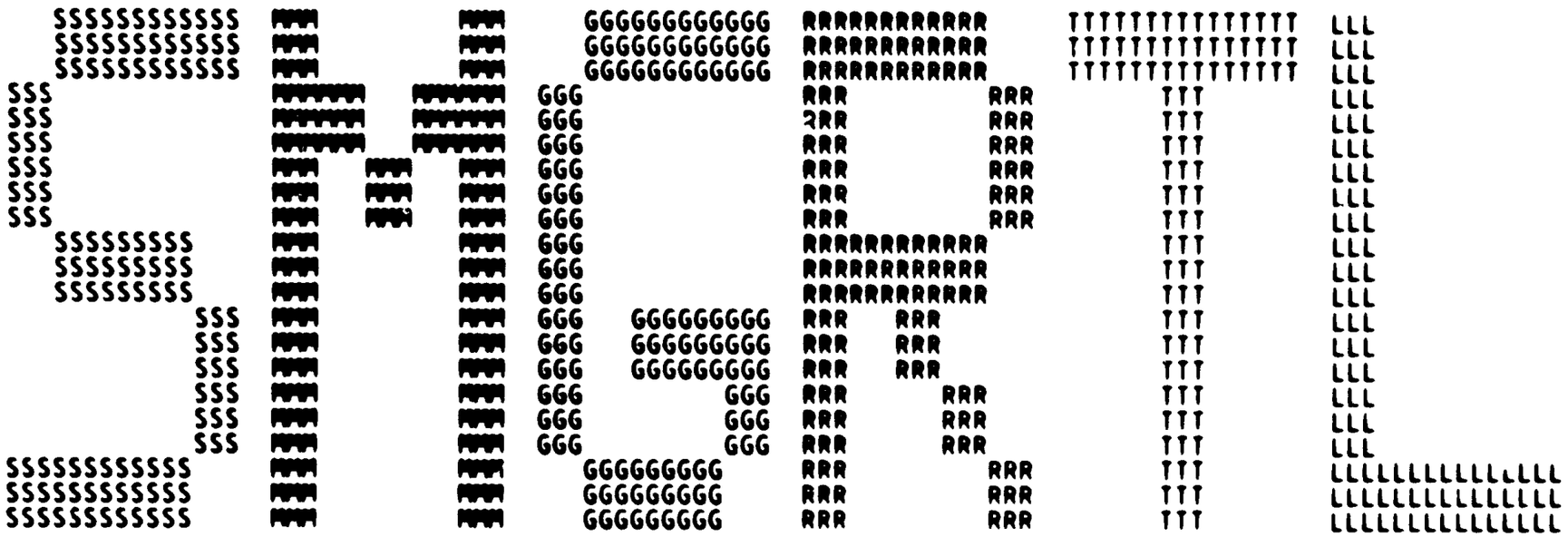
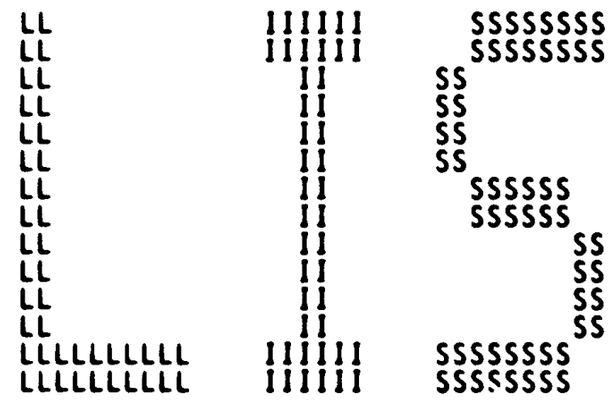

SSSSSSSSSSS MMM MMM GGGGGGGGGGG RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
SSSSSSSSSSS MMM MMM GGGGGGGGGGG RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
SSSSSSSSSSS MMM MMM GGGGGGGGGGG RRRRRRRRRRR TTTTTTTTTTTTTTT LLL
SSS MMMMMMM MMMMMMM GGG RRR RRR TTT LLL
SSS MMMMMMM MMMMMMM GGG RRR RRR TTT LLL
SSS MMMMMMM MMMMMMM GGG RRR RRR TTT LLL
SSS MMMM MMM MMMM GGG RRR RRR TTT LLL
SSS MMMM MMM MMMM GGG RRR RRR TTT LLL
SSS MMMM MMM MMMM GGG RRR RRR TTT LLL
SSSSSSSSS MMMM MMMM GGG RRRRRRRRRRRR TTT LLL
SSSSSSSSS MMMM MMMM GGG RRRRRRRRRRRR TTT LLL
SSSSSSSSS MMMM MMMM GGG RRRRRRRRRRRR TTT LLL
SSS MMMM MMMM GGG GGGGGGGGG RRR RRR TTT LLL
SSS MMMM MMMM GGG GGGGGGGGG RRR RRR TTT LLL
SSS MMMM MMMM GGG GGGGGGGGG RRR RRR TTT LLL
SSS MMMM MMMM GGG GGG RRR RRR TTT LLL
SSS MMMM MMMM GGG GGG RRR RRR TTT LLL
SSS MMMM MMMM GGG GGG RRR RRR TTT LLL
SSSSSSSSSSS MMMM MMMM GGGGGGGGG RRR RRR TTT LLLLLLLLLLLLLLL
SSSSSSSSSSS MMMM MMMM GGGGGGGGG RRR RRR TTT LLLLLLLLLLLLLLL
SSSSSSSSSSS MMMM MMMM GGGGGGGGG RRR RRR TTT LLLLLLLLLLLLLLL


SSSSSSSS MM MM GGGGGGGG MM MM IIIIII NN NN
SSSSSSSS MM MM GGGGGGGG MM MM IIIIII NN NN
SS MMMM MMMM GG MMMM MMMM II NN NN
SS MMMM MMMM GG MMMM MMMM II NN NN
SS MM MM MM GG MM MM MM II NNNN NN
SS MM MM MM GG MM MM MM II NNNN NN
SSSSSS MM MM GG MM MM II NN NN NN
SSSSSS MM MM GG MM MM II NN NN NN
SS MM MM GG GGGGGG MM MM II NN NNNN
SS MM MM GG GGGGGG MM MM II NN NNNN
SS MM MM GG GG MM MM II NN NN ....
SS MM MM GG GG MM MM II NN NN ....
SSSSSSSS MM MM GGGGGG MM MM IIIIII NN NN ....
SSSSSSSS MM MM GGGGGG MM MM IIIIII NN NN ....


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 'Minimal update calculation'
;     2          0002  0 MODULE SMG$MIN (
;001  :PLL1018   0003  0                 IDENT = '1-018' ! File: SMGMIN.B32      Edit:PLL1018
;     4-1        0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                      *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
```

```
;    31          0030  1 !++
;    32          0031  1 ! FACILITY:     Screen Management
;    33          0032  1 !
;    34          0033  1 ! ABSTRACT:
;    35          0034  1 !
;    36          0035  1 !       This module contains routines which inspect two screen
;    37          0036  1 !       representations and calculate the near-minimal sequence of
;    38          0037  1 !       terminal commands to change the current contents of the screen
;    39          0038  1 !       to the new representation of the screen.
;    40          0039  1 !
;    41          0040  1 ! ENVIRONMENT:  User mode, SMG package.
;    42          0041  1 !
;    43          0042  1 ! AUTHOR: Stanley Rabinowitz, CREATION DATE: 1-May-1983.
;    44          0043  1 !         FIND_MIN_CURSOR_POS is by PKR.
;    45          0044  1 !
;    46          0045  1 ! MODIFIED BY:
;    47          0046  1 !
;001  :PLL1018   0047  1 ! 1-018 - STAN, 21-Oct-1984. Change + to OR and optimize call to PUT_SCREEN.
;002  :PLL1018   0048  1 ! 1-017 - PLL,  12-Oct-1984. Fix to SMG$$OUTPUT_MINIMAL_UPDATE to minimize
;003  :PLL1018   0049  1 !                            the number of QIOs.
;004  :PLL1018   0050  1 !-----------+
;005  :PLL1018   0051  1 ! VMS V4.0  !
;006  :PLL1018   0052  1 !-----------+
;    48          0053  1 ! 1-016 - STAN  6-Jun-1984. Change error messages in MSG$SET_PHYSICAL_CURSOR.
;    49          0054  1 ! 1-001 - STAN, 1-May-1983. Initial version, mimicked SCRMIN.B32.
;    50          0055  1 !--
```

```
;    52          0056  1 %SBTTL 'Declarations'
;    53          0057  1 !
;    54          0058  1 ! SWITCHES:
;    55          0059  1 !
;    56          0060  1 !       NONE
;    57          0061  1 !
;    58          0062  1 ! LINKAGES:
;    59          0063  1 !
;    60          0064  1 !       NONE
;    61          0065  1 !
;    62          0066  1 ! TABLE OF CONTENTS:
;    63          0067  1 !
;    64          0068  1
;    65          0069  1 FORWARD ROUTINE
;    66          0070  1
;    67          0071  1         SMG$SET_PHYSICAL_CURSOR,        ! Move physical cursor on screen
;    68          0072  1         SMG$$OUTPUT_MINIMAL_UPDATE,     ! Output minumal update sequence
;    69          0073  1         SMG$$FIND_MIN_CURSOR_POS,       ! Output minimum cursor sequence
;    70          0074  1         SMG$$UPDATE_PHYSICAL_CURSOR,    ! Update physical cursor position
;    71          0075  1         ERASE_LINE,                     ! Erase to end-of-line
;    72          0076  1         SET_CURSOR;                     ! Generate general set-cursor
;    73          0077  1                                         ! positioning sequence.
;    74          0078  1
;    75          0079  1 !
;    76          0080  1 ! INCLUDE FILES
;    77          0081  1 !
;    78          0082  1
;    79          0083  1 REQUIRE 'RTLIN:SMGPROLOG';              ! defines psects, macros, structures,
;    80          0161  1                                         ! & terminal symbols
;    81          0162  1 REQUIRE 'RTLIN:STRLNK.REQ';             ! JSB linkages
;    82          0347  1
;    83          0348  1 !
;    84          0349  1 ! EXTERNAL REFERENCES
;    85          0350  1 !
;    86          0351  1
;    87          0352  1 EXTERNAL ROUTINE
;    88          0353  1
;001  :PLL1018   0354  1         SMG$$OUTPUT,
;002  :PLL1018   0355  1         SMG$$FLUSH_BUFFER;
;003  :PLL1018   0356  1
;004  :PLL1018   0357  1 !+
;005  :PLL1018   0358  1 ! $OUTPUT_STRING
;006  :PLL1018   0359  1 ! --------------
;007  :PLL1018   0360  1 !
;008  :PLL1018   0361  1 !       In order that we should go through a faster code path
;009  :PLL1018   0362  1 !       in SMG$$PUT_SCREEN, we omit the last argument if it is 0.
;    94-5        0363  1 !-
;    95          0364  1
;    96          0365  1 MACRO
;    97          0366  1
;    98        M 0367  1     $OUTPUT_STRING(LEN,ADDR,ATTR) =
;    99        M 0368  1
;   100        M 0369  1         BEGIN
;   101        M 0370  1         EXTERNAL ROUTINE SMG$$PUT_SCREEN;
;   102        M 0371  1         LOCAL STATUS;
;001  :PLL1018 M 0372  1         IF ATTR EQL 0
;002  :PLL1018 M 0373  1           THEN  STATUS=SMG$$PUT_SCREEN(PBCB,LEN,ADDR,0,0)
```

```
:003 :PLL1018 M 0374  1          ELSE  STATUS=SMG$$PUT_SCREEN(PBCB,LEN,ADDR,0,0,ATTR);
:   104-1     M 0375  1        IF NOT .STATUS THEN RETURN .STATUS
:   105       M 0376  1        END
:   106         0377  1 %;
:   107         0378  1
:   108         0379  1 !+
:   109         0380  1 ! $L
:   110         0381  1 ! --
:   111         0382  1 ! Macro $L linearizes a two dimensional subcript formed by a 1-based
:   112         0383  1 ! row number and a 1-based column number, into a single 0-based
:   113         0384  1 ! subscript.
:   114         0385  1 !-
:   115         0386  1
:   116         0387  1 MACRO
:   117         0388  1
:   118       M 0389  1        $L (ROW_NUMBER, COLUMN_NUMBER) =
:   119         0390  1        (ROW_NUMBER-1)*.NUM_COLS + COLUMN_NUMBER -1 %;
:   120         0391  1
:   121         0392  1 !+
:   122         0393  1 ! $MAKE_ROW_COL
:   123         0394  1 ! -------------
:   124         0395  1 ! Macro $MAKE_ROW_COL takes as an input a 0-based linear index into
:   125         0396  1 ! and array and converts it into a 1-based row and 1-based column
:   126         0397  1 ! form.  INDEX needs to be re-expressed as a quadword for use in the
:   127         0398  1 ! EDIV instruction.
:   128         0399  1 !-
:   129         0400  1
:   130         0401  1 MACRO
:   131         0402  1
:   132       M 0403  1     $MAKE_ROW_COL ( INDEX, ROW_NUMBER, COLUMN_NUMBER) =
:   133       M 0404  1     BEGIN       ! MAKE_ROW_COL
:   134       M 0405  1     BUILTIN
:   135       M 0406  1         EDIV;
:   136       M 0407  1     LOCAL
:   137       M 0408  1         WIDTH,
:   138       M 0409  1         LOCAL_INDEX : VECTOR [2, LONG];
:   139       M 0410  1     LOCAL_INDEX [1] = 0; ! Second longword is always 0
:   140       M 0411  1     LOCAL_INDEX [0] = .INDEX;
:   141       M 0412  1     WIDTH=.NUM_COLS;    ! Store width in longword
:   142       M 0413  1
:   143       M 0414  1     EDIV ( WIDTH, LOCAL_INDEX, ROW_NUMBER, COLUMN_NUMBER);
:   144       M 0415  1     ROW_NUMBER = .ROW_NUMBER + 1;
:   145       M 0416  1     COLUMN_NUMBER = .COLUMN_NUMBER + 1;
:   146       M 0417  1     END;        ! MAKE_ROW_COL
:   147         0418  1     %;
```

```
;  149     0419  1 %SBTTL 'SMG$$OUTPUT_MINIMAL_UPDATE - Calculate minimum update sequence'
;  150     0420  1 GLOBAL ROUTINE SMG$$OUTPUT_MINIMAL_UPDATE (P_PBCB) =
;  151     0421  1 !++
;  152     0422  1 ! FUNCTIONAL DESCRIPTION:
;  153     0423  1 !
;  154     0424  1 !       This routine compares CURR_TEXT and CURR_ATTR (which reflect
;  155     0425  1 !       what is currently on the screen), with NEW_TEXT and NEW_ATTR
;  156     0426  1 !       (which reflect what should be on the screen) and calculates a
;  157     0427  1 !       sequences of characters which when output to the screen changes
;  158     0428  1 !       the current screen contents to reflect the new (desired) screen
;  159     0429  1 !       contents. These characters are actually output to the screen.
;  160     0430  1 !
;  161     0431  1 ! CALLING SEQUENCE:
;  162     0432  1 !
;  163     0433  1 !       ret_status.wlc.v = SMG$$MINIMUM_UPDATE ( P_PBCB.rab.r)
;  164     0434  1 !
;  165     0435  1 ! FORMAL PARAMETERS:
;  166     0436  1 !
;  167     0437  1 !       P_PBCB,rab.r            Address of pasteboard control block
;  168     0438  1 !
;  169     0439  1 ! IMPLICIT INPUTS:
;  170     0440  1 !
;  171     0441  1 !       Contents of PBCB and WCB
;  172     0442  1 !
;  173     0443  1 ! IMPLICIT OUTPUTS:
;  174     0444  1 !
;  175     0445  1 !       Internal buffers change.
;  176     0446  1 !
;  177     0447  1 ! COMPLETION STATUS:
;  178     0448  1 !
;  179     0449  1 !       SS$_NORMAL      Normal successful completion
;  180     0450  1 !
;  181     0451  1 ! SIDE EFFECTS:
;  182     0452  1 !
;  183     0453  1 !       NONE
;  184     0454  1 !--
```

```
;  186     0455  2 BEGIN
;  187     0456  2
;  188     0457  2 BUILTIN
;  189     0458  2
;  190     0459  2         CMPC3;
;  191     0460  2
;  192     0461  2 BIND
;  193     0462  2
;  194     0463  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  195     0464  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  196     0465  2         NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  197     0466  2         NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  198     0467  2         CUR_TEXT        = .WCB[WCB_A_SCR_TEXT_BUF]      : VECTOR[,BYTE],
;  199     0468  2         CUR_ATTR        = .WCB[WCB_A_SCR_ATTR_BUF]      : VECTOR[,BYTE],
;  200     0469  2         NEW_TEXT        = .WCB[WCB_A_TEXT_BUF]          : VECTOR[,BYTE],
;  201     0470  2         NEW_ATTR        = .WCB[WCB_A_ATTR_BUF]          : VECTOR[,BYTE],
;  202     0471  2         NEW_LCV         = .WCB[WCB_A_LINE_CHAR]         : VECTOR[,BYTE],
;  203     0472  2         CUR_LCV         = .WCB[WCB_A_SCR_LINE_CHAR]     : VECTOR[,BYTE],
;  204     0473  2         OLD_CURSOR_ROW  =  WCB[WCB_W_OLD_CUR_ROW]       : WORD,
;  205     0474  2         OLD_CURSOR_COL  =  WCB[WCB_W_OLD_CUR_COL]       : WORD,
;  206     0475  2         NEW_CURSOR_ROW  =  WCB[WCB_W_CURR_CUR_ROW]      : WORD,
;  207     0476  2         NEW_CURSOR_COL  =  WCB[WCB_W_CURR_CUR_COL]      : WORD,
;  208     0477  2         SIZE            =  WCB[WCB_L_BUFSIZE],          ! Size of buffers
;  209     0478  2         FIRST_ROW       =  PBCB[PBCB_W_FIRST_CHANGED_ROW] : WORD,
;  210     0479  2         LAST_ROW        =  PBCB[PBCB_W_LAST_CHANGED_ROW]  : WORD,
;  211     0480  2         FIRST_COL       =  PBCB[PBCB_W_FIRST_CHANGED_COL] : WORD,
;  212     0481  2         LAST_COL        =  PBCB[PBCB_W_LAST_CHANGED_COL]  : WORD,
;  213     0482  2         TERM_TYPE       =  PBCB[PBCB_B_DEVTYPE]         : BYTE;
;  214     0483  2
;  215     0484  2 LOCAL
;  216     0485  2
;  217     0486  2         STATUS,         ! Status to return to caller
;  218     0487  2         INDEX,          ! Working index into the buffers
;  219     0488  2         ROW,            ! Working row number
;  220     0489  2         COL,            ! Working column number
;  221     0490  2         LEN,            ! local length
;  222     0491  2         ADJUSTED_COL,   ! Wide line adjusted column number
;  223     0492  2         CUR_TEXT_PTR : REF VECTOR [,BYTE],  ! Current pointer into
;  224     0493  2                                             ! current text buffer
;  225     0494  2         CUR_ATTR_PTR : REF VECTOR [,BYTE],  ! Current pointer into
;  226     0495  2                                             ! current attribute buffer
;  227     0496  2         NEW_TEXT_PTR : REF VECTOR [,BYTE],  ! Current pointer into new
;  228     0497  2                                             ! text buffer
;  229     0498  2         NEW_ATTR_PTR : REF VECTOR [,BYTE],  ! Current pointer into new
;  230     0499  2                                             ! attribute buffer
;  231     0500  2         END_ROW_INDEX,  ! Index to last character in current row
;  232     0501  2         RENDITION,      ! local rendition
;  233     0502  2         FINAL_INDEX,    ! local index representing end of a changed sequence
;  234     0503  2         CURSOR_ROW,     ! Current cursor row
;  235     0504  2         CURSOR_COL,     ! Current cursor column
;  236     0505  2
;  237     0506  2         NEW_CHARS_LEFT,
;001  :PLL1018  0507  2         CHARS_LEFT,     ! Number of characters left to be inspected.
;002  :PLL1018  0508  2                         ! Starts out equal to number of characters
;003  :PLL1018  0509  2                         ! in the four buffers.
;004  :PLL1018  0510  2         OLD_MODE;       ! Original mode bit settings
```

```
;  242          0511  2 !+
;  243          0512  2 ! If CTRL/O was typed previously, some QIO has returned with
;  244          0513  2 ! that success status and our CTRL/O bit is set.  We don't
;  245          0514  2 ! really know what the screen looks like anymore, so we
;  246          0515  2 ! clear out the screen buffer.
;  247          0516  2 !-
;  248          0517  2
;  249          0518  2 IF .PBCB[PBCB_V_CONTROLO]
;  250          0519  3   THEN  BEGIN   ! Clear screen buffer
;  251          0520  3         CH$FILL(0,.SIZE,CUR_TEXT);
;  252          0521  3         FIRST_ROW=1;
;  253          0522  3         FIRST_COL=1;
;  254          0523  3         LAST_ROW =.NUM_ROWS;
;  255          0524  3         LAST_COL =.NUM_COLS;
;  256          0525  3         PBCB[PBCB_V_CONTROLO]=0
;  257          0526  2         END;    ! Clear screen buffer
;  258          0527  2
;  259          0528  2 !+
;  260          0529  2 ! Initialize our working pointers into the buffers.
;  261          0530  2 ! For now: we invalidate the initial cursor position
;  262          0531  2 ! to force the first update to use full cursor addressing.
;  263          0532  2 !-
;  264          0533  2
;001  :PLL1018   0534  2 CURSOR_ROW=0;
;002  :PLL1018   0535  2 CURSOR_COL=0;
;003  :PLL1018   0536  2
;004  :PLL1018   0537  2 !+
;005  :PLL1018   0538  2 ! Most of the following output calls result in QIOs.  There is no
;006  :PLL1018   0539  2 ! need to use multiple output calls for line characteristics, rendition,
;007  :PLL1018   0540  2 ! cursor positioning, and text; all this could be combined in 1 QIO if
;008  :PLL1018   0541  2 ! the number of bytes fits within our QIO buffer.
;009  :PLL1018   0542  2 !
;010  :PLL1018   0543  2 ! As a quick fix, we'll turn on buffering at the beginning of this routine,
;011  :PLL1018   0544  2 ! and turn it off at the end.  This should result in 1 QIO if everything
;012  :PLL1018   0545  2 ! fits in the buffer.
;013  :PLL1018   0546  2 !+
;014  :PLL1018   0547  2
;015  :PLL1018   0548  2 OLD_MODE = .PBCB [PBCB_L_MODE_SETTINGS];
;016  :PLL1018   0549  2
;017  :PLL1018   0550  2 IF NOT .PBCB [PBCB_V_BUF_ENABLED]
;018  :PLL1018   0551  2 THEN
;019  :PLL1018   0552  2     PBCB [PBCB_L_MODE_SETTINGS] = .OLD_MODE OR SMG$M_BUF_ENABLED;
;020  :PLL1018   0553  2
;  271-6        0554  2 INCR ROW FROM .FIRST_ROW TO .LAST_ROW DO
;  272          0555  3     BEGIN       ! Scan row .ROW
;  273          0556  3     LOCAL PTEXT,PATTR;
;  274          0557  3     LOCAL BLANK_COL;
;  275          0558  3     LOCAL PRE_PTR_IN_ROW;       ! Pointer position just before first character
;  276          0559  3                                 ! in this row
;  277          0560  3     CUR_TEXT_PTR = CUR_TEXT+(.ROW-1)*.NUM_COLS;
;  278          0561  3     CUR_ATTR_PTR = CUR_ATTR+(.ROW-1)*.NUM_COLS;
;  279          0562  3     NEW_TEXT_PTR = NEW_TEXT+(.ROW-1)*.NUM_COLS;
;  280          0563  3     NEW_ATTR_PTR = NEW_ATTR+(.ROW-1)*.NUM_COLS;
;  281          0564  3
;  282          0565  3     IF .NEW_LCV[.ROW] EQL 0
;  283          0566  3     THEN
;  284          0567  3         CHARS_LEFT=.NUM_COLS
```

```
;  285     0568  3     ELSE
;  286     0569  3         CHARS_LEFT=.NUM_COLS/2;
;  287     0570  !     !  CHARS_LEFT=.NUM_COLS;
;  288     0571  3     PRE_PTR_IN_ROW=.CUR_TEXT_PTR-1;
;  289     0572  3
;  290     0573  3     !+
;  291     0574  3     ! See if the characteristics of this line must change.
;  292     0575  3     !-
;  293     0576  3
;  294     0577  3     IF .CUR_LCV[.ROW] NEQ .NEW_LCV[.ROW]
;  295     0578  3     THEN
;  296     0579  4         BEGIN   ! Change line characteristics
;  297     0580  4
;  298     0581  4         LOCAL   BUFFER  : VECTOR[SMG$K_LONGEST_SEQUENCE,BYTE],
;  299     0582  4                 BUFLEN;
;  300     0583  4
;  301     0584  4         EXTERNAL ROUTINE
;  302     0585  4
;  303     0586  4                 SMG$$OUTPUT;
;  304     0587  4
;  305     0588  4         !+
;  306     0589  4         ! Move to the desired row.
;  307     0590  4         !-
;  308     0591  4
;  309     0592  4         SMG$$FIND_MIN_CURSOR_POS (  PBCB,         ! Pasteboard Control block
;  310     0593  4                                    .CURSOR_ROW,   ! Current row
;  311     0594  4                                    .CURSOR_COL,   ! Current column
;  312     0595  4                                    .ROW,          ! Desired row
;  313     0596  4                                    1);            ! Desired column
;  314     0597  4
;  315     0598  4         !+
;  316     0599  4         ! Update our record of where we are on screen.
;  317     0600  4         !-
;  318     0601  4
;  319     0602  4         CURSOR_ROW = .ROW ;
;  320     0603  4         CURSOR_COL = 1 ;
;  321     0604  4
;  322     0605  4         BUFLEN=0;
;  323     0606  4
;  324     0607  4         !+
;  325     0608  4         ! Get escape sequence to change the line characteristics.
;  326     0609  4         !-
;  327     0610  4
;  328     0611  4         SELECTONE .NEW_LCV[.ROW] OF
;  329     0612  4                 SET
;  330     0613  4                 [LINE_K_WIDE]:          $SMG$GET_TERM_DATA(DOUBLE_WIDE);
;  331     0614  4                 [LINE_K_UPPER_HIGH]:    $SMG$GET_TERM_DATA(DOUBLE_HIGH_TOP);
;  332     0615  4                 [LINE_K_LOWER_HIGH]:    $SMG$GET_TERM_DATA(DOUBLE_HIGH_BOTTOM);
;  333     0616  5                 [LINE_K_NORMAL]:        $SMG$GET_TERM_DATA(SINGLE_HIGH)
;  334     0617  4                 TES;
;  335     0618  4
;  336     0619  4         !+
;  337     0620  4         ! Output it.
;  338     0621  4         !-
;  339     0622  4
;  340     0623  4         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  341     0624  5           THEN  BEGIN
```

```
;  342          0625  5                   STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  343          0626  5                                     .PBCB[PBCB_A_CAP_BUFFER]);
;001  :PLL1018  0627  5                   IF NOT .STATUS
;002  :PLL1018  0628  5                   THEN
;003  :PLL1018  0629  6                       BEGIN
;004  :PLL1018  0630  6                       PBCB [PBCB_L_MODE_SETTINGS] = .OLD_MODE;
;005  :PLL1018  0631  6                       RETURN .STATUS
;006  :PLL1018  0632  5                       END;
;  345-1        0633  5                   END
;  346          0634  5
;  347          0635  3           END;    ! Change line characteristics
;  348          0636  3
;  349          0637  3           !+
;  350          0638  3           ! Scan backwards looking for the largest sequence of trailing spaces.
;  351          0639  3           ! Set BLANK_COL to the column number of the start of such a suffix.
;  352          0640  3           !-
;  353          0641  3
;  354          0642  3           BLANK_COL=.NUM_COLS+1;
;  355          0643  3           PTEXT=NEW_TEXT+.ROW*.NUM_COLS;
;  356          0644  3           PATTR=NEW_ATTR+.ROW*.NUM_COLS;
;  357          0645  3           DECR C FROM .NUM_COLS TO 1 DO
;  358          0646  4               BEGIN
;  359          0647  4               PTEXT=.PTEXT-1;
;  360          0648  4               PATTR=.PATTR-1;
;  361          0649  5                       BEGIN
;  362          0650  5                       BIND    TEXT_CHAR=.PTEXT : BYTE,
;  363          0651  5                               ATTR_CHAR=.PATTR : BYTE;
;  364          0652  5                       IF .TEXT_CHAR EQL %C' ' AND .ATTR_CHAR EQL 0
;  365          0653  5                         THEN  BLANK_COL=.C
;  366          0654  5                         ELSE  EXITLOOP
;  367          0655  4                       END;
;  368          0656  3               END;
;  369          0657  3
;  370          0658  3           WHILE .CHARS_LEFT NEQ 0 DO
;  371          0659  4               BEGIN   ! scan
;  372          0660  4               IF .CUR_TEXT_PTR[0] EQL .NEW_TEXT_PTR[0] AND
;  373          0661  4                  .CUR_ATTR_PTR[0] EQL .NEW_ATTR_PTR[0]
;  374          0662  5                 THEN  BEGIN   ! Characters agree
;  375          0663  5                       CUR_TEXT_PTR=.CUR_TEXT_PTR+1;
;  376          0664  5                       CUR_ATTR_PTR=.CUR_ATTR_PTR+1;
;  377          0665  5                       NEW_TEXT_PTR=.NEW_TEXT_PTR+1;
;  378          0666  5                       NEW_ATTR_PTR=.NEW_ATTR_PTR+1;
;  379          0667  5                       CHARS_LEFT=.CHARS_LEFT-1
;  380          0668  5                       END     ! Characters agree
;  381          0669  5                 ELSE  BEGIN   ! Characters disagree
;  382          0670  5
;  383          0671  5                       INDEX=.CUR_TEXT_PTR-CUR_TEXT;
;  384          0672  5
;  391-6        0673  5                       COL=.CUR_TEXT_PTR-.PRE_PTR_IN_ROW;
;  392          0674  5
;  393          0675  5                       !+
;  394          0676  5                       ! At this point, the cursor is positioned at
;  395          0677  5                       ! .CURSOR_ROW, .CURSOR_COL.  The first character that
;  396          0678  5                       ! needs to be rewritten is at .ROW, .COL.
;  397          0679  5                       ! Determine a minimal update sequence to get us from
;  398          0680  5                       ! where cursor is to where it needs to be to do rewrite.
;  399          0681  5                       !-
```

```
;  400          0682  5
;  401          0683  5                    !+
;  402          0684  5                    ! Set the column to "unknown" if we are past the end of
;  403          0685  5                    ! the terminal width.  We cannot assume that the cursor
;  404          0686  5                    ! has become stuck in the last column, because the
;  405          0687  5                    ! user may have done a SET TERMINAL/WIDTH=n command
;  406          0688  5                    ! to shorten his logical terminal width.
;  407          0689  5                    !-
;  408          0690  5
;  409          0691  5                    IF .CURSOR_COL GTRU .NUM_COLS
;  410          0692  5                      THEN  CURSOR_COL=0;
;  411          0693  5
;  412          0694  5                    SMG$$FIND_MIN_CURSOR_POS (  PBCB,        ! Pasteboard Control block
;  413          0695  5                                                .CURSOR_ROW, ! Current row
;  414          0696  5                                                .CURSOR_COL, ! Current column
;  415          0697  5                                                .ROW,        ! Desired row
;  416          0698  5                                                .COL);       ! Desired column
;  417          0699  5
;  418          0700  5                    !+
;  419          0701  5                    ! Update our record of where we are on screen after
;  420          0702  5                    ! we output as much of the string as is currently in
;  421          0703  5                    ! our buffer.
;  422          0704  5                    !-
;  423          0705  5
;  424          0706  5                    CURSOR_ROW = .ROW ;
;  425          0707  5                    CURSOR_COL = .COL ;
;  426          0708  5
;  427          0709  5                    !+
;  428          0710  5                    ! Now that we are positioned at first difference,
;  429          0711  5                    ! figure out what needs to be written.
;  430          0712  5                    !-
;  431          0713  5
;  432          0714  5                    !+
;  433          0715  5                    ! If we are at or past the blank pointer, then
;  434          0716  5                    ! just blank the remainder of the line and exit.
;  435          0717  5                    !-
;  436          0718  5
;  437          0719  5                    IF .CURSOR_COL GEQU .BLANK_COL
;  438          0720  6                      THEN  BEGIN   ! erase rest of line
;  439          0721  6                            LOCAL STATUS;
;  440          0722  6                            STATUS=ERASE_LINE(PBCB);
;001  :PLL1018  0723  6                            IF NOT .STATUS
;002  :PLL1018  0724  6                            THEN
;003  :PLL1018  0725  7                                BEGIN
;004  :PLL1018  0726  7                                PBCB [PBCB_L_MODE_SETTINGS] = .OLD_MODE;
;005  :PLL1018  0727  7                                RETURN .STATUS
;006  :PLL1018  0728  6                                END;
;  442-1        0729  6                            EXITLOOP
;  443          0730  5                            END;    ! erase rest of line
;  444          0731  5
;  445          0732  5                    !+
;  446          0733  5                    ! Note that our linear position within the buffer
;  447          0734  5                    ! is given by the index INDEX.
;  448          0735  5                    ! We now calculate the linear position of the last
;  449          0736  5                    ! character on this row, storing the resulting index
;  450          0737  5                    ! in END_ROW_INDEX.
;  451          0738  5                    !-
```

```
;  452     0739  5
;  453     0740  5                     END_ROW_INDEX=$L(.ROW,.NUM_COLS);
;  454     0741  5
;  455     0742  5                     !+
;  456     0743  5                     ! We now must search between INDEX and END_ROW_INDEX
;  457     0744  5                     ! for the longest sequence (all of the same rendition)
;  458     0745  5                     ! of changed characters.
;  459     0746  5                     !-
;  460     0747  5
;  461     0748  5                     !+
;  462     0749  5                     ! Step 1: find the longest sequence of characters
;  463     0750  5                     ! that are all of the same rendition.
;  464     0751  5                     ! Put our currently desired attributes in RENDITION.
;  465     0752  5                     !-
;  466     0753  5
;  467     0754  5                     RENDITION = .NEW_ATTR[.INDEX];
;  468     0755  5                     FINAL_INDEX = .END_ROW_INDEX+1;
;  469     0756  5
;  470     0757  5                     !+
;  471     0758  5                     ! Set up FINAL_INDEX to be the first index part
;  472     0759  5                     ! the longest such difference sequence.
;  473     0760  5                     !-
;  474     0761  5
;  475     0762  5                     INCR I FROM  INDEX+1 TO .END_ROW_INDEX DO
;  476     0763  6                             BEGIN    ! scan for end of change
;  477     0764  7                             IF (.NEW_TEXT[.I] EQL .CUR_TEXT[.I] AND
;  478     0765  7                                 .NEW_ATTR[.I] EQL .CUR_ATTR[.I])
;  479     0766  6                             OR .NEW_ATTR[.I] NEQ .RENDITION
;  480     0767  7                               THEN  BEGIN    ! end-of-change
;  481     0768  7                                     FINAL_INDEX=.I;
;  482     0769  7                                     EXITLOOP
;  483     0770  6                                     END;     ! end-of-change
;  484     0771  5                             END;     ! scan for end of change
;  485     0772  5
;  486     0773  5                     !+
;  487     0774  5                     ! We now must update the screen from .INDEX to .FINAL_INDEX-1
;  488     0775  5                     ! positions using the attributes stored in RENDITION.
;  489     0776  5                     ! The final SPACE_COUNT positions are to be erased.
;  490     0777  5                     !-
;  491     0778  5
;  492     0779  5                     LEN=.FINAL_INDEX-.INDEX;
;  493     0780  5
;  494     0781  5                     IF .LEN GTRU 0
;  495     0782  6                       THEN  BEGIN    ! output revised sequence
;  496     0783  6                             $OUTPUT_STRING( .LEN,.NEW_TEXT_PTR,.RENDITION);
;  497     0784  6                             CURSOR_COL=.CURSOR_COL+.LEN
;  498     0785  5                             END;     ! output revised sequence
;  499     0786  5
;  500     0787  5                     !+
;  501     0788  5                     ! Update our pointers and the number of chars left.
;  502     0789  5                     !-
;  503     0790  5
;  504     0791  5                     CUR_TEXT_PTR =.CUR_TEXT_PTR+.LEN;
;  505     0792  5                     CUR_ATTR_PTR =.CUR_ATTR_PTR+.LEN;
;  506     0793  5                     NEW_TEXT_PTR =.NEW_TEXT_PTR +.LEN;
;  507     0794  5                     NEW_ATTR_PTR =.NEW_ATTR_PTR +.LEN;
;  508     0795  5
```

```
;  509          0796  5                   CHARS_LEFT=.CHARS_LEFT-.LEN
;  510          0797  5
;  511          0798  5                   END     ! Characters disagree
;  512          0799  5
;  513          0800  3           END;     ! scan
;  514          0801  2       END;         ! scan row .ROW
;  515          0802  2
;  516          0803  2 !+
;  517          0804  2 ! Make the two buffers agree.
;  518          0805  2 ! The screen now contains what we think should be there.
;  519          0806  2 !-
;  520          0807  2
;  521          0808  2 CH$MOVE(.SIZE,NEW_TEXT,CUR_TEXT);
;  522          0809  2 CH$MOVE(.SIZE,NEW_ATTR,CUR_ATTR);
;  523          0810  2 CH$MOVE(.NUM_ROWS+1,NEW_LCV,CUR_LCV);
;  524          0811  2
;  525          0812  2 !+
;  526          0813  2 ! Move the cursor to the place where the user thinks it is.
;  527          0814  2 ! (But only if we are not already there.)
;  528          0815  2 !-
;  529          0816  2
;  530          0817  2 IF .CUR_LCV[.NEW_CURSOR_ROW] NEQ 0
;  531          0818  2   THEN  ADJUSTED_COL=.CURSOR_COL
;  532          0819  2   ELSE  ADJUSTED_COL=2*.CURSOR_COL-1;
;  533          0820  2
;  534          0821  2 OLD_CURSOR_ROW=.CURSOR_ROW;
;  535          0822  2 OLD_CURSOR_COL=.CURSOR_COL;
;  536          0823  2
;  537          0824  2 SMG$$UPDATE_PHYSICAL_CURSOR(PBCB);
;  538          0825  2
:001  :PLL1018  0' 6  2 !+
:002  :PLL1018  0827  2 ! Reset to user's value.  Flush buffer if switching to non-buffered mode.
:003  :PLL1018  0828  2 !-
:004  :PLL1018  0829  2
:005  :PLL1018  0830  2 PBCB [PBCB_L_MODE_SETTINGS] = .OLD_MODE;
:006  :PLL1018  0831  2
:007  :PLL1018  0832  2 IF NOT .PBCB [PBCB_V_BUF_ENABLED]
:008  :PLL1018  0833  2 THEN
:009  :PLL1018  0834  2     SMG$$FLUSH_BUFFER (PBCB);
:010  :PLL1018  0835  2
;  539          0836  2 RETURN SS$_NORMAL
;  540          0837  2
;  541          0838  1 END;                    ! End of routine SMG$$OUTPUT_MINIMAL_UPDATE


                                                        .TITLE  SMG$MIN Minimal update calculation
                                                        .IDENT  \1-018\

                                                        .EXTRN  SMG$$OUTPUT, SMG$$FLUSH_BUFFER
                                                        .EXTRN  SMG$GET_TERM_DATA
                                                        .EXTRN  SMG$$PUT_SCREEN

                                                        .PSECT  _SMG$CODE,NOWRT,  SHR,  PIC,2

                                        0FFC 00000      .ENTRY  SMG$$OUTPUT_MINIMAL_UPDATE, Save R2,R3,R4,- ; 0420
                                                                R5,R6,R7,R8,R9,R10,R11                      :
                                5E      FEB8   CE  9E 00002     MOVAB   -328(SP), SP                        :
```

```
                                59      04    AC   D0 00007          MOVL    P_PBCB, R9                                      ; 0463
                                5A      08    A9   D0 0000B          MOVL    8(R9), R10                                      ; 0464
                                        14    AA   DD 0000F          PUSHL   20(R10)                                         ; 0467
                                        0C    AA   DD 00012          PUSHL   12(R10)                                         ; 0470
                        24      00D0    C9         06   E1 00015     BBC     #6, 208(R9), 1$                                 ; 0518
      28  AA                00          6E         00   2C 0001B     MOVC5   #0, (SP), #0, 40(R10), @4(SP)                   ; 0520
                                                04   BE    00021
                                00A8    C9         01   B0 00023     MOVW    #1, 168(R9)                                     ; 0521
                                00AC    C9         01   B0 00028     MOVW    #1, 172(R9)                                     ; 0522
                                00AA    C9    02   AA   B0 0002D     MOVW    2(R10), 170(R9)                                 ; 0523
                                00AE    C9    06   AA   B0 00033     MOVW    6(R10), 174(R9)                                 ; 0524
                                00D0    C9    40   8F   8A 00039     BICB2   #64, 208(R9)                                    ; 0525
                                              1C   AE   D4 0003F 1$:  CLRL    CURSOR_ROW                                      ; 0534
                                              10   AE   D4 00042     CLRL    CURSOR_COL                                      ; 0535
                                  14    AE    0C   A9   9E 00045     MOVAB   12(R9), 20(SP)                                  ; 0548
                                  24    AE    14   BE   D0 0004A     MOVL    @20(SP), OLD_MODE                               ;
                                        06    14   BE   E8 0004F     BLBS    @20(SP), 2$                                     ; 0550
                    14  BE          24    AE         01   C9 00053  BISL3   #1, OLD_MODE, @20(SP)                           ; 0552
                                  3C    AE  00AA   C9   3C 00059 2$:  MOVZWL  170(R9), 60(SP)                                 ; 0554
                                        52  00A8   C9   3C 0005F     MOVZWL  168(R9), ROW                                    ;
                                              52   D7   00064        DECL    ROW                                             ;
                                                 026B  31   00066     BRW     32$                                             ;
                                  08    AE    FF   A2   9E 00069 3$:  MOVAB   -1(R2), 8(SP)                                  ; 0560
                                        5B    06   AA   3C 0006E     MOVZWL  6(R10), R11                                     ;
                                  08    AE         5B   C4 00072     MULL2   R11, 8(SP)                                      ;
                        53        04    AE   08    AE   C1 00076     ADDL3   8(SP), 4(SP), CUR_TEXT_PTR                       ;
                    34  AE        18    AA   08    AE   C1 0007C     ADDL3   8(SP), 24(R10), CUR_ATTR_PTR                     ; 0561
                    20  AE        08    AA   08    AE   C1 00083     ADDL3   8(SP), 8(R10), NEW_TEXT_PTR                      ; 0562
                    30  AE              6E   08    AE   C1 0008A     ADDL3   8(SP), (SP), NEW_ATTR_PTR                        ; 0563
                                        50   2C BA42   9A 00090     MOVZBL  @44(R10)[ROW], R0                               ; 0565
                                                 06   12   00095     BNEQ    4$                                              ;
                                  28    AE         5B   D0 00097     MOVL    R11, CHARS_LEFT                                 ; 0567
                                                 05   11   0009B     BRB     5$                                              ;
                    28  AE              5B         02   C7 0009D 4$:  DIVL3   #2, R11, CHARS_LEFT                           ; 0569
                                        55      FF    A3   9E 000A2 5$:  MOVAB   -1(R3), PRE_PTR_IN_ROW                       ; 0571
                                        50      30 BA42   91   000A6     CMPB    @48(R10)[ROW], R0                           ; 0577
                                                    03    12   000AB     BNEQ    6$                                          ;
                                                 00F1     31   000AD     BRW     14$                                         ;
                                                    01    DD   000B0 6$:  PUSHL   #1                                          ; 0592
                                                    52    DD   000B2     PUSHL   ROW                                         ; 0595
                                                  18    AE    DD   000B4     PUSHL   CURSOR_COL                                  ; 0594
                                                  28    AE    DD   000B7     PUSHL   CURSOR_ROW                                  ; 0593
                                                    59    DD   000BA     PUSHL   R9                                          ; 0592
                                0000V   CF                  05   FB 000BC  CALLS   #5, SMG$$FIND_MIN_CURSOR_POS            ;
                                  1C    AE                  52   D0 000C1  MOVL    ROW, CURSOR_ROW                         ; 0602
                                  10    AE                  01   D0 000C5  MOVL    #1, CURSOR_COL                          ; 0603
                                                    50    D4   000C9     CLRL    BUFLEN                                      ; 0605
                                        50      2C BA42   9A   000CB     MOVZBL  @44(R10)[ROW], R0                           ; 0611
                                        01              50   91   000D0     CMPB    R0, #1                                  ; 0613
                                                        20   12   000D3     BNEQ    7$                                      ;
                                              00FC      C9   D5   000D5     TSTL    252(R9)                                 ;
                                                        6E   13   000D9     BEQL    10$                                     ;
                                                  44    AE   D4   000DB     CLRL    INPUT_ARGS                              ;
                                                  44    AE   9F   000DE     PUSHAB  INPUT_ARGS                              ;
                                              0104      C9   DD   000E1     PUSHL   260(R9)                                 ;
                                              0108      C9   9F   000E5     PUSHAB  264(R9)                                 ;
                                              0100      C9   9F   000E9     PUSHAB  256(R9)                                 ;
```

```
                 1C    AE      01CE    8F   3C 00CED          MOVZWL  #462, 28(SP)                    ;
                                       72   11 000F3          BRB     12$                             ;
                       02              50   91 000F5 7$:      CMPB    R0, #2                          ; 0614
                                       20   12 000F8          BNEQ    8$                              ;
                             00FC      C9   D5 000FA          TSTL    252(R9)                         ;
                                       49   13 000FE          BEQL    10$                             ;
                               44      AE   D4 00100          CLRL    INPUT_ARGS                      ;
                               44      AE   9F 00103          PUSHAB  INPUT_ARGS                      ;
                             0104      C9   DD 00106          PUSHL   260(R9)                         ;
                             0108      C9   9F 0010A          PUSHAB  264(R9)                         ;
                             0100      C9   9F 0010E          PUSHAB  256(R9)                         ;
                 1C    AE      01CD    8F   3C 00112          MOVZWL  #461, 28(SP)                    ;
                                       4D   11 00118          BRB     12$                             ;
                       03              50   91 0011A 8$:      CMPB    R0, #3                          ; 0615
                                       20   12 0011D          BNEQ    9$                              ;
                             00FC      C9   D5 0011F          TSTL    252(R9)                         ;
                                       24   13 00123          BEQL    10$                             ;
                               44      AE   D4 00125          CLRL    INPUT_ARGS                      ;
                               44      AE   9F 00128          PUSHAB  INPUT_ARGS                      ;
                             0104      C9   DD 0012B          PUSHL   260(R9)                         ;
                             0108      C9   9F 0012F          PUSHAB  264(R9)                         ;
                             0100      C9   9F 00133          PUSHAB  256(R9)                         ;
                 1C    AE      01CC    8F   3C 00137          MOVZWL  #460, 28(SP)                    ;
                                       28   11 0013D          BRB     12$                             ;
                                       50   D5 0013F 9$:      TSTL    R0                              ; 0616
                                       36   12 00141          BNEQ    13$                             ;
                             00FC      C9   D5 00143          TSTL    252(R9)                         ;
                                       06   12 00147          BNEQ    11$                             ;
                             0108      C9   D4 00149 10$:     CLRL    264(R9)                         ;
                                       2A   11 0014D          BRB     13$                             ;
                               44      AE   D4 0014F 11$:     CLRL    INPUT_ARGS                      ;
                               44      AE   9F 00152          PUSHA3  INPUT_ARGS                      ;
                             0104      C9   DD 00155          PUSHL   260(R9)                         ;
                             0108      C9   9F 00159          PUSHAB  264(R9)                         ;
                             0100      C9   9F 0015D          PUSHAB  256(R9)                         ;
                       1C    AE      023E    8F   3C 00161    MOVZWL  #574, 28(SP)                    ;
                               1C      AE   9F 00167 12$:     PUSHAB  28(SP)                          ;
                             00FC      C9   9F 0016A          PUSHAB  252(R9)                         ;
           00000000G   00              06   FB 0016E          CALLS   #6, SMG$GET_TERM_DATA           ;
                       01              50   E8 00175          BLBS    STATUS, 13$                     ;
                                       04   00178          RET                                     ;
                       50      0108    C9   D0 00179 13$:     MOVL    264(R9), R0                     ; 0623
                                       21   13 0017E          BEQL    14$                             ;
                             0104      C9   DD 00180          PUSHL   260(R9)                         ; 0626
                                       50   DD 00184          PUSHL   R0                              ; 062C
                                       59   DD 00186          PUSHL   R9                              ;
           00000000G   00              03   FB 00188          CALLS   #3, SMG$$OUTPUT                 ;
                 40    AE              50   D0 0018F          MOVL    R0, STATUS                      ;
                       0A      40      AE   E8 00193          BLBS    STATUS, 14$                     ; 0627
                 14    BE      24      AE   D0 00197          MOVL    OLD_MODE, @20(SP)               ; 0630
                       50      40      AE   D0 0019C          MOVL    STATUS, R0                      ; 0631
                                       04   001A0          RET                                     ;
                       54      01      AB   9E 001A1 14$:     MOVAB   1(R11), BLANK_COL               ; 0642
           50          52              5B   C5 001A5          MULL3   R11, ROW, R0                    ; 0643
                 0C    AE      08  BA40    9E 001A9          MOVAB   @8(R10)[R0], PTEXT              ;
           51          6E              50   C1 001AF          ADDL3   R0, (SP), PATTR                 ; 0644
                       50      01      AB   9E 001B3          MOVAB   1(R11), C                       ; 0645
```

```
                                12  11 001B7          BRB     16$
                           0C   AE  D7 001B9 15$:     DECL    PTEXT                                    ; 0647
                                51  D7 001BC          DECL    PATTR                                    ; 0648
                  20       0C   BE  91 001BE          CMPB    @PTEXT, #32                              ; 0652
                                0A  12 001C2          BNEQ    17$
                                61  95 001C4          TSTB    (PATTR)
                                06  12 001C6          BNEQ    17$
                  54            50  D0 001C8          MOVL    C, BLANK_COL                             ; 0653
                  EB            50  F5 001CB 16$:     SOBGTR  C, 15$                                   ; 0645
                           28   AE  D5 001CE 17$:     TSTL    CHARS_LEFT                               ; 0658
                                03  12 001D1          BNEQ    19$
                              00FE  31 001D3 18$:     BRW     32$
             20   BE            63  91 001D6 19$:     CMPB    (CUR_TEXT_PTR), @NEW_TEXT_PTP            ; 0660
                                17  12 001DA          BNEQ    20$
             30   BE       34   BE  91 001DC          CMPB    @CUR_ATTR_PTR, @NEW_ATTR_PTR             ; 0661
                                10  12 001E1          BNEQ    20$
                                53  D6 001E3          INCL    CUR_TEXT_PTR                             ; 0663
                           34   AE  D6 001E5          INCL    CUR_ATTR_PTR                             ; 0664
                           20   AE  D6 001E8          INCL    NEW_TEXT_PTR                             ; 0665
                           30   AE  D6 001EB          INCL    NEW_ATTR_PTR                             ; 0666
                           28   AE  D7 001EE          DECL    CHARS_LEFT                               ; 0667
                                DB  11 001F1          BRB     17$
        56        53       04   AE  C3 001F3 20$:     SUBL3   4(SP), CUR_TEXT_PTR, INDEX               ; 0671
   38   AE        53            55  C3 001F8          SUBL3   PRE_PTR_IN_ROW, CUR_TEXT_PTR, COL        ; 0673
                  5B       10   AE  D1 001FD          CMPL    CURSOR_COL, R11                          ; 0691
                                03  1B 00201          BLEQU   21$
                           10   AE  D4 00203          CLRL    CURSOR_COL                               ; 0692
                           38   AE  DD 00206 21$:     PUSHL   COL                                      ; 0698
                                52  DD 00209          PUSHL   ROW                                      ; 0697
                           18   AE  DD 0020B          PUSHL   CURSOR_COL                               ; 0696
                           28   AE  DD 0020E          PUSHL   CURSOR_ROW                               ; 0695
                                59  DD 00211          PUSHL   R9                                       ; 0694
          0000V   CF            05  FB 00213          CALLS   #5, SMG$$FIND_MIN_CURSOR_POS
             1C   AE            52  D0 00218          MOVL    ROW, CURSOR_ROW                          ; 0706
             10   AE       38   AE  D0 0021C          MOVL    COL, CURSOR_COL                          ; 0707
                  54       10   AE  D1 00221          CMPL    CURSOR_COL, BLANK_COL                    ; 0719
                                10  1F 00225          BLSSU   22$
                                59  DD 00227          PUSHL   R9                                       ; 0722
          0000V   CF            01  FB 00229          CALLS   #1, ERASE_LINE
                  A2            50  E8 0022E          BLBS    STATUS, 18$                              ; 0723
             14   BE       24   AE  D0 00231          MOVL    OLD_MODE, @20(SP)                        ; 0726
                                04  00236             RET                                              ; 0727
                  50       08   AE  D0 00237 22$:     MOVL    8(SP), R0                                ; 0740
                  58       FF AB40  9E 0023B          MOVAB   -1(R11)[R0], END_ROW_INDEX
                  50            6E  D0 00240          MOVL    (SP), R0                                 ; 075'
             2C   AE          6640  9A 00243          MOVZBL  (INDEX)[R0], RENDITION
                  57       01   A8  9E 00248          MOVAB   1(R8), FINAL_INDEX                       ; 0755
                  50            56  D0 0024C          MOVL    INDEX, I                                 ; 0762
                                28  11 0024F          BRB     26$
                  51       04   AE  D0 00251 23$:     MOVL    4(SP), R1                                ; 0764
                6041       08 BA40  91 00255          CMPB    @8(R10)[I], (I)[R1]
                                0B  12 0025B          BNEQ    24$
                  51            6E  D0 0025D          MOVL    (SP), R1                                 ; 0765
             18 BA40          6041  91 00260          CMPB    (I)[R1], @24(R10)[I]
                                0C  13 00266          BEQL    25$
                  51            6E  D0 00268 24$:     MOVL    (SP), R1                                 ; 0766
 2C   AE      6041   08         00  ED 0026B          CMPZV   #0, #8, (I)[R1], RENDITION
```

```
                                          05  13 00272       BEQL     26$                                     ;
                          57              50  D0 00274 25$:  MOVL     I, FINAL_INDEX                          ; 0768
                                          04  11 00277       BRB      27$                                     ; 0767
                D4        50              58  F3 00279 26$:  AOBLEQ   END_ROW_INDEX, I, 23$                   ; 0762
          18    AE        57              56  C3 0027D 27$:  SUBL3    INDEX, FINAL_INDEX, LEN                 ; 0779
                                          35  13 00282       BEQL     31$                                     ; 0781
                                   2C     AE  D5 00284       TSTL     RENDITION                               ; 0783
                                          13  12 00287       BNEQ     28$                                     ;
                                          7E  7C 00289       CLRQ     -(SP)                                   ;
                                   28     AE  DD 0028B       PUSHL    NEW_TEXT_PTR                            ;
                                   24     AE  DD 0028E       PUSHL    LEN                                     ;
                                          59  DD 00291       PUSHL    R9                                      ;
                00000000G  00             05  FB 00293       CALLS    #5, SMG$$PUT_SCREEN                     ;
                                          14  11 0029A       BRB      29$                                     ;
                                   2C     AE  DD 0029C 28$:  PUSHL    RENDITION                               ;
                                          7E  7C 0029F       CLRQ     -(SP)                                   ;
                                   2C     AE  DD 002A1       PUSHL    NEW_TEXT_PTR                            ;
                                   28     AE  DD 002A4       PUSHL    LEN                                     ;
                                          59  DD 002A7       PUSHL    R9                                      ;
                00000000G  00             06  FB 002A9       CALLS    #6, SMG$$PUT_SCREEN                     ;
                           01             50  E8 002B0 29$:  BLBS     STATUS, 30$                             ;
                                              04 002B3       RET                                              ;
                     10    AE      18     AE  C0 002B4 30$:  ADDL2    LEN, CURSOR_COL                         ; 0784
                           53      18     AE  C0 002B9 31$:  ADDL2    LEN, CUR_TEXT_PTR                       ; 0791
                     34    AE      18     AE  C0 002BD       ADDL2    LEN, CUR_ATTR_PTR                       ; 0792
                     20    AE      18     AE  C0 002C2       ADDL2    LEN, NEW_TEXT_PTR                       ; 0793
                     30    AE      18     AE  C0 002C7       ADDL2    LEN, NEW_ATTR_PTR                       ; 0794
                     28    AE      18     AE  C2 002CC       SUBL2    LEN, CHARS_LEFT                         ; 0796
                                        FEFA  31 002D1       BRW      17$                                     ; 0659
    FD8E        52         01      3C     AE  F1 002D4 32$:  ACBL     60(SP), #1, ROW, 3$                     ; 0554
          04    BE         08      BA     28  AA 28 002DB    MOVC3    40(R10), @8(R10), @4(SP)                ; 0808
          18    BA         00      BE     28  AA 28 002E2    MOVC3    40(R10), @0(SP), @24(R10)               ; 0809
                           50      02     AA  3C 002E9       MOVZWL   2(R10), R0                              ; 0810
                                          50  D6 002ED       INCL     R0                                      ;
          30    BA         2C      BA     50  28 002EF       MOVC3    R0, @44(R10), @48(R10)                  ;
                           50      20     AA  3C 002F5       MOVZWL   32(R10), R0                             ; 0817
                           50      30     AA  C0 002F9       ADDL2    48(R10), R0                             ;
                                          60  95 002FD       TSTB     (R0)                                    ;
                                          06  13 002FF       BEQL     33$                                     ;
                           50      10     AE  D0 00301       MOVL     CURSOR_COL, ADJUSTED_COL                ; 0818
                                          07  11 00305       BRB      34$                                     ;
                50         10      AE     01  78 00307 33$:  ASHL     #1, CURSOR_COL, ADJUSTED_COL            ; 0819
                                          50  D7 0030C       DECL     ADJUSTED_COL                            ;
                     24    AA      1C     AE  B0 0030E 34$:  MOVW     CURSOR_ROW, 36(R10)                     ; 0821
                     26    AA      10     AE  B0 00313       MOVW     CURSOR_COL, 38(R10)                     ; 0822
                                          59  DD 00318       PUSHL    R9                                      ; 0824
                     0000V CF             01  FB 0031A       CALLS    #1, SMG$$UPDATE_PHYSICAL_CURSOR         ;
                     14    BE      24     AE  D0 0031F       MOVL     OLD_MODE, @20(SP)                       ; 0830
                           09      14     BE  E8 00324       BLBS     @20(SP), 35$                            ; 0832
                                          59  DD 00328       PUSHL    R9                                      ; 0834
                00000000G  00             01  FB 0032A       CALLS    #1, SMG$$FLUSH_BUFFER                   ;
                           50             01  D0 00331 35$:  MOVL     #1, R0                                  ; 0836
                                              04 00334       RET                                              ; 0838

; Routine Size:  821 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
;  543     0839  1 %SBTTL 'SMG$$UPDATE_PHYSICAL_CURSOR'
;  544     0840  1 GLOBAL ROUTINE SMG$$UPDATE_PHYSICAL_CURSOR (P_PBCB) =
;  545     0841  1 !++
;  546     0842  1 ! FUNCTIONAL DESCRIPTION:
;  547     0843  1 !
;  548     0844  1 !       This routine forces the physical cursor to move to
;  549     0845  1 !       a new location specified in the WCB.
;  550     0846  1 !       It also updates any internal structures.
;  551     0847  1 !       The cursor is clipped to an appropriate place if it
;  552     0848  1 !       falls outside the physical screen.
;  553     0849  1 !
;  554     0850  1 ! CALLING SEQUENCE:
;  555     0851  1 !
;  556     0852  1 !       ret_status.wlc.v = SMG$$UPDATE_PHYSICAL_CURSOR ( P_PBCB.rab.r)
;  557     0853  1 !
;  558     0854  1 ! FORMAL PARAMETERS:
;  559     0855  1 !
;  560     0856  1 !       P_PBCB,rab.r            Address of pasteboard control block
;  561     0857  1 !
;  562     0858  1 ! IMPLICIT INPUTS:
;  563     0859  1 !
;  564     0860  1 !       WCB[WCB_W_CURR_CUR_ROW] Desired new row for physical cursor
;  565     0861  1 !
;  566     0862  1 !       WCB[WCB_W_CURR_CUR_COL] Desired new col for physical cursor
;  567     0863  1 !
;  568     0864  1 !       WCB[WCB_W_OLD_CUR_ROW]  Physical row where cursor now is
;  569     0865  1 !
;  570     0866  1 !       WCB[WCB_W_OLD_CUR_COL]  Physical col where cursor now is
;  571     0867  1 !
;  572     0868  1 ! IMPLICIT OUTPUTS:
;  573     0869  1 !
;  574     0870  1 !       WCB[WCB_W_CURR_CUR_ROW] New cursor row
;  575     0871  1 !
;  576     0872  1 !       WCB[WCB_W_CURR_CUR_COL] New cursor col
;  577     0873  1 !
;  578     0874  1 !       WCB[WCB_W_OLD_CUR_ROW]  New cursor row
;  579     0875  1 !
;  580     0876  1 !       WCB[WCB_W_OLD_CUR_COL]  New cursor col
;  581     0877  1 !
;  582     0878  1 ! COMPLETION STATUS:
;  583     0879  1 !
;  584     0880  1 !       SS$_NORMAL      Normal successful completion
;  585     0881  1 !
;  586     0882  1 ! SIDE EFFECTS:
;  587     0883  1 !
;  588     0884  1 !       The cursor may move to a new physical location
;  589     0885  1 !--
```

```
;  591     0886  2 BEGIN
;  592     0887  2
;  593     0888  2 BIND
;  594     0889  2
;  595     0890  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  596     0891  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  597     0892  2         NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  598     0893  2         NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  599     0894  2         NEW_LCV         = .WCB[WCB_A_LINE_CHAR]         : VECTOR[,BYTE],
;  600     0895  2         CUR_LCV         = .WCB[WCB_A_SCR_LINE_CHAR]     : VECTOR[,BYTE],
;  601     0896  2         OLD_CURSOR_ROW  =  WCB[WCB_W_OLD_CUR_ROW]       : SIGNED WORD,
;  602     0897  2         OLD_CURSOR_COL  =  WCB[WCB_W_OLD_CUR_COL]       : SIGNED WORD,
;  603     0898  2         NEW_CURSOR_ROW  =  WCB[WCB_W_CURR_CUR_ROW]      : SIGNED WORD,
;  604     0899  2         NEW_CURSOR_COL  =  WCB[WCB_W_CURR_CUR_COL]      : SIGNED WORD;
```

```
;  606     0900  2 IF .OLD_CURSOR_ROW NEQ .NEW_CURSOR_ROW
;  607     0901  2 OR .OLD_CURSOR_COL NEQ .NEW_CURSOR_COL
;  608     0902  3   THEN  BEGIN
;  609     0903  3
;  610     0904  3         !+
;  611     0905  3         ! If the desired location is off the screen,
;  612     0906  3         ! clip it to the nearest edge.
;  613     0907  3         !-
;  614     0908  3
;  615     0909  3         IF .NEW_CURSOR_ROW LSS 1
;  616     0910  3           THEN  NEW_CURSOR_ROW=1;
;  617     0911  3
;  618     0912  3         IF .NEW_CURSOR_COL LSS 1
;  619     0913  3           THEN  NEW_CURSOR_COL=1;
;  620     0914  3
;  621     0915  3         IF .NEW_CURSOR_ROW GTRU .NUM_ROWS
;  622     0916  3           THEN  NEW_CURSOR_ROW=.NUM_ROWS;
;  623     0917  3
;  624     0918  3         IF .NEW_CURSOR_COL GTRU .NUM_COLS
;  625     0919  3           THEN  NEW_CURSOR_COL=.NUM_COLS;
;  626     0920  3
;  627     0921  3         !+
;  628     0922  3         ! Physically move the cursor there.
;  629     0923  3         !-
;  630     0924  3
;  631     0925  3         SMG$$FIND_MIN_CURSOR_POS(
;  632     0926  3                   PBCB,                 ! Pasteboard control block
;  633     0927  3                   .OLD_CURSOR_ROW,      ! Current location on screen
;  634     0928  3                   .OLD_CURSOR_COL,
;  635     0929  3                   .NEW_CURSOR_ROW,      ! Desired location
;  636     0930  3                   .NEW_CURSOR_COL);
;  637     0931  3
;  638     0932  2         END;
;  639     0933  2
;  640     0934  2 !+
;  641     0935  2 ! Make the new and the old cursor positions agree.
;  642     0936  2 !-
;  643     0937  2
;  644     0938  2 OLD_CURSOR_ROW=.NEW_CURSOR_ROW;
;  645     0939  2 OLD_CURSOR_COL=.NEW_CURSOR_COL;
;  646     0940  2
;  647     0941  2 ! Special try:
;  648     0942  2 ! If current line is special, mark the column as unknown.
;  649     0943  2
;  650     0944  2 IF .CUR_LCV[.NEW_CURSOR_ROW] NEQ 0
;  651     0945  2   THEN  OLD_CURSOR_COL=0;
;  652     0946  2
;  653     0947  2 RETURN  SS$_NORMAL
;  654     0948  2
;  655     0949  1 END;


                                003C 00000         .ENTRY  SMG$$UPDATE_PHYSICAL_CURSOR, Save R2,R3,R4,-; 0840
                                                           R5                                           ;
```

```
                                50      04  AC  D0 00002        MOVL    P_PBCB, R0                              ; 0890
                                52      08  A0  D0 00006        MOVL    8(R0), R2                               ; 0891
                                55      30  A2  D0 0000A        MOVL    48(R2), R5                              ; 0895
                                53      20  A2  9E 0000E        MOVAB   32(R2), R3                              ; 0898
                                54      22  A2  9E 00012        MOVAB   34(R2), R4                              ; 0899
                                63      24  A2  B1 00016        CMPW    36(R2), (R3)                            ; 0900
                                            06  12 0001A        BNEQ    1$                                      ;
                                64      26  A2  B1 0001C        CMPW    38(R2), (R4)                            ; 0901
                                            41  13 00020        BEQL    6$                                      ;
                                            63  B5 00022 1$:    TSTW    (R3)                                    ; 0909
                                            03  14 00024        BGTR    2$                                      ;
                                63          01  B0 00026        MOVW    #1, (R3)                                ; 0910
                                            64  B5 00029 2$:    TSTW    (R4)                                    ; 0912
                                            03  14 0002B        BGTR    3$                                      ;
                                64          01  B0 0002D        MOVW    #1, (R4)                                ; 0913
                                51      02  A2  3C 00030 3$:    MOVZWL  2(R2), R1                               ; 0915
        51          63          10      00  EC 00035        CMPV    #0, #16, (R3), R1                       ;
                                            04  1B 00039        BLEQU   4$                                      ;
                                63      02  A2  B0 0003B        MOVW    2(R2), (R3)                             ; 0916
                                51      06  A2  3C 0003F 4$:    MOVZWL  6(R2), R1                               ; 0918
        51          64          10      00  EC 00043        CMPV    #0, #16, (R4), R1                       ;
                                            04  1B 00048        BLEQU   5$                                      ;
                                64      06  A2  B0 0004A        MOVW    6(R2), (R4)                             ; 0919
                                7E          64  32 0004E 5$:    CVTWL   (R4), -(SP)                             ; 0930
                                7E          63  32 00051        CVTWL   (R3), -(SP)                             ; 0929
                                7E      26  A2  32 00054        CVTWL   38(R2), -(SP)                           ; 0928
                                7E      24  A2  32 00058        CVTWL   36(R2), -(SP)                           ; 0927
                                            50  DD 0005C        PUSHL   R0                                      ; 0925
                        0000V   CF          05  FB 0005E        CALLS   #5, SMG$$FIND_MIN_CURSOR_POS            ;
                                50          63  32 00063 6$:    CVTWL   (R3), R0                                ; 0938
                          24    A2          50  B0 00066        MOVW    R0, 36(R2)                              ;
                          26    A2          64  B0 0006A        MOVW    (R4), 38(R2)                            ; 0939
                                          6045  95 0006E        TSTB    (R0)[R5]                                ; 0944
                                            03  13 00071        BEQL    7$                                      ;
                                        26  A2  B4 00073        CLRW    38(R2)                                  ; 0945
                                50          01  D0 00076 7$:    MOVL    #1, R0                                  ; 0947
                                                04 00079        RET                                             ; 0949

; Routine Size:  122 bytes,    Routine Base:  _SMG$CODE + 0335
```

```
;  657     0950  1 %SBTTL 'SMG$SET_PHYSICAL_CURSOR'
;  658     0951  1 GLOBAL ROUTINE SMG$SET_PHYSICAL_CURSOR (PBID,P_ROW,P_COL) =
;  659     0952  1 !++
;  660     0953  1 ! FUNCTIONAL DESCRIPTION:
;  661     0954  1 !
;  662     0955  1 !       This routine moves the physical cursor on a physical
;  663     0956  1 !       screen to a particular location.
;  664     0957  1 !
;  665     0958  1 ! CALLING SEQUENCE:
;  666     0959  1 !
;  667     0960  1 !       ret_status.wlc.v = SMG$SET_PHYSICAL_CURSOR ( PBID.rl.r,P_ROW.rl.r,
;  668     0961  1 !                                                    P_COL.rl.r)
;  669     0962  1 !
;  670     0963  1 ! FORMAL PARAMETERS:
;  671     0964  1 !
;  672     0965  1 !       PBID.rl.r               Pasteboard id
;  673     0966  1 !
;  674     0967  1 !       P_ROW.rl.r              The row number to move to
;  675     0968  1 !
;  676     0969  1 !       P_COL.rl.r              The column number to move to
;  677     0970  1 !
;  678     0971  1 ! IMPLICIT INPUTS:
;  679     0972  1 !
;  680     0973  1 !       NONE
;  681     0974  1 !
;  682     0975  1 ! IMPLICIT OUTPUTS:
;  683     0976  1 !
;  684     0977  1 !       NONE
;  685     0978  1 !
;  686     0979  1 ! COMPLETION STATUS:
;  687     0980  1 !
;  688     0981  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  689     0982  1 !       SMG$_INVPAS_ID  Invalid pasteboard id
;  690     0983  1 !       SMG$_INVROW     Position is not within pasteboard (off top or bottom)
;  691     0984  1 !       SMG$_INVCOL     Position is not within pasteboard (off left or right)
;  692     0985  1 !       SS$_NORMAL      Normal successful completion
;  693     0986  1 !
;  694     0987  1 ! SIDE EFFECTS
;  695     0988  1 !
;  696     0989  1 !       NONE
;  697     0990  1 !--
```

```
;  699     0991  2 BEGIN
;  700     0992  2 BIND
;  701     0993  2
;  702     0994  2         ROW             = .P_ROW,
;  703     0995  2         COL             = .P_COL;
;  704     0996  2
;  705     0997  2 LOCAL
;  706     0998  2
;  707     0999  2         STATUS,
;  708     1000  2         PBCB            : REF $PBCB_DECL,
;  709     1001  2         WCB             : REF $WCB_DECL;
;  710     1002  2
;  711     1003  2 EXTERNAL LITERAL
;  712     1004  2
;  713     1005  2         SMG$_INVROW,
;  714     1006  2         SMG$_INVCOL;
```

```
;  716          1007  2 $SMG$VALIDATE_ARGCOUNT(3,3);
;  717          1008  2
;  718          1009  2 $SMG$GET_PBCB(.PBID,PBCB);
;  719          1010  2
;  720          1011  2 WCB=.PBCB[PBCB_A_WCB];
;  721          1012  2
;  722          1013  3         BEGIN
;  723          1014  3
;  724          1015  3         BIND
;  725          1016  3
;  726          1017  3                 NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  727          1018  3                 NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  728          1019  3                 CUR_ROW         =  WCB[WCB_W_CURR_CUR_ROW]      : WORD,
;  729          1020  3                 CUR_COL         =  WCB[WCB_W_CURR_CUR_COL]      : WORD;
;  730          1021  3
;  731          1022  3         IF .ROW GTRU .NUM_ROWS
;  732          1023  3           THEN  RETURN  SMG$_INVROW;
;  733          1024  3         IF .COL GTRU .NUM_COLS
;  734          1025  3           THEN  RETURN  SMG$_INVCOL;
;  735          1026  3
;  736          1027  3         CUR_ROW=.ROW;
;  737          1028  3         CUR_COL=.COL;
;  738          1029  3
;  739          1030  2         END;
;  740          1031  2
;  741          1032  2 !+
;  742          1033  2 ! Immediately move it there now if batching is not in effect.
;  743          1034  2 !-
;  744          1035  2
;  745          1036  2 IF .PBCB[PBCB_L_BATCH_LEVEL] EQL 0
;  746          1037  3   THEN  BEGIN   ! Move cursor
;  747          1038  3         STATUS=SMG$$UPDATE_PHYSICAL_CURSOR(.PBCB);
;  748          1039  3         IF NOT .STATUS THEN RETURN .STATUS
;  749          1040  2         END;    ! Move cursor
;  750          1041  2
;  751          1042  2 RETURN  SS$_NORMAL
;  752          1043  2
;  753          1044  1 END;


                                                        .EXTRN  SMG$_INVROW, SMG$_INVCOL
                                                        .EXTRN  SMG$_WRONUMARG, SMG$_INVPAS_ID
                                                        .EXTRN  PBD_L_COUNT, PBD_A_PBCB
                                                        .EXTRN  PBD_V_PB_AVAIL

                                    0000 00000          .ENTRY  SMG$SET_PHYSICAL_CURSOR, Save nothing           ; 0951
                          03            6C  91 00002    CMPB    (AP), #3                                        ; 1007
                                        08  13 00005    BEQL    1$                                              ;
                          50 00000000G  8F  D0 00007    MOVL    #SMG$_WRONUMARG, R0                             ;
                                        04  0000E       RET                                                     ;
                          50        04  BC  D0 0000F 1$:  MOVL    @PBID, R0                                       ; 1009
                                        11  19 00013    BLSS    2$                                              ;
                   00000000G  00        50  D1 00015    CMPL    R0, PBD_L_COUNT                                 ;
                                        08  14 0001C    BGTR    2$                                              ;
                08 00000000G  00        50  E0 0001E    BBS     R0, PBD_V_PB_AVAIL, 3$                          ;
                          50 00000000G  8F  D0 00026 2$:  MOVL    #SMG$_INVPAS_ID, R0                             ;
```

```
                                                    04 0002D                RET
                                                51  00000000G0040  DO 0002E 3$:     MOVL    PBD_A_PBCB[R0], PBCB                    ;
                                                50        08   A1  DO 00036         MOVL    8(PBCB), WCB                            ; 1011
      08   BC        02   A0                    10             00  ED 0003A         CMPZV   #0, #16, 2(WCB), @P_ROW                 ; 1022
                                                               08  1E 00041         BGEQU   4$                                      ;
                                                50  00000000G  8F  DO 00043         MOVL    #SMG$_INVROW, R0                        ; 1023
                                                                   04 0004A         RET                                             ;
      0C   BC        06   A0                    10             00  ED 0004B 4$:     CMPZV   #0, #16, 6(WCB), @P_COL                 ; 1024
                                                               08  1F 00052         BGEQU   5$                                      ;
                                                50  00000000G  8F  DO 00054         MOVL    #SMG$_INVCOL, R0                        ; 1025
                                                                   04 0005B         RET                                             ;
                                          20    A0        08   BC  B0 0005C 5$:     MOVW    @P_ROW, 32(WCB)                         ; 1027
                                          22    A0        0C   BC  B0 00061         MOVW    @P_COL, 34(WCB)                         ; 1028
                                                     00A4   C1  D5 00066         TSTL    164(PBCB)                               ; 1036
                                                            0A  12 0006A         BNEQ    6$                                      ;
                                                            51  DD 0006C         PUSHL   PBCB                                    ; 1038
                                          FF13  CF              01  FB 0006E         CALLS   #1, SMG$$UPDATE_PHYSICAL_CURSOR         ;
                                                03              50  E9 00073         BLBC    STATUS, 7$                              ; 1039
                                                50              01  D0 00076 6$:     MOVL    #1, R0                                  ; 1042
                                                                   04 00079 7$:     RET                                             ; 1044

; Routine Size:  122 bytes,    Routine Base:  _SMG$CODE + 03AF
```

```
;  755     1045  1 %SBTTL 'SMG$$FIND_MIN_CURSOR_POS - Find minimum cursor pos. sequence'
;  756     1046  1 GLOBAL ROUTINE SMG$$FIND_MIN_CURSOR_POS (
;  757     1047  1                                          P_PBCB,
;  758     1048  1                                          LINE_NO,
;  759     1049  1                                          COL_NO,
;  760     1050  1                                          DESIRED_LINE_NO,
;  761     1051  1                                          DESIRED_COL_NO
;  762     1052  1                                          ) =
;  763     1053  1 !++
;  764     1054  1 ! FUNCTIONAL DESCRIPTION:
;  765     1055  1 !
;  766     1056  1 !
;  767     1057  1 ! CALLING SEQUENCE:
;  768     1058  1 !
;  769     1059  1 !       ret_status.wlc.v = SMG$$FIND_MIN_CURSOR_POS (
;  770     1060  1 !                                          P_PBCB.rab.r,
;  771     1061  1 !                                          LINE_NO.rl.v,
;  772     1062  1 !                                          COL_NO.rl.v,
;  773     1063  1 !                                          DESIRED_LINE_NO.rl.v,
;  774     1064  1 !                                          DESIRED_COL_NO.rl.v)
;  775     1065  1 !
;  776     1066  1 ! FORMAL PARAMETERS:
;  777     1067  1 !
;  778     1068  1 !       P_PBCB.rab.r            Address of PBCB
;  779     1069  1 !
;  780     1070  1 !       LINE_NO.rl.v            Current cursor line number
;  781     1071  1 !                               0 means it is unknown.
;  782     1072  1 !
;  783     1073  1 !       COL_NO.rl.v             Current cursor column number
;  784     1074  1 !                               0 means it is unknown.
;  785     1075  1 !
;  786     1076  1 !       DESIRED_LINE_NO.rl.v    Desired cursor line number position
;  787     1077  1 !
;  788     1078  1 !       DESIRED_COL_NO.rl.v     Desired cursor column number position
;  789     1079  1 !
;  790     1080  1 ! IMPLICIT INPUTS:
;  791     1081  1 !
;  792     1082  1 !       NONE
;  793     1083  1 !
;  794     1084  1 ! IMPLICIT OUTPUTS:
;  795     1085  1 !
;  796     1086  1 !       NONE
;  797     1087  1 !
;  798     1088  1 ! COMPLETION STATUS:
;  799     1089  1 !
;  800     1090  1 !       SS$_NORMAL      Normal successful completion
;  801     1091  1 !
;  802     1092  1 ! SIDE EFFECTS:
;  803     1093  1 !
;  804     1094  1 !       NONE
;  805     1095  1 !--
```

```
;  807     1096  2 BEGIN
;  808     1097  2
;  809     1098  2 BIND
;  810     1099  2
;  811     1100  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  812     1101  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  813     1102  2         NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  814     1103  2         NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  815     1104  2         CURR_TEXT       = .WCB[WCB_A_SCR_TEXT_BUF]      : VECTOR[,BYTE],
;  816     1105  2         CURR_ATTR       = .WCB[WCB_A_SCR_ATTR_BUF]      : VECTOR[,BYTE],
;  817     1106  2         LCV             = .WCB[WCB_A_LINE_CHAR]         : VECTOR[,BYTE],
;  818     1107  2         TERM_TYPE       =  PBCB[PBCB_B_DEVTYPE]         : BYTE;
;  819     1108  2
;  820     1109  2 LITERAL
;  821     1110  2
;  822     1111  2         INFINITY = 1000;        ! Prohibitively large number, used
;  823     1112  2                                 ! to reject a sequence.
;  824     1113  2
;  825     1114  2 BUILTIN
;  826     1115  2
;  827     1116  2         EDIV;
;  828     1117  2
;  829     1118  2 LOCAL
;  830     1119  2
;  831     1120  2         TRIAL_STRING : VECTOR [SMG$K_LONGEST_SEQUENCE,BYTE],
;  832     1121  2                                 ! Buffer in which to construct string
;  833     1122  2                                 ! to be output.
;  834     1123  2         TS_LEN,                 ! Current length of TRIAL_STRING.
;  835     1124  2         ADJUSTED_WIDTH,         ! Width or width/2
;  836     1125  2         SET_CUR_LEN;            ! Length of the general set_cursor
;  837     1126  2                                 ! sequence to reposition cursor.
```

```
;  839          1127  2 !+
;  840          1128  2 ! If the current position is unknown,
;  841          1129  2 ! then we must use the most general sequence.
;  842          1130  2 !-
;  843          1131  2
;  844          1132  2 IF .LINE_NO EQL 0
;  845          1133  2 OR .COL_NO  EQL 0
;  846          1134  2   THEN  RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  847          1135  2
;  848          1136  2 !+
;  849          1137  2 ! General strategy is to come up with a sequence of characters that
;  850          1138  2 ! will position us to the desired line and column number in less
;  851          1139  2 ! characters than a set_cursor sequence will need.
;  852          1140  2 ! The short-cut sequences to get to a specific line include:
;  853          1141  2 !       1. <LF's> to move down the screen.
;  854          1142  2 ! The short-cut sequences to get to a specific column include:
;  855          1143  2 !       1. <TAB> to tab-stop immediately before desired column and
;  856          1144  2 !       repeat a number of the current characters until we get to
;  857          1145  2 !       desired column position.
;  858          1146  2 !       2. <TAB> to tab-stop immediately beyond desired column and
;  859          1147  2 !       follow that by a number of <BS's> to get to the desired column.
;  860          1148  2 !       If at any point the trial sequence of characters gets to be
;  861          1149  2 ! greater than the set_cursor sequence, abandon the effort and use the
;  862          1150  2 ! set_cursor sequence.
;  863          1151  2 !-
;  864          1152  2
;  865          1153  2 TS_LEN = 0;                        ! Length of string constructed so far
;  866          1154  2
;  867          1155  2 !+
;  868          1156  2 ! Calculate what the cost of a set_cursor sequence is will be for the
;  869          1157  2 ! desired line and column number.  This will give us the lower bound we
;  870          1158  2 ! must beat if an alternate sequence is better.
;  871          1159  2 !-
;  872          1160  2
;  873          1161  2 $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.DESIRED_LINE_NO,.DESIRED_COL_NO);
;  874          1162  2 SET_CUR_LEN = .PBCB[PBCB_L_CAP_LENGTH];
;  875          1163  2
;  876          1164  2 !+
;  877          1165  2 ! Now see if we are already on the proper line.
;  878          1166  2 !-
;  879          1167  2
;  880          1168  2 IF .LINE_NO NEQ .DESIRED_LINE_NO
;  881          1169  2 THEN
;  882          1170  3         BEGIN   ! Adjust line number
;  883          1171  3         IF .DESIRED_LINE_NO LSS .LINE_NO
;  884          1172  3         THEN
;  885          1173  4             BEGIN       ! Move upward
;  886          1174  4
;  887          1175  4             !+
;  888          1176  4             ! No choice -- must use general cursor sequencing to move
;  889          1177  4             ! upward.  Output general set_cursor sequence
;  890          1178  4             ! (using DESIRED_LINE_NO and
;  891          1179  4             ! DESIRED_COL_NO) and return to caller.
;  892          1180  4             !-
;  893          1181  4
;  894          1182  4             RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO)
;  895          1183  4
```

```
;  896     1184  4              END         ! Move upward
;  897     1185  3          ELSE
;  898     1186  4              BEGIN       ! Move downward
;  899     1187  4              LOCAL
;  900     1188  4                  WIDE_WARNING,   ! TRUE if spanning across a wide line
;  901     1189  4                  LINES_DOWN ;    ! No. of lines down we need to move
;  902     1190  4
;  903     1191  4              !+
;  904     1192  4              ! See if we can reach DESIRED_LINE_NO in a number of <LF's>
;  905     1193  4              ! which is less than the number of characters in the
;  906     1194  4              ! set_cursor sequence.
;  907     1195  4              ! We do not permit line feed through the bottom of the scrolling
;  908     1196  4              ! region, since the cursor would not be able to cross it that way
;  909     1197  4              ! (and it would cause a scroll to occur).
;  910     1198  4              ! We do not permit line feed through a double wide (or double high)
;  911     1199  4              ! line, because in some cases, this doesn't work.  In particular,
;  912     1200  4              ! on a VT100, if you are in column 60, say and line feed down
;  913     1201  4              ! through a double wide line, when you get back to a single
;  914     1202  4              ! wide line, the cursor has now gotten to column 40!
;  915     1203  4              !-
;  916     1204  4
;  917     1205  4              LINES_DOWN = .DESIRED_LINE_NO - .LINE_NO;
;  918     1206  4
;  919     1207  4              !+
;  920     1208  4              ! Set WIDE_WARNING to TRUE if we would cross through or into or
;  921     1209  4              ! from a wide line.  Double high lines are considered to be wide.
;  922     1210  4              !-
;  923     1211  4
;  924     1212  4              WIDE_WARNING=0;
;  925     1213  4              IF .[CV[0] NEQ 0
;  926     1214  4              THEN
;  927     1215  4                  INCR L FROM .LINE_NO TO .DESIRED_LINE_NO DO
;  928     1216  4                          IF .[CV[.[] NEQ 0
;  929     1217  5                            THEN  BEGIN
;  930     1218  5                                  WIDE_WARNING=1;
;  931     1219  5                                  EXIT[OOP
;  932     1220  4                                  END;
;  933     1221  4
;  934     1222  4              IF (.LINES_DOWN LSS .SET_CUR_LEN)      AND
;  935     1223  5                 (.LINE_NO + .LINES_DOWN LEQU .PBCB[PBCB_W_BOT_SCROLL_LINE]
;  936     1224  4                  OR .LINE_NO GTRU .PBCB[PBCB_W_BOT_SCRO[L_LINE]) AND
;  937     1225  5                 (NOT .WIDE_WARNING)
;  938     1226  4              THEN
;  939     1227  5                  BEGIN   ! Do it with <LF's>
;  940     1228  5                  !+
;  941     1229  5                  ! Put (.LINES_DOWN) <LF's> into TRIAL_STRING and set
;  942     1230  5                  ! TS_LEN to .LINES_DOWN.
;  943     1231  5                  !-
;  944     1232  5                  CH$FILL (LF, .LINES_DOWN, TRIAL_STRING);
;  945     1233  5                  TS_LEN = .LINES_DOWN;
;  946     1234  5                  END     ! Do it with <LF's>
;  947     1235  4              ELSE
;  948     1236  5                  BEGIN   ! Too far
;  949     1237  5                  !+
;  950     1238  5                  ! Too far down or we would be crossing a lower scroll
;  951     1239  5                  ! boundary or a wide line -- use general set cursor sequence
;  952     1240  5                  !-
```

```
;  953     1241  5                  RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO)
;  954     1242  4                  END;        ! Too far
;  955     1243  3              END;            ! Move downward
;  956     1244  2          END;    ! Adjust line number
;  957     1245  2
;  958     1246  2  !+
;  959     1247  2  ! Reach here when we have constructed the minimal sequence to reach the
;  960     1248  2  ! desired line --not using general cursor addressing sequence.  TS_LEN
;  961     1249  2  ! tells us how long that sequence is.
;  962     1250  2  !-
;  963     1251  2
;  964     1252  2  IF .COL_NO NEQ .DESIRED_COL_NO
;  965     1253  2  THEN
;  966     1254  3          BEGIN    ! Column adjustment
;  967     1255  3          LOCAL
;  968     1256  3              LEAST_COST, ! Least cost among considered strategies
;  969     1257  3              BEST_STRAT, ! Best update strategy which is better
;  970     1258  3                          ! then general cursor positioning sequence.
;  971     1259  3              INDEX,      ! Index into CURR_TEXT and CURR_ATTR
;  972     1260  3              DCN_QUAD : VECTOR [2,LONG], ! Desired column number
;  973     1261  3                                          ! as a quadword
;  974     1262  3              DELTA_COL,  ! No. of columns between where we are and where
;  975     1263  3                          ! we want to be.
;  976     1264  3              NO_TABS,    ! No. of <TAB's> to get to tab-stop before
;  977     1265  3                          ! DESIRED_COL_NO.
;  978     1266  3              NO_RETYPES, ! No. of chars that need to be retyped if we
;  979     1267  3                          ! tab to tab-stop before
;  980     1268  3              NO_BS;      ! No. of <BS's> to get from tab-stop beyond
;  981     1269  3                          ! DESIRED_COL_NO back to DESIRED_COL_NO.
;  982     1270  3
;  983     1271  3          !+
;  984     1272  3          ! Construct short-cut sequence to position to desired column
;  985     1273  3          ! number.
;  986     1274  3          ! If earlier on line, 3 strategies are possible:
;  987     1275  3          !    1.  Do it with backspaces
;  988     1276  3          !    2.  Do it with <CR> and <TAB's> to tab-stop before followed
;  989     1277  3          !        by retypes.
;  990     1278  3          !    3.  Do it with <CR> and <TAB's> to tab-stop beyond followed
;  991     1279  3          !        by <BS's>.
;  992     1280  3          ! If later on line, 3 strategies are possible:
;  993     1281  3          !    4.  Do it with retypes.
;  994     1282  3          !    5.  Do it with <TAB's> to tab-stop before followed by
;  995     1283  3          !        retypes.
;  996     1284  3          !    6.  Do it with <TAB's> to tab-stop after followed by <BS's>.
;  997     1285  3          !-
;  998     1286  3
;  999     1287  3          !+
; 1000     1288  3          ! Calc. no of <TAB's> needed to get to tab-stop before
; 1001     1289  3          ! DESIRED_COL_NO and the no. of subsequent retypes needed.
; 1002     1290  3          !-
; 1003     1291  3
; 1004     1292  3          DCN_QUAD [0] = .DESIRED_COL_NO -1;
; 1005     1293  3          DCN_QUAD [1] = 0;
; 1006     1294  3          EDIV ( %REF(8), DCN_QUAD[0], NO_TABS, NO_RETYPES);
; 1007     1295  3
; 1008     1296  3          !+
; 1009     1297  3          ! If terminal doesn't support tabs,
```

```
;1010     1298  3          ! or user doesn't want them,
;1011     1299  3          ! then set NO_TABS to infinity.
;1012     1300  3          !-
;1013     1301  3
;1014     1302  3          IF .PBCB[PBCB_V_NOTABS] OR NOT .PBCB[PBCB_V_TABS]
;1015     1303  3            THEN  NO_TABS=INFINITY;
;1016     1304  3
;1017     1305  3          !+
;1018     1306  3          ! Calc. number of <BS's> needed if we go to tab-stop after
;1019     1307  3          ! DESIRED_COL_NO. This strategy can't be followed if the
;1020     1308  3          ! next tab stop is off past the right of the screen.  In
;1021     1309  3          ! that case, we make NO_BS prohibitively large.
;1022     1310  3          !-
;1023     1311  3
;1024     1312  3          IF .LCV[.DESIRED_LINE_NO] NEQ 0
;1025     1313  3            THEN  ADJUSTED_WIDTH=.NUM_COLS/2
;1026     1314  3            ELSE  ADJUSTED_WIDTH=.NUM_COLS;
;1027     1315  3
;1028     1316  3          IF (.NO_TABS+1)*8+1 LSSU .ADJUSTED_WIDTH
;1029     1317  3            THEN  NO_BS = 8 - .NO_RETYPES
;1030     1318  3            ELSE  NO_BS = INFINITY;
;1031     1319  3
;1032     1320  3          !+
;1033     1321  3          ! Set NO_BS to infinity if the terminal does not support backspacing.
;1034     1322  3          !-
;1035     1323  3
;1036     1324  3          IF NOT .PBCB[PBCB_V_BS]
;1037     1325  3            THEN  NO_BS=INFINITY;
;1038     1326  3
;1039     1327  3          !+
;1040     1328  3          ! In case we need to do retypes, calc. where in CURR_TEXT and
;1041     1329  3          ! CURR_ATTR we need to look.
;1042     1330  3          !-
;1043     1331  3
;1044     1332  3          INDEX = $L ( .DESIRED_LINE_NO, ((.NO_TABS*8) + 1));
;1045     1333  3
;1046     1334  3          IF .DESIRED_COL_NO  LEQ  .COL_NO
;1047     1335  3          THEN
;1048     1336  4              BEGIN       ! Earlier in line
;1049     1337  4              LOCAL
;1050     1338  4
;1051     1339  4                  S1_COST, S2_COST, S3_COST;      ! Cost of strategies
;1052     1340  4                                                  ! S1: just BS
;1053     1341  4                                                  ! S2: tabs then retype
;1054     1342  4                                                  ! S3: tabs then BS
;1055     1343  4
;1056     1344  4              ! Find the cost of stategies for moving back in line
;1057     1345  4
;1058     1346  4              IF .PBCB[PBCB_V_BS]
;1059     1347  4              THEN
;1060     1348  4                      S1_COST = .COL_NO - .DESIRED_COL_NO     ! No of <BS's>
;1061     1349  4              ELSE
;1062     1350  4                      S1_COST=INFINITY;
;1063     1351  4
;1064     1352  4              S2_COST = 1                     ! For <CR>
;1065     1353  4                      + .NO_TABS              ! For no. of tabs to tab-stop
;1066     1354  4                                              ! before
```

```
;1067   1355  4                          + .NO_RETYPES;   ! For no. of retypes
;1068   1356  4
;1069   1357  4              S3_COST = 1                  ! For <CR>
;1070   1358  4                        + .NO_TABS + 1     ! For no. of tabs to tab-stop
;1071   1359  4                                           ! after
;1072   1360  4                        +   '0_BS;         ! For no. of <BS's>
;1073   1361  4
;1074   1362  4              ! Find best strategy for moving backward in line
;1075   1363  4
;1076   1364  4              BEST_STRAT = 1;              LEAST_COST = .S1_COST;
;1077   1365  4
;1078   1366  4              IF .S2_COST LSS .LEAST_COST THEN
;1079   1367  4                  BEGIN  BEST_STRAT = 2;  LEAST_COST = .S2_COST;  END;
;1080   1368  4
;1081   1369  4              IF .S3_COST LSS .LEAST_COST THEN
;1082   1370  4                  BEGIN BEST_STRAT =3 ;   LEAST_COST = .S3_COST; END;
;1083   1371  4              END ! Earlier in line
;1084   1372  4
;1085   1373  3          ELSE
;1086   1374  3
;1087   1375  4              BEGIN       ! Later in line
;1088   1376  4              LOCAL
;1089   1377  4                  S4_COST, S5_COST, S6_COST;      ! Cost of strategies
;1090   1378  4
;1091   1379  4              ! Find costs of strategies for moving forward in line
;1092   1380  4
;1093   1381  4              S4_COST = .DESIRED_COL_NO - .COL_NO; ! For just retypes
;1094   1382  4
;1095   1383  4              IF (.NO_TABS * 8)+1 GTR .COL_NO AND .PBCB[PBCB_V_TABS]
;1096   1384  4                  AND NOT .PBCB[PBCB_V_NOTABS]
;1097   1385  4              THEN
;1098   1386  5                  BEGIN   ! Tabbing forward is possible
;1099   1387  5                  LOCAL
;1100   1388  5                      COL_QUAD : VECTOR [2,LONG], ! COL_NO as quadword
;1101   1389  5                      NEW_NO_TABS,
;1102   1390  5                      NEW_NO_RETYPES;
;1103   1391  5
;1104   1392  5                  COL_QUAD [0]  = .COL_NO  - 1;
;1105   1393  5                  COL_QUAD [1] = 0;
;1106   1394  5                  EDIV (%REF(8), COL_QUAD [0], NEW_NO_TABS, NEW_NO_RETYPES);
;1107   1395  5                  NO_TABS = .NO_TABS - .NEW_NO_TABS;
;1108   1396  5                  S5_COST = .NO_TABS          ! For no. of tabs to tab-stop
;1109   1397  5                                              ! before from current position
;1110   1398  5                          + .NO_RETYPES;! For no. of retypes
;1111   1399  5
;1112   1400  5                  S6_COST = .NO_TABS + 1  ! For no. of tabs to tab-stop
;1113   1401  5                                          ! after from current position
;1114   1402  5                          + .NO_BS;       ! For no. of <BS's>
;1115   1403  5                  END     ! Tabbing forward is possible
;1116   1404  4              ELSE
;1117   1405  5                  BEGIN   ! Tabbing forward not possible
;1118   1406  5                  S5_COST = INFINITY;     ! Set to prohibitive value
;1119   1407  5                  S6_COST = INFINITY;     ! Set to prohibitive value
;1120   1408  4                  END;    ! Tabbing forward not possible
;1121   1409  4
;1122   1410  4              ! Find best strategy
;1123   1411  4
```

```
;  1124     1412  4              BEST_STRAT = 4;                         LEAST_COST = .S4_COST;
;  1125     1413  4
;  1126     1414  4              IF .S5_COST LSS .LEAST_COST THEN
;  1127     1415  4                  BEGIN  BEST_STRAT = 5;  LEAST_COST = .S5_COST;  END;
;  1128     1416  4
;  1129     1417  4              IF .S6_COST LSS .LEAST_COST THEN
;  1130     1418  4                  BEGIN  BEST_STRAT = 6; LEAST_COST = .S6_COST; END;
;  1131     1419  3              END;        ! Later in line
;  1132     1420  3
;  1133     1421  3          IF .TS_LEN + .LEAST_COST GTR .SET_CUR_LEN
;  1134     1422  3          THEN
;  1135     1423  4              BEGIN       ! Abandon effort
;  1136     1424  4              RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO)
;  1137     1425  3              END;        ! Abandon effort
;  1138     1426  3
;  1139     1427  3          CASE .BEST_STRAT FROM 1 TO 6 OF
;  1140     1428  3          SET
;  1141     1429  4              [1]:BEGIN   ! Backspaces only.
;  1142     1430  4                  NO_BS = .COL_NO - .DESIRED_COL_NO ;
;  1143     1431  4                  CH$FILL ( BS, .NO_BS, TRIAL_STRING [.TS_LEN]);
;  1144     1432  4                  TS_LEN = .TS_LEN + .NO_BS;
;  1145     1433  3                  END;    ! Backspace only.
;  1146     1434  3
;  1147     1435  4              [2]:BEGIN   ! <CR>, <TAB's> to tab-stop before, retypes.
;  1148     1436  4
;  1149     1437  4                  !+
;  1150     1438  4                  ! If there are actually characters to be retyped and
;  1151     1439  4                  ! attributes are involved, give up and resort to general
;  1152     1440  4                  ! cursor positioning sequence.
;  1153     1441  4                  ! It will cost us too much to select-graphic-rendition
;  1154     1442  4                  ! and undo select graphic rendition.
;  1155     1443  4                  !-
;  1156     1444  4
;  1157     1445  4                  IF .NO_RETYPES NEQ 0  AND
;  1158     1446  4                     CH$COMPARE (0, 0,     ! len, addr
;  1159     1447  4                                 .NO_RETYPES, CURR_ATTR[.INDEX],
;  1160     1448  4                                 0         ! fill
;  1161     1449  4                                ) NEQ 0
;  1162     1450  4                  THEN
;  1163     1451  4                      RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  1164     1452  4
;  1165     1453  4                  TRIAL_STRING [.TS_LEN] = CR;
;  1166     1454  4                  TS_LEN = .TS_LEN + 1;
;  1167     1455  4                  CH$FILL ( TAB, .NO_TABS, TRIAL_STRING [.TS_LEN]);
;  1168     1456  4                  TS_LEN = .TS_LEN + .NO_TABS;
;  1169     1457  4                  CH$MOVE ( .NO_RETYPES, CURR_TEXT [.INDEX],
;  1170     1458  4                            TRIAL_STRING [.TS_LEN]);
;  1171     1459  4                  TS_LEN = .TS_LEN + .NO_RETYPES;
;  1172     1460  3                  END;    ! <CR>, <TAB's> to tab-stop before, retypes.
;  1173     1461  3
;  1174     1462  4              [3]:BEGIN   ! <CR>, <TAB's> to tab-stop after, <BS's>
;  1175     1463  4                  TRIAL_STRING [.TS_LEN] = CR;
;  1176     1464  4                  TS_LEN = .TS_LEN + 1;
;  1177     1465  4                  CH$FILL ( TAB, .NO_TABS + 1, TRIAL_STRING [.TS_LEN]);
;  1178     1466  4                  TS_LEN = .TS_LEN + .NO_TABS + 1;
;  1179     1467  4                  CH$FILL ( BS, .NO_BS, TRIAL_STRING [.TS_LEN]);
;  1180     1468  4                  TS_LEN = .TS_LEN + .NO_BS;
```

```
;  1181     1469  3                END;    ! <CR>, <TAB's> to tab-stop after, <BS's>
;  1182     1470  3
;  1183     1471  4            [4]:BEGIN   ! Retypes only.
;  1184     1472  4
;  1185     1473  4                !+
;  1186     1474  4                ! If there are actually characters to be retyped and
;  1187     1475  4                ! attributes are involved, give up and resort to general
;  1188     1476  4                ! cursor positioning sequence.
;  1189     1477  4                ! It will cost us too much to select-graphic-rendition
;  1190     1478  4                ! and undo select graphic rendition.
;  1191     1479  4                !-
;  1192     1480  4
;  1193     1481  4                NO_RETYPES = .DESIRED_COL_NO - .COL_NO;
;  1194     1482  4                INDEX = $L ( .DESIRED_LINE_NO, .COL_NO);
;  1195     1483  4                IF .NO_RETYPES NEQ 0  AND
;  1196     1484  4                   CH$COMPARE (0, 0,     ! len, addr
;  1197     1485  4                               .NO_RETYPES, CURR_ATTR[.INDEX],
;  1198     1486  4                               0         ! fill
;  1199     1487  4                              ) NEQ 0
;  1200     1488  4                THEN
;  1201     1489  4                    RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  1202     1490  4
;  1203     1491  4                CH$MOVE ( .NO_RETYPES, CURR_TEXT [.INDEX],
;  1204     1492  4                          TRIAL_STRING [.TS_LEN]);
;  1205     1493  4                TS_LEN = .TS_LEN + .NO_RETYPES;
;  1206     1494  3                END;    ! Retypes only.
;  1207     1495  3
;  1208     1496  4            [5]:BEGIN   ! <TAB's> to tab-stop before, retypes.
;  1209     1497  4
;  1210     1498  4                !+
;  1211     1499  4                ! If there are actually characters to be retyped and
;  1212     1500  4                ! attributes are involved, give up and resort to general
;  1213     1501  4                ! cursor positioning sequence.
;  1214     1502  4                ! It will cost us too much to select-graphic-rendition
;  1215     1503  4                ! and undo select graphic rendition.
;  1216     1504  4                !-
;  1217     1505  4
;  1218     1506  4                IF .NO_RETYPES NEQ 0  AND
;  1219     1507  4                   CH$COMPARE (0, 0,     ! len, addr
;  1220     1508  4                               .NO_RETYPES, CURR_ATTR[.INDEX],
;  1221     1509  4                               0         ! fill
;  1222     1510  4                              ) NEQ 0
;  1223     1511  4                THEN
;  1224     1512  4                    RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  1225     1513  4
;  1226     1514  4                CH$FILL ( TAB, .NO_TABS, TRIAL_STRING [.TS_LEN]);
;  1227     1515  4                TS_LEN = .TS_LEN + .NO_TABS;
;  1228     1516  4                CH$MOVE ( .NO_RETYPES, CURR_TEXT [.INDEX],
;  1229     1517  4                          TRIAL_STRING [.TS_LEN]);
;  1230     1518  4                TS_LEN = .TS_LEN + .NO_RETYPES;
;  1231     1519  3                END;    ! <TAB's> to tab-stop before, retypes.
;  1232     1520  3
;  1233     1521  4            [6]:BEGIN   ! <TAB's> to tab-stop after, <BS's>.
;  1234     1522  4                CH$FILL ( TAB, .NO_TABS + 1, TRIAL_STRING [.TS_LEN]);
;  1235     1523  4                TS_LEN = .TS_LEN + .NO_TABS + 1;
;  1236     1524  4                CH$FILL ( BS, .NO_BS, TRIAL_STRING [.TS_LEN]);
;  1237     1525  4                TS_LEN = .TS_LEN + .NO_BS;
```

```
;1238     1526  3                   END;    ! <TAB's> to tab-stop after, <BS's>.
;1239     1527  3           TES;
;1240     1528  2           END;    ! Column adjustment
;1241     1529  2
;1242     1530  2 !+
;1243     1531  2 ! At this point in the code we have a proper sequence of characters to
;1244     1532  2 ! reposition the cursor from .LINE_NO/.COL_NO to .DESIRED_LINE_NO/
;1245     1533  2 ! .DESIRED_COL_NO with a relatively minimum number of characters.
;1246     1534  2 ! This sequence of characters is contained in TRIAL_STRING and its
;1247     1535  2 ! length is contained in .TS_LEN
;1248     1536  2 ! We output this string to the screen.
;1249     1537  2 !-
;1250     1538  2
;1251     1539  2 $OUTPUT_STRING(.TS_LEN,TRIAL_STRING,0);
;1252     1540  2
;1253     1541  2 RETURN SS$_NORMAL
;1254     1542  2
;1255     1543  1 END;             ! End of routine SMG$$FIND_MIN_CURSOR_POS


                                        0FFC 00000          .ENTRY  SMG$$FIND_MIN_CURSOR_POS, Save R2,R3,R4,R5,-  ;1046
                                                                    R6,R7,R8,R9,R10,R11                               ;
                        5E      FEE8    CE   9E 00002       MOVAB   -280(SP), SP                                      ;
                                  04    AC   DD 00007       PUSHL   P_PBCB                                            ;1100
                   50           6E      08   C1 0000A       ADDL3   #8, (SP), R0                                      ;1101
                                5B      60   D0 0000E       MOVL    (R0), R11                                         ;
                                  08    AC   D5 00011       TSTL    LINE_NO                                           ;1132
                                        7D   13 00014       BEQL    3$                                                ;
                                  0C    AC   D5 00016       TSTL    COL_NO                                            ;1133
                                        78   13 00019       BEQL    3$                                                ;
                                        57   D4 0001B       CLRL    TS_LEN                                            ;1153
                   50                6E 000000FC 8F   C1 0001D       ADDL3   #252, (SP), R0                            ;1161
                                        60   D5 00025       TSTL    (R0)                                              ;
                                        0C   12 00027       BNEQ    1$                                                ;
                   52                6E 00000108 8F   C1 00029       ADDL3   #264, (SP), R2                            ;
                                        62   D4 00031       CLRL    (R2)                                              ;
                                        4C   11 00033       BRB     2$                                                ;
                            10    AE          02   D0 00035 1$:     MOVL    #2, INPUT_ARGS                            ;
                            14    AE      10  AC   7D 00039       MOVQ    DESIRED_LINE_NO, INPUT_ARGS+4             ;
                                    10  AE   9F   0003E       PUSHAB  INPUT_ARGS                                        ;
                   53       04    AE 00000104 8F   C1 00041       ADDL3   #260, 4(SP), R3                           ;
                                        63   DD 0004A       PUSHL   (R3)                                              ;
                   52       08    AE 00000108 8F   C1 0004C       ADDL3   #264, 8(SP), R2                           ;
                                        52   DD 00055       PUSHL   R2                                                ;
                   50       0C    AE 00000100 8F   C1 00057       ADDL3   #256, 12(SP), R0                          ;
                                        50   DD 00060       PUSHL   R0                                                ;
                            18    AE      023A 8F   3C 00062       MOVZWL  #570, 24(SP)                              ;
                                    18  AE   9F   00068       PUSHAB  24(SP)                                            ;
                   50       14    AE 000000FC 8F   C1 0006B       ADDL3   #252, 20(SP), R0                          ;
                                        50   DD 00074       PUSHL   R0                                                ;
                00000000G       00          06   FB 00076       CALLS   #6, SMG$GET_TERM_DATA                     ;
                                01          50   E8 0007D       BLBS    STATUS, 2$                                ;
                                            04   00080       RET                                                       ;
                            08    AE          62   D0 00081 2$:     MOVL    (R2), SET_CUR_LEN                         ;1162
```

```
                        10    AC          08    AC   D1 00085        CMPL    LINE_NO, DESIRED_LINE_NO                    ; 1168
                                                65   13 0008A        BEQL    9$
                        08    AC          10    AC   D1 0008C        CMPL    DESIRED_LINE_NO, LINE_NO                    ; 1171
                                                03   18 00091        BGEQ    4$
                                              023E   31 00093 3$:    BRW     39$
                  56    10    AC          08    AC   C3 00096 4$:    SUBL3   LINE_NO, DESIRED_LINE_NO, LINES_DOWN        ; 1205
                                                51   D4 00099C       CLRL    WIDE_WARNING                                ; 1212
                                          2C    BB   95 0009E        TSTB    @44(R11)                                    ; 1213
                                                17   13 000A1        BEQL    7$
                  50    08    AC                01   C3 000A3        SUBL3   #1, LINE_NO, L                              ; 1215
                                                08   11 000A8        BRB     6$
                                          2C  BB40   95 000AA 5$:    TSTB    @44(R11)[L]                                 ; 1216
                                                05   13 000AE        BEQL    6$
                              51                01   D0 000B0        MOVL    #1, WIDE_WARNING                            ; 1218
                                                05   11 000B3        BRB     7$                                          ; 1217
                  F0          50          10    AC   F3 000B5 6$:    AOBLEQ  DESIRED_LINE_NO, L, 5$                      ; 1216
                        08    AE                56   D1 000BA 7$:    CMPL    LINES_DOWN, SET_CUR_LEN                     ; 1222
                                                D3   18 000BE        BGEQ    3$
                  50          56          08    AC   C1 000C0        ADDL3   LINE_NO, LINES_DOWN, R0                     ; 1223
                  52          6E 000000F6       8F   C1 000C5        ADDL3   #246, (SP), R2
         50       62          10                00   ED 000CD        CMPZV   #0, #16, (R2), R0
                                                10   1E 000D2        BGEQU   8$
                  50          6E 000000F6       8F   C1 000D4        ADDL3   #246, (SP), R0                              ; 1224
   08    AC       60          10                00   ED 000DC        CMPZV   #0, #16, (R0) LINE_NO
                                                AF   1E 000E2        BGEQU   3$
                              AC                51   E8 000E4 8$     BLBS    WIDE_WARNING, 3$                            ; 1225
         56       0A          6E                00   2C 000E7        MOVC5   #0, (SP), #10, LINES_DOWN, TRIAL_STRING     ; 1232
                                          1C    AE      000EC
                              57                56   D0 000EE        MOVL    LINES_DOWN, TS_LEN                          ; 1233
                              54          0C    AC   D0 000F1 9$     MOVL    COL_NO, R4                                  ; 1252
                        14    AC                54   D1 000F5        CMPL    R4, DESIRED_COL_NO
                                                03   12 000F9        BNEQ    10$
                                              0218   31 000FB        BRW     46$
            14    AE    14    AC                01   C3 000FE 10$.   SUBL3   #1, DESIRED_COL_NO, DCN_QUAD                ; 1292
                                          18    AE   D4 00104        CLRL    DCN_QUAD+4                                  ; 1293
         59       56    14    AE                08   7B 00107        EDIV    #8, DCN_QUAD, NO_TABS, NO_RETYPES           ; 1294
                  50          6E                0C   C1 0010D        ADDL3   #12, (SP), R0                               ; 1302
                  0C          60                03   E0 00111        BBS     #3, (R0), 11$
                  51          6E 000000FA       8F   C1 00115        ADDL3   #250, (SP), R1
                  05          61                02   E0 0011D        BBS     #2, (R1), 12$
                              56        03E8    8F   3C 00121 11$:   MOVZWL  #1000, NO_TABS                              ; 1303
                              50          10    AC   D0 00126 12$    MOVL    DESIRED_LINE_NO, R0                         ; 1312
                                          2C  BB40   95 0012A        TSTB    @44(R11)[R0]
                                                0A   13 0012E        BEQL    13$
                              51          06    AB   3C 00130        MOVZWL  6(R11), R1                                  ; 1317
                  52          51                02   C7 00134        DIVL3   #2, R1, ADJUSTED_WIDTH
                                                07   11 00138        BRB     14$
                              51          06    AB   3C 0013A 13$:   MOVZWL  6(R11), R1                                  ; 1314
                              52                51   D0 0013E        MOVL    R1, ADJUSTED_WIDTH
                  50          56                03   78 00141 14$:   ASHL    #3, NO_TABS, R0                             ; 1316
                              50                09   C0 00145        ADDL2   #9, R0
                              52                50   D1 00148        CMPL    R0, ADJUSTED_WIDTH
                                                06   1E 0014B        BGEQU   15$
                  58          08                59   C3 0014D        SUBL3   NO_RETYPES, #8, NO_BS                       ; 1317
                                                05   11 00151        BRB     16$
                              58        03E8    8F   3C 00153 15$:   MOVZWL  #1000, NO_BS                                ; 1318
                  50          6E 000000D1       8F   C1 00158 16$:   ADDL3   #209, (SP), R0                              ; 1324
```

```
                                05            60  E8 00160        BLBS    (R0), 17$
                                58      03E8  8F  3C 00163        MOVZWL  #1000, NO_BS                                    ; 1325
                  52       10   AC            01  C3 00168 17$:   SUBL3   #1, DESIRED_LINE_NO, R2                        ; 1332
                                52            51  C4 0016C        MULL2   R1, R2
                                5A          6246  7E 00170        MOVAQ   (R2)[NO_TABS], INDEX
                                54       14   AC  D1 00174        CMPL    DESIRED_COL_NO, R4                             ; 1334
                                              39  14 00178        BGTR    21$
                  51            6E  000000D1  8F  C1 0017A        ADDL3   #209, (SP), R1                                 ; 1346
                                07            61  E9 00182        BLBC    (R1), 18$
                  50            54       14   AC  C3 00185        SUBL3   DESIRED_COL_NO, R4, S1_COST                    ; 1348
                                              05  11 0018A        BRB     19$
                                50      03E8  8F  3C 0018C 18$:   MOVZWL  #1000, S1_COST                                 ; 1350
                                51     01 A946    9E 00191 19$:   MOVAB   1(NO_RETYPES)[NO_TABS], S2_COST                ; 1355
                                55     02 A846    9E 00196        MOVAB   2(NO_BS)[NO_TABS], S3_COST                     ; 1360
                                53            01  D0 0019B        MOVL    #1, BEST_STRAT                                 ; 1364
                                50            51  D1 0019E        CMPL    S2_COST, LEAST_COST                            ; 1366
                                              06  18 001A1        BGEQ    20$
                                53            02  D0 001A3        MOVL    #2, BEST_STRAT                                 ; 1367
                                50            51  D0 001A6        MOVL    S2_COST, LEAST_COST
                                50            55  D1 001A9 20$:   CMPL    S3_COST, LEAST_COST                            ; 1369
                                              6D  18 001AC        BGEQ    26$
                                53            03  D0 001AE        MOVL    #3, BEST_STRAT                                 ; 1370
                                              65  11 001B1        BRB     25$
             04   AE       14   AC            54  C3 001B3 21$:   SUBL3   R4, DESIRED_COL_NO, S4_COST                    ; 1381
                  50            56            03  78 001B9        ASHL    #3, NO_TABS, R0                                ; 1383
                                              50  D6 001BD        INCL    R0
                                54            50  D1 001BF        CMPL    R0, R4
                                              30  15 001C2        BLEQ    22$
                  51            6E  000000FA  8F  C1 001C4        ADDL3   #250, (SP), R1
                  24            61            02  E1 001CC        BBC     #2, (R1), 22$
                  50            6E            0C  C1 001D0        ADDL3   #12, (SP), R0                                  ; 1384
                  1C            60            03  E0 001D4        BBS     #3, (R0), 22$
                           0C   AE       FF   A4  9E 001D8        MOVAB   -1(R4), COL_QUAD                               ; 1392
                                         10   AE  D4 001DD        CLRL    COL_QUAD+4                                     ; 1393
        55        51       0C   AE            08  7B 001E0        EDIV    #8, COL_QUAD, NEW_NO_TABS, NEW_NO_RETYPES      ; 1394
                                56            51  C2 001E6        SUBL2   NEW_NO_TABS, NO_TABS                           ; 1395
                  51            56            59  C1 001E9        ADDL3   NO_RETYPES, NO_TABS, S5_COST                   ; 1398
                                55     01 A846    9E 001ED        MOVAB   1(NO_BS)[NO_TABS], S6_COST                     ; 1402
                                              0A  11 001F2        BRB     23$                                            ; 1383
                                51      03E8  8F  3C 001F4 22$:   MOVZWL  #1000, S5_COST                                 ; 1406
                                55      03E8  8F  3C 001F9        MOVZWL  #1000, S6_COST                                 ; 1407
                                53            04  D0 001FE 23$:   MOVL    #4, BEST_STRAT                                 ; 1412
                                50       04   AE  D0 00201        MOVL    S4_COST, LEAST_COST
                                50            51  D1 00205        CMPL    S5_COST, LEAST_COST                            ; 1414
                                              06  18 00208        BGEQ    24$
                                53            05  D0 0020A        MOVL    #5, BEST_STRAT                                 ; 1415
                                50            51  D0 0020D        MOVL    S5_COST, LEAST_COST
                                50            55  D1 00210 24$:   CMPL    S6_COST, LEAST_COST                            ; 1417
                                              06  18 00213        BGEQ    26$
                                53            06  D0 00215        MOVL    #6, BEST_STRAT                                 ; 1418
                                50            55  D0 00218 25$:   MOVL    S6_COST, LEAST_COST
                                50            57  C0 0021B 26$:   ADDL2   TS_LEN, R0                                     ; 1421
                           08   AE            50  D1 0021E        CMPL    R0, SET_CUR_LEN
                                              3F  14 00222        BGTR    31$
                           08   AE     1C AE47    9E 00224        MOVAB   TRIAL_STRING[TS_LEN], 8(SP)                    ; 1431
                  05            01            53  CF 0022A        CASEL   BEST_STRAT, #1, #5                             ; 1427
     005B       0047          001B          000C     0022E 27$:   .WORD   28$-27$,-
```

```
                         00CD     008A     00236              29$-27$,-                                 :
                                                              33$-27$,-                                 :
                                                              34$-27$,-                                 :
                                                              37$-27$,-                                 :
                                                              43$-27$                                   :
              58         54     14  AC  C3 0023A 28$:  SUBL3  DESIRED_COL_NO, R4, NO_BS                 : 1430
   58         08         6E         00  2C 0023F       MOVC5  #0, (SP), #8, NO_BS, a8(SP)               : 1431
                                08  BE     00244                                                        :
                                  00CA  31 00246       BRW    45$                                       : 1432
                                    59  D5 00249 29$:  TSTL   NO_RETYPES                                : 1445
                                    18  13 0024B       BEQL   32$                                       :
                         54         01  D0 0024D       MOVL   #1, R4                                    : 1447
   59         00 00000000 9F        00  2C 00250       CMPC5  #0, a#^X00000000, #0, NO_RETYPES, a24(R11)- :
                                18 BB4A    00259              [INDEX]                                   :
                                    03  1A 0025C       BGTRU  30$                                       :
                         54         01  D9 0025E       SBWC   #1, R4                                    :
                                    54  D5 00261 30$:  TSTL   R4                                        : 1449
                                    6F  12 00263 31$:  BNEQ   39$                                       :
                    08   BE         0D  90 00265 32$:  MOVB   #13, a8(SP)                               : 1453
                                    57  D6 00269       INCL   TS_LEN                                    : 1454
   56         09         6E         00  2C 0026B       MOVC5  #0, (SP), #9, NO_TABS, TRIAL_STRING[TS_LEN] : 1455
                                1C AE47    00270                                                        :
                                    76  11 00273       BRB    41$                                       : 1456
                    08   BE         0D  90 00275 33$:  MOVB   #13, a8(SP)                               : 1463
                                    57  D6 00279       INCL   TS_LEN                                    : 1464
                         50     01  A6  9E 0027B       MOVAB  1(R6), R0                                 : 1465
   50         09         6E         00  2C 0027F       MOVC5  #0, (SP), #9, R0, TRIAL_STRING[TS_LEN]    :
                                1C AE47    00284                                                        :
                                    7D  11 00287       BRB    44$                                       : 1466
              59    14   AC         54  C3 00289 34$:  SUBL3  R4, DESIRED_COL_NO, NO_RETYPES            : 1481
                         5A    FF A442  9E 0028E       MOVAB  -1(R4)[R2], INDEX                         : 1482
                                    59  D5 00293       TSTL   NO_RETYPES                                : 1483
                                    18  13 00295       BEQL   36$                                       :
                         54         01  D0 00297       MOVL   #1, R4                                    : 1485
   59         00 00000000 9F        00  2D 0029A       CMPC5  #0, a#^X00000000, #0, NO_RETYPES, a24(R11)- :
                                18 BB4A    002A3              [INDEX]                                   :
                                    03  1A 002A6       BGTRU  35$                                       :
                         54         01  D9 002A8       SBWC   #1, R4                                    :
                                    54  D5 002AB 35$   TSTL   R4                                        : 1487
                                    25  12 002AD       BNEQ   39$                                       :
         08   BE    14 BB4A         59  28 002AF 36$:  MOVC3  NO_RETYPES, a20(R11)[INDEX], a8(SP)       : 1492
                                    3E  11 002B6       BRB    42$                                       : 1493
                                    59  D5 002B8 37$:  TSTL   NO_RETYPES                                : 1506
                                    28  13 002BA       BEQL   40$                                       :
                         54         01  D0 002BC       MOVL   #1, R4                                    : 1508
   59         00 00000000 9F        00  2D 002BF       CMPC5  #0, a#^X00000000, #0, NO_RETYPES, a24(R11)- :
                                18 BB4A    002C8              [INDEX]                                   :
                                    03  1A 002CB       BGTRU  38$                                       :
                         54         01  D9 002CD       SBWC   #1, R4                                    :
                                    54  D5 002D0 38$:  TSTL   R4                                        : 1510
                                    10  13 002D2       BEQL   40$                                       :
                                08  AC  DD 002D4 39$:  PUSHL  LINE_NO                                   : 1512
                         7E     10  AC  7D 002D7       MOVQ   DESIRED_LINE_NO, -(SP)                    :
                                0C  AE  DD 002DB       PUSHL  12(SP)                                    :
                   0000V CF         04  FB 002DE       CALLS  #4, SET_CURSOR                            :
                                        04 002E3       RET                                              :
   56         09         6E         00  2C 002E4 40$:  MOVC5  #0, (SP), #9, NO_TABS, a8(SP)             : 1514
```

```
                                     08    BE       002E9
                          57             56   C0 002EB 41$:      ADDL2   NO_TABS, TS_LEN                                 ; 1515
         1C AE47     14 BB4A             59   28 002EE           MOVC3   NO_RETYPES, @20(R11)[INDEX], TRIAL_STRING-      ; 1517
                                                                         [TS_LEN]
                          57             59   C0 002F6 42$:      ADDL2   NO_RETYPES, TS_LEN                              ; 1518
                                         1B   11 002F9           BRB     46$                                             ; 1427
                          50     01      A6   9E 002FB 43$:      MOVAB   1(R6), R0                                       ; 1522
   50          09         6E             00   2C 002FF           MOVC5   #0, (SP), #9, R0, @8(SP)                        ;
                                     08  BE      00304                                                                   ;
                          57     01 A647  9E 00306 44$:          MOVAB   1(NO_TABS)[TS_LEN], TS_LEN                      ; 1523
   58          08         6E             00   2C 0030B           MOVC5   #0, (SP), #8, NO_BS, TRIAL_STRING[TS_LEN]       ; 1524
                                  1C AE47        00310                                                                   ;
                          57             58   C0 00313 45$:      ADDL2   NO_BS, TS_LEN                                   ; 1525
                                         7E   7C 00316 46$:      CLRQ    -(SP)                                           ; 1539
                                 24      AE   9F 00318           PUSHAB  TRIAL_STRING                                    ;
                                         57   DD 0031B           PUSHL   TS_LEN                                          ;
                                 10      AE   DD 0031D           PUSHL   16(SP)                                          ;
                  00000000G  00          05   FB 00320           CALLS   #5, SMG$$PUT_SCREEN                             ;
                             03          50   E9 00327           BLBC    STATUS, 47$                                     ;
                             50          01   D0 0032A           MOVL    #1, R0                                          ; 1541
                                              04 0032D 47$:      RET                                                     ; 1543

; Routine Size:  814 bytes,    Routine Base:  _SMG$CODE + 0429
```

```
;  1257          1544  1 %SBTTL 'SET_CURSOR - Generate set-cursor sequence'
;  1258          1545  1 ROUTINE SET_CURSOR (
;  1259          1546  1                     P_PBCB,
;  1260          1547  1                     DESIRED_LINE_NO,
;  1261          1548  1                     DESIRED_COL_NO,
;  1262          1549  1                     CURRENT_ROW
;  1263          1550  1                    ) =
;  1264          1551  1 !++
;  1265          1552  1 ! FUNCTIONAL DESCRIPTION:
;  1266          1553  1 !
;  1267          1554  1 ! Routine SET_CURSOR constructs the general set cursor
;  1268          1555  1 ! sequence to position to .DESIRED_LINE_NO/.DESIRED_COL_NO and outputs
;  1269          1556  1 ! it to the screen.
;  1270          1557  1 !
;  1271          1558  1 ! CALLING SEQUENCE:
;  1272          1559  1 !
;  1273          1560  1 !       ret_status.wlc.v = SET_CURSOR ( P_PBCB.rab.r,
;  1274          1561  1 !                                       DESIRED_LINE_NO.rl.v,
;  1275          1562  1 !                                       DESIRED_COL_NO.rl.v,
;  1276          1563  1 !                                       CURRENT_ROW.rl.v)
;  1277          1564  1 !
;  1278          1565  1 ! FORMAL PARAMETERS:
;  1279          1566  1 !
;  1280          1567  1 !       P_PBCB.rab.r            Address of PBCB
;  1281          1568  1 !
;  1282          1569  1 !       DESIRED_LINE_NO.rl.v    Desired cursor line number position
;  1283          1570  1 !
;  1284          1571  1 !       DESIRED_COL_NO.rl.v     Desired cursor column number position
;  1285          1572  1 !
;  1286          1573  1 !       CURRENT_ROW.rl.v        Current row (0 means unknown)
;  1287          1574  1 !                               This matters if we are on a wide row.
;  1288          1575  1 !
;  1289          1576  1 ! IMPLICIT INPUTS:
;  1290          1577  1 !
;  1291          1578  1 !       NONE
;  1292          1579  1 !
;  1293          1580  1 ! IMPLICIT OUTPUTS:
;  1294          1581  1 !
;  1295          1582  1 !       NONE
;  1296          1583  1 !
;  1297          1584  1 ! COMPLETION STATUS:
;  1298          1585  1 !
;  1299          1586  1 !       SS$_NORMAL      Normal successful completion
;  1300          1587  1 !                       errors from SMG$$OUTPUT
;  1301          1588  1 !
;  1302          1589  1 ! SIDE EFFECTS:
;  1303          1590  1 !
;  1304          1591  1 !       NONE
;  1305          1592  1 !--
```

```
;  1307          1593  2 BEGIN
;  1308          1594  2
;  1309          1595  2 BIND
;  1310          1596  2
;  1311          1597  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  1312          1598  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  1313          1599  2         LCV             = .WCB[WCB_A_SCR_LINE_CHAR]     : VECTOR[,BYTE];
;  1314          1600  2
;  1315          1601  2 LOCAL
;  1316          1602  2
;  1317          1603  2         STATUS;         ! local status
```

```
;  1319     1604  2 !+
;  1320     1605  2 ! If we are currently on a double wide or high row (or if the
;  1321     1606  2 ! possibility exists) then because of bugs in the VT100 hardware,
;  1322     1607  2 ! we first position to column 1 of the desired line.
;  1323     1608  2 !-
;  1324     1609  2
;  1325     1610  3 IF (.CURRENT_ROW EQL 0 AND .LCV[0] NEQ 0)
;  1326     1611  2 OR .LCV[.CURRENT_ROW] NEQ 0
;  1327     1612  3   THEN  BEGIN           ! Move to beginning of desired line
;  1328     1613  3
;  1329     1614  3         $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.DESIRED_LINE_NO,1);
;  1330     1615  3
;  1331     1616  3         !+
;  1332     1617  3         ! Output the escape sequence.
;  1333     1618  3         !-
;  1334     1619  3
;  1335     1620  3         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  1336     1621  4           THEN  BEGIN
;  1337     1622  4                 STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  1338     1623  4                                    .PBCB[PBCB_A_CAP_BUFFER]);
;  1339     1624  4                 IF NOT .STATUS THEN RETURN .STATUS
;  1340     1625  3                 END;
;  1341     1626  3
;  1342     1627  2         END;            ! Move to beginning of desired line
;  1343     1628  2
;  1344     1629  2 !+
;  1345     1630  2 ! Create the appropriate escape sequence.
;  1346     1631  2 !-
;  1347     1632  2
;  1348     1633  2 $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.DESIRED_LINE_NO,.DESIRED_COL_NO);
;  1349     1634  2
;  1350     1635  2 !+
;  1351     1636  2 ! Output the escape sequence.
;  1352     1637  2 !-
;  1353     1638  2
;  1354     1639  2 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  1355     1640  3   THEN  BEGIN
;  1356     1641  3         STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  1357     1642  3                            .PBCB[PBCB_A_CAP_BUFFER]);
;  1358     1643  3         IF NOT .STATUS THEN RETURN .STATUS
;  1359     1644  2         END;
;  1360     1645  2
;  1361     1646  2 RETURN SS$_NORMAL
;  1362     1647  2
;  1363     1648  1 END;     ! Routine SET_CURSOR


                                007C 00000 SET_CURSOR:
                                           .WORD   Save R2,R3,R4,R5,R6                    ; 1545
                    56 00000000G   00  9E 00002    MOVAB   SMG$GET_TERM_DATA, R6          ;
                    55 00000000G   00  9E 00009    MOVAB   SMG$$OUTPUT, R5                ;
                    5E             10  C2 00010    SUBL2   #16, SP                        ;
                    52       04    AC  D0 00013    MOVL    P_PBCB, R2                     ; 1597
                    50       08    A2  D0 00017    MOVL    8(R2), R0                      ; 1598
```

```
                                10  AC  D5 0001B          TSTL    CURRENT_ROW                                     ; 1610
                                    05  12 0001E          BNEQ    1$
                                30  B0  95 00020          TSTB    @48(R0)
                                    0A  12 00023          BNEQ    2$
            50      30  A0      10  AC  C1 00025 1$:      ADDL3   CURRENT_ROW, 48(R0), R0                      ; 1611
                                    60  95 0002B          TSTB    (R0)
                                    56  13 0002D          BEQL    5$
                            00FC    C2  D5 0002F 2$:      TSTL    252(R2)                                     ; 1614
                                    09  12 00033          BNEQ    3$
                                53  0108  C2  9E 00035    MOVAB   264(R2), R3
                                    63  D4 0003A          CLRL    (R3)
                                    32  11 0003C          BRB     4$
                        04  AE          02  D0 0003E 3$:      MOVL    #2, INPUT_ARGS
                        08  AE      08  AC  D0 00042      MOVL    DESIRED_LINE_NO, INPUT_ARGS+4
                        0C  AE          01  D0 00047      MOVL    #1, INPUT_ARGS+8
                                04  AE  9F 0004B          PUSHAB  INPUT_ARGS
                            0104    C2  DD 0004E          PUSHL   260(R2)
                                53  0108  C2  9E 00052    MOVAB   264(R2), R3
                                    53  DD 00057          PUSHL   R3
                            0100    C2  9F 00059          PUSHAB  256(R2)
                        10  AE      023A  8F 3C 0005D     MOVZWL  #570, 16(SP)
                                10  AE  9F 00063          PUSHAB  16(SP)
                            00FC    C2  9F 00066          PUSHAB  252(R2)
                                66      06  FB 0006A      CALLS   #6, SMG$GET_TERM_DATA
                                6E      50  E9 0006D      BLBC    STATUS, 10$
                                        63  D5 00070 4$:      TSTL    (R3)                                    ; 1620
                                        11  13 00072      BEQL    5$
                                0104    C2  DD 00074      PUSHL   260(R2)                                     ; 1623
                                        63  DD 00078      PUSHL   (R3)                                        ; 1622
                                        52  DD 0007A      PUSHL   R2
                                65      03  FB 0007C      CALLS   #3, SMG$$OUTPUT
                                54      50  D0 0007F      MOVL    R0, STATUS
                                52      54  E9 00082      BLBC    STATUS, 8$                                  ; 1624
                                    00FC    C2  D5 00085 5$:      TSTL    252(R2)                             ; 1633
                                            09  12 00089      BNEQ    6$
                                53      0108  C2  9E 0008B    MOVAB   264(R2), R3
                                            63  D4 00090      CLRL    (R3)
                                            2E  11 00092      BRB     7$
                        04  AE              02  D0 00094 6$:      MOVL    #2, INPUT_ARGS
                        08  AE          08  AC  7D 00098      MOVQ    DESIRED_LINE_NO, INPUT_ARGS+4
                                    04  AE  9F 0009D          PUSHAB  INPUT_ARGS
                                0104    C2  DD 000A0          PUSHL   260(R2)
                                53      0108  C2  9E 000A4    MOVAB   264(R2), R3
                                            53  DD 000A9      PUSHL   R3
                                0100    C2  9F 000AB          PUSHAB  256(R2)
                        10  AE      023A  8F 3C 000AF     MOVZWL  #570, 16(SP)
                                10  AE  9F 000B5          PUSHAB  16(SP)
                            00FC    C2  9F 000B8          PUSHAB  252(R2)
                                66      06  FB 000BC      CALLS   #6, SMG$GET_TERM_DATA
                                1C      50  E9 000BF      BLBC    STATUS, 10$
                                        63  D5 000C2 7$:      TSTL    (R3)                                    ; 1639
                                        15  13 000C4      BEQL    9$
                                0104    C2  DD 000C6      PUSHL   260(R2)                                     ; 1642
                                        63  DD 000CA      PUSHL   (R3)                                        ; 1641
                                        52  DD 000CC      PUSHL   R2
                                65      03  FB 000CE      CALLS   #3, SMG$$OUTPUT
                                54      50  D0 000D1      MOVL    R0, STATUS
```

```
                     04          54  E8 000D4          BLBS    STATUS, 9$                      ; 1643
                     50          54  D0 000D7 8$:      MOVL    STATUS, R0                      ;
                                     04 000DA          RET                                     ;
                     50          01  D0 000DB 9$:      MOVL    #1, R0                          ; 1646
                                     04 000DE 10$:     RET                                     ; 1648

; Routine Size:  223 bytes,    Routine Base:  _SMG$CODE + 0757
```

```
;  1365     1649  1 %SBTTL 'ERASE_LINE - Erase to end-of-line'
;  1366     1650  1 ROUTINE ERASE_LINE(P_PBCB) =
;  1367     1651  1
;  1368     1652  1 !++
;  1369     1653  1 ! FUNCTIONAL DESCRIPTION:
;  1370     1654  1 !
;  1371     1655  1 ! Outputs an erase-to-end-of-line sequence to the screen.
;  1372     1656  1 !
;  1373     1657  1 ! CALLING SEQUENCE:
;  1374     1658  1 !
;  1375     1659  1 !       ret_status.wlc.v = ERASE_LINE ( P_PBCB.rab.r)
;  1376     1660  1 !
;  1377     1661  1 ! FORMAL PARAMETERS:
;  1378     1662  1 !
;  1379     1663  1 !       P_PBCB.rab.r            Address of PBCB
;  1380     1664  1 !
;  1381     1665  1 ! IMPLICIT INPUTS:
;  1382     1666  1 !
;  1383     1667  1 !       NONE
;  1384     1668  1 !
;  1385     1669  1 ! IMPLICIT OUTPUTS:
;  1386     1670  1 !
;  1387     1671  1 !       NONE
;  1388     1672  1 !
;  1389     1673  1 ! COMPLETION STATUS:
;  1390     1674  1 !
;  1391     1675  1 !       SS$_NORMAL      Normal successful completion
;  1392     1676  1 !                       errors from SMG$$OUTPUT
;  1393     1677  1 !
;  1394     1678  1 ! SIDE EFFECTS:
;  1395     1679  1 !
;  1396     1680  1 !       NONE
;  1397     1681  1 !--
```

```
;  1399          1682  2 BEGIN
;  1400          1683  2
;  1401          1684  2 BIND
;  1402          1685  2
;  1403          1686  2         PBCB            = .P_PBCB               : BLOCK[,BYTE];
;  1404          1687  2
;  1405          1688  2 LOCAL
;  1406          1689  2
;  1407          1690  2         STATUS;         ! local status
```

```
;  1409          1691  2 !+
;  1410          1692  2 ! Create the appropriate escape sequence.
;  1411          1693  2 !-
;  1412          1694  2
;  1413          1695  2 $SMG$GET_TERM_DATA(ERASE_TO_END_LINE);
;  1414          1696  2
;  1415          1697  2 !+
;  1416          1698  2 ! Output the escape sequence.
;  1417          1699  2 !-
;  1418          1700  2
;  1419          1701  2 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  1420          1702  3   THEN  BEGIN
;  1421          1703  3         STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  1422          1704  3                            .PBCB[PBCB_A_CAP_BUFFER]);
;  1423          1705  3         IF NOT .STATUS THEN RETURN .STATUS;
;  1424          1706  2         END;
;  1425          1707  2
;  1426          1708  2 RETURN SS$_NORMAL
;  1427          1709  2
;  1428          1710  1 END;     ! Routine ERASE_LINE


                                000C 00000 ERASE_LINE:
                                           .WORD   Save R2,R3                      ; 1650
                             5E      10    C2 00002           SUBL2   #16, SP
                             52      04    AC D0 00005        MOVL    P_PBCB, R2   ; 1686
                                   00FC    C2 D5 00009        TSTL    252(R2)      ; 1695
                                           09 12 0000D        BNEQ    1$
                             53    0108    C2 9E 0000F        MOVAB   264(R2), R3
                                           63 D4 00014        CLRL    (R3)
                                           2C 11 00016        BRB     2$
                                     04    AE D4 00018 1$:    CLRL    INPUT_ARGS
                                     04    AE 9F 0001B        PUSHAB  INPUT_ARGS
                                   0104    C2 DD 0001E        PUSHL   260(R2)
                             53    0108    C2 9E 00022        MOVAB   264(R2), R3
                                           53 DD 00027        PUSHL   R3
                                   0100    C2 9F 00029        PUSHAB  256(R2)
                                     10 AE 01D9 8F 3C 0002D   MOVZWL  #473, 16(SP)
                                     10    AE 9F 00033        PUSHAB  16(SP)
                                   00FC    C2 9F 00036        PUSHAB  252(R2)
                        00000000G    00      0C FB 0003A     CALLS   #6, SMG$GET_TERM_DATA
                                     19      50 E9 00041     BLBC    STATUS, 4$
                                             63 D5 00044 2$: TSTL    (R3)         ; 1701
                                             12 13 00046     BEQL    3$
                                   0104      C2 DD 00048     PUSHL   260(R2)      ; 1704
                                             63 DD 0004C     PUSHL   (R3)         ; 1703
                                             52 DD 0004E     PUSHL   R2
                        00000000G    00      03 FB 00050     CALLS   #3, SMG$$OUTPUT
                                     03      50 E9 00057     BLBC    STATUS, 4$   ; 1705
                                     50      01 D0 0005A 3$: MOVL    #1, R0       ; 1708
                                             04 0005D 4$:    RET                  ; 1710

; Routine Size:  94 bytes,    Routine Base:  _SMG$CODE + 0836
```

```
; 1430          1711  1 END
; 1431          1712  0 ELUDOM
```

```
;                                   PSECT SUMMARY

;       Name                    Bytes                   Attributes

;  _SMG$CODE                      2196  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                                 Library Statistics

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

;  _$255$DUA18:[SYSLIB]STARLET.L32;1         9776         10         0         581       00:01.0
;  _$255$DUA18:[SMGRTL.OBJ]RTLLIB.L32;1        36          0         0           8       00:00.1
;  _$255$DUA18:[SMGRTL.OBJ]SMGLIB.L32;1       469         48        10          38       00:00.4
```

```
;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:SMGMIN/OBJ=OBJ$:SMGMIN MSRC$:SMGMIN/UPDATE=(BUG$:SMGMIN)

; Size:          2196 code + 0 data bytes
; Run Time:        00:49.7
; Elapsed Time:    01:04.9
; Lines/CPU Min:    2765
; Lexemes/CPU-Min: 15057
; Memory Used:  306 pages
; Compilation Complete
```

0446 AH-EF71A-SE
VAX/VMS V4.1 SRC LST MCRF UPD

SMGMIN
LIS

SMGDISOUT
LIS

0447 AH-EF71A-SE
VAX/VMS V4.1 SRC LST MCRF UPD

SYSLOA

SMGMINUPD
LIS

CSP
MAP

SMGPUTTEX
LIS

SYSLOAWS1
MAP