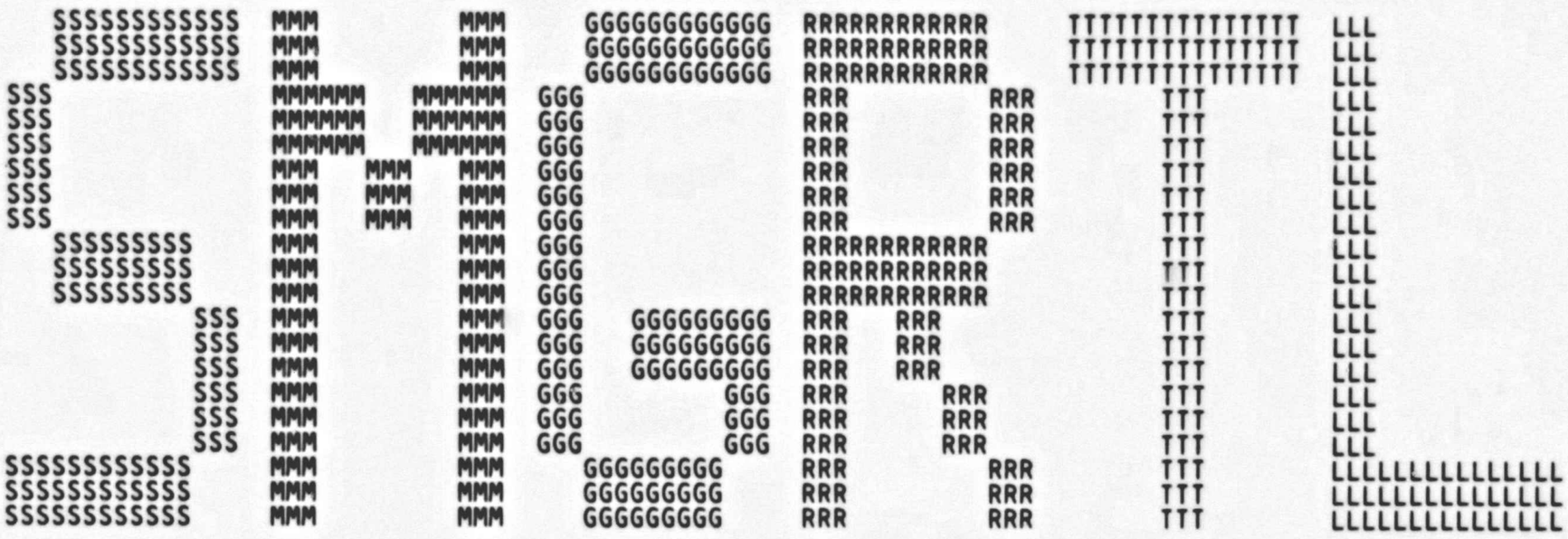

_$2

Val

001
001
001
001
001
001
001
001
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF

FILEID**SMGMIN


```
  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
```

SMG$
1-01

```
;  1          0001  0 %TITLE 'Minimal update calculation'
;  2          0002  0 MODULE SMG$MIN (
;  3          0003  0                 IDENT = '1-016' ! File: SMGMIN.B32      Edit:STAN1016
;  4          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 RKR.
;    45     0044  1 !
;    46     0045  1 ! MODIFIED BY:
;    47     0046  1 !
;    48     0047  1 ! 1-016 - STAN  6-Jun-1984. Change error messages in MSG$SET_PHYSICAL_CURSOR.
;    49     0048  1 ! 1-001 - STAN, 1-May-1983. Initial version, mimicked SCRMIN.B32.
;    50     0049  1 !--
```

```
;    52          0050  1 %SBTTL 'Declarations'
;    53          0051  1 !
;    54          0052  1 ! SWITCHES:
;    55          0053  1 !
;    56          0054  1 !       NONE
;    57          0055  1 !
;    58          0056  1 ! LINKAGES:
;    59          0057  1 !
;    60          0058  1 !       NONE
;    61          0059  1 !
;    62          0060  1 ! TABLE OF CONTENTS:
;    63          0061  1 !
;    64          0062  1
;    65          0063  1 FORWARD ROUTINE
;    66          0064  1
;    67          0065  1         SMG$SET_PHYSICAL_CURSOR,        ! Move physical cursor on screen
;    68          0066  1         SMG$$OUTPUT_MINIMAL_UPDATE,     ! Output minumal update sequence
;    69          0067  1         SMG$$FIND_MIN_CURSOR_POS,       ! Output minimum cursor sequence
;    70          0068  1         SMG$$UPDATE_PHYSICAL_CURSOR,    ! Update physical cursor position
;    71          0069  1         ERASE_LINE,                     ! Erase to end-of-line
;    72          0070  1         SET_CURSOR;                     ! Generate general set-cursor
;    73          0071  1                                         ! positioning sequence.
;    74          0072  1
;    75          0073  1 !
;    76          0074  1 ! INCLUDE FILES
;    77          0075  1 !
;    78          0076  1
;    79          0077  1 REQUIRE 'RTLIN:SMGPROLOG';              ! defines psects, macros, structures,
;    80          0155  1                                         ! & terminal symbols
;    81          0156  1 REQUIRE 'RTLIN:STRLNK.REQ';             ! JSB linkages
;    82          0341  1
;    83          0342  1 !
;    84          0343  1 ! EXTERNAL REFERENCES
;    85          0344  1 !
;    86          0345  1
;    87          0346  1 EXTERNAL ROUTINE
;    88          0347  1
;    89          0348  1         SMG$$OUTPUT;
;    90          0349  1
;    91          0350  1 !+
;    92          0351  1 ! $OUTPUT_STRING
;    93          0352  1 ! --------------
;    94          0353  1 !-
;    95          0354  1
;    96          0355  1 MACRO
;    97          0356  1
;    98        M 0357  1     $OUTPUT_STRING(LEN,ADDR,ATTR) =
;    99        M 0358  1
;   100        M 0359  1         BEGIN
;   101        M 0360  1         EXTERNAL ROUTINE SMG$$PUT_SCREEN;
;   102        M 0361  1         LOCAL STATUS;
;   103        M 0362  1         STATUS=SMG$$PUT_SCREEN(PBCB,LEN,ADDR,0,0,ATTR);
;   104        M 0363  1         IF NOT .STATUS THEN RETURN .STATUS
;   105        M 0364  1         END
;   106          0365  1 %;
;   107          0366  1
;   108          0367  1 !+
```

```
;  109          0368  1 !+
;  110          0369  1 ! $L
;  111          0370  1 ! --
;  112          0371  1 ! Macro $L linearizes a two dimensional subcript formed by a 1-based
;  113          0372  1 ! row number and a 1-based column number, into a single 0-based
;  114          0373  1 ! subscript.
;  115          0374  1 !-
;  116          0375  1
;  117          0376  1 MACRO
;  118        M 0377  1         $L (ROW_NUMBER, COLUMN_NUMBER) =
;  119          0378  1         (ROW_NUMBER-1)*.NUM_COLS + COLUMN_NUMBER -1 %;
;  120          0379  1
;  121          0380  1 !+
;  122          0381  1 ! $MAKE_ROW_COL
;  123          0382  1 ! -------------
;  124          0383  1 ! Macro $MAKE_ROW_COL takes as an input a 0-based linear index into
;  125          0384  1 ! and array and converts it into a 1-based row and 1-based column
;  126          0385  1 ! form.  INDEX needs to be re-expressed as a quadword for use in the
;  127          0386  1 ! EDIV instruction.
;  128          0387  1 !-
;  129          0388  1
;  130          0389  1 MACRO
;  131          0390  1
;  132        M 0391  1     $MAKE_ROW_COL ( INDEX, ROW_NUMBER, COLUMN_NUMBER) =
;  133        M 0392  1     BEGIN       ! MAKE_ROW_COL
;  134        M 0393  1     BUILTIN
;  135        M 0394  1         EDIV;
;  136        M 0395  1     LOCAL
;  137        M 0396  1         WIDTH,
;  138        M 0397  1         LOCAL_INDEX : VECTOR [2, LONG];
;  139        M 0398  1     LOCAL_INDEX [1] = 0; ! Second longword is always 0
;  140        M 0399  1     LOCAL_INDEX [0] = .INDEX;
;  141        M 0400  1     WIDTH=.NUM_COLS;     ! Store width in longword
;  142        M 0401  1
;  143        M 0402  1     EDIV ( WIDTH, LOCAL_INDEX, ROW_NUMBER, COLUMN_NUMBER);
;  144        M 0403  1     ROW_NUMBER = .ROW_NUMBER + 1;
;  145        M 0404  1     COLUMN_NUMBER = .COLUMN_NUMBER + 1;
;  146        M 0405  1     END;          ! MAKE_ROW_COL
;  147          0406  1     %;
```

```
;  149          0407  1 %SBTTL 'SMG$$OUTPUT_MINIMAL_UPDATE - Calculate minimum update sequence'
;  150          0408  1 GLOBAL ROUTINE SMG$$OUTPUT_MINIMAL_UPDATE (P_PBCB) =
;  151          0409  1 !++
;  152          0410  1 ! FUNCTIONAL DESCRIPTION:
;  153          0411  1 !
;  154          0412  1 !       This routine compares CURR_TEXT and CURR_ATTR (which reflect
;  155          0413  1 !       what is currently on the screen), with NEW_TEXT and NEW_ATTR
;  156          0414  1 !       (which reflect what should be on the screen) and calculates a
;  157          0415  1 !       sequences of characters which when output to the screen changes
;  158          0416  1 !       the current screen contents to reflect the new (desired) screen
;  159          0417  1 !       contents. These characters are actually output to the screen.
;  160          0418  1 !
;  161          0419  1 ! CALLING SEQUENCE:
;  162          0420  1 !
;  163          0421  1 !       ret_status.wlc.v = SMG$$MINIMUM_UPDATE ( P_PBCB.rab.r)
;  164          0422  1 !
;  165          0423  1 ! FORMAL PARAMETERS:
;  166          0424  1 !
;  167          0425  1 !       P_PBCB,rab.r            Address of pasteboard control block
;  168          0426  1 !
;  169          0427  1 ! IMPLICIT INPUTS:
;  170          0428  1 !
;  171          0429  1 !       Contents of PBCB and WCB
;  172          0430  1 !
;  173          0431  1 ! IMPLICIT OUTPUTS:
;  174          0432  1 !
;  175          0433  1 !       Internal buffers change.
;  176          0434  1 !
;  177          0435  1 ! COMPLETION STATUS:
;  178          0436  1 !
;  179          0437  1 !       SS$_NORMAL      Normal successful completion
;  180          0438  1 !
;  181          0439  1 ! SIDE EFFECTS:
;  182          0440  1 !
;  183          0441  1 !       NONE
;  184          0442  1 !--
```

```
;  186     0443  2 BEGIN
;  187     0444  2
;  188     0445  2 BUILTIN
;  189     0446  2
;  190     0447  2         CMPC3;
;  191     0448  2
;  192     0449  2 BIND
;  193     0450  2
;  194     0451  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  195     0452  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  196     0453  2         NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  197     0454  2         NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  198     0455  2         CUR_TEXT        = .WCB[WCB_A_SCR_TEXT_BUF]      : VECTOR[,BYTE],
;  199     0456  2         CUR_ATTR        = .WCB[WCB_A_SCR_ATTR_BUF]      : VECTOR[,BYTE],
;  200     0457  2         NEW_TEXT        = .WCB[WCB_A_TEXT_BUF]          : VECTOR[,BYTE],
;  201     0458  2         NEW_ATTR        = .WCB[WCB_A_ATTR_BUF]          : VECTOR[,BYTE],
;  202     0459  2         NEW_LCV         = .WCB[WCB_A_LINE_CHAR]         : VECTOR[,BYTE],
;  203     0460  2         CUR_LCV         = .WCB[WCB_A_SCR_LINE_CHAR]     : VECTOR[,BYTE],
;  204     0461  2         OLD_CURSOR_ROW  =  WCB[WCB_W_OLD_CUR_ROW]       : WORD,
;  205     0462  2         OLD_CURSOR_COL  =  WCB[WCB_W_OLD_CUR_COL]       : WORD,
;  206     0463  2         NEW_CURSOR_ROW  =  WCB[WCB_W_CURR_CUR_ROW]      : WORD,
;  207     0464  2         NEW_CURSOR_COL  =  WCB[WCB_W_CURR_CUR_COL]      : WORD,
;  208     0465  2         SIZE            =  WCB[WCB_L_BUFSIZE],          ! Size of buffers
;  209     0466  2         FIRST_ROW       =  PBCB[PBCB_W_FIRST_CHANGED_ROW] : WORD,
;  210     0467  2         LAST_ROW        =  PBCB[PBCB_W_LAST_CHANGED_ROW]  : WORD,
;  211     0468  2         FIRST_COL       =  PBCB[PBCB_W_FIRST_CHANGED_COL] : WORD,
;  212     0469  2         LAST_COL        =  PBCB[PBCB_W_LAST_CHANGED_COL]  : WORD,
;  213     0470  2         TERM_TYPE       =  PBCB[PBCB_B_DEVTYPE]         : BYTE;
;  214     0471  2
;  215     0472  2 LOCAL
;  216     0473  2
;  217     0474  2         STATUS,             ! Status to return to caller
;  218     0475  2         INDEX,              ! Working index into the buffers
;  219     0476  2         ROW,                ! Working row number
;  220     0477  2         COL,                ! Working column number
;  221     0478  2         LEN,                ! local length
;  222     0479  2         ADJUSTED_COL,       ! Wide line adjusted column number
;  223     0480  2         CUR_TEXT_PTR : REF VECTOR [,BYTE],  ! Current pointer into
;  224     0481  2                                             ! current text buffer
;  225     0482  2         CUR_ATTR_PTR : REF VECTOR [,BYTE],  ! Current pointer into
;  226     0483  2                                             ! current attribute buffer
;  227     0484  2         NEW_TEXT_PTR : REF VECTOR [,BYTE],  ! Current pointer into new
;  228     0485  2                                             ! text buffer
;  229     0486  2         NEW_ATTR_PTR : REF VECTOR [,BYTE],  ! Current pointer into new
;  230     0487  2                                             ! attribute buffer
;  231     0488  2         END_ROW_INDEX,      ! Index to last character in current row
;  232     0489  2         RENDITION,          ! local rendition
;  233     0490  2         FINAL_INDEX,        ! local index representing end of a changed sequence
;  234     0491  2         CURSOR_ROW,         ! Current cursor row
;  235     0492  2         CURSOR_COL,         ! Current cursor column
;  236     0493  2
;  237     0494  2         NEW_CHARS_LEFT,
;  238     0495  2         CHARS_LEFT;         ! Number of characters left to be inspected.
;  239     0496  2                             ! Starts out equal to number of characters
;  240     0497  2                             ! in the four buffers.
```

```
;  242     0498  2 !+
;  243     0499  2 ! If CTRL/O was typed previously, some QIO has returned with
;  244     0500  2 ! that success status and our CTRL/O bit is set.  We don't
;  245     0501  2 ! really know what the screen looks like anymore, so we
;  246     0502  2 ! clear out the screen buffer.
;  247     0503  2 !-
;  248     0504  2
;  249     0505  2 IF .PBCB[PBCB_V_CONTROLO]
;  250     0506  3   THEN  BEGIN   ! Clear screen buffer
;  251     0507  3               CH$FILL(0,,SIZE,CUR_TEXT);
;  252     0508  3               FIRST_ROW=1;
;  253     0509  3               FIRST_COL=1;
;  254     0510  3               LAST_ROW =.NUM_ROWS;
;  255     0511  3               LAST_COL =.NUM_COLS;
;  256     0512  3               PBCB[PBCB_V_CONTROLO]=0
;  257     0513  2               END;    ! Clear screen buffer
;  258     0514  2
;  259     0515  2 !+
;  260     0516  2 ! Initialize our working pointers into the buffers.
;  261     0517  2 ! For now: we invalidate the initial cursor position
;  262     0518  2 ! to force the first update to use full cursor addressing.
;  263     0519  2 !-
;  264     0520  2
;  265     0521  2 !CURSOR_ROW = .OLD_CURSOR_ROW;
;  266     0522  2 !CURSOR_COL = .OLD_CURSOR_COL;
;  267     0523  2
;  268     0524  2 CURSOR_ROW=0;
;  269     0525  2 CURSOR_COL=0;
;  270     0526  2
;  271     0527  2 INCR ROW FROM .FIRST_ROW TO .LAST_ROW DO
;  272     0528  3     BEGIN       ! Scan row .ROW
;  273     0529  3     LOCAL PTEXT,PATTR;
;  274     0530  3     LOCAL BLANK_COL;
;  275     0531  3     LOCAL PRE_PTR_IN_ROW;       ! Pointer position just before first character
;  276     0532  3                                 ! in this row
;  277     0533  3     CUR_TEXT_PTR = CUR_TEXT+(.ROW-1)*.NUM_COLS;
;  278     0534  3     CUR_ATTR_PTR = CUR_ATTR+(.ROW-1)*.NUM_COLS;
;  279     0535  3     NEW_TEXT_PTR = NEW_TEXT+(.ROW-1)*.NUM_COLS;
;  280     0536  3     NEW_ATTR_PTR = NEW_ATTR+(.ROW-1)*.NUM_COLS;
;  281     0537  3
;  282     0538  3     IF .NEW_LCV[.ROW] EQL 0
;  283     0539  3     THEN
;  284     0540  3         CHARS_LEFT=.NUM_COLS
;  285     0541  3     ELSE
;  286     0542  3         CHARS_LEFT=.NUM_COLS/2;
;  287     0543  3 !     CHARS_LEFT=.NUM_COLS;
;  288     0544  3     PRE_PTR_IN_ROW=.CUR_TEXT_PTR-1;
;  289     0545  3
;  290     0546  3     !+
;  291     0547  3     ! See if the characteristics of this line must change.
;  292     0548  3     !-
;  293     0549  3
;  294     0550  3     IF .CUR_LCV[.ROW] NEQ .NEW_LCV[.ROW]
;  295     0551  3     THEN
;  296     0552  4         BEGIN   ! Change line characteristics
;  297     0553  4
;  298     0554  4         LOCAL   BUFFER  : VECTOR[SMG$K_LONGEST_SEQUENCE,BYTE],
```

```
;  299     0555  4                    BUFLEN;
;  300     0556  4
;  301     0557  4            EXTERNAL ROUTINE
;  302     0558  4
;  303     0559  4                    SMG$$OUTPUT;
;  304     0560  4
;  305     0561  4            !+
;  306     0562  4            ! Move to the desired row.
;  307     0563  4            !-
;  308     0564  4
;  309     0565  4            SMG$$FIND_MIN_CURSOR_POS (  PBCB,          ! Pasteboard Control block
;  310     0566  4                                       .CURSOR_ROW,    ! Current row
;  311     0567  4                                       .CURSOR_COL,    ! Current column
;  312     0568  4                                       .ROW,           ! Desired row
;  313     0569  4                                       1);             ! Desired column
;  314     0570  4
;  315     0571  4            !+
;  316     0572  4            ! Update our record of where we are on screen.
;  317     0573  4            !-
;  318     0574  4
;  319     0575  4            CURSOR_ROW = .ROW ;
;  320     0576  4            CURSOR_COL = 1 ;
;  321     0577  4
;  322     0578  4            BUFLEN=0;
;  323     0579  4
;  324     0580  4            !+
;  325     0581  4            ! Get escape sequence to change the line characteristics.
;  326     0582  4            !-
;  327     0583  4
;  328     0584  4            SELECTONE .NEW_LCV[.ROW] OF
;  329     0585  4                    SET
;  330     0586  4                    [LINE_K_WIDE]:          $SMG$GET_TERM_DATA(DOUBLE_WIDE);
;  331     0587  4                    [LINE_K_UPPER_HIGH]:    $SMG$GET_TERM_DATA(DOUBLE_HIGH_TOP);
;  332     0588  4                    [LINE_K_LOWER_HIGH]:    $SMG$GET_TERM_DATA(DOUBLE_HIGH_BOTTOM);
;  333     0589  5                    [LINE_K_NORMA[]:        $SMG$GET_TERM_DATA(SINGLE_HIGH)
;  334     0590  4                    TES;
;  335     0591  4
;  336     0592  4            !+
;  337     0593  4            ! Output it.
;  338     0594  4            !-
;  339     0595  4
;  340     0596  4            IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  341     0597  5              THEN  BEGIN
;  342     0598  5                    STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  343     0599  5                                       .PBCB[PBCB_A_CAP_BUFFER]);
;  344     0600  5                    IF NOT .STATUS THEN RETURN .STATUS
;  345     0601  5                    END
;  346     0602  5
;  347     0603  3            END;    ! Change line characteristics
;  348     0604  3
;  349     0605  3        !+
;  350     0606  3        ! Scan backwards looking for the largest sequence of trailing spaces.
;  351     0607  3        ! Set BLANK_COL to the column number of the start of such a suffix.
;  352     0608  3        !-
;  353     0609  3
;  354     0610  3        BLANK_COL=.NUM_COLS+1;
;  355     0611  3        PTEXT=NEW_TEXT[.ROW*.NUM_COLS;
```

```
;  356     0612  3     PATTR=NEW_ATTR+.ROW*.NUM_COLS;
;  357     0613  3     DECR C FROM .NUM_COLS TO 1 DO
;  358     0614  4         BEGIN
;  359     0615  4         PTEXT=.PTEXT-1;
;  360     0616  4         PATTR=.PATTR-1;
;  361     0617  5                 BEGIN
;  362     0618  5                 BIND    TEXT_CHAR=.PTEXT : BYTE,
;  363     0619  5                         ATTR_CHAR=.PATTR : BYTE;
;  364     0620  5                 IF .TEXT_CHAR EQL %C' ' AND .ATTR_CHAR EQL 0
;  365     0621  5                   THEN  BLANK_COL=.C
;  366     0622  5                   ELSE  EXITLOOP
;  367     0623  4                 END;
;  368     0624  3         END;
;  369     0625  3
;  370     0626  3     WHILE .CHARS_LEFT NEQ 0 DO
;  371     0627  4         BEGIN   ! scan
;  372     0628  4         IF .CUR_TEXT_PTR[0] EQL .NEW_TEXT_PTR[0] AND
;  373     0629  4            .CUR_ATTR_PTR[0] EQL .NEW_ATTR_PTR[0]
;  374     0630  5           THEN  BEGIN   ! Characters agree
;  375     0631  5                 CUR_TEXT_PTR=.CUR_TEXT_PTR+1;
;  376     0632  5                 CUR_ATTR_PTR=.CUR_ATTR_PTR+1;
;  377     0633  5                 NEW_TEXT_PTR=.NEW_TEXT_PTR+1;
;  378     0634  5                 NEW_ATTR_PTR=.NEW_ATTR_PTR+1;
;  379     0635  5                 CHARS_LEFT=.CHARS_LEFT-1
;  380     0636  5                 END     ! Characters agree
;  381     0637  5           ELSE  BEGIN   ! Characters disagree
;  382     0638  5
;  383     0639  5                 INDEX=.CUR_TEXT_PTR-CUR_TEXT;
;  384     0640  5
;  385     0641  5 !               !+
;  386     0642  5 !               ! Re-express address as a row and column number
;  387     0643  5 !               !-
;  388     0644  5 !
;  389     0645  5 !               $MAKE_ROW_COL(INDEX,ROW,COL);
;  390     0646  5
;  391     0647  5                 COL=.CUR_TEXT_PTR-.PRE_PTR_IN_ROW;
;  392     0648  5
;  393     0649  5                 !+
;  394     0650  5                 ! At this point, the cursor is positioned at
;  395     0651  5                 ! .CURSOR_ROW, .CURSOR_COL.  The first character that
;  396     0652  5                 ! needs to be rewritten is at .ROW, .COL.
;  397     0653  5                 ! Determine a minimal update sequence to get us from
;  398     0654  5                 ! where cursor is to where it needs to be to do rewrite.
;  399     0655  5                 !-
;  400     0656  5
;  401     0657  5                 !+
;  402     0658  5                 ! Set the column to "unknown" if we are past the end of
;  403     0659  5                 ! the terminal width.  We cannot assume that the cursor
;  404     0660  5                 ! has become stuck in the last column, because the
;  405     0661  5                 ! user may have done a SET TERMINAL/WIDTH=n command
;  406     0662  5                 ! to shorten his logical terminal width.
;  407     0663  5                 !-
;  408     0664  5
;  409     0665  5                 IF .CURSOR_COL GTRU .NUM_COLS
;  410     0666  5                   THEN  CURSOR_COL=0;
;  411     0667  5
;  412     0668  5                 SMG$$FIND_MIN_CURSOR_POS (  PBCB,       ! Pasteboard Control block
```

```
;  413     0669  5                                        .CURSOR_ROW, ! Current row
;  414     0670  5                                        .CURSOR_COL, ! Current column
;  415     0671  5                                        .ROW,        ! Desired row
;  416     0672  5                                        .COL);       ! Desired column
;  417     0673  5
;  418     0674  5                    !+
;  419     0675  5                    ! Update our record of where we are on screen after
;  420     0676  5                    ! we output as much of the string as is currently in
;  421     0677  5                    ! our buffer.
;  422     0678  5                    !-
;  423     0679  5
;  424     0680  5                    CURSOR_ROW = .ROW ;
;  425     0681  5                    CURSOR_COL = .COL ;
;  426     0682  5
;  427     0683  5                    !+
;  428     0684  5                    ! Now that we are positioned at first difference,
;  429     0685  5                    ! figure out what needs to be written.
;  430     0686  5                    !-
;  431     0687  5
;  432     0688  5                    !+
;  433     0689  5                    ! If we are at or past the blank pointer, then
;  434     0690  5                    ! just blank the remainder of the line and exit.
;  435     0691  5                    !-
;  436     0692  5
;  437     0693  5                    IF .CURSOR_COL GEQU .BLANK_COL
;  438     0694  6                      THEN  BEGIN   ! erase rest of line
;  439     0695  6                            LOCAL STATUS;
;  440     0696  6                            STATUS=ERASE_LINE(PBCB);
;  441     0697  6                            IF NOT .STATUS THEN RETURN .STATUS;
;  442     0698  6                            EXITLOOP
;  443     0699  5                            END;    ! erase rest of line
;  444     0700  5
;  445     0701  5                    !+
;  446     0702  5                    ! Note that our linear position within the buffer
;  447     0703  5                    ! is given by the index INDEX.
;  448     0704  5                    ! We now calculate the linear position of the last
;  449     0705  5                    ! character on this row, storing the resulting index
;  450     0706  5                    ! in END_ROW_INDEX.
;  451     0707  5                    !-
;  452     0708  5
;  453     0709  5                    END_ROW_INDEX=$L(.ROW,.NUM_COLS);
;  454     0710  5
;  455     0711  5                    !+
;  456     0712  5                    ! We now must search between INDEX and END_ROW_INDEX
;  457     0713  5                    ! for the longest sequence (all of the same rendition)
;  458     0714  5                    ! of changed characters.
;  459     0715  5                    !-
;  460     0716  5
;  461     0717  5                    !+
;  462     0718  5                    ! Step 1: find the longest sequence of characters
;  463     0719  5                    ! that are all of the same rendition.
;  464     0720  5                    ! Put our currently desired attributes in RENDITION.
;  465     0721  5                    !-
;  466     0722  5
;  467     0723  5                    RENDITION = .NEW_ATTR[.INDEX];
;  468     0724  5                    FINAL_INDEX = .END_ROW_INDEX+1;
;  469     0725  5
```

```
;  470     0726  5                    !+
;  471     0727  5                    ! Set up FINAL_INDEX to be the first index past
;  472     0728  5                    ! the longest such difference sequence.
;  473     0729  5                    !-
;  474     0730  5
;  475     0731  5                    INCR I FROM .INDEX+1 TO .END_ROW_INDEX DO
;  476     0732  6                            BEGIN   ! scan for end of change
;  477     0733  7                            IF (.NEW_TEXT[.I] EQL .CUR_TEXT[.I] AND
;  478     0734  7                                .NEW_ATTR[.I] EQL .CUR_ATTR[.I])
;  479     0735  6                            OR .NEW_ATTR[.I] NEQ .RENDITION
;  480     0736  7                              THEN  BEGIN   ! end-of-change
;  481     0737  7                                    FINAL_INDEX=.I;
;  482     0738  7                                    EXITLOOP
;  483     0739  6                                    END;    ! end-of-change
;  484     0740  5                            END;    ! scan for end of change
;  485     0741  5
;  486     0742  5                    !+
;  487     0743  5                    ! We now must update the screen from .INDEX to .FINAL_INDEX-1
;  488     0744  5                    ! positions using the attributes stored in RENDITION.
;  489     0745  5                    ! The final SPACE_COUNT positions are to be erased.
;  490     0746  5                    !-
;  491     0747  5
;  492     0748  5                    LEN=.FINAL_INDEX-.INDEX;
;  493     0749  5
;  494     0750  5                    IF .LEN GTRU 0
;  495     0751  6                      THEN  BEGIN   ! output revised sequence
;  496     0752  6                            $OUTPUT_STRING( .LEN,.NEW_TEXT_PTR,.RENDITION);
;  497     0753  6                            CURSOR_COL=.CURSOR_COL+.LEN
;  498     0754  5                            END;    ! output revised sequence
;  499     0755  5
;  500     0756  5                    !+
;  501     0757  5                    ! Update our pointers and the number of chars left.
;  502     0758  5                    !-
;  503     0759  5
;  504     0760  5                    CUR_TEXT_PTR =.CUR_TEXT_PTR+.LEN;
;  505     0761  5                    CUR_ATTR_PTR =.CUR_ATTR_PTR+.LEN;
;  506     0762  5                    NEW_TEXT_PTR =.NEW_TEXT_PTR +.LEN;
;  507     0763  5                    NEW_ATTR_PTR =.NEW_ATTR_PTR +.LEN;
;  508     0764  5
;  509     0765  5                    CHARS_LEFT=.CHARS_LEFT-.LEN
;  510     0766  5
;  511     0767  5                    END     ! Characters disagree
;  512     0768  5
;  513     0769  3                END;    ! scan
;  514     0770  2            END;        ! scan row .ROW
;  515     0771  2
;  516     0772  2 !+
;  517     0773  2 ! Make the two buffers agree.
;  518     0774  2 ! The screen now contains what we think should be there.
;  519     0775  2 !-
;  520     0776  2
;  521     0777  2 CH$MOVE(.SIZE,NEW_TEXT,CUR_TEXT);
;  522     0778  2 CH$MOVE(.SIZE,NEW_ATTR,CUR_ATTR);
;  523     0779  2 CH$MOVE(.NUM_ROWS+1,NEW_LCV,CUR_LCV);
;  524     0780  2
;  525     0781  2 !+
;  526     0782  2 ! Move the cursor to the place where the user thinks it is.
```

```
;   527          0783  2 ! (But only if we are not already there.)
;   528          0784  2 !-
;   529          0785  2
;   530          0786  2 IF .CUR_LCV[.NEW_CURSOR_ROW] NEQ 0
;   531          0787  2   THEN  ADJUSTED_COL=.CURSOR_COL
;   532          0788  2   ELSE  ADJUSTED_COL=2*.CURSOR_COL-1;
;   533          0789  2
;   534          0790  2 OLD_CURSOR_ROW=.CURSOR_ROW;
;   535          0791  2 OLD_CURSOR_COL=.CURSOR_COL;
;   536          0792  2
;   537          0793  2 SMG$$UPDATE_PHYSICAL_CURSOR(PBCB);
;   538          0794  2
;   539          0795  2 RETURN SS$_NORMAL
;   540          0796  2
;   541          0797  1 END;                    ! End of routine SMG$$OUTPUT_MINIMAL_UPDATE

                                                .TITLE  SMG$MIN Minimal update calculation
                                                .IDENT  \1-016\

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

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

                                    0FFC 00000  .ENTRY  SMG$$OUTPUT_MINIMAL_UPDATE, Save R2,R3,R4,- ; 0408
                                                        R5,R6,R7,R8,R9,R10,R11
                                5E      FEC0  CE  9E 00002  MOVAB   -320(SP), SP                    ;
                                59        04  AC  D0 00007  MOVL    P_PBCB, R9                      ; 0451
                                5A        08  A9  D0 0000B  MOVL    8(R9), R10                      ; 0452
                                          14  AA  DD 0000F  PUSHL   20(R10)                         ; 0455
                                          0C  AA  DD 00012  PUSHL   12(R10)                         ; 0458
                                  24    00D0  C9     06 E1 00015  BBC     #6, 208(R9), 1$          ; 0505
28  AA                            00          6E     00 2C 0001B  MOVC5   #0, (SP), #0, 40(R10), @4(SP) ; 0507
                                              04  BE    00021
                                      00A8  C9     01 B0 00023  MOVW    #1, 168(R9)                ; 0508
                                      00AC  C9     01 B0 00028  MOVW    #1, 172(R9)                ; 0509
                                      00AA  C9     02 AA B0 0002D  MOVW    2(R10), 170(R9)         ; 0510
                                      00AE  C9     06 AA B0 00033  MOVW    6(R10), 174(R9)         ; 0511
                                      00D0  C9     40 8F 8A 00039  BICB2   #64, 208(R9)            ; 0512
                                                   18 AE D4 0003F 1$:  CLRL    CURSOR_ROW          ; 0524
                                                   10 AE D4 00042  CLRL    CURSOR_COL              ; 0525
                                        34  AE  00AA  C9 3C 00045  MOVZWL  170(R9), 52(SP)         ; 0527
                                        52      00A8  C9 3C 0004B  MOVZWL  168(R9), ROW
                                                    52 D7 00050  DECL    ROW
                                                 024A 31 00052  BRW     28$
                                          5B    FF    A2 9E 00055 2$:  MOVAB   -1(R2), R11         ; 0533
                                        08  AE    06    AA 3C 00059  MOVZWL  6(R10), 8(SP)
                                          5B    08    AE C4 0005E  MULL2   8(SP), R11
                                  53      5B    04    AE C1 00062  ADDL3   4(SP), R11, CUR_TEXT_PTR
                                        28  AE    18 BA4B 9E 00067  MOVAB   @24(R10)[R11], CUR_ATTR_PTR ; 0534
                                        20  AE    08 BA4B 9E 0006D  MOVAB   @8(R10)[R11], NEW_TEXT_PTR ; 0535
                                24  AE          6E    5B C1 00073  ADDL3   R11, (SP), NEW_ATTR_PTR  ; 0536
                                              50  2C BA42 9A 00078  MOVZBL  @44(R10)[ROW], R0        ; 0538
                                                    07 12 0007D  BNEQ    3$
                                        1C  AE      08  AE D0 0007F  MOVL    8(SP), CHARS_LEFT       ; 0540
                                                    06 11 00084  BRB     4$
```

```
                1C   AE        08   AE                02   C7 00086 3$:     DIVL3   #2, 8(SP), CHARS_LEFT                    ; 0542
                                   55            FF    A3   9E 0008C 4$:     MOVAB   -1(R3), PRE_PTR_IN_ROW                   ; 0544
                                   50            30 BA42    91 00090          CMPB    @48(R10)[ROW], R0                        ; 0550
                                                    03    12 00095          BNEQ    5$                                       ;
                                                  00EC    31 00097          BRW     13$                                      ;
                                                    01    DD 0009A 5$:      PUSHL   #1                                       ; 0565
                                                    52    DD 0009C          PUSHL   ROW                                      ; 0568
                                                    18   AE DD 0009E        PUSHL   CURSOR_COL                               ; 0567
                                                    24   AE DD 000A1        PUSHL   CURSOR_ROW                               ; 0566
                                                    59    DD 000A4          PUSHL   R9                                       ; 0565
                                          0000V     CF    FB 000A6          CALLS   #5, SMG$$FIND_MIN_CURSOR_POS             ;
                                            18      AE    D0 000AB          MOVL    ROW, CURSOR_ROW                          ; 0575
                                            10      AE    D0 000AF          MOVL    #1, CURSOR_COL                           ; 0576
                                                    50    D4 000B3          CLRL    BUFLEN                                   ; 0578
                                                    50            2C BA42    9A 000B5          MOVZBL  @44(R10)[ROW], R0        ; 0584
                                                    01            50    91 000BA          CMPB    R0, #1                   ; 0586
                                                                  20    12 000BD          BNEQ    6$                       ;
                                                                00FC    C9 D5 000BF       TSTL    252(R9)                  ;
                                                                  6E    13 000C3          BEQL    9$                       ;
                                                                  3C    AE D4 000C5       CLRL    INPUT_ARGS               ;
                                                                  3C    AE 9F 000C8       PUSHAB  INPUT_ARGS               ;
                                                                0104    C9 DD 000CB       PUSHL   260(R9)                  ;
                                                                0108    C9 9F 000CF       PUSHAB  264(R9)                  ;
                                                                0100    C9 9F 000D3       PUSHAB  256(R9)                  ;
                                            1C      AE          01CE    8F 3C 000D7       MOVZWL  #462, 28(SP)             ;
                                                                  72    11 000DD          BRB     11$                      ;
                                                    02            50    91 000DF 6$:      CMPB    R0, #2                   ; 0587
                                                                  20    12 000E2          BNEQ    7$                       ;
                                                                00FC    C9 D5 000E4       TSTL    252(R9)                  ;
                                                                  49    13 000E8          BEQL    9$                       ;
                                                                  3C    AE D4 000EA       CLRL    INPUT_ARGS               ;
                                                                  3C    AE 9F 000ED       PUSHAB  INPUT_ARGS               ;
                                                                0104    C9 DD 000F0       PUSHL   260(R9)                  ;
                                                                0108    C9 9F 000F4       PUSHAB  264(R9)                  ;
                                                                0100    C9 9F 000F8       PUSHAB  256(R9)                  ;
                                          1C    AE          01CD    8F 3C 000FC       MOVZWL  #461, 28(SP)             ;
                                                                  4D    11 00102          BRB     11$                      ;
                                                    03            50    91 00104 7$:      CMPB    R0, #3                   ; 0588
                                                                  20    12 00107          BNEQ    8$                       ;
                                                                00FC    C9 D5 00109       TSTL    252(R9)                  ;
                                                                  24    13 0010D          BEQL    9$                       ;
                                                                  3C    AE D4 0010F       CLRL    INPUT_ARGS               ;
                                                                  3C    AE 9F 00112       PUSHAB  INPUT_ARGS               ;
                                                                0104    C9 DD 00115       PUSHL   260(R9)                  ;
                                                                0108    C9 9F 00119       PUSHAB  264(R9)                  ;
                                                                0100    C9 9F 0011D       PUSHAB  256(R9)                  ;
                                          1C    AE          01CC    8F 3C 00121       MOVZWL  #460, 28(SP)             ;
                                                                  28    11 00127          BRB     11$                      ;
                                                                  50    D5 00129 8$:      TSTL    R0                       ; 0589
                                                                  36    12 0012B          BNEQ    12$                      ;
                                                                00FC    C9 D5 0012D       TSTL    252(R9)                  ;
                                                                  06    12 00131          BNEQ    10$                      ;
                                                                0108    C9 D4 00133 9$:   CLRL    264(R9)                  ;
                                                                  2A    11 00137          BRB     12$                      ;
                                                                  3C    AE D4 00139 10$:  CLRL    INPUT_ARGS               ;
                                                                  3C    AE 9F 0013C       PUSHAB  INPUT_ARGS               ;
                                                                0104    C9 DD 0013F       PUSHL   260(R9)                  ;
```

```
                                0108    C9  9F 00143          PUSHAB  264(R9)                         ;
                                0100    C9  9F 00147          PUSHAB  256(R9)                         ;
                1C      AE      023E    8F  3C 0014B          MOVZWL  #574, 28(SP)                    ;
                                  1C    AE  9F 00151   11$:   PUSHAB  28(SP)                          ;
                                00FC    C9  9F 00154          PUSHAB  252(R9)                         ;
         00000000G      00              06  FB 00158          CALLS   #6, SMG$GET_TERM_DATA           ;
                        01              50  E8 0015F          BLBS    STATUS, 12$                     ;
                                            04 00162          RET                                     ;
                        50      0108    C9  D0 00163   12$:   MOVL    264(R9), R0                     ; 0596
                                        1C  13 00168          BEQL    13$                             ;
                                0104    C9  DD 0016A          PUSHL   260(R9)                         ; 0599
                                        50  DD 0016E          PUSHL   R0                              ; 0598
                                        59  DD 00170          PUSHL   R9                              ;
         00000000G      00              03  FB 00172          CALLS   #3, SMG$$OUTPUT                 ;
                38      AE              50  D0 00179          MOVL    R0, STATUS                      ;
                        05        38    AE  E8 0017D          BLBS    STATUS, 13$                     ; 0600
                        50        38    AE  D0 00181          MOVL    STATUS, R0                      ;
                                            04 00185          RET                                     ;
      54        08      AE              01  C1 00186   13$:   ADDL3   #1, 8(SP), BLANK_COL            ; 0610
      50                52        08    AE  C5 0018B          MULL3   8(SP), ROW, R0                  ; 0611
                0C      AE      08 BA40     9E 00190          MOVAB   @8(R10)[R0], PTEXT              ;
      51                6E              50  C1 00196          ADDL3   R0, (SP), PATTR                 ; 0612
      50        08      AE              01  C1 0019A          ADDL3   #1, 8(SP), C                    ; 0613
                                        12  11 0019F          BRB     15$                             ;
                                  0C    AE  D7 001A1   14$:   DECL    PTEXT                           ; 0615
                                        51  D7 001A4          DECL    PATTR                           ; 0616
                        20        0C    BE  91 001A6          CMPB    @PTEXT, #32                     ; 0620
                                        0A  12 001AA          BNEQ    16$                             ;
                                        61  95 001AC          TSTB    (PATTR)                         ;
                                        06  12 001AE          BNEQ    16$                             ;
                        54              50  D0 001B0          MOVL    C, BLANK_COL                    ; 0621
                        EB              50  F5 001B3   15$:   SOBGTR  C, 14$                          ; 0613
                                  1C    AE  D5 001B6   16$:   TSTL    CHARS_LEFT                      ; 0626
                                        03  12 001B9          BNEQ    18$                             ;
                                      00E1  31 001BB   17$:   BRW     28$                             ;
                20      BE              63  91 001BE   18$:   CMPB    (CUR_TEXT_PTR), @NEW_TEXT_PTR   ; 0628
                                        17  12 001C2          BNEQ    19$                             ;
                24      BE        28    BE  91 001C4          CMPB    @CUR_ATTR_PTR, @NEW_ATTR_PTR    ; 0629
                                        10  12 001C9          BNEQ    19$                             ;
                                        53  D6 001CB          INCL    CUR_TEXT_PTR                    ; 0631
                                  28    AE  D6 001CD          INCL    CUR_ATTR_PTR                    ; 0632
                                  20    AE  D6 001D0          INCL    NEW_TEXT_PTR                    ; 0633
                                  24    AE  D6 001D3          INCL    NEW_ATTR_PTR                    ; 0634
                                  1C    AE  D7 001D6          DECL    CHARS_LEFT                      ; 0635
                                        DB  11 001D9          BRB     16$                             ;
      56                53        04    AE  C3 001DB   19$:   SUBL3   4(SP), CUR_TEXT_PTR, INDEX      ; 0639
 30   AE                53              55  C3 001E0          SUBL3   PRE_PTR_IN_ROW, CUR_TEXT_PTR, COL ; 0647
                08      AE        10    AE  D1 001E5          CMPL    CURSOR_COL, 8(SP)               ; 0665
                                        03  1B 001EA          BLEQU   20$                             ;
                                  10    AE  D4 001EC          CLRL    CURSOR_COL                      ; 0666
                                  30    AE  DD 001EF   20$:   PUSHL   COL                             ; 0672
                                        52  DD 001F2          PUSHL   ROW                             ; 0671
                                  18    AE  DD 001F4          PUSHL   CURSOR_COL                      ; 0670
                                  24    AE  DD 001F7          PUSHL   CURSOR_ROW                      ; 0669
                                        59  DD 001FA          PUSHL   R9                              ; 0668
             0000V      CF              05  FB 001FC          CALLS   #5, SMG$$FIND_MIN_CURSOR_POS    ;
                18      AE              52  D0 00201          MOVL    ROW, CURSOR_ROW                 ; 0680
```

```
                                10   AE        30   AE   D0 00205          MOVL    COL, CURSOR_COL                          ; 0681
                                54             10   AE   D1 0020A          CMPL    CURSOR_COL, BLANK_COL                    ; 0693
                                               0B   1F 0020E          BLSSU   21$                                      ;
                                               59   DD 00210          PUSHL   R9                                       ; 0696
                           0000V CF                  01   FB 00212          CALLS   #1, ERASE_LINE                   ;
                                 A1                  50   E8 00217          BLBS    STATUS, 17$                      ; 0697
                                                     04   0021A          RET                                      ;
                        50      08   AE             01   C3 0021B 21$:     SUBL3   #1, 8(SP), R0                    ; 0709
                        58           5B             50   C1 00220          ADDL3   R0, R11, END_ROW_INDEX           ;
                                     50             6E   D0 00224          MOVL    (SP), R0                         ; 0723
                                2C   AE          6640   9A 00227          MOVZBL  (INDEX)[R0], RENDITION           ;
                                     57       01    A8   9E 0022C          MOVAB   1(R8), FINAL_INDEX               ; 0724
                                     50             56   D0 00230          MOVL    INDEX, I                         ; 0731
                                                    28   11 00233          BRB     25$                              ;
                                     51             04   AE   D0 00235 22$:     MOVL    4(SP), R1                        ; 0733
                                   6041             08 BA40   91 00239          CMPB    @8(R10)[I], (I)[R1]              ;
                                                    0B   12 0023F          BNEQ    23$                              ;
                                     51                 6E   D0 00241          MOVL    (SP), R1                         ; 0734
                                18 BA40               6041   91 00244          CMPB    (I)[R1], @24(R10)[I]             ;
                                                    0C   13 0024A          BEQL    24$                              ;
                                     51                 6E   D0 0024C 23$:     MOVL    (SP), R1                         ; 0735
    2C   AE                   6041                   08   00   ED 0024F          CMPZV   #0, #8, (I)[R1], RENDITION       ;
                                                    05   13 00256          BEQL    25$                              ;
                                     57             50   D0 00258 24$:     MOVL    I, FINAL_INDEX                   ; 0737
                                                    04   11 0025B          BRB     26$                              ; 0736
                                D4   50             58   F3 0025D 25$:     AOBLEQ  END_ROW_INDEX, I, 22$            ; 0731
                           14   AE   57             56   C3 00261 26$:     SUBL3   INDEX, FINAL_INDEX, LEN          ; 0748
                                                    1C   13 00266          BEQL    27$                              ; 0750
                                                2C   AE   DD 00268          PUSHL   RENDITION                        ; 0752
                                                7E   7C 0026B          CLRQ    -(SP)                            ;
                                                2C   AE   DD 0026D          PUSHL   NEW_TEXT_PTR                     ;
                                                24   AE   DD 00270          PUSHL   LEN                              ;
                                                     59   DD 00273          PUSHL   R9                               ;
                                00000000G       00        06   FB 00275          CALLS   #6, SMG$$PUT_SCREEN              ;
                                                6E        50   E9 0027C          BLBC    STATUS, 31$                      ;
                                     10         AE        14   AE   C0 0027F          ADDL2   LEN, CURSOR_COL                  ; 0753
                                     53         14        AE   C0 00284 27$:     ADDL2   LEN, CUR_TEXT_PTR                ; 0760
                                     28         AE        14   AE   C0 00288          ADDL2   LEN, CUR_ATTR_PTR                ; 0761
                                     20         AE        14   AE   C0 0028D          ADDL2   LEN, NEW_TEXT_PTR                ; 0762
                                     24         AE        14   AE   C0 00292          ADDL2   LEN, NEW_ATTR_PTR                ; 0763
                                     1C         AE        14   AE   C2 00297          SUBL2   LEN, CHARS_LEFT                  ; 0765
                                                     FF17   31   0029C          BRW     16$                              ; 0627
FDAF                    52                   01        34   AE   F1 0029F 28$:     ACBL    52(SP), #1, ROW, 2$              ; 0527
                   04   BE        08        BA        28   AA   28 002A6          MOVC3   40(R10), @8(R10), @4(SP)         ; 0777
                   18   BA        00        BE        28   AA   28 002AD          MOVC3   40(R10), @0(SP), @24(R10)        ; 0778
                                     50             02   AA   3C 002B4          MOVZWL  2(R10), R0                       ; 0779
                                                    50   D6 002B8          INCL    R0                               ;
                   30   BA        2C        BA        50   28 002BA          MOVC3   R0, @44(R10), @48(R10)           ;
                                     50        20   AA   3C 002C0          MOVZWL  32(R10), R0                      ; 0786
                                     50        30   AA   C0 002C4          ADDL2   48(R10), R0                      ;
                                                    60   95 002C8          TSTB    (R0)                             ;
                                                    06   13 002CA          BEQL    29$                              ;
                                     50        10   AE   D0 002CC          MOVL    CURSOR_COL, ADJUSTED_COL         ; 0787
                                                    07   11 002D0          BRB     30$                              ;
                        50           10   AE             01   78 002D2 29$:     ASHL    #1, CURSOR_COL, ADJUSTED_COL     ; 0788
                                                    50   D7 002D7          DECL    ADJUSTED_COL                     ;
                                     24   AA        18   AE   B0 002D9 30$:     MOVW    CURSOR_ROW, 36(R10)              ; 0790
```

```
                     26  AA      10  AE  B0 002DE          MOVW    CURSOR_COL, 38(R10)                     ; 0791
                                     59  DD 002E3          PUSHL   R9                                      ; 0793
                 0000V  CF            01  FB 002E5          CALLS   #1, SMG$$UPDATE_PHYSICAL_CURSOR         ;
                        50            01  D0 002EA          MOVL    #1, R0                                  ; 0795
                                          04 002ED 31$:     RET                                             ; 0797

; Routine Size:  750 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
;  543     0798  1 %SBTTL 'SMG$$UPDATE_PHYSICAL_CURSOR'
;  544     0799  1 GLOBAL ROUTINE SMG$$UPDATE_PHYSICAL_CURSOR (P_PBCB) =
;  545     0800  1 !++
;  546     0801  1 ! FUNCTIONAL DESCRIPTION:
;  547     0802  1 !
;  548     0803  1 !       This routine forces the physical cursor to move to
;  549     0804  1 !       a new location specified in the WCB.
;  550     0805  1 !       It also updates any internal structures.
;  551     0806  1 !       The cursor is clipped to an appropriate place if it
;  552     0807  1 !       falls outside the physical screen.
;  553     0808  1 !
;  554     0809  1 ! CALLING SEQUENCE:
;  555     0810  1 !
;  556     0811  1 !       ret_status.wlc.v = SMG$$UPDATE_PHYSICAL_CURSOR ( P_PBCB.rab.r)
;  557     0812  1 !
;  558     0813  1 ! FORMAL PARAMETERS:
;  559     0814  1 !
;  560     0815  1 !       P_PBCB,rab.r            Address of pasteboard control block
;  561     0816  1 !
;  562     0817  1 ! IMPLICIT INPUTS:
;  563     0818  1 !
;  564     0819  1 !       WCB[WCB_W_CURR_CUR_ROW] Desired new row for physical cursor
;  565     0820  1 !
;  566     0821  1 !       WCB[WCB_W_CURR_CUR_COL] Desired new col for physical cursor
;  567     0822  1 !
;  568     0823  1 !       WCB[WCB_W_OLD_CUR_ROW]  Physical row where cursor now is
;  569     0824  1 !
;  570     0825  1 !       WCB[WCB_W_OLD_CUR_COL]  Physical col where cursor now is
;  571     0826  1 !
;  572     0827  1 ! IMPLICIT OUTPUTS:
;  573     0828  1 !
;  574     0829  1 !       WCB[WCB_W_CURR_CUR_ROW] New cursor row
;  575     0830  1 !
;  576     0831  1 !       WCB[WCB_W_CURR_CUR_COL] New cursor col
;  577     0832  1 !
;  578     0833  1 !       WCB[WCB_W_OLD_CUR_ROW]  New cursor row
;  579     0834  1 !
;  580     0835  1 !       WCB[WCB_W_OLD_CUR_COL]  New cursor col
;  581     0836  1 !
;  582     0837  1 ! COMPLETION STATUS:
;  583     0838  1 !
;  584     0839  1 !       SS$_NORMAL      Normal successful completion
;  585     0840  1 !
;  586     0841  1 ! SIDE EFFECTS:
;  587     0842  1 !
;  588     0843  1 !       The cursor may move to a new physical location
;  589     0844  1 !--
```

```
;  591     0845  2 BEGIN
;  592     0846  2
;  593     0847  2 BIND
;  594     0848  2
;  595     0849  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  596     0850  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  597     0851  2         NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  598     0852  2         NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  599     0853  2         NEW_LCV         = .WCB[WCB_A_LINE_CHAR]         : VECTOR[,BYTE],
;  600     0854  2         CUR_LCV         = .WCB[WCB_A_SCR_LINE_CHAR]     : VECTOR[,BYTE],
;  601     0855  2         OLD_CURSOR_ROW  =  WCB[WCB_W_OLD_CUR_ROW]       : SIGNED WORD,
;  602     0856  2         OLD_CURSOR_COL  =  WCB[WCB_W_OLD_CUR_COL]       : SIGNED WORD,
;  603     0857  2         NEW_CURSOR_ROW  =  WCB[WCB_W_CURR_CUR_ROW]      : SIGNED WORD,
;  604     0858  2         NEW_CURSOR_COL  =  WCB[WCB_W_CURR_CUR_COL]      : SIGNED WORD;
```

```
;  606     0859  2 IF .OLD_CURSOR_ROW NEQ .NEW_CURSOR_ROW
;  607     0860  2 OR .OLD_CURSOR_COL NEQ .NEW_CURSOR_COL
;  608     0861  3   THEN  BEGIN
;  609     0862  3
;  610     0863  3         !+
;  611     0864  3         ! If the desired location is off the screen,
;  612     0865  3         ! Clip it to the nearest edge.
;  613     0866  3         !-
;  614     0867  3
;  615     0868  3         IF .NEW_CURSOR_ROW LSS 1
;  616     0869  3           THEN  NEW_CURSOR_ROW=1;
;  617     0870  3
;  618     0871  3         IF .NEW_CURSOR_COL LSS 1
;  619     0872  3           THEN  NEW_CURSOR_COL=1;
;  620     0873  3
;  621     0874  3         IF .NEW_CURSOR_ROW GTRU .NUM_ROWS
;  622     0875  3           THEN  NEW_CURSOR_ROW=.NUM_ROWS;
;  623     0876  3
;  624     0877  3         IF .NEW_CURSOR_COL GTRU .NUM_COLS
;  625     0878  3           THEN  NEW_CURSOR_COL=.NUM_COLS;
;  626     0879  3
;  627     0880  3         !+
;  628     0881  3         ! Physically move the cursor there.
;  629     0882  3         !-
;  630     0883  3
;  631     0884  3         SMG$$FIND_MIN_CURSOR_POS(
;  632     0885  3                 PBCB,                   ! Pasteboard control block
;  633     0886  3                 .OLD_CURSOR_ROW,        ! Current location on screen
;  634     0887  3                 .OLD_CURSOR_COL,
;  635     0888  3                 .NEW_CURSOR_ROW,        ! Desired location
;  636     0889  3                 .NEW_CURSOR_COL);
;  637     0890  3
;  638     0891  2         END;
;  639     0892  2
;  640     0893  2 !+
;  641     0894  2 ! Make the new and the old cursor positions agree.
;  642     0895  2 !-
;  643     0896  2
;  644     0897  2 OLD_CURSOR_ROW=.NEW_CURSOR_ROW;
;  645     0898  2 OLD_CURSOR_COL=.NEW_CURSOR_COL;
;  646     0899  2
;  647     0900  2 ! Special try:
;  648     0901  2 ! If current line is special, mark the column as unknown.
;  649     0902  2
;  650     0903  2 IF .CUR_LCV[.NEW_CURSOR_ROW] NEQ 0
;  651     0904  2   THEN  OLD_CURSOR_COL=0;
;  652     0905  2
;  653     0906  2 RETURN  SS$_NORMAL
;  654     0907  2
;  655     0908  1 END;


003C 00000         .ENTRY  SMG$$UPDATE_PHYSICAL_CURSOR, Save R2,R3,R4,-; 0799
                           R5                                          ;
```

```
                                50      04   AC   D0 00002          MOVL    P_PBCB, R0                              ; 0849
                                52      08   A0   D0 00006          MOVL    8(R0), R2                               ; 0850
                                55      30   A2   D0 0000A          MOVL    48(R2), R5                              ; 0854
                                53      20   A2   9E 0000E          MOVAB   32(R2), R3                              ; 0857
                                54      22   A2   9E 00012          MOVAB   34(R2), R4                              ; 0858
                                63      24   A2   B1 00016          CMPW    36(R2), (R3)                            ; 0859
                                                  06 12 0001A          BNEQ    1$
                                64      26   A2   B1 0001C          CMPW    38(R2), (R4)                            ; 0860
                                                  41 13 00020          BEQL    6$
                                                  63 B5 00022 1$:      TSTW    (R3)                                    ; 0868
                                                  03 14 00024          BGTR    2$
                                63                01 B0 00026          MOVW    #1, (R3)                                ; 0869
                                                  64 B5 00029 2$:      TSTW    (R4)                                    ; 0871
                                                  03 14 0002B          BGTR    3$
                                64                01 B0 0002D          MOVW    #1, (R4)                                ; 0872
                                51      02   A2   3C 00030 3$:      MOVZWL  2(R2), R1                               ; 0874
        51              63      10                00 EC 00034          CMPV    #0, #16, (R3), R1
                                                  04 1B 00039          BLEQU   4$
                                63      02   A2   B0 0003B          MOVW    2(R2), (R3)                             ; 0875
                                51      06   A2   3C 0003F 4$:      MOVZWL  6(R2), R1                               ; 0877
        51              64      10                00 EC 00043          CMPV    #0, #16, (R4), R1
                                                  04 1B 00048          BLEQU   5$
                                64      06   A2   B0 0004A          MOVW    6(R2), (R4)                             ; 0878
                                7E              64 32 0004E 5$:      CVTWL   (R4), -(SP)                             ; 0889
                                7E              63 32 00051          CVTWL   (R3), -(SP)                             ; 0888
                                7E      26   A2   32 00054          CVTWL   38(R2), -(SP)                           ; 0887
                                7E      24   A2   32 00058          CVTWL   36(R2), -(SP)                           ; 0886
                                                  50 DD 0005C          PUSHL   R0                                      ; 0884
                        0000V   CF                05 FB 0005E          CALLS   #5, SMG$$FIND_MIN_CURSOR_POS
                                50                63 32 00063 6$:      CVTWL   (R3), R0                                ; 0897
                          24    A2                50 B0 00066          MOVW    R0, 36(R2)
                          26    A2                64 B0 0006A          MOVW    (R4), 38(R2)                            ; 0898
                                              6045 95 0006E          TSTB    (R0)[R5]                                ; 0903
                                                  03 13 00071          BEQL    7$
                                        26    A2   B4 00073          CLRW    38(R2)                                  ; 0904
                                50                01 D0 00076 7$:      MOVL    #1, R0                                  ; 0906
                                                  04 00079          RET                                             ; 0908

; Routine Size:  122 bytes,    Routine Base:  _SMG$CODE + 02EE
```

```
;  657     0909  1 %SBTTL 'SMG$SET_PHYSICAL_CURSOR'
;  658     0910  1 GLOBAL ROUTINE SMG$SET_PHYSICAL_CURSOR (PBID,P_ROW,P_COL) =
;  659     0911  1 !++
;  660     0912  1 ! FUNCTIONAL DESCRIPTION:
;  661     0913  1 !
;  662     0914  1 !       This routine moves the physical cursor on a physical
;  663     0915  1 !       screen to a particular location.
;  664     0916  1 !
;  665     0917  1 ! CALLING SEQUENCE:
;  666     0918  1 !
;  667     0919  1 !       ret_status.wlc.v = SMG$SET_PHYSICAL_CURSOR ( PBID.rl.r,P_ROW.rl.r,
;  668     0920  1 !                                                    P_COL.rl.r)
;  669     0921  1 !
;  670     0922  1 ! FORMAL PARAMETERS:
;  671     0923  1 !
;  672     0924  1 !       PBID.rl.r               Pasteboard id
;  673     0925  1 !
;  674     0926  1 !       P_ROW.rl.r              The row number to move to
;  675     0927  1 !
;  676     0928  1 !       P_COL.rl.r              The column number to move to
;  677     0929  1 !
;  678     0930  1 ! IMPLICIT INPUTS:
;  679     0931  1 !
;  680     0932  1 !       NONE
;  681     0933  1 !
;  682     0934  1 ! IMPLICIT OUTPUTS:
;  683     0935  1 !
;  684     0936  1 !       NONE
;  685     0937  1 !
;  686     0938  1 ! COMPLETION STATUS:
;  687     0939  1 !
;  688     0940  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  689     0941  1 !       SMG$_INVPAS_ID  Invalid pasteboard id
;  690     0942  1 !       SMG$_INVROW     Position is not within pasteboard (off top or bottom)
;  691     0943  1 !       SMG$_INVCOL     Position is not within pasteboard (off left or right)
;  692     0944  1 !       SS$_NORMAL      Normal successful completion
;  693     0945  1 !
;  694     0946  1 ! SIDE EFFECTS:
;  695     0947  1 !
;  696     0948  1 !       NONE
;  697     0949  1 !--
```

```
;  699     0950  2 BEGIN
;  700     0951  2 BIND
;  701     0952  2
;  702     0953  2         ROW             = .P_ROW,
;  703     0954  2         COL             = .P_COL;
;  704     0955  2
;  705     0956  2 LOCAL
;  706     0957  2
;  707     0958  2         STATUS,
;  708     0959  2         PBCB            : REF $PBCB_DECL,
;  709     0960  2         WCB             : REF $WCB_DECL;
;  710     0961  2
;  711     0962  2 EXTERNAL LITERAL
;  712     0963  2
;  713     0964  2         SMG$_INVROW,
;  714     0965  2         SMG$_INVCOL;
```

```
;  716          0966  2 $SMG$VALIDATE_ARGCOUNT(3,3);
;  717          0967  2
;  718          0968  2 $SMG$GET_PBCB(.PBID,PBCB);
;  719          0969  2
;  720          0970  2 WCB=.PBCB[PBCB_A_WCB];
;  721          0971  2
;  722          0972  3         BEGIN
;  723          0973  3
;  724          0974  3         BIND
;  725          0975  3
;  726          0976  3                 NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  727          0977  3                 NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  728          0978  3                 CUR_ROW         =  WCB[WCB_W_CURR_CUR_ROW]      : WORD,
;  729          0979  3                 CUR_COL         =  WCB[WCB_W_CURR_CUR_COL]      : WORD;
;  730          0980  3
;  731          0981  3         IF .ROW GTRU .NUM_ROWS
;  732          0982  3           THEN  RETURN  SMG$_INVROW;
;  733          0983  3         IF .COL GTRU .NUM_COLS
;  734          0984  3           THEN  RETURN  SMG$_INVCOL;
;  735          0985  3
;  736          0986  3         CUR_ROW=.ROW;
;  737          0987  3         CUR_COL=.COL;
;  738          0988  3
;  739          0989  2         END;
;  740          0990  2
;  741          0991  2 !+
;  742          0992  2 ! Immediately move it there now if batching is not in effect.
;  743          0993  2 !-
;  744          0994  2
;  745          0995  2 IF .PBCB[PBCB_L_BATCH_LEVEL] EQL 0
;  746          0996  3   THEN  BEGIN    ! Move cursor
;  747          0997  3         STATUS=SMG$$UPDATE_PHYSICAL_CURSOR(.PBCB);
;  748          0998  3         IF NOT .STATUS THEN RETURN .STATUS
;  749          0999  2         END;    ! Move cursor
;  750          1000  2
;  751          1001  2 RETURN  SS$_NORMAL
;  752          1002  2
;  753          1003  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$SET_PHYSICAL_CURSOR, Save nothing   ; 0910
                          03        6C   91 00002       CMPB    (AP), #3                                ;  0966
                                    08   13 00005       BEQL    1$                                      ;
                50 00000000G        8F   D0 00007       MOVL    #SMG$_WRONUMARG, R0                     ;
                                    04   0000E         RET                                             ;
                50          04      BC   D0 0000F 1$:   MOVL    @PBID, R0                               ;  0968
                                    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      D0 0002E 3$:  MOVL    PBD_A_PBCB[R0], PBCB                 ;
                    50           08  A1   D0 00036      MOVL    8(PBCB), WCB                         ; 0970
   08  BC     02  A0             10  00   ED 0003A      CMPZV   #0, #16, 2(WCB), @P_ROW              ; 0981
                                     08   1E 00041      BGEQU   4$                                   ;
                    50 00000000G     8F   D0 00043      MOVL    #SMG$_INVROW, R0                     ; 0982
                                          04 0004A      RET                                          ;
   0C  BC     06  A0             10  00   ED 0004B 4$:  CMPZV   #0, #16, 6(WCB), @P_COL              ; 0983
                                     08   1E 00052      BGEQU   5$                                   ;
                    50 00000000G     8F   D0 00054      MOVL    #SMG$_INVCOL, R0                     ; 0984
                                          04 0005B      RET                                          ;
              20  A0           08    BC   B0 0005C 5$:  MOVW    @P_ROW, 32(WCB)                      ; 0986
              22  A0           0C    BC   B0 00061      MOVW    @P_COL, 34(WCB)                      ; 0987
                             00A4    C1   D5 00066      TSTL    164(PBCB)                            ; 0995
                                     0A   12 0006A      BNEQ    6$                                   ;
                                     51   DD 0006C      PUSHL   PBCB                                 ; 0997
            FF13  CF                 01   FB 0006E      CALLS   #1, SMG$$UPDATE_PHYSICAL_CURSOR      ;
                  03                 50   E9 00073      BLBC    STATUS, 7$                           ; 0998
                  50                 01   D0 00076 6$:  MOVL    #1, R0                               ; 1001
                                          04 00079 7$:  RET                                          ; 1003

; Routine Size:  122 bytes,    Routine Base:  _SMG$CODE + 0368
```

```
;  755     1004  1 %SBTTL 'SMG$$FIND_MIN_CURSOR_POS - Find minimum cursor pos. sequence'
;  756     1005  1 GLOBAL ROUTINE SMG$$FIND_MIN_CURSOR_POS (
;  757     1006  1                                         P_PBCB,
;  758     1007  1                                         LINE_NO,
;  759     1008  1                                         COL_NO,
;  760     1009  1                                         DESIRED_LINE_NO,
;  761     1010  1                                         DESIRED_COL_NO
;  762     1011  1                                         ) =
;  763     1012  1 !++
;  764     1013  1 ! FUNCTIONAL DESCRIPTION:
;  765     1014  1 !
;  766     1015  1 !
;  767     1016  1 ! CALLING SEQUENCE:
;  768     1017  1 !
;  769     1018  1 !       ret_status.wlc.v = SMG$$FIND_MIN_CURSOR_POS (
;  770     1019  1 !                                         P_PBCB.rab.r,
;  771     1020  1 !                                         LINE_NO.rl.v,
;  772     1021  1 !                                         COL_NO.rl.v,
;  773     1022  1 !                                         DESIRED_LINE_NO.rl.v,
;  774     1023  1 !                                         DESIRED_COL_NO.rl.v)
;  775     1024  1 !
;  776     1025  1 ! FORMAL PARAMETERS:
;  777     1026  1 !
;  778     1027  1 !       P_PBCB.rab.r            Address of PBCB
;  779     1028  1 !
;  780     1029  1 !       LINE_NO.rl.v            Current cursor line number
;  781     1030  1 !                               0 means it is unknown.
;  782     1031  1 !
;  783     1032  1 !       COL_NO.rl.v             Current cursor column number
;  784     1033  1 !                               0 means it is unknown.
;  785     1034  1 !
;  786     1035  1 !       DESIRED_LINE_NO.rl.v    Desired cursor line number position
;  787     1036  1 !
;  788     1037  1 !       DESIRED_COL_NO.rl.v     Desired cursor column number position
;  789     1038  1 !
;  790     1039  1 ! IMPLICIT INPUTS:
;  791     1040  1 !
;  792     1041  1 !       NONE
;  793     1042  1 !
;  794     1043  1 ! IMPLICIT OUTPUTS:
;  795     1044  1 !
;  796     1045  1 !       NONE
;  797     1046  1 !
;  798     1047  1 ! COMPLETION STATUS:
;  799     1048  1 !
;  800     1049  1 !       SS$_NORMAL      Normal successful completion
;  801     1050  1 !
;  802     1051  1 ! SIDE EFFECTS:
;  803     1052  1 !
;  804     1053  1 !       NONE
;  805     1054  1 !--
```

```
;  807     1055  2 BEGIN
;  808     1056  2
;  809     1057  2 BIND
;  810     1058  2
;  811     1059  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  812     1060  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  813     1061  2         NUM_ROWS        =  WCB[WCB_W_NO_ROWS]           : WORD,
;  814     1062  2         NUM_COLS        =  WCB[WCB_W_NO_COLS]           : WORD,
;  815     1063  2         CURR_TEXT       = .WCB[WCB_A_SCR_TEXT_BUF]      : VECTOR[,BYTE],
;  816     1064  2         CURR_ATTR       = .WCB[WCB_A_SCR_ATTR_BUF]      : VECTOR[,BYTE],
;  817     1065  2         LCV             = .WCB[WCB_A_LINE_CHAR]         : VECTOR[,BYTE],
;  818     1066  2         TERM_TYPE       =  PBCB[PBCB_B_DEVTYPE]         : BYTE;
;  819     1067  2
;  820     1068  2 LITERAL
;  821     1069  2
;  822     1070  2         INFINITY = 1000;        ! Prohibitively large number, used
;  823     1071  2                                 ! to reject a sequence.
;  824     1072  2
;  825     1073  2 BUILTIN
;  826     1074  2
;  827     1075  2         EDIV;
;  828     1076  2
;  829     1077  2 LOCAL
;  830     1078  2
;  831     1079  2         TRIAL_STRING : VECTOR [SMG$K_LONGEST_SEQUENCE,BYTE],
;  832     1080  2                                 ! Buffer in which to construct string
;  833     1081  2                                 ! to be output.
;  834     1082  2         TS_LEN,                 ! Current length of TRIAL_STRING.
;  835     1083  2         ADJUSTED_WIDTH,         ! Width or width/2
;  836     1084  2         SET_CUR_LEN;            ! Length of the general set_cursor
;  837     1085  2                                 ! sequence to reposition cursor.
```

```
;  839          1086  2 !+
;  840          1087  2 ! If the current position is unknown,
;  841          1088  2 ! then we must use the most general sequence.
;  842          1089  2 !-
;  843          1090  2
;  844          1091  2 IF .LINE_NO EQL 0
;  845          1092  2 OR .COL_NO  EQL 0
;  846          1093  2   THEN  RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  847          1094  2
;  848          1095  2 !+
;  849          1096  2 ! General strategy is to come up with a sequence of characters that
;  850          1097  2 ! will position us to the desired line and column number in less
;  851          1098  2 ! characters than a set_cursor sequence will need.
;  852          1099  2 ! The short-cut sequences to get to a specific line include:
;  853          1100  2 !       1. <LF's> to move down the screen.
;  854          1101  2 ! The short-cut sequences to get to a specific column include:
;  855          1102  2 !       1. <TAB> to tab-stop immediately before desired column and
;  856          1103  2 !       repeat a number of the current characters until we get to
;  857          1104  2 !       desired column position.
;  858          1105  2 !       2. <TAB> to tab-stop immediately beyond desired column and
;  859          1106  2 !       follow that by a number of <BS's> to get to the desired column.
;  860          1107  2 !       If at any point the trial sequence of characters gets to be
;  861          1108  2 ! greater than the set_cursor sequence, abandon the effort and use the
;  862          1109  2 ! set_cursor sequence.
;  863          1110  2 !-
;  864          1111  2
;  865          1112  2 TS_LEN = 0;                        ! Length of string constructed so far
;  866          1113  2
;  867          1114  2 !+
;  868          1115  2 ! Calculate what the cost of a set_cursor sequence is will be for the
;  869          1116  2 ! desired line and column number.  This will give us the lower bound we
;  870          1117  2 ! must beat if an alternate sequence is better.
;  871          1118  2 !-
;  872          1119  2
;  873          1120  2 $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.DESIRED_LINE_NO,.DESIRED_COL_NO);
;  874          1121  2 SET_CUR_LEN = .PBCB[PBCB_L_CAP_LENGTH];
;  875          1122  2
;  876          1123  2 !+
;  877          1124  2 ! Now see if we are already on the proper line.
;  878          1125  2 !-
;  879          1126  2
;  880          1127  2 IF .LINE_NO NEQ .DESIRED_LINE_NO
;  881          1128  2 THEN
;  882          1129  3         BEGIN   ! Adjust line number
;  883          1130  3         IF .DESIRED_LINE_NO LSS .LINE_NO
;  884          1131  3         THEN
;  885          1132  4             BEGIN       ! Move upward
;  886          1133  4
;  887          1134  4             !+
;  888          1135  4             ! No choice -- must use general cursor sequencing to move
;  889          1136  4             ! upward.  Output general set_cursor sequence
;  890          1137  4             ! (using DESIRED_LINE_NO and
;  891          1138  4             ! DESIRED_COL_NO) and return to caller.
;  892          1139  4             !-
;  893          1140  4
;  894          1141  4             RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO)
;  895          1142  4
```

```
;  896     1143  4              END             ! Move upward
;  897     1144  3          ELSE
;  898     1145  4              BEGIN           ! Move downward
;  899     1146  4              LOCAL
;  900     1147  4                  WIDE_WARNING,   ! TRUE if spanning across a wide line
;  901     1148  4                  LINES_DOWN ;    ! No. of lines down we need to move
;  902     1149  4
;  903     1150  4               !+
;  904     1151  4               ! See if we can reach DESIRED_LINE_NO in a number of <LF's>
;  905     1152  4               ! which is less than the number of characters in the
;  906     1153  4               ! set_cursor sequence.
;  907     1154  4               ! We do not permit line feed through the bottom of the scrolling
;  908     1155  4               ! region, since the cursor would not be able to cross it that way
;  909     1156  4               ! (and it would cause a scroll to occur).
;  910     1157  4               ! We do not permit line feed through a double wide (or double high)
;  911     1158  4               ! line, because in some cases, this doesn't work.  In particular,
;  912     1159  4               ! on a VT100, if you are in column 60, say and line feed down
;  913     1160  4               ! through a double wide line, when you get back to a single
;  914     1161  4               ! wide line, the cursor has now gotten to column 40!
;  915     1162  4               !-
;  916     1163  4
;  917     1164  4               LINES_DOWN = .DESIRED_LINE_NO - .LINE_NO;
;  918     1165  4
;  919     1166  4               !+
;  920     1167  4               ! Set WIDE_WARNING to TRUE if we would cross through or into or
;  921     1168  4               ! from a wide line.  Double high lines are considered to be wide.
;  922     1169  4               !-
;  923     1170  4
;  924     1171  4               WIDE_WARNING=0;
;  925     1172  4               IF .[CV[0] NEQ 0
;  926     1173  4               THEN
;  927     1174  4                   INCR L FROM .LINE_NO TO .DESIRED_LINE_NO DO
;  928     1175  4                           IF .LCV[.[] NEQ 0
;  929     1176  5                             THEN  BEGIN
;  930     1177  5                                   WIDE_WARNING=1;
;  931     1178  5                                   EXIT[OOP
;  932     1179  4                                   END;
;  933     1180  4
;  934     1181  4               IF (.LINES_DOWN LSS .SET_CUR_LEN)      AND
;  935     1182  5                  (.LINE_NO + .LINES_DOWN LEQU .PBCB[PBCB_W_BOT_SCROLL_LINE]
;  936     1183  4                   OR .LINE_NO GTRU .PBCB[PBCB_W_BOT_SCRO[L_LINE]) AND
;  937     1184  5                 (NOT .WIDE_WARNING)
;  938     1185  4               THEN
;  939     1186  5                   BEGIN   ! Do it with <LF's>
;  940     1187  5                   !+
;  941     1188  5                   ! Put (.LINES_DOWN) <LF's> into TRIAL_STRING and set
;  942     1189  5                   ! TS_LEN to .[INES_DOWN.
;  943     1190  5                   !-
;  944     1191  5                   CH$FILL (LF, .LINES_DOWN, TRIAL_STRING);
;  945     1192  5                   TS_LEN = .LINES_DOWN;
;  946     1193  5                   END     ! Do it with <LF's>
;  947     1194  4               ELSE
;  948     1195  5                   BEGIN   ! Too far
;  949     1196  5                   !+
;  950     1197  5                   ! Too far down or we would be crossing a lower scroll
;  951     1198  5                   ! boundary or a wide line -- use general set cursor sequence
;  952     1199  5                   !-
```

```
;  953     1200  5                  RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO)
;  954     1201  4                  END;        ! Too far
;  955     1202  3              END;            ! Move downward
;  956     1203  2          END;    ! Adjust line number
;  957     1204  2
;  958     1205  2          !+
;  959     1206  2          ! Reach here when we have constructed the minimal sequence to reach the
;  960     1207  2          ! desired line --not using general cursor addressing sequence.  TS_LEN
;  961     1208  2          ! tells us how long that sequence is.
;  962     1209  2          !-
;  963     1210  2
;  964     1211  2          IF .COL_NO NEQ .DESIRED_COL_NO
;  965     1212  2          THEN
;  966     1213  3                  BEGIN    ! Column adjustment
;  967     1214  3                  LOCAL
;  968     1215  3                      LEAST_COST, ! Least cost among considered strategies
;  969     1216  3                      BEST_STRAT, ! Best update strategy which is better
;  970     1217  3                                  ! then general cursor positioning sequence.
;  971     1218  3                      INDEX,      ! Index into CURR_TEXT and CURR_ATTR
;  972     1219  3                      DCN_QUAD : VECTOR [2,LONG], ! Desired column number
;  973     1220  3                                  ! as a quadword
;  974     1221  3                      DELTA_COL,  ! No. of columns between where we are and where
;  975     1222  3                                  ! we want to be.
;  976     1223  3                      NO_TABS,    ! No. of <TAB's> to get to tab-stop before
;  977     1224  3                                  ! DESIRED_COL_NO.
;  978     1225  3                      NO_RETYPES, ! No. of chars that need to be retyped if we
;  979     1226  3                                  ! tab to tab-stop before
;  980     1227  3                      NO_BS;      ! No. of <BS's> to get from tab-stop beyond
;  981     1228  3                                  ! DESIRED_COL_NO back to DESIRED_COL_NO.
;  982     1229  3
;  983     1230  3                  !+
;  984     1231  3                  ! Construct short-cut sequence to position to desired column
;  985     1232  3                  ! number.
;  986     1233  3                  ! If earlier on line, 3 strategies are possible:
;  987     1234  3                  !    1.  Do it with backspaces
;  988     1235  3                  !    2.  Do it with <CR> and <TAB's> to tab-stop before followed
;  989     1236  3                  !        by retypes.
;  990     1237  3                  !    3.  Do it with <CR> and <TAB's> to tab-stop beyond followed
;  991     1238  3                  !        by <BS's>.
;  992     1239  3                  ! If later on line, 3 strategies are possible:
;  993     1240  3                  !    4.  Do it with retypes.
;  994     1241  3                  !    5.  Do it with <TAB's> to tab-stop before followed by
;  995     1242  3                  !        retypes.
;  996     1243  3                  !    6.  Do it with <TAB's> to tab-stop after followed by <BS's>.
;  997     1244  3                  !-
;  998     1245  3
;  999     1246  3                  !+
; 1000     1247  3                  ! Calc. no of <TAB's> needed to get to tab-stop before
; 1001     1248  3                  ! DESIRED_COL_NO and the no. of subsequent retypes needed.
; 1002     1249  3                  !-
; 1003     1250  3
; 1004     1251  3                  DCN_QUAD [0] = .DESIRED_COL_NO -1;
; 1005     1252  3                  DCN_QUAD [1] = 0;
; 1006     1253  3                  EDIV ( %REF(8), DCN_QUAD[0], NO_TABS, NO_RETYPES);
; 1007     1254  3
; 1008     1255  3                  !+
; 1009     1256  3                  ! If terminal doesn't support tabs,
```

```
;  1010     1257  3          ! or user doesn't want them,
;  1011     1258  3          ! then set NO_TABS to infinity.
;  1012     1259  3          !-
;  1013     1260  3
;  1014     1261  3          IF .PBCB[PBCB_V_NOTABS] OR NOT .PBCB[PBCB_V_TABS]
;  1015     1262  3            THEN  NO_TABS=INFINITY;
;  1016     1263  3
;  1017     1264  3          !+
;  1018     1265  3          ! Calc. number of <BS's> needed if we go to tab-stop after
;  1019     1266  3          ! DESIRED_COL_NO. This strategy can't be followed if the
;  1020     1267  3          ! next tab stop is off past the right of the screen.  In
;  1021     1268  3          ! that case, we make NO_BS prohibitively large.
;  1022     1269  3          !-
;  1023     1270  3
;  1024     1271  3          IF .LCV[.DESIRED_LINE_NO] NEQ 0
;  1025     1272  3            THEN  ADJUSTED_WIDTH=.NUM_COLS/2
;  1026     1273  3            ELSE  ADJUSTED_WIDTH=.NUM_COLS;
;  1027     1274  3
;  1028     1275  3          IF (.NO_TABS+1)*8+1 LSSU .ADJUSTED_WIDTH
;  1029     1276  3            THEN  NO_BS = 8 - .NO_RETYPES
;  1030     1277  3            ELSE  NO_BS = INFINITY;
;  1031     1278  3
;  1032     1279  3          !+
;  1033     1280  3          ! Set NO_BS to infinity if the terminal does not support backspacing.
;  1034     1281  3          !-
;  1035     1282  3
;  1036     1283  3          IF NOT .PBCB[PBCB_V_BS]
;  1037     1284  3            THEN  NO_BS=INFINITY;
;  1038     1285  3
;  1039     1286  3          !+
;  1040     1287  3          ! In case we need to do retypes, calc. where in CURR_TEXT and
;  1041     1288  3          ! CURR_ATTR we need to look.
;  1042     1289  3          !-
;  1043     1290  3
;  1044     1291  3          INDEX = $L ( .DESIRED_LINE_NO, ((.NO_TABS*8) + 1));
;  1045     1292  3
;  1046     1293  3          IF .DESIRED_COL_NO  LEQ  .COL_NO
;  1047     1294  3          THEN
;  1048     1295  4              BEGIN       ! Earlier in line
;  1049     1296  4              LOCAL
;  1050     1297  4
;  1051     1298  4                  S1_COST, S2_COST, S3_COST;      ! Cost of strategies
;  1052     1299  4                                                  ! S1: just BS
;  1053     1300  4                                                  ! S2: tabs then retype
;  1054     1301  4                                                  ! S3: tabs then BS
;  1055     1302  4
;  1056     1303  4              ! Find the cost of stategies for moving back in line
;  1057     1304  4
;  1058     1305  4              IF .PBCB[PBCB_V_BS]
;  1059     1306  4              THEN
;  1060     1307  4                      S1_COST = .COL_NO - .DESIRED_COL_NO     ! No of <BS's>
;  1061     1308  4              ELSE
;  1062     1309  4                      S1_COST=INFINITY;
;  1063     1310  4
;  1064     1311  4              S2_COST = 1                     ! For <CR>
;  1065     1312  4                      + .NO_TABS              ! For no. of tabs to tab-stop
;  1066     1313  4                                              ! before
```

```
;  1067     1314  4                                + .NO_RETYPES;    ! For no. of retypes
;  1068     1315  4
;  1069     1316  4                    S3_COST = 1                    ! For <CR>
;  1070     1317  4                                + .NO_TABS + 1     ! For no. of tabs to tab-stop
;  1071     1318  4                                                   ! after
;  1072     1319  4                                + .NO_BS;          ! For no. of <BS's>
;  1073     1320  4
;  1074     1321  4                    ! Find best strategy for moving backward in line
;  1075     1322  4
;  1076     1323  4                    BEST_STRAT = 1;              LEAST_COST = .S1_COST;
;  1077     1324  4
;  1078     1325  4                    IF .S2_COST LSS .LEAST_COST THEN
;  1079     1326  4                        BEGIN  BEST_STRAT = 2;  LEAST_COST = .S2_COST;  END;
;  1080     1327  4
;  1081     1328  4                    IF .S3_COST LSS .LEAST_COST THEN
;  1082     1329  4                        BEGIN BEST_STRAT =3 ;   LEAST_COST = .S3_COST; END;
;  1083     1330  4                    END ! Earlier in line
;  1084     1331  4
;  1085     1332  3                ELSE
;  1086     1333  3
;  1087     1334  4                    BEGIN       ! Later in line
;  1088     1335  4                    LOCAL
;  1089     1336  4                        S4_COST, S5_COST, S6_COST;      ! Cost of strategies
;  1090     1337  4
;  1091     1338  4                    ! Find costs of strategies for moving forward in line
;  1092     1339  4
;  1093     1340  4                    S4_COST = .DESIRED_COL_NO - .COL_NO; ! For just retypes
;  1094     1341  4
;  1095     1342  4                    IF (.NO_TABS * 8)+1 GTR .COL_NO AND .PBCB[PBCB_V_TABS]
;  1096     1343  4                        AND NOT .PBCB[PBCB_V_NOTABS]
;  1097     1344  4                    THEN
;  1098     1345  5                        BEGIN   ! Tabbing forward is possible
;  1099     1346  5                        LOCAL
;  1100     1347  5                            COL_QUAD : VECTOR [2,LONG], ! COL_NO as quadword
;  1101     1348  5                            NEW_NO_TABS,
;  1102     1349  5                            NEW_NO_RETYPES;
;  1103     1350  5
;  1104     1351  5                        COL_QUAD [0]  = .COL_NO  - 1;
;  1105     1352  5                        COL_QUAD [1] = 0;
;  1106     1353  5                        EDIV (%REF(8), COL_QUAD [0], NEW_NO_TABS, NEW_NO_RETYPES);
;  1107     1354  5                        NO_TABS = .NO_TABS - .NEW_NO_TABS;
;  1108     1355  5                        S5_COST = .NO_TABS          ! For no. of tabs to tab-stop
;  1109     1356  5                                                    ! before from current position
;  1110     1357  5                                  + .NO_RETYPES;! For no. of retypes
;  1111     1358  5
;  1112     1359  5                        S6_COST = .NO_TABS + 1  ! For no. of tabs to tab-stop
;  1113     1360  5                                                ! after from current position
;  1114     1361  5                                  + .NO_BS;     ! For no. of <BS's>
;  1115     1362  5                        END     ! Tabbing forward is possible
;  1116     1363  4                    ELSE
;  1117     1364  5                        BEGIN   ! Tabbing forward not possible
;  1118     1365  5                        S5_COST = INFINITY;     ! Set to prohibitive value
;  1119     1366  5                        S6_COST = INFINITY;     ! Set to prohibitive value
;  1120     1367  4                        END;    ! Tabbing forward not possible
;  1121     1368  4
;  1122     1369  4                    ! Find best strategy
;  1123     1370  4
```

```
;  1124     1371  4              BEST_STRAT = 4;                         LEAST_COST = .S4_COST;
;  1125     1372  4
;  1126     1373  4              IF .S5_COST LSS .LEAST_COST THEN
;  1127     1374  4                  BEGIN  BEST_STRAT = 5;  LEAST_COST = .S5_COST;  END;
;  1128     1375  4
;  1129     1376  4              IF .S6_COST LSS .LEAST_COST THEN
;  1130     1377  4                  BEGIN  BEST_STRAT = 6; LEAST_COST = .S6_COST; END;
;  1131     1378  3              END;        ! Later in line
;  1132     1379  3
;  1133     1380  3          IF .TS_LEN + .LEAST_COST GTR .SET_CUR_LEN
;  1134     1381  3          THEN
;  1135     1382  4              BEGIN       ! Abandon effort
;  1136     1383  4              RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO)
;  1137     1384  3              END;        ! Abandon effort
;  1138     1385  3
;  1139     1386  3          CASE .BEST_STRAT FROM 1 TO 6 OF
;  1140     1387  3          SET
;  1141     1388  4              [1]:BEGIN   ! Backspaces only.
;  1142     1389  4                  NO_BS = .COL_NO - .DESIRED_COL_NO ;
;  1143     1390  4                  CH$FILL ( BS, .NO_BS, TRIAL_STRING [.TS_LEN]);
;  1144     1391  4                  TS_LEN = .TS_LEN + .NO_BS;
;  1145     1392  3                  END;    ! Backspace only.
;  1146     1393  3
;  1147     1394  4              [2]:BEGIN   ! <CR>, <TAB's> to tab-stop before, retypes.
;  1148     1395  4
;  1149     1396  4                  !+
;  1150     1397  4                  ! If there are actually characters to be retyped and
;  1151     1398  4                  ! attributes are involved, give up and resort to general
;  1152     1399  4                  ! cursor positioning sequence.
;  1153     1400  4                  ! It will cost us too much to select-graphic-rendition
;  1154     1401  4                  ! and undo select graphic rendition.
;  1155     1402  4                  !-
;  1156     1403  4
;  1157     1404  4                  IF .NO_RETYPES NEQ 0  AND
;  1158     1405  4                     CH$COMPARE (0, 0,     ! len, addr
;  1159     1406  4                                 .NO_RETYPES, CURR_ATTR[.INDEX],
;  1160     1407  4                                 0         ! fill
;  1161     1408  4                                ) NEQ 0
;  1162     1409  4                  THEN
;  1163     1410  4                      RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  1164     1411  4
;  1165     1412  4                  TRIAL_STRING [.TS_LEN] = CR;
;  1166     1413  4                  TS_LEN = .TS_LEN + 1;
;  1167     1414  4                  CH$FILL ( TAB, .NO_TABS, TRIAL_STRING [.TS_LEN]);
;  1168     1415  4                  TS_LEN = .TS_LEN + .NO_TABS;
;  1169     1416  4                  CH$MOVE ( .NO_RETYPES, CURR_TEXT [.INDEX],
;  1170     1417  4                            TRIAL_STRING [.TS_LEN]);
;  1171     1418  4                  TS_LEN = .TS_LEN + .NO_RETYPES;
;  1172     1419  3                  END;    ! <CR>, <TAB's> to tab-stop before, retypes.
;  1173     1420  3
;  1174     1421  4              [3]:BEGIN   ! <CR>, <TAB's> to tab-stop after, <BS's>
;  1175     1422  4                  TRIAL_STRING [.TS_LEN] = CR;
;  1176     1423  4                  TS_LEN = .TS_LEN + 1;
;  1177     1424  4                  CH$FILL ( TAB, .NO_TABS + 1, TRIAL_STRING [.TS_LEN]);
;  1178     1425  4                  TS_LEN = .TS_LEN + .NO_TABS + 1;
;  1179     1426  4                  CH$FILL ( BS, .NO_BS, TRIAL_STRING [.TS_LEN]);
;  1180     1427  4                  TS_LEN = .TS_LEN + .NO_BS;
```

```
;  1181     1428  3                END;    ! <CR>, <TAB's> to tab-stop after, <BS's>
;  1182     1429  3
;  1183     1430  4            [4]:BEGIN   ! Retypes only.
;  1184     1431  4
;  1185     1432  4                !+
;  1186     1433  4                ! If there are actually characters to be retyped and
;  1187     1434  4                ! attributes are involved, give up and resort to general
;  1188     1435  4                ! cursor positioning sequence.
;  1189     1436  4                ! It will cost us too much to select-graphic-rendition
;  1190     1437  4                ! and undo select graphic rendition.
;  1191     1438  4                !-
;  1192     1439  4
;  1193     1440  4                NO_RETYPES = .DESIRED_COL_NO - .COL_NO;
;  1194     1441  4                INDEX = $L ( .DESIRED_LINE_NO, .COL_NO);
;  1195     1442  4                IF .NO_RETYPES NEQ 0  AND
;  1196     1443  4                   CH$COMPARE (0, 0,    ! len, addr
;  1197     1444  4                               .NO_RETYPES, CURR_ATTR[.INDEX],
;  1198     1445  4                               0        ! fill
;  1199     1446  4                              ) NEQ 0
;  1200     1447  4                THEN
;  1201     1448  4                    RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  1202     1449  4
;  1203     1450  4                CH$MOVE ( .NO_RETYPES, CURR_TEXT [.INDEX],
;  1204     1451  4                          TRIAL_STRING [.TS_LEN]);
;  1205     1452  4                TS_LEN = .TS_LEN + .NO_RETYPES;
;  1206     1453  3                END;    ! Retypes only.
;  1207     1454  3
;  1208     1455  4            [5]:BEGIN   ! <TAB's> to tab-stop before, retypes.
;  1209     1456  4
;  1210     1457  4                !+
;  1211     1458  4                ! If there are actually characters to be retyped and
;  1212     1459  4                ! attributes are involved, give up and resort to general
;  1213     1460  4                ! cursor positioning sequence.
;  1214     1461  4                ! It will cost us too much to select-graphic-rendition
;  1215     1462  4                ! and undo select graphic rendition.
;  1216     1463  4                !-
;  1217     1464  4
;  1218     1465  4                IF .NO_RETYPES NEQ 0  AND
;  1219     1466  4                   CH$COMPARE (0, 0,    ! len, addr
;  1220     1467  4                               .NO_RETYPES, CURR_ATTR[.INDEX],
;  1221     1468  4                               0        ! fill
;  1222     1469  4                              ) NEQ 0
;  1223     1470  4                THEN
;  1224     1471  4                    RETURN SET_CURSOR(PBCB,.DESIRED_LINE_NO,.DESIRED_COL_NO,.LINE_NO);
;  1225     1472  4
;  1226     1473  4                CH$FILL ( TAB, .NO_TABS, TRIAL_STRING [.TS_LEN]);
;  1227     1474  4                TS_LEN = .TS_LEN + .NO_TABS;
;  1228     1475  4                CH$MOVE ( .NO_RETYPES, CURR_TEXT [.INDEX],
;  1229     1476  4                          TRIAL_STRING [.TS_LEN]);
;  1230     1477  4                TS_LEN = .TS_LEN + .NO_RETYPES;
;  1231     1478  3                END;    ! <TAB's> to tab-stop before, retypes.
;  1232     1479  3
;  1233     1480  4            [6]:BEGIN   ! <TAB's> to tab-stop after, <BS's>.
;  1234     1481  4                CH$FILL ( TAB, .NO_TABS + 1, TRIAL_STRING [.TS_LEN]);
;  1235     1482  4                TS_LEN = .TS_LEN + .NO_TABS + 1;
;  1236     1483  4                CH$FILL ( BS, .NO_BS, TRIAL_STRING [.TS_LEN]);
;  1237     1484  4                TS_LEN = .TS_LEN + .NO_BS;
```

```
;  1238     1485  3                    END;    ! <TAB's> to tab-stop after, <BS's>.
;  1239     1486  3            TES;
;  1240     1487  2            END;    ! Column adjustment
;  1241     1488  2
;  1242     1489  2    !+
;  1243     1490  2    ! At this point in the code we have a proper sequence of characters to
;  1244     1491  2    ! reposition the cursor from .LINE_NO/.COL_NO to .DESIRED_LINE_NO/
;  1245     1492  2    ! .DESIRED_COL_NO with a relatively minimum number of characters.
;  1246     1493  2    ! This sequence of characters is contained in TRIAL_STRING and its
;  1247     1494  2    ! length is contained in .TS_LEN
;  1248     1495  2    ! We output this string to the screen.
;  1249     1496  2    !-
;  1250     1497  2
;  1251     1498  2    $OUTPUT_STRING(.TS_LEN,TRIAL_STRING,0);
;  1252     1499  2
;  1253     1500  2    RETURN SS$_NORMAL
;  1254     1501  2
;  1255     1502  1    END;            ! End of routine SMG$$FIND_MIN_CURSOR_POS


                                        0FFC 00000          .ENTRY  SMG$$FIND_MIN_CURSOR_POS, Save R2,R3,R4,R5,-; 1005
                                                                    R6,R7,R8,R9,R10,R11
                              5E      FEE8   CE   9E 00002          MOVAB   -280(SP), SP                                ;
                                        04   AC   DD 00007          PUSHL   P_PBCB                                      ; 1059
                        50    6E              08   C1 0000A          ADDL3   #8, (SP), R0                                ; 1060
                              5B              60   D0 0000E          MOVL    (R0), R11                                   ;
                                        08   AC   D5 00011          TSTL    LINE_NO                                     ; 1091
                                             7D   13 00014          BEQL    3$                                          ;
                                        0C   AC   D5 00016          TSTL    COL_NO                                      ; 1092
                                             78   13 00019          BEQL    3$                                          ;
                                             57   D4 0001B          CLRL    TS_LEN                                      ; 1112
                        50                6E 000000FC   8F   C1 0001D          ADDL3   #252, (SP), R0                      ; 1120
                                             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                   ; 1121
```

```
                                10   AC        08   AC  D1 00085          CMPL    LINE_NO, DESIRED_LINE_NO                          ; 1127
                                                    65  13 0008A          BEQL    9$                                                ;
                                08   AC        10   AC  D1 0008C          CMPL    DESIRED_LINE_NO, LINE_NO                          ; 1130
                                                    03  18 00091          BGEQ    4$                                                ;
                                                 023E   31 00093 3$:       BRW     39$                                               ;
                           56   10   AC        08   AC  C3 00096 4$:      SUBL3   LINE_NO, DESIRED_LINE_NO, LINES_DOWN              ; 1164
                                                    51  D4 0009C          CLRL    WIDE_WARNING                                      ; 1171
                                                2C  BB  95 0009E          TSTB    @44(R11)                                          ; 1172
                                                    17  13 000A1          BEQL    7$                                                ;
                           50   08   AC              01  C3 000A3          SUBL3   #1, LINE_NO, L                                    ; 1174
                                                    0B  11 000A8          BRB     6$                                                ;
                                                2C BB40 95 000AA 5$:      TSTB    @44(R11)[L]                                       ; 1175
                                                    05  13 000AE          BEQL    6$                                                ;
                                51                  01  D0 000B0          MOVL    #1, WIDE_WARNING                                  ; 1177
                                                    05  11 000B3          BRB     7$                                                ; 1176
                           F0   50        10   AC  F3 000B5 6$:          AOBLEQ  DESIRED_LINE_NO, L, 5$                            ; 1175
                                08   AE             56  D1 000BA 7$:      CMPL    LINES_DOWN, SET_CUR_LEN                           ; 1181
                                                    D3  18 000BE          BGEQ    3$                                                ;
                           50        56        08   AC  C1 000C0          ADDL3   LINE_NO, LINES_DOWN, R0                           ; 1182
                           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                                    ; 1183
                08   AC    60        10              00  ED 000DC          CMPZV   #0, #16, (R0), LINE_NO                            ;
                                                    AF  1E 000E2          BGEQU   3$                                                ;
                                AC                  51  E8 000E4 8$:      BLBS    WIDE_WARNING, 3$                                  ; 1184
                      56        0A              6E      00  2C 000E7          MOVC5   #0, (SP), #10, LINES_DOWN, TRIAL_STRING           ; 1191
                                                1C  AE      000EC                                                                     ;
                                57              56  D0 000EE              MOVL    LINES_DOWN, TS_LEN                                ; 1192
                                54        0C    AC  D0 000F1 9$:          MOVL    COL_NO, R4                                        ; 1211
                                     14   AC             54  D1 000F5          CMPL    R4, DESIRED_COL_NO                                ;
                                                    03  12 000F9          BNEQ    10$                                               ;
                                                 0218   31 000FB          BRW     46$                                               ;
                     14   AE   14   AC                01  C3 000FE 10$:     SUBL3   #1, DESIRED_COL_NO, DCN_QUAD                      ; 1251
                                                18  AE  D4 00104          CLRL    DCN_QUAD+4                                        ; 1252
                59        56   14   AE                08  7B 00107          EDIV    #8, DCN_QUAD, NO_TABS, NO_RETYPES                 ; 1253
                          50   6E                     0C  C1 0010D          ADDL3   #12, (SP), R0                                     ; 1261
                          0C   60                     03  E0 00111          BBS     #3, (R0), 11$                                     ;
                          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$:     MOVZWL  #1000, NO_TABS                                    ; 1262
                               50        10   AC       D0 00126 12$:     MOVL    DESIRED_LINE_NO, R0                               ; 1271
                                        2C BB40         95 0012A          TSTB    @44(R11)[R0]                                      ;
                                                    0A  13 0012E          BEQL    13$                                               ;
                               51        06   AB       3C 00130          MOVZWL  6(R11), R1                                        ; 1272
                          52   51                     02  C7 00134          DIVL3   #2, R1, ADJUSTED_WIDTH                            ;
                                                    07  11 00138          BRB     14$                                               ;
                               51        06   AB       3C 0013A 13$:     MOVZWL  6(R11), R1                                        ; 1273
                               52                   51  D0 0013E          MOVL    R1, ADJUSTED_WIDTH                                ;
                          50   56                   03  78 00141 14$:     ASHL    #3, NO_TABS, R0                                   ; 1275
                               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                             ; 1276
                                                    05  11 00151          BRB     16$                                               ;
                               58        03E8         8F  3C 00153 15$:     MOVZWL  #1000, NO_BS                                      ; 1277
                          50   6E 000000D1           8F  C1 00158 16$:     ADDL3   #209, (SP), R0                                    ; 1283
```

```
                         05          60 E8 00160        BLBS    (R0), 17$                                        ;
                         58    03E8  8F 3C 00163        MOVZWL  #1000, NO_BS                                     ; 1284
            52     10    AC          01 C3 00168 17$:   SUBL3   #1, DESIRED_LINE_NO, R2                          ; 1291
                         52          51 C4 0016D        MULL2   R1, R2                                           ;
                         5A        6246 7E 00170        MOVAQ   (R2)[NO_TABS], INDEX                             ;
                         54      14  AC D1 00174        CMPL    DESIRED_COL_NO, R4                               ; 1293
                         39          14 00178           BGTR    21$                                              ;
            51           6E 000000D1  8F C1 0017A       ADDL3   #209, (SP), R1                                   ; 1305
                         07          61 E9 00182        BLBC    (R1), 18$                                        ;
            50           54      14  AC C3 00185        SUBL3   DESIRED_COL_NO, R4, S1_COST                      ; 1307
                         05          11 0018A           BRB     19$                                              ;
                         50    03E8  8F 3C 0018C 18$:   MOVZWL  #1000, S1_COST                                   ; 1309
                         51 01 A946  9E 00191 19$:      MOVAB   1(NO_RETYPES)[NO_TABS], S2_COST                  ; 1314
                         55 02 A846  9E 00196           MOVAB   2(NO_BS)[NO_TABS], S3_COST                       ; 1319
                         53          01 D0 0019B        MOVL    #1, BEST_STRAT                                   ; 1323
                         50          51 D1 0019E        CMPL    S2_COST, LEAST_COST                              ; 1325
                         06          18 001A1           BGEQ    20$                                              ;
                         53          02 D0 001A3        MOVL    #2, BEST_STRAT                                   ; 1326
                         50          51 D0 001A6        MOVL    S2_COST, LEAST_COST                              ;
                         50          55 D1 001A9 20$:   CMPL    S3_COST, LEAST_COST                              ; 1328
                         6D          18 001AC           BGEQ    26$                                              ;
                         53          03 D0 001AE        MOVL    #3, BEST_STRAT                                   ; 1329
                         65          11 001B1           BRB     25$                                              ;
     04     AE     14    AC          54 C3 001B3 21$:   SUBL3   R4, DESIRED_COL_NO, S4_COST                      ; 1340
            50           56          03 78 001B9        ASHL    #3, NO_TABS, R0                                  ; 1342
                                     50 D6 001BD        INCL    R0                                               ;
            54           50             D1 001BF        CMPL    R0, R4                                           ;
                         30             15 001C2        BLEQ    22$                                              ;
            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                                    ; 1343
            1C           60          03 E0 001D4        BBS     #3, (R0), 22$                                    ;
                   0C    AE      FF  A4 9E 001D8        MOVAB   -1(R4), COL_QUAD                                 ; 1351
                                 10  AE D4 001DD        CLRL    COL_QUAD+4                                       ; 1352
 55         51     0C    AE          08 7B 001E0        EDIV    #8, COL_QUAD, NEW_NO_TABS, NEW_NO_RETYPES        ; 1353
                         56          51 C2 001E6        SUBL2   NEW_NO_TABS, NO_TABS                             ; 1354
            51           56          59 C1 001E9        ADDL3   NO_RETYPES, NO_TABS, S5_COST                     ; 1357
                         55 01 A846  9E 001ED           MOVAB   1(NO_BS)[NO_TABS], S6_COST                       ; 1361
                         0A          11 001F2           BRB     23$                                              ; 1342
                         51    03E8  8F 3C 001F4 22$:   MOVZWL  #1000, S5_COST                                   ; 1365
                         55    03E8  8F 3C 001F9        MOVZWL  #1000, S6_COST                                   ; 1366
                         53          04 D0 001FE 23$:   MOVL    #4, BEST_STRAT                                   ; 1371
                         50      04  AE D0 00201        MOVL    S4_COST, LEAST_COST                              ;
                         50          51 D1 00205        CMPL    S5_COST, LEAST_COST                              ; 1373
                         06          18 00208           BGEQ    24$                                              ;
                         53          05 D0 0020A        MOVL    #5, BEST_STRAT                                   ; 1374
                         50          51 D0 0020D        MOVL    S5_COST, LEAST_COST                              ;
                         50          55 D1 00210 24$:   CMPL    S6_COST, LEAST_COST                              ; 1376
                         06          18 00213           BGEQ    26$                                              ;
                         53          06 D0 00215        MOVL    #6, BEST_STRAT                                   ; 1377
                         50          55 D0 00218 25$:   MOVL    S6_COST, LEAST_COST                              ;
                         50          57 C0 0021B 26$:   ADDL2   TS_LEN, R0                                       ; 1380
                   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)                      ; 1390
                         05          01 53 CF 0022A     CASEL   BEST_STRAT, #1, #5                               ; 1386
 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$:    SUBL3        DESIRED_COL_NO, R4, NO_BS                 ; 1389
 58       08        6E           00 2C 0023F         MOVC5        #0, (SP), #8, NO_BS, @8(SP)               ; 1390
                                 08 BE   00244                                                              ;
                                 00CA 31 00246       BRW          45$                                       ; 1391
                                    59 D5 00249 29$: TSTL         NO_RETYPES                                ; 1404
                                    18 13 0024B      BEQL         32$                                       ;
                    54              01 D0 0024D      MOVL         #1, R4                                    ; 1406
 59       00 00000000 9F            00 2D 00250      CMPC5        #0, @#^X00000000, #0, NO_RETYPES, @24(R11)- ;
                                 18 BB4A  00259                   [INDEX]                                   ;
                                    03 1A 0025C      BGTRU        30$                                       ;
                    54              01 D9 0025E      SBWC         #1, R4                                    ;
                                    54 D5 00261 30$: TSTL         R4                                        ; 1408
                                    6F 12 00263 31$: BNEQ         39$                                       ;
               08   BE              0D 90 00265 32$: MOVB         #13, @8(SP)                               ; 1412
                                    57 D6 00269      INCL         TS_LEN                                    ; 1413
 56       09        6E              00 2C 0026B      MOVC5        #0, (SP), #9, NO_TABS, TRIAL_STRING[TS_LEN] ; 1414
                                 1C AE47  00270                                                             ;
                                    76 11 00273      BRB          41$                                       ; 1415
               08   BE              0D 90 00275 33$: MOVB         #13, @8(SP)                               ; 1422
                                    57 D6 00279      INCL         TS_LEN                                    ; 1423
                    50          01  A6 9E 0027B      MOVAB        1(R6), R0                                 ; 1424
 50       09        6E              00 2C 0027F      MOVC5        #0, (SP), #9, R0, TRIAL_STRING[TS_LEN]    ;
                                 1C AE47  00284                                                             ;
                                    7D 11 00287      BRB          44$                                       ; 1425
          59    14  AC              54 C3 00289 34$: SUBL3        R4, DESIRED_COL_NO, NO_RETYPES            ; 1440
                    5A         FF A442 9E 0028E      MOVAB        -1(R4)[R2], INDEX                         ; 1441
                                    59 D5 00293      TSTL         NO_RETYPES                                ; 1442
                                    18 13 00295      BEQL         36$                                       ;
                    54              01 D0 00297      MOVL         #1, R4                                    ; 1444
 59       00 00000000 9F            00 2D 0029A      CMPC5        #0, @#^X00000000, #0, NO_RETYPES, @24(R11)- ;
                                 18 BB4A  002A3                   [INDEX]                                   ;
                                    03 1A 002A6      BGTRU        35$                                       ;
                    54              01 D9 002A8      SBWC         #1, R4                                    ;
                                    54 D5 002AB 35$: TSTL         R4                                        ; 1446
                                    25 12 002AD      BNEQ         39$                                       ;
       08   BE   14 BB4A            59 28 002AF 36$: MOVC3        NO_RETYPES, @20(R11)[INDEX], @8(SP)       ; 1451
                                    3E 11 002B6      BRB          42$                                       ; 1452
                                    59 D5 002B8 37$: TSTL         NO_RETYPES                                ; 1465
                                    28 13 002BA      BEQL         40$                                       ;
                    54              01 D0 002BC      MOVL         #1, R4                                    ; 1467
 59       00 00000000 9F            00 2D 002BF      CMPC5        #0, @#^X00000000, #0, NO_RETYPES, @24(R11)- ;
                                 18 BB4A  002C8                   [INDEX]                                   ;
                                    03 1A 002CB      BGTRU        38$                                       ;
                    54              01 D9 002CD      SBWC         #1, R4                                    ;
                                    54 D5 002D0 38$: TSTL         R4                                        ; 1469
                                    10 13 002D2      BEQL         40$                                       ;
                                 08 AC DD 002D4 39$: PUSHL        LINE_NO                                   ; 1471
                    7E           10 AC 7D 002D7      MOVQ         DESIRED_LINE_NO, -(SP)                    ;
                                 0C AE DD 002DB      PUSHL        12(SP)                                    ;
              0000V CF              04 FB 002DE      CALLS        #4, SET_CURSOR                            ;
                                       04 002E3      RET                                                    ;
 56       09        6E              00 2C 002E4 40$: MOVC5        #0, (SP), #9, NO_TABS, @8(SP)             ; 1473
```

```
                                          08     BE          002E9                                                      ;
                                57               56     C0  002EB 41$:    ADDL2   NO_TABS, TS_LEN                       ; 1474
        1C AE47          14 BB4A               59     28  002EE           MOVC3   NO_RETYPES, @20(R11)[INDEX], TRIAL_STRING- ; 1476
                                                                                  [TS_LEN]                              ;
                                57               59     C0  002F6 42$:    ADDL2   NO_RETYPES, TS_LEN                    ; 1477
                                                 1B     11  002F9           BRB     46$                                   ; 1386
                                50        01     A6     9E  002FB 43$:    MOVAB   1(R6), R0                             ; 1481
  50             09             6E               00     2C  002FF           MOVC5   #0, (SP), #9, R0, @8(SP)              ;
                                          08     BE          00304                                                      ;
                                57        01  A647     9E  00306 44$:    MOVAB   1(NO_TABS)[TS_LEN], TS_LEN            ; 1482
  58             08             6E               00     2C  0030B           MOVC5   #0, (SP), #8, NO_BS, TRIAL_STRING[TS_LEN] ; 1483
                                        1C AE47              00310                                                      ;
                                57               58     C0  00313 45$:    ADDL2   NO_BS, TS_LEN                         ; 1484
                                                 7E     7C  00316 46$:    CLRQ    -(SP)                                 ; 1498
                                                 7E     D4  00318           CLRL    -(SP)                                 ;
                                          28     AE     9F  0031A           PUSHAB  TRIAL_STRING                          ;
                                                 57     DD  0031D           PUSHL   TS_LEN                                ;
                                          14     AE     DD  0031F           PUSHL   20(SP)                                ;
                      00000000G  00               06     FB  00322           CALLS   #6, SMG$$PUT_SCREEN                   ;
                                 03               50     E9  00329           BLBC    STATUS, 47$                           ;
                                 50               01     D0  0032C           MOVL    #1, R0                                ; 1500
                                                         04  0032F 47$:    RET                                           ; 1502

; Routine Size:  816 bytes,    Routine Base:  _SMG$CODE + 03E2
```

```
;  1257     1503  1 %SBTTL 'SET_CURSOR - Generate set-cursor sequence'
;  1258     1504  1 ROUTINE SET_CURSOR (
;  1259     1505  1                      P_PBCB,
;  1260     1506  1                      DESIRED_LINE_NO,
;  1261     1507  1                      DESIRED_COL_NO,
;  1262     1508  1                      CURRENT_ROW
;  1263     1509  1                     ) =
;  1264     1510  1 !++
;  1265     1511  1 ! FUNCTIONAL DESCRIPTION:
;  1266     1512  1 !
;  1267     1513  1 ! Routine SET_CURSOR constructs the general set cursor
;  1268     1514  1 ! sequence to position to .DESIRED_LINE_NO/.DESIRED_COL_NO and outputs
;  1269     1515  1 ! it to the screen.
;  1270     1516  1 !
;  1271     1517  1 ! CALLING SEQUENCE:
;  1272     1518  1 !
;  1273     1519  1 !       ret_status.wlc.v = SET_CURSOR ( P_PBCB.rab.r,
;  1274     1520  1 !                                       DESIRED_LINE_NO.rl.v,
;  1275     1521  1 !                                       DESIRED_COL_NO.rl.v,
;  1276     1522  1 !                                       CURRENT_ROW.rl.v)
;  1277     1523  1 !
;  1278     1524  1 ! FORMAL PARAMETERS:
;  1279     1525  1 !
;  1280     1526  1 !       P_PBCB.rab.r            Address of PBCB
;  1281     1527  1 !
;  1282     1528  1 !       DESIRED_LINE_NO.rl.v    Desired cursor line number position
;  1283     1529  1 !
;  1284     1530  1 !       DESIRED_COL_NO.rl.v     Desired cursor column number position
;  1285     1531  1 !
;  1286     1532  1 !       CURRENT_ROW.rl.v        Current row (0 means unknown)
;  1287     1533  1 !                               This matters if we are on a wide row.
;  1288     1534  1 !
;  1289     1535  1 ! IMPLICIT INPUTS:
;  1290     1536  1 !
;  1291     1537  1 !       NONE
;  1292     1538  1 !
;  1293     1539  1 ! IMPLICIT OUTPUTS:
;  1294     1540  1 !
;  1295     1541  1 !       NONE
;  1296     1542  1 !
;  1297     1543  1 ! COMPLETION STATUS:
;  1298     1544  1 !
;  1299     1545  1 !       SS$_NORMAL      Normal successful completion
;  1300     1546  1 !                       errors from SMG$$OUTPUT
;  1301     1547  1 !
;  1302     1548  1 ! SIDE EFFECTS:
;  1303     1549  1 !
;  1304     1550  1 !       NONE
;  1305     1551  1 !--
```

```
;  1307     1552  2 BEGIN
;  1308     1553  2
;  1309     1554  2 BIND
;  1310     1555  2
;  1311     1556  2         PBCB            = .P_PBCB                       : BLOCK[,BYTE],
;  1312     1557  2         WCB             = .PBCB[PBCB_A_WCB]             : BLOCK[,BYTE],
;  1313     1558  2         LCV             = .WCB[WCB_A_SCR_LINE_CHAR]     : VECTOR[,BYTE];
;  1314     1559  2
;  1315     1560  2 LOCAL
;  1316     1561  2
;  1317     1562  2         STATUS;         ! local status
```

```
;  1319     1563  2 !+
;  1320     1564  2 ! If we are currently on a double wide or high row (or if the
;  1321     1565  2 ! possibility exists) then because of bugs in the VT100 hardware,
;  1322     1566  2 ! we first position to column 1 of the desired line.
;  1323     1567  2 !-
;  1324     1568  2
;  1325     1569  2 IF (.CURRENT_ROW EQL 0 AND .LCV[0] NEQ 0)
;  1326     1570  2 OR .LCV[.CURRENT_ROW] NEQ 0
;  1327     1571  2   THEN  BEGIN            ! Move to beginning of desired line
;  1328     1572  3
;  1329     1573  3         $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.DESIRED_LINE_NO,1);
;  1330     1574  3
;  1331     1575  3         !+
;  1332     1576  3         ! Output the escape sequence.
;  1333     1577  3         !-
;  1334     1578  3
;  1335     1579  3         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  1336     1580  4           THEN  BEGIN
;  1337     1581  4                 STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  1338     1582  4                                           .PBCB[PBCB_A_CAP_BUFFER]);
;  1339     1583  4                 IF NOT .STATUS THEN RETURN .STATUS
;  1340     1584  3                 END;
;  1341     1585  3
;  1342     1586  2         END;            ! Move to beginning of desired line
;  1343     1587  2
;  1344     1588  2 !+
;  1345     1589  2 ! Create the appropriate escape sequence.
;  1346     1590  2 !-
;  1347     1591  2
;  1348     1592  2 $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.DESIRED_LINE_NO,.DESIRED_COL_NO);
;  1349     1593  2
;  1350     1594  2 !+
;  1351     1595  2 ! Output the escape sequence.
;  1352     1596  2 !-
;  1353     1597  2
;  1354     1598  2 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  1355     1599  3   THEN  BEGIN
;  1356     1600  3         STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  1357     1601  3                                 .PBCB[PBCB_A_CAP_BUFFER]);
;  1358     1602  3         IF NOT .STATUS THEN RETURN .STATUS
;  1359     1603  2         END;
;  1360     1604  2
;  1361     1605  2 RETURN SS$_NORMAL
;  1362     1606  2
;  1363     1607  1 END;    ! Routine SET_CURSOR


                                 007C 00000 SET_CURSOR:
                                                .WORD   Save R2,R3,R4,R5,R6                     ; 1504
                      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                      ; 1556
                      50       08    A2  D0 00017       MOVL    8(R2), R0                       ; 1557
```

```
                                10   AC   D5 0001B          TSTL    CURRENT_ROW                                     ; 1569
                                     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$:     ADDL3   CURRENT_ROW, 48(R0), R0                         ; 1570
                                     60   95 0002B          TSTB    (R0)                                            ;
                                     56   13 0002D          BEQL    5$                                              ;
                                00FC C2   D5 0002F 2$:      TSTL    252(R2)                                         ; 1573
                                     09   12 00033          BNEQ    3$                                              ;
                                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$:      TSTL    (R3)                                            ; 1579
                                     11   13   00072         BEQL    5$                                              ;
                                0104 C2   DD   00074         PUSHL   260(R2)                                         ; 1582
                                     63   DD   00078         PUSHL   (R3)                                            ; 1581
                                     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$                                      ; 1583
                                     00FC C2   D5   00085 5$:      TSTL    252(R2)                                         ; 1592
                                     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)                                            ; 1598
                                     15   13   000C4         BEQL    9$                                              ;
                                0104 C2   DD   000C6         PUSHL   260(R2)                                         ; 1601
                                     63   DD   000CA         PUSHL   (R3)                                            ; 1600
                                     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$                      ; 1602
                    50          54  D0 000D7 8$:          MOVL    STATUS, R0                      ;
                                    04 000DA              RET                                     ;
                    50          01  D0 000DB 9$:          MOVL    #1, R0                          ; 1605
                                    04 000DE 10$:         RET                                     ; 1607

; Routine Size:  223 bytes,    Routine Base:  _SMG$CODE + 0712
```

```
;  1365     1608  1 %SBTTL 'ERASE_LINE - Erase to end-of-line'
;  1366     1609  1 ROUTINE ERASE_LINE(P_PBCB) =
;  1367     1610  1
;  1368     1611  1 !++
;  1369     1612  1 ! FUNCTIONAL DESCRIPTION:
;  1370     1613  1 !
;  1371     1614  1 ! Outputs an erase-to-end-of-line sequence to the screen.
;  1372     1615  1 !
;  1373     1616  1 ! CALLING SEQUENCE:
;  1374     1617  1 !
;  1375     1618  1 !       ret_status.wlc.v = ERASE_LINE ( P_PBCB.rab.r)
;  1376     1619  1 !
;  1377     1620  1 ! FORMAL PARAMETERS:
;  1378     1621  1 !
;  1379     1622  1 !       P_PBCB.rab.r            Address of PBCB
;  1380     1623  1 !
;  1381     1624  1 ! IMPLICIT INPUTS:
;  1382     1625  1 !
;  1383     1626  1 !       NONE
;  1384     1627  1 !
;  1385     1628  1 ! IMPLICIT OUTPUTS:
;  1386     1629  1 !
;  1387     1630  1 !       NONE
;  1388     1631  1 !
;  1389     1632  1 ! COMPLETION STATUS:
;  1390     1633  1 !
;  1391     1634  1 !       SS$_NORMAL      Normal successful completion
;  1392     1635  1 !                       errors from SMG$$OUTPUT
;  1393     1636  1 !
;  1394     1637  1 ! SIDE EFFECTS:
;  1395     1638  1 !
;  1396     1639  1 !       NONE
;  1397     1640  1 !--
```

```
;  1399     1641  2 BEGIN
;  1400     1642  2
;  1401     1643  2 BIND
;  1402     1644  2
;  1403     1645  2         PBCB            = .P_PBCB               : BLOCK[,BYTE];
;  1404     1646  2
;  1405     1647  2 LOCAL
;  1406     1648  2
;  1407     1649  2         STATUS;         ! local status
```

```
;  1409          1650  2 !+
;  1410          1651  2 ! Create the appropriate escape sequence.
;  1411          1652  2 !-
;  1412          1653  2
;  1413          1654  2 $SMG$GET_TERM_DATA(ERASE_TO_END_LINE);
;  1414          1655  2
;  1415          1656  2 !+
;  1416          1657  2 ! Output the escape sequence.
;  1417          1658  2 !-
;  1418          1659  2
;  1419          1660  2 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  1420          1661  3   THEN  BEGIN
;  1421          1662  3         STATUS=SMG$$OUTPUT(PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  1422          1663  3                            .PBCB[PBCB_A_CAP_BUFFER]);
;  1423          1664  3         IF NOT .STATUS THEN RETURN .STATUS;
;  1424          1665  2         END;
;  1425          1666  2
;  1426          1667  2 RETURN SS$_NORMAL
;  1427          1668  2
;  1428          1669  1 END;    ! Routine ERASE_LINE


                                000C 00000 ERASE_LINE:
                                           .WORD   Save R2,R3                                  ; 1609
                             5E      10 C2 00002           SUBL2   #16, SP                     ;
                             52      04 AC D0 00005        MOVL    P_PBCB, R2                  ; 1645
                                00FC C2 D5 00009           TSTL    252(R2)                     ; 1654
                                   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$:      CLRL    INPUT_ARGS                  ;
                                   04 AE 9F 0001B          PUSHAB  INPUT_ARGS                  ;
                                 0104 C2 DD 0001E          PUSHL   260(R2)                     ;
                                53 0108 C2 9E 00022        MOVAB   264(R2), R3                 ;
                                   53 DD 00027             PUSHL   R3                          ;
                                 0100 C2 9F 00029          PUSHAB  256(R2)                     ;
                             10  AE 01D9 8F 3C 0002D       MOVZWL  #473, 16(SP)                ;
                                   10 AE 9F 00033          PUSHAB  16(SP)                      ;
                                 00FC C2 9F 00036          PUSHAB  252(R2)                     ;
                 00000000G   00      06 FB 0003A           CALLS   #6, SMG$GET_TERM_DATA       ;
                             19      50 E9 00041           BLBC    STATUS, 4$                  ;
                                   63 D5 00044 2$:         TSTL    (R3)                        ; 1660
                                   12 13 00046             BEQL    3$                          ;
                                 0104 C2 DD 00048          PUSHL   260(R2)                     ; 1663
                                   63 DD 0004C             PUSHL   (R3)                        ; 1662
                                   52 DD 0004E             PUSHL   R2                          ;
                 00000000G   00      03 FB 00050           CALLS   #3, SMG$$OUTPUT             ;
                             03      50 E9 00057           BLBC    STATUS, 4$                  ; 1664
                             50      01 D0 0005A 3$:       MOVL    #1, R0                      ; 1667
                                        04 0005D 4$:       RET                                 ; 1669

; Routine Size:  94 bytes,    Routine Base:  _SMG$CODE + 07F1
```

```
; 1430          1670  1 END
; 1431          1671  0 ELUDOM



;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _SMG$CODE                      2127  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                                 Library Statistics
;
;                                          --------- Symbols ---------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped       Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1         9776         9         0         581       00:01.0
;  _$255$DUA28:[SMGRTL.OBJ]RTLLIB.L32;1        36         0         0           8       00:00.1
;  _$255$DUA28:[SMGRTL.OBJ]SMGLIB.L32;1       469        46         9          38       00:00.5



;                                       COMMAND QUALIFIERS

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

; Size:          2127 code + 0 data bytes
; Run Time:        00:46.7
; Elapsed Time:    02:19.9
; Lines/CPU Min:     2145
; Lexemes/CPU-Min: 15513
; Memory Used:  293 pages
; Compilation Complete
```

SMGMINUPD
LIS

SMGMIN
LIS

SMGMAPTRM
LIS