, LINE_POINTER                           ; 3600
                           04B4 C7   7C 0001F           CLRQ    INT_LINE_LENGTH                                  ;
                           04BC C7   D4 00023           CLRL    TMS_LINE_LENGTH                                  ;
              04B8    C7             56    D0 00027     MOVL    IND, EXT_LINE_LENGTH                             ; 3601
          51    FC    A8             68    C1 0002C     ADDL3   CMDBLK+16, CMDBLK+12, R1                         ; 3603
                      51             56    D1 00031     CMPL    IND, R1                                          ;
                      3E             19       00034     BLSS    2$                                               ;
                      02       F4    A8    B1 00036     CMPW    CMDBLK+4, #2                                     ; 3604
                      06             13       0003A     BEQL    1$                                               ;
                      03       F4    A8    B1 0003C     CMPW    CMDBLK+4, #3                                     ;
                      32             12       00040     BNEQ    2$                                               ;
              00    B7        00G    8F    90 00042 1$: MOVB    #TAB, @LINE_POINTER                              ; 3611
                                     67    D6 00047     INCL    LINE_POINTER                                     ;
              00    B7        00G    8F    90 00049     MOVB    #TAB, @LINE_POINTER                              ; 3612
                                     67    D6 0004E     INCL    LINE_POINTER                                     ;
              04B4  C7               02    C0 00050     ADDL2   #2, INT_LINE_LENGTH                              ; 3613
          50        56         FC    A8    C3 00055     SUBL3   CMDBLK+12, IND, R0                               ; 3615
          56        50               68    C3 0005A     SUBL3   CMDBLK+16, R0, IND                               ;
                    51 00000000G     8F    C4 0005E     MULL2   #TMSSTD, R1                                      ; 3617
          50        56               02    C7 00065     DIVL3   #2, IND, R0                                      ; 3618
                    50               59    C4 00069     MULL2   R9, R0                                           ;
     04BC C7        51               50    C1 0006C     ADDL3   R0, R1, TMS_LINE_LENGTH                          ;
                                     0A    11 00072     BRB     3$                                               ; 3603
          50        56               02    C7 00074 2$: DIVL3   #2, IND, R0                                      ; 3621
     04BC C7        50               59    C5 00078     MULL3   R9, R0, TMS_LINE_LENGTH                          ;
56        20        6E               00    2C 0007E 3$: MOVC5   #0, (SP), #32, IND, @LINE_POINTER                ; 3623
                               00    B7       00083                                                              ;
              04B4  C7               56    C0 00085     ADDL2   IND, INT_LINE_LENGTH                             ; 3624
                    50         04    A7    9E 0008A     MOVAB   RNO_LINE, R0                                     ; 3625
                    67   04B4 D740         9E 0008E     MOVAB   @INT_LINE_LENGTH[R0], LINE_POINTER               ;
                                           04 00094     RET                                                      ; 3626

; Routine Size:  149 bytes,    Routine Base:  $CODE$ + 125D
```

```
1 %SBTTL 'PADLIN -- Pad line to column width'
1 GLOBAL ROUTINE PADLIN (INT_LEN, PTR, EXT_LEN, DSC) : NOVALUE =
1 !++
1 !
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This routine pads the input line to the column width
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       INT_LEN - Internal length of line
1 !       PTR     - CH$PTR to line
1 !       EXT_LEN - Number of print positions used by line
1 !       DSC     - Address of string descriptor where padded line is returned
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       CMDBLK [NDX$G_GUTTER_WID] - Gutter width
1 !       CMDBLK [NDX$G_COLUMN_WID] - Column width
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       None
1 !
1 ! ROUTINE VALUE:
1 ! COMPLETION CODES:
1 !
1 !       None
1 !
1 ! SIDE EFFECTS:
1 !
1 !       None
1 !--
2    BEGIN
2
2    IF .CMDBLK [NDX$H_FORMAT] NEQ DSR              ! For TMS and TEX output
3    OR (.CMDBLK [NDX$H_LAYOUT] NEQ TWO_COLUMN)     ! For all output except TWO_COLUMN
3    OR (.EXT_LEN EQL 0)                            ! Or previously padded output
3    OR (.EXT_LEN GTR .CMDBLK [NDX$G_COLUMN_WID])
2    THEN
2        !
2        ! Line does not need padding. Just copy it.
2        !
3        $STR_COPY (STRING = (.INT_LEN, .PTR), TARGET = .DSC)
2    ELSE
3        BEGIN
3        !
3        ! Compute output length =
3        !         internal length + (column width - external length) + gutter width
3        !
3        LOCAL
3            LINE : VECTOR [CH$ALLOCATION (1200)],
3            LEN;
3
3        LEN = .INT_LEN + (.CMDBLK [NDX$G_COLUMN_WID] - .EXT_LEN) + .CMDBLK [NDX$G_GUTTER_WID];
3
3        CH$COPY (.INT_LEN, .PTR, %C' ', .LEN, CH$PTR (LINE));
```

```
; 2921          3684  3          $STR_COPY (STRING = (.LEN, CH$PTR (LINE)), TARGET = .DSC);
; 2922          3685  2          END;
; 2923          3686  2
; 2924          3687  1      END;


                                       01FC 00000          .ENTRY  PADLIN, Save R2,R3,R4,R5,R6,R7,R8          ; 3628
                   58 00000000G    EF    9E 00002          MOVAB   STR$FAILURE, R8                            ;
                   57 00000000G    EF    9E 00009          MOVAB   CMDBLK+4, R7                               ;
                   5E        FB48  CE    9E 00010          MOVAB   -1208(SP), SP                              ;
                   01              67    B1 00015          CMPW    CMDBLK+4, #1                               ; 3662
                                   12    12 00018          BNEQ    1$                                         ;
                   01          02  A7    B1 0001A          CMPW    CMDBLK+6, #1                               ; 3663
                                   0C    12 0001E          BNEQ    1$                                         ;
                               0C  AC    D5 00020          TSTL    EXT_LEN                                    ; 3664
                                   07    13 00023          BEQL    1$                                         ;
              08   A7          0C  AC    D1 00025          CMPL    EXT_LEN, CMDBLK+12                         ; 3665
                                   1E    15 0002A          BLEQ    2$                                         ;
              F8   AD          04  AC    B0 0002C 1$:      MOVW    INT_LEN, $STR$STRING                       ; 3670
              FA   AD              0E    90 00031          MOVB    #14, $STR$STRING+2                         ;
              FB   AD              01    90 00035          MOVB    #1, $STR$STRING+3                          ;
              FC   AD          08  AC    D0 00039          MOVL    PTR, $STR$STRING+4                         ;
                                   58    DD 0003E          PUSHL   R8                                         ;
                                   7E    D4 00040          CLRL    -(SP)                                      ;
                               10  AC    DD 00042          PUSHL   DSC                                        ;
                               F8  AD    9F 00045          PUSHAB  $STR$STRING                                ;
                                   32    11 00048          BRB     3$                                         ;
         50   08   A7          0C  AC    C3 0004A 2$:      SUBL3   EXT_LEN, CMDBLK+12, R0                     ; 3681
                   50          04  AC    C0 00050          ADDL2   INT_LEN, R0                                ;
         56        50          0C  A7    C1 00054          ADDL3   CMDBLK+16, R0, LEN                         ;
 56      20   08   BC          04  AC    2C 00059          MOVC5   INT_LEN, @PTR, #32, LEN, LINE              ; 3683
                               08  AE       00060                                                             ;
                   6E              56    B0 00062          MOVW    LEN, $STR$STRING                           ; 3684
              02   AE              0E    90 00065          MOVB    #14, $STR$STRING+2                         ;
              03   AE              01    90 00069          MOVB    #1, $STR$STRING+3                          ;
              04   AE          08  AE    9E 0006D          MOVAB   LINE, $STR$STRING+4                        ;
                                   58    DD 00072          PUSHL   R8                                         ;
                                   7E    D4 00074          CLRL    -(SP)                                      ;
                               10  AC    DD 00076          PUSHL   DSC                                        ;
                               0C  AE    9F 00079          PUSHAB  $STR$STRING                                ;
                                   7E    D4 0007C 3$:      CLRL    -(SP)                                      ;
         00000000G   EF            05    FB 0007E          CALLS   #5, XST$COPY                               ;
                                         04 00085          RET                                                ; 3687

; Routine Size:  134 bytes,    Routine Base:  $CODE$ + 12F2


; 2925          3688  1
; 2926          3689  1 END                                 !End of module
; 2927          3690  0 ELUDOM
```

```
                                                                .EXTRN  LIB$SIGNAL

;                                 PSECT SUMMARY
;
;       Name                      Bytes                        Attributes
;
;  $PLIT$                           972  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $OWN$                           1428  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                          4984  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       163        27         252        00:00.1


;                                 COMMAND QUALIFIERS

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

; Size:           4984 code + 2400 data bytes
; Run Time:          02:25.3
; Elapsed Time:      04:02.8
; Lines/CPU Min:      1523
; Lexemes/CPU-Min: 64870
; Memory Used:  556 pages
; Compilation Complete
```

0343 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NDXDAT
LIS

LSTOPS
LIS

LOEMPH
LIS

LP1
LIS

MAKNDX
LIS

NATE
LIS

NDXFMT
LIS

LOHORI
LIS

LOVERT
LIS

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

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

NDXMSG
LIS

NDXOUT
LIS

NDXINI
LIS