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

```
_$2

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

RLI
RNO
RNO
RTY
SAV
STR
STR
STR
STR
STR
STR

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

FILEID**LOVERT

N 5

```
LL            000000    VV      VV  EEEEEEEEEE  RRRRRRRR    TTTTTTTTTT
LL            000000    VV      VV  EEEEEEEEEE  RRRRRRRR    TTTTTTTTTT
LL          00      00  VV      VV  EE          RR      RR      TT
LL          00      00  VV      VV  EE          RR      RR      TT
LL          00      00  VV      VV  EE          RR      RR      TT
LL          00      00  VV      VV  EE          RR      RR      TT
LL          00      00  VV      VV  EEEEEEEE    RRRRRRRR        TT
LL          00      00  VV      VV  EEEEEEEE    RRRRRRRR        TT
LL          00      00  VV      VV  EE          RR  RR          TT
LL          00      00  VV      VV  EE          RR  RR          TT
LL          00      00    VV  VV    EE          RR    RR        TT          ....
LL          00      00    VV  VV    EE          RR    RR        TT          ....
LLLLLLLLLL    000000        VV      EEEEEEEEEE  RR      RR      TT          ....
LLLLLLLLLL    000000        VV      EEEEEEEEEE  RR      RR      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
```

```
;     1          0001  0 %TITLE 'Line output (vertical motion)'
;     2          0002  0 MODULE LOVERT ( IDENT = 'V04-000'
;     3        P 0003  0                 %BLISS32[,ADDRESSING_MODE(EXTERNAL =    LONG_RELATIVE,
;     4          0004  0                                          NONEXTERNAL = LONG_RELATIVE)]
;     5          0005  0                 ) =
;     6          0006  1 BEGIN
;     7          0007  1 !
;     8          0008  1 !*****************************************************************************
;     9          0009  1 !*                                                                           *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                      *
;    13          0013  1 !*                                                                           *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    19          0019  1 !*  TRANSFERRED.                                                              *
;    20          0020  1 !*                                                                           *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    23          0023  1 !*  CORPORATION.                                                              *
;    24          0024  1 !*                                                                           *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*                                                                           *
;    29          0029  1 !*****************************************************************************
;    30          0030  1 !
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT: Translation from intermediate format to final output.
;    35          0035  1 !
;    36          0036  1 !
;    37          0037  1 ! ENVIRONMENT: Transportable
;    38          0038  1 !
;    39          0039  1 ! AUTHOR: K. A. Dawson   CREATION DATE: December 1983
;    40          0040  1 !
```

```
;    42          0041  1 %SBTTL 'Revision History'
;    43          0042  1 !
;    44          0043  1 ! MODIFIED BY:
;    45          0044  1 !
;    46          0045  1 !       013     REM00013        Ray Marshall    30-Nov-1983
;    47          0046  1 !               Initialize the local BRANCH within routine LOUT to eliminate
;    48          0047  1 !                 informational message generated by the compiler.
;    49          0048  1 !
;    50          0049  1 !       012     KFA00012        Ken Alden       05-Oct-1983
;    51          0050  1 !               Removed left margin fix from #11 and added checking for
;    52          0051  1 !                 sca_margin_pad.
;    53          0052  1 !
;    54          0053  1 !       011     KFA00011        Ken Alden       27-Jul-1983
;    55          0054  1 !               For DSRPLUS: Added a few comments and included the left margin
;    56          0055  1 !                 in the figuring of the special cased CREF (only) line.  This
;    57          0056  1 !                 is found in the first line of LOUT.
;    58          0057  1 !
;    59          0058  1 !       010     KFA00010        Ken Alden       8-Jul-1983
;    60          0059  1 !               Just changed the conditional for dsrplus so the cref stuff
;    61          0060  1 !                 would not be seen be DSR.
;    62          0061  1 !
;    63          0062  1 !       009     KFA00009        Ken Alden       30-Jun-1983
;    64          0063  1 !               Logic that was put into this routine in #8 was not totally
;    65          0064  1 !                 correct so a cref was getting held up for quite some time.
;    66          0065  1 !               Output of crefs is handled in this module and LOUT1.  Crefs,
;    67          0066  1 !                 however, will increment the tsf_int_hl and there may not
;    68          0067  1 !                 be anything on the line.  Therefore we test tsf_ext_hl so
;    69          0068  1 !                 see if there is any text.
;    70          0069  1 !
;    71          0070  1 !       008     KFA00008        Ken Alden       29-Jun-1983
;    72          0071  1 !               Lout now checks tsf_cref_data before returning.
;    73          0072  1 !
;    74          0073  1 !       007     KFA00007        Ken Alden       29-Jun-1983
;    75          0074  1 !               In the very weird cases where the mra only contains vertical
;    76          0075  1 !                 rintes sequences and any number of crefs, LOUT1 is NOT
;    77          0076  1 !                 called to dump the crefs since going over to LOUT1 meant never
;    78          0077  1 !                 returning without a crlf getting output.
;    79          0078  1 !
;    80          0079  1 !       006     KFA00006        Ken Alden       28-Jun-1983
;    81          0080  1 !               Calling lout1 now if tsf_cref_data GTR 0.
;    82          0081  1 !
;    83          0082  1 !       005     REM00005        Ray Marshall    17-Jun-1983
;    84          0083  1 !               Remove call to OUTCREF because it's been moved to LOUT1 (in
;    85          0084  1 !                 module LOHORI) based on a new escape sequence in the MRA.
;    86          0085  1 !
;    87          0086  1 !       004     KAD00004        Keith Dawson    3-Jun-1983
;    88          0087  1 !               Call OUTCREF from here (not from LOHORI), in order to
;    89          0088  1 !                 assure that cref records are written to the .BRN even
;    90          0089  1 !                 when Quick is set.
;    91          0090  1 !
;    92          0091  1 !       003     KAD00003        Keith Dawson    9-May-1983
;    93          0092  1 !               Remove support for .DX, .PX (remove references to ASGXTN).
;    94          0093  1 !
;    95          0094  1 !       002     KFA000021       Ken Alden       29-Apr-1983
;    96          0095  1 !               Added code for START_ODD chapters.
;    97          0096  1 !
;    98          0097  1 !--
```

```
;  100          0098  1 %SBTTL 'Module Level Declarations'
;  101          0099  1
;  102          0100  1 !
;  103          0101  1 ! TABLE OF CONTENTS:
;  104          0102  1 !
;  105          0103  1
;  106          0104  1 REQUIRE 'REQ:RNODEF';                   ! RUNOFF variant definitions
;  107          0235  1
;  108          0236  1 FORWARD ROUTINE
;  109          0237  1     JUSTF : NOVALUE,
;  110          0238  1     LOUT : NOVALUE,
;  111          0239  1     find_next_dot : NOVALUE,
;  112          0240  1     ascfin : NOVALUE;
;  113          0241  1
;  114          0242  1 !
;  115          0243  1 ! INCLUDE FILES:
;  116          0244  1 !
;  117          0245  1 LIBRARY 'NXPORT:XPORT';                 ! XPORT Library
;  118          0246  1
;  119        U 0247  1 %IF DSRPLUS %THEN
;  120        U 0248  1 LIBRARY 'REQ:DPLLIB';                  ! DSRPLUS BLISS Library
;  121          0249  1 %ELSE
;  122          0250  1 LIBRARY 'REQ:DSRLIB';                  ! DSR BLISS Library
;  123          0251  1 %FI
;  124          0252  1 !
;  125          0253  1 ! MACROS:
;  126          0254  1 !
;  127          0255  1
;  128          0256  1 MACRO
;  129        M 0257  1     exchange (a, b) =
;  130        M 0258  1         BEGIN
;  131        M 0259  1         LOCAL h;
;  132        M 0260  1         h = .(a); a = .(b); b = .h;
;  133        M 0261  1         END
;  134          0262  1     %;
;  135          0263  1
;  136          0264  1 MACRO
;  137        M 0265  1     save_xtn (page_ref, xtn) =
;  138        M 0266  1         BEGIN
;  139        M 0267  1 !!      IF NOT .gca_bix
;  140        M 0268  1 !!      THEN
;  141        M 0269  1 !!          asgxtn (page_ref, xtn);
;  142        M 0270  1         IF .gca_bix
;  143        M 0271  1         THEN
;  144        M 0272  1             putxtn (page_ref, xtn);
;  145        M 0273  1         END
;  146          0274  1     %;
;  147          0275  1
;  148          0276  1 !
;  149          0277  1 ! EQUATED SYMBOLS:
;  150          0278  1 !
;  151          0279  1
;  152          0280  1 EXTERNAL LITERAL
;  153          0281  1     rintes : UNSIGNED (8);
;  154          0282  1
;  155          0283  1 !
;  156          0284  1 ! OWN STORAGE:
```

```
;  157          0285  1 !
;  158          0286  1
;  159          0287  1 !
;  160          0288  1 ! EXTERNAL REFERENCES:
;  161          0289  1 !
;  162          0290  1
;  163          0291  1 EXTERNAL
;  164          0292  1     fnct : fnct_definition,
;  165          0293  1     gca : gca_definition,
;  166        U 0294  1 %IF DSRPLUS %THEN
;  167        U 0295  1     sca : sca_definition,
;  168          0296  1 %FI
;  169          0297  1     hct : hct_definition,
;  170          0298  1     mra : ref fixed_string,
;  171          0299  1     tsf : tsf_definition,
;  172          0300  1     npagen : page_definition,
;  173          0301  1     pagen : page_definition,
;  174          0302  1     phan : phan_definition;
;  175          0303  1
;  176          0304  1 EXTERNAL LITERAL                        ! Error messages
;  177          0305  1     rnfile;
;  178          0306  1
;  179          0307  1 EXTERNAL ROUTINE
;  180          0308  1     cskipl,     erms,
;  181        U 0309  1 %IF DSRPLUS %THEN
;  182        U 0310  1     outcref,
;  183          0311  1 %FI
;  184          0312  1     lout1,      newpag,
;  185          0313  1     putxtn,     tpr,    uskipl;
```

```
;  187          0314  1 %SBTTL 'LOUT -- generate output line'
;  188          0315  1 GLOBAL ROUTINE lout : NOVALUE =
;  189          0316  1
;  190          0317  1 !++
;  191          0318  1 ! FUNCTIONAL DESCRIPTION:
;  192          0319  1 !
;  193          0320  1 !       LOUT accepts as input a line of text as encoded by SCANT and described
;  194          0321  1 !       by the TSF data structure.  If GCA_SKIP_OUT is true, the line is
;  195          0322  1 !       ignored; otherwise LOUT generates the necessary device control
;  196          0323  1 !       sequences that cause the line to be printed correctly and leaves its
;  197          0324  1 !       output in the FRA to be output by OUTLIN.
;  198          0325  1 !
;  199          0326  1 ! FORMAL PARAMETERS:    None
;  200          0327  1 !
;  201          0328  1 ! IMPLICIT INPUTS:      None
;  202          0329  1 !
;  203          0330  1 ! IMPLICIT OUTPUTS:
;  204          0331  1 !
;  205          0332  1 !       PHAN_FIGURE     -
;  206          0333  1 !       PHAN_TOP_PAGE   -
;  207          0334  1 !       PHAN_FORM_PEND  - Controls whether a formfeed is written to output
;  208          0335  1 !                         (if nonzero).
;  209          0336  1 !
;  210          0337  1 ! ROUTINE VALUE:
;  211          0338  1 ! COMPLETION CODES:     None
;  212          0339  1 !
;  213          0340  1 ! SIDE EFFECTS:        None
;  214          0341  1 !
;  215          0342  1 !--
;  216          0343  1
;  217          0344  2     BEGIN
;  218          0345  2
;  219          0346  2     LOCAL
;  220          0347  2         branch : INITIAL (false),
;  221          0348  2         ptr_copy,
;  222          0349  2         tsf_phregs : REF VECTOR [tsf_nregs];
;  223          0350  2
;  224          0351  2         !                       Nothing to output??
;  225          0352  2         ! TSF_INT_?L NEQ 0     means either paper control codes and/or some text to
;  226          0353  2         !                       be output.
;  227          0354  2         ! TSF_FIRST_XTN NEQ 0  means that there is an index entry referring to
;  228          0355  2         !                       something on the current page;  it is possible for just
;  229          0356  2         !                       this item to be set.  That can happen if, for example,
;  230          0357  2         !                       the very first command in the file is a .INDEX command,
;  231          0358  2         !                       and a .SKIP command follows that.
;  232          0359  2
;  233        U 0360  2 %IF DSRPLUS %THEN
;  234        U 0361  2         !
;  235        U 0362  2         ! Text to do (processing needed in LOUT1)?
;  236        U 0363  2         ! If the following branch is taken, then the only 'text' in the MRA is
;  237        U 0364  2         ! cref information.  In some instances, there also may be a few spaces
;  238        U 0365  2         ! at the beginning of the line.  This is from the left margin padding in
;  239        U 0366  2         ! FCIMRA.  If this is the case, then we just ignore the line since lout
;  240        U 0367  2         ! would normally exit at this point since FCIMRA never would have
;  241        U 0368  2         ! been called from CREF and the padding never would have occured.
;  242        U 0369  2         !
;  243        U 0370  2         IF  .tsf_int_hl - (3 * .tsf_cref_count) EQL 0
```

```
;  244     U 0371  2                            OR
;  245     U 0372  2          (.tsf_int_hl - (3 * .tsf_cref_count) EQL .sca_margin_pad
;  246     U 0373  2                            AND
;  247     U 0374  2          .tsf_ext_hl EQL .sca_margin_pad)
;  248     U 0375  2      THEN
;  249     U 0376  2          BEGIN
;  250     U 0377  2          WHILE .tsf_cref_data NEQ 0 DO outcref (); !Dump all pending crefs.
;  251     U 0378  2          tsf_int_hl = .tsf_int_hl - (3 * .tsf_cref_count);
;  252     U 0379  2          tsf_cref_count = 0;                 ! Clear this flag since all crefs are done
;  253     U 0380  2          IF (.tsf_int_vl EQL 0)              !  Vertical positioning?
;  254     U 0381  2              AND (.tsf_first_xtn EQL 0)      !  Indexing to do?
;  255     U 0382  2              AND (.tsf_footw EQL 0)          !  Footnotes attached to this line?
;  256     U 0383  2          THEN
;  257     U 0384  2              RETURN;                         ! THEN do absolutely nothing.
;  258     U 0385  2          END;
;  259       0386  2  %ELSE
;  260       0387  2
;  261       0388  2      IF  .tsf_int_hl EQL 0 !  Text to do (processing needed in LOUT1)?
;  262       0389  3          AND (.tsf_int_vl EQL 0)             !  Vertical positioning?                    ;
;  263       0390  3          AND (.tsf_first_xtn EQL 0)          !  Indexing to do?                          ;
;  264       0391  3          AND (.tsf_footw EQL 0)              !  Footnotes attached to this line?         ;
;  265       0392  2      THEN                                                                                 ;
;  266       0393  2          RETURN;                             ! THEN do absolutely nothing.            ;;;;;
;  267       0394  2  %FI                                                                                     ;
;  268       0395  2                                                                                          ;
;  269       0396  2      tsf_phregs = tsf__phregs;                                                           ;
;  270       0397  2      ptr_copy = .fs_start (mra);                                                         ;
;  271       0398  2                                                                                          ;
;  272       0399  2      INCR k FROM 1 TO .tsf_int_vl DO         ! Process vertical movement                 ;
;  273       0400  3          BEGIN                                                                           ;
;  274       0401  3                                                                                          ;
;  275       0402  3          LOCAL                                                                           ;
;  276       0403  3              hold_khar;                                                                  ;
;  277       0404  3                                                                                          ;
;  278       0405  3          hold_khar = CH$RCHAR_A (ptr_copy);                                              ;
;  279       0406  3                                                                                          ;
;  280       0407  3          ! All vertical motion code starts with RINTES.  Otherwise it shouldn't          ;
;  281       0408  3          ! be there and is an error (see the ELSE branch, below).                        ;
;  282       0409  3          IF .hold_khar EQL RINTES                                                        ;
;  283       0410  3          THEN                                                                            ;
;  284       0411  4              BEGIN                                                                       ;
;  285       0412  4                                                                                          ;
;  286       0413  4              LOCAL                                                                       ;
;  287       0414  4                  op_code,                                                                ;
;  288       0415  4                  operand;                                                                ;
;  289       0416  4                                                                                          ;
;  290       0417  4              ! The character after RINTES indicates what type of vertical motion         ;
;  291       0418  4              ! is to be done, and gets saved as OP_CODE.  The character after            ;
;  292       0419  4              ! that is either a dummy or is a parameter; it gets saved as                ;
;  293       0420  4              ! OPERAND.                                                                  ;
;  294       0421  4              op_code = CH$RCHAR_A (ptr_copy);                                            ;
;  295       0422  4              operand = CH$RCHAR_A (ptr_copy);                                            ;
;  296       0423  4              k = .k + 2;                                                                 ;
;  297       0424  4                                                                                          ;
;  298       0425  4              ! Process the particular type of vertical motion                            ;
;  299       0426  4              SELECTONE .op_code OF                                                       ;
;  300       0427  4                  SET                                                                     ;
```

```
;   301         0428  4
;   302         0429  4                 [%C'p'] :
;   303         0430  4                     phan_top_page = true;       . Start a new page
;   304         0431  4
;   305         0432  4                 [%C's'] :
;   306         0433  4                     cskipl (.tsf_phregs [.operand]);    ! skip lines if not top
;   307         0434  4                                                         ! of page
;   308         0435  4                 [%C'u'] :
;   309         0436  5                     BEGIN    ! skip lines regardless of page position
;   310         0437  5
;   311         0438  6                     IF (.phan_top_page
;   312         0439  6                         AND NOT .fnct_expanding)
;   313         0440  5                     THEN
;   314         0441  5                         newpag ();
;   315         0442  5
;   316         0443  5                     uskipl (.tsf_phregs [.operand]);
;   317         0444  4                     END;
;   318         0445  4
;   319         0446  4                 [%C'd'] :
;   320         0447  4                     ! Defer blank lines until top of page
;   321         0448  4                     phan_figure = .phan_figure + .tsf_phregs [.operand];
;   322         0449  4
;   323         0450  4                 [%C'g'] :
;   324         0451  4                     ! Go to a specific line immediately.
;   325         0452  4                     ! Start a new page if necessary.
;   326         0453  4                     ! NOTE: You can only go to a line within the text area of
;   327         0454  4                     !       the page.  So a negative line number is interpreted
;   328         0455  4                     !       as that many lines above the footer area (but not
;   329         0456  4                     !       above footnotes).
;   330         0457  5                     BEGIN
;   331         0458  5
;   332         0459  5                     LOCAL
;   333         0460  5                         x,      ! TRUE, if not already past that position.
;   334         0461  5                         y;      ! Skip this many lines to position.
;   335         0462  5
;   336         0463  6                     IF (.tsf_phregs [.operand] LSS 0)
;   337         0464  5                     THEN        ! Count from bottom.
;   338         0465  5                         x = tpr (ABS (.tsf_phregs [.operand]) )
;   339         0466  5                     ELSE        ! Absolute line number.
;   340         0467  5                         x = (.tsf_phregs [.operand]) GEQ .phan_lines_tp;
;   341         0468  5
;   342         0469  5                     IF NOT .x
;   343         0470  5                     THEN
;   344         0471  5                         newpag ();      ! Already too far.  Start a new page.
;   345         0472  5
;   346         0473  6                     IF (.tsf_phregs [.operand] LSS 0)
;   347         0474  5                     THEN
;   348         0475  5                         y = .phan_llines - .hct_layoutn - .phan_lines_tp + .tsf_phregs [.operand]
;   349         0476  5                     ELSE
;   350         0477  5                         y = .tsf_phregs [.operand] - .phan_lines_tp;
;   351         0478  5
;   352         0479  5                     ! If already at that position do nothing.  However, if not
;   353         0480  5                     ! at that position, get there.  Note the following special
;   354         0481  5                     ! case:  if we're at the top of a page then we have to get
;   355         0482  5                     ! past the top of the page for the counting to work
;   356         0483  5                     ! correctly.
;   357         0484  5                     IF .y NEQ 0
```

```
;   358     0485  5                     THEN
;   359     0486  6                         BEGIN
;   360     0487  6                         uskipl (1);     ! Force a blank line to get past the
;   361     0488  6                                         ! top of the page, if that's the case.
;   362     0489  6
;   363     0490  6                         ! Now recompute position all over again, since
;   364     0491  6                         ! we may have just gotten past the top of a page.
;   365     0492  7                         IF  (.tsf_phregs [.operand] LSS 0)
;   366     0493  6                         THEN
;   367     0494  6                             y = .phan_llines - .phan_lines_tp - .hct_layoutn + .tsf_phregs [.operand]
;   368     0495  6                         ELSE
;   369     0496  6                             y = .tsf_phregs [.operand] - .phan_lines_tp;
;   370     0497  6
;   371     0498  6                         ! Now do the remainder of the positioning to the
;   372     0499  6                         ! proper spot.
;   373     0500  6                         uskipl (.y)
;   374     0501  6                         END
;   375     0502  4                     END;
;   376     0503  4
;   377     0504  4                 [%C'.'] :
;   378     0505  5                     BEGIN
;   379     0506  5
;   380     0507  5                     IF .branch
;   381     0508  5                     THEN
;   382     0509  5                     ! Skip until next '.' is found.
;   383     0510  5                         find_next_dot (k, ptr_copy, hold_khar);
;   384     0511  5
;   385     0512  5                     branch = not .branch
;   386     0513  4                     END;
;   387     0514  4
;   388     0515  4                 [%C't'] :
;   389     0516  4                     ! Test page
;   390     0517  5                     BEGIN
;   391     0518  5
;   392     0519  5                     IF tpr (.tsf_phregs [.operand])
;   393     0520  5                     THEN
;   394     0521  5                     ! Take the 'THEN' branch
;   395     0522  5                         branch = true
;   396     0523  5                     ELSE
;   397     0524  5                     ! Skip the 'THEN' branch.
;   398     0525  6                         BEGIN
;   399     0526  6                         find_next_dot (k, ptr_copy, hold_khar);
;   400     0527  6                         branch = false;
;   401     0528  6                         END
;   402     0529  6
;   403     0530  4                     END;
;   404     0531  4
;   405     0532  4                 [%C'w'] :
;   406     0533  4                     IF (.pagen[sct_run_page] MOD 2) EQL 0
;   407     0534  4                     THEN
;   408     0535  4                         phan_top_page = true            !Simply start a new page
;   409     0536  4                     ELSE
;   410     0537  5                         BEGIN
;   411     0538  5                         IF .phan_top_first EQL 0
;   412     0539  5                         THEN
;   413     0540  6                             BEGIN
;   414     0541  6                             LOCAL
```

```
;   415          0542  6                                  hold_headers;
;   416          0543  6
;   417          0544  6                              hold_headers = .hct_headers;        !Save the status of headers
;   418          0545  6                              hct_headers = false;
;   419          0546  6                              newpag ();                          !Throw a 'blank' page
;   420          0547  6                              uskipl (1);                         ! force a blank line to get past the
;   421          0548  6                                                                  ! top of the page, if that's the case.
;   422          0549  6                              npagen [sct_page] = 1;              !number the next page "i"
;   423          0550  6                              hct_headers = .hold_headers;
;   424          0551  6                              phan_top_page = true;               !And start another new page
;   425          0552  6                              END
;   426          0553  5                          ELSE
;   427          0554  5                              phan_top_page = true;               !And start another new page
;   428          0555  4                          END;
;   429          0556  4
;   430          0557  4                      [OTHERWISE] :
;   431          0558  4                          ! Unrecognized sequence!!
;   432          0559  4                          ! Issue error message and carry on.
;   433          0560  5                          BEGIN
;   434          0561  5                          erms (rnfile, CH$PTR (UPLIT ('lout')), 4);
;   435          0562  4                          END;
;   436          0563  4                      TES;
;   437          0564  4
;   438          0565  4                  END
;   439          0566  3              ELSE
;   440          0567  3              ! Not an escape sequence.
;   441          0568  3              ! Issue an error message and ignore the character.
;   442          0569  4                  BEGIN
;   443          0570  4                  erms (rnfile, CH$PTR (UPLIT ('lout')), 4);
;   444          0571  3                  END;
;   445          0572  3
;   446          0573  2              END;
;   447          0574  2
;   448          0575  2          ! Paper-motion codes have been taken care of.
;   449          0576  2
;   450          0577  2          ! The index entry is associated with the current page number.  However,
;   451          0578  2          ! what the current page number is depends on whether or not there is some
;   452          0579  2          ! text on the page.  According to "strict" usage of indexing, indexing
;   453          0580  2          ! commands should go immediately after the line to which they apply.
;   454          0581  2          ! However, it can be expected that users will do the natural thing,
;   455          0582  2          ! i.e., put the indexing commands before the text to which they apply.
;   456          0583  2          ! This leads to the situation where the (other) indexing routines have no
;   457          0584  2          ! text with which to associate the index entry; for example, this is
;   458          0585  2          ! obviously true at the top of the very first page, or the top of any page
;   459          0586  2          ! for that matter.
;   460          0587  2          IF  .tsf_int_hl NEQ 0
;   461          0588  2          THEN
;   462          0589  2              ! There is some text to be output.
;   463          0590  3              BEGIN
;   464        U 0591  3 %IF DSRPLUS %THEN
;   465        U 0592  3              IF (.tsf_int_hl EQL .sca_margin_pad
;   466        U 0593  3                                  AND
;   467        U 0594  3                  .tsf_ext_hl EQL .sca_margin_pad)
;   468        U 0595  3              THEN
;   469        U 0596  3                  RETURN;
;   470          0597  3 %FI
;   471          0598  3
```

```
;  472          0599  3          lout1 (.ptr_copy);                  ! Output text.
;  473          0600  3
;  474          0601  3          IF .tsf_last_xtn NEQ  0
;  475          0602  3          THEN
;  476          0603  3              save_xtn (pagen, .tsf_last_xtn);    ! Associate current page number
;  477          0604  3          END                                     ! with transaction number.
;  478          0605  2      ELSE
;  479          0606  2          !
;  480          0607  2          ! Promote index entries attached to this line, if any.
;  481          0608  2          IF .tsf_last_xtn NEQ 0
;  482          0609  2          THEN
;  483          0610  3              BEGIN
;  484          0611  4              IF (.phan_top_page
;  485          0612  4                  AND NOT .phan_top_first)
;  486          0613  3              THEN
;  487          0614  4                  save_xtn (npagen, .tsf_last_xtn)
;  488          0615  3              ELSE
;  489          0616  3                  save_xtn (pagen, .tsf_last_xtn);
;  490          0617  2              END;
;  491          0618  2
;  492          0619  2      ! Clear transaction numbers in the TSF.  This is done so that for things
;  493          0620  2      ! such as titles, that reoccur, no attempt is made to "redefine" the page
;  494          0621  2      ! number associated with the transaction number.
;  495          0622  2      tsf_first_xtn = 0;
;  496          0623  2      tsf_last_xtn = 0;
;  497          0624  2
;  498          0625  2      ! If there were any footnotes associated with this line, associate them
;  499          0626  2      ! with this page.
;  500          0627  2      ascftn ();
;  501          0628  1      END;                                        ! End of LOUT


                                                    .TITLE  LOVERT Line output (vertical motion)
                                                    .IDENT  \V04-000\

                                                    .PSECT  $PLIT$,NOWRT,NOEXE,2

                    74  75  6F  6C  00000 P.AAA:    .ASCII  \lout\                                          ;
                    74  75  6F  6C  00004 P.AAB:    .ASCII  \lout\                                          ;

                                                    .EXTRN  RINTES, FNCT, GCA
                                                    .EXTRN  HCT, MRA, TSF, NPAGEN
                                                    .EXTRN  PAGEN, PHAN, RNFILE
                                                    .EXTRN  CSKIPL, ERMS, LOUT1
                                                    .EXTRN  NEWPAG, PUTXTN, TPR
                                                    .EXTRN  USKIPL

                                                    .PSECT  $CODE$,NOWRT,2

                        0FFC 00000                  .ENTRY  LOUT, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11      ; 0315
5B 00000000G    EF  9E 00002                        MOVAB   NEWPAG, R11                                     ;
5A 00000000G    EF  9E 00009                        MOVAB   TSF, R10                                        ;
59 00000000G    EF  9E 00010                        MOVAB   HCT+8, R9                                       ;
58 00000000G    EF  9E 00017                        MOVAB   PHAN+12, R8                                     ;
5E              0C  C2 0001E                        SUBL2   #12, SP                                         ;
                56  D4 00021                        CLRL    BRANCH                                          ; 0344
50              6A  D0 00023                        MOVL    TSF, R0                                         ; 0388
```

```
                                    60  D5 00026          TSTL    (R0)                                                    ;
                                    10  12 00028          BNEQ    1$                                                      ;
                                18  A0  D5 0002A          TSTL    24(R0)                                                  ; 0389
                                    0B  12 0002D          BNEQ    1$                                                      ;
                                38  A0  D5 0002F          TSTL    56(R0)                                                  ; 0390
                                    06  12 00032          BNEQ    1$                                                      ;
                                0C  A0  D5 00034          TSTL    12(R0)                                                  ; 0391
                                    01  12 00037          BNEQ    1$                                                      ;
                                        04 00039          RET                                                             ;
                        52  008C    C0  9E 0003A 1$:      MOVAB   140(R0), TSF_PHREGS                                     ; 0396
                    04  AE  00000000G FF  D0 0003F         MOVL    @MRA, PTR_COPY                                          ; 0397
                        57  18  A0  D0 00047              MOVL    24(R0), R7                                              ; 0399
                            08  AE  D4 0004B              CLRL    K                                                       ;
                                70  11 0004E              BRB     8$                                                      ;
                        6E  04  BE  9A 00050 2$:          MOVZBL  @PTR_COPY, HOLD_KHAR                                    ; 0405
                            04  AE  D6 00054              INCL    PTR_COPY                                                ;
                00000000G   8F      6E  D1 00057          CMPL    HOLD_KHAR, #RINTES                                      ; 0409
                                    03  13 0005E          BEQL    3$                                                      ;
                                0170    31 00060          BRW     29$                                                     ;
                        54  04  BE  9A 00063 3$:          MOVZBL  @PTR_COPY, OP_CODE                                      ; 0421
                            04  AE  D6 00067              INCL    PTR_COPY                                                ;
                        53  04  BE  9A 0006A              MOVZBL  @PTR_COPY, OPERAND                                      ; 0422
                            04  AE  D6 0006E              INCL    PTR_COPY                                                ;
                    08  AE      02  C0 00071              ADDL2   #2, K                                                   ; 0423
                00000070    8F      54  D1 00075          CMPL    OP_CODE, #112                                           ; 0429
                                    03  12 0007C          BNEQ    4$                                                      ;
                                0142    31 0007E          BRW     26$                                                     ;
                00000073    8F      54  D1 00081 4$:      CMPL    OP_CODE, #115                                           ; 0432
                                    0C  12 00088          BNEQ    5$                                                      ;
                                6243    DD 0008A          PUSHL   (TSF_PHREGS)[OPERAND]                                   ; 0433
                00000000G   EF      01  FB 0008D          CALLS   #1, TSKIPL                                              ;
                                    2A  11 00094          BRB     8$                                                      ;
                00000075    8F      54  D1 00096 5$:      CMPL    OP_CODE, #117                                           ; 0435
                                    13  12 0009D          BNEQ    7$                                                      ;
                        0A      F4  A8  E9 0009F          BLBC    PHAN, 6$                                                ; 0438
                        03  00000000G   EF  E8 000A3      BLBS    FNCT+24, 6$                                             ; 0439
                        6B          00  FB 000AA          CALLS   #0, NEWPAG                                              ; 0441
                                6243    DD 000AD 6$:      PUSHL   (TSF_PHREGS)[OPERAND]                                   ; 0443
                                    7D  11 000B0          BRB     18$                                                     ;
                00000064    8F      54  D1 000B2 7$:      CMPL    OP_CODE, #100                                           ; 0446
                                    07  12 000B9          BNEQ    9$                                                      ;
                    04  A8      6243    C0 000BB          ADDL2   (TSF_PHREGS)[OPERAND], PHAN+16                          ; 0448
                                    74  11 000C0 8$:      BRB     19$                                                     ;
                00000067    8F      54  D1 000C2 9$:      CMPL    OP_CODE, #103                                           ; 0450
                                    6D  12 000C9          BNEQ    20$                                                     ;
                        54          6243    D0 000CB      MOVL    (TSF_PHREGS)[OPERAND], R4                               ; 0463
                                    55  D4 000CF          CLRL    R5                                                      ;
                                    54  D5 000D1          TSTL    R4                                                      ;
                                    12  18 000D3          BGEQ    11$                                                     ;
                                    55  D6 000D5          INCL    R5                                                      ;
                                    54  DD 000D7          PUSHL   R4                                                      ; 0465
                                    03  18 000D9          BGEQ    10$                                                     ;
                        6E          6E  CE 000DB          MNEGL   (SP), (SP)                                              ;
                00000000G   EF      01  FB 000DE 10$:     CALLS   #1, TPR                                                 ;
                                    09  11 000E5          BRB     12$                                                     ;
                                    50  D4 000E7 11$:     CLRL    R0                                                      ; 0467
                        68          54  D1 000E9          CMPL    R4, PHAN+12                                             ;
```

```
                                                02  19 000EC          BLSS    12$                                     ;
                                                50  D6 000EE          INCL    R0                                      ;
                                 03             50  E8 000F0 12$:     BLBS    X, 13$                                  ; 0469
                                 6B             00  FB 000F3         CALLS   #0, NEWPAG                              ; 0471
                                 0F             55  E9 000F6 13$:     BLBC    R5, 14$                                 ; 0475
                50       F8      B8     18      A9  C3 000F9         SUBL3   HCT+32, @PHAN+4, R0                     ;
                                 50             68  C2 000FF         SUBL2   PHAN+12, R0                             ;
                53               50             54  C1 00102         ADDL3   R4, R0, Y                               ;
                                                04  11 00106         BRB     15$                                     ;
                53               54             68  C3 00108 14$:     SUBL3   PHAN+12, R4, Y                          ; 0477
                                                72  13 0010C 15$:     BEQL    24$                                     ; 0484
                                                01  DD 0010E         PUSHL   #1                                      ; 0487
            00000000G   EF                      01  FB 00110         CALLS   #1, USKIPL                              ;
                                 0F             55  E9 00117         BLBC    R5, 16$                                 ; 0494
                50       F8      B8             68  C3 0011A         SUBL3   PHAN+12, @PHAN+4, R0                    ;
                                 50     18      A9  C2 0011F         SUBL2   HCT+32, R0                              ;
                53               50             54  C1 00123         ADDL3   R4, R0, Y                               ;
                                                04  11 00127         BRB     17$                                     ;
                53               54             68  C3 00129 16$:     SUBL3   PHAN+12, R4, Y                          ; 0496
                                                53  DD 0012D 17$:     PUSHL   Y                                       ; 0500
            00000000G   EF                      01  FB 0012F 18$:     CALLS   #1, USKIPL                              ;
                                                48  11 00136 19$:     BRB     24$                                     ; 0484
                                 2E             54  D1 00138 20$:     CMPL    OP_CODE, #46                            ; 0504
                                                17  12 0013B         BNEQ    22$                                     ;
                                 0F             56  E9 0013D         BLBC    BRANCH, 21$                             ; 0507
                                                5E  DD 00140         PUSHL   SP                                      ; 0510
                                         08     AE  9F 00142         PUSHAB  PTR_COPY                                ;
                                         10     AE  9F 00145         PUSHAB  K                                       ;
            00000000V   EF                      03  FB 00148         CALLS   #3, FIND_NEXT_DOT                       ;
                                 56             56  D2 0014F 21$:     MCOML   BRANCH, BRANCH                          ; 0512
                                                73  11 00152         BRB     27$                                     ;
            00000074    8F                      54  D1 00154 22$:     CMPL    OP_CODE, #116                           ; 0515
                                                25  12 0015B         BNEQ    25$                                     ;
                                           6243  DD 0015D         PUSHL   (TSF_PHREGS)[OPERAND]                   ; 0519
            00000000G   EF                      01  FB 00160         CALLS   #1, TPR                                 ;
                                 05             50  E9 00167         BLBC    R0, 23$                                 ;
                                 56             01  D0 0016A         MOVL    #1, BRANCH                              ; 0522
                                                79  11 0016D         BRB     31$                                     ;
                                                5E  DD 0016F 23$:     PUSHL   SP                                      ; 0526
                                         08     AE  9F 00171         PUSHAB  PTR_COPY                                ;
                                         10     AE  9F 00174         PUSHAB  K                                       ;
            00000000V   EF                      03  FB 00177         CALLS   #3, FIND_NEXT_DOT                       ;
                                                56  D4 0017E         CLRL    BRANCH                                  ; 0527
                                                66  11 00180 24$:     BRB     31$                                     ; 0517
            00000077    8F                      54  D1 00182 25$:     CMPL    OP_CODE, #119                           ; 0532
                                                3E  12 00189         BNEQ    28$                                     ;
                                 50 00000000G   EF  3C 0018B         MOVZWL  PAGEN+14, R0                            ; 0533
7E              00               50             01  7A 00192         EMUL    #1, R0, #0, -(SP)                       ;
50              50               8E             02  7B 00197         EDIV    #2, (SP)+, R0, R0                       ;
                                                50  D5 0019C         TSTL    R0                                      ;
                                                23  13 0019E         BEQL    26$                                     ;
                                         0C     A8  D5 001A0         TSTL    PHAN+24                                 ; 0538
                                                1E  12 001A3         BNEQ    26$                                     ;
                                 53     00      B9  D0 001A5         MOVL    @HCT+8, HOLD_HEADERS                    ; 0544
                                         00     B9  D4 001A9         CLRL    @HCT+8                                  ; 0545
                                 6B             00  FB 001AC         CALLS   #0, NEWPAG                              ; 0546
                                                01  DD 001AF         PUSHL   #1                                      ; 0547
```

```
                    00000000G   EF                  01   FB 001B1          CALLS   #1, USKIPL
                    00000000G   EF                  01   D0 001B8          MOVL    #1, NPAGEN+8                                    ;  0549
                            00  B9                  53   D0 001BF          MOVL    HOLD_HEADERS, @HCT+8                            ;  0550
                            F4  A8                  01   D0 001C3 26$:     MOVL    #1, PHAN                                        ;  0554
                                                    1F   11 001C7 27$:     BRB     31$                                             ;  0533
                                                    04   DD 001C9 28$:     PUSHL   #4                                              ;  0561
                                        00000000'   EF   9F 001CB          PUSHAB  P.AAA
                                                    08   11 001D1          BRB     30$
                                                    04   DD 001D3 29$:     PUSHL   #4                                              ;  0570
                                        00000000'   EF   9F 001D5          PUSHAB  P.AAB
                                        00000000G   8F   DD 001DB 30$:     PUSHL   #RNFILE
                    00000000G   EF                  03   FB 001E1          CALLS   #3, ERMS
            FE61        08  AE          01          57   F1 001E8 31$:     ACBL    R7, #1, K, 2$                                   ;  0399
                                        50          6A   D0 001EF          MOVL    TSF, R0                                         ;  0587
                                                    60   D5 001F2          TSTL    (R0)
                                                    1F   13 001F4          BEQL    32$
                                            04      AE   DD 001F6          PUSHL   PTR_COPY                                        ;  0599
                    00000000G   EF                  01   FB 001F9          CALLS   #1, LOUT1
                                        50          6A   D0 00200          MOVL    TSF, R0                                         ;  0601
                                            3C      A0   D5 00203          TSTL    60(R0)
                                                    44   13 00206          BEQL    36$
                        3C 00000000G   EF           02   E1 00208          BBC     #2, GCA+124, 36$                                ;  0603
                                            3C      A0   DD 00210          PUSHL   60(R0)
                                                    2A   11 00213          BRB     34$
                                        50          3C   A0   D0 00215 32$:  MOVL    60(R0), R0                                      ;  0608
                                                    31   13 00219          BEQL    36$
                                        16          F4   A8   E9 0021B     BLBC    PHAN, 33$                                       ;  0611
                                        12          0C   A8   E8 0021F     BLBS    PHAN+24, 33$                                    ;  0612
                        21 00000000G   EF           02   E1 00223          BBC     #2, GCA+124, 36$                                ;  0614
                                                    50   DD 0022B          PUSHL   R0
                                        00000000G   EF   9F 0022D          PUSHAB  NPAGEN
                                                    10   11 00233          BRB     35$
                    0F 00000000G   EF               02   E1 00235 33$:     BBC     #2, GCA+124, 36$                                ;  0616
                                                    50   DD 0023D          PUSHL   R0
                                        00000000G   EF   9F 0023F 34$:     PUSHAB  PAGEN
                    00000000G   EF                  02   FB 00245 35$:     CALLS   #2, PUTXTN
                                        50          6A   D0 0024C 36$:     MOVL    TSF, R0                                         ;  0617
                                                38      A0   7C 0024F          CLRQ    56(R0)                                          ;  0622
                    00000000V   EF                  00   FB 00252          CALLS   #0, ASCFTN                                      ;  0627
                                                        04 00259          RET                                                     ;  0628

; Routine Size:  602 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  503          0629  1 %SBTTL 'JUSTF -- Compute spaces needed for justification'
;  504          0630  1 GLOBAL ROUTINE JUSTF (PADDING, INSERT_COUNT, SPACE_COUNT, ALGORITHM) : NOVALUE =
;  505          0631  1
;  506          0632  1 !++
;  507          0633  1 ! FUNCTIONAL DESCRIPTION:
;  508          0634  1 !
;  509          0635  1 !       Computes the number of spaces to be dropped between words
;  510          0636  1 !       and their locations.
;  511          0637  1 !
;  512          0638  1 ! FORMAL PARAMETERS:
;  513          0639  1 !
;  514          0640  1 !       INSERT_COUNT is the number of places where spaces can
;  515          0641  1 !       be inserted.  SPACE_COUNT is the number of spaces to
;  516          0642  1 !       be distributed.
;  517          0643  1 !       ALGORITHM specifies how spaces are to be distributed.
;  518          0644  1 !       The space distribution is returned in PADDING as a set
;  519          0645  1 !       of insert counts.
;  520          0646  1 !
;  521          0647  1 ! IMPLICIT INPUTS:      None
;  522          0648  1 !
;  523          0649  1 ! IMPLICIT OUTPUTS:     None
;  524          0650  1 !
;  525          0651  1 ! ROUTINE VALUE:
;  526          0652  1 ! COMPLETION CODES:     None
;  527          0653  1 !
;  528          0654  1 ! SIDE EFFECTS:         None
;  529          0655  1 !
;  530          0656  1 !--
;  531          0657  1
;  532          0658  2     BEGIN
;  533          0659  2
;  534          0660  2     MAP
;  535          0661  2         PADDING : REF VECTOR;
;  536          0662  2
;  537          0663  2     LOCAL
;  538          0664  2         SPACES_PER_I,           ! This many spaces inserted per word.
;  539          0665  2         REMAINING,              ! This many "odd" spaces remain.
;  540          0666  2         RIGHT_INSERT,           ! Put this many spaces at the right.
;  541          0667  2         LEFT_INSERT,            ! Put this many spaces between words on left.
;  542          0668  2         RIGHT_COUNT,            ! There are this many words on the right.
;  543          0669  2         LEFT_COUNT,             ! There are this many words on the left.
;  544          0670  2         PI;                     ! Index into padding.
;  545          0671  2
;  546          0672  2     SPACES_PER_I = .SPACE_COUNT/.INSERT_COUNT;
;  547          0673  2     REMAINING = .SPACE_COUNT MOD .INSERT_COUNT;
;  548          0674  2     LEFT_INSERT = .SPACES_PER_I;
;  549          0675  2     RIGHT_INSERT = .SPACES_PER_I + (.REMAINING NEQ 0);
;  550          0676  2     RIGHT_COUNT = .REMAINING;
;  551          0677  2     LEFT_COUNT = .INSERT_COUNT - .RIGHT_COUNT;
;  552          0678  2
;  553          0679  2     IF .ALGORITHM
;  554          0680  2     THEN
;  555          0681  3         BEGIN                                   ! Use alternate space distribution algorithm.
;  556          0682  3         EXCHANGE (LEFT_INSERT, RIGHT_INSERT);
;  557          0683  3         EXCHANGE (LEFT_COUNT, RIGHT_COUNT);
;  558          0684  2         END;
;  559          0685  2
```

```
;  560          0686  2     PI = 0;
;  561          0687  2
;  562          0688  2     INCR I FROM 1 TO .RIGHT_COUNT DO
;  563          0689  3         BEGIN
;  564          0690  3         PADDING [.PI] = .RIGHT_INSERT;
;  565          0691  3         PI = .PI + 1;
;  566          0692  2         END;
;  567          0693  2
;  568          0694  2     INCR I FROM 1 TO .LEFT_COUNT DO
;  569          0695  3         BEGIN
;  570          0696  3         PADDING [.PI] = .LEFT_INSERT;
;  571          0697  3         PI = .PI + 1;
;  572          0698  2         END;
;  573          0699  2
;  574          0700  1     END;                                    ! End of JUSTF



                                    003C 00000          .ENTRY  JUSTF, Save R2,R3,R4,R5                         ; 0630
                52      0C  AC      08  AC  C7 00002        DIVL3   INSERT_COUNT, SPACE_COUNT, SPACES_PER_I      ; 0672
          7E    00      0C  AC          01  7A 00008        EMUL    #1, SPACE_COUNT, #0, -(SP)                  ; 0673
          51    51          8E      08  AC  7B 0000E        EDIV    INSERT_COUNT, (SP)+, REMAINING, REMAINING   ;
                            54          52  D0 00014        MOVL    SPACES_PER_I, LEFT_INSERT                   ; 0674
                                        50  D4 00017        CLRL    R0                                          ; 0675
                                        51  D5 00019        TSTL    REMAINING                                   ;
                                        02  13 0001B        BEQL    1$                                          ;
                                        50  D6 0001D        INCL    R0                                          ;
                55              50      52  C1 0001F 1$:    ADDL3   SPACES_PER_I, R0, RIGHT_INSERT              ;
                                52      51  D0 00023        MOVL    REMAINING, RIGHT_COUNT                      ; 0676
                53      08  AC          52  C3 00026        SUBL3   RIGHT_COUNT, INSERT_COUNT, LEFT_COUNT       ; 0677
                                12  10  AC  E9 0002B        BLBC    ALGORITHM, 2$                               ; 0679
                                50      54  D0 0002F        MOVL    LEFT_INSERT, H                              ; 0682
                                54      55  D0 00032        MOVL    RIGHT_INSERT, LEFT_INSERT                   ;
                                55      50  D0 00035        MOVL    H, RIGHT_INSERT                             ;
                                50      53  D0 00038        MOVL    LEFT_COUNT, H                               ; 0683
                                53      52  D0 0003B        MOVL    RIGHT_COUNT, LEFT_COUNT                     ;
                                52      50  D0 0003E        MOVL    H, RIGHT_COUNT                              ;
                                        50  7C 00041 2$:     CLRQ    PI                                          ; 0686
                                        07  11 00043        BRB     4$                                          ; 0690
                        04 BC40         55  D0 00045 3$:    MOVL    RIGHT_INSERT, @PADDING[PI]                  ;
                                        50  D6 0004A        INCL    PI                                          ; 0691
                F5              51      52  F3 0004C 4$:    AOBLEQ  RIGHT_COUNT, I, 3$                          ; 0688
                                        52  D4 00050        CLRL    I                                           ; 0696
                                        07  11 00052        BRB     6$                                          ;
                        04 BC40         54  D0 00054 5$:    MOVL    LEFT_INSERT, @PADDING[PI]                   ;
                                        50  D6 00059        INCL    PI                                          ; 0697
                F5              52      53  F3 0005B 6$:    AOBLEQ  LEFT_COUNT, I, 5$                           ; 0694
                                            04 0005F        RET                                                 ; 0700

; Routine Size:  96 bytes,    Routine Base:  $CODE$ + 025A
```

```
;  576     0701  1 %SBTTL 'FIND_NEXT_DOT -- Skip to next "." in text.'
;  577     0702  1 ROUTINE FIND_NEXT_DOT (N, PTR, KHAR) : NOVALUE =
;  578     0703  1
;  579     0704  1 !++
;  580     0705  1 ! FUNCTIONAL DESCRIPTION:
;  581     0706  1 !
;  582     0707  1 !       Skip text until a "." is seen
;  583     0708  1 !
;  584     0709  1 ! FORMAL PARAMETERS:
;  585     0710  1 !
;  586     0711  1 !       N       -  Address of current offset into the string.
;  587     0712  1 !       PTR     -  Address of a String descriptor pointing to a text string.
;  588     0713  1 !       KHAR    -  Character just beyond the period (returned value).
;  589     0714  1 !
;  590     0715  1 ! IMPLICIT INPUTS:      None
;  591     0716  1 !
;  592     0717  1 ! IMPLICIT OUTPUTS:
;  593     0718  1 !
;  594     0719  1 !       N and PTR are advanced to reflect updated position.
;  595     0720  1 !
;  596     0721  1 ! ROUTINE VALUE:
;  597     0722  1 ! COMPLETION CODES:     None
;  598     0723  1 !
;  599     0724  1 ! SIDE EFFECTS:         None
;  600     0725  1 !
;  601     0726  1 !--
;  602     0727  1
;  603     0728  2     BEGIN
;  604     0729  2
;  605     0730  2     MACRO
;  606     0731  2         K = .N %,
;  607     0732  2         PTR_COPY = .PTR %,
;  608     0733  2         HOLD_KHAR = .KHAR %;
;  609     0734  2
;  610     0735  2     WHILE 1 DO
;  611     0736  3     BEGIN
;  612     0737  3     HOLD_KHAR = CH$RCHAR_A (PTR_COPY);
;  613     0738  3     K = .K + 1;
;  614     0739  3
;  615     0740  3     IF .HOLD_KHAR EQL RINTES
;  616     0741  3     THEN
;  617     0742  4         BEGIN
;  618     0743  4         LOCAL
;  619     0744  4             TEMP;
;  620     0745  4         TEMP = CH$RCHAR_A (PTR_COPY);
;  621     0746  4         K = .K + 1;
;  622     0747  4         HOLD_KHAR = CH$RCHAR_A (PTR_COPY);
;  623     0748  4         K = .K + 1;
;  624     0749  4
;  625     0750  4         IF .TEMP EQL %C'.'
;  626     0751  4         THEN
;  627     0752  4             RETURN;
;  628     0753  4         END
;  629     0754  3     END
;  630     0755  1     END;                                    ! End of find_next_dot
```

```
                                        0004 00000 FIND_NEXT_DOT:
                                                   .WORD   Save R2                                          ; 0702
                    50      04      AC  D0 00002           MOVL    N, R0                                    ; 0737
                    51      08      BC  D0 00006 1$:       MOVL    @PTR, R1                                 ;
                OC  BC              61  9A 0000A           MOVZBL  (R1), @KHAR                              ;
                            08      BC  D6 0000E           INCL    @PTR                                     ;
                                    60  D6 00011           INCL    (R0)                                     ; 0738
        00000000G   8F      0C      BC  D1 00013           CMPL    @KHAR, #RINTES                           ; 0740
                                    E9  12 0001B           BNEQ    1$                                       ;
                    52      08      BC  D0 0001D           MOVL    @PTR, R2                                 ; 0745
                    51              62  9A 00021           MOVZBL  (R2), TEMP                               ;
                            08      BC  D6 00024           INCL    @PTR                                     ;
                                    60  D6 00027           INCL    (R0)                                     ; 0746
                    52      08      BC  D0 00029           MOVL    @PTR, R2                                 ; 0747
                OC  BC              62  9A 0002D           MOVZBL  (R2), @KHAR                              ;
                            08      BC  D6 00031           INCL    @PTR                                     ;
                                    60  D6 00034           INCL    (R0)                                     ; 0748
                    2E              51  D1 00036           CMPL    TEMP, #46                                ; 0750
                                    CB  12 00039           BNEQ    1$                                       ;
                                        04 0003B           RET                                              ; 0755

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 02BA
```

```
;  632          0756  1 %SBTTL 'ASCFTN -- Associate this line''s fn''s to this page'
;  633          0757  1 GLOBAL ROUTINE ASCFTN : NOVALUE =
;  634          0758  1 !++
;  635          0759  1 ! FUNCTIONAL DESCRIPTION:
;  636          0760  1 !
;  637          0761  1 !       Associates footnotes associated with the current line with
;  638          0762  1 !       the current page.  Such footnotes may be output at any point after
;  639          0763  1 !       this.
;  640          0764  1 !
;  641          0765  1 ! FORMAL PARAMETERS:    None
;  642          0766  1 !
;  643          0767  1 ! IMPLICIT INPUTS:     None
;  644          0768  1 !
;  645          0769  1 ! IMPLICIT OUTPUTS:    None
;  646          0770  1 !
;  647          0771  1 ! ROUTINE VALUE:
;  648          0772  1 ! COMPLETION CODES:    None
;  649          0773  1 !
;  650          0774  1 ! SIDE EFFECTS:        None
;  651          0775  1 !
;  652          0776  1 !--
;  653          0777  1
;  654          0778  2     BEGIN
;  655          0779  2
;  656          0780  2     FNCT_READY = .FNCT_READY + .TSF_FOOTW;      ! Number of footnotes that can be output.
;  657          0781  2     ! The following code is not really necessary.  All it does is establish a
;  658          0782  2     ! consistancy check to make sure footnotes are being counted correctly.
;  659          0783  2     FNCT_WAITING = .FNCT_WAITING - .TSF_FOOTW;  ! Reduce number of footnotes "in limbo."
;  660          0784  2     TSF_FOOTW = 0;                              ! Forget these footnotes now, so they don't get counted twic
;  661          0785  2
;  662          0786  2     ! Now perform the consistancy check.
;  663          0787  2     IF
;  664          0788  2         .FNCT_WAITING LSS 0
;  665          0789  2     THEN
;  666          0790  2         ! Something's wrong. Complain (INTERNAL LOGIC) and try to recover.
;  667          0791  3         BEGIN
;  668          0792  3         ERMS (RNFILE, CH$PTR (UPLIT ('ASCFTN')), 6);
;  669          0793  3         ! Attempt to reset this consistancy check so this message
;  670          0794  3         ! doesn't continue to happen.
;  671          0795  3         FNCT_WAITING = .FNCT_COUNT - .FNCT_READY
;  672          0796  2         END;
;  673          0797  2
;  674          0798  2     ! Clear out the counter, so these footnotes don't get counted twice.
;  675          0799  2     TSF_FOOTW = 0;
;  676          0800  2
;  677          0801  1     END;                                        ! End of ascftn


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                    00  00  4E  54  46  43  53  41  00008 P.AAC:  .ASCII  \ASCFTN\<0><0>                                ;


                                                        .PSECT  $CODE$,NOWRT,2
```

```
                                                000C 00000          .ENTRY  ASCFTN, Save R2,R3                              ; 0757
                                     53 00000000G   EF   9E 00002   MOVAB   TSF, R3                                        ;
                                     52 00000000G   EF   9E 00009   MOVAB   FNCT+16, R2                                    ;
                                     50             63   D0 00010   MOVL    TSF, R0                                       ; 0780
                                  F4 A2        0C   A0   C0 00013   ADDL2   12(R0), FNCT+4                                ;
                                     62        0C   A0   C2 00018   SUBL2   12(R0), FNCT+16                               ; 0783
                                               0C   A0   D4 0001C   CLRL    12(R0)                                        ; 0784
                                                    62   D5 0001F   TSTL    FNCT+16                                       ; 0788
                                                    1B   18 00021   BGEQ    1$                                            ;
                                                    06   DD 00023   PUSHL   #6                                            ; 0792
                                        00000000'   EF   9F 00025   PUSHAB  P.AAC                                         ;
                                        00000000G   8F   DD 0002B   PUSHL   #RNFILE                                       ;
                                  00000000G EF              03   FB 00031   CALLS   #3, ERMS                               ;
                               62       F0 A2        F4   A2   C3 00038   SUBL3   FNCT+4, FNCT, FNCT+16                    ; 0795
                                     50             63   D0 0003E 1$:    MOVL    TSF, R0                                       ; 0796
                                               0C   A0   D4 00041         CLRL    12(R0)                                        ; 0799
                                                    04   00044         RET                                                   ; 0801

; Routine Size:  69 bytes,    Routine Base:  $CODE$ + 02F6

;  678          0802  1
;  679          0803  1 END                                                    ! End of module
;  680          0804  0 ELUDOM




;                                   PSECT SUMMARY
;
;       Name                            Bytes                           Attributes

;  $PLIT$                                 16  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                                827  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)



;                                 Library Statistics
;                                                 -------- Symbols --------      Pages       Processing
;       File                                      Total    Loaded   Percent      Mapped      Time

;  _$255$DUA28:[SYSLIB]XPORT.L32;1                 590         0         0         252         00:00.1
;  _$255$DUA28:[RUNOFF.SRC]DSRLIB.L32;1           1248        37         2          86         00:00.3



;                                     COMMAND QUALIFIERS

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

```
; Size:          827 code + 16 data bytes
; Run Time:        00:16.0
; Elapsed Time:    00:37.2
; Lines/CPU Min:    3015
; Lexemes/CPU-Min: 15585
; Memory Used:  154 pages
; Compilation Complete
```