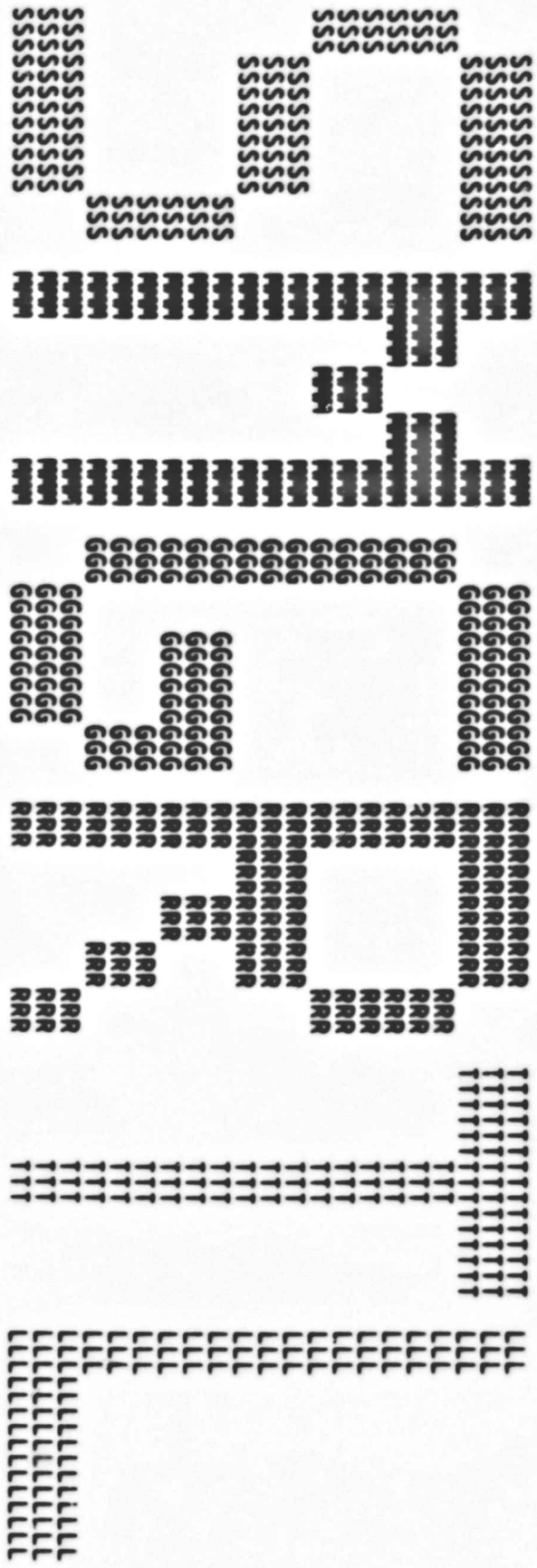
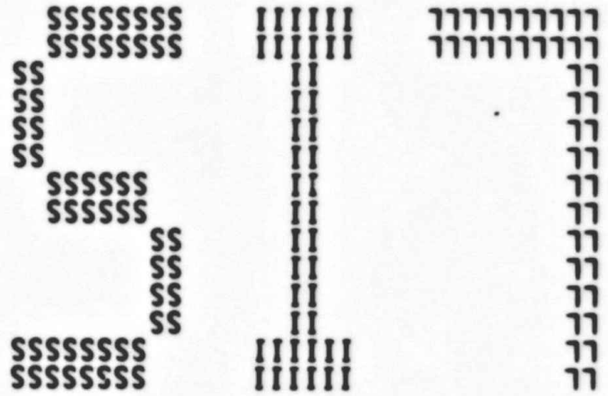
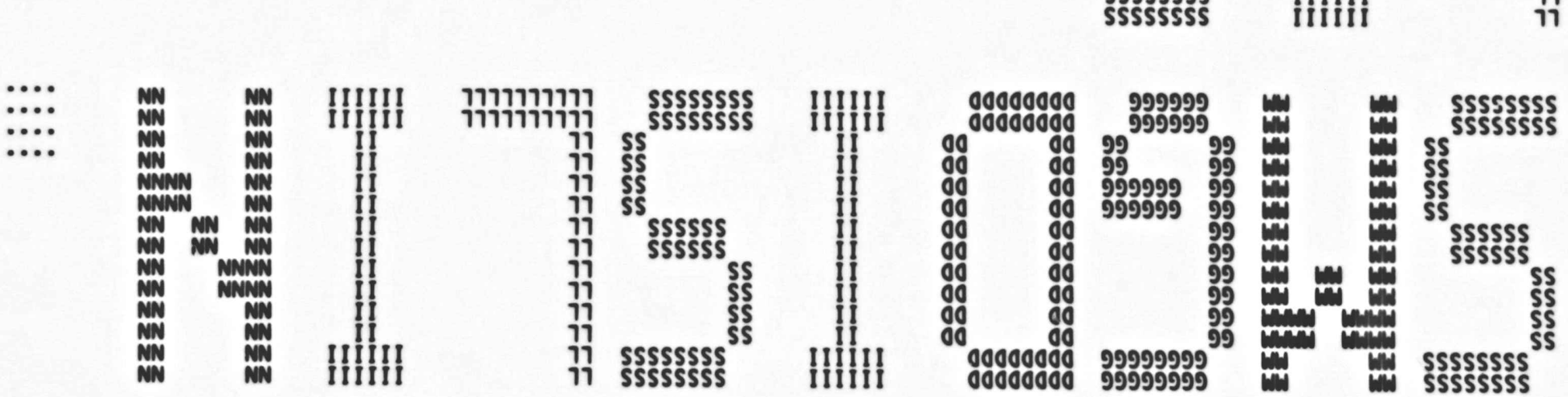


k 7





```
;     1          0001  0 %TITLE 'SMG$DISPLAY_LINKS - Virtual Display Linkages'
;     2          0002  0 MODULE SMG$DISPLAY_LINKS (
;001 PL1097      0003  0                 IDENT = '1-097' ! File: SMGDISLIN.B32 Edit: PL1097
;     4-1        0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                      *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:     Screen Management
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !       The procedures in this module are conerned only with the
;    36          0036  1 ! allocation/deallocation of virtual displays, and with the pasting/
;    37          0037  1 ! unpasting of these virtual displays to pasteboards.  The are not
;    38          0038  1 ! concerned with their contents or output.
;    39          0039  1 !
;    40          0040  1 !       For the procedures which maintain and update the contents of
;    41          0041  1 ! virtual displays, see the module SMG$DISPLAY_CHANGE.
;    42          0042  1 !
;    43          0043  1 !       For the procedures which actually do output from these virtual
;    44          0044  1 ! displays, see the module SMG$DISPLAY_OUTPUT.
;    45          0045  1 !
;    46          0046  1 !       For procedures that support input operations, see the module
;    47          0047  1 ! SMG$DISPLAY_INPUT.
;    48          0048  1 !
;    49          0049  1 ! ENVIRONMENT:  User mode, Shared library routines.
;    50          0050  1 !
;    51          0051  1 ! AUTHOR: R. Reichert, CREATION DATE: 26-Jan-1983
;    52          0052  1 !
;    53          0053  1 ! MODIFIED BY:
;    54          0054  1 !
;001 PL1097      0055  1 ! 1-097 - SMG$RESTORE_PHYSICAL_SCREEN should set the hardware scrolling
;002 PL1097      0056  1 !         region to something reasonable, like the whole screen, rather
;003 PL1097      0057  1 !         than leaving it in an unknown state.  PLL 10-Oct-1984
```

```
;    55          0058  1 ! 1-096 - Don't allow paste or unpaste if display is batched.
;    56          0059  1 !           STAN 27-Jun-1984.
;    57          0060  1 ! 1-095 - Use symbolic names SMG$K_TOP, etc. in SMG$LABEL_BORDER.
;    58          0061  1 !           Change error messages by SMG$SET_DISPLAY_SCROLLING_REGION.
;    59          0062  1 !           STAN 3-Jun-1984.
;    60          0063  1 ! 1-094 - Fix bug re borders occluding other borders, STAN 7-May-1984.
;    61          0064  1 ! 1-001 - Original.  Skeleton for future code.  RKR 26-Jan-1983
;    62          0065  1 !--
```

```
;    64          0066  1 %SBTTL 'Declarations'
;    65          0067  1 !
;    66          0068  1 ! SWITCHES:
;    67          0069  1 !
;    68          0070  1
;    69          0071  1 !
;    70          0072  1 ! LINKAGES:
;    71          0073  1 !
;    72          0074  1 !       NONE
;    73          0075  1 !
;    74          0076  1 ! INCLUDE FILES
;    75          0077  1 !
;    76          0078  1
;    77          0079  1 REQUIRE 'RTLIN:SMGPROLOG';              ! defines psects, macros, tcb,
;    78          0157  1                                         !  wcb, & terminal symbols
;    79          0158  1
;    80          0159  1 REQUIRE 'RTLIN:STRLNK';                 ! Linkages to string JSB
;    81          0344  1
;    82          0345  1 !
;    83          0346  1 ! TABLE OF CONTENTS:
;    84          0347  1 !
;    85          0348  1
;    86          0349  1 FORWARD ROUTINE
;    87          0350  1
;    88          0351  1 ! Public entry points
;    89          0352  1
;    90          0353  1     SMG$CHANGE_PBD_CHARACTERISTICS,     ! Change characteristics of
;    91          0354  1                                         ! physical terminal
;    92          0355  1
;    93          0356  1     SMG$CHANGE_VIRTUAL_DISPLAY,         ! Change characteristics of
;    94          0357  1                                         ! existing virtual display
;    95          0358  1
;    96          0359  1     SMG$CHECK_FOR_OCCLUSION,            ! Check to see if a virtual
;    97          0360  1                                         ! display is occluded.
;    98          0361  1
;    99          0362  1     SMG$CREATE_PASTEBOARD,              ! Create pasteboard
;   100          0363  1
;   101          0364  1     SMG$CREATE_VIRTUAL_DISPLAY,         ! Create virtual display
;   102          0365  1
;   103          0366  1     SMG$DELETE_PASTEBOARD,              ! Get rid of pasteboard, terminate
;   104          0367  1                                         ! all operations on this display
;   105          0368  1
;   106          0369  1     SMG$DELETE_VIRTUAL_DISPLAY,         ! Delete virtual display
;   107          0370  1
;   108          0371  1     SMG$GET_DISPLAY_ATTR,               ! Return current attributes of
;   109          0372  1                                         ! virtual display
;   110          0373  1
;   111          0374  1     SMG$LABEL_BORDER,                   ! Supply label for border
;   112          0375  1
;   113          0376  1     SMG$MOVE_VIRTUAL_DISPLAY,           ! Move position of virtual
;   114          0377  1                                         ! display on pasteboard
;   115          0378  1
;   116          0379  1     SMG$PASTE_VIRTUAL_DISPLAY,          ! Paste virtual display to
;   117          0380  1                                         ! pasteboard
;   118          0381  1
;   119          0382  1     SMG$POP_VIRTUAL_DISPLAY,            ! Pop off (and delete) all
;   120          0383  1                                         ! virtual displays from given
```

```
;   121     0384  1                                   ! to top of pasting stack.
;   122     0385  1
;   123     0386  1     SMG$REPASTE_VIRTUAL_DISPLAY,    ! Repaste virtual display to
;   124     0387  1                                     ! pasteboard in new position
;   125     0388  1
;   126     0389  1     SMG$RESTORE_PHYSICAL_SCREEN,    ! Restore screen to where it
;   127     0390  1                                     ! was after non-SMG user
;   128     0391  1                                     ! munged it.
;   129     0392  1
;   130     0393  1     SMG$SAVE_PHYSICAL_SCREEN,       ! Save physical screen before
;   131     0394  1                                     ! non-SMG user mungs its up.
;   132     0395  1
;   133     0396  1     SMG$SET_DISPLAY_SCROLL_REGION,  ! Set the scrolling region in
;   134     0397  1                                     ! a virtual display
;   135     0398  1
;   136     0399  1     SMG$UNPASTE_VIRTUAL_DISPLAY,    ! Unpaste virtual display from
;   137     0400  1                                     ! pasteboard.
;   138     0401  1
;   139     0402  1 ! Private entry points
;   140     0403  1
;   141     0404  1     SMG$$CALC_PASTE_TRANSF,         ! Calculate pasting
;   142     0405  1                                     ! transformation constants.
;   143     0406  1
;   144     0407  1     SMG$$CHECK_OCCLUSION,           ! Check current complement of
;   145     0408  1                                     ! pasted virtual displays to
;   146     0409  1                                     ! see who is occluded.
;   147     0410  1
;   148     0411  1     SMG$$CHECK_OCCLUSION_FIRST,     ! Check occlusion caused by
;   149     0412  1                                     ! highest pasted virtual display.
;   150     0413  1
;   151     0414  1     SMG$$CREATE_PASTEBOARD,         ! Create pasteboard
;   152     0415  1
;   153     0416  1     SMG$$CREATE_VIRTUAL_DISPLAY,    ! Inner-most Create Virtual
;   154     0417  1                                     ! Display routine
;   155     0418  1
;   156     0419  1     SMG$$CREATE_WCB,                ! Create WCB and its buffers
;   157     0420  1
;   158     0421  1     SMG$$DEALLOCATE_WCB,            ! Get rid of WCB and its buffers.
;   159     0422  1
;   160     0423  1     SMG$$DUPL_VIRTUAL_DISPLAY,      ! Duplicate a virtual display
;   161     0424  1
;   162     0425  1     SMG$$LOCATE_PP,                 ! Locate PP which matches a
;   163     0426  1                                     ! DCB and a PBCB.
;   164     0427  1
;   165     0428  1     SMG$$PASTE_VIRTUAL_DISPLAY,     ! Inner-most Paste Virtual
;   166     0429  1                                     ! Display routine.
;   167     0430  1
;   168     0431  1     SMG$$RECALC_PP_FIELDS,          ! Recalculate pasting packet
;   169     0432  1                                     ! fields after virtual display
;   170     0433  1                                     ! batching ceases.
;   171     0434  1     SMG$$UNPASTE_VIRTUAL_DISPLAY;   ! Inner-most Unpaste Virtual
;   172     0435  1                                     ! Display routine.
;   173     0436  1
;   174     0437  1
;   175     0438  1                                     ! routines.
;   176     0439  1
;   177     0440  1 !
```

```
;  178     0441  1 ! EXTERNAL REFERENCES
;  179     0442  1 !
;  180     0443  1 EXTERNAL ROUTINE
;  181     0444  1     LIB$ANALYZE_SDESC_R2 : LIB$ANALYZE_SDESC_JSB_LINK,
;  182     0445  1                                    ! Get length and address of a string
;  183     0446  1
;  184     0447  1     LIB$FREE_VM,                   ! Deallocate heap storage
;  185     0448  1
;  186     0449  1     LIB$FREE_EF,                   ! Free an event flag
;  187     0450  1
;  188     0451  1     LIB$GET_EF,                    ! Get an event flag
;  189     0452  1
;  190     0453  1     LIB$GET_VM,                    ! Allocate heap storage
;  191     0454  1
;  192     0455  1     LIB$SCOPY_DXDX,                ! String copy by descriptor
;  193     0456  1
;  194     0457  1     LIB$SFREE1_DD,                 ! Free a dynamic string
;  195     0458  1
;  196     0459  1     SMG$$BEGIN_PASTEBOARD_UPDATE_R1 : SMG$$BEGIN_PBD_UPDATE$LNK,
;  197     0460  1                                    ! Increase buffering level by 1
;  198     0461  1
;  199     0462  1     SMG$$END_PASTEBOARD_UPDATE_R2   : SMG$$END_PBD_UPDATE$LNK,
;  200     0463  1                                    ! Decrease buffering level by 1
;  201     0464  1
;  202     0465  1     SMG$$ERASE_PASTEBOARD,         ! Erase the physical screen
;  203     0466  1
;  204     0467  1     SMG$$CHECK_FOR_OUTPUT_DCB,     ! Force output if now is the time
;  205     0468  1
;  206     0469  1     SMG$$CHECK_FOR_OUTPUT_PBCB,    ! Force output
;  207     0470  1
;  208     0471  1     SMG$$FILL_WINDOW_BUFFER,       ! Move stuff from virt. display to
;  209     0472  1                                    ! pasteboard buffer and output.
;  210     0473  1
;  211     0474  1     SMG$$FIND_MIN_CURSOR_POS,      ! Set cursor on physical screen
;  212     0475  1     SMG$$FLUSH_BUFFER,             ! Flush output buffer
;  213     0476  1
;  214     0477  1     SMG$$FORCE_SCROLL_REG,         ! Force scrolling region on screen.
;  215     0478  1
;  216     0479  1     SMG$$OUTPUT,                   ! Output a string to terminal
;  217     0480  1
;  218     0481  1     SMG$$OCCLUDE,                  ! Check for how two rectangular areas
;  219     0482  1                                    ! overlap.
;  220     0483  1
;  221     0484  1     SMG$$PBCB_EXIT_HANDLER,        ! Output exit handler
;  222     0485  1
;  223     0486  1     SMG$$SETUP_TERMINAL_TYPE;      ! Get device characteristics
;  224     0487  1
;  225     0488  1 EXTERNAL LITERAL
;  226     0489  1
;  227     0490  1     LIB$_EF_ALRFRE,        ! Event flag already free
;  228     0491  1     SMG$_BATWAS_ON,        ! Batching was enabled
;  229     0492  1     SMG$_FATERRLIB,        ! Fatal error in library
;  230     0493  1     SMG$_INVARG,           ! Invalid argument
;  231     0494  1     SMG$_ILLBATFNC,        ! Operation not legal to batched display
;  232     0495  1     SMG$_INVDIS_ID,        ! Invalid virtual display id
;  233     0496  1     SMG$_INVPAS_ID,        ! Invalid pasteboard id
;  234     0497  1     SMG$_INVROW,           ! Invalid row
```

```
;  235          0498  1     SMG$_NOTPASTED,     ! Given display is not pasted to given
;  236          0499  1                         ! pasteboard
;  237          0500  1     SMG$_PASALREXI,     ! Pasteboard already exists for this device
;  238          0501  1     SMG$_TOOMANDIS,     ! Too many virtual displays requested
;  239          0502  1     SMG$_TOOMANPAS,     ! Too many pasteboards requested
;  240          0503  1     SMG$_WRONUMARG;     ! Wrong number of arguments
```

```
;  242          0504  1 !+
;  243          0505  1 ! Pasteboard Directory (PBD)
;  244          0506  1 ! -------------------------
;  245          0507  1 ! This data structure resides in OWN storage.  It is the primary vehicle
;  246          0508  1 ! for getting from a pasteboard id to the associated pasteboard control
;  247          0509  1 ! block.
;  248          0510  1 !-
;  249          0511  1
;  250          0512  1 GLOBAL
;  251          0513  1     PBD_L_COUNT : INITIAL (0),  ! No. of pasteboards we currently know
;  252          0514  1                                ! about.
;  253          0515  1
;  254          0516  1     PBD_A_PBCB : VECTOR  [PBD_K_MAX_PB, LONG]
;  255          0517  1                  INITIAL (REP PBD_K_MAX_PB OF (0)),
;  256          0518  1                          ! List of pasteboard addresses. Indexed by
;  257          0519  1                          ! pasteboard id (PID) to find address of
;  258          0520  1                          ! corresponding PBCB.
;  259          0521  1
;  260          0522  1     PBD_V_PB_AVAIL : BITVECTOR [PBD_K_MAX_PB]
;  261          0523  1                      INITIAL ( BYTE (REP ((PBD_K_MAX_PB+7)/8) OF (0)));
;  262          0524  1                          ! This is a bit-vector of pasteboard id's
;  263          0525  1                          ! still available.  The next available number
;  264          0526  1                          ! is found by doing a FFC instruction to find
;  265          0527  1                          ! first bit which is a 0.  The bit position so
;  266          0528  1                          ! computed is the next available PID.  The
;  267          0529  1                          ! bit found is set to 1 to mark it as in use.
;  268          0530  1                          ! (Presumably, a check has already been made to
;  269          0531  1                          ! insure that PBD_L_COUNT is LSS PBD_K_MAX_PB.)
;  270          0532  1
;  271          0533  1 ! Some constants needed by reference for FFC instruction
;  272          0534  1 OWN
;  273          0535  1     ZERO                : INITIAL ( 0 ),
;  274          0536  1     PBD_K_MAX_PB_BY_REF : INITIAL ( PBD_K_MAX_PB );
```

```
;  276     0537  1 %SBTTL 'SMG$CHANGE_PBD_CHARACTERISTICS'
;  277     0538  1 GLOBAL ROUTINE SMG$CHANGE_PBD_CHARACTERISTICS
;  278     0539  1         (PBID,
;  279     0540  1          P_DESIRED_WIDTH,
;  280     0541  1          P_RESULTING_WIDTH,
;  281     0542  1          P_DESIRED_HEIGHT,
;  282     0543  1          P_RESULTING_HEIGHT,
;  283     0544  1          P_DESIRED_BACKGROUND_COLOR,
;  284     0545  1          P_RESULTING_BACKGROUND_COLOR
;  285     0546  1         )=
;  286     0547  1
;  287     0548  1 !++
;  288     0549  1 ! FUNCTIONAL DESCRIPTION:
;  289     0550  1 !
;  290     0551  1 !       This routine lets you change the physical dimensions
;  291     0552  1 !       of a pasteboard.  It also lets you change the background color.
;  292     0553  1 !
;  293     0554  1 ! CALLING SEQUENCE:
;  294     0555  1 !
;  295     0556  1 !       ret_status.wlc.v = SMG$CHANGE_PBD_CHARACTERISTICS
;  296     0557  1 !                                       (PBID.rl.r
;  297     0558  1 !                                       [,DESIRED_WIDTH.rl.r]
;  298     0559  1 !                                       [,RESULTING_WIDTH.wl.r]
;  299     0560  1 !                                       [,DESIRED_HEIGHT.rl.r]
;  300     0561  1 !                                       [,RESULTING_HEIGHT.wl.r]
;  301     0562  1 !                                       [,DESIRED_BACKGROUND_COLOR.rl.r]
;  302     0563  1 !                                       [,RESULTING_BACKGROUND_COLOR.wl.r]
;  303     0564  1 !                                       )
;  304     0565  1 !
;  305     0566  1 ! FORMAL PARAMETERS:
;  306     0567  1 !
;  307     0568  1 !       PBID.rl.r               Pasteboard id of pasteboard.
;  308     0569  1 !
;  309     0570  1 !       DESIRED_WIDTH.rl.r      New width desired for pasteboard.
;  310     0571  1 !                               If omitted, the width is not changed.
;  311     0572  1 !
;  312     0573  1 !       RESULTING_WIDTH.wl.r    Physical width that resulted.  This may
;  313     0574  1 !                               be larger than the width requested if the
;  314     0575  1 !                               terminal width couldn't be set exactly to
;  315     0576  1 !                               the desired width.  This may be smaller
;  316     0577  1 !                               than the width requested if the terminal
;  317     0578  1 !                               width couldn't be set that wide.
;  318     0579  1 !                               In this case, the terminal was set to
;  319     0580  1 !                               it's maximum width.
;  320     0581  1 !
;  321     0582  1 !                               Example:        (for VT100)
;  322     0583  1 !
;  323     0584  1 !                               Width Desired   Width resulting
;  324     0585  1 !
;  325     0586  1 !                               60              80
;  326     0587  1 !                               110             132
;  327     0588  1 !                               150             132
;  328     0589  1 !
;  329     0590  1 !                               If desired width was omitted, this
;  330     0591  1 !                               argument receives the current pasteboard
;  331     0592  1 !                               width.
;  332     0593  1 !                               To find out what the pasteboard width is
```

```
;  333     0594  1 !                                (as opposed to the terminal width),
;  334     0595  1 !                                the caller should take the minimum
;  335     0596  1 !                                of his desired width and the resulting width.
;  336     0597  1 !
;  337     0598  1 !       DESIRED_HEIGHT.rl.r      New height desired for pasteboard.
;  338     0599  1 !                                If omitted, the height is not changed.
;  339     0600  1 !
;  340     0601  1 !       RESULTING_HEIGHT.wl.r    Physical height that resulted.  This may
;  341     0602  1 !                                be larger than the height requested if the
;  342     0603  1 !                                terminal height couldn't be set exactly to
;  343     0604  1 !                                the desired height.  This may be smaller
;  344     0605  1 !                                than the height requested if the terminal
;  345     0606  1 !                                height couldn't be set that high.
;  346     0607  1 !                                In this case, the terminal was set to
;  347     0608  1 !                                it's maximum height.
;  348     0609  1 !
;  349     0610  1 !                                Example:        (for VT100)
;  350     0611  1 !
;  351     0612  1 !                                Height Desired  Height resulting
;  352     0613  1 !
;  353     0614  1 !                                15              24
;  354     0615  1 !
;  355     0616  1 !                                35              24
;  356     0617  1 !
;  357     0618  1 !                                To find out what the pasteboard height is
;  358     0619  1 !                                (as opposed to the terminal height),
;  359     0620  1 !                                the caller should take the minimum
;  360     0621  1 !                                of his desired height and the resulting height.
;  361     0622  1 !
;  362     0623  1 !       DESIRED_BACKGROUND_COLOR.rl.r   Symbolic name for the background
;  363     0624  1 !                                color wanted.  For example,
;  364     0625  1 !                                SMG$C_COLOR_WHITE.  These symbols
;  365     0626  1 !                                are defined in SMGDEF.SDL.
;  366     0627  1 !                                If omitted, the background color
;  367     0628  1 !                                is not changed.
;  368     0629  1 !
;  369     0630  1 !       RESULTING_BACKGROUND_COLOR.wl.r Receives the actual background color
;  370     0631  1 !                                that was chosen.  If the terminal
;  371     0632  1 !                                does not support the exact color
;  372     0633  1 !                                desired, the nearest approximation
;  373     0634  1 !                                will be chosen.  This is determined
;  374     0635  1 !                                by comparing the frequency of the
;  375     0636  1 !                                desired light wave against the
;  376     0637  1 !                                available frequencies.  For more
;  377     0638  1 !                                information about colorimetry,
;  378     0639  1 !                                consult National Bureau of Standards
;  379     0640  1 !                                Circular 553, The ISCC-NBS method of
;  380     0641  1 !                                designating Colors.
;  381     0642  1 !
;  382     0643  1 !                                Example: (VT100)
;  383     0644  1 !
;  384     0645  1 !                                Color desired   Resulting Color
;  385     0646  1 !
;  386     0647  1 !                                yellowish pink  white
;  387     0648  1 !
;  388     0649  1 !                                navy blue       black
;  389     0650  1 !
```

```
;  390     0651  1 !                                   If the desired color is omitted,
;  391     0652  1 !                                   the value of this variable is not
;  392     0653  1 !                                   affected.
;  393     0654  1 !
;  394     0655  1 ! IMPLICIT INPUTS:
;  395     0656  1 !
;  396     0657  1 !       NONE
;  397     0658  1 !
;  398     0659  1 ! IMPLICIT OUTPUTS:
;  399     0660  1 !
;  400     0661  1 !       NONE
;  401     0662  1 !
;  402     0663  1 ! COMPLETION STATUS:
;  403     0664  1 !
;  404     0665  1 !       SS$_NORMAL      Normal successful completion
;  405     0666  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  406     0667  1 !       SMG$_PBDIN_USE  Can't change characteristics while buffering is on
;  407     0668  1 !       SMG$_INVWIDARG  Invalid width of 0 desired
;  408     0669  1 !       SMG$_INVPAGARG  Invalid height of 0 desired
;  409     0670  1 !       SMG$_INVCOLARG  Unknown background color specified
;  410     0671  1 !
;  411     0672  1 ! SIDE EFFECTS:
;  412     0673  1 !
;  413     0674  1 !       Physical width and background color of terminal may change.
;  414     0675  1 !--
```

```
;  416          0676  2 BEGIN
;  417          0677  2
;  418          0678  2 EXTERNAL ROUTINE
;  419          0679  2
;  420          0680  2         SMG$$CHECK_FOR_OUTPUT_PBCB,
;  421          0681  2         SMG$$CALC_PASTE_TRANSF,
;  422          0682  2         SMG$$CREATE_WCB,
;  423          0683  2         SMG$$DEALLOCATE_WCB,
;  424          0684  2         SMG$$ERASE_PASTEBOARD,
;  425          0685  2         SMG$$OUTPUT;
;  426          0686  2
;  427          0687  2 EXTERNAL LITERAL
;  428          0688  2
;  429          0689  2         SMG$_INVWIDARG,          ! width=0
;  430          0690  2         SMG$_INVPAGARG,          ! HEIGHT=0
;  431          0691  2         SMG$_INVCOLARG,          ! unknown color
;  432          0692  2         SMG$_PBDIN_USE;          ! pasteboard was batched
;  433          0693  2
;  434          0694  2 BUILTIN
;  435          0695  2
;  436          0696  2         NULLPARAMETER;
;  437          0697  2
;  438          0698  2 LOCAL
;  439          0699  2
;  440          0700  2         STATUS,                          ! Status of subroutine calls
;  441          0701  2         PASTING_PACKET_PANIC,            ! TRUE if we must adjust pasting packets
;  442          0702  2         CURR_PP : REF $PP_DECL,          ! Pasting packet pointer
;  443          0703  2
;  444          0704  2         PBCB    : REF $PBCB_DECL;        ! Address of pasteboard control block
```

```
;  446          0705  2 $SMG$VALIDATE_ARGCOUNT (1, 7);  ! Test for right no. of args
;  447          0706
;  448          0707    $SMG$GET_PBCB (.PBID,PBCB);     ! Get address of PBCB
;  449          0708
;  450          0709    PASTING_PACKET_PANIC=0;
;  451          0710
;  452          0711    !+
;  453          0712    ! If a desired width is specified, get it now.
;  454          0713    !-
;  455          0714
;  456          0715    IF NOT NULLPARAMETER(P_DESIRED_WIDTH)
;  457          0716      THEN  BEGIN   ! Change pasteboard width
;  458          0717            BIND    DESIRED_WIDTH=.P_DESIRED_WIDTH;
;  459          0718    !(a)    LOCAL   CURRENT_MAX, DESIRED_MAX;
;  460          0719            LOCAL   PREVIOUS_WIDTH;
;  461          0720            LOCAL   RESULTANT_WIDTH;
;  462          0721
;  463          0722            IF .DESIRED_WIDTH EQL 0
;  464          0723              THEN  RETURN  SMG$_INVWIDARG;
;  465          0724
;  466          0725            !+
;  467          0726            ! Determine the physical setting of the terminal by rounding
;  468          0727            ! up to 80 or 132 as necessary.  Do the same for the desired
;  469          0728            ! width.  Compare these two numbers to see if we must change
;  470          0729            ! the width.  This algorithm will have to change if we ever
;  471          0730            ! support terminals with widths other than 80 and 132.
;  472          0731            !-
;  473          0732
;  474          0733            IF .PBCB[PBCB_L_BATCH_LEVEL] NEQ 0
;  475          0734              THEN  RETURN  SMG$_PBDIN_USE;
;  476          0735    !(a)    IF .PBCB[PBCB_W_WIDTH] LEQU 80
;  477          0736    !(a)      THEN  CURRENT_MAX=80
;  478          0737    !(a)      ELSE  CURRENT_MAX=132;
;  479          0738    !(a)    IF .DESIRED_WIDTH LEQU 80
;  480          0739    !(a)      THEN  DESIRED_MAX=80
;  481          0740    !(a)      ELSE  DESIRED_MAX=132;
;  482          0741
;  483          0742    !(a)    !+
;  484          0743    !(a)    ! If the desired max is the same as the current max,
;  485          0744    !(a)    ! then no escape sequence need be sent to the terminal.
;  486          0745    !(a)    ! Just adjust our internal width in the PBCB.
;  487          0746    !(a)    !-
;  488          0747
;  489          0748    !(a)    IF .DESIRED_MAX NEQ .CURRENT_MAX
;  490          0749    !(a)      THEN
;  491          0750
;  492          0751    !+
;  493          0752    !
;  494          0753    !       Note: (a)
;  495          0754    !
;  496          0755    !       The lines marked !(a) could be added back in
;  497          0756    !       if you want to avoid outputting the escape sequence
;  498          0757    !       to change the terminal width if it isn't necessary.
;  499          0758    !       However, that will mean the screen doesn't physically
;  500          0759    !       blank and so extra code would have to be written to
;  501          0760    !       blank the right part of a screen when changing width
;  502          0761    !       (say) from 70 to 50 columns.
```

```
;  503          0762  3 !-
;  504          0763  3
;  505          0764  4                 BEGIN   ! Change physical width
;  506          0765  4
;  507          0766  4                 LOCAL
;  508          0767  4
;  509          0768  4                         NORMAL_WIDTH,
;  510          0769  4                         WIDE_WIDTH;
;  511          0770  4
;  512          0771  4                 !+
;  513          0772  4                 ! First, clear the screen.
;  514          0773  4                 !-
;  515          0774  4
;  516          0775  4                 $SMG$GET_TERM_DATA(ERASE_WHOLE_DISPLAY);
;  517          0776  4                 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  518          0777  5                   THEN  BEGIN
;  519          0778  5                         STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  520          0779  5                                              .PBCB[PBCB_A_CAP_BUFFER]);
;  521          0780  5                         IF NOT .STATUS THEN RETURN .STATUS
;  522          0781  4                         END;
;  523          0782  4
;  524          0783  4                 !+
;  525          0784  4                 ! Second, get the normal size.
;  526          0785  4                 !-
;  527          0786  4
;  528          0787  4                 $SMG$GET_TERM_DATA(COLUMNS);
;  529          0788  4                 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  530          0789  5                   THEN  BEGIN
;  531          0790  5                         BIND RESULT=.PBCB[PBCB_A_CAP_BUFFER];
;  532          0791  5                         STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  533          0792  5                                              .PBCB[PBCB_A_CAP_BUFFER]);
;  534          0793  5                         IF NOT .STATUS THEN RETURN .STATUS;
;  535          0794  5                         NORMAL_WIDTH=.RESULT
;  536          0795  5                         END
;  537          0796  4                   ELSE  NORMAL_WIDTH=80;
;  538          0797  4
;  539          0798  4                 !+
;  540          0799  4                 ! Third, get the wide size.
;  541          0800  4                 !-
;  542          0801  4
;  543          0802  4                 $SMG$GET_TERM_DATA(WIDTH_WIDE);
;  544          0803  4                 IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  545          0804  5                   THEN  BEGIN
;  546          0805  5                         BIND RESULT=.PBCB[PBCB_A_CAP_BUFFER];
;  547          0806  5                         STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  548          0807  5                                              .PBCB[PBCB_A_CAP_BUFFER]);
;  549          0808  5                         IF NOT .STATUS THEN RETURN .STATUS;
;  550          0809  5                         WIDE_WIDTH=.RESULT
;  551          0810  5                         END
;  552          0811  4                   ELSE  WIDE_WIDTH=80;
;  553          0812  4
;  554          0813  4                 !+
;  555          0814  4                 ! Decide which sequence to send.
;  556          0815  4                 !-
;  557          0816  4
;  558          0817  4                 IF .DESIRED_WIDTH LEQ .NORMAL_WIDTH
;  559          0818  5                   THEN  BEGIN
```

```
;  560     0819  5                        $SMG$GET_TERM_DATA(WIDTH_NARROW);
;  561     0820  5                        RESULTANT_WIDTH=.NORMAL_WIDTH
;  562     0821  5                        END
;  563     0822  5                    ELSE  BEGIN
;  564     0823  5                        $SMG$GET_TERM_DATA(WIDTH_WIDE);
;  565     0824  5                        RESULTANT_WIDTH=.WIDE_WIDTH
;  566     0825  4                        END;
;  567     0826  4
;  568     0827  4                    IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  569     0828  5                      THEN  BEGIN
;  570     0829  5                            STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  571     0830  5                                                 .PBCB[PBCB_A_CAP_BUFFER]);
;  572     0831  5                            IF NOT .STATUS THEN RETURN .STATUS;
;  573     0832  4                            END;
;  574     0833  4
;  575     0834  4                    !+
;  576     0835  4                    ! If we asked for something smaller than the terminal
;  577     0836  4                    ! could handle (like a width of 60 on an 80-column terminal)
;  578     0837  4                    ! then we will software simulate the smaller width.
;  579     0838  4                    !-
;  580     0839  4
;  581     0840  4                    RESULTANT_WIDTH=MINU(.RESULTANT_WIDTH,.DESIRED_WIDTH);
;  582     0841  4
;  583     0842  4                    END;    ! Change physical width
;  584     0843
;  585     0844                  !+
;  586     0845                  ! Should we go back to the old scheme whereby we
;  587     0846                  ! output the escape sequence only if the max width has
;  588     0847                  ! changed, then we need the following line:
;  589     0848                  !
;  590     0849                  !(a)  ELSE      RESULTANT_WIDTH=.DESIRED_WIDTH;
;  591     0850                  !-
;  592     0851
;  593     0852                  !+
;  594     0853                  ! Save away new pasteboard width in the PBCB.
;  595     0854                  !-
;  596     0855
;  597     0856                  PREVIOUS_WIDTH=.PBCB[PBCB_W_WIDTH];
;  598     0857                  PBCB[PBCB_W_WIDTH]=.RESULTANT_WIDTH;
;  599     0858
;  600     0859                  !+
;  601     0860                  ! If the width changed, we must recalculate all the pasting
;  602     0861                  ! packet parameters pronto. Make a note.
;  603     0862                  !-
;  604     0863
;  605     0864  !               IF .PREVIOUS_WIDTH NEQ .RESULTANT_WIDTH
;  606     0865  !                 THEN  PASTING_PACKET_PANIC=1;
;  607     0866                  PASTING_PACKET_PANIC=1;
;  608     0867
;  609     0868                  !+
;  610     0869                  ! At some point in the future, we might want to tell VMS
;  611     0870                  ! about this new width.  If so, we would add that code here.
;  612     0871                  ! There is probably no need to do that since we will restore
;  613     0872                  ! the original width when we delete this pasteboard.
;  614     0873                  !-
;  615     0874
;  616     0875  2               END;    ! Change pasteboard width
```

```
;  617          0876  2
;  618          0877  2 !+
;  619          0878  2 ! If the user wants the pasteboard width, give it to him now.
;  620          0879  2 !-
;  621          0880  2
;  622          0881  2 IF NOT NULLPARAMETER(P_RESULTING_WIDTH)
;  623          0882  2   THEN  BEGIN    ! Return pasteboard width
;  624          0883  2         BIND     RESULTING_WIDTH = .P_RESULTING_WIDTH;
;  625          0884  2         RESULTING_WIDTH=.PBCB[PBCB_W_WIDTH]
;  626          0885  2         END;     ! Return pasteboard width
;  627          0886  2
;  628          0887  2 !+
;  629          0888  2 ! If the user wants to change his height, do that now.
;  630          0889  2 ! If he specifies an illegal height, that's his problem;
;  631          0890  2 ! we don't know what sort of funny terminal he might have.
;  632          0891  2 ! This code will have to change if we ever support terminals
;  633          0892  2 ! that can change height by sending them escape sequences.
;  634          0893  2 !-
;  635          0894  2
;  636          0895  2 IF NOT NULLPARAMETER(P_DESIRED_HEIGHT)
;  637          0896  2   THEN  BEGIN    ! Change pasteboard height
;  638          0897  2         BIND     DESIRED_HEIGHT = .P_DESIRED_HEIGHT;
;  639          0898  2         IF .PBCB[PBCB_L_BATCH_LEVEL] NEQ 0
;  640          0899  2           THEN  RETURN  SMG$_PBDIN_USE;
;  641          0900  2         IF .DESIRED_HEIGHT EQL 0
;  642          0901  2           THEN  RETURN  SMG$_INVPAGARG;
;  643          0902  2         IF .PBCB[PBCB_B_ROWS] NEQ .DESIRED_HEIGHT
;  644          0903  4           THEN  BEGIN
;  645          0904  4                 !+
;  646          0905  4                 ! Blank screen if we are making screen smaller,
;  647          0906  4                 ! so as to get rid of items after the bottom of
;  648          0907  4                 ! the pasteboard.
;  649          0908  4                 !-
;  650          0909  4                 IF MINU(24,.DESIRED_HEIGHT) LSSU .PBCB[PBCB_B_ROWS]
;  651          0910  5                   THEN  BEGIN
;  652          0911  5                         STATUS=SMG$$ERASE_PASTEBOARD(.PBCB);
;  653          0912  5                         IF NOT .STATUS THEN RETURN .STATUS
;  654          0913  4                         END;
;  655          0914  4                 !+
;  656          0915  4                 ! All existing terminals have a maximum height of 24.
;  657          0916  4                 !-
;  658          0917  4                 PBCB[PBCB_B_ROWS]=MINU(24,.DESIRED_HEIGHT);
;  659          0918  4                 PASTING_PACKET_PANIC=1
;  660          0919  4                 END
;  661          0920  2         END;     ! Change pasteboard height
;  662          0921  2
;  663          0922  2 !+
;  664          0923  2 ! If the user wants the pasteboard height, give it to him now.
;  665          0924  2 !-
;  666          0925  2
;  667          0926  2 IF NOT NULLPARAMETER(P_RESULTING_HEIGHT)
;  668          0927  2   THEN  BEGIN    ! Return pasteboard height
;  669          0928  2         BIND     RESULTING_HEIGHT=.P_RESULTING_HEIGHT;
;  670          0929  2         RESULTING_HEIGHT=.PBCB[PBCB_B_ROWS]
;  671          0930  2         END;     ! Return pasteboard height
;  672          0931  2
;  673          0932  2 !+
```

```
;  674          0933  2 ! If we changed either the width or height of the pasteboard,
;  675          0934  2 ! then we must go adjust all the pasting packets now.
;  676          0935  2 ! We must also reallocate and reshape the buffers in the WCB.
;  677          0936  2 !-
;  678          0937  2
;  679          0938  2 IF .PASTING_PACKET_PANIC
;  680          0939  3   THEN  BEGIN   ! Update all pasting packets
;  681          0940  3         LOCAL CURR_PP    : REF $PP_DECL;
;  682          0941  3         !+
;  683          0942  3         ! Deallocate the old WCB.
;  684          0943  3         !-
;  685          0944  3         STATUS=SMG$$DEALLOCATE_WCB(.PBCB[PBCB_A_WCB]);
;  686          0945  3         IF NOT .STATUS THEN RETURN .STATUS;
;  687          0946  3         !+
;  688          0947  3         ! Allocate a new WCB.
;  689          0948  3         !-
;  690          0949  3         STATUS=SMG$$CREATE_WCB( %REF(.PBCB[PBCB_B_ROWS]),
;  691          0950  3                                 %REF(.PBCB[PBCB_W_WIDTH]),
;  692          0951  3                                 PBCB[PBCB_A_WCB]);
;  693          0952  3         IF NOT .STATUS THEN RETURN .STATUS;
;  694          0953  3         !+
;  695          0954  3         ! Walk chain of DCB's for all displays currently pasted
;  696          0955  3         ! to this pasteboard, and go update their pasting packet.
;  697          0956  3         ! Start with first packet.
;  698          0957  3         !-
;  699          0958  3         CURR_PP=.PBCB[PBCB_A_PP_NEXT];
;  700          0959  3         WHILE .CURR_PP NEQ PBCB[PBCB_A_PP_NEXT] DO
;  701          0960  4                 BEGIN   ! Update a pasting packet
;  702          0961  4                 LOCAL
;  703          0962  4                     PP_BASE     : REF $PP_DECL;          ! Base addr of this PP
;  704          0963  4
;  705          0964  4                 PP_BASE = .CURR_PP - PP_PBCB_QUEUE_OFFSET;  ! Since queue header
;  706          0965  4                                                             ! not at top of
;  707          0966  4                                                             ! structure.
;  708          0967  4                 STATUS=SMG$$CALC_PASTE_TRANSF(.PP_BASE);
;  709          0968  4                 IF NOT .STATUS THEN RETURN .STATUS;
;  710          0969  4                 CURR_PP = .PP_BASE [PP_A_NEXT_PBCB] ! Step to next PP
;  711          0970  3                 END;    ! Update a pasting packet
;  712          0971  3
;  713          0972  3         !+
;  714          0973  3         ! Force an update.
;  715          0974  3         !-
;  716          0975  3
;  717          0976  3         STATUS=SMG$$CHECK_FOR_OUTPUT_PBCB(.PBCB);
;  718          0977  3         IF NOT .STATUS THEN SIGNAL(.STATUS);
;  719          0978  3
;  720          0979  3         END;    ! Update all pasting packets
;  721          0980  3
;  722          0981  3 !+
;  723          0982  3 ! If a new background color is desired, go do that now.
;  724          0983  3 !-
;  725          0984  3
;  726          0985  3 IF NOT NULLPARAMETER(P_DESIRED_BACKGROUND_COLOR)
;  727          0986  3   THEN  BEGIN   ! Change background color
;  728          0987  3         BIND    DESIRED_COLOR=.P_DESIRED_BACKGROUND_COLOR;
;  729          0988  3         BIND    RESULTING_COLOR=PBCB[PBCB_B_BACKGROUND_COLOR];
;  730          0989  3
```

```
;   731          0990  3          IF .DESIRED_COLOR EQL SMG$C_COLOR_WHITE
;   732          0991  4            THEN  $SMG$GET_TERM_DATA(LIGHT_SCREEN)
;   733          0992  4            ELSE  $SMG$GET_TERM_DATA(DARK_SCREEN);
;   734          0993  4
;   735          0994  4          IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;   736          0995  4            THEN  BEGIN
;   737          0996  4                  STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;   738          0997  4                                       .PBCB[PBCB_A_CAP_BUFFER]);
;   739          0998  4                  IF NOT .STATUS THEN RETURN .STATUS
;   740          0999  4                  END;
;   741          1000  4
;   742          1001  4          RESULTING_COLOR=.DESIRED_COLOR;
;   743          1002  4
;   744          1003  4          END;    ! Change terminal coror
;   745          1004  2
;   746          1005  2  !+
;   747          1006  2  ! If the user wants the new background color, give it to him now.
;   748          1007  2  !-
;   749          1008  2
;   750          1009  2  IF NOT NULLPARAMETER(P_RESULTING_BACKGROUND_COLOR)
;   751          1010  3    THEN  BEGIN    ! Return background color
;   752          1011  3          BIND     RESULTING_COLOR=.P_RESULTING_BACKGROUND_COLOR;
;   753          1012  3          RESULTING_COLOR=.PBCB[PBCB_B_BACKGROUND_COLOR]
;   754          1013  3          END;    ! Return background color
;   755          1014  2
;   756          1015  2  RETURN  SS$_NORMAL
;   757          1016  2
;   758          1017  1  END;                    ! Routine SMG$CHANGE_PBD_CHARACTERISTICS


;                                                .TITLE  SMG$DISPLAY_LINKS SMG$DISPLAY_LINKS - Virtual D
                                                         isplay Linkages
                                                .IDENT  \1-097\

                                                .PSECT  _SMG$DATA,NOEXE,  PIC,2

                                    00000000  00000 PBD_L_COUNT::
                                                    .LONG   0                                           ;
                                    00000000# 00004 PBD_A_PBCB::
                                                    .LONG   0[16]                                       ;
                                          00# 00044 PBD_V_PB_AVAIL::
                                                    .BYTE   0[2]                                        ;
                                              00046         .BLKB   2
                                    00000000  00048 ZERO:   .LONG   0                                   ;
                                    00000010  0004C PBD_K_MAX_PB_BY_REF:
                                                    .LONG   16                                          ;

                                                .EXTRN  LIB$ANALYZE_SDESC_R2
                                                .EXTRN  LIB$FREE_VM, LIB$FREE_EF
                                                .EXTRN  LIB$GET_EF, LIB$GET_VM
                                                .EXTRN  LIB$SCOPY_DXDX, LIB$SFREE1_DD
                                                .EXTRN  SMG$$BEGIN_PASTEBOARD_UPDATE_R1
                                                .EXTRN  SMG$$END_PASTEBOARD_UPDATE_R2
                                                .EXTRN  SMG$$ERASE_PASTEBOARD
                                                .EXTRN  SMG$$CHECK_FOR_OUTPUT_DCB
                                                .EXTRN  SMG$$CHECK_FOR_OUTPUT_PBCB
                                                .EXTRN  SMG$$FILL_WINDOW_BUFFER
```

```
                                        .EXTRN  SMG$$FIND_MIN_CURSOR_POS
                                        .EXTRN  SMG$$FLUSH_BUFFER
                                        .EXTRN  SMG$$FORCE_SCROLL_REG
                                        .EXTRN  SMG$$OUTPUT, SMG$$OCCLUDE
                                        .EXTRN  SMG$$PBCB_EXIT_HANDLER
                                        .EXTRN  SMG$$SETUP_TERMINAL_TYPE
                                        .EXTRN  LIB$_EF_ALRFRE, SMG$_BATWAS_ON
                                        .EXTRN  SMG$_FATERRLIB, SMG$_INVARG
                                        .EXTRN  SMG$_ILLBATFNC, SMG$_INVDIS_ID
                                        .EXTRN  SMG$_INVPAS_ID, SMG$_INVROW
                                        .EXTRN  SMG$_NOTPASTED, SMG$_PASALREXI
                                        .EXTRN  SMG$_TOOMANDIS, SMG$_TOOMANPAS
                                        .EXTRN  SMG$_WRONUMARG, SMG$$CALC_PASTE_TRANSF
                                        .EXTRN  SMG$$CREATE_WCB
                                        .EXTRN  SMG$$DEALLOCATE_WCB
                                        .EXTRN  SMG$_INVWIDARG, SMG$_INVPAGARG
                                        .EXTRN  SMG$_INVCOLARG, SMG$_PBDIN_USE
                                        .EXTRN  SMG$GET_TERM_DATA

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

                            0FFC 00000  .ENTRY  SMG$CHANGE_PBD_CHARACTERISTICS, Save R2,R3,-; 0538
                                                R4,R5,R6,R7,R8,R9,R10,R11
                 5B 00000000G   00  9E 00002  MOVAB  SMG$$OUTPUT, R11
                 5A 00000000G   00  9E 00009  MOVAB  SMG$GET_TERM_DATA, R10
                 5E             14  C2 00010  SUBL2  #20, SP
              50 6C             01  83 00013  SUBB3  #1, (AP), DIFF                          0705
                 06             50  91 00017  CMPB   DIFF, #6
                                08  18 0001A  BLEQU  1$
                 50 00000000G   8F  D0 0001C  MOVL   #SMG$_WRONUMARG, R0
                                04  00023  RET
                 50       04    BC  D0 00024 1$:  MOVL   @PBID, R0                           0707
                                11  19 00028  BLSS   2$
           00000000'  EF        50  D1 0002A  CMPL   R0, PBD_L_COUNT
                                08  14 00031  BGTR   2$
        08 00000000'  EF        50  E0 00033  BBS    R0, PBD_V_PB_AVAIL, 3$
                 50 00000000G   8F  D0 00038 2$:  MOVL   #SMG$_INVPAS_ID, R0
                                04  00042  RET
                 54 00000000'EF40  D0 00043 3$:  MOVL   PBD_A_PBCB[R0], PBCB
                                59  D4 0004B  CLRL   PASTING_PACKET_PANIC                    0709
                 02             6C  91 0004D  CMPB   (AP), #2                               0715
                                03  1E 00050  BGEQU  5$
                             0189  31 00052 4$:  BRW    33$
                          08    AC  D5 00055 5$:  TSTL   8(AP)
                                F8  13 00058  BEQL   4$
                 58       08    BC  D0 0005A  MOVL   @P_DESIRED_WIDTH, R8                   0722
                                08  12 0005E  BNEQ   6$
                 50 00000000G   8F  D0 00060  MOVL   #SMG$_INVWIDARG, R0                    0723
                                04  00067  RET
                       00A4     C4  D5 00068 6$:  TSTL   164(PBCB)                          0733
                                03  13 0006C  BEQL   7$
                             018C  31 0006E  BRW    35$
                 53       0108    C4  9E 00071 7$:  MOVAB  264(PBCB), R3                    0775
                 52       00FC    C4  9E 00076  MOVAB  252(PBCB), R2
                                62  D5 0007B  TSTL   (R2)
                                04  12 0007D  BNEQ   8$
                                63  D4 0007F  CLRL   (R3)
```

```
                          21  11 00081       BRB     9$
                  08      AE  D4 00083 8$:   CLRL    INPUT_ARGS
                  08      AE  9F 00086       PUSHAB  INPUT_ARGS
                0104      C4  DD 00089       PUSHL   260(PBCB)
                          53  DD 0008D       PUSHL   R3
                0100      C4  9F 0008F       PUSHAB  256(PBCB)
        14  AE  01DA      8F  3C 00093       MOVZWL  #474, 20(SP)
                  14      AE  9F 00099       PUSHAB  20(SP)
                          52  DD 0009C       PUSHL   R2
            6A            06  FB 0009E       CALLS   #6, SMG$GET_TERM_DATA
            3A            50  E9 000A1       BLBC    STATUS, 12$
                          63  D5 000A4 9$:   TSTL    (R3)                            : 0776
                          11  13 000A6       BEQL    10$
                0104      C4  DD 000A8       PUSHL   260(PBCB)                       : 0779
                          63  DD 000AC       PUSHL   (R3)                            : 0778
                          54  DD 000AE       PUSHL   PBCB
            6B            03  FB 000B0       CALLS   #3, SMG$$OUTPUT
            55            50  D0 000B3       MOVL    R0, STATUS
            3F            55  E9 000B6       BLBC    STATUS, 14$                     : 0780
                          62  D5 000B9 10$:  TSTL    (R2)                            : 0787
                          04  12 000BB       BNEQ    11$
                          63  D4 000BD       CLRL    (R3)
                          20  11 000BF       BRB     13$
                  08      AE  D4 000C1 11$:  CLRL    INPUT_ARGS
                  08      AE  9F 000C4       PUSHAB  INPUT_ARGS
                0104      C4  DD 000C7       PUSHL   260(PBCB)
                          53  DD 000CB       PUSHL   R3
                0100      C4  9F 000CD       PUSHAB  256(PBCB)
        14  AE    DD      8F  9A 000D1       MOVZBL  #221, 20(SP)
                  14      AE  9F 000D6       PUSHAB  20(SP)
                          52  DD 000D9       PUSHL   R2
            6A            06  FB 000DB       CALLS   #6, SMG$GET_TERM_DATA
            49            50  E9 000DE 12$:  BLBC    STATUS, 18$
                          63  D5 000E1 13$:  TSTL    (R3)                            : 0788
                          1B  13 000E3       BEQL    15$
            56  0104      C4  D0 000E5       MOVL    260(PBCB), R6                   : 0790
                0104      C4  DD 000EA       PUSHL   260(PBCB)                       : 0792
                          63  DD 000EE       PUSHL   (R3)                            : 0791
                          54  DD 000F0       PUSHL   PBCB
            6B            03  FB 000F2       CALLS   #3, SMG$$OUTPUT
            55            50  D0 000F5       MOVL    R0, STATUS
            49            55  E9 000F8 14$:  BLBC    STATUS, 20$                     : 0793
            57            66  D0 000FB       MOVL    (R6), NORMAL_WIDTH              : 0794
                          04  11 000FE       BRB     16$
            57    50      8F  9A 00100 15$:  MOVZBL  #80, NORMAL_WIDTH               : 0794
                          62  D5 00104 16$:  TSTL    (R2)                            : 0802
                          04  12 00106       BNEQ    17$
                          63  D4 00108       CLRL    (R3)
                          21  11 0010A       BRB     19$
                  08      AE  D4 0010C 17$:  CLRL    INPUT_ARGS
                  08      AE  9F 0010F       PUSHAB  INPUT_ARGS
                0104      C4  DD 00112       PUSHL   260(PBCB)
                          53  DD 00116       PUSHL   R3
                0100      C4  9F 00118       PUSHAB  256(PBCB)
        14  AE  0246      8F  3C 0011C       MOVZWL  #582, 20(SP)
                  14      AE  9F 00122       PUSHAB  20(SP)
                          52  DD 00125       PUSHL   R2
```

```
                        6A            06  FB 00127              CALLS   #6, SMG$GET_TERM_DATA                    ;
                        7C            50  E9 0012A 18$:        BLBC    STATUS, 27$                              ;
                                      63  D5 0012D 19$:        TSTL    (R3)                                     ; 0803
                                      1B  13 0012F             BEQL    21$                                      ;
                        56      0104  C4  D0 00131             MOVL    260(PBCB), R6                            ; 0805
                                0104  C4  DD 00136             PUSHL   260(PBCB)                                ; 0807
                                      63  DD 0013A             PUSHL   (R3)                                     ; 0806
                                      54  DD 0013C             PUSHL   PBCB                                     ;
                        6B            03  FB 0013E             CALLS   #3, SMG$$OUTPUT                          ;
                        55            50  D0 00141             MOVL    R0, STATUS                               ;
                        7B            55  E9 00144 20$:        BLBC    STATUS, 30$                              ; 0808
                        56            66  D0 00147             MOVL    (R6), WIDE_WIDTH                         ; 0809
                                      04  11 0014A             BRB     22$                                      ;
                        56      50    8F  9A 0014C 21$:        MOVZBL  #80, WIDE_WIDTH                          ; 0811
                        57            58  D1 00150 22$:        CMPL    R8, NORMAL_WIDTH                         ; 0817
                                      2E  14 00153             BGTR    25$                                      ;
                                      62  D5 00155             TSTL    (R2)                                     ; 0819
                                      04  12 00157             BNEQ    23$                                      ;
                                      63  D4 00159             CLRL    (R3)                                     ;
                                      21  11 0015B             BRB     24$                                      ;
                              08      AE  D4 0015D 23$:        CLRL    INPUT_ARGS                               ;
                              08      AE  9F 00160             PUSHAB  INPUT_ARGS                               ;
                          0104        C4  DD 00163             PUSHL   260(PBCB)                                ;
                                      53  DD 00167             PUSHL   R3                                       ;
                          0100        C4  9F 00169             PUSHAB  256(PBCB)                                ;
                    14    AE      0245  8F  3C 0016D             MOVZWL  #581, 20(SP)                             ;
                              14      AE  9F 00173             PUSHAB  20(SP)                                   ;
                                      52  DD 00176             PUSHL   R2                                       ;
                        6A            06  FB 00178             CALLS   #6, SMG$GET_TERM_DATA                    ;
                        2B            50  E9 0017B             BLBC    STATUS, 27$                              ;
                        52            57  D0 0017E 24$:        MOVL    NORMAL_WIDTH, RESULTANT_WIDTH            ; 0820
                                      2D  11 00181             BRB     29$                                      ;
                                      62  D5 00183 25$:        TSTL    (R2)                                     ; 0823
                                      04  12 00185             BNEQ    26$                                      ;
                                      63  D4 00187             CLRL    (R3)                                     ;
                                      22  11 00189             BRB     28$                                      ;
                              08      AE  D4 0018B 26$:        CLRL    INPUT_ARGS                               ;
                              08      AE  9F 0018E             PUSHAB  INPUT_ARGS                               ;
                          0104        C4  DD 00191             PUSHL   260(PBCB)                                ;
                                      53  DD 00195             PUSHL   R3                                       ;
                          0100        C4  9F 00197             PUSHAB  256(PBCB)                                ;
                    14    AE      0246  8F  3C 0019B             MOVZWL  #582, 20(SP)                             ;
                              14      AE  9F 001A1             PUSHAB  20(SP)                                   ;
                                      52  DD 001A4             PUSHL   R2                                       ;
                        6A            06  FB 001A6             CALLS   #6, SMG$GET_TERM_DATA                    ;
                        01            50  E8 001A9 27$:        BLBS    STATUS, 28$                              ;
                                          04 001AC             RET                                              ;
                        52            56  D0 001AD 28$:        MOVL    WIDE_WIDTH, RESULTANT_WIDTH              ; 0824
                                      63  D5 001B0 29$:        TSTL    (R3)                                     ; 0827
                                      11  13 001B2             BEQL    31$                                      ;
                                0104  C4  DD 001B4             PUSHL   260(PBCB)                                ; 0830
                                      63  DD 001B8             PUSHL   (R3)                                     ; 0829
                                      54  DD 001BA             PUSHL   PBCB                                     ;
                        6B            03  FB 001BC             CALLS   #3, SMG$$OUTPUT                          ;
                        55            50  D0 001BF             MOVL    R0, STATUS                               ;
                        72            55  E9 001C2 30$:        BLBC    STATUS, 39$                              ; 0831
                        50            52  D0 001C5 31$:        MOVL    RESULTANT_WIDTH, R0                      ; 0840
```

```
                                58                      50  D1 001C8          CMPL    R0, R8                                          ;
                                                        03  1B 001CB          BLEQU   32$                                             ;
                                50                      58  D0 001CD          MOVL    R8, R0                                          ;
                                52                      50  D0 001D0 32$:     MOVL    R0, RESULTANT_WIDTH                             ;
                                50          5A          A4  3C 001D3          MOVZWL  90(PBCB), PREVIOUS_WIDTH                        ; 0856
                             5A A4                      52  B0 001D7          MOVW    RESULTANT_WIDTH, 90(PBCB)                       ; 0857
                                59                      01  D0 001DB          MOVL    #1, PASTING_PACKET_PANIC                        ; 0866
                                03                      6C  91 001DE 33$:     CMPB    (AP), #3                                        ; 0881
                                                        0A  1F 001E1          BLSSU   34$                                             ;
                                            0C          AC  D5 001E3          TSTL    12(AP)                                          ;
                                                        05  13 001E6          BEQL    34$                                             ;
                             0C BC          5A          A4  3C 001E8          MOVZWL  90(PBCB), @P_RESULTING_WIDTH                    ; 0884
                                04                      6C  91 001ED 34$:     CMPB    (AP), #4                                        ; 0895
                                                        4F  1F 001F0          BLSSU   41$                                             ;
                                            10          AC  D5 001F2          TSTL    16(AP)                                          ;
                                                        4A  13 001F5          BEQL    41$                                             ;
                                          00A4          C4  D5 001F7          TSTL    164(PBCB)                                       ; 0898
                                                        08  13 001FB          BEQL    36$                                             ;
                                50 00000000G            8F  D0 001FD 35$:     MOVL    #SMG$_PBDIN_USE, R0                             ; 0899
                                                            04 00204          RET                                                     ;
                                52          10          BC  D0 00205 36$:     MOVL    @P_DESIRED_HEIGHT, R2                           ; 0900
                                                        08  12 00209          BNEQ    37$                                             ;
                                50 00000000G            8F  D0 00208          MOVL    #SMG$_INVPAGARG, R0                             ; 0901
                                                            04 00212          RET                                                     ;
                          52    5F    A4                08  00  ED 00213 37$:     CMPZV   #0, #8, 95(PBCB), R2                            ; 0902
                                                        26  13 00219          BEQL    41$                                             ;
                                18                      52  D1 0021B          CMPL    R2, #24                                         ; 0909
                                                        03  1B 0021E          BLEQU   38$                                             ;
                                52                      18  D0 00220          MOVL    #24, R2                                         ;
                          52    5F    A4                08  00  ED 00223 38$:     CMPZV   #0, #8, 95(PBCB), R2                            ;
                                                        0F  1B 00229          BLEQU   40$                                             ;
                                54                      DD  0022B          PUSHL   PBCB                                            ; 0911
                       00000000G    00                  01  FB 0022D          CALLS   #1, SMG$$ERASE_PASTEBOARD                       ;
                                55                      50  D0 00234          MOVL    R0, STATUS                                      ;
                                61                      55  E9 00237 39$:     BLBC    STATUS, 44$                                     ; 0912
                             5F A4                      52  90 0023A 40$:     MOVB    R2, 95(PBCB)                                    ; 0917
                                59                      01  D0 0023E          MOVL    #1, PASTING_PACKET_PANIC                        ; 0918
                                05                      6C  91 00241 41$:     CMPB    (AP), #5                                        ; 0926
                                                        0A  1F 00244          BLSSU   42$                                             ;
                                          14            AC  D5 00246          TSTL    20(AP)                                          ;
                                                        05  13 00249          BEQL    42$                                             ;
                             14 BC          5F          A4  9A 0024B          MOVZBL  95(PBCB), @P_RESULTING_HEIGHT                   ; 0929
                                6C                      59  E9 00250 42$:     BLBC    PASTING_PACKET_PANIC, 47$                       ; 0938
                                          08            A4  DD 00253          PUSHL   8(PBCB)                                         ; 0944
                       00000000G    00                  01  FB 00256          CALLS   #1, SMG$$DEALLOCATE_WCB                         ;
                                55                      50  D0 0025D          MOVL    R0, STATUS                                      ;
                                38                      55  E9 00260          BLBC    STATUS, 44$                                     ; 0945
                                          08            A4  9F 00263          PUSHAB  8(PBCB)                                         ; 0951
                             08 AE          5A          A4  3C 00266          MOVZWL  90(PBCB), 8(SP)                                 ; 0950
                                          08 AE         9F  00 26B            PUSHAB  8(SP)                                           ;
                             08 AE          5F          A4  9A 0026E          MOVZBL  95(PBCB), 8(SP)                                 ; 0949
                                          08 AE         9F  00 273            PUSHAB  8(SP)                                           ;
                       00000000G    00                  03  FB 00276          CALLS   #3, SMG$$CREATE_WCB                             ; 0951
                                55                      50  D0 0027D          MOVL    R0, STATUS                                      ;
                                18                      55  E9 00280          BLBC    STATUS, 44$                                     ; 0952
                                53                      64  D0 00283          MOVL    (PBCB), CURR_PP                                 ; 0958
                                54                      53  D1 00286 43$:     CMPL    CURR_PP, PBCB                                   ; 0959
```

```
                              1C 13 00289        BEQL    46$
               52        F8   A3 9E 0028B        MOVAB   -8(R3), PP_BASE                         ; 0964
                              52 DD 0028F        PUSHL   PP_BASE                                 ; 0967
      00000000G  00           01 FB 00291        CALLS   #1, SMG$$CALC_PASTE_TRANSF
                 55           50 D0 00298        MOVL    R0, STATUS
                 03           55 E8 0029B  44$:  BLBS    STATUS, 45$                             ; 0968
                            0092 31 0029E        BRW     53$
               53        08   A2 D0 002A1  45$:  MOVL    8(PP_BASE), CURR_PP                     ; 0969
                              DF 11 002A5        BRB     43$
                              54 DD 002A7  46$:  PUSHL   PBCB                                    ; 0976
      00000000G  00           01 FB 002A9        CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB
                 55           50 D0 002B0        MOVL    R0, STATUS
                 09           55 E8 002B3        BLBS    STATUS, 47$                             ; 0977
                              55 DD 002B6        PUSHL   STATUS
      00000000G  00           01 FB 002B8        CALLS   #1, LIB$SIGNAL
                 06           6C 91 002BF  47$:  CMPB    (AP), #6                                ; 0985
                              79 1F 002C2        BLSSU   55$
                         18   AC D5 002C4        TSTL    24(AP)
                              74 13 002C7        BEQL    55$
                 50    00FC   C4 9E 002C9        MOVAB   252(PBCB), R0                           ; 0991
                 52    0108   C4 9E 002CE        MOVAB   264(PBCB), R2
                 01      18   BC D1 002D3        CMPL    @P_DESIRED_BACKGROUND_COLOR, #1         ; 0990
                              1C 12 002D7        BNEQ    48$
                              60 D5 002D9        TSTL    (R0)                                    ; 0991
                              1C 13 002DB        BEQL    49$
                         08   AE D4 002DD        CLRL    INPUT_ARGS
                         08   AE 9F 002E0        PUSHAB  INPUT_ARGS
                       0104   C4 DD 002E3        PUSHL   260(PBCB)
                              52 DD 002E7        PUSHL   R2
                       0100   C4 9F 002E9        PUSHAB  256(PBCB)
          14     AE    0228   8F 3C 002ED        MOVZWL  #552, 20(SP)
                              1E 11 002F3        BRB     51$
                              60 D5 002F5  48$:  TSTL    (R0)                                    ; 0992
                              04 12 002F7        BNEQ    50$
                              62 D4 002F9  49$:  CLRL    (R2)
                              21 11 002FB        BRB     52$
                         08   AE D4 002FD  50$:  CLRL    INPUT_ARGS
                         08   AE 9F 00300        PUSHAB  INPUT_ARGS
                       0104   C4 DD 00303        PUSHL   260(PBCB)
                              52 DD 00307        PUSHL   R2
                       0100   C4 9F 00309        PUSHAB  256(PBCB)
          14     AE    01C8   8F 3C 0030D        MOVZWL  #456, 20(SP)
                         14   AE 9F 00313  51$:  PUSHAB  20(SP)
                              50 DD 00316        PUSHL   R0
                 6A           06 FB 00318        CALLS   #6, SMG$GET_TERM_DATA
                 32           50 E9 0031B        BLBC    STATUS, 57$
                              62 D5 0031E  52$:  TSTL    (R2)                                    ; 0994
                              15 13 00320        BEQL    54$
                       0104   C4 DD 00322        PUSHL   260(PBCB)                               ; 0997
                              62 DD 00326        PUSHL   (R2)                                    ; 0996
                              54 DD 00328        PUSHL   PBCB
                 6B           03 FB 0032A        CALLS   #3, SMG$$OUTPUT
                 55           50 D0 0032D        MOVL    R0, STATUS
                 04           55 E8 00330        BLBS    STATUS, 54$                             ; 0998
                 50           55 D0 00333  53$:  MOVL    STATUS, R0
                                 04 00336        RET
          00F9   C4      18   BC D0 00337  54$:  MOVL    @P_DESIRED_BACKGROUND_COLOR, 249(PBCB)  ; 1001
```

```
                                07              6C  91  00033D 55$:    CMPB    (AP), #7                                        ; 1009
                                                0B  1F  00340          BLSSU   56$                                             ;
                                        1C      AC  D5  00342          TSTL    28(AP)                                          ;
                                                06  13  00345          BEQL    56$                                             ;
                            1C  BC      00F9    C4  9A  00347          MOVZBL  249(PBCB), @P_RESULTING_BACKGROUND_COLOR         ; 1012
                                50              01  D0  00034D 56$:    MOVL    #1, R0                                          ; 1015
                                                    04  00350 57$:    RET                                                     ; 1017

; Routine Size:  849 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
;  760          1018  1 %SBTTL 'SMG$CHANGE_VIRTUAL_DISPLAY - Change Virtual Display'
;  761          1019  1 GLOBAL ROUTINE SMG$CHANGE_VIRTUAL_DISPLAY (
;  762          1020  1                                             DISPLAY_ID,
;  763          1021  1                                             NUM_ROWS,
;  764          1022  1                                             NUM_COLS,
;  765          1023  1                                             DISPLAY_ATTRIBUTES,
;  766          1024  1                                             VIDEO_ATTRIBUTES,
;  767          1025  1                                             CHAR_SET
;  768          1026  1                                            ) =
;  769          1027  1 !++
;  770          1028  1 ! FUNCTIONAL DESCRIPTIOM:
;  771          1029  1 !
;  772          1030  1 !       This routine changes the size or default attributes of an
;  773          1031  1 !       existing virtual display.  The text which is currently in this
;  774          1032  1 !       virtual display is remapped to fit the new dimensions starting
;  775          1033  1 !       at row 1 column 1.  Resulting cursor position will be at row 1
;  776          1034  1 !       column 1.
;  777          1035  1 !
;  778          1036  1 ! CALLING SEQUENCE:
;  779          1037  1 !
;  780          1038  1 !       ret_status.wlc.v = SMG$CHANGE_VIRTUAL_DISPLAY (
;  781          1039  1 !                                  DISPLAY_ID.rl.r,
;  782          1040  1 !                                  [,NUM_ROWS.rl.r]
;  783          1041  1 !                                  [,NUM_COLS.rl.r]
;  784          1042  1 !                                  [,DISPLAY_ATTRIBUTES.rl.r]
;  785          1043  1 !                                  [,VIDEO_ATTRIBUTES.rl.r]
;  786          1044  1 !                                  [,CHAR_SET.rl.r])
;  787          1045  1 !
;  788          1046  1 ! FORMAL PARAMETERS:
;  789          1047  1 !
;  790          1048  1 !       DISPLAY_ID.rl.r Display id of virtual display to be changed.
;  791          1049  1 !
;  792          1050  1 !       NUM_ROWS.rl.r   Number of rows in new virtual display.
;  793          1051  1 !                       If omitted, the number of rows remains the same.
;  794          1052  1 !
;  795          1053  1 !       NUM_COLS.rl.r   Number of columns in new virtual display.
;  796          1054  1 !                       If omitted, the number of columns remains the same.
;  797          1055  1 !
;  798          1056  1 !       DISPLAY_ATTRIBUTES.rl.r The default display attributes:
;  799          1057  1 !
;  800          1058  1 !                       SMG$M_BORDER if virtual display is to be
;  801          1059  1 !                                    displayed with a border.
;  802          1060  1 !
;  803          1061  1 !                       SMG$M_TRUNC_ICON if an icon should be displayed
;  804          1062  1 !                                    when text overflows the display bounds.
;  805          1063  1 !
;  806          1064  1 !                       SMG$M_DISPLAY_CONTROLS if carriage controls (CR, LF,
;  807          1065  1 !                                    FF, VT, HT) should be displayed instead
;  808          1066  1 !                                    of executed.
;  809          1067  1 !
;  810          1068  1 !                       If omitted, the default display attributes
;  811          1069  1 !                       currently associated with the display will be
;  812          1070  1 !                       retained.
;  813          1071  1 !
;  814          1072  1 !       VIDEO_ATTRIBUTES.rl.r   The default rendition code to be
;  815          1073  1 !                       applied to all output to this display unless
;  816          1074  1 !                       overridden on a particular output call.
```

```
;  817          1075  1 !                          If omitted, the current video attributes are
;  818          1076  1 !                          retained.
;  819          1077  1 !
;  820          1078  1 !                          Values:
;  821          1079  1 !
;  822          1080  1 !                          SMG$M_BLINK      displays characters blinking.
;  823          1081  1 !
;  824          1082  1 !                          SMG$M_BOLD       displays characters in
;  825          1083  1 !                                           higher-than-normal intensity.
;  826          1084  1 !
;  827          1085  1 !                          SMG$M_REVERSE    displays characters in reverse
;  828          1086  1 !                                           video -- that is, using the
;  829          1087  1 !                                           opposite default rendition of
;  830          1088  1 !                                           the virtual display.
;  831          1089  1 !
;  832          1090  1 !                          SMG$M_UNDERLINE displays characters underlined.
;  833          1091  1 !
;  834          1092  1 !               CHAR_SET.rl.r   The default character set for all text
;  835          1093  1 !                               associated with this display.
;  836          1094  1 !                               Recognized values are:
;  837          1095  1 !                                           SMG$C_UNITED_KINGDOM
;  838          1096  1 !                                           SMG$C_ASCII (default)
;  839          1097  1 !                                           SMG$C_SPEC_GRAPHICS
;  840          1098  1 !                                           SMG$C_ALT_CHAR
;  841          1099  1 !                                           SMG$C_ALT_GRAPHICS
;  842          1100  1 !
;  843          1101  1 ! IMPLICIT INPUTS:
;  844          1102  1 !
;  845          1103  1 !       NONE
;  846          1104  1 !
;  847          1105  1 ! IMPLICIT OUTPUTS:
;  848          1106  1 !
;  849          1107  1 !       NONE
;  850          1108  1 !
;  851          1109  1 ! COMPLETION STATUS:
;  852          1110  1 !
;  853          1111  1 !       SS$_NORMAL      Normal successful completion
;  854          1112  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to reallocate needed
;  855          1113  1 !                       buffers.
;  856          1114  1 !       SMG$_INVARG     Unrecognized Video Attributes
;  857          1115  1 !                    or Unrecognized Display Attributes
;  858          1116  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  859          1117  1 !
;  860          1118  1 ! SIDE EFFECTS:
;  861          1119  1 !
;  862          1120  1 !       Cursor for virtual display will be forced to row 1 column 1 if
;  863          1121  1 !       display is redimensioned.
;  864          1122  1 !       If a labeled border applies and does not fit newly redimensioned
;  865          1123  1 !       display, the label will be deleted.
;  866          1124  1 !--
;  867          1125  2     BEGIN
;  868          1126  2     BUILTIN
;  869          1127  2         NULLPARAMETER;
;  870          1128  2
;  871          1129  2     LOCAL
;  872          1130  2         STATUS,                 ! Status of subroutine calls
;  873          1131  2         PP  : REF $PP_DECL,     ! Addr. of a pasting packet
```

```
;  874          1132  2          NEW_ROWS,                   ! New number of rows
;  875          1133  2          NEW_COLS,                   ! New number of columns
;  876          1134  2          DCB : REF $DCB_DECL,        ! Addr of display control block
;  877          1135  2          NEW_SIZE;                   ! New rows * columns
;  878          1136
;  879          1137  2      $SMG$VALIDATE_ARGCOUNT (1, 6);      ! Test for right no. of args
;  880          1138
;  881          1139  2      $SMG$GET_DCB (.DISPLAY_ID, DCB);    ! Get address of virtual display
;  882          1140  2                                          ! control block.
;  883          1141  2
;  884          1142  2  !+
;  885          1143  2  ! Determine size of new buffer we need.
;  886          1144  2  !-
;  887          1145  2      IF NOT NULLPARAMETER (NUM_ROWS)     ! If new number of rows specified
;  888          1146  2      THEN
;  889          1147  2          NEW_ROWS = ..NUM_ROWS
;  890          1148  2      ELSE
;  891          1149  2          NEW_ROWS = .DCB [DCB_W_NO_ROWS];
;  892          1150  2
;  893          1151  2      IF NOT NULLPARAMETER (NUM_COLS)     ! If new number of columns specified
;  894          1152  2      THEN
;  895          1153  2          NEW_COLS = ..NUM_COLS
;  896          1154  2      ELSE
;  897          1155  2          NEW_COLS = .DCB [DCB_W_NO_COLS];
;  898          1156  2
;  899          1157  2      NEW_SIZE = .NEW_ROWS * .NEW_COLS;
;  900          1158  2
;  901          1159  2  !+
;  902          1160  2  ! Adjust default display, video attributes and default character set if
;  903          1161  2  ! they are specified.
;  904          1162  2  !-
;  905          1163  2      IF NOT NULLPARAMETER (DISPLAY_ATTRIBUTES)   ! If display attributes specified
;  906          1164  2      THEN
;  907          1165  2          DCB [DCB_B_DEF_DISPLAY_ATTR] = ..DISPLAY_ATTRIBUTES;
;  908          1166  2
;  909          1167  2      IF NOT NULLPARAMETER (VIDEO_ATTRIBUTES)     ! If video attributes specified
;  910          1168  2      THEN
;  911          1169  2          DCB [DCB_B_DEF_VIDEO_ATTR] = ..VIDEO_ATTRIBUTES;
;  912          1170  2
;  913          1171  2      IF NOT NULLPARAMETER (CHAR_SET)     ! If char set specified
;  914          1172  2      THEN
;  915          1173  2          DCB [DCB_B_DEF_CHAR_SET] = ..CHAR_SET;
;  916          1174  2
;  917          1175  2  !+
;  918          1176  2  ! If the dimensions of the old buffer and the new buffers are different,
;  919          1177  2  ! we will have to allocate new buffer space and copy existing text into
;  920          1178  2  ! new buffers.
;  921          1179  2  !-
;  922          1180  2      IF .DCB [DCB_L_BUFSIZE] NEQ .NEW_SIZE       OR
;  923          1181  2         .DCB [DCB_W_NO_ROWS] NEQ .NEW_ROWS       OR
;  924          1182  2         .DCB [DCB_W_NO_COLS] NEQ .NEW_COLS
;  925          1183  2      THEN
;  926          1184  3          BEGIN   ! Redimensioning required
;  927          1185  3          LOCAL
;  928          1186  3              STATUS,                         ! Status of subroutine calls
;  929          1187  3              ROWS_TO_MOVE,           ! No of rows that will be moved from
;  930          1188  3                                      ! old buffer to new.
```

```
;  931     1189  3              COLS_TO_MOVE,       ! No of columns that will be moved from
;  932     1190  3              ;                   ! old to new.
;  933     1191  3              NEW_TEXT_BUF : REF VECTOR [,BYTE],  ! Addr of new text
;  934     1192  3              ;                                   ! buffer
;  935     1193  3              NEW_ATTR_BUF : REF VECTOR [,BYTE],  ! Addr of new attr
;  936     1194  3              ;                                   ! buffer
;  937     1195  3              NEW_CHAR_BUF : REF VECTOR [,BYTE],  ! Addr of new char_set
;  938     1196  3              ;                                   ! buffer
;  939     1197  3
;  940     1198  3              TEXT_PTR : REF VECTOR [, BYTE],     ! Address of current
;  941     1199  3              ;                                   ! text buffer in DCB.
;  942     1200  3
;  943     1201  3              ATTR_PTR : REF VECTOR [,BYTE],      ! Address of current
;  944     1202  3              ;                                   ! attr buffer in DCB
;  945     1203  3
;  946     1204  3              CHAR_PTR : REF VECTOR [,BYTE];      ! Address of current
;  947     1205  3              ;                                   ! char_set buffer in
;  948     1206  3              ;                                   ! DCB
;  949     1207  3
;  950     1208  3          !+
;  951     1209  3          ! Get space for two new, properly-dimensioned buffers.
;  952     1210  3          !-
;  953     1211  3          IF NOT (STATUS = LIB$GET_VM (%REF (2 * .NEW_SIZE),
;  954     1212  3          ;                            NEW_TEXT_BUF))
;  955     1213  3          THEN
;  956     1214  3              RETURN (.STATUS);
;  957     1215  3
;  958     1216  3          NEW_ATTR_BUF = .NEW_TEXT_BUF + .NEW_SIZE;
;  959     1217  3
;  960     1218  3          !+
;  961     1219  3          ! Now need to copy text and attribute information from
;  962     1220  3          ! .DCB [DCB_A_TEXT_BUF] and .DCB [DCB_A_ATTR_BUF] to
;  963     1221  3          ! .NEW_TEST_BUF and .NEW_ATTR_BUF, preserving the line context.
;  964     1222  3          !       First pre-blank new text_buffer and attribute buffer in
;  965     1223  3          !  case old do not cover new area.
;  966     1224  3          !-
;  967     1225  3          CH$FILL ( %C' ',                              .NEW_SIZE, .NEW_TEXT_BUF);
;  968     1226  3          CH$FILL ( .DCB [DCB_B_DEF_VIDEO_ATTR], .NEW_SIZE, .NEW_ATTR_BUF);
;  969     1227  3
;  970     1228  3          TEXT_PTR = .DCB [DCB_A_TEXT_BUF];
;  971     1229  3          ATTR_PTR = .DCB [DCB_A_ATTR_BUF];
;  972     1230  3          CHAR_PTR = .DCB [DCB_A_CHAR_SET_BUF];
;  973     1231  3
;  974     1232  3          ROWS_TO_MOVE = MIN (.DCB [DCB_W_NO_ROWS], .NEW_ROWS);
;  975     1233  3          COLS_TO_MOVE = MIN (.DCB [DCB_W_NO_COLS], .NEW_COLS);
;  976     1234  3
;  977     1235  3          INCR I FROM 1 TO .ROWS_TO_MOVE
;  978     1236  3          DO
;  979     1237  4              BEGIN       ! Move text and attrib. to new buffers.
;  980     1238  4              LOCAL
;  981     1239  4                  SOURCE_INDEX,
;  982     1240  4                  DEST_INDEX;
;  983     1241  4
;  984     1242  4              SOURCE_INDEX = (.I -1) * .DCB [DCB_W_NO_COLS] ;
;  985     1243  4              DEST_INDEX   = (.I -1) * ..NUM_COLS           ;
;  986     1244  4
;  987     1245  4              CH$MOVE ( .COLS_TO_MOVE,                    ! No of chars.
```

```
;  988     1246  4                    TEXT_PTR [.SOURCE_INDEX],          ! From
;  989     1247  4                    NEW_TEXT_BUF [.DEST_INDEX]);       ! To
;  990     1248  4
;  991     1249  4                CH$MOVE ( .COLS_TO_MOVE,               ! No. of chars.
;  992     1250  4                          ATTR_PTR [.SOURCE_INDEX],    ! From
;  993     1251  4                          NEW_ATTR_BUF [.DEST_INDEX]); ! To
;  994     1252  4
;  995     1253  4
;  996     1254  3                END;        ! Move text and attrib to new buffers.
;  997     1255  3
;  998     1256  3            !+
;  999     1257  3            ! Deal with alternate character set buffers if they exist.
; 1000     1258  3            !-
; 1001     1259  3            IF .DCB [DCB_A_CHAR_SET_BUF] NEQ 0
; 1002     1260  3            THEN
; 1003     1261  4                BEGIN       ! Alt. char set buffer exists
; 1004     1262  4                !+
; 1005     1263  4                ! Allocate a new alternate character set buffer and init. it
; 1006     1264  4                !-
; 1007     1265  5                IF NOT (STATUS = LIB$GET_VM (NEW_SIZE, NEW_CHAR_BUF))
; 1008     1266  4                THEN
; 1009     1267  5                    BEGIN
; 1010     1268  5                    LIB$FREE_VM (%REF (2* .NEW_SIZE), NEW_TEXT_BUF);
; 1011     1269  5                    RETURN (.STATUS); ! Return LIB$_INSVIRMEM from GET call
; 1012     1270  4                    END;
; 1013     1271  4
; 1014     1272  4                CH$FILL ( .DCB [DCB_B_DEF_CHAR_SET], .NEW_SIZE,
; 1015     1273  4                          .NEW_CHAR_BUF);
; 1016     1274  4
; 1017     1275  4                !+
; 1018     1276  4                ! Move current contents row by row
; 1019     1277  4                !-
; 1020     1278  4                INCR I FROM 1 TO .ROWS_TO_MOVE
; 1021     1279  4                DO
; 1022     1280  5                    BEGIN   ! Move loop
; 1023     1281  5                    LOCAL
; 1024     1282  5                        SOURCE_INDEX,
; 1025     1283  5                        DEST_INDEX;
; 1026     1284  5
; 1027     1285  5                    SOURCE_INDEX = (.I-1) * .DCB [DCB_W_NO_COLS] ;
; 1028     1286  5                    DEST_INDEX   = (.I-1) * ..NUM_COLS ;
; 1029     1287  5                    CH$MOVE ( .COLS_TO_MOVE,                 ! No of chars
; 1030     1288  5                              CHAR_PTR [.SOURCE_INDEX],      ! From
; 1031     1289  5                              NEW_CHAR_BUF [.DEST_INDEX]);   ! To
; 1032     1290  4                    END;    ! Move loop
; 1033     1291  4
; 1034     1292  4                !+
; 1035     1293  4                ! Free old alternate char. set buffer and plug in new addr.
; 1036     1294  4                !-
; 1037     1295  5                IF NOT (STATUS = LIB$FREE_VM ( DCB [DCB_L_BUFSIZE],
; 1038     1296  5                                               DCB [DCB_A_CHAR_SET_BUF]))
; 1039     1297  4                THEN
; 1040     1298  4                    RETURN (.STATUS);
; 1041     1299  4
; 1042     1300  4                DCB [DCB_A_CHAR_SET_BUF] = .NEW_CHAR_BUF;
; 1043     1301  4
; 1044     1302  3                END;        ! Alt. char set buffer exists
```

```
;  1045     1303  3          !+
;  1046     1304  3          ! Now that the text and attributes are safe in their
;  1047     1305  3          ! new buffers, we release the old buffers and put the addresses
;  1048     1306  3          ! of the new buffers in the DCB.
;  1049     1307  3          !-
;  1050     1308  3
;  1051     1309  4          IF NOT (STATUS = LIB$FREE_VM (%REF ( 2 * .DCB [DCB_L_BUFSIZE]),
;  1052     1310  4                                        DCB [DCB_A_TEXT_BUF]))
;  1053     1311  4          THEN
;  1054     1312  4              RETURN (.STATUS);
;  1055     1313  3
;  1056     1314  3          DCB [DCB_A_TEXT_BUF] = .NEW_TEXT_BUF;    ! Plug in new addresses
;  1057     1315  3          DCB [DCB_A_ATTR_BUF] = .NEW_ATTR_BUF;
;  1058     1316  3
;  1059     1317  3          !+
;  1060     1318  3          ! If the number of rows changed, we need to reallocate the
;  1061     1319  3          ! line characteristics vector and copy over as much of it as
;  1062     1320  3          ! fits.
;  1063     1321  3          !-
;  1064     1322  3          IF .DCB [DCB_W_NO_ROWS] NEQ .NEW_ROWS
;  1065     1323  3          THEN
;  1066     1324  4              BEGIN       ! No. of rows changed
;  1067     1325  4              LOCAL
;  1068     1326  4                  NEW_LINE_CHAR : REF VECTOR [,BYTE],      ! Addr of a new
;  1069     1327  4                                                           ! line Char.
;  1070     1328  4                                                           ! vector
;  1071     1329  4
;  1072     1330  4                  LINE_CHAR_PTR : REF VECTOR [,BYTE];      ! Addr of curr.
;  1073     1331  4                                                           ! line Char.
;  1074     1332  4                                                           ! vector
;  1075     1333  4
;  1076     1334  4              LINE_CHAR_PTR = .DCB [DCB_A_LINE_CHAR];
;  1077     1335  4
;  1078     1336  4              !+
;  1079     1337  4              ! Allocate a new line characteristics vector of the right
;  1080     1338  4              ! length.  Quit if we can't get it.
;  1081     1339  4              !-
;  1082     1340  5              IF NOT (STATUS = LIB$GET_VM ( %REF (.NEW_ROWS + 1),
;  1083     1341  5                                            NEW_LINE_CHAR))
;  1084     1342  4              THEN
;  1085     1343  5                  BEGIN   ! Error path
;  1086     1344  5                  !+
;  1087     1345  5                  ! Give back all space acquired on this transaction,
;  1088     1346  5                  ! ignoring further errors, and quit.
;  1089     1347  5                  !-
;  1090     1348  5                  LIB$FREE_VM ( %REF (2 * .NEW_SIZE), NEW_TEXT_BUF);
;  1091     1349  5                  RETURN ( .STATUS );
;  1092     1350  4                  END;    ! Error path
;  1093     1351  4
;  1094     1352  4              !+
;  1095     1353  4              ! Clear entire allocated vector to zero
;  1096     1354  4              !-
;  1097     1355  4              CH$FILL ( 0, .NEW_ROWS+1, .NEW_LINE_CHAR);
;  1098     1356  4
;  1099     1357  4              !+
;  1100     1358  4              ! Copy over as much of old line characteristics vector as
;  1101     1359  4              ! will fit.
```

```
;  1102     1360  4                !-
;  1103     1361  4                CH$MOVE ( .ROWS_TO_MOVE,
;  1104     1362  4                          LINE_CHAR_PTR [1],
;  1105     1363  4                          NEW_LINE_CHAR [1]);
;  1106     1364  4
;  1107     1365  4                !+
;  1108     1366  4                ! Free former line characteristics vector.
;  1109     1367  4                !-
;  1110     1368  5                IF NOT (STATUS = LIB$FREE_VM (
;  1111     1369  5                                    %REF (.DCB [DCB_W_NO_ROWS] +1),
;  1112     1370  5                                    DCB [DCB_A_LINE_CHAR]))
;  1113     1371  4                THEN
;  1114     1372  4                    RETURN (.STATUS);
;  1115     1373  4
;  1116     1374  4                !+
;  1117     1375  4                ! Store address of new line characteristics vector in DCB
;  1118     1376  4                !-
;  1119     1377  4                DCB [DCB_A_LINE_CHAR] = .NEW_LINE_CHAR;
;  1120     1378  3                END;            ! No. of rows changed
;  1121     1379  3
;  1122     1380  3            !+
;  1123     1381  3            ! Adjust the no. of rows and no. of cols. recorded in the DCB.
;  1124     1382  3            !-
;  1125     1383  3            DCB [DCB_W_NO_ROWS] = .NEW_ROWS;        ! Adjust row/column size
;  1126     1384  3            DCB [DCB_W_NO_COLS] = .NEW_COLS;
;  1127     1385  3            DCB [DCB_L_BUFSIZE] = .NEW_SIZE;
;  1128     1386  3
;  1129     1387  3            !+
;  1130     1388  3            ! Force cursor to home.
;  1131     1389  3            !-
;  1132     1390  3            DCB [DCB_W_CURSOR_ROW] = 1;
;  1133     1391  3            DCB [DCB_W_CURSOR_COL] = 1;
;  1134     1392  3
;  1135     1393  3            !+
;  1136     1394  3            ! Knock down flags that indicate we are at end of a row and that
;  1137     1395  3            ! we are in last line.
;  1138     1396  3            !-
;  1139     1397  3            DCB [DCB_V_FULL]   = 0;
;  1140     1398  3            DCB [DCB_V_COL_80] = 0;
;  1141     1399  3
;  1142     1400  3            !+
;  1143     1401  3            ! Reset the scrolling region within the redimensioned virtual
;  1144     1402  3            ! display to be the whole display.
;  1145     1403  3            !-
;  1146     1404  3            DCB [DCB_W_TOP_OF_SCRREG]    = 1;
;  1147     1405  3            DCB [DCB_W_BOTTOM_OF_SCRREG] = .NEW_ROWS;
;  1148     1406  3
;  1149     1407  3            !+
;  1150     1408  3            ! Now deal with border data, if any exists.
;  1151     1409  3            !-
;  1152     1410  3            IF .DCB [DCB_V_BORDERED]
;  1153     1411  3            THEN
;  1154     1412  4                BEGIN       ! Bordered
;  1155     1413  4                LOCAL
;  1156     1414  4                    DESC : REF BLOCK [8,BYTE]; ! Pointer to dynamic string
;  1157     1415  4                                               ! desc. for border label
;  1158     1416  4
```

```
;  1159          1417  4              DESC = DCB [DCB_Q_LABEL_DESC];
;  1160          1418  4              IF .DESC [DSC$A_POINTER] NEQ 0      ! If label exists
;  1161          1419  4              THEN
;  1162          1420  5                  BEGIN   ! Label exists
;  1163          1421  5                  LOCAL
;  1164          1422  5                      TEMP;
;  1165          1423  5                  TEMP = .DCB [DCB_W_LABEL_UNITS];
;  1166          1424  5
;  1167          1425  5                  !+
;  1168          1426  5                  ! Try to reapply our existing border label on this
;  1169          1427  5                  ! redimensioned virtual display.  If it now doesn't
;  1170          1428  5                  ! fit because of the new dimensions, delete the label.
;  1171          1429  5                  !-
;  1172          1430  6                  IF NOT (SMG$LABEL_BORDER (
;  1173          1431  6                                  .DISPLAY_ID,
;  1174          1432  6                                  .DESC,
;  1175          1433  6                                  %REF (.DCB [DCB_B_LABEL_POS]),
;  1176          1434  6                                  !+
;  1177          1435  6                                  ! Conditionalize UNITS parameter to
;  1178          1436  6                                  ! LABEL_BORDER depending on whether
;  1179          1437  6                                  ! caller originally specified
;  1180          1438  6                                  ! "centering" or gave us specific units.
;  1181          1439  6                                  !-
;  1182          1440  7                                  (IF .DCB [DCB_V_LABEL_CENTER] THEN 0
;  1183          1441  6                                                                ELSE TEMP),
;  1184          1442  6                                  %REF ( .DCB [DCB_B_LABEL_REND])
;  1185          1443  6                                  ))
;  1186          1444  5                  THEN
;  1187          1445  5                      LIB$SFREE1_DD ( .DESC);     ! Delete label
;  1188          1446  4                  END;    ! Label exists
;  1189          1447  3              END;        ! Bordered
;  1190          1448  2          END;    ! Redimensioning required
;  1191          1449  2
;  1192          1450  2 !+
;  1193          1451  2 ! Since the dimension of the virtual display may have changed, or we
;  1194          1452  2 ! may have added or deleted a border,  we need to recalculate the
;  1195          1453  2 ! transformation constants that occur in each pasting packet we are
;  1196          1454  2 ! involved in.
;  1197          1455  2 ! Check to see if we can do it now or must wait because we are batched.
;  1198          1456  2 !-
;  1199          1457  2      IF .DCB [DCB_L_BATCH_LEVEL] EQL 0
;  1200          1458  2      THEN
;  1201          1459  3          BEGIN   ! Can do it now
;  1202          1460  3          LOCAL
;  1203          1461  3              CURR_PP : REF $PP_DECL;     ! Addr of a pasting packet
;  1204          1462  3
;  1205          1463  3          IF NOT (STATUS = SMG$$RECALC_PP_FIELDS ( .DCB))
;  1206          1464  3          THEN
;  1207          1465  3              RETURN (.STATUS) ;
;  1208          1466  3
;  1209          1467  3          !+
;  1210          1468  3          ! Remap all pasteboard buffers to which we are pasted, from the
;  1211          1469  3          ! bottom outward.
;  1212          1470  3          !-
;  1213          1471  3          CURR_PP = .DCB [DCB_A_PP_NEXT];
;  1214          1472  3          WHILE .CURR_PP NEQ DCB [DCB_A_PP_NEXT]
;  1215          1473  3          DO
```

```
;  1216          1474  4              BEGIN       ! Remap all pasteboards
;  1217          1475  4              LOCAL
;  1218          1476  4                  PBCB : REF $PBCB_DECL;  ! Addr of a pasteboard control
;  1219          1477  4                                          ! block
;  1220          1478  4              PBCB = .CURR_PP [PP_A_PBCB_ADDR];
;  1221          1479  5              IF NOT (STATUS = SMG$$CHECK_FOR_OUTPUT_PBCB ( .PBCB))
;  1222          1480  4              THEN
;  1223          1481  4                  RETURN ( .STATUS);      ! Quit if any one of them fails
;  1224          1482  4
;  1225          1483  4              CURR_PP = .CURR_PP [PP_A_NEXT_DCB]; ! To next pasting packet
;  1226          1484  3              END;        ! Remap all pasteboards
;  1227          1485  3
;  1228          1486  3          RETURN ( SS$_NORMAL);
;  1229          1487  3          END     ! Can do it now
;  1230          1488  3
;  1231          1489  2      ELSE
;  1232          1490  2
;  1233          1491  3          BEGIN   ! Must delay until end_display_batch
;  1234          1492  3          DCB [DCB_V_PP_MISMATCH] = 1;    ! Mark it for later update
;  1235          1493  2          END;    ! Must delay until end_display_batch
;  1236          1494  2
;  1237          1495  2      RETURN ( SS$_NORMAL);
;  1238          1496  1      END;                            ! Routine SMG$CHANGE_VIRTUAL_DISPLAY



                                        0FFC 00000          .ENTRY  SMG$CHANGE_VIRTUAL_DISPLAY, Save R2,R3,R4,-      ;  1019
                                                                    R5,R6,R7,R8,R9,R10,R11
                                5E              30  C2 00002        SUBL2   #48, SP
          50                    6C              01  83 00005        SUBB3   #1, (AP), DIFF                           ;  1137
                                05              50  91 00009        CMPB    DIFF, #5
                                                08  1B 0000C        BLEQU   1$
                                50 00000000G    8F  D0 0000E        MOVL    #SMG$_WRONUMARG, R0
                                                04  00015          RET
                                50          04  BC  D0 00016 1$:    MOVL    @DISPLAY_ID, R0                          ;  1139
                          04    BC          38  A0  D1 0001A        CMPL    56(R0), @DISPLAY_ID
                                                06  12 0001F        BNEQ    2$
                                11          44  A0  91 00021        CMPB    68(R0), #17
                                                08  13 00025        BEQL    3$
                                50 00000000G    8F  D0 00027 2$:    MOVL    #SMG$_INVDIS_ID, R0
                                                04  0002E          RET
                                56          04  BC  D0 0002F 3$:    MOVL    @DISPLAY_ID, DCB
                                02                  6C  91 00033    CMPB    (AP), #2                                 ;  114
                                                    0B  1F 00036    BLSSU   4$
                                            08      AC  D5 00038    TSTL    8(AP)
                                                    06  13 0003B    BEQL    4$
                                58          08      BC  D0 0003D    MOVL    @NUM_ROWS, NEW_ROWS                      ;  1147
                                                    04  11 00041    BRB     5$
                                58          02      A6  3C 00043 4$:    MOVZWL  2(DCB), NEW_ROWS                 ;  1149
                                03                  6C  91 00047 5$:    CMPB    (AP), #3                         ;  1151
                                                    0C  1F 0004A    BLSSU   6$
                                            0C      AC  D5 0004C    TSTL    12(AP)
                                                    07  13 0004F    BEQL    6$
                          18    AE          0C      BC  D0 00051    MOVL    @NUM_COLS, NEW_COLS                      ;  1153
                                                    05  11 00056    BRB     7$
```

```
                                18   AE      06   A6  3C  00058 6$:     MOVZWL  6(DCB), NEW_COLS                                ; 1155
                          20    AE          58   18  AE  C5  0005D 7$:     MULL3   NEW_COLS, NEW_ROWS, NEW_SIZE                    ; 1157
                                            04   6C  91  00063           CMPB    (AP), #4                                        ; 1163
                                                 0A  1F  00066           BLSSU   8$
                                            10   AC  D5  00068           TSTL    16(AP)
                                                 05  13  0006B           BEQL    8$
                                2F   A6     10   BC  90  0006D           MOVB    @DISPLAY_ATTRIBUTES, 47(DCB)                    ; 1165
                                            05   6C  91  00072 8$:        CMPB    (AP), #5                                        ; 1167
                                                 0A  1F  00075           BLSSU   9$
                                            14   AC  D5  00077           TSTL    20(AP)
                                                 05  13  0007A           BEQL    9$
                                2E   A6     14   BC  90  0007C           MOVB    @VIDEO_ATTRIBUTES, 46(DCB)                      ; 1169
                                            06   6C  91  00081 9$:        CMPB    (AP), #6                                        ; 1171
                                                 0A  1F  00084           BLSSU   10$
                                            18   AC  D5  00086           TSTL    24(AP)
                                                 05  13  00089           BEQL    10$
                                30   A6     18   BC  90  0008B           MOVB    @CHAR_SET, 48(DCB)                              ; 1173
                                            57   20  AE  D0  00090 10$:    MOVL    NEW_SIZE, R7                                    ; 1180
                                            57   3C  A6  D1  00094           CMPL    60(DCB), R7
                                                 14  12  00098           BNEQ    11$
                     58   02    A6           10       00  ED  0009A           CMPZV   #0, #16, 2(DCB), NEW_ROWS                       ; 1181
                                                 0C  12  000A0           BNEQ    11$
                18   AE   06    A6           10       00  ED  000A2           CMPZV   #0, #16, 6(DCB), NEW_COLS                       ; 1182
                                                 03  12  000A9           BNEQ    11$
                                            01F7 31  000AB           BRW     29$
                                            28   AE  9F  000AE 11$:     PUSHAB  NEW_TEXT_BUF                                    ; 1211
                                18   AE          57   01  78  000B1           ASHL    #1, R7, 24(SP)
                                            18   AE  9F  000B6           PUSHAB  24(SP)
                          00000000G         00       02  FB  000B9           CALLS   #2, LIB$GET_VM
                                            6E       50  D0  000C0           MOVL    R0, STATUS
                                            03       6E  E8  000C3           BLBS    STATUS, 12$
                                            016B     31  000C6           BRW     24$
                                0C   AE     28   BE47 9E  000C9 12$:     MOVAB   @NEW_TEXT_BUF[R7], NEW_ATTR_BUF                 ; 1216
                57                20         6E       00  2C  000CF           MOVC5   #0, (SP), #32, R7, @NEW_TEXT_BUF                ; 1225
                                            28   BE      000D4
                57        2E   A6           6E       00  2C  000D6           MOVC5   #0, (SP), 46(DCB), R7, @NEW_ATTR_BUF            ; 1226
                                            0C   BE      000DC
                                            5B   10  A6  D0  000DE           MOVL    16(DCB), TEXT_PTR                               ; 1228
                                            5A   14  A6  D0  000E2           MOVL    20(DCB), ATTR_PTR                               ; 1229
                                0C   AE     04   18  A6  D0  000E6           MOVL    24(DCB), CHAR_PTR                               ; 1230
                                            52   02  A6  3C  000EB           MOVZWL  2(DCB), R2                                      ; 1232
                                            58       52  D1  000EF           CMPL    R2, NEW_ROWS
                                                     03  15  000F2           BLEQ    13$
                                            52       58  D0  000F4           MOVL    NEW_ROWS, R2
                                0E   AE     14   AE  52  D0  000F7 13$:     MOVL    R2, ROWS_TO_MOVE
                                            52   06  A6  3C  000FB           MOVZWL  6(DCB), R2                                      ; 1233
                                0C   AE     18   AE  52  D1  000FF           CMPL    R2, NEW_COLS
                                                     04  15  00103           BLEQ    14$
                                            52       18  AE  D0  00105           MOVL    NEW_COLS, R2
                                0A   AE     10   AE      52  D0  00109 14$:     MOVL    R2, COLS_TO_MOVE
                                                         57  D4  0010D           CLRL    I                                       ; 1250
                                                         26  11  0010F           BRB     16$
                                            50       FF  A7  9E  00111 15$:     MOVAB   -1(R7), R0                              ; 1242
                                            51       06  A6  3C  00115           MOVZWL  6(DCB), R1
                                08   AE     50           51  C5  00119           MULL3   R1, R0, SOURCE_INDEX
                                       59   50       0C  BC  C5  0011E           MULL3   @NUM_COLS, R0, DEST_INDEX               ; 1243
                          28   BE49         08   BE4B    10  AE  28  00123           MOVC3   COLS_TO_MOVE, @SOURCE_INDEX[TEXT_PTR], -    ; 1247
```

```
                                                             @NEW_TEXT_BUF[DEST_INDEX]                    ;
                                   0C BE49  9F 0012C         PUSHAB  @NEW_ATTR_BUF[R9]                    ; 1251
               9E       0C BE4A    14   AE  28 00130         MOVC3   COLS_TO_MOVE, @SOURCE_INDEX[ATTR_PTR], - ;
                                                             @(SP)+                                       ;
               D5            57    14   AE  F3 00137 16$:    AOBLEQ  ROWS_TO_MOVE, I, 15$                 ; 1235
                                   18   A6  D5 0013C         TSTL    24(DCB)                              ; 1259
                                        6E  13 0013F         BEQL    22$                                  ;
                                   1C   AE  9F 00141         PUSHAB  NEW_CHAR_BUF                         ; 1265
                                   24   AE  9F 00144         PUSHAB  NEW_SIZE                             ;
                    00000000G  00       02  FB 00147         CALLS   #2, LIB$GET_VM                       ;
                               6E       50  D0 0014E         MOVL    R0, STATUS                           ;
                               16       6E  E8 00151         BLBS    STATUS, 19$                          ;
                                   28   AE  9F 00154         PUSHAB  NEW_TEXT_BUF                         ; 1268
          0C   AE       24     AE       01  78 00157         ASHL    #1, NEW_SIZE, 12(SP)                 ;
                                   0C   AE  9F 0015D         PUSHAB  12(SP)                               ;
                    00000000G  00       02  FB 00160 17$:    CALLS   #2, LIB$FREE_VM                      ;
                                      00CA  31 00167 18$:    BRW     24$                                  ; 1269
 20   AE  30   A6              6E       00  2C 0016A 19$:    MOVC5   #0, (SP), 48(DCB), NEW_SIZE, @NEW_CHAR_BUF ; 1273
                                   1C   BE     00171                                                      ;
                                        59  D4 00173         CLRL    I                                    ; 1288
                                        1B  11 00175         BRB     21$                                  ;
                               50  FF   A9  9E 00177 20$:    MOVAB   -1(R9), R0                           ; 1285
                               51  06   A6  3C 0017B         MOVZWL  6(DCB), SOURCE_INDEX                 ;
                               51       50  C4 0017F         MULL2   R0, SOURCE_INDEX                     ;
                               50  0C   BC  C4 00182         MULL2   @NUM_COLS, DEST_INDEX                ; 1286
                               57  04   AE  D0 00186         MOVL    CHAR_PTR, R7                         ; 1289
          1C BE40            6147  10   AE  28 0018A         MOVC3   COLS_TO_MOVE, (SOURCE_INDEX)[R7], -  ;
                                                             @NEW_CHAR_BUF[DEST_INDEX]                    ;
               E0            59    14   AE  F3 00192 21$:    AOBLEQ  ROWS_TO_MOVE, I, 20$                 ; 1278
                                   18   A6  9F 00197         PUSHAB  24(DCB)                              ; 1296
                                   3C   A6  9F 0019A         PUSHAB  60(DCB)                              ; 1295
                    00000000G  00       02  FB 0019D         CALLS   #2, LIB$FREE_VM                      ; 1296
                               6E       50  D0 001A4         MOVL    R0, STATUS                           ;
                               BD       6E  E9 001A7         BLBC    STATUS, 18$                          ;
                      18       A6  1C   AE  D0 001AA         MOVL    NEW_CHAR_BUF, 24(DCB)                ; 1300
                                   10   A6  9F 001AF 22$:    PUSHAB  16(DCB)                              ; 1310
          14   AE       3C     A6       01  78 001B2         ASHL    #1, 60(DCB), 20(SP)                  ; 1309
                                   14   AE  9F 001B8         PUSHAB  20(SP)                               ;
                    00000000G  00       02  FB 001BB         CALLS   #2, LIB$FREE_VM                      ; 1310
                               6E       50  D0 001C2         MOVL    R0, STATUS                           ;
                               6C       6E  E9 001C5         BLBC    STATUS, 24$                          ;
                      10       A6  28   AE  D0 001C8         MOVL    NEW_TEXT_BUF, 16(DCB)                ; 1314
                      14       A6  0C   AE  D0 001CD         MOVL    NEW_ATTR_BUF, 20(DCB)                ; 1315
      58  02   A6              10       00  ED 001D2         CMPZV   #0, #16, 2(DCB), NEW_ROWS            ; 1322
                                        62  13 001D8         BEQL    26$                                  ;
                               59  4C   A6  D0 001DA         MOVL    76(DCB), LINE_CHAR_PTR               ; 1334
                                   24   AE  9F 001DE         PUSHAB  NEW_LINE_CHAR                        ; 1340
                               52  01   A8  9E 001E1         MOVAB   1(R8), R2                            ;
                      14       AE       52  D0 001E5         MOVL    R2, 20(SP)                           ;
                                   14   AE  9F 001E9         PUSHAB  20(SP)                               ;
                    00000000G  00       02  FB 001EC         CALLS   #2, LIB$GET_VM                       ;
                               6E       50  D0 001F3         MOVL    R0, STATUS                           ;
                               0F       6E  E8 001F6         BLBS    STATUS, 23$                          ;
                                   28   AE  9F 001F9         PUSHAB  NEW_TEXT_BUF                         ; 1348
          14   AE       24     AE       01  78 001FC         ASHL    #1, NEW_SIZE, 20(SP)                 ;
                                   14   AE  9F 00202         PUSHAB  20(SP)                               ;
                                      FF58  31 00205         BRW     17$                                  ;
```

```
                                 5A        24  AE  D0 00208 23$:    MOVL    NEW_LINE_CHAR, R10                          ; 1355
          52          00           6E            00  2C 0020C          MOVC5   #0, (SP), #0, R2, (R10)
                                                 6A 00211
                   01    AA     01    A9        14  AE  28 00212          MOVC3   ROWS_TO_MOVE, 1(LINE_CHAR_PTR), 1(R10)      ; 1363
                                             4C  A6  9F 00219          PUSHAB  76(DCB)                                     ; 1370
                                     18    AE  02  A6  3C 0021C          MOVZWL  2(DCB), 24(SP)                              ; 1369
                                             18  AE  D6 00221          INCL    24(SP)
                                             18  AE  9F 00224          PUSHAB  24(SP)
                         00000000G   00            02  FB 00227          CALLS   #2, LIB$FREE_VM                             ; 1370
                                     6E            50  D0 0022E          MOVL    R0, STATUS
                                     04            6E  E8 00231          BLBS    STATUS, 25$
                                     50            6E  D0 00234 24$:     MOVL    STATUS, R0                                  ; 1372
                                                   04  00237          RET
                                 4C  A6            5A  D0 00238 25$:     MOVL    R10, 76(DCB)                                ; 1377
                                 02  A6            58  B0 0023C 26$:     MOVW    NEW_ROWS, 2(DCB)                            ; 1383
                                 06  A6      18    AE  B0 00240          MOVW    NEW_COLS, 6(DCB)                            ; 1384
                                 3C  A6      20    AE  D0 00245          MOVL    NEW_SIZE, 60(DCB)                           ; 1385
                                 28  A6  00010001    8F  D0 0024A          MOVL    #65537, 40(DCB)                             ; 1390
                                 34  A6            03  8A 00252          BICB2   #3, 52(DCB)                                 ; 1398
                                 48  A6            01  B0 00256          MOVW    #1, 72(DCB)                                 ; 1404
                                 4A  A6            58  B0 0025A          MOVW    NEW_ROWS, 74(DCB)                           ; 1405
                                     43      2F    A6  E9 0025E          BLBC    47(DCB), 29$                                ; 1410
                                     52      08    A6  9E 00262          MOVAB   8(R6), DESC                                 ; 1417
                                             04    A2  D5 00266          TSTL    4(DESC)                                     ; 1418
                                                   3A  13 00269          BEQL    29$
                                 2C  AE      2C    A6  3C 00268          MOVZWL  44(DCB), TEMP                               ; 1423
                                 18  AE      33    A6  9A 00270          MOVZBL  51(DCB), 24(SP)                             ; 1442
                                             18    AE  9F 00275          PUSHAB  24(SP)
                             04      34    A6            02  E1 00278          BBC     #2, 52(DCB), 27$                    ; 1440
                                                   7E  D4 0027D          CLRL    -(SP)
                                                   06  11 0027F          BRB     28$
                                     50      30    AE  9E 00281 27$:     MOVAB   TEMP, R0
                                                   50  DD 00285          PUSHL   R0
                                 1C  AE      31    A6  9A 00287 28$:     MOVZBL  49(DCB), 28(SP)                             ; 1433
                                             1C    AE  9F 002RC          PUSHAB  28(SP)
                                                   52  DD 0028F          PUSHL   DESC                                        ; 1432
                                             04    AC  DD 00291          PUSHL   DISPLAY_ID                                  ; 1431
                             0000V   CF            05  FB 00294          CALLS   #5, SMG$LABEL_BORDER
                                     09            50  E8 00299          BLBS    R0, 29$
                                                   52  DD 0029C          PUSHL   DESC                                        ; 1445
                         00000000G   00            01  FB 0029E          CALLS   #1, LIB$SFREE1_DD
                                             1C    A6  D5 002A5 29$:     TSTL    28(DCB)                                     ; 1457
                                                   2C  12 002A8          BNEQ    31$
                                                   56  DD 002AA          PUSHL   DCB                                         ; 1463
                             0000V   CF            01  FB 002AC          CALLS   #1, SMG$$RECALC_PP_FIELDS
                                     29            50  E9 002B1          BLBC    STATUS, 33$
                                     52      20    A6  D0 002B4          MOVL    32(DCB), CURR_PP                            ; 1471
                                     51      20    A6  9E 002B8 30$:     MOVAB   32(DCB), R1                                 ; 1472
                                     51            52  D1 002BC          CMPL    CURR_PP, R1
                                                   19  13 002BF          BEQL    32$
                                     51      14    A2  D0 002C1          MOVL    20(CURR_PP), PBCB                           ; 1478
                                                   51  DD 002C5          PUSHL   PBCB                                        ; 1479
                         00000000G   00            01  FB 002C7          CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB
                                     0C            50  E9 002CE          BLBC    STATUS, 33$
                                     52            62  D0 002D1          MOVL    (CURR_PP), CURR_PP                          ; 1483
                                                   E2  11 002D4          BRB     30$                                         ; 1472
                                 34  A6            08  88 002D6 31$:     BISB2   #8, 52(DCB)                                 ; 1492
```

```
                                50          01  D0 002DA 32$:    MOVL    #1, R0                                  ; 1495
                                                04 002DD 33$:    RET                                             ; 1496

; Routine Size:  734 bytes,    Routine Base:  _SMG$CODE + 0351

; 1239          1497  1 !<BLF/PAGE>
```

```
;  1241          1498  1 %SBTTL 'SMG$CHECK_FOR_OCCLUSION - Check to see if display is occluded'
;  1242          1499  1 GLOBAL ROUTINE SMG$CHECK_FOR_OCCLUSION (
;  1243          1500  1                                         DISPLAY_ID,
;  1244          1501  1                                         PASTEBOARD_ID,
;  1245          1502  1                                         OCCLUSION_STATE
;  1246          1503  1                                        ) =
;  1247          1504  1 !++
;  1248          1505  1 ! FUNCTIONAL DESCRIPTION:
;  1249          1506  1 !
;  1250          1507  1 !       This procedure determines if the given virtual display, as
;  1251          1508  1 !       pasted to the given pasteboard, is occluded by another virtual
;  1252          1509  1 !       display.  The OCCLUSION state is set to:
;  1253          1510  1 !
;  1254          1511  1 !                       1       : if virtual display is occluded
;  1255          1512  1 !                       0       : if virtual display is not occluded.
;  1256          1513  1 !               not meaningfull : if status is not SS$_NORMAL.
;  1257          1514  1 !
;  1258          1515  1 !       The returned status reflects whether the question could be
;  1259          1516  1 !       answered at all.
;  1260          1517  1 !
;  1261          1518  1 ! CALLING SEQUENCE:
;  1262          1519  1 !
;  1263          1520  1 !       ret_status.wlc.v = SMG$CHECK_FOR_OCCLUSION (
;  1264          1521  1 !                                         DISPLAY_ID.rl.r,
;  1265          1522  1 !                                         PASTEBOARD_ID.rl.r,
;  1266          1523  1 !                                         OCCLUSION_STATE.wl.r)
;  1267          1524  1 ! FORMAL PARAMETERS:
;  1268          1525  1 !
;  1269          1526  1 !       DISPLAY_ID.rl.r         Address of a display id.
;  1270          1527  1 !
;  1271          1528  1 !       PASTEBOARD_ID.rl.r      Address of a pasteboard id.
;  1272          1529  1 !
;  1273          1530  1 !       OCCLUSION_STATE.wl.r    Set to          1  if occluded
;  1274          1531  1 !                                               0  if not occluded
;  1275          1532  1 !
;  1276          1533  1 !
;  1277          1534  1 ! IMPLICIT INPUTS:
;  1278          1535  1 !
;  1279          1536  1 !       NONE
;  1280          1537  1 !
;  1281          1538  1 ! IMPLICIT OUTPUTS:
;  1282          1539  1 !
;  1283          1540  1 !       NONE
;  1284          1541  1 !
;  1285          1542  1 ! COMPLETION STATUS:
;  1286          1543  1 !
;  1287          1544  1 !       SS$_NORMAL      Normal success.  OCCLUSION_STATE calculated.
;  1288          1545  1 !
;  1289          1546  1 !       SMG$_NOTPASTED  Given virtual display is not pasted to given
;  1290          1547  1 !                       pasteboard.
;  1291          1548  1 !
;  1292          1549  1 !       SMG$_INVPAS_ID  Invalid pasteboard id.
;  1293          1550  1 !
;  1294          1551  1 !       SMG$_INVDIS_ID  Invalid display id.
;  1295          1552  1 !
;  1296          1553  1 ! SIDE EFFECTS:
;  1297          1554  1 !
```

```
;  1298          1555  1 !       NONE
;  1299          1556  1 !--
;  1300          1557  1
;  1301          1558  2     BEGIN
;  1302          1559  2     LOCAL
;  1303          1560  2         STATUS,                  ! Status of subroutine calls
;  1304          1561  2
;  1305          1562  2         PP : REF $PP_DECL,       ! Address of relevant pasting packet
;  1306          1563  2
;  1307          1564  2         PBCB: REF $PBCB_DECL,    ! Address of Pasteboard Control Block
;  1308          1565  2
;  1309          1566  2         DCB : REF $DCB_DECL;     ! Address of Display Control Block
;  1310          1567  2
;  1311          1568  2 !+
;  1312          1569  2 ! Validate number of arguments.
;  1313          1570  2 !-
;  1314          1571  2     $SMG$VALIDATE_ARGCOUNT (3, 3);
;  1315          1572  2
;  1316          1573  2 !+
;  1317          1574  2 ! Get DCB and PBCB addresses that go with these display ids and
;  1318          1575  2 ! pasteboard ids.
;  1319          1576  2 !-
;  1320          1577  2     $SMG$GET_DCB (.DISPLAY_ID, DCB);
;  1321          1578  2     $SMG$GET_PBCB (.PASTEBOARD_ID, PBCB);
;  1322          1579  2
;  1323          1580  2 !+
;  1324          1581  2 ! Try to find the pasting packet that binds these two.  Return
;  1325          1582  2 ! SMG$_NOTPASTED it can't be located.
;  1326          1583  2 !-
;  1327          1584  3     IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
;  1328          1585  2     THEN
;  1329          1586  2         RETURN (.STATUS);
;  1330          1587  2
;  1331          1588  2 !+
;  1332          1589  2 ! Check to see if occluded and return appropriate OCCLUSION_STATE.
;  1333          1590  2 !-
;  1334          1591  3     .OCCLUSION_STATE = ( IF .PP [PP_V_OCCLUDED] THEN 1    ! Occluded
;  1335          1592  2                                                ELSE 0);  ! Not occluded
;  1336          1593  2
;  1337          1594  2     RETURN SS$_NORMAL;
;  1338          1595  1     END;                     ! End of routine SMG$CHECK_FOR_OCCLUSION


                                        0004 00000          .ENTRY  SMG$CHECK_FOR_OCCLUSION, Save R2                ; 1499
                52 00000000'    EF   9E 00002               MOVAB   PBD_L_COUNT, R2                                 ;
                5E              04   C2 00009               SUBL2   #4, SP                                          ;
                03              6C   91 0000C               CMPB    (AP), #3                                        ; 1571
                                08   13 0000F               BEQL    1$                                              ;
                50 00000000G    8F   D0 00011               MOVL    #SMG$_WRONUMARG, R0                             ;
                                04   00018               RET                                                     ;
                50         04   BC   D0 00019 1$:          MOVL    @DISPLAY_ID, R0                                 ; 1577
          04    BC         38   A0   D1 0001D               CMPL    56(R0), @DISPLAY_ID                             ;
                                06   12 00022               BNEQ    2$                                              ;
                11         44   A0   91 00024               CMPB    68(R0), #17                                     ;
```

```
                                        08  13  00028          BEQL    3$
                                50 00000000G  8F  D0  0002A 2$:  MOVL    #SMG$_INVDIS_ID, R0                    ;
                                              04  00031          RET                                            ;
                                51        04  BC  D0  00032 3$:  MOVL    @DISPLAY_ID, DCB                       ;
                                50        08  BC  D0  00036      MOVL    @PASTEBOARD_ID, R0                     ; 1578
                                              0A  19  0003A      BLSS    4$                                     ;
                                62            50  D1  0003C      CMPL    R0, PBD_L_COUNT                        ;
                                              05  14  0003F      BGTR    4$                                     ;
                          08        44    A2    50  E0  00041      BBS     R0, PBD_V_PB_AVAIL, 5$                 ;
                                50 00000000G  8F  D0  00046 4$:  MOVL    #SMG$_INVPAS_ID, R0                    ;
                                              04  0004D          RET                                            ;
                                50        04 A240  D0  0004E 5$:  MOVL    PBD_A_PBCB[R0], PBCB                   ;
                                        4001    8F  BB  00053      PUSHR   #^M<R0,SP>                             ; 1584
                                                51  DD  00057      PUSHL   DCB                                    ;
                             0000V      CF        03  FB  00059      CALLS   #3, SMG$$LOCATE_PP                     ;
                                15            50  E9  0005E      BLBC    STATUS, 8$                             ;
                                50            6E  D0  00061      MOVL    PP, R0                                 ; 1591
                                05        2A  A0  E9  00064      BLBC    42(R0), 6$                             ;
                                50            01  D0  00068      MOVL    #1, R0                                 ;
                                              02  11  0006B      BRB     7$                                     ;
                                              50  D4  0006D 6$:  CLRL    R0                                     ;
                             0C     BC        50  D0  0006F 7$:  MOVL    R0, @OCCLUSION_STATE                   ;
                                50            01  D0  00073      MOVL    #1, R0                                 ; 1594
                                              04  00076 8$:  RET                                                ; 1595

; Routine Size.  119 bytes,    Routine Base:  _SMG$CODE + 062F

; 1339          1596  1 !<BLF/PAGE>
```

```
;  1341          1597  1 %SBTTL 'SMG$CREATE_PASTEBOARD - Create Pasteboard'
;  1342          1598  1 GLOBAL ROUTINE SMG$CREATE_PASTEBOARD (
;  1343          1599  1                                       NEW_PBID,
;  1344          1600  1                                       OUT_DEVICE,
;  1345          1601  1                                       PB_ROWS,
;  1346          1602  1                                       PB_COLS,
;  1347          1603  1                                       PRESERVE_SCREEN_FLAG
;  1348          1604  1                                           ) =
;  1349          1605  1 !++
;  1350          1606  1 ! FUNCTIONAL DESCRIPTION:
;  1351          1607  1 !
;  1352          1608  1 !       This routine creates a new pasteboard -- returning its assigned
;  1353          1609  1 !       pasteboard_id.  OUT_DEVICE is the device upon which this
;  1354          1610  1 !       pasteboard is to be written.  If not supplied, output will flow
;  1355          1611  1 !       to SYS$OUTPUT.
;  1356          1612  1 !
;  1357          1613  1 !       If PB_ROWS and/or PB_COLS are provided, they are filled in with
;  1358          1614  1 !       the number of rows and number of columns on the physical device.
;  1359          1615  1 !
;  1360          1616  1 !       If called upon to create a 2nd pasteboard on a device that
;  1361          1617  1 !       already has a pasteboard associated with it, we simply return
;  1362          1618  1 !       the id of the already-existing pasteboard and the qualified
;  1363          1619  1 !       success SMG$_PASALREXI.
;  1364          1620  1 !
;  1365          1621  1 ! CALLING SEQUENCE:
;  1366          1622  1 !
;  1367          1623  1 !       ret_status.wlc.v = SMG$CREATE_PASTEBOARD (
;  1368          1624  1 !                                       NEW_PBID.wl.r
;  1369          1625  1 !                                       [,OUT_DEVICE.rt.dx]
;  1370          1626  1 !                                       [,PB_ROWS.wl.r]
;  1371          1627  1 !                                       [,PB_COLS.wl.r]
;  1372          1628  1 !                                       [,PRESERVE_SCREEN_FLAG.rl.r])
;  1373          1629  1 !
;  1374          1630  1 ! FORMAL PARAMETERS:
;  1375          1631  1 !
;  1376          1632  1 !       NEW_PBID.wl.r     Pasteboard id of newly-created pasteboard.
;  1377          1633  1 !
;  1378          1634  1 !       OUT_DEVICE.rt.dx        [Optional].  If supplied, this parameter
;  1379          1635  1 !                               is the file specification or logical
;  1380          1636  1 !                               name upon which the output associated
;  1381          1637  1 !                               with this pasteboard will be written.
;  1382          1638  1 !                               If omitted, output goes to SYS$OUTPUT.
;  1383          1639  1 !
;  1384          1640  1 !       PB_ROWS         [Optional].  If provided, it is filled in with
;  1385          1641  1 !                       the number of rows on the physical device.
;  1386          1642  1 !
;  1387          1643  1 !       PB_COLS         [Optional].  If provided, it is filled in with
;  1388          1644  1 !                       the number of columns on the physical device.
;  1389          1645  1 !
;  1390          1646  1 !       PRESERVE_SCREEN_FLAG    [Optional].  If provided, and if has a
;  1391          1647  1 !                               value of 1, then the screen will not
;  1392          1648  1 !                               be initially cleared.
;  1393          1649  1 !
;  1394          1650  1 !
;  1395          1651  1 ! IMPLICIT INPUTS:
;  1396          1652  1 !
;  1397          1653  1 !       NONE
```

```
;  1398          1654  1 !
;  1399          1655  1 ! IMPLICIT OUTPUTS:
;  1400          1656  1 !
;  1401          1657  1 !       NONE
;  1402          1658  1 !
;  1403          1659  1 ! COMPLETION STATUS:
;  1404          1660  1 !
;  1405          1661  1 !       SS$_NORMAL      Normal successful completion
;  1406          1662  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
;  1407          1663  1 !                       buffer.
;  1408          1664  1 !       SMG$_PASALREXI  Pasteboard already exists for this device
;  1409          1665  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  1410          1666  1 !
;  1411          1667  1 ! SIDE EFFECTS:
;  1412          1668  1 !
;  1413          1669  1 !       NONE
;  1414          1670  1 !--
```

```
;  1416     1671  2     BEGIN
;  1417     1672  2
;  1418     1673  2     BUILTIN
;  1419     1674  2         NULLPARAMETER;
;  1420     1675  2
;  1421     1676  2     LOCAL
;  1422     1677  2         FS_LEN : WORD INITIAL (0),      ! Length of filespec name to use.
;  1423     1678  2
;  1424     1679  2         FS_ADDR,                    ! Address of filespec name to use.
;  1425     1680  2
;  1426     1681  2         STATUS,                     ! Status of subroutine calls
;  1427     1682  2
;  1428     1683  2         TERM_TYPE,                  ! terminal type
;  1429     1684  2
;  1430     1685  2         CLEAR_FLAG,                 ! TRUE means clear screen
;  1431     1686  2
;  1432     1687  2         PBID,                       ! Id of pasteboard being created.
;  1433     1688  2
;  1434     1689  2         PBCB : REF $PBCB_DECL;      ! Address of pasteboard control block
;  1435     1690  2                                     ! being created.
;  1436     1691  2
;  1437     1692  2     EXTERNAL ROUTINE
;  1438     1693  2
;  1439     1694  2         SMG$$ERASE_PASTEBOARD,
;  1440     1695  2         SMG$$OUT_OF_BAND_HANDLER;
;  1441     1696  2
;  1442     1697  2     $SMG$VALIDATE_ARGCOUNT (1, 5);      ! Test for right no. of args
;  1443     1698  2
;  1444     1699  2     $SMG$GET_NEXT_PID ( PBID);  ! Allocate a new PBID
;  1445     1700  2
;  1446     1701  2     !+
;  1447     1702  2     ! Decide what output device is to receive the the output of this
;  1448     1703  2     ! pasteboard.
;  1449     1704  2     !-
;  1450     1705  2         FS_LEN  = %CHARCOUNT  ('SYS$OUTPUT');       ! Assume default
;  1451     1706  2         FS_ADDR = UPLIT (BYTE ('SYS$OUTPUT'));
;  1452     1707  2
;  1453     1708  2         IF NOT NULLPARAMETER (OUT_DEVICE)
;  1454     1709  2         THEN
;  1455     1710  3             BEGIN   ! User-supplied filespec
;  1456     1711  4             IF NOT (STATUS = LIB$ANALYZE_SDESC_R2 ( .OUT_DEVICE ;
;  1457     1712  4                                                     FS_LEN, FS_ADDR))
;  1458     1713  3             THEN
;  1459     1714  3                 RETURN ( .STATUS);
;  1460     1715  2             END;    ! User-supplied filespec
;  1461     1716  2
;  1462     1717  2     !+
;  1463     1718  2     ! Create a PBCB.  Allocate buffers, etc.
;  1464     1719  2     ! Extract the necessary device attributes and store in PBCB.
;  1465     1720  2     !-
;  1466     1721  2         STATUS = SMG$$SETUP_TERMINAL_TYPE (
;  1467     1722  2                     .FS_ADDR,                   ! filespec addr
;  1468     1723  2                     .FS_LEN,                    ! Len of filespec
;  1469     1724  2                      TERM_TYPE,                 ! Gets terminal type
;  1470     1725  2                      PBCB);                     ! Address to receive address of PBCB
;  1471     1726  2
;  1472     1727  2         IF NOT .STATUS
```

```
;  1473     1728  2      THEN
;  1474     1729  3          BEGIN
;  1475     1730  3          PBD_V_PB_AVAIL [.PBID] = 0;      ! Release PBID number
;  1476     1731  4          RETURN (.STATUS)
;  1477     1732  2          END;
;  1478     1733  2
;  1479     1734  2      !+
;  1480     1735  2      ! Decide whether we want to handle this output device ourselves
;  1481     1736  2      ! or use RMS to handle it.  We use RMS if the output device is
;  1482     1737  2      ! not a terminal.  We also use RMS if the output device is a terminal,
;  1483     1738  2      ! but one we can't handle, such as a hardcopy terminal.
;  1484     1739  2      !-
;  1485     1740  2
;  1486     1741  2      PBCB [PBCB_V_RMS] = (.PBCB[PBCB_B_CLASS]   NEQ DC$_TERM ) OR
;  1487     1742  2                          (.PBCB[PBCB_B_DEVTYPE] EQL UNKNOWN  ) OR
;  1488     1743  2                          (.PBCB[PBCB_B_DEVTYPE] EQL HARDCOPY );
;  1489     1744  2
;  1490     1745  2      !+
;  1491     1746  2      ! Loop through all the pasteboards we currently have trying to find
;  1492     1747  2      ! one whose associated resultant name string is the same as the one we
;  1493     1748  2      ! just created.
;  1494     1749  2      ! If we can find one, we have just created a 2nd pasteboard for the same
;  1495     1750  2      ! physical device and we want to get rid of the pasteboard we just
;  1496     1751  2      ! created and return to the caller the id of the pasteboard that already
;  1497     1752  2      ! exists for this device.
;  1498     1753  2      ! We do this only if the output device is a terminal.
;  1499     1754  2      ! If the output device is a file, we assume that the user wants
;  1500     1755  2      ! to create a new file for each pasteboard he creates.
;  1501     1756  2      !-
;  1502     1757  2
;  1503     1758  2      IF NOT .PBCB [PBCB_V_RMS]
;  1504     1759  2      THEN
;  1505     1760  2          INCR I FROM 0 TO .PBD_L_COUNT -1
;  1506     1761  2          DO
;  1507     1762  3              BEGIN   ! Loop thru pasteboards
;  1508     1763  3              LOCAL
;  1509     1764  3                  SEARCH_PBCB : REF $PBCB_DECL;        ! Addr of pasteboard
;  1510     1765  3                                                       ! control blocks that
;  1511     1766  3                                                       ! we are inspecting.
;  1512     1767  3
;  1513     1768  3              IF (SEARCH_PBCB = .PBD_A_PBCB [.I]) NEQ 0
;  1514     1769  3              THEN
;  1515     1770  4                  BEGIN       ! A valid pasteboard address
;  1516     1771  4                  IF .SEARCH_PBCB [PBCB_W_DEVNAM_LEN] EQL
;  1517     1772  4                     .PBCB        [PBCB_W_DEVNAM_LEN]
;  1518     1773  4                  THEN
;  1519     1774  5                      BEGIN ! Lengths match
;  1520     1775  5                      IF CH$EQL ( .SEARCH_PBCB [PBCB_W_DEVNAM_LEN],   ! length
;  1521     1776  5                                  SEARCH_PBCB [PBCB_T_DEVNAM],        ! addr
;  1522     1777  5                                 .PBCB [PBCB_W_DEVNAM_LEN],           ! length
;  1523     1778  5                                  PBCB [PBCB_T_DEVNAM])               ! addr
;  1524     1779  5                      THEN
;  1525     1780  6                          BEGIN       ! Match found
;  1526     1781  6                          LOCAL
;  1527     1782  6                              STATUS;         ! Local status of subr. calls
;  1528     1783  6
;  1529     1784  6                          !+
```

```
;  1530     1785  6                    ! This physical device already has a pasteboard
;  1531     1786  6                    ! associated with it.
;  1532     1787  6                    ! Get rid of the one we just created.
;  1533     1788  6                    ! First return PBID number we consumed -- we won't
;  1534     1789  6                    ! be using it.
;  1535     1790  6                    !-
;  1536     1791  6                    PBD_V_PB_AVAIL [.PBID] = 0;
;  1537     1792  6
;  1538     1793  6                    !+
;  1539     1794  6                    ! Second deallocate the WCB that got allocated.
;  1540     1795  6                    !-
;  1541     1796  6                    IF .PBCB [PBCB_A_WCB] NEQ 0
;  1542     1797  6                    THEN
;  1543     1798  7                        IF NOT (STATUS = SMG$$DEALLOCATE_WCB (
;  1544     1799  7                                         .PBCB [PBCB_A_WCB]) )
;  1545     1800  6                        THEN
;  1546     1801  6                            RETURN (.STATUS);
;  1547     1802  6
;  1548     1803  6                    !+
;  1549     1804  6                    ! Next release output buffer.
;  1550     1805  6                    !-
;  1551     1806  6                    IF .PBCB [PBCB_A_OUTPUT_BUFFER] NEQ 0
;  1552     1807  6                    THEN
;  1553     1808  7                        IF NOT ( STATUS = LIB$FREE_VM (
;  1554     1809  7                                  %REF (.PBCB [PBCB_W_OUTPUT_BUFSIZ]),
;  1555     1810  7                                  PBCB [PBCB_A_OUTPUT_BUFFER]))
;  1556     1811  6                        THEN
;  1557     1812  6                            RETURN (.STATUS);
;  1558     1813  6
;  1559     1814  6                    !+
;  1560     1815  6                    ! Finally release the PBCB itself.
;  1561     1816  6                    !-
;  1562     1817  7                    IF NOT (STATUS = LIB$FREE_VM ( %REF (PBCB_K_SIZE),
;  1563     1818  7                                                   PBCB))
;  1564     1819  6                    THEN
;  1565     1820  6                        RETURN (.STATUS);
;  1566     1821  6
;  1567     1822  6                    !+
;  1568     1823  6                    ! Return as an id the id of the one that already
;  1569     1824  6                    ! exists.
;  1570     1825  6                    !-
;  1571     1826  6                    .NEW_PBID = .SEARCH_PBCB [PBCB_L_PBID];
;  1572     1827  6
;  1573     1828  6                    !+
;  1574     1829  6                    ! If caller requested number of rows and columns on
;  1575     1830  6                    ! device, tell him.
;  1576     1831  6                    !-
;  1577     1832  6                    IF NOT NULLPARAMETER (PB_ROWS)
;  1578     1833  6                    THEN .PB_ROWS = .SEARCH_PBCB [PBCB_B_ROWS];
;  1579     1834  6
;  1580     1835  6                    IF NOT NULLPARAMETER (PB_COLS)
;  1581     1836  6                    THEN .PB_COLS = .SEARCH_PBCB [PBCB_W_WIDTH];
;  1582     1837  6
;  1583     1838  6                    RETURN ( SMG$_PASALREXI );
;  1584     1839  5                    END;          ! Match found
;  1585     1840  4                END;  ! Lengths match
;  1586     1841  3            END;      ! A valid pasteboard address
```

```
;  1587          1842  2          END;    ! Loop thru pasteboards
;  1588          1843  2
;  1589          1844  2      !+
;  1590          1845  2      ! If we fall out of loop, none of our current pasteboards are pasted to
;  1591          1846  2      ! the same device.  Continue with the creation process.
;  1592          1847  2      ! Store pasteboard id in the PBCB itself.
;  1593          1848  2      !-
;  1594          1849  2          PBCB [PBCB_L_PBID] = .PBID;
;  1595          1850  2
;  1596          1851  2      !+
;  1597          1852  2      ! Store the original name (that the user specified) for this device
;  1598          1853  2      ! in the PBCB.  This name may include a filename as well as a
;  1599          1854  2      ! device name.
;  1600          1855  2      ! First we allocate virtual memory for this buffer and
;  1601          1856  2      ! then we store the length and address in the PBCB for future reference.
;  1602          1857  2      !-
;  1603          1858  2
;  1604          1859  2          STATUS=LIB$GET_VM(%REF(.FS_LEN), PBCB[PBCB_A_OUTNAM]);
;  1605          1860  2          IF NOT .STATUS THEN RETURN (.STATUS);
;  1606          1861  2          PBCB[PBCB_W_OUTNAM_LEN]=.FS_LEN;
;  1607          1862  2          CH$MOVE(.FS_LEN,.FS_ADDR,.PBCB[PBCB_A_OUTNAM]);
;  1608          1863  2
;  1609          1864  2      !+
;  1610          1865  2      ! If device is a terminal, assign a channel to it.
;  1611          1866  2      ! If the device is not a terminal, allocate a FAB and RAB
;  1612          1867  2      ! and open the file for output using RMS.
;  1613          1868  2      !-
;  1614          1869  2
;  1615          1870  2      IF .PBCB[PBCB_V_RMS]
;  1616          1871  3        THEN  BEGIN   ! use RMS to open output
;  1617          1872  3
;  1618          1873  3              !+
;  1619          1874  3              ! Allocate a FAB and RAB to be used to talk to this file.
;  1620          1875  3              !-
;  1621          1876  3
;  1622          1877  3              STATUS=LIB$GET_VM(%REF(FAB$C_BLN),PBCB[PBCB_A_FAB]);
;  1623          1878  3              IF NOT .STATUS THEN RETURN .STATUS;
;  1624          1879  3
;  1625          1880  3              STATUS=LIB$GET_VM(%REF(RAB$C_BLN),PBCB[PBCB_A_RAB]);
;  1626          1881  3              IF NOT .STATUS THEN RETURN .STATUS;
;  1627          1882  3
;  1628          1883  3              !+
;  1629          1884  3              ! Allocate a record buffer.
;  1630          1885  3              ! This will be one byte larger than the width of
;  1631          1886  3              ! the pasteboard because sometimes we will prepend
;  1632          1887  3              ! a formfeed to the record.
;  1633          1888  3              !-
;  1634          1889  3
;  1635          1890  3              STATUS=LIB$GET_VM(%REF(.PBCB[PBCB_W_WIDTH]+1),PBCB[PBCB_A_RBF]);
;  1636          1891  3              IF NOT .STATUS THEN RETURN .STATUS;
;  1637          1892  3
;  1638          1893  3              !+
;  1639          1894  3              ! Initialize the FAB and RAB.
;  1640          1895  3              !-
;  1641          1896  3
;  1642        P 1897  3              $FAB_INIT(      FAB     = .PBCB[PBCB_A_FAB],
;  1643        P 1898  3                              DNM     = 'SMGOUTPUT.LIS',      ! default filename
```

```
;  1644        P 1899  3                       CTX     = .PBCB,        ! why not? pass the PBCB as user context
;  1645        P 1900  3                       FAC     = PUT,          ! write access only
;  1646        P 1901  3                       FNA     = .FS_ADDR,
;  1647        P 1902  3                       FNS     = .FS_LEN,
;  1648        P 1903  3                       ORG     = SEQ,          ! sequential file
;  1649        P 1904  3                       FOP     = SQO,          ! sequential operations only
;  1650        P 1905  3                       RAT     = CR,           ! carriage control
;  1651        P 1906  3                       RFM     = VAR,          ! variable length records
;  1652          1907  3                       MRS     = .PBCB[PBCB_W_WIDTH]+1); ! max record size
;  1653          1908  3
;  1654        P 1909  3               $RAB_INIT(      RAB     = .PBCB[PBCB_A_RAB],
;  1655        P 1910  3                       CTX     = .PBCB,        ! pass the PBCB as user context
;  1656        P 1911  3                       FAB     = .PBCB[PBCB_A_FAB],
;  1657        P 1912  3                       RBF     = .PBCB[PBCB_A_RBF],
;  1658          1913  3                       RAC     = SEQ);         ! sequential output
;  1659          1914  3
;  1660          1915  3               !+
;  1661          1916  3               ! Open the file for output.
;  1662          1917  3               !-
;  1663          1918  3
;  1664          1919  3               STATUS=$CREATE( FAB     = .PBCB[PBCB_A_FAB]);
;  1665          1920  3               IF NOT .STATUS THEN RETURN .STATUS;
;  1666          1921  3
;  1667          1922  3               !+
;  1668          1923  3               ! Connect a record stream to the file.
;  1669          1924  3               !-
;  1670          1925  3
;  1671          1926  3               STATUS=$CONNECT( RAB    = .PBCB[PBCB_A_RAB]);
;  1672          1927  3               IF NOT .STATUS THEN RETURN .STATUS;
;  1673          1928  3
;  1674          1929  3               END     ! use RMS to open output
;  1675          1930  3       ELSE    BEGIN   ! assigning channel
;  1676          1931  3
;  1677          1932  3               LOCAL   NAME_DESC       : VECTOR[2],    ! Fixed length descriptor
;  1678          1933  3                       ASYNC_EFN       : LONG,         ! Longword to hold efn
;  1679          1934  3                       TTIOSB          : VECTOR[4,WORD],! IOSB for SENSE MODE
;  1680          1935  3                       CHARBUF         : BLOCK[12,BYTE];! 12-byte characteristics buffer
;  1681          1936  3
;  1682          1937  3               !+
;  1683          1938  3               ! Create a fixed length descriptor for our device name string
;  1684          1939  3               ! for use by $ASSIGN.
;  1685          1940  3               !-
;  1686          1941  3
;  1687          1942  3               NAME_DESC[0]=.PBCB[PBCB_W_DEVNAM_LEN];
;  1688          1943  3               NAME_DESC[1]= PBCB[PBCB_T_DEVNAM];
;  1689          1944  3
;  1690          1945  3               !+
;  1691          1946  3               ! Assign the channel.
;  1692          1947  3               ! Put the resulting channel number in PBCB[PBCB_W_CHAN].
;  1693          1948  3               !-
;  1694          1949  3
;  1695        P 1950  3               STATUS=$ASSIGN( DEVNAM  = NAME_DESC,
;  1696          1951  3                               CHAN    = PBCB[PBCB_W_CHAN]);
;  1697          1952  3               IF NOT .STATUS THEN RETURN .STATUS;
;  1698          1953  3
;  1699          1954  3               !+
;  1700          1955  3               ! Assign an asynchronous event flag.
```

```
;  1701          1956  3          !-
;  1702          1957  3
;  1703          1958  3          STATUS=LIB$GET_EF(ASYNC_EFN);
;  1704          1959  3          IF NOT .STATUS THEN RETURN .STATUS;
;  1705          1960  3
;  1706          1961  3          !+
;  1707          1962  3          ! Store the value into a byte in the PBCB.
;  1708          1963  3          !-
;  1709          1964  3
;  1710          1965  3          PBCB [PBCB_B_ASYNC_EFN] = .ASYNC_EFN;
;  1711          1966  3
;  1712          1967  3          !+
;  1713          1968  3          ! Do a SENSE MODE QIO to get additional characteristics
;  1714          1969  3          ! of interest.
;  1715          1970  3          ! Ignore everything returned in the characteristics buffer.
;  1716          1971  3          ! (We already got that stuff.)
;  1717          1972  3          ! The I/O status block has neat things of interest.
;  1718          1973  3          !-
;  1719          1974  3
;  1720        P 1975  3          STATUS=$QIOW(   CHAN    = .PBCB[PBCB_W_CHAN],
;  1721        P 1976  3                          FUNC    = IO$_SENSEMODE,
;  1722        P 1977  3                          IOSB    = TTIOSB,
;  1723        P 1978  3                          P1      = CHARBUF,
;  1724          1979  3                          P2      = 12);
;  1725          1980  3          IF NOT .STATUS THEN RETURN .STATUS;
;  1726          1981  3          IF NOT .TTIOSB[0] THEN RETURN .TTIOSB[0];
;  1727          1982  3
;  1728          1983  3          PBCB [PBCB_W_SPEED]  = .TTIOSB[1];
;  1729          1984  3          PBCB [PBCB_W_FILL]   = .TTIOSB[2];
;  1730          1985  3          PBCB [PBCB_B_PARITY] = .TTIOSB[3]
;  1731          1986  3
;  1732          1987  2          END;    ! assigning channel
;  1733          1988  2
;  1734          1989  2  !+
;  1735          1990  2  ! Set up our exit block which is contained within the PBCB.
;  1736          1991  2  ! This exit block is used to establish an exit handler for
;  1737          1992  2  ! this terminal. When the exit handler is called,
;  1738          1993  2  ! it will flush the output buffers.
;  1739          1994  2  ! This guarantees that the user will see all his output even if
;  1740          1995  2  ! his program exits and he doesn't manually flush the buffers.
;  1741          1996  2  !-
;  1742          1997  2
;  1743          1998  2      PBCB [PBCB_A_EXIT_ADDR] = SMG$$PBCB_EXIT_HANDLER;
;                                                      ! Address of our exit handler
;  1744          1999  2
;  1745          2000  2      PBCB [PBCB_B_EXIT_ARGCNT] = 2;
;                                                      ! Our exit handler gets called with
;  1746          2001  2                                 ! two arguments.
;  1747          2002  2      PBCB [PBCB_A_EXIT_RSN] = PBCB [PBCB_L_EXIT_REASON];
;                                                      ! The first argument is the address
;  1748          2003  2                                 ! of the longword to receive the
;  1749          2004  2                                 ! exit reason.  This longword appears
;  1750          2005  2                                 ! elsewhere in the PBCB (not in
;  1751          2006  2                                 ! the exit block).
;  1752          2007  2
;  1753          2008  2      PBCB [PBCB_A_EXIT_PBCB] = .PBCB;
;                                                      ! The second argument is the address
;  1754          2009  2                                 ! of this PBCB.  This is needed
;  1755          2010  2                                 ! because there are many PBCBs and
;  1756          2011  2                                 ! one exit routine serves them all.
;  1757          2012  2                                 ! There is a separate exit block for
```

```
;  1758          2013  2                                               ! each pasteboard.
;  1759          2014  2
;  1760          2015  2 !+
;  1761          2016  2 ! Establish the exit handler, using the exit block just created.
;  1762          2017  2 !-
;  1763          2018  2
;  1764          2019  2     STATUS=$DCLEXH(DESBLK=PBCB [PBCB_R_EXIT_BLOCK]);
;  1765          2020  2     IF NOT .STATUS THEN RETURN .STATUS;
;  1766          2021  2
;  1767          2022  2 !+
;  1768          2023  2 ! Now we do an incredible strange thing.
;  1769          2024  2 ! We build a 10-byte routine in the PBCB to service out-of-band ASTs.
;  1770          2025  2 ! The routine has the form:
;  1771          2026  2 !
;  1772          2027  2 !       0000    entry mask
;  1773          2028  2 !         FA    CALLG
;  1774          2029  2 !         6C    (AP)
;  1775          2030  2 !         9F    absolute addressing
;  1776          2031  2 !    address    longword address of SMG$$OUT_OF_BAND_HANDLER
;  1777          2032  2 !         04    RET
;  1778          2033  2 !
;  1779          2034  2 ! Symbolically, the routine looks as follows:
;  1780          2035  2 !
;  1781          2036  2 !    ROUTINE BAND_HANDLER =
;  1782          2037  2 !       BEGIN
;  1783          2038  2 !       EXTERNAL ROUTINE SMG$$OUT_OF_BAND_HANDER : ADDRESSING_MODE(ABSOLUTE);
;  1784          2039  2 !       BUILTIN AP,CALLG;
;  1785          2040  2 !       RETURN CALLG(.AP,SMG$$OUT_OF_BAND_HANDLER);
;  1786          2041  2 !       END;
;  1787          2042  2 !
;  1788          2043  2 ! However, we don't actually create this routine in BLISS and then
;  1789          2044  2 ! move it into our structure, because we can't be guaranteed that
;  1790          2045  2 ! BLISS will continue to generate the same code in future releases.
;  1791          2046  2 ! Thus we create the entire routine ourselves.
;  1792          2047  2 ! This code would have to change if we ever tried to run this
;  1793          2048  2 ! on a machine with a new arhitecture.
;  1794          2049  2 !-
;  1795          2050  2
;  1796          2051  2     PBCB[PBCB_W_ENTRY_MASK]     = %X'0000';
;  1797          2052  2     PBCB[PBCB_B_CALLG]          = %X   'FA';
;  1798          2053  2     PBCB[PBCB_B_REG_AP]         = %X   '6C';
;  1799          2054  2     PBCB[PBCB_B_ABS]            = %X   '9F';
;  1800          2055  2     PBCB[PBCB_A_BAND_HANDLER]   = SMG$$OUT_OF_BAND_HANDLER;
;  1801          2056  2     PBCB[PBCB_B_RET]            = %X   '04';
;  1802          2057  2
;  1803          2058  2 !+
;  1804          2059  2 ! Since all went well, we can now adjust the count of how many PBCB's
;  1805          2060  2 ! we have and plug its address into the pasteboard directory.
;  1806          2061  2 !-
;  1807          2062  2     PBD_L_COUNT = .PBD_L_COUNT + 1;
;  1808          2063  2
;  1809          2064  2     PBD_A_PBCB [.PBID] = .PBCB;
;  1810          2065  2
;  1811          2066  2 !+
;  1812          2067  2 ! Initially clear the screen (unless we are asked to preserve it).
;  1813          2068  2 !-
;  1814          2069  2
```

```
;  1815          2070  2      CLEAR_FLAG=1;
;  1816          2071  2      IF NOT NULLPARAMETER(PRESERVE_SCREEN_FLAG)
;  1817          2072  2      THEN  CLEAR_FLAG=NOT ..PRESERVE_SCREEN_FLAG;
;  1818          2073  2
;  1819          2074  2      IF .CLEAR_FLAG
;  1820          2075  2      THEN
;  1821          2076  2          BEGIN
;  1822          2077  2          STATUS=SMG$$ERASE_PASTEBOARD(.PBCB);
;  1823          2078  2          IF NOT .STATUS THEN RETURN .STATUS;
;  1824          2079  2          END
;  1825          2080  2      ELSE
;  1826          2081  2          BEGIN   ! Just pretend we cleared the screen.
;  1827          2082  2          LOCAL   WCB     : REF $WCB_DECL;
;  1828          2083  2          WCB=.PBCB[PBCB_A_WCB];
;  1829          2084  2
;  1830          2085  2          CH$FILL(%C' ',.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_TEXT_BUF]);
;  1831          2086  2          CH$FILL(%C' ',.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_SCR_TEXT_BUF]);
;  1832          2087  2          CH$FILL(0,    .WCB[WCB_L_BUFSIZE],.WCB[WCB_A_ATTR_BUF]);
;  1833          2088  2          CH$FILL(0,    .WCB[WCB_L_BUFSIZE],.WCB[WCB_A_SCR_ATTR_BUF]);
;  1834          2089  2
;  1835          2090  2          IF .WCB[WCB_A_CHAR_SET_BUF] NEQ 0
;  1836          2091  2          THEN
;  1837          2092  4                  BEGIN
;  1838          2093  4                  CH$FILL(0,.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_CHAR_SET_BUF]);
;  1839          2094  4                  END;
;  1840          2095  2
;  1841          2096  2          IF .WCB[WCB_A_SCR_CHAR_SET_BUF] NEQ 0
;  1842          2097  2          THEN
;  1843          2098  4                  BEGIN
;  1844          2099  4                  CH$FILL(0,.WCB[WCB_L_BUFSIZE],.WCB[WCB_A_SCR_CHAR_SET_BUF]);
;  1845          2100  4                  END;
;  1846          2101  2
;  1847          2102  2          !+
;  1848          2103  2          ! The physical cursor moves to (1,1).
;  1849          2104  2          !-
;  1850          2105  2
;  1851          2106  2          ! WCB[WCB_W_CURR_CUR_ROW]=1;
;  1852          2107  2          ! WCB[WCB_W_OLD_CUR_ROW] =1;
;  1853          2108  2          ! WCB[WCB_W_CURR_CUR_COL]=1;
;  1854          2109  2          ! WCB[WCB_W_OLD_CUR_COL] =1;
;  1855          2110  2
;  1856          2111  2          !+
;  1857          2112  2          ! The line characteristics get set back to 0.
;  1858          2113  2          !-
;  1859          2114  2
;  1860          2115  2          CH$FILL(0,.WCB[WCB_W_NO_ROWS]+1,.WCB[WCB_A_LINE_CHAR]);
;  1861          2116  2          CH$FILL(0,.WCB[WCB_W_NO_ROWS]+1,.WCB[WCB_A_SCR_LINE_CHAR]);
;  1862          2117  2
;  1863          2118  2          END;    ! Just pretend we cleared the screen
;  1864          2119  2
;  1865          2120  2  !+
;  1866          2121  2  ! If caller is interested in number of rows and columns on device, tell
;  1867          2122  2  ! him.
;  1868          2123  2  !-
;  1869          2124  2      IF NOT NULLPARAMETER(PB_ROWS) THEN .PB_ROWS = .PBCB [PBCB_B_ROWS];
;  1870          2125  2      IF NOT NULLPARAMETER(PB_COLS) THEN .PB_COLS = .PBCB [PBCB_W_WIDTH];
;  1871          2126  2
```

```
; 1872          2127  2 !+
; 1873          2128  2 ! Return the new pasteboard id to caller
; 1874          2129  2 !-
; 1875          2130  2     .NEW_PBID = .PBID;
; 1876          2131  2
; 1877          2132  3     RETURN (SS$_NORMAL)
; 1878          2133  1     END;                                ! Routine SMG$CREATE_PASTEBOARD


                                                          006A6           .BLKB    2
                54 55 50 54 55 4F 24 53 59 53  006A8  P.AAA:   .ASCII   \SYS$OUTPUT\                                  ;
       53 49 4C 2E 54 55 50 54 55 4F 47 4D 53  006B2  P.AAB:   .ASCII   \SMGOUTPUT.LIS\                               ;

                                                                   .EXTRN   SMG$$OUT_OF_BAND_HANDLER
                                                                   .EXTRN   SYS$CREATE, SYS$CONNECT
                                                                   .EXTRN   SYS$ASSIGN, SYS$QIOW
                                                                   .EXTRN   SYS$DCLEXH

                                                 0FFC 00000        .ENTRY   SMG$CREATE_PASTEBOARD, Save R2,R3,R4,R5,R6,-; 1598
                                                                            R7,R8,R9,R10,R11                              ;
                            5E            2C   C2 00002            SUBL2    #44, SP                                       ;
                                          58   B4 00005            CLRW     FS_LEN                                        ; 1671
                   50       6C            01   83 00007            SUBB3    #1, (AP), DIFF                                ; 1697
                            04            50   91 0000B            CMPB     DIFF, #4                                      ;
                                          08   1B 0000E            BLEQU    1$                                            ;
                            50 00000000G  8F   D0 00010            MOVL     #SMG$_WRONUMARG, R0                           ;
                                               04 00017            RET                                                    ;
                   50 00000000'  EF       01   C1 00018  1$:       ADDL3    #1, PBD_L_COUNT, R0                           ; 1699
                            10            50   D1 00020            CMPL     R0, #16                                       ;
                                          08   15 00023            BLEQ     2$                                            ;
                            50 00000000G  8F   D0 00025            MOVL     #SMG$_TOOMANPAS, R0                           ;
                                               04 0002C            RET                                                    ;
 5A 00000000'  EF 00000000'  EF 00000000'  EF  EB 0002D  2$:       FFC      ZERO, PBD_K_MAX_PB_BY_REF, PBD_V_PB_AVAIL, -;
                                                                            PBID                                          ;
                   00 00000000'  EF       5A   E2 0003E            BBSS     PBID, PBD_V_PB_AVAIL, 3$                      ;
                            58            0A   B0 00046  3$:       MOVW     #10, FS_LEN                                   ; 1705
                            5B       9D   AF   9E 00049            MOVAB    P.AAA, FS_ADDR                                ; 1706
                            02            6C   91 0004D            CMPB     (AP), #2                                      ; 1708
                            1B                 1F 00050            BLSSU    4$                                            ;
                                     08   AC   D5 00052            TSTL     8(AP)                                         ;
                            16                 13 00055            BEQL     4$                                            ;
                   50                08   AC   D0 00057            MOVL     OUT_DEVICE, R0                                ; 1711
                              00000000G   00   16 0005B            JSB      LIB$ANALYZE_SDESC_R2                          ;
                            59            50   D0 00061            MOVL     R0, STATUS                                    ;
                            5B            52   D0 00064            MOVL     R2, R11                                       ;
                            58            51   D0 00067            MOVL     R1, FS_LEN                                    ;
                            20            59   E9 0006A            BLBC     STATUS, 5$                                    ;
                                     08   AE   9F 0006D  4$:       PUSHAB   PBCB                                          ; 1721
                                     08   AE   9F 00070            PUSHAB   TERM_TYPE                                     ;
                            7E            58   3C 00073            MOVZWL   FS_LEN, -(SP)                                 ; 1723
                                          5B   DD 00076            PUSHL    FS_ADDR                                       ; 1722
                      00000000G   00      04   FB 00078            CALLS    #4, SMG$$SETUP_TERMINAL_TYPE                  ;
                            59            50   D0 0007F            MOVL     R0, STATUS                                    ;
                            0B            59   E8 00082            BLBS     STATUS, 6$                                    ; 1727
                   00 00000000'  EF       5A   E5 00085            BBCC     PBID, PBD_V_PB_AVAIL, 5$                      ; 1730
                                        02CC   31 0008D  5$:       BRW      31$                                           ; 1731
```

```
                                    50      08   AE  D0 00090 6$:     MOVL    PBCB, R0                                        ; 1741
                                                 51  D4 00094         CLRL    R1
                                42  8F          58  A0 91 00096      CMPB    88(R0), #66
                                                 02  13 0009B        BEQL    7$
                                                 51  D6 0009D        INCL    R1
                                                 52  D4 0009F 7$:     CLRL    R2                                              ; 1742
                                    10          A0  95 000A1        TSTB    16(R0)
                                                 02  12 000A4        BNEQ    8$
                                                 52  D6 000A6        INCL    R2
                                    52           51  C8 000A8 8$:     BISL2   R1, R2
                                                 51  D4 000AB        CLRL    R1                                              ; 1743
                                    05      10   A0  91 000AD        CMPB    16(R0), #5
                                                 02  12 000B1        BNEQ    9$
                                                 51  D6 000B3        INCL    R1
                        53                       52  89 000B5 9$:     BISB3   R2, R1, R3
OOD0    C0              01                       53  F0 000B9        INSV    R3, #3, #1, 208(R0)
                        03      OOD0     C0      03  E1 000C0        BBC     #3, 208(R0), 10$                                ; 1758
                                         00AB    31  000C6        BRW     22$
                                    57 00000000' EF  D0 000C9 10$:    MOVL    PBD_L_COUNT, R7                                 ; 1760
                                    56           01  CE 000D0        MNEGL   #1, I
                                         0095    31  000D3 11$:     BRW     20$
                                    55 00000000'EF46 D0 000D6 12$:    MOVL    PBD_A_PBCB[I], SEARCH_PBCB                      ; 1768
                                                 F3  13 000DE        BEQL    11$
                                    54      08   AE  D0 000E0        MOVL    PBCB, R4                                        ; 1772
                                12  A4      12   A5  B1 000E4        CMPW    18(SEARCH_PBCB), 18(R4)
                                                 E8  12 000E9        BNEQ    11$
  12    A4              00        18   A5      12   A5  2D 000EB        CMPC5   18(SEARCH_PBCB), 24(SEARCH_PBCB), #0, -         ; 1778
                                                 18  A4      000F3        18(R4), 24(R4)
                                                 74  12 000F5        BNEQ    20$
                        00 00000000'  EF                 5A  E5 000F7        BBCC    PBID, PBD_V_PB_AVAIL, 13$               ; 1791
                                    08          A4  D5 000FF 13$:     TSTL    8(R4)                                           ; 1796
                                                 0B  13 00102        BEQL    14$
                                    08          A4  DD 00104        PUSHL   8(R4)                                           ; 1799
                0000V   CF                       01  FB 00107        CALLS   #1, SMG$$DEALLOCATE_WCB
                        2D                       50  E9 0010C        BLBC    STATUS, 16$                                     ; 1798
                                    6C          A4  D5 0010F 14$:     TSTL    108(R4)                                         ; 1806
                                                 15  13 00112        BEQL    15$
                                    6C          A4  9F 00114        PUSHAB  108(R4)                                         ; 1810
                    04  AE          70          A4  3C 00117        MOVZWL  112(R4), 4(SP)                                  ; 1809
                                    04          AE  9F 0011C        PUSHAB  4(SP)
00000000G           00                           02  FB 0011F        CALLS   #2, LIB$FREE_VM                                 ; 1810
                        13                       50  E9 00126        BLBC    STATUS, 16$
                                    08          AE  9F 00129 15$:     PUSHAB  PBCB                                            ; 1817
                    04  AE      014C            8F  3C 0012C        MOVZWL  #332, 4(SP)
                                    04          AE  9F 00132        PUSHAB  4(SP)
00000000G           00                           02  FB 00135        CALLS   #2, LIB$FREE_VM
                        01                       50  E8 0013C 16$:     BLBS    STATUS, 17$
                                                 04  0013F        RET
                    04  BC          14          A5  D0 00140 17$:     MOVL    20(SEARCH_PBCB), @NEW_PBID                      ; 1826
                        03                       6C  91 00145        CMPB    (AP), #3                                        ; 1832
                                                 0A  1F 00148        BLSSU   18$
                                    0C          AC  D5 0014A        TSTL    12(AP)
                                                 05  13 0014D        BEQL    18$
                    0C  BC          5F          A5  9A 0014F        MOVZBL  95(SEARCH_PBCB), @PB_ROWS                       ; 1833
                        04                       6C  91 00154 18$:     CMPB    (AP), #4                                        ; 1835
                                                 0A  1F 00157        BLSSU   19$
                                    10          AC  D5 00159        TSTL    16(AP)
```

```
                                  05  13 0015C         BEQL    19$                                          ;
             10   BC         5A   A5  3C 0015E         MOVZWL  90(SEARCH_PBCB), @PB_COLS                    ; 1836
                  50  00000000G   8F  D0 00163  19$:   MOVL    #SMG$_PASALREXI, R0                          ; 1838
                                      04 0016A         RET                                                  ;
        02        56              57  F2 0016B  20$:   AOBLSS  R7, I, 21$                                   ; 1760
                                  03  11 0016F         BRB     22$                                          ;
                                FF62  31 00171  21$:   BRW     12$                                          ;
                  57         08   AE  D0 00174  22$:   MOVL    PBCB, R7                                     ; 1849
             14   A7              5A  D0 00178         MOVL    PBID, 20(R7)                                 ;
                           00B0   C7  9F 0017C         PUSHAB  176(R7)                                      ; 1859
             04   AE              58  3C 00180         MOVZWL  FS_LEN, 4(SP)                                ;
                             04   AE  9F 00184         PUSHAB  4(SP)                                        ;
      00000000G   00              02  FB 00187         CALLS   #2, LIB$GET_VM                               ;
                  59              50  D0 0018E         MOVL    R0, STATUS                                   ;
                  5F              59  E9 00191         BLBC    STATUS, 24$                                  ; 1860
           00E4   C7              58  B0 00194         MOVW    FS_LEN, 228(R7)                              ; 1861
   00B0   D7      6B              58  28 00199         MOVC3   FS_LEN, (FS_ADDR), @176(R7)                  ; 1862
          03    00D0   C7         03  E0 0019F         BBS     #3, 208(R7), 23$                             ; 1870
                                00D3  31 001A5         BRW     27$                                          ;
                           00E8   C7  9F 001A8  23$:   PUSHAB  232(R7)                                      ; 1877
             04   AE         50   8F  9A 001AC         MOVZBL  #80, 4(SP)                                   ;
                             04   AE  9F 001B1         PUSHAB  4(SP)                                        ;
      00000000G   00              02  FB 001B4         CALLS   #2, LIB$GET_VM                               ;
                  59              50  D0 001BB         MOVL    R0, STATUS                                   ;
                  32              59  E9 001BE         BLBC    STATUS, 24$                                  ; 1878
                           00EC   C7  9F 001C1         PUSHAB  236(R7)                                      ; 1880
             04   AE         44   8F  9A 001C5         MOVZBL  #68, 4(SP)                                   ;
                             04   AE  9F 001CA         PUSHAB  4(SP)                                        ;
      00000000G   00              02  FB 001CD         CALLS   #2, LIB$GET_VM                               ;
                  59              50  D0 001D4         MOVL    R0, STATUS                                   ;
                  19              59  E9 001D7         BLBC    STATUS, 24$                                  ; 1881
                           00F0   C7  9F 001DA         PUSHAB  240(R7)                                      ; 1890
             04   AE         5A   A7  3C 001DE         MOVZWL  90(R7), 4(SP)                                ;
                             04   AE  D6 001E3         INCL    4(SP)                                        ;
                             04   AE  9F 001E6         PUSHAB  4(SP)                                        ;
      00000000G   00              02  FB 001E9         CALLS   #2, LIB$GET_VM                               ;
                  59              50  D0 001F0         MOVL    R0, STATUS                                   ;
                  70              59  E9 001F3  24$:   BLBC    STATUS, 25$                                  ; 1891
                  56       00E8   C7  D0 001F6         MOVL    232(R7), R6                                  ; 1907
 0050   8F     00   6E            00  2C 001FB         MOVC5   #0, (SP), #0, #80, (R6)                      ;
                                  66     00202                                                              ;
                  66       5003   8F  B0 00203         MOVW    #20483, (R6)                                 ;
             04   A6         40   8F  9A 00208         MOVZBL  #64, 4(R6)                                   ;
             16   A6              01  90 0020D         MOVB    #1, 22(R6)                                   ;
             18   A6              57  D0 00211         MOVL    R7, 24(R6)                                   ;
             1D   A6       0200   8F  B0 00215         MOVW    #512, 29(R6)                                 ;
             1F   A6              02  90 0021B         MOVB    #2, 31(R6)                                   ;
             2C   A6              5B  D0 0021F         MOVL    FS_ADDR, 44(R6)                              ;
             30   A6       FDCC   CF  9E 00223         MOVAB   P.AAB, 48(R6)                                ;
             34   A6              58  90 00229         MOVB    FS_LEN, 52(R6)                               ;
             35   A6              0D  90 0022D         MOVB    #13, 53(R6)                                  ;
        36   A6   5A              A7  01  A1 00231     ADDW3   #1, 90(R7), 54(R6)                           ;
                  58       00EC   C7  D0 00237         MOVL    236(R7), R8                                  ; 1913
 0044   8F     00   6E            00  2C 0023C         MOVC5   #0, (SP), #0, #68, (R8)                      ;
                                  68     00243                                                              ;
                  68       4401   8F  B0 00244         MOVW    #17409, (R8)                                 ;
             18   A8              57  D0 00249         MOVL    R7, 24(R8)                                   ;
```

```
                        1E   A8  94 0024D       CLRB    30(R8)                                          ;
      28   A8         00F0   C7  D0 00250       MOVL    240(R7), 40(R8)                                 ;
      3C   A8                56  D0 00256       MOVL    R6, 60(R8)                                      ;
                             56  DD 0025A       PUSHL   R6                                              ; 1919
00000000G  00                01  FB 0025C       CALLS   #1, SYS$CREATE                                  ;
           59                50  D0 00263       MOVL    R0, STATUS                                      ;
           0F                59  E9 00266 25$:  BLBC    STATUS, 26$                                     ; 1920
                             58  DD 00269       PUSHL   R8                                              ; 1926
00000000G  00                01  FB 0026B       CALLS   #1, SYS$CONNECT                                 ;
           59                50  D0 00272       MOVL    R0, STATUS                                      ;
           6E                59  E8 00275       BLBS    STATUS, 29$                                     ; 1927
                           00E1  31 00278 26$:  BRW     31$                                             ;
      24   AE           12   A7  3C 0027B 27$:  MOVZWL  18(R7), NAME_DESC                               ; 1942
      28   AE           18   A7  9E 00280       MOVAB   24(R7), NAME_DESC+4                             ; 1943
                             7E  7C 00285       CLRQ    -(SP)                                           ; 1951
                        64   A7  9F 00287       PUSHAB  100(R7)                                         ;
                        30   AE  9F 0028A       PUSHAB  NAME_DESC                                       ;
00000000G  00                04  FB 0028D       CALLS   #4, SYS$ASSIGN                                  ;
           59                50  D0 00294       MOVL    R0, STATUS                                      ;
           DE                59  E9 00297       BLBC    STATUS, 26$                                     ; 1952
                        0C   AE  9F 0029A       PUSHAB  ASYNC_EFN                                       ; 1958
00000000G  00                01  FB 0029D       CALLS   #1, LIB$GET_EF                                  ;
           59                50  D0 002A4       MOVL    R0, STATUS                                      ;
           CE                59  E9 002A7       BLBC    STATUS, 26$                                     ; 1959
      67   A7           0C   AE  90 002AA       MOVB    ASYNC_EFN, 103(R7)                              ; 1965
                             7E  7C 002AF       CLRQ    -(SP)                                           ; 1979
                             7E  7C 002B1       CLRQ    -(SP)                                           ;
                             0C  DD 002B3       PUSHL   #12                                             ;
                        24   AE  9F 002B5       PUSHAB  CHARBUF                                         ;
                             7E  7C 002B8       CLRQ    -(SP)                                           ;
                        3C   AE  9F 002BA       PUSHAB  TTIOSB                                          ;
                             27  DD 002BD       PUSHL   #39                                             ;
           7E           64   A7  3C 002BF       MOVZWL  100(R7), -(SP)                                  ;
                             7E  D4 002C3       CLRL    -(SP)                                           ;
00000000G  00                0C  FB 002C5       CALLS   #12, SYS$QIOW                                   ;
           59                50  D0 002CC       MOVL    R0, STATUS                                      ;
           A6                59  E9 002CF       BLBC    STATUS, 26$                                     ; 1980
           05           1C   AE  E8 002D2       BLBS    TTIOSB, 28$                                     ; 1981
           50           1C   AE  3C 002D6       MOVZWL  TTIOSB, R0                                      ;
                                 04 002DA       RET                                                     ;
    00B4   C7           1E   AE  D0 002DB 28$:  MOVL    TTIOSB+2, 180(R7)                               ; 1983
      11   A7           22   AE  90 002E1       MOVB    TTIOSB+6, 17(R7)                                ; 1985
      78   A7    00000000G   00  9E 002E6 29$:  MOVAB   SMG$$PBCB_EXIT_HANDLER, 120(R7)                 ; 1998
      7C   A7                02  90 002EE       MOVB    #2, 124(R7)                                     ; 2000
    0080   C7         0088   C7  9E 002F2       MOVAB   136(R7), 128(R7)                                ; 2002
    0084   C7                57  D0 002F9       MOVL    R7, 132(R7)                                     ; 2006
                        74   A7  9F 002FE       PUSHAB  116(R7)                                         ; 2019
00000000G  00                01  FB 00301       CALLS   #1, SYS$DCLEXH                                  ;
           59                50  D0 00308       MOVL    R0, STATUS                                      ;
           4E                59  E9 0030B       BLBC    STATUS, 31$                                     ; 2020
    008C   C7 6CFA0000       8F  D0 0030E       MOVL    #1828323328, 140(R7)                            ; 2051
    0090   C7           9F   8F  90 00317       MOVB    #-97, 144(R7)                                   ; 2054
    0091   C7    00000000G   00  9E 0031D       MOVAB   SMG$$OUT_OF_BAND_HANDLER, 145(R7)               ; 2055
    0095   C7                04  90 00326       MOVB    #4, 149(R7)                                     ; 2056
                 00000000'   EF  D6 0032B       INCL    PBD_L_COUNT                                     ; 2062
00000000'EF4A                57  D0 00331       MOVL    R7, PBD_A_PBCB[PBID]                            ; 2064
           50                01  D0 00339       MOVL    #1, CLEAR_FLAG                                  ; 2070
```

```
                              05          6C  91 0033C       CMPB    (AP), #5                              ; 2071
                                          09  1F 0033F       BLSSU   30$
                                    14    AC  D5 00341       TSTL    20(AP)
                                          04  13 00344       BEQL    30$
                              50    14    BC  D2 00346       MCOML   @PRESERVE_SCREEN_FLAG, CLEAR_FLAG     ; 2072
                              13          50  E9 0034A 30$:  BLBC    CLEAR_FLAG, 32$                       ; 2074
                                          57  DD 0034D       PUSHL   R7                                    ; 2077
                   00000000G  00          01  FB 0034F       CALLS   #1, SMG$ERASE_PASTEBOARD
                              59          50  D0 00356       MOVL    R0, STATUS
                              54          59  E8 00359       BLBS    STATUS, 35$                           ; 2078
                              50          59  D0 0035C 31$:  MOVL    STATUS, R0
                                              04 0035F       RET
                              59    08    A7  D0 00360 32$:  MOVL    8(R7), WCB                            ; 2083
                              56    28    A9  D0 00364       MOVL    40(WCB), R6                           ; 2085
         56          20       6E          00  2C 00368       MOVC5   #0, (SP), #32, R6, @8(WCB)
                                    08    B9     0036D
         56          20       6E          00  2C 0036F       MOVC5   #0, (SP), #32, R6, @20(WCB)           ; 2086
                                    14    B9     00374
         56          00       6E          00  2C 00376       MOVC5   #0, (SP), #0, R6, @12(WCB)            ; 2087
                                    0C    B9     0037B
         56          00       6E          00  2C 0037D       MOVC5   #0, (SP), #0, R6, @24(WCB)            ; 2088
                                    18    B9     00382
                                    10    A9  D5 00384       TSTL    16(WCB)                               ; 2090
                                          07  13 00387       BEQL    33$
         56          00       6E          00  2C 00389       MOVC5   #0, (SP), #0, R6, @16(WCB)            ; 2093
                                    10    B9     0038E
                                    1C    A9  D5 00390 33$:  TSTL    28(WCB)                               ; 2096
                                          07  13 00393       BEQL    34$
         56          00       6E          00  2C 00395       MOVC5   #0, (SP), #0, R6, @28(WCB)            ; 2099
                                    1C    B9     0039A
                              56    02    A9  3C 0039C 34$:  MOVZWL  2(WCB), R6                            ; 2115
                                          56  D6 003A0       INCL    R6
         56          00       6E          00  2C 003A2       MOVC5   #0, (SP), #0, R6, @44(WCB)
                                    2C    B9     003A7
         56          00       6E          00  2C 003A9       MOVC5   #0, (SP), #0, R6, @48(WCB)            ; 2116
                                    30    B9     003AE
                              03          6C  91 003B0 35$:  CMPB    (AP), #3                              ; 2124
                                          0A  1F 003B3       BLSSU   36$
                                    0C    AC  D5 003B5       TSTL    12(AP)
                                          05  13 003B8       BEQL    36$
                       0C     BC    5F    A7  9A 003BA       MOVZBL  95(R7), @PB_ROWS
                              04          6C  91 003BF 36$:  CMPB    (AP), #4                              ; 2125
                                          0A  1F 003C2       BLSSU   37$
                                    10    AC  D5 003C4       TSTL    16(AP)
                                          05  13 003C7       BEQL    37$
                       10     BC    5A    A7  3C 003C9       MOVZWL  90(R7), @PB_COLS
                       04     BC          5A  D0 003CE 37$:  MOVL    PBID, @NEW_PBID                       ; 2130
                              50          01  D0 003D2       MOVL    #1, R0                                ; 2132
                                              04 003D5       RET                                           ; 2133

; Routine Size:  982 bytes,    Routine Base:  _SMG$CODE + 06BF

; 1879          2134  1 !<BLF/PAGE>
```

```
;  1881          2135  1 %SBTTL 'SMG$DELETE_PASTEBOARD - Delete Pasteboard'
;  1882          2136  1 GLOBAL ROUTINE SMG$DELETE_PASTEBOARD ( PBID, CLEAR_SCREEN_FLAG ) =
;  1883          2137  1
;  1884          2138  1 !++
;  1885          2139  1 ! FUNCTIONAL DESCRIPTION:
;  1886          2140  1 !
;  1887          2141  1 !       This routine terminates all use of a given physical display.
;  1888          2142  1 !       It deallocates the pasteboard control block and all its
;  1889          2143  1 !       substructures.  It gets rid of the event flag and the channel
;  1890          2144  1 !       number.  It removes any associated exit handler.
;  1891          2145  1 !
;  1892          2146  1 ! CALLING SEQUENCE:
;  1893          2147  1 !
;  1894          2148  1 !       ret_status.wlc.v = SMG$DELETE_PASTEBOARD ( PBID.rl.r
;  1895          2149  1 !                                                  [,CLEAR_SCREEN_FLAG.rl.r])
;  1896          2150  1 !
;  1897          2151  1 ! FORMAL PARAMETERS:
;  1898          2152  1 !
;  1899          2153  1 !       PBID.rl.r                Pasteboard id of pasteboard.
;  1900          2154  1 !
;  1901          2155  1 !       CLEAR_SCREEN_FLAG.rl.r   Set to 1 to clear the screen,
;  1902          2156  1 !                                0 to keep it as is.
;  1903          2157  1 !                                The default is to clear the screen.
;  1904          2158  1 !
;  1905          2159  1 ! IMPLICIT INPUTS:
;  1906          2160  1 !
;  1907          2161  1 !       NONE
;  1908          2162  1 !
;  1909          2163  1 ! IMPLICIT OUTPUTS:
;  1910          2164  1 !
;  1911          2165  1 !       NONE
;  1912          2166  1 !
;  1913          2167  1 ! COMPLETION STATUS:
;  1914          2168  1 !
;  1915          2169  1 !       SS$_NORMAL       Normal successful completion
;  1916          2170  1 !       SMG$_WRONUMARG   Wrong number of arguments.
;  1917          2171  1 !       SS$_xyz          errors from $DASSGN
;  1918          2172  1 !       LIB$_xyz         errors from LIB$FREE_VM or LIB$FREE_EF
;  1919          2173  1 !       SMG$_xyz         errors from SMG$$FLUSH_BUFFER
;  1920          2174  1 !
;  1921          2175  1 ! SIDE EFFECTS:
;  1922          2176  1 !
;  1923          2177  1 !       NONE
;  1924          2178  1 !--
```

```
;  1926          2179  2 BEGIN
;  1927          2180  2
;  1928          2181  2 BUILTIN
;  1929          2182  2         NULLPARAMETER;
;  1930          2183  2
;  1931          2184  2 LOCAL
;  1932          2185  2
;  1933          2186  2         STATUS,                         ! Status of subroutine calls
;  1934          2187  2
;  1935          2188  2         CURR_PP : REF $PP_DECL,         ! Pasting packet pointer
;  1936          2189  2
;  1937          2190  2         WCB     : REF $WCB_DECL,        ! Window control block.
;  1938          2191  2
;  1939          2192  2         PBCB    : REF $PBCB_DECL;       ! Address of pasteboard control
;  1940          2193  2                                         ! block
;  1941          2194  2
;  1942          2195  2 EXTERNAL ROUTINE
;  1943          2196  2
;  1944          2197  2         SMG$$FORCE_SCROLL_REG,
;  1945          2198  2         SMG$$ERASE_PASTEBOARD,
;  1946          2199  2         SMG$$FLUSH_BUFFER,
;  1947          2200  2         SMG$CHANGE_PBD_CHARACTERISTICS;
;  1948          2201  2
;  1949          2202  2 $SMG$VALIDATE_ARGCOUNT (1, 2);  ! Test for right no. of args
;  1950          2203  2
;  1951          2204  2 $SMG$GET_PBCB (.PBID,PBCB);     ! Get address of PBCB
;  1952          2205  2
;  1953          2206  2 !+
;  1954          2207  2 ! Batch up the unpastes, so that the whole screen disappears at once.
;  1955          2208  2 !-
;  1956          2209  3     IF NOT (STATUS = SMG$$BEGIN_PASTEBOARD_UPDATE_R1(.PBCB))
;  1957          2210  2     THEN
;  1958          2211  2         RETURN (.STATUS);
;  1959          2212  2
;  1960          2213  2 !+
;  1961          2214  2 ! Walk chain of all DCB's pasted to this pasteboard and unpaste each.
;  1962          2215  2 !-
;  1963          2216  2     CURR_PP = .PBCB [PBCB_A_PP_PREV];
;  1964          2217  2     WHILE .CURR_PP NEQ PBCB [PBCB_A_PP_NEXT]
;  1965          2218  2     DO
;  1966          2219  3         BEGIN   ! Walk chain
;  1967          2220  3         LOCAL
;  1968          2221  3             DCB         : REF $DCB_DECL,        ! Address of DCB involved
;  1969          2222  3             PP_BASE     : REF $PP_DECL;         ! Base addr of this PP
;  1970          2223  3
;  1971          2224  3         PP_BASE = .CURR_PP - PP_PBCB_QUEUE_OFFSET;  ! Since queue header
;  1972          2225  3                                                     ! not at top of
;  1973          2226  3                                                     ! structure.
;  1974          2227  3         DCB = .PP_BASE [PP_A_DCB_ADDR];
;  1975          2228  4         IF NOT (STATUS = SMG$$UNPASTE_VIRTUAL_DISPLAY (
;  1976          2229  4                                 .DCB,                 ! DCB involved
;  1977          2230  4                                 .PBCB))               ! PBCB involved
;  1978          2231  3         THEN
;  1979          2232  3             RETURN (.STATUS);
;  1980          2233  3
;  1981          2234  3         CURR_PP = .PP_BASE [PP_A_PREV_PBCB]; ! Step to next PP
;  1982          2235  2         END;    ! Walk chain
```

```
;  1983          2236  2
;  1984          2237  2      PBCB[PBCB_L_BATCH_LEVEL]=0;
;  1985          2238  2
;  1986          2239  2  !+
;  1987          2240  2  ! If the user asked for the screen to be erased, then
;  1988          2241  2  ! release lock on pasteboard, force output of now-blank screen, and
;  1989          2242  2  ! flush it out.
;  1990          2243  2  !-
;  1991          2244  2      IF NULLPARAMETER(CLEAR_SCREEN_FLAG)
;  1992          2245  2      OR (NOT NULLPARAMETER(CLEAR_SCREEN_FLAG) AND ..CLEAR_SCREEN_FLAG)
;  1993          2246  2      THEN
;  1994          2247  3          BEGIN   ! clear screen
;  1995          2248  3
;  1996          2249  3  !(b)      IF NOT (STATUS = SMG$$END_PASTEBOARD_UPDATE_R2(.PBCB))
;  1997          2250  3  !(b)      THEN
;  1998          2251  3  !(b)          RETURN (.STATUS);
;  1999          2252  3
;  2000          2253  3  !(b)      IF NOT ( STATUS = SMG$$CHECK_FOR_OUTPUT_PBCB(.PBCB))
;  2001          2254  3  !(b)      THEN
;  2002          2255  3  !(b)          RETURN (.STATUS);
;  2003          2256  3
;  2004          2257  3  !(b)      IF NOT (STATUS = SMG$$FLUSH_BUFFER(.PBCB))
;  2005          2258  3  !(b)      THEN
;  2006          2259  3  !(b)          RETURN (.STATUS);
;  2007          2260  3
;  2008          2261  3  ! Note (b): Erase pasteboard should clear the screen and
;  2009          2262  3  !           we can bypass flushing since the user is deleting his
;  2010          2263  3  !           pasteboard anyhow.
;  2011          2264  3
;  2012          2265  3          PBCB[PBCB_V_BUF_ENABLED]=0;
;  2013          2266  3
;  2014          2267  4          IF NOT (STATUS = SMG$$ERASE_PASTEBOARD(.PBCB))
;  2015          2268  3          THEN
;  2016          2269  3              RETURN (.STATUS);
;  2017          2270  3
;  2018          2271  3          !+
;  2019          2272  3          ! Set terminal back to it's orignal width.
;  2020          2273  3          ! This requires batching to be off.
;  2021          2274  3          !-
;  2022          2275  3
;  2023          2276  3          IF .PBCB[PBCB_W_WIDTH] NEQ .PBCB[PBCB_W_ORIG_WIDTH]
;  2024          2277  4            THEN  BEGIN
;  2025          2278  4                  STATUS=SMG$CHANGE_PBD_CHARACTERISTICS(.PBID,
;  2026          2279  4                                  %REF(.PBCB[PBCB_W_ORIG_WIDTH]));
;  2027          2280  4                  IF NOT .STATUS THEN RETURN .STATUS;
;  2028          2281  4                  STATUS=SMG$$FLUSH_BUFFER(.PBCB);
;  2029          2282  4                  IF NOT .STATUS THEN RETURN .STATUS
;  2030          2283  3                  END;
;  2031          2284  3
;  2032          2285  3          END     ! clear screen
;  2033          2286  2      ELSE
;  2034          2287  3          BEGIN
;  2035          2288  3          SMG$$FLUSH_BUFFER(.PBCB);
;  2036          2289  3          PBCB[PBCB_V_BUF_ENABLED]=0;
;  2037          2290  2          END;
;  2038          2291  2
;  2039          2292  2  WCB=.PBCB[PBCB_A_WCB];
```

```
;  2040          2293  2
;  2041          2294  2     !+
;  2042          2295  2     ! If a scrolling region is set (other than the full screen),
;  2043          2296  2     ! then reset it now, being careful to leave the cursor alone
;  2044          2297  2     ! even though SET SCROLLING REGION may move it.
;  2045          2298  2     ! Note that if we never established any scrolling regions,
;  2046          2299  2     ! the TOP_SCROLL line will be 0.
;  2047          2300  2     !-
;  2048          2301  2
;  2049          2302  2     IF    .PBCB[PBCB_W_TOP_SCROLL_LINE] NEQ 0
;  2050          2303  2     AND  (.PBCB[PBCB_W_TOP_SCROLL_LINE] NEQ 1   OR
;  2051          2304  2           .PBCB[PBCB_W_BOT_SCROLL_LINE] NEQ .WCB[WCB_W_NO_ROWS])
;  2052          2305  2     THEN
;  2053          2306  3         BEGIN       ! Remove scrolling regions
;  2054          2307  3
;  2055          2308  3         LOCAL
;  2056          2309  3
;  2057          2310  3             FINAL_ROW,      ! Final cursor row
;  2058          2311  3             FINAL_COL;      ! Final cursor column
;  2059          2312  3
;  2060          2313  3         !+
;  2061          2314  3         ! Construct escape sequence (possibly null if not a supporting terminal)
;  2062          2315  3         ! to set the hardware scroll region to the full height of the screen.
;  2063          2316  3         !-
;  2064          2317  3
;  2065        P 2318  3         $SMG$GET_TERM_DATA(SET_SCROLL_REGION,
;  2066        P 2319  3                            1,
;  2067          2320  3                            .WCB [WCB_W_NO_ROWS]);
;  2068          2321  3
;  2069          2322  3         !+
;  2070          2323  3         ! Output BUFFER.
;  2071          2324  3         !-
;  2072          2325  3
;  2073          2326  3         IF .PBCB[PBCB_L_CAP_LENGTH] NEQ 0
;  2074          2327  3         THEN
;  2075          2328  4             BEGIN   ! Issue the reset
;  2076          2329  4
;  2077          2330  4             !+
;  2078          2331  4             ! Remember where the user left the physical cursor, since
;  2079          2332  4             ! changing scrolling regions might upset this.
;  2080          2333  4             !-
;  2081          2334  4
;  2082          2335  4             FINAL_ROW=.WCB[WCB_W_CURR_CUR_ROW];
;  2083          2336  4             FINAL_COL=.WCB[WCB_W_CURR_CUR_COL];
;  2084          2337  4
;  2085          2338  4             STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  2086          2339  4                                  .PBCB[PBCB_A_CAP_BUFFER]);
;  2087          2340  4             IF NOT .STATUS THEN RETURN .STATUS;
;  2088          2341  4
;  2089          2342  4             !+
;  2090          2343  4             ! Move the cursor back to where it was.
;  2091          2344  4             !-
;  2092          2345  4
;  2093          2346  4             IF NOT NULLPARAMETER(CLEAR_SCREEN_FLAG)
;  2094          2347  4             AND NOT ..CLEAR_SCREEN_FLAG
;  2095          2348  5               THEN  BEGIN   ! Restore final cursor position
;  2096          2349  5
```

```
;  2097     2350  5                  $SMG$GET_TERM_DATA(SET_CURSOR_ABS,.FINAL_ROW,.FINAL_COL);
;  2098     2351  5
;  2099     2352  5                  STATUS = SMG$$OUTPUT(.PBCB,.PBCB[PBCB_L_CAP_LENGTH],
;  2100     2353  5                                       .PBCB[PBCB_A_CAP_BUFFER]);
;  2101     2354  5                  IF NOT .STATUS THEN RETURN .STATUS
;  2102     2355  5
;  2103     2356  5                  END     ! Restore final cursor position
;  2104     2357  5
;  2105     2358  4              END     ! Issue the reset
;  2106     2359  4
;  2107     2360  3          END;        ! Remove scrolling regions
;  2108     2361  3
;  2109     2362  3      !+
;  2110     2363  3      ! Get rid of our exit handler.  Ignore a no handler found error.
;  2111     2364  3      !-
;  2112     2365  3
;  2113     2366  3      STATUS=$CANEXH(DESBLK=PBCB[PBCB_R_EXIT_BLOCK]);
;  2114     2367  3      IF (NOT .STATUS) AND (.STATUS NEQ SS$_NOHANDLER)
;  2115     2368  3        THEN  RETURN  .STATUS;
;  2116     2369  3
;  2117     2370  3      !+
;  2118     2371  3      ! Deallocate the WCB if there is one.
;  2119     2372  3      !-
;  2120     2373  3
;  2121     2374  3      IF .PBCB[PBCB_A_WCB] NEQ 0
;  2122     2375  3        THEN  BEGIN   ! getting rid of WCB
;  2123     2376  3
;  2124     2377  3              STATUS=SMG$$DEALLOCATE_WCB(.PBCB[PBCB_A_WCB]);
;  2125     2378  3              PBCB[PBCB_A_WCB]=0;                          ! safety
;  2126     2379  3              IF NOT .STATUS THEN RETURN .STATUS
;  2127     2380  3
;  2128     2381  3              END;    ! getting rid of WCB
;  2129     2382  3
;  2130     2383  3      !+
;  2131     2384  3      ! If there is a channel assigned, deassign it now.
;  2132     2385  3      ! This automatically cancels any I/O on the channel.
;  2133     2386  3      ! In particular, it removes any out-of-band ASTs that
;  2134     2387  3      ! were enabled.
;  2135     2388  3      !-
;  2136     2389  3
;  2137     2390  3      IF .PBCB[PBCB_W_CHAN] NEQ 0
;  2138     2391  3        THEN  BEGIN   ! deassigning channel
;  2139     2392  3
;  2140     2393  3              STATUS=$DASSGN(CHAN=.PBCB[PBCB_W_CHAN]);
;  2141     2394  3              PBCB[PBCB_W_CHAN]=0;                         ! just in case we get called
;  2142     2395  3                                                           ! again after returning an error
;  2143     2396  3              IF NOT .STATUS THEN RETURN .STATUS
;  2144     2397  3
;  2145     2398  3              END;    ! deassigning channel
;  2146     2399  3
;  2147     2400  3      !+
;  2148     2401  3      ! Free the event flags now.
;  2149     2402  3      ! Ignore error if it was already free.
;  2150     2403  3      !-
;  2151     2404  3
;  2152     2405  3      IF .PBCB[PBCB_B_EFN] NEQ 0
;  2153     2406  3        THEN  BEGIN
```

```
;  2154     2407  3          STATUS=LIB$FREE_EF( %REF(.PBCB[PBCB_B_EFN]) );
;  2155     2408  4          IF (NOT .STATUS) AND (.STATUS NEQ LIB$_EF_ALRFRE)
;  2156     2409                THEN  RETURN .STATUS;
;  2157     2410          PBCB[PBCB_B_EFN]=0
;  2158     2411          END;
;  2159     2412
;  2160     2413     IF .PBCB[PBCB_B_ASYNC_EFN] NEQ 0
;  2161     2414       THEN  BEGIN
;  2162     2415          STATUS=LIB$FREE_EF( %REF(.PBCB[PBCB_B_ASYNC_EFN]) );
;  2163     2416          IF (NOT .STATUS) AND (.STATUS NEQ LIB$_EF_ALRFRE)
;  2164     2417            THEN  RETURN .STATUS;
;  2165     2418          PBCB[PBCB_B_ASYNC_EFN]=0
;  2166     2419          END;
;  2167     2420
;  2168     2421     !+
;  2169     2422     ! Free the output buffer now.
;  2170     2423     !-
;  2171     2424
;  2172     2425     IF .PBCB[PBCB_A_OUTPUT_BUFFER] NEQ 0
;  2173     2426       THEN  BEGIN    ! freeing output buffer
;  2174     2427
;  2175     2428          STATUS=LIB$FREE_VM(%REF (.PBCB[PBCB_W_OUTPUT_BUFSIZ] ),
;  2176     2429                             PBCB[PBCB_A_OUTPUT_BUFFER] );
;  2177     2430          PBCB[PBCB_A_OUTPUT_BUFFER]=0;
;  2178     2431          IF NOT .STATUS THEN RETURN .STATUS
;  2179     2432
;  2180     2433          END;     ! freeing output buffer
;  2181     2434
;  2182     2435     !+
;  2183     2436     ! Free the output filename.
;  2184     2437     !-
;  2185     2438
;  2186     2439     IF .PBCB[PBCB_W_OUTNAM_LEN] NEQ 0
;  2187     2440       THEN  BEGIN    ! freeing outname
;  2188     2441          STATUS=LIB$FREE_VM(%REF (.PBCB[PBCB_W_OUTNAM_LEN] ),
;  2189     2442                             PBCB[PBCB_A_OUTNAM] );
;  2190     2443          PBCB[PBCB_W_OUTNAM_LEN]=0;
;  2191     2444          IF NOT .STATUS THEN RETURN .STATUS
;  2192     2445          END;     ! freeing outname
;  2193     2446
;  2194     2447     !+
;  2195     2448     ! Close the output file, if there was one.
;  2196     2449     !-
;  2197     2450
;  2198     2451     IF .PBCB[PBCB_A_FAB] NEQ 0
;  2199     2452       THEN  BEGIN    ! Close output file
;  2200     2453          STATUS=$CLOSE(  FAB = .PBCB[PBCB_A_FAB]);
;  2201     2454          IF NOT .STATUS THEN RETURN .STATUS
;  2202     2455          END;     ! Close output file
;  2203     2456
;  2204     2457     !+
;  2205     2458     ! Free the record buffer, if there was one.
;  2206     2459     !-
;  2207     2460
;  2208     2461     IF .PBCB[PBCB_A_RBF] NEQ 0
;  2209     2462       THEN  BEGIN
;  2210     2463  3          STATUS=LIB$GET_VM(%REF(.PBCB[PBCB_W_WIDTH]+1),PBCB[PBCB_A_RBF]);
```

```
;  2211          2464  3          PBCB[PBCB_A_RBF]=0;
;  2212          2465  3          IF NOT .STATUS THEN RETURN .STATUS;
;  2213          2466  3          END;
;  2214          2467  2
;  2215          2468  2      !+
;  2216          2469  2      ! Free any FAB or RAB that was created.
;  2217          2470  2      !-
;  2218          2471  2
;  2219          2472  2      IF .PBCB[PBCB_A_FAB] NEQ 0
;  2220          2473  3        THEN  BEGIN    ! freeing FAB
;  2221          2474  3              STATUS=LIB$FREE_VM(%REF (FAB$C_BLN),
;  2222          2475  3                                 PBCB[PBCB_A_FAB] );
;  2223          2476  3              PBCB[PBCB_A_FAB]=0;
;  2224          2477  3              IF NOT .STATUS THEN RETURN .STATUS
;  2225          2478  2              END;     ! freeing FAB
;  2226          2479  2
;  2227          2480  2      IF .PBCB[PBCB_A_RAB] NEQ 0
;  2228          2481  3        THEN  BEGIN    ! freeing RAB
;  2229          2482  3              STATUS=LIB$FREE_VM(%REF (RAB$C_BLN),
;  2230          2483  3                                 PBCB[PBCB_A_RAB] );
;  2231          2484  3              PBCB[PBCB_A_RAB]=0;
;  2232          2485  3              IF NOT .STATUS THEN RETURN .STATUS
;  2233          2486  2              END;     ! freeing RAB
;  2234          2487  2
;  2235          2488  2      !+
;  2236          2489  2      ! Now go free the PBCB itself.
;  2237          2490  2      !-
;  2238          2491  2
;  2239          2492  2      IF NOT (STATUS=LIB$FREE_VM (%REF (PBCB_K_SIZE), PBCB))
;  2240          2493  2      THEN
;  2241          2494  2          RETURN (.STATUS);
;  2242          2495  2
;  2243          2496  2      !+
;  2244          2497  2      ! Since all went well, we can now adjust the count of how many PBCB's
;  2245          2498  2      ! we have and remove its address from the pasteboard directory.
;  2246          2499  2      !-
;  2247          2500  2
;  2248          2501  2      PBD_V_PB_AVAIL [..PBID] = 0;
;  2249          2502  2
;  2250          2503  2      PBD_L_COUNT = .PBD_L_COUNT - 1;
;  2251          2504  2
;  2252          2505  2      PBD_A_PBCB [..PBID] = 0;
;  2253          2506  2
;  2254          2507  2      RETURN  SS$_NORMAL
;  2255          2508  2
;  2256          2509  1  END;                        ! Routine SMG$DELETE_PASTEBOARD


                                                     .EXTRN  SYS$CANEXH, SYS$DASSGN
                                                     .EXTRN  SYS$CLOSE

                                        0FFC 00000   .ENTRY  SMG$DELETE_PASTEBOARD, Save R2,R3,R4,R5,R6,-; 2136
                                                             R7,R8,R9,R10,R11                               ;
5B 00000000G   00   9E 00002                         MOVAB   LIB$FREE_VM, R11                               ;
5A 00000000'   EF   9E 00009                         MOVAB   PBD_L_COUNT, R10                               ;
5E             14   C2 00010                         SUBL2   #20, SP                                        ;
```

```
                50              6C              01  83 00013          SUBB3   #1, (AP), DIFF                          ; 2202
                01                              50  91 00017          CMPB    DIFF, #1                                ;
                                                08  1B 0001A          BLEQU   1$                                      ;
                                50 00000000G    8F  D0 0001C          MOVL    #SMG$_WRONUMARG, R0                     ;
                                                04  00023          RET                                             ;
                                59          04  AC  D0 00024 1$:     MOVL    PBID, R9                                ; 2204
                                50              69  D0 00028          MOVL    (R9), R0                                ;
                                                0A  19 0002B          BLSS    2$                                      ;
                                6A                  50  D1 0002D      CMPL    R0, PBD_L_COUNT                         ;
                                                    05  14 00030      BGTR    2$                                      ;
                        08      44  AA              50  E0 00032      BBS     R0, PBD_V_PB_AVAIL, 3$                  ;
                                50 00000000G    8F  D0 00037 2$:     MOVL    #SMG$_INVPAS_ID, R0                     ;
                                                04  0003E          RET                                             ;
                                04  AE          04 AA40  D0 0003F 3$:     MOVL    PBD_A_PBCB[R0], PBCB                    ;
                                54      04      AE  D0 00045          MOVL    PBCB, R4                                ; 2209
                                50              54  D0 00049          MOVL    R4, R0                                  ;
                                00000000G       00  16 0004C          JSB     SMG$$BEGIN_PASTEBOARD_UPDATE_R1         ;
                                55              50  D0 00052          MOVL    R0, STATUS                              ;
                                6E              55  E9 00055          BLBC    STATUS, 7$                              ;
                                53      04      A4  D0 00058          MOVL    4(R4), CURR_PP                          ; 2216
                                54              53  D1 0005C 4$:      CMPL    CURR_PP, R4                             ; 2217
                                                1B  13 0005F          BEQL    5$                                      ;
                                52      F8      A3  9E 00061          MOVAB   -8(R3), PP_BASE                         ; 2224
                                50      10      A2  D0 00065          MOVL    16(PP_BASE), DCB                        ; 2227
                                                11  BB 00069          PUSHR   #^M<R0,R4>                              ; 2229
                        0000V   CF                  02  FB 0006B      CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY        ;
                                55                  50  D0 00070      MOVL    R0, STATUS                              ;
                                62                  55  E9 00073      BLBC    STATUS, 8$                              ; 2228
                                53      0C          A2  D0 00076      MOVL    12(PP_BASE), CURR_PP                    ; 2234
                                                    E0  11 0007A      BRB     4$                                      ; 2217
                                    00A4        C4  D4 0007C 5$:      CLRL    164(R4)                                 ; 2237
                                02                  6C  91 00080      CMPB    (AP), #2                                ; 2244
                                                    13  1F 00083      BLSSU   6$                                      ;
                                        08      AC  D5 00085          TSTL    8(AP)                                   ;
                                                0E  13 00088          BEQL    6$                                      ;
                                02                  6C  91 0008A      CMPB    (AP), #2                                ; 2245
                                                    4C  1F 0008D      BLSSU   9$                                      ;
                                        08      AC  D5 0008F          TSTL    8(AP)                                   ;
                                                47  13 00092          BEQL    9$                                      ;
                                43      08      BC  E9 00094          BLBC    @CLEAR_SCREEN_FLAG, 9$                  ;
                            0C  A4              01  8A 00098 6$:      BICB2   #1, 12(R4)                              ; 2265
                                                54  DD 0009C          PUSHL   R4                                      ; 2267
                00000000G   00                  01  FB 0009E          CALLS   #1, SMG$$ERASE_PASTEBOARD               ;
                            55                  50  D0 000A5          MOVL    R0, STATUS                              ;
                            2D                  55  E9 000A8          BLBC    STATUS, 8$                              ;
                        00E6    C4      5A      A4  B1 000AB          CMPW    90(R4), 230(R4)                         ; 2276
                                                35  13 000B1          BEQL    10$                                     ;
                                6E      00E6    C4  3C 000B3          MOVZWL  230(R4), (SP)                           ; 2279
                                        4200    8F  BB 000B8          PUSHR   #^M<R9,SP>                              ; 2278
                00000000G   00                  02  FB 000BC          CALLS   #2, SMG$CHANGE_PBD_CHARACTERISTICS      ;
                            55                  50  D0 000C3          MOVL    R0, STATUS                              ;
                            0F                  55  E9 000C6 7$:      BLBC    STATUS, 8$                              ; 2280
                                                54  DD 000C9          PUSHL   R4                                      ; 2281
                00000000G   00                  01  FB 000CB          CALLS   #1, SMG$$FLUSH_BUFFER                   ;
                            55                  50  D0 000D2          MOVL    R0, STATUS                              ;
                            10                  55  E8 000D5          BLBS    STATUS, 10$                             ; 2282
                                            023D    31 000D8 8$:      BRW     38$                                     ;
```

```
                                                54  DD 000DB 9$:    PUSHL   R4                                      ; 2288
                00000000G  00                   01  FB 000DD        CALLS   #1, SMG$$FLUSH_BUFFER
                      0C   A4                   01  8A 000E4        BICB2   #1, 12(R4)                             ; 2289
                           53          08       A4  D0 000E8 10$:    MOVL    8(R4), WCB                             ; 2292
                           50        00F4       C4  3C 000EC        MOVZWL  244(R4), R0                            ; 2302
                                                7A  13 000F1        BEQL    14$
                           01                   50  B1 000F3        CMPW    R0, #1                                 ; 2303
                                                08  12 000F6        BNEQ    11$
                           02   A3      00F6     C4  B1 000F8        CMPW    246(R4), 2(WCB)                        ; 2304
                                                6D  13 000FE        BEQL    14$
                           52        0108       C4  9E 00100 11$:    MOVAB   264(R4), R2                            ; 2320
                           56        00FC       C4  9E 00105        MOVAB   252(R4), R6
                                                66  D5 0010A        TSTL    (R6)
                                                04  12 0010C        BNEQ    12$
                                                62  D4 0010E        CLRL    (R2)
                                                2F  11 00110        BRB     13$
                      08   AE                   02  D0 00112 12$:    MOVL    #2, INPUT_ARGS
                      0C   AE                   01  D0 00116        MOVL    #1, INPUT_ARGS+4
                      10   AE          02       A3  3C 0011A        MOVZWL  2(WCB), INPUT_ARGS+8
                                08      AE  9F 0011F        PUSHAB  INPUT_ARGS
                                      0104  C4  DD 00122        PUSHL   260(R4)
                                            52  DD 00126        PUSHL   R2
                                      0100  C4  9F 00128        PUSHAB  256(R4)
                      10   AE       023C  8F  3C 0012C        MOVZWL  #572, 16(SP)
                                        10  AE  9F 00132        PUSHAB  16(SP)
                                            56  DD 00135        PUSHL   R6
                00000000G  00                   06  FB 00137        CALLS   #6, SMG$GET_TERM_DATA
                           63                   50  E9 0013E        BLBC    STATUS, 16$
                                                62  D5 00141 13$:    TSTL    (R2)                                   ; 2326
                                                76  13 00143        BEQL    19$
                           58          20       A3  32 00145        CVTWL   32(WCB), FINAL_ROW                     ; 2335
                           57          22       A3  32 00149        CVTWL   34(WCB), FINAL_COL                     ; 2336
                           53        0104       C4  9E 0014D        MOVAB   260(R4), R3                            ; 2339
                                                63  DD 00152        PUSHL   (R3)
                                                62  DD 00154        PUSHL   (R2)                                   ; 2338
                                                54  DD 00156        PUSHL   R4
                00000000G  00                   03  FB 00158        CALLS   #3, SMG$$OUTPUT
                           55                   50  D0 0015F        MOVL    R0, STATUS
                           53                   55  E9 00162        BLBC    STATUS, 18$                            ; 2340
                           02                   6C  91 00165        CMPB    (AP), #2                               ; 2346
                                                51  1F 00168        BLSSU   19$
                                          08    AC  D5 0016A        TSTL    8(AP)
                                                4C  13 0016D 14$:    BEQL    19$
                           48           08      BC  E8 0016F        BLBS    @CLEAR_SCREEN_FLAG, 19$                ; 2347
                                                66  D5 00173        TSTL    (R6)                                   ; 2350
                                                04  12 00175        BNEQ    15$
                                                62  D4 00177        CLRL    (R2)
                                                2D  11 00179        BRB     17$
                      08   AE                   02  D0 0017B 15$:    MOVL    #2, INPUT_ARGS
                      0C   AE                   58  D0 0017F        MOVL    FINAL_ROW, INPUT_ARGS+4
                      10   AE                   57  D0 00183        MOVL    FINAL_COL, INPUT_ARGS+8
                                08      AE  9F 00187        PUSHAB  INPUT_ARGS
                                            63  DD 0018A        PUSHL   (R3)
                                            52  DD 0018C        PUSHL   R2
                                      0100  C4  9F 0018E        PUSHAB  256(R4)
                      10   AE       023A  8F  3C 00192        MOVZWL  #570, 16(SP)
                                        10  AE  9F 00198        PUSHAB  16(SP)
```

```
                                 56  DD 0019B         PUSHL   R6                                   :
00000000G  00                    06  FB 0019D         CALLS   #6, SMG$GET_TERM_DATA                :
           01                    50  E8 001A4  16$:   BLBS    STATUS, 17$                          :
                                     04 001A7         RET                                          :
                                 63  DD 001A8  17$:   PUSHL   (R3)                                 : 2353
                                 62  DD 001AA         PUSHL   (R2)                                 : 2352
                                 54  DD 001AC         PUSHL   R4                                   :
00000000G  00                    03  FB 001AE         CALLS   #3, SMG$$OUTPUT                      :
           55                    50  D0 001B5         MOVL    R0, STATUS                           :
           48                    55  E9 001B8  18$:   BLBC    STATUS, 22$                          : 2354
                         74      A4  9F 001BB  19$:   PUSHAB  116(R4)                              : 2366
00000000G  00                    01  FB 001BE         CALLS   #1, SYS$CANEXH                       :
           55                    50  D0 001C5         MOVL    R0, STATUS                           :
           09                    55  E8 001C8         BLBS    STATUS, 20$                          :
000008F8   8F                    55  D1 001CB         CMPL    STATUS, #2296                        : 2367
                                 78  12 001D2         BNEQ    26$                                  :
                         08      A4  D5 001D4  20$:   TSTL    8(R4)                                : 2374
                                 14  13 001D7         BEQL    21$                                  :
                         08      A4  DD 001D9         PUSHL   8(R4)                                : 2377
    0000V  CF                    01  FB 001DC         CALLS   #1, SMG$$DEALLOCATE_WCB              :
           55                    50  D0 001E1         MOVL    R0, STATUS                           :
                         08      A4  D4 001E4         CLRL    8(R4)                                : 2378
           03                    55  E8 001E7         BLBS    STATUS, 21$                          : 2379
                               0080  31 001EA         BRW     30$                                  :
                         64      A4  B5 001ED  21$:   TSTW    100(R4)                              : 2390
                                 17  13 001F0         BEQL    23$                                  :
           7E            64      A4  3C 001F2         MOVZWL  100(R4), -(SP)                       : 2393
00000000G  00                    01  FB 001F6         CALLS   #1, SYS$DASSGN                       :
           55                    50  D0 001FD         MOVL    R0, STATUS                           :
                         64      A4  B4 00200         CLRW    100(R4)                              : 2394
           03                    55  E8 00203  22$:   BLBS    STATUS, 23$                          : 2396
                               0083  31 00206         BRW     32$                                  :
                         66      A4  95 00209  23$:   TSTB    102(R4)                              : 2405
                                 1F  13 0020C         BEQL    25$                                  :
           6E            66      A4  9A 0020E         MOVZBL  102(R4), (SP)                        : 2407
                                 5E  DD 00212         PUSHL   SP                                   :
00000000G  00                    01  FB 00214         CALLS   #1, LIB$FREE_EF                      :
           55                    50  D0 0021B         MOVL    R0, STATUS                           :
           09                    55  E8 0021E         BLBS    STATUS, 24$                          : 2408
00000000G  8F                    55  D1 00221         CMPL    STATUS, #LIB$_EF_ALRFRE              :
                                 22  12 00228         BNEQ    26$                                  :
                         66      A4  94 0022A  24$:   CLRB    102(R4)                              : 2410
                         67      A4  95 0022D  25$:   TSTB    103(R4)                              : 2413
                                 22  13 00230         BEQL    29$                                  :
           6E            67      A4  9A 00232         MOVZBL  103(R4), (SP)                        : 2415
                                 5E  DD 00236         PUSHL   SP                                   :
00000000G  00                    01  FB 00238         CALLS   #1, LIB$FREE_EF                      :
           55                    50  D0 0023F         MOVL    R0, STATUS                           :
           0C                    55  E8 00242         BLBS    STATUS, 28$                          : 2416
00000000G  8F                    55  D1 00245         CMPL    STATUS, #LIB$_EF_ALRFRE              :
                                 03  13 0024C  26$:   BEQL    28$                                  :
                               00C7  31 0024E  27$:   BRW     38$                                  :
                         67      A4  94 00251  28$:   CLRB    103(R4)                              : 2418
                         6C      A4  D5 00254  29$:   TSTL    108(R4)                              : 2425
                                 17  13 00257         BEQL    31$                                  :
                         6C      A4  9F 00259         PUSHAB  108(R4)                              : 2429
       04  AE            70      A4  3C 0025C         MOVZWL  112(R4), 4(SP)                       : 2428
```

```
                                 04   AE  9F 00261          PUSHAB  4(SP)
                              6B      02  FB 00264          CALLS   #2, LIB$FREE_VM                 ;  2429
                              55      50  D0 00267          MOVL    R0, STATUS                      ;
                                 6C   A4  D4 0026A          CLRL    108(R4)                         ;  2430
                              DE      55  E9 0026D 30$:     BLBC    STATUS, 27$                     ;  2431
                              50 00E4 C4  3C 00270 31$:     MOVZWL  228(R4), R0                     ;  2439
                                      18  13 00275          BEQL    33$                             ;
                                 00B0 C4  9F 00277          PUSHAB  176(R4)                         ;  2442
                           04    AE      50  D0 0027B       MOVL    R0, 4(SP)                       ;  2441
                                 04   AE  9F 0027F          PUSHAB  4(SP)                           ;
                              6B      02  FB 00282          CALLS   #2, LIB$FREE_VM                 ;  2442
                              55      50  D0 00285          MOVL    R0, STATUS                      ;
                                 00E4 C4  B4 00288          CLRW    228(R4)                         ;  2443
                              BF      55  E9 0028C 32$:     BLBC    STATUS, 27$                     ;  2444
                              53 00E8 C4  9E 0028F 33$:     MOVAB   232(R4), R3                     ;  2451
                                      63  D5 00294          TSTL    (R3)                            ;
                                      0F  13 00296          BEQL    34$                             ;
                                      63  DD 00298          PUSHL   (R3)                            ;  2453
                  00000000G   00      01  FB 0029A          CALLS   #1, SYS$CLOSE                   ;
                              55      50  D0 002A1          MOVL    R0, STATUS                      ;
                              71      55  E9 002A4          BLBC    STATUS, 38$                     ;  2454
                              52 00F0 C4  9E 002A7 34$:     MOVAB   240(R4), R2                     ;  2461
                                      62  D5 002AC          TSTL    (R2)                            ;
                                      1C  13 002AE          BEQL    35$                             ;
                                      52  DD 002B0          PUSHL   R2                              ;  2463
                           04    AE   5A  A4  3C 002B2      MOVZWL  90(R4), 4(SP)                   ;
                                 04   AE  D6 002B7          INCL    4(SP)                           ;
                                 04   AE  9F 002BA          PUSHAB  4(SP)                           ;
                  00000000G   00      02  FB 002BD          CALLS   #2, LIB$GET_VM                  ;
                              55      50  D0 002C4          MOVL    R0, STATUS                      ;
                                      62  D4 002C7          CLRL    (R2)                            ;  2464
                              4C      55  E9 002C9          BLBC    STATUS, 38$                     ;  2465
                                      63  D5 002CC 35$:     TSTL    (R3)                            ;  2472
                                      15  13 002CE          BEQL    36$                             ;
                                      53  DD 002D0          PUSHL   R3                              ;  2475
                           04    AE   50  8F  9A 002D2      MOVZBL  #80, 4(SP)                      ;  2474
                                 04   AE  9F 002D7          PUSHAB  4(SP)                           ;
                              6B      02  FB 002DA          CALLS   #2, LIB$FREE_VM                 ;  2475
                              55      50  D0 002DD          MOVL    R0, STATUS                      ;
                                      63  D4 002E0          CLRL    (R3)                            ;  2476
                              33      55  E9 002E2          BLBC    STATUS, 38$                     ;  2477
                              52 00EC C4  9E 002E5 36$:     MOVAB   236(R4), R2                     ;  2480
                                      62  D5 002EA          TSTL    (R2)                            ;
                                      15  13 002EC          BEQL    37$                             ;
                                      52  DD 002EE          PUSHL   R2                              ;  248?
                           04    AE   44  8F  9A 002F0      MOVZBL  #68, 4(SP)                      ;  2482
                                 04   AE  9F 002F5          PUSHAB  4(SP)                           ;
                              6B      02  FB 002F8          CALLS   #2, LIB$FREE_VM                 ;  2483
                              55      50  D0 002FB          MOVL    R0, STATUS                      ;
                                      62  D4 002FE          CLRL    (R2)                            ;  2484
                              15      55  E9 00300          BLBC    STATUS, 38$                     ;  2485
                                 04   AE  9F 00303 37$:     PUSHAB  PBCB                            ;  2492
                           04    AE 014C  8F  3C 00306      MOVZWL  #332, 4(SP)                     ;
                                 04   AE  9F 0030C          PUSHAB  4(SP)                           ;
                              6B      02  FB 0030F          CALLS   #2, LIB$FREE_VM                 ;
                              55      50  D0 00312          MOVL    R0, STATUS                      ;
                              04      55  E8 00315          BLBS    STATUS, 39$                     ;
```

```
                        50          55  D0 00318 38$:   MOVL    STATUS, R0                      ; 2494
                                        04 0031B        RET                                     ;
                        50          69  D0 0031C 39$:   MOVL    (R9), R0                        ; 2501
                  00        44    AA    50  E5 0031F    BBCC    R0, PBD_V_PB_AVAIL, 40$         ;
                                        6A  D7 00324 40$:   DECL    PBD_L_COUNT                 ; 2503
                                  04 AA40   D4 00326        CLRL    PBD_A_PBCB[R0]              ; 2505
                        50          01    D0 0032A        MOVL    #1, R0                      ; 2507
                                        04 0032D        RET                                     ; 2509

; Routine Size:  814 bytes,    Routine Base:  _SMG$CODE + 0A95

; 2257          2510  1 !<BLF/PAGE>
```

```
;  2259          2511  1 %SBTTL 'SMG$CREATE_VIRTUAL_DISPLAY - Create Virtual Display'
;  2260          2512  1 GLOBAL ROUTINE SMG$CREATE_VIRTUAL_DISPLAY (
;  2261          2513  1                                           NUM_ROWS,      ! height
;  2262          2514  1                                           NUM_COLS,      ! width
;  2263          2515  1                                           NEW_DISPLAY_ID,
;  2264          2516  1                                           DISPLAY_ATTRIBUTES,
;  2265          2517  1                                           VIDEO_ATTRIBUTES,
;  2266          2518  1                                           CHAR_SET
;  2267          2519  1                                          ) =
;  2268          2520  1 !++
;  2269          2521  1 ! FUNCTIONAL DESCRIPTION:
;  2270          2522  1 !
;  2271          2523  1 !       This routine creates a new virtual display -- returning its
;  2272          2524  1 !       assigned display_id.  Its initial contents are blanks with
;  2273          2525  1 !       video attributes set to those specified or zero.  The cursor
;  2274          2526  1 !       will be at row 1 column 1.
;  2275          2527  1 !
;  2276          2528  1 ! CALLING SEQUENCE:
;  2277          2529  1 !
;  2278          2530  1 !       ret_status.wlc.v = SMG$CREATE_VIRTUAL_DISPLAY (
;  2279          2531  1 !                                   NUM_ROWS.rl.r,          ! Height
;  2280          2532  1 !                                   NUM_COLS.rl.r,          ! Width
;  2281          2533  1 !                                   NEW_DISPLAY_ID.wl.r
;  2282          2534  1 !                                   [,DISPLAY_ATTRIBUTES.rl.r]
;  2283          2535  1 !                                   [,VIDEO_ATTRIBUTES.rl.r]
;  2284          2536  1 !                                   [,CHAR_SET.rl.r])
;  2285          2537  1 !
;  2286          2538  1 ! FORMAL PARAMETERS:
;  2287          2539  1 !
;  2288          2540  1 !       NUM_ROWS.rl.r    Number of rows in new virtual display.
;  2289          2541  1 !
;  2290          2542  1 !       NUM_COLS.rl.r    Number of columns in new virtual display.
;  2291          2543  1 !
;  2292          2544  1 !       NEW_DISPLAY_ID.wl.r     Virtual display id of newly-created
;  2293          2545  1 !                               virtual display.
;  2294          2546  1 !
;  2295          2547  1 !       DISPLAY_ATTRIBUTES.rl.r The default display attributes.
;  2296          2548  1 !
;  2297          2549  1 !                       SMG$M_BORDER if virtual display is to be
;  2298          2550  1 !                                    displayed with a border.
;  2299          2551  1 !
;  2300          2552  1 !                       SMG$M_TRUNC_ICON if an icon should be displayed
;  2301          2553  1 !                                    when text overflows the display bounds.
;  2302          2554  1 !
;  2303          2555  1 !                       SMG$M_DISPLAY_CONTROLS if carriage controls (CR, LF,
;  2304          2556  1 !                                    FF, VT, HT) should be displayed instead
;  2305          2557  1 !                                    of executed.
;  2306          2558  1 !                       If omitted, none of the attributes will be set.
;  2307          2559  1 !
;  2308          2560  1 !       VIDEO_ATTRIBUTES.rl.r   The default rendition code to be
;  2309          2561  1 !                       applied to all output to this display unless
;  2310          2562  1 !                       overridden on a particular output call.
;  2311          2563  1 !                       If not supplied, default will be all zero (no
;  2312          2564  1 !                       attributes).
;  2313          2565  1 !
;  2314          2566  1 !                       Values:
;  2315          2567  1 !
```

```
;  2316     2568  1 !                          SMG$M_BLINK     displays characters blinking.
;  2317     2569  1 !
;  2318     2570  1 !                          SMG$M_BOLD      displays characters in
;  2319     2571  1 !                                          higher-than-normal intensity.
;  2320     2572  1 !
;  2321     2573  1 !                          SMG$M_REVERSE   displays characters in reverse
;  2322     2574  1 !                                          video -- that is, using the
;  2323     2575  1 !                                          opposite default rendition of
;  2324     2576  1 !                                          the virtual display.
;  2325     2577  1 !
;  2326     2578  1 !                          SMG$M_UNDERLINE displays characters underlined.
;  2327     2579  1 !
;  2328     2580  1 !          CHAR_SET.rb.r   [Optional].  If provided, specifies the default
;  2329     2581  1 !                          character set to be used for this display.
;  2330     2582  1 !                          Recognized values are:
;  2331     2583  1 !                                          SMG$C_UNITED_KINGDOM
;  2332     2584  1 !                                          SMG$C_ASCII (default)
;  2333     2585  1 !                                          SMG$C_SPEC_GRAPHICS
;  2334     2586  1 !                                          SMG$C_ALT_CHAR
;  2335     2587  1 !                                          SMG$C_ALT_GRAPHICS
;  2336     2588  1 !
;  2337     2589  1 ! IMPLICIT INPUTS:
;  2338     2590  1 !
;  2339     2591  1 !       NONE
;  2340     2592  1 !
;  2341     2593  1 ! IMPLICIT OUTPUTS:
;  2342     2594  1 !
;  2343     2595  1 !       NONE
;  2344     2596  1 !
;  2345     2597  1 ! COMPLETION STATUS:
;  2346     2598  1 !
;  2347     2599  1 !       SS$_NORMAL      Normal successful completion
;  2348     2600  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
;  2349     2601  1 !                       buffer.
;  2350     2602  1 !       SMG$_INVARG     Unrecognized Video Attributes
;  2351     2603  1 !                    or Unrecognized Display Attributes
;  2352     2604  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  2353     2605  1 !
;  2354     2606  1 ! SIDE EFFECTS:
;  2355     2607  1 !
;  2356     2608  1 !       NONE
;  2357     2609  1 !--
;  2358     2610  2     BEGIN
;  2359     2611  2     BUILTIN
;  2360     2612  2         NULLPARAMETER;
;  2361     2613  2
;  2362     2614  2     $SMG$VALIDATE_ARGCOUNT (3, 6);      ! Test for right no. of args
;  2363     2615  2
;  2364     2616  2     RETURN (SMG$$CREATE_VIRTUAL_DISPLAY(
;  2365     2617  3                 .NUM_ROWS,
;  2366     2618  3                 .NUM_COLS,
;  2367     2619  3                 .NEW_DISPLAY_ID,          ! Gets the DCB address for the display created
;  2368     2620  4                 (IF NOT NULLPARAMETER(DISPLAY_ATTRIBUTES)
;  2369     2621  4                   THEN  .DISPLAY_ATTRIBUTES
;  2370     2622  3                   ELSE  UPLIT(0)),
;  2371     2623  4                 (IF NOT NULLPARAMETER(VIDEO_ATTRIBUTES)
;  2372     2624  4                   THEN  .VIDEO_ATTRIBUTES
```

```
;  2373          2625  3                    ELSE  UPLIT(0) )
;  2374          2626  4                  (IF NOT NULLPARAMETER(CHAR_SET)
;  2375          2627  4                    THEN  .CHAR_SET
;  2376          2628  2                    ELSE  UPLIT(0) ) ) );
;  2377          2629  2
;  2378          2630  1     END;                          ! Routine SMG$CREATE_VIRTUAL_DISPLAY


                                                00DC3                .BLKB   1
                                     00000000   00DC4 P.AAC:         .LONG   0                                       ;
                                     00000000   00DC8 P.AAD:         .LONG   0                                       ;
                                     00000000   00DCC P.AAE:         .LONG   0                                       ;


                                                0000 00000           .ENTRY  SMG$CREATE_VIRTUAL_DISPLAY, Save nothing ; 2512
                     50                    6C    03  83 00002       SUBB3   #3, (AP), DIFF                           ; 2614
                     03                    50    91 00006       CMPB    DIFF, #3                                    ;
                                           08    1B 00009       BLEQU   1$                                          ;
                     50 00000000G          8F    D0 0000B       MOVL    #SMG$_WRONUMARG, R0                         ;
                                                 04 00012       RET                                                 ;
                     06                          6C 91 00013 1$:    CMPB    (AP), #6                                ; 2626
                                                 0A 1F 00016        BLSSU   2$                                      ;
                                           18    AC D5 00018        TSTL    24(AP)                                  ;
                                                 05 13 0001B        BEQL    2$                                      ;
                                           18    AC DD 0001D        PUSHL   CHAR_SET                                ; 2627
                                                 06 11 00020        BRB     3$                                      ;
                     50                    D7    AF 9E 00022 2$:    MOVAB   P.AAE, R0                               ; 2628
                                                 50 DD 00026        PUSHL   R0                                      ;
                     05                          6C 91 00028 3$:    CMPB    (AP), #5                                ; 2623
                                                 0A 1F 0002B        BLSSU   4$                                      ;
                                           14    AC D5 0002D        TSTL    20(AP)                                  ;
                                                 05 13 00030        BEQL    4$                                      ;
                                           14    AC DD 00032        PUSHL   VIDEO_ATTRIBUTES                        ; 2624
                                                 06 11 00035        BRB     5$                                      ;
                     50                    BE    AF 9E 00037 4$:    MOVAB   P.AAD, R0                               ; 2625
                                                 50 DD 0003B        PUSHL   R0                                      ;
                     04                          6C 91 0003D 5$:    CMPB    (AP), #4                                ; 2620
                                                 0A 1F 00040        BLSSU   6$                                      ;
                                           10    AC D5 00042        TSTL    16(AP)                                  ;
                                                 05 13 00045        BEQL    6$                                      ;
                                           10    AC DD 00047        PUSHL   DISPLAY_ATTRIBUTES                      ; 2621
                                                 06 11 0004A        BRB     7$                                      ;
                     50                    A5    AF 9E 0004C 6$:    MOVAB   P.AAC, R0                               ; 2622
                                                 50 DD 00050        PUSHL   R0                                      ;
                     7E                    08    AC 7D 00052 7$:    MOVQ    NUM_COLS, -(SP)                         ; 2618
                                           04    AC DD 00056        PUSHL   NUM_ROWS                                ; 2617
                0000V CF                         06 FB 00059        CALLS   #6, SMG$$CREATE_VIRTUAL_DISPLAY         ;
                                                 04 0005E           RET                                             ; 2630

; Routine Size:  95 bytes,    Routine Base:  _SMG$CODE + 0DD0

;  2379          2631  1 !<BLF/PAGE>
```

```
;  2381          2632  1 %SBTTL 'SMG$DELETE_VIRTUAL_DISPLAY - Delete virtual display'
;  2382          2633  1 GLOBAL ROUTINE SMG$DELETE_VIRTUAL_DISPLAY ( DISPLAY_ID ) =
;  2383          2634  1 !++
;  2384          2635  1 ! FUNCTIONAL DESCRIPTION:
;  2385          2636  1 !
;  2386          2637  1 !       This routine deletes a virtual display.  It is automatically
;  2387          2638  1 !       "unpasted" from any pasteboards on which it is pasted and
;  2388          2639  1 !       its associated buffer space is deallocated.
;  2389          2640  1 !
;  2390          2641  1 ! CALLING SEQUENCE:
;  2391          2642  1 !
;  2392          2643  1 !       ret_status.wlc.v = SMG$DELETE_VIRTUAL_DISPLAY (DISPLAY_ID.rl.r )
;  2393          2644  1 !
;  2394          2645  1 ! FORMAL PARAMETERS:
;  2395          2646  1 !
;  2396          2647  1 !       DISPLAY_ID.rl.r         Id of virtual display to be deleted.
;  2397          2648  1 !
;  2398          2649  1 ! IMPLICIT INPUTS:
;  2399          2650  1 !
;  2400          2651  1 !       NONE
;  2401          2652  1 !
;  2402          2653  1 ! IMPLICIT OUTPUTS:
;  2403          2654  1 !
;  2404          2655  1 !       NONE
;  2405          2656  1 !
;  2406          2657  1 ! COMPLETION STATUS:
;  2407          2658  1 !
;  2408          2659  1 !       SS$_NORMAL      Normal successful completion
;  2409          2660  1 !       SMG$_INVDIS_ID  Invalid display id.
;  2410          2661  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  2411          2662  1 !
;  2412          2663  1 ! SIDE EFFECTS:
;  2413          2664  1 !
;  2414          2665  1 !       NONE
;  2415          2666  1 !--
;  2416          2667  2     BEGIN
;  2417          2668  2     LOCAL
;  2418          2669  2         STATUS,                         ! Status of subroutine calls
;  2419          2670  2         CURR_PP : REF $PP_DECL,         ! Addr of current pasting packet
;  2420          2671  2         DCB     : REF $DCB_DECL;        ! Addr of display control block
;  2421          2672  2
;  2422          2673  2     $SMG$VALIDATE_ARGCOUNT (1, 1);      ! Test for right no. of args
;  2423          2674  2
;  2424          2675  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);   ! Get DCB address
;  2425          2676  2
;  2426          2677  2     CURR_PP = .DCB [DCB_A_PP_NEXT];
;  2427          2678  2
;  2428          2679  2 !+
;  2429          2680  2 ! Loop through all pasteboards we're pasted to, undoing our linkage to
;  2430          2681  2 ! each.
;  2431          2682  2 !-
;  2432          2683  2
;  2433          2684  2     WHILE .CURR_PP NEQ DCB [DCB_A_PP_NEXT] ! While any remain...
;  2434          2685  2     DO
;  2435          2686  3         BEGIN   ! Overall loop
;  2436          2687  3         LOCAL
;  2437          2688  3             PBCB : REF $PBCB_DECL;      ! Addr of pasteboard control blk
```

```
; 2438          2689  3
; 2439          2690  3          PBCB = .CURR_PP [PP_A_PBCB_ADDR];
; 2440          2691  3
; 2441          2692  3          !+
; 2442          2693  3          ! Update pasting packet pointer to next pasting packet, before
; 2443          2694  3          ! the unpaste operation makes current on go away.
; 2444          2695  3          !-
; 2445          2696  3          CURR_PP = .CURR_PP [PP_A_NEXT_DCB];
; 2446          2697  3
; 2447          2698  3          !+
; 2448          2699  3          ! Now we can unpaste this linkage.
; 2449          2700  3          !-
; 2450          2701  3          IF NOT (STATUS = SMG$$UNPASTE_VIRTUAL_DISPLAY (
; 2451          2702  3                                                  .DCB,
; 2452          2703  3                                                  .PBCB ))
; 2453          2704  3          THEN
; 2454          2705  3              RETURN (.STATUS);
; 2455          2706  3
; 2456          2707  3          END;    ! Overall loop
; 2457          2708  2
; 2458          2709  2      !+
; 2459          2710  2      ! Having successfully severed our linkage with all the pasteboards to
; 2460          2711  2      ! to which we were pasted, we can now get rid of the DCB itself.
; 2461          2712  2      ! Before we can delete this DCB we must check to see if there is a
; 2462          2713  2      ! backup DCB in existance.  If so, call outselves recursively to delete
; 2463          2714  2      ! the backup DCB first.
; 2464          2715  2        IF .DCB [DCB_A_BACKUP_DCB] NEQ 0
; 2465          2716  2        THEN
; 2466          2717  3            IF NOT ( STATUS =SMG$DELETE_VIRTUAL_DISPLAY (
; 2467          2718  3                                            DCB [DCB_A_BACKUP_DCB]))
; 2468          2719  2            THEN
; 2469          2720  2                RETURN (.STATUS);
; 2470          2721  2
; 2471          2722  2      ! One remaining chore is to first release the buffer areas whose
; 2472          2723  2      ! addresses are in the DCB.  Recall that the two buffer (text and
; 2473          2724  2      ! attr) were initially allocated as a double-size buffer and split in
; 2474          2725  2      ! two.  This means we can return both at once by supplying the address
; 2475          2726  2      ! of the *he text buffer and a length equal to twice its size.
; 2476          2727  2      !-
; 2477          2728  3          IF NOT (STATUS = LIB$FREE_VM ( %REF (2* .DCB [DCB_L_BUFSIZE]),
; 2478          2729  3                                         DCB [DCB_A_TEXT_BUF]))
; 2479          2730  2          THEN
; 2480          2731  2              RETURN (.STATUS);
; 2481          2732  2
; 2482          2733  2      !+
; 2483          2734  2      ! Free the line characteristics vector
; 2484          2735  2      !-
; 2485          2736  3          IF NOT (STATUS = LIB$FREE_VM (%REF ( .DCB [DCB_W_NO_ROWS] +1),
; 2486          2737  3                                         DCB [DCB_A_LINE_CHAR]))
; 2487          2738  2          THEN
; 2488          2739  2              RETURN ( .STATUS);
; 2489          2740  2
; 2490          2741  2      !+
; 2491          2742  2      ! Free the char_set buffer if there is one.
; 2492          2743  2      !-
; 2493          2744  2          IF .DCB [DCB_A_CHAR_SET_BUF] NEQ 0
; 2494          2745  2          THEN
```

```
; 2495        2746 3          IF NOT (STATUS = LIB$FREE_VM ( DCB [DCB_L_BUFSIZE],
; 2496        2747 3                                         DCB [DCB_A_CHAR_SET_BUF]))
; 2497        2748 2          THEN
; 2498        2749 2              RETURN (.STATUS);
; 2499        2750 2
; 2500        2751 2  !+
; 2501        2752 2  ! If we have a dynamic string containing a border label, free the string
; 2502        2753 2  !-
; 2503        2754 2      IF .DCB [DCB_V_BORDERED]
; 2504        2755 2      THEN
; 2505        2756 3          BEGIN   ! Bordered
; 2506        2757 3          LOCAL
; 2507        2758 3              DESC : REF BLOCK [8,BYTE];  ! Pointer to dynamic string
; 2508        2759 3                                          ! descriptor in DCB
; 2509        2760 3          DESC = DCB [DCB_Q_LABEL_DESC];
; 2510        2761 3          IF .DESC [DSC$A_POINTER] NEQ 0
; 2511        2762 3          THEN
; 2512        2763 4              IF NOT (STATUS = LIB$SFREE1_DD ( .DESC))
; 2513        2764 3              THEN
; 2514        2765 3                  RETURN (.STATUS);
; 2515        2766 2          END;    ! Bordered
; 2516        2767 2  !+
; 2517        2768 2  ! Now the DCB itself...
; 2518        2769 2  ! Before freeing this area, we clobber the byte that makes it
; 2519        2770 2  ! recognizable as a DCB.  That way, if someone inadvertantly tries to
; 2520        2771 2  ! pass us this DCB address as a DCB after having deleted the virtual
; 2521        2772 2  ! display, we can tell that it no longer is a valid DCB.
; 2522        2773 2  !-
; 2523        2774 2      DCB [DCB_B_STRUCT_TYPE] = 0;
; 2524        2775 2      RETURN (LIB$FREE_VM (%REF ( DCB_K_SIZE), DCB ));
; 2525        2776 2
; 2526        2777 1      END;                        ! Routine SMG$DELETE_VIRTUAL_DISPLAY


                                      001C 00000          .ENTRY  SMG$DELETE_VIRTUAL_DISPLAY, Save R2,R3,R4   ; 2633
               54 00000000G     00     9E 00002          MOVAB   LIB$FREE_VM, R4                              ;
               5E               08     C2 00009          SUBL2   #8, SP                                       ;
               01               6C     91 0000C          CMPB    (AP), #1                                     ; 2c73
                                08     13 0000F          BEQL    1$                                           ;
               50 00000000G     8F     D0 00011          MOVL    #SMG$_WRONUMARG, R0                          ;
                                04        00018          RET                                                  ;
               50       04      BC     D0 00019 1$:      MOVL    @DISPLAY_I), R0                              ; 2675
         04    BC       38      A0     D1 0001D          CMPL    56(R0), @DISPLAY_ID                          ;
                                06     12 00022          BNEQ    2$                                           ;
               11       44      A0     91 00024          CMPB    68(R0), #17                                  ;
                                08     13 00028          BEQL    3$                                           ;
               50 00000000G     8F     D0 0002A 2$:      MOVL    #SMG$_INVDIS_ID, R0                          ;
                                04        00031          RET                                                  ;
         04    AE       04      BC     D0 00032 3$:      MOVL    @DISPLAY_ID, DCB                             ;
               52       04      AE     D0 00037          MOVL    DCB, R2                                      ; 2677
               53       20      A2     D0 0003B          MOVL    32(R2), CURR_PP                              ;
               51       20      A2     9E 0003F 4$:      MOVAB   32(R2), R1                                   ; 2684
               51               53     D1 00043          CMPL    CURR_PP, R1                                  ;
                                14     13 00046          BEQL    5$                                           ;
```

```
                                 51      14   A3  D0 00048          MOVL    20(CURR_PP), PBCB                       ;2690
                                 53           63  D0 0004C          MOVL    (CURR_PP), CURR_PP                      ;2696
                                              51  DD 0004F          PUSHL   PBCB                                    ;2703
                                              52  DD 00051          PUSHL   R2                                      ;2702
                          0000V  CF           02  FB 00053          CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY
                                 E4           50  E8 00058          BLBS    STATUS, 4$                              ;2701
                                              04  00058          RET                                             ;2705
                                      40      A2  D5 0005C  5$:     TSTL    64(R2)                                  ;2715
                                              0A  13 0005F          BEQL    6$
                                      40      A2  9F 00061          PUSHAB  64(R2)                                  ;2718
                              98      AF          01  FB 00064      CALLS   #1, SMG$DELETE_VIRTUAL_DISPLAY
                                      61          50  E9 00068      BLBC    STATUS, 9$
                                      10      A2  9F 0006B  6$:     PUSHAB  16(R2)                                  ;2729
                    04    AE        3C      A2  01  78 0006E      ASHL    #1, 60(R2), 4(SP)                       ;2728
                                      04      AE  9F 00074          PUSHAB  4(SP)
                                      64          02  FB 00077      CALLS   #2, LIB$FREE_VM                         ;2729
                                      4F          50  E9 0007A      BLBC    STATUS, 9$
                                      4C      A2  9F 0007D          PUSHAB  76(R2)                                  ;2737
                              04      AE      02  A2  3C 00080      MOVZWL  2(R2), 4(SP)                            ;2736
                                      04      AE  D6 00085          INCL    4(SP)
                                      04      AE  9F 00088          PUSHAB  4(SP)
                                      64          02  FB 0008B      CALLS   #2, LIB$FREE_VM                         ;2737
                                      3B          50  E9 0008E      BLBC    STATUS, 9$
                                      18      A2  D5 00091          TSTL    24(R2)                                  ;2744
                                              0C  13 00094          BEQL    7$
                                      18      A2  9F 00096          PUSHAB  24(R2)                                  ;2747
                                      3C      A2  9F 00099          PUSHAB  60(R2)                                  ;2746
                                      64          02  FB 0009C      CALLS   #2, LIB$FREE_VM                         ;2747
                                      2A          50  E9 0009F      BLBC    STATUS, 9$
                                      15      2F  A2  E9 000A2  7$:     BLBC    47(R2), 8$                          ;2754
                                      51      08  A2  9E 000A6      MOVAB   8(R2), DESC                             ;2760
                                              04  A1  D5 000AA      TSTL    4(DESC)                                 ;2761
                                              0C  13 000AD          BEQL    8$
                                              51  DD 000AF          PUSHL   DESC                                    ;2763
                  00000000G   00              01  FB 000B1      CALLS   #1, LIB$SFREE1_DD
                                      11          50  E9 000B8      BLBC    STATUS, 9$
                                      44      A2  94 000BB  8$:     CLRB    68(R2)                                  ;2774
                                      04      AE  9F 000BE          PUSHAB  DCB                                     ;2775
                              04      AE      70  8F  9A 000C1      MOVZBL  #112, 4(SP)
                                      04      AE  9F 000C6          PUSHAB  4(SP)
                                      64          02  FB 000C9      CALLS   #2, LIB$FREE_VM
                                                  04  000CC  9$:     RET                                             ;2777

; Routine Size:  205 bytes,    Routine Base:  _SMG$CODE + 0E2F

; 2527          2778  1 !<BLF/PAGE>
```

```
;  2529     2779  1 %SBTTL 'SMG$GET_DISPLAY_ATTR - Get  display attributes'
;  2530     2780  1 GLOBAL ROUTINE SMG$GET_DISPLAY_ATTR (
;  2531     2781  1                                     DISPLAY_ID,
;  2532     2782  1                                     HEIGHT,
;  2533     2783  1                                     WIDTH,
;  2534     2784  1                                     DISPLAY_ATTRIBUTES,
;  2535     2785  1                                     VIDEO_ATTRIBUTES,
;  2536     2786  1                                     CHAR_SET
;  2537     2787  1                                         ) =
;  2538     2788  1 !++
;  2539     2789  1 ! FUNCTIONAL DESCRIPTION:
;  2540     2790  1 !
;  2541     2791  1 !       This routine returns attributes of the virtual display.
;  2542     2792  1 !
;  2543     2793  1 ! CALLING SEQUENCE:
;  2544     2794  1 !
;  2545     2795  1 !       ret_status.wlc.v = SMG$GET_DISPLAY_ATTR (
;  2546     2796  1 !                                     DISPLAY_ID.rl.r,
;  2547     2797  1 !                                     HEIGHT.wl.r,
;  2548     2798  1 !                                     WIDTH.wl.r,
;  2549     2799  1 !                                     [,DISPLAY_ATTRIBUTES.wl.r]
;  2550     2800  1 !                                     [,VIDEO_ATTRIBUTES.wl.r]
;  2551     2801  1 !                                     [,CHAR_SET.wl.r])
;  2552     2802  1 !
;  2553     2803  1 ! FORMAL PARAMETERS:
;  2554     2804  1 !
;  2555     2805  1 !       DISPLAY_ID.rl.r         The id of the display for which the
;  2556     2806  1 !                               information is requested.
;  2557     2807  1 !
;  2558     2808  1 !       HEIGHT.wl.r             Height of display in rows
;  2559     2809  1 !
;  2560     2810  1 !       WIDTH.wl.r              Width of display in columns
;  2561     2811  1 !
;  2562     2812  1 !       DISPLAY_ATTRIBUTES.wl.r Optional.  If provided, the current
;  2563     2813  1 !                               default display attributes will be returned.
;  2564     2814  1 !                               These may be:
;  2565     2815  1 !
;  2566     2816  1 !                               SMG$M_BORDER if display is displayed with a
;  2567     2817  1 !                               border.
;  2568     2818  1 !
;  2569     2819  1 !       VIDEO_ATTRIBUTES.wl.r   Optional.   If provided, the current
;  2570     2820  1 !                               default video attributes are returned.  These
;  2571     2821  1 !                               values may be:
;  2572     2822  1 !
;  2573     2823  1 !                               SMG$M_BLINK     displays characters blinking.
;  2574     2824  1 !
;  2575     2825  1 !                               SMG$M_BOLD      displays characters in
;  2576     2826  1 !                                               higher-than-normal intensity.
;  2577     2827  1 !
;  2578     2828  1 !                               SMG$M_REVERSE   displays characters in reverse
;  2579     2829  1 !                                               video -- that is, using the
;  2580     2830  1 !                                               opposite default rendition of
;  2581     2831  1 !                                               the virtual display.
;  2582     2832  1 !
;  2583     2833  1 !                               SMG$M_UNDERLINE displays characters underlined.
;  2584     2834  1 !
;  2585     2835  1 !       CHAR_SET.wb.r   Optional.  If provided, the current default
```

```
;  2586          2836  1 !                           character set code is returned.
;  2587          2837  1 !                           Possible values are:
;  2588          2838  1 !                                   SMG$C_UNITED_KINGDOM
;  2589          2839  1 !                                   SMG$C_ASCII (default)
;  2590          2840  1 !                                   SMG$C_SPEC_GRAPHICS
;  2591          2841  1 !                                   SMG$C_ALT_CHAR
;  2592          2842  1 !                                   SMG$C_ALT_GRAPHICS
;  2593          2843  1 !
;  2594          2844  1 ! IMPLICIT INPUTS:
;  2595          2845  1 !
;  2596          2846  1 !       NONE
;  2597          2847  1 !
;  2598          2848  1 ! IMPLICIT OUTPUTS:
;  2599          2849  1 !
;  2600          2850  1 !       NONE
;  2601          2851  1 !
;  2602          2852  1 ! COMPLETION STATUS:
;  2603          2853  1 !
;  2604          2854  1 !       SS$_NORMAL      Normal successful completion
;  2605          2855  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  2606          2856  1 !
;  2607          2857  1 ! SIDE EFFECTS:
;  2608          2858  1 !
;  2609          2859  1 !       NONE
;  2610          2860  1 !--
;  2611          2861  2     BEGIN
;  2612          2862  2     BUILTIN
;  2613          2863  2         NULLPARAMETER;
;  2614          2864  2
;  2615          2865  2     LOCAL
;  2616          2866  2         DCB : REF $DCB_DECL;                ! Addr of display control block
;  2617          2867  2
;  2618          2868  2     $SMG$VALIDATE_ARGCOUNT (3, 6);          ! Test for right no. of args
;  2619          2869  2
;  2620          2870  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);       ! Get DCB address
;  2621          2871  2
;  2622          2872  2     .HEIGHT = .DCB [DCB_W_NO_ROWS];
;  2623          2873  2     .WIDTH  = .DCB [DCB_W_NO_COLS];
;  2624          2874  2
;  2625          2875  2     IF NOT NULLPARAMETER (DISPLAY_ATTRIBUTES)
;  2626          2876  2     THEN .DISPLAY_ATTRIBUTES = .DCB [DCB_B_DEF_DISPLAY_ATTR];
;  2627          2877  2
;  2628          2878  2     IF NOT NULLPARAMETER (VIDEO_ATTRIBUTES)
;  2629          2879  2     THEN .VIDEO_ATTRIBUTES   = .DCB [DCB_B_DEF_VIDEO_ATTR];
;  2630          2880  2
;  2631          2881  2     IF NOT NULLPARAMETER (CHAR_SET)
;  2632          2882  2     THEN .CHAR_SET = .DCB [DCB_B_DEF_CHAR_SET];
;  2633          2883  2
;  2634          2884  2     RETURN (SS$_NORMAL);
;  2635          2885  1     END;                                ! Routine SMG$GET_DISPLAY_ATTR


                                        0000 00000          .ENTRY  SMG$GET_DISPLAY_ATTR, Save nothing      ; 2780
                50          6C      03   83 00002          SUBB3   #3, (AP), DIFF                          ; 2868
```

```
                                03              50  91 00006          CMPB    DIFF, #3                                        ;
                                                08  1B 00009          BLEQU   1$                                              ;
                                50 00000000G    8F  D0 0000B          MOVL    #SMG$_WRONUMARG, R0                             ;
                                                    04 00012          RET                                                     ;
                                50          04  BC  D0 00013 1$:      MOVL    @DISPLAY_ID, R0                                 ; 2870
                          04    BC          38  A0  D1 00017          CMPL    56(R0), @DISPLAY_ID                             ;
                                                06  12 0001C          BNEQ    2$                                              ;
                                11          44  A0  91 0001E          CMPB    68(R0), #17                                     ;
                                                08  13 00022          BEQL    3$                                              ;
                                50 00000000G    8F  D0 00024 2$:      MOVL    #SMG$_INVDIS_ID, R0                             ;
                                                    04 0002B          RET                                                     ;
                                50          04  BC  D0 0002C 3$:      MOVL    @DISPLAY_ID, DCB                                ;
                          08    BC          02  A0  3C 00030          MOVZWL  2(DCB), @HEIGHT                                 ; 2872
                          0C    BC          06  A0  3C 00035          MOVZWL  6(DCB), @WIDTH                                  ; 2873
                                04              6C  91 0003A          CMPB    (AP), #4                                        ; 2875
                                                0A  1F 0003D          BLSSU   4$                                              ;
                                            10  AC  D5 0003F          TSTL    16(AP)                                          ;
                                                05  13 00042          BEQL    4$                                              ;
                          10    BC          2F  A0  9A 00044          MOVZBL  47(DCB), @DISPLAY_ATTRIBUTES                    ; 2876
                                05              6C  91 00049 4$:      CMPB    (AP), #5                                        ; 2878
                                                0A  1F 0004C          BLSSU   5$                                              ;
                                            14  AC  D5 0004E          TSTL    20(AP)                                          ;
                                                05  13 00051          BEQL    5$                                              ;
                          14    BC          2E  A0  9A 00053          MOVZBL  46(DCB), @VIDEO_ATTRIBUTES                      ; 2879
                                06              6C  91 00058 5$:      CMPB    (AP), #6                                        ; 2881
                                                0A  1F 0005B          BLSSU   6$                                              ;
                                            18  AC  D5 0005D          TSTL    24(AP)                                          ;
                                                05  13 00060          BEQL    6$                                              ;
                          18    BC          30  A0  9A 00062          MOVZBL  48(DCB), @CHAR_SET                              ; 2882
                                50              01  D0 00067 6$:      MOVL    #1, R0                                          ; 2884
                                                    04 0006A          RET                                                     ; 2885

; Routine Size:  107 bytes,    Routine Base:  _SMG$CODE + 0EFC

; 2636          2886  1 !<BLF/PAGE>
```

```
;  2638          2887  1 %SBTTL 'SMG$LABEL_BORDER - Specify label for border'
;  2639          2888  1 GLOBAL ROUTINE SMG$LABEL_BORDER (
;  2640          2889  1                                  DISPLAY_ID,
;  2641          2890  1                                  LABEL_TEXT,
;  2642          2891  1                                  POSITION,
;  2643          2892  1                                  UNITS,
;  2644          2893  1                                  RENDITION_SET,
;  2645          2894  1                                  RENDITION_COMPLEMENT,
;  2646          2895  1                                  CHAR_SET
;  2647          2896  1                                ) =
;  2648          2897  1 !++
;  2649          2898  1 ! FUNCTIONAL DESCRIPTION:
;  2650          2899  1 !
;  2651          2900  1 !       This routine allows the caller to specify what label text is to
;  2652          2901  1 !       be used with this display.  If the specified DISPLAY_ID does not
;  2653          2902  1 !       have the display attribute of SMG$M_BORDER, this attribute is
;  2654          2903  1 !       forced.
;  2655          2904  1 !
;  2656          2905  1 !       If the text parameter is not supplied, this virtual display
;  2657          2906  1 !       no longer has a border text associated with it -- it becomes
;  2658          2907  1 !       an unlabeled border.  If a valid text string is provided, it
;  2659          2908  1 !       replaces the current label text for this border.
;  2660          2909  1 !       If the text (as positioned within the border) does not fit
;  2661          2910  1 !       fully within the border, SMG$_INVARG is returned.
;  2662          2911  1 !       E.g.
;  2663          2912  1 !                +-------LABEL
;  2664          2913  1 !                |       |
;  2665          2914  1 !                +-------+
;  2666          2915  1 !
;  2667          2916  1 !
;  2668          2917  1 !       POSITION and UNITS as a pair specify the starting position of
;  2669          2918  1 !       the label text within the border.
;  2670          2919  1 !       If POSITION is omitted, the top border is assumed.  If UNITS is
;  2671          2920  1 !       omitted, a starting position will be chosen so as to center the
;  2672          2921  1 !       text either horizontally or vertically -- depending on implicit
;  2673          2922  1 !       or explicit POSITION.
;  2674          2923  1 !       If both are omitted, the text will be centered in the top border
;  2675          2924  1 !       line.
;  2676          2925  1 !
;  2677          2926  1 !       The following encoding is used:
;  2678          2927  1 !
;  2679          2928  1 !       POSITION                        UNITS                   SYMBOLIC
;  2680          2929  1 !       --------                        -----                   --------
;  2681          2930  1 !       0 = Top border line             Starting column number  SMG$K_TOP
;  2682          2931  1 !       1 = Bottom border line          Starting column number  SMG$K_BOTTOM
;  2683          2932  1 !       2 = Left border line            Starting row number     SMG$K_LEFT
;  2684          2933  1 !       3 = Right border line           Starting row number     SMG$K_RIGHT
;  2685          2934  1 !
;  2686          2935  1 !       Examples:
;  2687          2936  1 !
;  2688          2937  1 !       POSITION=0, UNITS=4             POSITION=1, UNITS=4
;  2689          2938  1 !
;  2690          2939  1 !       +--LABEL----------------+       +-----------------------+
;  2691          2940  1 !       |                       |       |                       |
;  2692          2941  1 !       |                       |       |                       |
;  2693          2942  1 !       |                       |       |                       |
;  2694          2943  1 !       |                       |       |                       |
```

```
!          |                       |            |                       |
!          +-----------------------+            +--LABEL----------------+
!
!          POSITION=2, UNITS=3                  POSITION=3, UNITS=3
!
!          +-----------------------+            +-----------------------+
!          |                       |            |                       |
!          L                       |            |                       L
!          A                       |            |                       A
!          B                       |            |                       B
!          E                       |            |                       E
!          L                       |            |                       L
!          |                       |            |                       |
!          +-----------------------+            +-----------------------+
!
!
! CALLING SEQUENCE:
!
!          ret_status.wlc.v = SMG$LABEL_BORDER (
!                                         DISPLAY_ID.rl.r
!                                         [,LABEL_TEXT.rt.dx]
!                                         [,POSITION.rl.r]
!                                         [,UNITS.rl.r]
!                                         [,RENDITION_SET.rl.r]
!                                         [,RENDITION_COMPLEMENT.rlr.]
!                                         [,CHAR_SET.rl.r])
!
! FORMAL PARAMETERS:
!
!          DISPLAY_ID.rl.r The display id of the virtual display whose
!                          border is to be labeled.  This display must have
!                          the display attribute of SMG$M_BORDER.
!
!          LABEL_TEXT.rt.dx            [Optional].  If supplied becomes the new
!                                      label for this display's border.  If
!                                      omitted, display becomes unlabeled.
!
!          POSITION.rl.r   [Optional].
!                          Specifies which border will contain label.
!                          If omitted, default to top border.
!
!          UNITS.rl.r      [Optional].
!                          Specifies where within the border the text label
!                          will start.
!                          If omitted, center in line indicated by POSITION
!
!          RENDITION_SET.rl.r          [Optional].
!                                      Each 1 bit attribute in this parameter
!                                      causes the corresponding attribute to
!                                      be set in the display.  (See below for
!                                      list of settable attributes.)
!
!          RENDITION_COMPLEMENT.rl.r           [Optional].
!                                      Each 1 bit attribute in this parameter
!                                      causes the corresponding attribute to
```

```
;  2752     3001  1 !                          be complemented in the display.  (See
;  2753     3002  1 !                          below for list of complementable
;  2754     3003  1 !                          attributes.)
;  2755     3004  1 !
;  2756     3005  1 !       If the same bit is specified in both the RENDITION_SET parameter
;  2757     3006  1 !       and in the RENDITION_COMPLEMENT parameter, the application is
;  2758     3007  1 !       RENDITION_SET followed by RENDITION complement.  Using these two
;  2759     3008  1 !       parameters together the caller can exercise arbitrary and
;  2760     3009  1 !       independent control over each attribute on a single call.  On an
;  2761     3010  1 !       attribute by attribute basis he can cause the following
;  2762     3011  1 !       transformations:
;  2763     3012  1 !
;  2764     3013  1 !               SET     COMPLEMENT      Action
;  2765     3014  1 !               ---     ----------      ------
;  2766     3015  1 !               0       0               Attribute unchanged.
;  2767     3016  1 !               1       0               Attribute set to "on"
;  2768     3017  1 !               0       1               Attribute set to complement of
;  2769     3018  1 !                                       current setting.
;  2770     3019  1 !               1       1               Attribute set to "off".
;  2771     3020  1 !
;  2772     3021  1 !
;  2773     3022  1 !       Attributes which can be manipulated in this manner are:
;  2774     3023  1 !
;  2775     3024  1 !       SMG$M_BLINK  displays characters blinking.
;  2776     3025  1 !       SMG$M_BOLD  displays characters in higher-than-normal
;  2777     3026  1 !                   intensity.
;  2778     3027  1 !       SMG$M_REVERSE  displays characters in reverse video -- that is,
;  2779     3028  1 !                      using the opposite default rendition of the
;  2780     3029  1 !                      virtual display.
;  2781     3030  1 !       SMG$M_UNDERLINE  displays characters underlined.
;  2782     3031  1 !
;  2783     3032  1 !       CHAR_SET.rl.r   [Optional].  If provided, the character set to
;  2784     3033  1 !                       be used in displaying the label.
;  2785     3034  1 !                       Recognized values are:
;  2786     3035  1 !                                       SMG$C_UNITED_KINGDOM
;  2787     3036  1 !                                       SMG$C_ASCII (default)
;  2788     3037  1 !                                       SMG$C_SPEC_GRAPHICS
;  2789     3038  1 !                                       SMG$C_ALT_CHAR
;  2790     3039  1 !                                       SMG$C_ALT_GRAPHICS
;  2791     3040  1 !
;  2792     3041  1 !
;  2793     3042  1 ! IMPLICIT INPUTS:
;  2794     3043  1 !
;  2795     3044  1 !       None
;  2796     3045  1 !
;  2797     3046  1 ! IMPLICIT OUTPUTS:
;  2798     3047  1 !
;  2799     3048  1 !       None
;  2800     3049  1 !
;  2801     3050  1 ! COMPLETION STATUS:
;  2802     3051  1 !
;  2803     3052  1 !       SS$_NORMAL      Normal successful completion
;  2804     3053  1 !       SMG$_INVDIS_ID  Invalid virtual display id.
;  2805     3054  1 !       SMG$_INVARG     Positioning and/or units when considered with
;  2806     3055  1 !                       length of text results in a position that is
;  2807     3056  1 !                       outside of the border area.
;  2808     3057  1 !       SMG$_WRONUMARG  Wrong number of arguments.
```

```
;  2809     3058  1 !
;  2810     3059  1 ! SIDE EFFECTS:
;  2811     3060  1 !
;  2812     3061  1 !       NONE
;  2813     3062  1 !--
;  2814     3063  2     BEGIN
;  2815     3064  2
;  2816     3065  2     LITERAL
;  2817     3066  2         K_SET_ARG = 5,
;  2818     3067  2         K_COMP_ARG= 6;
;  2819     3068  2
;  2820     3069  2     BUILTIN
;  2821     3070  2         NULLPARAMETER;
;  2822     3071  2
;  2823     3072  2     LOCAL
;  2824     3073  2         LUNITS,                          ! Implicit or explicit UNITS
;  2825     3074  2         LPOS,                            ! Implicit or explicit POSITION
;  2826     3075  2         REND_CODE,                       ! Rendition to be applied to
;  2827     3076  2                                          ! border label
;  2828     3077  2         STATUS,                          ! Status of subroutine calls
;  2829     3078  2         DESC : REF BLOCK [,BYTE],        ! Pointer to dynamic string
;  2830     3079  2                                          ! descriptor in DCB for border
;  2831     3080  2                                          ! label.
;  2832     3081  2         DCB : REF $DCB_DECL;             ! Addr. of display control block
;  2833     3082  2
;  2834     3083  2     $SMG$VALIDATE_ARGCOUNT (1, 7);       ! Test for right no. of args
;  2835     3084  2
;  2836     3085  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);    ! Get addr of DCB
;  2837     3086  2
;  2838     3087  2 !+
;  2839     3088  2 ! Get a copy of the label.
;  2840     3089  2 !-
;  2841     3090  2     DESC = DCB [DCB_Q_LABEL_DESC];
;  2842     3091  2
;  2843     3092  2     IF NULLPARAMETER (LABEL_TEXT)
;  2844     3093  2     THEN
;  2845     3094  3         BEGIN   ! No text specified
;  2846     3095  3         RETURN (LIB$SFREE1_DD ( .DESC));
;  2847     3096  2         END;    ! No text specified
;  2848     3097  2
;  2849     3098  3     IF NOT (STATUS = LIB$SCOPY_DXDX (.LABEL_TEXT, .DESC))
;  2850     3099  2     THEN
;  2851     3100  2         RETURN (.STATUS);
;  2852     3101  2
;  2853     3102  2 !+
;  2854     3103  2 ! Check to see if combination of POSITION and UNITS fit.
;  2855     3104  2 !-
;  2856     3105  3     LPOS = ( IF NOT NULLPARAMETER (POSITION) THEN ..POSITION
;  2857     3106  2                                              ELSE 0);  ! Default to top row
;  2858     3107  2
;  2859     3108  2     CASE .LPOS FROM SMG$K_TOP TO SMG$K_RIGHT OF
;  2860     3109  2     SET
;  2861     3110  2         [SMG$K_TOP,SMG$K_BOTTOM]:        ! Top or bottom row
;  2862     3111  3             BEGIN
;  2863     3112  4             LUNITS = ( IF NOT NULLPARAMETER (UNITS)
;  2864     3113  4                        THEN ..UNITS
;  2865     3114  4                        ELSE     ! Center horizontally
```

```
;  2866          3115  6                          ((.DCB[DCB_W_NO_COLS] -.DESC [DSC$W_LENGTH])
;  2867          3116  6                            / 2) + 2 );
;  2868          3117
;  2869          3118  3                    IF .LUNITS LEQ 0              OR
;  2870          3119  3                       .LUNITS + .DESC[DSC$W_LENGTH] GTR .DCB [DCB_W_NO_COLS] +2
;  2871          3120  3                    THEN
;  2872          3121  4                        BEGIN
;  2873          3122  4                        LIB$SFREE1_DD (.DESC) ; ! Release our dynamic string
;  2874          3123  4                        RETURN (SMG$_INVARG);
;  2875          3124  4                        END;
;  2876          3125  3                    END;
;  2877          3126  3
;  2878          3127  3                [SMG$K_LEFT,SMG$K_RIGHT]:       ! Left or right column
;  2879          3128  3                    BEGIN
;  2880          3129  4                    LUNITS = ( IF NOT NULLPARAMETER (UNITS)
;  2881          3130  4                               THEN ..UNITS
;  2882          3131  4                               ELSE     ! Center vertically
;  2883          3132  6                                   ((.DCB[DCB_W_NO_ROWS] -.DESC[DSC$W_LENGTH])
;  2884          3133  6                                     / 2) + 2 );
;  2885          3134  4
;  2886          3135  3                    IF .LUNITS LEQ 0              OR
;  2887          3136  3                       .LUNITS + .DESC[DSC$W_LENGTH] GTR .DCB [DCB_W_NO_ROWS] +2
;  2888          3137  3                    THEN
;  2889          3138  4                        BEGIN
;  2890          3139  4                        LIB$SFREE1_DD (.DESC) ; ! Release our dynamic string
;  2891          3140  4                        RETURN (SMG$_INVARG);
;  2892          3141  4                        END;
;  2893          3142  3                    END;
;  2894          3143  2
;  2895          3144  2                    [OUTRANGE]:
;  2896          3145  2                        RETURN (SMG$_INVARG);
;  2897          3146  2                TES;
;  2898          3147  2
;  2899          3148  2                DCB [DCB_B_LABEL_POS] = .LPOS;
;  2900          3149  2                DCB [DCB_W_LABEL_UNITS] = .LUNITS;
;  2901          3150  2
;  2902          3151  2            !+
;  2903          3152  2            ! If UNITS parameter was omitted we centered the label.  Make a note of
;  2904          3153  2            ! this fact so that if he later does a CHANGE_VIRTUAL_DISPLAY we can
;  2905          3154  2            ! again center it in its new "center".
;  2906          3155  2                DCB [DCB_V_LABEL_CENTER] = 0;
;  2907          3156  2                IF NULLPARAMETER (UNITS)
;  2908          3157  2                THEN
;  2909          3158  2                    DCB [DCB_V_LABEL_CENTER] = 1;
;  2910          3159  2
;  2911          3160  2            !+
;  2912          3161  2            ! Calc. REND_CODE as a function of callers rendition arguments and
;  2913          3162  2            ! the default rendition in the DCB.
;  2914          3163  2            !-
;  2915          3164  2                $SMG$SET_REND_CODE (K_SET_ARG, K_COMP_ARG);
;  2916          3165  2                                   ! macro to use caller's args if present
;  2917          3166  2
;  2918          3167  2                DCB [DCB_B_LABEL_REND] = .REND_CODE;
;  2919          3168  2
;  2920          3169  2            !+
;  2921          3170  2            ! Deal with alternate character set.
;  2922          3171  2            !-
```

```
;  2923          3172  2     IF NOT NULLPARAMETER(CHAR_SET)
;  2924          3173  2     THEN
;  2925          3174  2         BEGIN
;  2926          3175  2         CASE ..CHAR_SET FROM SMG$C_UNITED_KINGDOM
;  2927          3176  2                         TO SMG$C_ALT_GRAPHICS OF
;  2928          3177  2         SET
;  2929          3178  2             [SMG$C_UNITED_KINGDOM,
;  2930          3179  2              SMG$C_ASCII,
;  2931          3180  2              SMG$C_SPEC_GRAPHICS,
;  2932          3181  2              SMG$C_ALT_CHAR,
;  2933          3182  2              SMG$C_ALT_GRAPHICS]:
;  2934          3183  2                                 DCB [DCB_B_LABEL_CHAR_SET] = ..CHAR_SET;
;  2935          3184  2
;  2936          3185  2             [INRANGE, OUTRANGE]:
;  2937          3186  2                                 RETURN (SMG$_INVARG);
;  2938          3187  2         TES;
;  2939          3188  2         END
;  2940          3189  2     ELSE        ! Use default for virtual display
;  2941          3190  2         DCB [DCB_B_LABEL_CHAR_SET] = .DCB [DCB_B_DEF_CHAR_SET];
;  2942          3191  2
;  2943          3192  2     DCB [DCB_V_BORDERED] = 1;   ! Force bordered attribute in case it
;  2944          3193  2                                 ! wasn't previously.
;  2945          3194  2 !+
;  2946          3195  2 ! We now need to recalculate the constants in the pasting packet.
;  2947          3196  2 ! We may be making the transition from unbordered to bordered, so
;  2948          3197  2 ! this virtual display now has a bigger footprint in the pasteboard
;  2949          3198  2 ! buffer, and some display which previously was not occluded may now be.
;  2950          3199  2 ! Even if we were previously bordered, the size and position of our
;  2951          3200  2 ! label may have changed.
;  2952          3201  2 ! If we are not batched at the display level, recalc. pasting packet
;  2953          3202  2 ! constants and initiate output.  Else, just remember that we need to do
;  2954          3203  2 ! it later when batch level drops to zero.
;  2955          3204  2 !-
;  2956          3205  2     IF .DCB [DCB_L_BATCH_LEVEL] EQL 0
;  2957          3206  2     THEN
;  2958          3207  3         BEGIN   ! Do it now
;  2959          3208  4         IF NOT (STATUS = SMG$$RECALC_PP_FIELDS ( .DCB))
;  2960          3209  3         THEN
;  2961          3210  3             RETURN (.STATUS);
;  2962          3211  3
;  2963          3212  3         RETURN ( SMG$$CHECK_FOR_OUTPUT_DCB ( .DCB, SMG$C_LABEL_BORDER));
;  2964          3213  3         END     ! Do it now
;  2965          3214  3
;  2966          3215  2     ELSE
;  2967          3216  2
;  2968          3217  3         BEGIN   ! Defer the action
;  2969          3218  3         DCB [DCB_V_PP_MISMATCH] =1; ! Remember for later
;  2970          3219  2         END;    ! Defer the action
;  2971          3220  2
;  2972          3221  2     RETURN (SS$_NORMAL);
;  2973          3222  1     END;                            ! Routine SMG$LABEL_BORDER



                                        00FC 00000          .ENTRY  SMG$LABEL_BORDER, Save R2,R3,R4,R5,R6,R7    ; 2888
```

```
                                 57 00000000G   00  9E 00002          MOVAB   LIB$SFREE1_DD, R7                                ;
                              50 6C             01  83 00009          SUBB3   #1, (AP), DIFF                                  ; 3083
                                 06             50  91 0000D          CMPB    DIFF, #6                                        ;
                                                08  1B 00010          BLEQU   1$                                              ;
                                 50 00000000G   8F  D0 00012          MOVL    #SMG$_WRONUMARG, R0                             ;
                                                04  00019          RET                                                     ;
                                 50       04    BC  D0 0001A 1$:     MOVL    @DISPLAY_ID, R0                                 ; 3085
                           04    BC       38    A0  D1 0001E          CMPL    56(R0), @DISPLAY_ID                             ;
                                                06  12 00023          BNEQ    2$                                              ;
                                 11       44    A0  91 00025          CMPB    68(R0), #17                                     ;
                                                08  13 00029          BEQL    3$                                              ;
                                 50 00000000G   8F  D0 0002B 2$:     MOVL    #SMG$_INVDIS_ID, R0                             ;
                                                04  00032          RET                                                     ;
                                 52       04    BC  D0 00033 3$:     MOVL    @DISPLAY_ID, DCB                                ;
                                 54       08    A2  9E 00037          MOVAB   8(R2), DESC                                     ; 3090
                                 02             6C  91 0003B          CMPB    (AP), #2                                        ; 3092
                                                05  1F 0003E          BLSSU   4$                                              ;
                                          08    AC  D5 00040          TSTL    8(AP)                                           ;
                                                06  12 00043          BNEQ    5$                                              ;
                                                54  DD 00045 4$:     PUSHL   DESC                                            ; 3095
                                 67             01  FB 00047          CALLS   #1, LIB$SFREE1_DD                               ;
                                                04  0004A          RET                                                     ;
                                                54  DD 0004B 5$:     PUSHL   DESC                                            ; 3098
                                          08    AC  DD 0004D          PUSHL   LABEL_TEXT                                      ;
                        00000000G  00           02  FB 00050          CALLS   #2, LIB$SCOPY_DXDX                              ;
                                 56             50  D0 00057          MOVL    R0, STATUS                                      ;
                                 03             56  E8 0005A          BLBS    STATUS, 6$                                      ;
                                             010B  31  0005D          BRW     29$                                             ;
                                 03             6C  91 00060 6$:     CMPB    (AP), #3                                        ; 3105
                                                0B  1F 00063          BLSSU   7$                                              ;
                                          0C    AC  D5 00065          TSTL    12(AP)                                          ;
                                                06  13 00068          BEQL    7$                                              ;
                                 55       0C    BC  D0 0006A          MOVL    @POSITION, LPOS                                 ;
                                                02  11 0006E          BRB     8$                                              ;
                                                55  D4 00070 7$:      CLRL    LPOS                                            ;
                           03       00           55  CF 00072 8$:      CASEL   LPOS, #0, #3                                    ; 3108
                     0039  0039     000A      000A      00076 9$:      .WORD   10$-9$,-                                        ;
                                                                                10$-9$,-                                        ;
                                                                                13$-9$,-                                        ;
                                                                                13$-9$                                          ;
                                                68  11 0007E          BRB     18$                                             ; 3145
                                 04             6C  91 00080 10$:     CMPB    (AP), #4                                        ; 3112
                                                0B  1F 00083          BLSSU   11$                                             ;
                                          10    AC  D5 00085          TSTL    16(AP)                                          ;
                                                06  13 00088          BEQL    11$                                             ;
                                 53       10    BC  D0 0008A          MOVL    @UNITS, LUNITS                                  ; 3113
                                                11  11 0008E          BRB     12$                                             ;
                                 50       06    A2  3C 00090 11$:     MOVZWL  6(DCB), R0                                      ; 3115
                                 51             64  3C 00094          MOVZWL  (DESC), R1                                      ;
                                 50             51  C2 00097          SUBL2   R1, R0                                          ;
                                 50             02  C6 0009A          DIVL2   #2, R0                                          ; 3116
                                 53       02    A0  9E 0009D          MOVAB   2(R0), LUNITS                                   ;
                                                40  15 000A1 12$:     BLEQ    17$                                             ; 3118
                                 51             64  3C 000A3          MOVZWL  (DESC), R1                                      ; 3119
                                 51             53  C0 000A6          ADDL2   LUNITS, R1                                      ;
                                 50       06    A2  3C 000A9          MOVZWL  6(DCB), R0                                      ;
                                                2C  11 000AD          BRB     16$                                             ;
```

```
                                04              6C  91  000AF 13$:    CMPB    (AP), #4                                        ; 3129
                                                0B  1F  000B2         BLSSU   14$                                             ;
                                    10          AC  D5  000B4         TSTL    16(AP)                                          ;
                                                06  13  000B7         BEQL    14$                                             ;
                                53              10  BC  D0  000B9     MOVL    @UNITS, LUNITS                                  ; 3130
                                                10  11  000BD         BRB     15$                                             ;
                                53          02  A2  3C  000BF 14$:    MOVZWL  2(DCB), R3                                      ; 3132
                                50              64  3C  000C3         MOVZWL  (DESC), R0                                      ;
                                53              50  C2  000C6         SUBL2   R0, R3                                          ;
                                53              02  C6  000C9         DIVL2   #2, R3                                          ; 3133
                                53              02  C0  000CC         ADDL2   #2, LUNITS                                      ;
                                                12  15  000CF 15$:    BLEQ    17$                                             ; 3135
                                51              64  3C  000D1         MOVZWL  (DESC), R1                                      ; 3136
                                51              53  C0  000D4         ADDL2   LUNITS, R1                                      ;
                                50          02  A2  3C  000D7         MOVZWL  2(DCB), R0                                      ;
                                50              02  C0  000DB 16$:    ADDL2   #2, R0                                          ;
                                50              51  D1  000DF         CMPL    R1, R0                                          ;
                                                07  15  000E1         BLEQ    19$                                             ;
                                                54  DD  000E3 17$:    PUSHL   DESC                                            ; 3139
                                67              01  FB  000E5         CALLS   #1, LIB$SFREE1_DD                               ;
                                                57  11  000E8 18$:    BRB     25$                                             ; 3140
                            31  A2              55  90  000EA 19$:    MOVB    LPOS, 49(DCB)                                   ; 3148
                            2C  A2              53  B0  000EE         MOVW    LUNITS, 44(DCB)                                 ; 3149
                            34  A2              04  8A  000F2         BICB2   #4, 52(DCB)                                     ; 3155
                                04              6C  91  000F6         CMPB    (AP), #4                                        ; 3156
                                                05  1F  000F9         BLSSU   20$                                             ;
                                    10          AC  D5  000FB         TSTL    16(AP)                                          ;
                                                04  12  000FE         BNEQ    21$                                             ;
                            34  A2              04  88  00100 20$:    BISB2   #4, 52(DCB)                                     ; 3158
                                50          2E  A2  9A  00104 21$:    MOVZBL  46(DCB), REND_CODE                              ; 3164
                                05              6C  91  00108         CMPB    (AP), #5                                        ;
                                                09  1F  0010B         BLSSU   22$                                             ;
                                    14          AC  D5  0010D         TSTL    20(AP)                                          ;
                                                04  13  00110         BEQL    22$                                             ;
                                50              14  BC  C8  00112     BISL2   @RENDITION_SET, REND_CODE                       ;
                                06              6C  91  00116 22$:    CMPB    (AP), #6                                        ;
                                                09  1F  00119         BLSSU   23$                                             ;
                                    18          AC  D5  0011B         TSTL    24(AP)                                          ;
                                                04  13  0011E         BEQL    23$                                             ;
                                50              18  BC  CC  00120     XORL2   @RENDITION_COMPLEMENT, REND_CODE                ;
                            33  A2              50  90  00124 23$:    MOVB    REND_CODE, 51(DCB)                              ; 3167
                                07              6C  91  00128         CMPB    (AP), #7                                        ; 3172
                                                23  1F  0012B         BLSSU   27$                                             ;
                                    1C          AC  D5  0012D         TSTL    28(AP)                                          ;
                                                1E  13  00130         BEQL    27$                                             ;
                                    00      1C  BC  CF  00132         CASEL   @CHAR_SET, #0, #4                               ; 3175
0012                        0012        0012    0012    00137 24$:    .WORD   26$-24$,-                                       ;
                                                0012    0013F                 26$-24$,-                                       ;
                                                                              26$-24$,-                                       ;
                                                                              26$-24$,-                                       ;
                                                                              26$-24$                                         ;
                                50 00000000G    8F  D0  00141 25$:    MOVL    #SMG$_INVARG, R0                                ; 3186
                                                04  00148             RET                                                     ;
                            32  A2          1C  BC  90  00149 26$:    MOVB    @CHAR_SET, 50(DCB)                              ; 3183
                                                05  11  0014E         BRB     28$                                             ; 3172
                            32  A2          30  A2  90  00150 27$:    MOVB    48(DCB), 50(DCB)                                ; 3190
                            2F  A2              01  88  00155 28$:    BISB2   #1, 47(DCB)                                     ; 3192
```

```
                                 1C   A2  D5 00159          TSTL    28(DCB)                                  ; 3205
                                      1D  12 0015C          BNEQ    31$                                      ;
                                      52  DD 0015E          PUSHL   DCB                                      ; 3208
                          0000V  CF       01  FB 00160      CALLS   #1, SMG$$RECALC_PP_FIELDS                ;
                                 56       50  D0 00165      MOVL    R0, STATUS                               ;
                                 04       56  E8 00168      BLBS    STATUS, 30$                              ;
                                 50       56  D0 0016B 29$: MOVL    STATUS, R0                               ; 3210
                                          04  00016E        RET                                              ;
                                      1C  DD 0016F 30$:     PUSHL   #28                                      ; 3212
                                      52  DD 00171          PUSHL   DCB                                      ;
                      00000000G   00      02  FB 00173      CALLS   #2, SMG$$CHECK_FOR_OUTPUT_DCB            ;
                                          04  0017A        RET                                              ;
                            34   A2       08  88 0017B 31$: BISB2   #8, 52(DCB)                             ; 3218
                                 50       01  D0 0017F      MOVL    #1, R0                                   ; 3221
                                          04  00182        RET                                              ; 3222

; Routine Size:  387 bytes,    Routine Base:  _SMG$CODE + 0F67

; 2974          3223  1 !<BLF/PAGE>
```

```
;  2976          3224  1 %SBTTL 'SMG$MOVE_VIRTUAL_DISPLAY - Move previously pasted virtual display'
;  2977          3225  1 GLOBAL ROUTINE SMG$MOVE_VIRTUAL_DISPLAY (
;  2978          3226  1                                          DISPLAY_ID,
;  2979          3227  1                                          PASTEBOARD_ID,
;  2980          3228  1                                          PASTEBOARD_ROW,
;  2981          3229  1                                          PASTEBOARD_COL
;  2982          3230  1                                         ) =
;  2983          3231  1 !++
;  2984          3232  1 ! FUNCTIONAL DESCRIPTION:
;  2985          3233  1 !
;  2986          3234  1 !       The specified virtual display is moved with respect to the
;  2987          3235  1 !       position where it is currently pasted to the specified pasteboard
;  2988          3236  1 !       preserving the pasting order.  If the display is not currently
;  2989          3237  1 !       pasted, it is pasted at the top of the pasting order in the
;  2990          3238  1 !       position specified.
;  2991          3239  1 !       This call is not permitted while display batching is in effect.
;  2992          3240  1 !
;  2993          3241  1 ! CALLING SEQUENCE:
;  2994          3242  1 !
;  2995          3243  1 !       ret_status.wlc.v = SMG$MOVE_VIRTUAL_DISPLAY (
;  2996          3244  1 !                                                  DISPLAY_ID.rl.r,
;  2997          3245  1 !                                                  PASTEBOARD_ID.rl.r,
;  2998          3246  1 !                                                  PASTEBOARD_ROW.rl.r,
;  2999          3247  1 !                                                  PASTEBOARD_COL.rl.r)
;  3000          3248  1 !
;  3001          3249  1 ! FORMAL PARAMETERS:
;  3002          3250  1 !
;  3003          3251  1 !       DISPLAY_ID.rl.r         Id of virtual display to be moved.
;  3004          3252  1 !
;  3005          3253  1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard on
;  3006          3254  1 !                               which the movement is to take place.
;  3007          3255  1 !
;  3008          3256  1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
;  3009          3257  1 !                               row 1 of the specified virtual display.
;  3010          3258  1 !
;  3011          3259  1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
;  3012          3260  1 !                               column 1 of the specified virtual
;  3013          3261  1 !                               display.
;  3014          3262  1 !
;  3015          3263  1 ! IMPLICIT INPUTS:
;  3016          3264  1 !
;  3017          3265  1 !       None
;  3018          3266  1 !
;  3019          3267  1 ! IMPLICIT OUTPUTS:
;  3020          3268  1 !
;  3021          3269  1 !       None
;  3022          3270  1 !
;  3023          3271  1 ! COMPLETION STATUS:
;  3024          3272  1 !
;  3025          3273  1 !       SS$_NORMAL          Normal successful completion
;  3026          3274  1 !       SMG$_INVDIS_ID      Invalid virtual display id.
;  3027          3275  1 !       SMG$_INVPAS_ID      Invalid pasteboard id.
;  3028          3276  1 !       SMG$_WRONUMARG      Wrong number of arguments.
;  3029          3277  1 !       SMG$_ILLBATFNC      Display is being batched, this operation is illegal.
;  3030          3278  1 !
;  3031          3279  1 ! SIDE EFFECTS:
;  3032          3280  1 !
```

```
;  3033     3281  1 !       NONE
;  3034     3282  1 !--
;  3035     3283  2     BEGIN
;  3036     3284  2     BUILTIN
;  3037     3285  2         AP,
;  3038     3286  2         CALLG;
;  3039     3287  2
;  3040     3288  2     LOCAL
;  3041     3289  2         STATUS,                         ! Status of subroutine calls
;  3042     3290  2
;  3043     3291  2         PP      : REF $PP_DECL,         ! Addr of the pasting packet
;  3044     3292  2         DCB     : REF $DCB_DECL,        ! Addr. of display control block
;  3045     3293  2         PBCB    : REF $PBCB_DECL;       ! Addr of pasteboard control block
;  3046     3294  2
;  3047     3295  2     $SMG$VALIDATE_ARGCOUNT (4, 4);      ! Test for right no. of args
;  3048     3296  2
;  3049     3297  2 !+
;  3050     3298  2 ! Get addresses of associated virtual display control block and
;  3051     3299  2 ! pasteboard control block, validating both the display id and the
;  3052     3300  2 ! pasteboard id.
;  3053     3301  2 !-
;  3054     3302  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);           ! Get addr of DCB
;  3055     3303  2     $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);      ! Get addr of PBCB
;  3056     3304  2
;  3057     3305  2 !+
;  3058     3306  2 ! Give an error if the display is batched.
;  3059     3307  2 !-
;  3060     3308  2
;  3061     3309  2     IF .DCB[DCB_L_BATCH_LEVEL] NEQ 0
;  3062     3310  2     THEN
;  3063     3311  2         RETURN  SMG$_ILLBATFNC;
;  3064     3312  2
;  3065     3313  2 !+
;  3066     3314  2 ! Determine if this virtual display is already pasted to this
;  3067     3315  2 ! pasteboard.  If it is we can do the MOVE.  If it isn't we'll do a
;  3068     3316  2 ! PASTE at the specified position.
;  3069     3317  2 !-
;  3070     3318  2     IF NOT SMG$$LOCATE_PP( .DCB, .PBCB, PP)
;  3071     3319  2     THEN
;  3072     3320  2         RETURN SMG$$PASTE_VIRTUAL_DISPLAY(.DCB,.PBCB,
;  3073     3321  2                         .PASTEBOARD_ROW,.PASTEBOARD_COL);
;  3074     3322  2
;  3075     3323  2 !+
;  3076     3324  2 ! Set new row and column into pasting packet
;  3077     3325  2 !-
;  3078     3326  2     PP [PP_W_ROW]       = ..PASTEBOARD_ROW;
;  3079     3327  2     PP [PP_W_COL]       = ..PASTEBOARD_COL;
;  3080     3328  2
;  3081     3329  2 !+
;  3082     3330  2 ! Recalc. occlusions.
;  3083     3331  2 !-
;  3084     3332  3     IF NOT ( STATUS = SMG$$CHECK_OCCLUSION ( .PBCB))
;  3085     3333  2     THEN
;  3086     3334  2         RETURN (.STATUS);
;  3087     3335  2
;  3088     3336  2 !+
;  3089     3337  2 ! Recalculate the transformation constants needed to copy this display's
```

```
;  3090          3338  2 ! buffers into the associated window's buffers.
;  3091          3339  2 !-
;  3092          3340  3     IF NOT ( STATUS = SMG$$CALC_PASTE_TRANSF (.PP))
;  3093          3341  2     THEN
;  3094          3342  2         RETURN (.STATUS);
;  3095          3343  2
;  3096          3344  2     RETURN (SMG$$CHECK_FOR_OUTPUT_PBCB (.PBCB));
;  3097          3345  2
;  3098          3346  1     END;                          ! Routine SMG$MOVE_VIRTUAL_DISPLAY
```

```
                                                001C 00000          .ENTRY  SMG$MOVE_VIRTUAL_DISPLAY, Save R2,R3,R4      ; 3225
                                    54 00000000'  EF   9E 00002      MOVAB   PBD_L_COUNT, R4                              ;
                                    5E            04   C2 00009      SUBL2   #4, SP                                       ;
                                    04                6C   91 0000C  CMPB    (AP), #4                                     ; 3295
                                                      08   13 0000F  BEQL    1$                                           ;
                                    50 00000000G  8F  D0   00011     MOVL    #SMG$_WRONUMARG, R0                          ;
                                                      04   00018     RET                                                  ;
                                    50        04  BC  D0   00019 1$: MOVL    @DISPLAY_ID, R0                              ; 3302
                                 04 BC        38  A0  D1   0001D     CMPL    56(R0), @DISPLAY_ID                          ;
                                                      06   12 00022  BNEQ    2$                                           ;
                                    11        44  A0  91   00024     CMPB    68(R0), #17                                  ;
                                                      08   13 00028  BEQL    3$                                           ;
                                    50 00000000G  8F  D0   0002A 2$: MOVL    #SMG$_INVDIS_ID, R0                          ;
                                                      04   00031     RET                                                  ;
                                    52        04  BC  D0   00032 3$: MOVL    @DISPLAY_ID, DCB                             ;
                                    50        08  BC  D0   00036     MOVL    @PASTEBOARD_ID, R0                           ; 3303
                                                      0A   19 0003A  BLSS    4$                                           ;
                                    64                50   D1 0003C  CMPL    R0, PBD_L_COUNT                              ;
                                                      05   14 0003F  BGTR    4$                                           ;
                              08        44  A4              50   E0 00041  BBS     R0, PBD_V_PB_AVAIL, 5$                       ;
                                    50 00000000G  8F  D0   00046 4$: MOVL    #SMG$_INVPAS_ID, R0                          ;
                                                      04   0004D     RET                                                  ;
                                    53        04 A440 D0   0004E 5$: MOVL    PBD_A_PBCB[R0], PBCB                         ;
                                              1C  A2  D5   00053     TSTL    28(DCB)                                      ; 3309
                                                      08   13 00056  BEQL    6$                                           ;
                                    50 00000000G  8F  D0   00058     MOVL    #SMG$_ILLBATFNC, R0                          ; 3311
                                                      04   0005F     RET                                                  ;
                                          400C  8F  BB   00060 6$: PUSHR   #^M<R2,R3,SP>                                ; 3318
                                 0000V    CF            03   FB 00064  CALLS   #3, SMG$$LOCATE_PP                          ;
                                    0C                50   E8 00069  BLBS    R0, 7$                                       ;
                                    7E        0C  AC  7D   0006C     MOVQ    PASTEBOARD_ROW, -(SP)                        ; 3321
                                                      0C   BB 00070  PUSHR   #^M<R2,R3>                                   ; 3320
                                 0000V    CF            04   FB 00072  CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY               ;
                                                      04   00077     RET                                                  ;
                                    52                6E   D0 00078 7$: MOVL    PP, R2                                       ; 3326
                                 18 A2        0C  BC  B0   0007B     MOVW    @PASTEBOARD_ROW, 24(R2)                      ;
                                 1A A2        10  BC  B0   00080     MOVW    @PASTEBOARD_COL, 26(R2)                      ; 3327
                                                      53   DD 00085  PUSHL   PBCB                                         ; 3332
                                 0000V    CF            01   FB 00087  CALLS   #1, SMG$$CHECK_OCCLUSION                    ;
                                    13                50   E9 0008C  BLBC    STATUS, 8$                                   ;
                                                      52   DD 0008F  PUSHL   R2                                           ; 3340
                                 0000V    CF            01   FB 00091  CALLS   #1, SMG$$CALC_PASTE_TRANSF                  ;
                                    09                50   E9 00096  BLBC    STATUS, 8$                                   ;
```

```
                                        53  DD 00099          PUSHL   PBCB                                                    ; 3344
                        00000000G   00  01  FB 0009B          CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB                          ;
                                        04 000A2 8$:          RET                                                             ; 3346

; Routine Size:  163 bytes,    Routine Base:  _SMG$CODE + 10EA

; 3099          3347  1 !<BLF/PAGE>
```

```
;  3101          3348  1 %SBTTL 'SMG$PASTE_VIRTUAL_DISPLAY - Paste virtual display to pasteboard'
;  3102          3349  1 GLOBAL ROUTINE SMG$PASTE_VIRTUAL_DISPLAY (
;  3103          3350  1                                           DISPLAY_ID,
;  3104          3351  1                                           PASTEBOARD_ID,
;  3105          3352  1                                           PASTEBOARD_ROW,
;  3106          3353  1                                           PASTEBOARD_COL
;  3107          3354  1                                          ) =
;  3108          3355  1 !++
;  3109          3356  1 ! FUNCTIONAL DESCRIPTION:
;  3110          3357  1 !
;  3111          3358  1 !       The specified virtual display is "pasted" (oriented
;  3112          3359  1 !       with respect to) a pasteboard.  This makes the display visible.
;  3113          3360  1 !
;  3114          3361  1 ! CALLING SEQUENCE:
;  3115          3362  1 !
;  3116          3363  1 !       ret_status.wlc.v = SMG$PASTE_VIRTUAL_DISPLAY (
;  3117          3364  1 !                                                     DISPLAY_ID.rl.r,
;  3118          3365  1 !                                                     PASTEBOARD_ID.rl.r,
;  3119          3366  1 !                                                     PASTEBOARD_ROW.rl.r,
;  3120          3367  1 !                                                     PASTEBOARD_COL.rl.r)
;  3121          3368  1 !
;  3122          3369  1 ! FORMAL PARAMETERS:
;  3123          3370  1 !
;  3124          3371  1 !       DISPLAY_ID.rl.r         Id of virtual display to be pasted.
;  3125          3372  1 !
;  3126          3373  1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard on
;  3127          3374  1 !                               which the pasting is to take place.
;  3128          3375  1 !
;  3129          3376  1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
;  3130          3377  1 !                               row 1 of the specified virtual display.
;  3131          3378  1 !
;  3132          3379  1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
;  3133          3380  1 !                               column 1 of the specified virtual
;  3134          3381  1 !                               display.
;  3135          3382  1 !
;  3136          3383  1 ! IMPLICIT INPUTS:
;  3137          3384  1 !
;  3138          3385  1 !       None
;  3139          3386  1 !
;  3140          3387  1 ! IMPLICIT OUTPUTS:
;  3141          3388  1 !
;  3142          3389  1 !       None
;  3143          3390  1 !
;  3144          3391  1 ! COMPLETION STATUS:
;  3145          3392  1 !
;  3146          3393  1 !       SS$_NORMAL      Normal successful completion
;  3147          3394  1 !       SMG$_INVDIS_ID  Invalid virtual display id.
;  3148          3395  1 !       SMG$_INVPAS_ID  Invalid pasteboard id.
;  3149          3396  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  3150          3397  1 !       SMG$_ILLBATFNC  Display is batched.
;  3151          3398  1 !
;  3152          3399  1 ! SIDE EFFECTS:
;  3153          3400  1 !
;  3154          3401  1 !       NONE
;  3155          3402  1 !--
;  3156          3403  2     BEGIN
;  3157          3404  2     BUILTIN
```

```
;  3158          3405  2          AP,
;  3159          3406  2          CALLG;
;  3160          3407  2
;  3161          3408  2      LOCAL
;  3162          3409  2          STATUS,                          ! Status of subroutine calls
;  3163          3410  2
;  3164          3411  2          PP      : REF $PP_DECL,          ! Addr of the pasting packet
;  3165          3412  2                                           ! being created.
;  3166          3413  2          DCB     : REF $DCB_DECL,         ! Addr. of display control block
;  3167          3414  2          WCB     : REF $WCB_DECL,         ! Addr. of window control block
;  3168          3415  2          PBCB    : REF $PBCB_DECL;        ! Addr of pasteboard control
;  3169          3416  2                                           ! block
;  3170          3417  2
;  3171          3418  2      $SMG$VALIDATE_ARGCOUNT (4, 4);       ! Test for right no. of args
;  3172          3419  2
;  3173          3420  2  !+
;  3174          3421  2  ! Get addresses of associated virtual display control block and
;  3175          3422  2  ! pasteboard control block, validating both the display id and the
;  3176          3423  2  ! pasteboard id.
;  3177          3424  2  !-
;  3178          3425  2      $SMG$GET_DCB ( .DISPLAY_ID, DCB);    ! Get addr of DCB
;  3179          3426  2      $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);       ! Get addr of PBCB
;  3180          3427  2
;  3181          3428  2  !+
;  3182          3429  2  ! Give an error if the display is batched.
;  3183          3430  2  !-
;  3184          3431  2
;  3185          3432  2      IF .DCB[DCB_L_BATCH_LEVEL] NEQ 0
;  3186          3433  2      THEN
;  3187          3434  2          RETURN  SMG$_ILLBATFNC;
;  3188          3435  2
;  3189          3436  2  !+
;  3190          3437  2  ! Check to make sure we're don't already have a pasting from this
;  3191          3438  2  ! virtual display to this pasteboard.  If it is, we employ the
;  3192          3439  2  ! repaste logic to remove the current pasting before allowing this new
;  3193          3440  2  ! pasting.  This is necessary because we don't want ambiguous pastings.
;  3194          3441  2  ! Note:  The repaste logic ends up recalling the paste routine
;  3195          3442  2  ! recursively (after doing an unpaste) -- but that's ok since there
;  3196          3443  2  ! can be at most one such pasting.  The second time we are called this
;  3197          3444  2  ! test will fail.
;  3198          3445  2  !-
;  3199          3446  2      IF SMG$$LOCATE_PP( .DCB, .PBCB, PP)
;  3200          3447  2      THEN
;  3201          3448  2          RETURN (CALLG (.AP, SMG$REPASTE_VIRTUAL_DISPLAY));
;  3202          3449  2
;  3203          3450  2      RETURN SMG$$PASTE_VIRTUAL_DISPLAY(.DCB,.PBCB,
;  3204          3451  3                          .PASTEBOARD_ROW,.PASTEBOARD_COL);
;  3205          3452  2
;  3206          3453  1      END;                             ! Routine SMG$PASTE_VIRTUAL_DISPLAY



                                001C 00000          .ENTRY  SMG$PASTE_VIRTUAL_DISPLAY, Save R2,R3,R4    ; 3349
                    54 00000000'  EF   9E 00002     MOVAB   PBD_L_COUNT, R4                             ;
                    5E            04   C2 00009     SUBL2   #4, SP                                      ;
```

```
                                04              6C  91 0000C          CMPB    (AP), #4                                        ; 3418
                                                08  13 0000F          BEQL    1$
                                50 00000000G    8F  D0 00011          MOVL    #SMG$_WRONUMARG, R0
                                                    04 00018          RET
                                50        04    BC  D0 00019 1$:      MOVL    @DISPLAY_ID, R0                                 ; 3425
                          04    BC        38    A0  D1 0001D          CMPL    56(R0), @DISPLAY_ID
                                                06  12 00022          BNEQ    2$
                                11        44    A0  91 00024          CMPB    68(R0), #17
                                                08  13 00028          BEQL    3$
                                50 00000000G    8F  D0 0002A 2$:      MOVL    #SMG$_INVDIS_ID, R0
                                                    04 00031          RET
                                52        04    BC  D0 00032 3$:      MOVL    @DISPLAY_ID, DCB
                                50        08    BC  D0 00036          MOVL    @PASTEBOARD_ID, R0                              ; 3426
                                                0A  19 0003A          BLSS    4$
                                64              50  D1 0003C          CMPL    R0, PBD_L_COUNT
                                                05  14 0003F          BGTR    4$
                          08    44        A4    50  E0 00041          BBS     R0, PBD_V_PB_AVAIL, 5$
                                50 00000000G    8F  D0 00046 4$:      MOVL    #SMG$_INVPAS_ID, R0
                                                    04 0004D          RET
                                53        04 A440  D0 0004E 5$:      MOVL    PBD_A_PBCB[R0], PBCB
                                          1C    A2  D5 00053          TSTL    28(DCB)                                         ; 3432
                                                08  13 00056          BEQL    6$
                                50 00000000G    8F  D0 00058          MOVL    #SMG$_ILLBATFNC, R0                             ; 3434
                                                    04 0005F          RET
                                        400C    8F  BB 00060 6$:      PUSHR   #^M<R2,R3,SP>                                   ; 3446
                          0000V CF              03  FB 00064          CALLS   #3, SMG$$LOCATE_PP
                                06              50  E9 00069          BLBC    R0, 7$
                          0000V CF              6C  FA 0006C          CALLG   (AP), SMG$REPASTE_VIRTUAL_DISPLAY               ; 3448
                                                    04 00071          RET
                                7E        0C    AC  7D 00072 7$:      MOVQ    PASTEBOARD_ROW, -(SP)                           ; 3451
                                                0C  BB 00076          PUSHR   #^M<R2,R3>                                      ; 3450
                          0000V CF              04  FB 00078          CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY
                                                    04 0007D          RET                                                     ; 3453

; Routine Size:  126 bytes,    Routine Base:  _SMG$CODE + 118D

; 3207          3454  1 !<BLF/PAGE>
```

```
;  3209          3455  1 %SBTTL 'SMG$POP_VIRTUAL_DISPLAY - Pop off (delete) a sequence of virtual displays'
;  3210          3456  1 GLOBAL ROUTINE SMG$POP_VIRTUAL_DISPLAY (
;  3211          3457  1                                         DISPLAY_ID,
;  3212          3458  1                                         PASTEBOARD_ID
;  3213          3459  1                                         ) =
;  3214          3460  1 !++
;  3215          3461  1 ! FUNCTIONAL DESCRIPTION:
;  3216          3462  1 !
;  3217          3463  1 !       This procedure deletes all the virtual displays on the specified
;  3218          3464  1 !       pasteboard, starting with the display specified, up through all
;  3219          3465  1 !       higher-pasted display.  Each of these displays is unpasted in
;  3220          3466  1 !       in the course of doing the deletion.
;  3221          3467  1 !
;  3222          3468  1 ! CALLING SEQUENCE:
;  3223          3469  1 !
;  3224          3470  1 !       ret_status.wlc.v = SMG$POP_VIRTUAL_DISPLAY ( DISPLAY_ID.rl.r,
;  3225          3471  1 !                                                    PASTEBOARD_ID.rl.r)
;  3226          3472  1 !
;  3227          3473  1 ! FORMAL PARAMETERS:
;  3228          3474  1 !
;  3229          3475  1 !       DISPLAY_ID.rl.r         Address of the display id of the lowest
;  3230          3476  1 !                               pasted virtual display to be deleted.
;  3231          3477  1 !                               All higher-pasted displays are deleted
;  3232          3478  1 !                               as well.
;  3233          3479  1 !
;  3234          3480  1 !       PASTEBOARD_ID.rl.r      Address of the pasteboard id involved.
;  3235          3481  1 !
;  3236          3482  1 ! IMPLICIT INPUTS:
;  3237          3483  1 !
;  3238          3484  1 !       NONE
;  3239          3485  1 !
;  3240          3486  1 ! IMPLICIT OUTPUTS:
;  3241          3487  1 !
;  3242          3488  1 !       NONE
;  3243          3489  1 !
;  3244          3490  1 ! COMPLETION STATUS:
;  3245          3491  1 !
;  3246          3492  1 !       SS$_NORMAL      Normal successful completion
;  3247          3493  1 !       SMG$_INVDIS_ID  Invalid display id
;  3248          3494  1 !       SMG$_INVPAS_ID  Invalid pasteboard id
;  3249          3495  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  3250          3496  1 !
;  3251          3497  1 ! SIDE EFFECTS:
;  3252          3498  1 !
;  3253          3499  1 !       NONE
;  3254          3500  1 !--
;  3255          3501  1
;  3256          3502  2     BEGIN
;  3257          3503  2     LOCAL
;  3258          3504  2         STATUS,                     ! Status of subr. calls
;  3259          3505  2
;  3260          3506  2         RET_STATUS,                 ! Accumulated status during
;  3261          3507  2                                     ! loop
;  3262          3508  2         PBCB : REF $PBCB_DECL,      ! Address of a pasteboard
;  3263          3509  2                                     ! control block
;  3264          3510  2
;  3265          3511  2         DCB  : REF $DCB_DECL,       ! Address of a virtual display
```

```
;  3266     3512  2                                        ! control block we started with
;  3267     3513  2
;  3268     3514  2          PP : REF $PP_DECL;             ! Addr of 2 longwords that form
;  3269     3515  2                                        ! queue header in PP currently
;  3270     3516  2                                        ! under inspection.
;  3271     3517  2      !+
;  3272     3518  2      ! Check for right number of arguments.
;  3273     3519  2      !
;  3274     3520  2          $SMG$VALIDATE_ARGCOUNT ( 2, 2);
;  3275     3521  2
;  3276     3522  2      !+
;  3277     3523  2      ! Get addresses of virtual display control block and pasteboard control
;  3278     3524  2      ! block and validate them.
;  3279     3525  2      !-
;  3280     3526  2          $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB );
;  3281     3527  2          $SMG$GET_DCB  ( .DISPLAY_ID,    DCB);
;  3282     3528  2
;  3283     3529  2      !+
;  3284     3530  2      ! Locate the pasting packet that reflects this pasting (if one exists)
;  3285     3531  2      ! .PP is the base address of the pasting packet.
;  3286     3532  2      !-
;  3287     3533  3          IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
;  3288     3534  2          THEN
;  3289     3535  2              RETURN (.STATUS);
;  3290     3536  2
;  3291     3537  2      !+
;  3292     3538  2      ! Change packet address to address of queue header.
;  3293     3539  2      !-
;  3294     3540  2          PP = .PP + PP_PBCB_QUEUE_OFFSET; ! Start with specified packet
;  3295     3541  2
;  3296     3542  2          RET_STATUS = SS$_NORMAL;     ! Assume success to follow
;  3297     3543  2
;  3298     3544  2      !+
;  3299     3545  2      ! Batch the sequence of updates we are about to do.
;  3300     3546  2      !-
;  3301     3547  3          IF NOT ( STATUS = SMG$$BEGIN_PASTEBOARD_UPDATE_R1 (.PBCB))
;  3302     3548  2          THEN
;  3303     3549  2              RETURN (.STATUS);
;  3304     3550  2
;  3305     3551  2      !+
;  3306     3552  2      ! Loop for all pasting packets starting with this one to the last-pasted
;  3307     3553  2      ! one...
;  3308     3554  2      !-
;  3309     3555  2          WHILE .PP NEQ PBCB [PBCB_A_PP_NEXT]
;  3310     3556  2          DO
;  3311     3557  3              BEGIN   ! For all displays that need to be deleted
;  3312     3558  3              LOCAL
;  3313     3559  3                  STATUS,                  ! Status of delete calls
;  3314     3560  3
;  3315     3561  3                  PP_BASE : REF $PP_DECL,  ! Base address of the PP
;  3316     3562  3
;  3317     3563  3                  DCB : REF $DCB_DECL;     ! Current virtual display that
;  3318     3564  3                                           ! needs to be deleted.
;  3319     3565  3              !+
;  3320     3566  3              ! Calc. the base address of this pasting packet since the queue
;  3321     3567  3              ! headers for this part of the chain are not at relative 0 in
;  3322     3568  3              ! the pasting packet.
```

```
;  3323     3569  3          !-
;  3324     3570  3          PP_BASE = .PP - PP_PBCB_QUEUE_OFFSET;
;  3325     3571  3
;  3326     3572  3          !+
;  3327     3573  3          ! Find DCB that is in this pairing.
;  3328     3574  3          !-
;  3329     3575  3          DCB = .PP_BASE [PP_A_DCB_ADDR];
;  3330     3576  3
;  3331     3577  3          !+
;  3332     3578  3          ! Delete this virtual display, causing it to be unpasted from
;  3333     3579  3          ! all pasteboards to which it is currently pasted.
;  3334     3580  3          !-
;  3335     3581  4          IF NOT ( STATUS = SMG$DELETE_VIRTUAL_DISPLAY ( DCB [DCB_L_DID]))
;  3336     3582  3          THEN
;  3337     3583  3              !+
;  3338     3584  3              ! If no error yet, save this one.
;  3339     3585  3              !-
;  3340     3586  4              BEGIN
;  3341     3587  4              IF .RET_STATUS THEN RET_STATUS = .STATUS;
;  3342     3588  3              END;
;  3343     3589  3
;  3344     3590  3          !+
;  3345     3591  3          ! Walk this chain backwards, from the packet we started with
;  3346     3592  3          ! back to the head of the chain -- since the most recently
;  3347     3593  3          ! pasted displays are at the head of the chain.
;  3348     3594  3          !-
;  3349     3595  3          PP = .PP_BASE [PP_A_PREV_PBCB];
;  3350     3596  2          END;    ! For all displays that need to be deleted
;  3351     3597  2
;  3352     3598  2
;  3353     3599  3      IF NOT (STATUS = SMG$$END_PASTEBOARD_UPDATE_R2 ( .PBCB ))
;  3354     3600  2      THEN
;  3355     3601  2          RETURN (.STATUS);
;  3356     3602  2
;  3357     3603  2      RETURN (.RET_STATUS);
;  3358     3604  2
;  3359     3605  1      END;                    ! End of routine SMG$POP_VIRTUAL_DISPLAY


                                                007C 00000          .ENTRY  SMG$POP_VIRTUAL_DISPLAY, Save R2,R3,R4,R5,- ; 3456
                                                                            R6
                                    56 00000000'  EF   9E 00002     MOVAB   PBD_L_COUNT, R6                              ;
                                    5E            04   C2 00009     SUBL2   #4, SP                                       ;
                                    02            6C   91 0000C     CMPB    (AP), #2                                     ; 3520
                                                  08   13 0000F     BEQL    1$                                           ;
                                    50 00000000G  8F   D0 00011     MOVL    #SMG$_WRONUMARG, R0                          ;
                                                  04   00018     RET                                                  ;
                                    50        08  BC   D0 00019 1$: MOVL    @PASTEBOARD_ID, R0                           ; 3526
                                                  0A   19 0001D     BLSS    2$                                           ;
                                    66            50   D1 0001F     CMPL    R0, PBD_L_COUNT                              ;
                                                  05   14 00022     BGTR    2$                                           ;
                08        44      A6            50   E0 00024     BBS     R0, PBD_V_PB_AVAIL, 3$                       ;
                                    50 00000000G  8F   D0 00029 2$: MOVL    #SMG$_INVPAS_ID, R0                          ;
                                                  04   00030     RET                                                  ;
```

```
                                55        04 A640   DO 00031 3$:    MOVL    PBD_A_PBCB[R0], PBCB
                                50        04   BC   DO 00036        MOVL    @DISPLAY_ID, R0                         ;  3527
                             04 BC        38   A0   D1 0003A        CMPL    56(R0), @DISPLAY_ID
                                                06   12 0003F        BNEQ    4$
                                11        44   A0   91 00041        CMPB    68(R0), #17
                                                08   13 00045        BEQL    5$
                                50 00000000G   8F   DO 00047 4$:    MOVL    #SMG$_INVDIS_ID, R0
                                                04   04 0004E        RET
                                50        04   BC   DO 0004F 5$:    MOVL    @DISPLAY_ID, DCB
                                        4021   8F   BB 00053        PUSHR   #^M<R0,R5,SP>                           ;  3533
                          0000V CF              03   FB 00057        CALLS   #3, SMG$$LOCATE_PP
                                53              50   DO 0005C        MOVL    R0, STATUS
                                48              53   E9 0005F        BLBC    STATUS, 9$
                                6E              08   C0 00062        ADDL2   #8, PP                              ;  3540
                                54              01   DO 00065        MOVL    #1, RET_STATUS                     ;  3542
                                50              55   DO 00068        MOVL    PBCB, R0                           ;  3547
                                   00000000G    00   16 0006B        JSB     SMG$$BEGIN_PASTEBOARD_UPDATE_R1
                                53              50   DO 00071        MOVL    R0, STATUS
                                33              53   E9 00074        BLBC    STATUS, 9$
                                55              6E   D1 00077 6$:    CMPL    PP, PBCB                           ;  3555
                                                1F   13 0007A        BEQL    8$
                             52                 6E   C3 0007C        SUBL3   #8, PP, PP_BASE                    ;  3570
                                50        10   A2   DO 00080        MOVL    16(PP_BASE), DCB                   ;  3575
                                          38   A0   9F 00084        PUSHAB  56(DCB)                            ;  3581
                          FB98 CF              01   FB 00087        CALLS   #1, SMG$DELETE_VIRTUAL_DISPLAY
                                06              50   E8 0008C        BLBS    STATUS, 7$
                                03              54   E9 0008F        BLBC    RET_STATUS, 7$                     ;  3587
                                54              50   DO 00092        MOVL    STATUS, RET_STATUS
                                6E        0C    A2   DO 00095 7$:    MOVL    12(PP_BASE), PP                    ;  3595
                                                DC   11 00099        BRB     6$                                 ;  3555
                                50              55   DO 0009B 8$:    MOVL    PBCB, R0                           ;  3599
                                   00000000G    00   16 0009E        JSB     SMG$$END_PASTEBOARD_UPDATE_R2
                                53              50   DO 000A4        MOVL    R0, STATUS
                                04              53   E8 000A7        BLBS    STATUS, 10$
                                50              53   DO 000AA 9$:    MOVL    STATUS, R0                         ;  3601
                                                04   04 000AD        RET
                                50              54   DO 000AE 10$:   MOVL    RET_STATUS, R0                     ;  3603
                                                04   04 000B1        RET                                        ;  3605

; Routine Size:  178 bytes,    Routine Base:  _SMG$CODE + 120B

; 3360          3606  1 :<BLF/PAGE>
```

```
;  3362     3607  1 %SBTTL 'SMG$REPASTE_VIRTUAL_DISPLAY - Repaste virtual display to pasteboard'
;  3363     3608  1 GLOBAL ROUTINE SMG$REPASTE_VIRTUAL_DISPLAY (
;  3364     3609  1                                             DISPLAY_ID,
;  3365     3610  1                                             PASTEBOARD_ID,
;  3366     3611  1                                             PASTEBOARD_ROW,
;  3367     3612  1                                             PASTEBOARD_COL
;  3368     3613  1                                            ) =
;  3369     3614  1 !++
;  3370     3615  1 ! FUNCTIONAL DESCRIPTION:
;  3371     3616  1 !
;  3372     3617  1 !       The specified virtual display is "unpasted" from the specified
;  3373     3618  1 !       pasteboard.  It is then "repasted" in the new position to the
;  3374     3619  1 !       same pasteboard.  The unpasting and repasting operation is done
;  3375     3620  1 !       under cover of a SMG$$BEGIN_PASTEBOARD_UPDATE_R1 and
;  3376     3621  1 !       SMG$$END_PASTEBOARD_UPDATE_R2 pair so that there is no effect on the
;  3377     3622  1 !       screen while it is going on.  Only the completed results of the
;  3378     3623  1 !       operation become visible.
;  3379     3624  1 !
;  3380     3625  1 ! CALLING SEQUENCE:
;  3381     3626  1 !
;  3382     3627  1 !       ret_status.wlc.v = SMG$REPASTE_VIRTUAL_DISPLAY (
;  3383     3628  1 !                                                       DISPLAY_ID.rl.r,
;  3384     3629  1 !                                                       PASTEBOARD_ID.rl.r,
;  3385     3630  1 !                                                       PASTEBOARD_ROW.rl.r,
;  3386     3631  1 !                                                       PASTEBOARD_COL.rl.r)
;  3387     3632  1 !
;  3388     3633  1 ! FORMAL PARAMETERS:
;  3389     3634  1 !
;  3390     3635  1 !       DISPLAY_ID.rl.r         Id of virtual display to be repasted.
;  3391     3636  1 !
;  3392     3637  1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard on
;  3393     3638  1 !                               which the unpasting/pasting is to take
;  3394     3639  1 !                               place.
;  3395     3640  1 !
;  3396     3641  1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
;  3397     3642  1 !                               row 1 of the specified virtual display
;  3398     3643  1 !                               after repasting.
;  3399     3644  1 !
;  3400     3645  1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
;  3401     3646  1 !                               column 1 of the specified virtual
;  3402     3647  1 !                               display after repasting.
;  3403     3648  1 !
;  3404     3649  1 ! IMPLICIT INPUTS:
;  3405     3650  1 !
;  3406     3651  1 !       None
;  3407     3652  1 !
;  3408     3653  1 ! IMPLICIT OUTPUTS:
;  3409     3654  1 !
;  3410     3655  1 !       None
;  3411     3656  1 !
;  3412     3657  1 ! COMPLETION STATUS:
;  3413     3658  1 !
;  3414     3659  1 !       SS$_NORMAL      Normal successful completion
;  3415     3660  1 !       SMG$_INVDIS_ID  Invalid virtual display id.
;  3416     3661  1 !       SMG$_INVPAS_ID  Invalid pasteboard id.
;  3417     3662  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  3418     3663  1 !
```

```
;  3419     3664  1 ! SIDE EFFECTS:
;  3420     3665  1 !
;  3421     3666  1 !       NONE
;  3422     3667  1 !--
;  3423     3668  2     BEGIN
;  3424     3669  2
;  3425     3670  2     LOCAL
;  3426     3671  2         DCB     : REF $DCB_DECL,
;  3427     3672  2         PBCB    : REF $PBCB_DECL,
;  3428     3673  2         STATUS ;        ! Status of subroutine calls
;  3429     3674  2
;  3430     3675  2     $SMG$VALIDATE_ARGCOUNT (4, 4);      ! Test for right no. of args
;  3431     3676  2
;  3432     3677  2     $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  3433     3678  2     $SMG$GET_DCB(.DISPLAY_ID,DCB);
;  3434     3679  2
;  3435     3680  2 !+
;  3436     3681  2 ! Set up an extra level of output inhibiting so that our UNPASTE
;  3437     3682  2 ! operation won't find its way to the screen until we're done.
;  3438     3683  2 !-
;  3439     3684  3     IF NOT (STATUS = SMG$$BEGIN_PASTEBOARD_UPDATE_R1 (.PBCB))
;  3440     3685  2     THEN
;  3441     3686  2         RETURN (.STATUS);
;  3442     3687  2
;  3443     3688  2 !+
;  3444     3689  2 ! Unpaste it from where it is.
;  3445     3690  2 !-
;  3446     3691  3     IF NOT (STATUS = SMG$$UNPASTE_VIRTUAL_DISPLAY (.DCB, .PBCB))
;  3447     3692  2     THEN
;  3448     3693  3         BEGIN
;  3449     3694  3         SMG$$END_PASTEBOARD_UPDATE_R2 (.PBCB); ! Reduce buffering level
;  3450     3695  3         RETURN (.STATUS);                      ! Return error
;  3451     3696  2         END;
;  3452     3697  2
;  3453     3698  2 !+
;  3454     3699  2 ! Now repaste to the same pasteboard in a new position.
;  3455     3700  2 !-
;  3456     3701  2     STATUS = SMG$$PASTE_VIRTUAL_DISPLAY(.DCB,.PBCB,
;  3457     3702  2                         .PASTEBOARD_ROW,.PASTEBOARD_COL);
;  3458     3703  2
;  3459     3704  2 !+
;  3460     3705  2 ! Undo one buffering level so that we are back where we started.
;  3461     3706  2 !-
;  3462     3707  2     SMG$$END_PASTEBOARD_UPDATE_R2 (.PBCB);
;  3463     3708  2
;  3464     3709  2 !+
;  3465     3710  2 ! If last PASTE operation yielded an error, return that status, else
;  3466     3711  2 ! return SS$_NORMAL;
;  3467     3712  2 !-
;  3468     3713  2     IF NOT .STATUS THEN RETURN .STATUS;
;  3469     3714  2
;  3470     3715  2     RETURN (SS$_NORMAL);
;  3471     3716  2
;  3472     3717  1     END;                            ! Routine SMG$REPASTE_VIRTUAL_DISPLAY
```

```
                                                00FC 00000          .ENTRY  SMG$REPASTE_VIRTUAL_DISPLAY, Save R2,R3,R4,-;  3608
                                                                            R5,R6,R7
                                    57 00000000G   00   9E 00002           MOVAB   SMG$$END_PASTEBOARD_UPDATE_R2, R7
                                    56 00000000'   EF   9E 00009           MOVAB   PBD_L_COUNT, R6
                                    04             6C   91 00010           CMPB    (AP), #4                                3675
                                                   08   13 00013           BEQL    1$
                                    50 00000000G   8F   D0 00015           MOVL    #SMG$_WRONUMARG, R0
                                                        04 0001C           RET
                                    50       08    BC   D0 0001D 1$:       MOVL    @PASTEBOARD_ID, R0                      3677
                                                   0A   19 00021           BLSS    2$
                                    66             50   D1 00023           CMPL    R0, PBD_L_COUNT
                                                   05   14 00026           BGTR    2$
                              08    44       A6    50   E0 00028           BBS     R0, PBD_V_PB_AVAIL, 3$
                                    50 00000000G   8F   D0 0002D 2$:       MOVL    #SMG$_INVPAS_ID, R0
                                                        04 00034           RET
                                    54       04 A640   D0 00035 3$:       MOVL    PBD_A_PBCB[R0], PBCB
                                    50       04   BC   D0 0003A           MOVL    @DISPLAY_ID, R0                         3678
                                    04       BC   38   A0 D1 0003E         CMPL    56(R0), @DISPLAY_ID
                                                   06   12 00043           BNEQ    4$
                                    11       44   A0   91 00045           CMPB    68(R0), #17
                                                   08   13 00049           BEQL    5$
                                    50 00000000G   8F   D0 0004B 4$:       MOVL    #SMG$_INVDIS_ID, R0
                                                        04 00052           RET
                                    55       04   BC   D0 00053 5$:       MOVL    @DISPLAY_ID, DCB
                                    50             54   D0 00057           MOVL    PBCB, R0                                3684
                                       00000000G   00   16 0005A           JSB     SMG$$BEGIN_PASTEBOARD_UPDATE_R1
                                    53             50   D0 00060           MOVL    R0, STATUS
                                    2E             53   E9 00063           BLBC    STATUS, 7$
                                                   54   DD 00066           PUSHL   PBCB                                    3691
                                                   55   DD 00068           PUSHL   DCB
                              0000V CF             02   FB 0006A           CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY
                                    53             50   D0 0006F           MOVL    R0, STATUS
                                    07             53   E8 00072           BLBS    STATUS, 6$
                                    50             54   D0 00075           MOVL    PBCB, R0                                3694
                                                   67   16 00078           JSB     SMG$$END_PASTEBOARD_UPDATE_R2
                                                   18   11 0007A           BRB     7$                                      3695
                                    7E       0C    AC   7D 0007C 6$:       MOVQ    PASTEBOARD_ROW, -(SP)                   3702
                                                   54   DD 00080           PUSHL   PBCB                                    3701
                                                   55   DD 00082           PUSHL   DCB
                              0000V CF             04   FB 00084           CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY
                                    53             50   D0 00089           MOVL    R0, STATUS
                                    50             54   D0 0008C           MOVL    PBCB, R0                                3707
                                                   67   16 0008F           JSB     SMG$$END_PASTEBOARD_UPDATE_R2
                                    04             53   E8 00091           BLBS    STATUS, 8$                              3713
                                    50             53   D0 00094 7$:       MOVL    STATUS, R0
                                                        04 00097           RET
                                    50             01   D0 00098 8$:       MOVL    #1, R0                                  3715
                                                        04 0009B           RET                                             3717

; Routine Size:  156 bytes,    Routine Base:  _SMG$CODE + 12BD

; 3473          3718  1 !<BLF/PAGE>
```

```
;  3475          3719  1 %SBTTL 'SMG$RESTORE_PHYSICAL_SCREEN - Restore physical screen'
;  3476          3720  1 GLOBAL ROUTINE SMG$RESTORE_PHYSICAL_SCREEN (
;  3477          3721  1                                             PASTEBOARD_ID,
;  3478          3722  1                                             DISPLAY_ID
;  3479          3723  1                                           ) =
;  3480          3724  1 !++
;  3481          3725  1 ! FUNCTIONAL DESCRIPTION:
;  3482          3726  1 !
;  3483          3727  1 !       This routine reverses the effect of SMG$SAVE_PHYSICAL_SCREEN,
;  3484          3728  1 !       thereby putting the physical screen back to the point it was
;  3485          3729  1 !       at just prior to the call to SMG$SAVE_PHYSICAL_SCREEN.
;  3486          3730  1 !       The display id returned by SMG$SAVE_PHYSICAL_SCREEN must be
;  3487          3731  1 !       passed to this routine to allow the restoration to happen.
;  3488          3732  1 !
;  3489          3733  1 ! CALLING SEQUENCE:
;  3490          3734  1 !
;  3491          3735  1 !       ret_status.wlc.v = SMG$RESTORE_PHYSICAL_SCREEN (
;  3492          3736  1 !                                             PASTEBOARD_ID.rl.r,
;  3493          3737  1 !                                             DISPLAY_ID.rl.r)
;  3494          3738  1 !
;  3495          3739  1 ! FORMAL PARAMETERS:
;  3496          3740  1 !
;  3497          3741  1 !       PASTEBOARD_ID.rl.r      Address of a pasteboard id which is to
;  3498          3742  1 !                               be "restored".
;  3499          3743  1 !
;  3500          3744  1 !       DISPLAY_ID.rl.r         Returned display id invented to
;  3501          3745  1 !                               perform requested function.
;  3502          3746  1 !                               This must be the display id returned
;  3503          3747  1 !                               by SMG$SAVE_PHYSCIAL_SCREEN.
;  3504          3748  1 !
;  3505          3749  1 ! IMPLICIT INPUTS:
;  3506          3750  1 !
;  3507          3751  1 !       NONE
;  3508          3752  1 !
;  3509          3753  1 ! IMPLICIT OUTPUTS:
;  3510          3754  1 !
;  3511          3755  1 !       NONE
;  3512          3756  1 !
;  3513          3757  1 ! COMPLETION STATUS:
;  3514          3758  1 !
;  3515          3759  1 !    SS$_NORMAL         Normal successful completion
;  3516          3760  1 !    SMG$_INVDIS_ID     Invalid Display Id.
;  3517          3761  1 !    SMG$_INVPAS_ID     Invalid Pasteboard Id.
;  3518          3762  1 !
;  3519          3763  1 !
;  3520          3764  1 ! SIDE EFFECTS:
;  3521          3765  1 !       NONE
;  3522          3766  1 !
;  3523          3767  1 !--
;  3524          3768  1
;  3525          3769  2     BEGIN
;  3526          3770  2     LOCAL
;  3527          3771  2         DCB  : REF $DCB_DECL,   ! Address of virtual display control
;  3528          3772  2                                ! block involved.
;  3529          3773  2
;  3530          3774  2         PBCB : REF $PBCB_DECL,  ! Address of pasteboard control block
;  3531          3775  2
```

```
;  3532          3776  2          PP   : REF $FP_DECL,     ! Address of the pasting packet that
;  3533          3777  2                                   ! joins the virtual display to the
;  3534          3778  2                                   ! pasteboard.
;  3535          3779  2
;  3536          3780  2          STATUS;                  ! Status of subr. calls
;  3537          3781  2
;  3538          3782  2      !+
;  3539          3783  2      ! Validate number of arguments.
;  3540          3784  2      !-
;  3541          3785  2          $SMG$VALIDATE_ARGCOUNT( 2,2);
;  3542          3786  2
;  3543          3787  2      !+
;  3544          3788  2      ! Map pasteboard id into a PBCB address, and display id into a DCB addr.
;  3545          3789  2      !-
;  3546          3790  2          $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);
;  3547          3791  2          $SMG$GET_DCB  ( .DISPLAY_ID,    DCB);
;  3548          3792  2
;  3549          3793  2      !+
;  3550          3794  2      ! Locate the pasting packet that joins this virtual display with this
;  3551          3795  2      ! pasteboard.
;  3552          3796  2      !-
;  3553          3797  3          IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
;  3554          3798  2          THEN
;  3555          3799  2              RETURN (.STATUS);
;  3556          3800  2
;  3557          3801  2      !+
;001 PL1097      3802  2      ! Set the hardware scrolling region to be the whole screen.  We have no
;002 PL1097      3803  2      ! idea what state it's currently in.
;003 PL1097      3804  2      !-
;004 PL1097      3805  3          IF NOT (STATUS = SMG$$FORCE_SCROLL_REG (.PBCB, 1, .PBCB [PBCB_B_ROWS]))
;005 PL1097      3806  2          THEN
;006 PL1097      3807  3              RETURN (.STATUS);
;007 PL1097      3808  2
;  3565-7        3809  2      !+
;  3566          3810  2      ! Determine best way to clear affected area.  If the whole screen is
;  3567          3811  2      ! involved we erase the whole screen in one operation.  If only part
;  3568          3812  2      ! of the screen is involved, we have to do it a line at at time.
;  3569          3813  2      !-
;  3570          3814  2          IF .PP [PP_W_FIRST_WCB_ROW] LEQ 1            AND
;  3571          3815  2             .PP [PP_W_LAST_WCB_ROW] GEQ .PBCB [PBCB_B_ROWS]
;  3572          3816  2          THEN
;  3573          3817  3              BEGIN   ! Full screen involved
;  3574          3818  3              !+
;  3575          3819  3              ! Clear the whole physical screen to get rid of what the non-SMG
;  3576          3820  3              ! user may have put there.
;  3577          3821  3              !-
;  3578          3822  4              IF NOT (STATUS = SMG$$ERASE_PASTEBOARD (.PBCB))
;  3579          3823  3              THEN
;  3580          3824  3                  RETURN (.STATUS);
;  3581          3825  3
;  3582          3826  3              END     ! Full screen involved
;  3583          3827  3
;  3584          3828  2          ELSE
;  3585          3829  2
;  3586          3830  3              BEGIN   ! Only part of screen involved
;  3587          3831  3              !+
;  3588          3832  3              ! Clear only the part of the screen involved.  We'll have to do
```

```
;  3589          3833  3          ! it line by line.
;  3590          3834  3          ! The code to do that should really reside in module SMGMINUPD
;  3591          3835  3          ! for modularity.  However, it is here for now.
;  3592          3836  3          !-
;  3593          3837  3          LOCAL
;  3594          3838  3              WCB : REF $WCB_DECL;        ! Addr of window control block
;  3595          3839  3                                          ! involved.
;  3596          3840  3
;  3597          3841  3          WCB = .PBCB [PBCB_A_WCB];
;  3598          3842  3
;  3599          3843  3          !+
;  3600          3844  3          ! For each line involved, set cursor to column 1 of that line
;  3601          3845  3          ! and emit erase sequence.  Setting the cursor to column 1 of
;  3602          3846  3          ! the line is necessay for non-VT100 terminals.
;  3603          3847  3          !-
;  3604          3848  3
;  3605          3849  3          INCR I FROM .PP [PP_W_FIRST_WCB_ROW] TO .PP [PP_W_LAST_WCB_ROW]
;  3606          3850  3          DO
;  3607          3851  4              BEGIN       ! Row by row
;  3608          3852  4              !+
;  3609          3853  4              ! Set cursor to column 1 of row .I.
;  3610          3854  4              !-
;  3611          3855  4              SMG$$FIND_MIN_CURSOR_POS (
;  3612          3856  4                          .PBCB,
;  3613          3857  4                          .WCB [WCB_W_OLD_CUR_ROW],       ! Current row
;  3614          3858  4                          .WCB [WCB_W_OLD_CUR_COL],       ! Current col
;  3615          3859  4                          .I,                             ! Desired row
;  3616          3860  4                          1);                             ! Desired col
;  3617          3861  4
;  3618          3862  4              !+
;  3619          3863  4              ! Get escape sequence needed to erase a line.
;  3620          3864  4              ! (Can't move this outside the loop since data is left
;  3621          3865  4              ! in memory that FIND_MIN_CURSOR_POS might touch.
;  3622          3866  4              !-
;  3623          3867  4
;  3624          3868  4              $SMG$GET_TERM_DATA(ERASE_WHOLE_LINE);
;  3625          3869  4
;  3626          3870  4              !+
;  3627          3871  4              ! Erase the Ith line.
;  3628          3872  4              !-
;  3629          3873  4
;  3630          3874  5              IF NOT (STATUS = SMG$$OUTPUT ( .PBCB,
;  3631          3875  5                                             .PBCB[PBCB_L_CAP_LENGTH],
;  3632          3876  5                                             .PBCB[PBCB_A_CAP_BUFFER]))
;  3633          3877  4              THEN
;  3634          3878  4                  RETURN (.STATUS);
;  3635          3879  4
;  3636          3880  3              END;        ! Row by row
;  3637          3881  2          END;    ! Only part of screen involved
;  3638          3882  2
;  3639          3883  2          !+
;  3640          3884  2          ! Pop off the virtual display that SMG$SAVE_PHYSICAL_SCREEN placed on
;  3641          3885  2          ! top to cover everything up.
;  3642          3886  2          !-
;  3643          3887  3          IF NOT (STATUS = SMG$POP_VIRTUAL_DISPLAY ( .DISPLAY_ID,
;  3644          3888  3                                                     .PASTEBOARD_ID))
;  3645          3889  2          THEN
```

```
;  3646     3890  2          RETURN (.STATUS);
;  3647     3891  2
;  3648     3892  2      RETURN (SS$_NORMAL);
;  3649     3893  2
;  3650     3894  1      END;                    ! End of routine SMG$RESTORE_PHYSICAL_SCREEN


                                    03FC 00000          .ENTRY  SMG$RESTORE_PHYSICAL_SCREEN, Save R2,R3,R4,-;  3720
                                                                R5,R6,R7,R8,R9
                          59 00000000'  EF   9E 0C002           MOVAB   PBD_L_COUNT, R9
                          5E            14   C2 00009           SUBL2   #20, SP
                          02            6C   91 0000C           CMPB    (AP), #2                                3703
                                        08   13 0000F           BEQL    1$
                          50 00000000G  8F   D0 00011           MOVL    #SMG$_WRONUMARG, R0
                                        04   00018             RET
                          50      04    BC   D0 00019 1$:       MOVL    @PASTEBOARD_ID, R0                      3790
                                        0A   19 0001D           BLSS    2$
                          69            50   D1 0001F           CMPL    R0, PBD_L_COUNT
                                        05   14 00022           BGTR    2$
                    08    44    A9      50   E0 00024           BBS     R0, PBD_V_PB_AVAIL, 3$
                          50 00000000G  8F   D0 00029 2$:       MOVL    #SMG$_INVPAS_ID, R0
                                        04   00030             RET
                          53      04 A940   D0 00031 3$:       MOVL    PBD_A_PBCB[R0], PBCB
                          50      08   BC   D0 00036           MOVL    @DISPLAY_ID, R0                         3791
                       08 BC      38   A0   D1 0003A           CMPL    56(R0), @DISPLAY_ID
                                       06   12 0003F           BNEQ    4$
                          11      44   A0   91 00041           CMPB    68(R0), #17
                                       08   13 00045           BEQL    5$
                          50 00000000G  8F   D0 00047 4$:       MOVL    #SMG$_INVDIS_ID, R0
                                        04   0004E             RET
                          50      08    BC   D0 0004F 5$:       MOVL    @DISPLAY_ID, DCB
                                  04    AE   9F 00053           PUSHAB  PP                                      3797
                                        09   BB 00056           PUSHR   #^M<R0,R3>
                    0000V CF            03   FB 00058           CALLS   #3, SMG$$LOCATE_PP
                          55            50   D0 0005D           MOVL    R0, STATUS
                          38            55   E9 00060           BLBC    STATUS, 6$
                          7E      5F    A3   9A 00062           MOVZBL  95(PBCB), -(SP)                         3805
                                        01   DD 00067           PUSHL   #1
                                        53   DD 00069           PUSHL   PBCB
                00000000G 00            03   FB 0006B           CALLS   #3, SMG$$FORCE_SCROLL_REG
                          55            50   D0 00072           MOVL    R0, STATUS
                          23            55   E9 00075           BLBC    STATUS, 6$
                          52      04    AE   D0 00078           MOVL    PP, R2                                  3814
                          01      2F    A2   B1 0007C           CMPW    47(R2), #1
                                        1C   1A 00080           BGTRU   7$
                          50      5F    A3   9A 00082           MOVZBL  95(PBCB), R0                            3815
                       31 A2            50   B1 00086           CMPW    R0, 49(R2)
                                        12   1A 0008A           BGTRU   7$
                                        53   DD 0008C           PUSHL   PBCB                                    3822
                00000000G 00            01   FB 0008E           CALLS   #1, SMG$$ERASE_PASTEBOARD
                          55            50   D0 00095           MOVL    R0, STATUS
                          78            55   E8 00098           BLBS    STATUS, 12$
                                  0086  31   00099 6$:          BRW     13$                                     3824
                          54      08    A3   D0 0009E 7$:       MOVL    8(PBCB), WCB                            3841
```

```
                                58      31   A2  3C 000A2          MOVZWL  49(R2), R8                      ; 3849
                                57    00FC   C3  9E 000A6          MOVAB   252(PBCB), R7                   ; 3868
                                56    0108   C3  9E 000AB          MOVAB   264(PBCB), R6
                                52      2F   A2  3C 000B0          MOVZWL  47(R2), I                       ; 3876
                                             52  D7 000B4          DECL    I
                                             57  11 000B6          BRB     11$
                                             01  DD 000B8 8$:      PUSHL   #1                              ; 3855
                                             52  DD 000BA          PUSHL   I                               ; 3859
                                7E      26   A4  32 000BC          CVTWL   38(WCB), -(SP)                  ; 3858
                                7E      24   A4  32 000C0          CVTWL   36(WCB), -(SP)                  ; 3857
                                             53  DD 000C4          PUSHL   PBCB                            ; 3856
                  00000000G     00           05  FB 000C6          CALLS   #5, SMG$$FIND_MIN_CURSOR_POS
                                             67  D5 000CD          TSTL    (R7)                            ; 3868
                                             04  12 000CF          BNEQ    9$
                                             66  D4 000D1          CLRL    (R6)
                                             25  11 000D3          BRB     10$
                                      08     AE  D4 000D5 9$:      CLRL    INPUT_ARGS
                                      08     AE  9F 000D8          PUSHAB  INPUT_ARGS
                                    0104     C3  DD 000DB          PUSHL   260(PBCB)
                                             56  DD 000DF          PUSHL   R6
                                    0100     C3  9F 000E1          PUSHAB  256(PBCB)
                              10    AE   01DB   8F  3C 000E5       MOVZWL  #475, 16(SP)
                                      10     AE  9F 000EB          PUSHAB  16(SP)
                                             57  DD 000EE          PUSHL   R7
                  00000000G     00           06  FB 000F0          CALLS   #6, SMG$GET_TERM_DATA
                                31           50  E9 000F7          BLBC    STATUS, 15$
                                    0104     C3  DD 000FA 10$:     PUSHL   260(PBCB)                       ; 3876
                                             66  DD 000FE          PUSHL   (R6)                            ; 3875
                                             53  DD 00100          PUSHL   PBCB                            ; 3874
                     00000000G     00           03  FB 00102      CALLS   #3, SMG$$OUTPUT
                                55           50  D0 00109          MOVL    R0, STATUS
                                15           55  E9 0010C          BLBC    STATUS, 13$
                        A5      52           58  F3 0010F 11$:     AOBLEQ  R8, I, 8$                       ; 3849
                                      04     AC  DD 00113 12$:     PUSHL   PASTEBOARD_ID                   ; 3888
                                      08     AC  DD 00116          PUSHL   DISPLAY_ID                      ; 3887
                           FD94     CF           02  FB 00119      CALLS   #2, SMG$POP_VIRTUAL_DISPLAY
                                55           50  D0 0011E          MOVL    R0, STATUS
                                04           55  E8 00121          BLBS    STATUS, 14$
                                50           55  D0 00124 13$:     MOVL    STATUS, R0                      ; 3890
                                             04  00127            RET
                                50           01  D0 00128 14$:     MOVL    #1, R0                          ; 3892
                                             04  0012B 15$:       RET                                     ; 3894

; Routine Size:  300 bytes,    Routine Base:  _SMG$CODE + 1359

; 3651          3895  1 !<BLF/PAGE>
```

```
;  3653     3896  1 %SBTTL 'SMG$SAVE_PHYSICAL_SCREEN - Save physical screen'
;  3654     3897  1 GLOBAL ROUTINE SMG$SAVE_PHYSICAL_SCREEN (
;  3655     3898  1                                          PASTEBOARD_ID,
;  3656     3899  1                                          DISPLAY_ID,
;  3657     3900  1                                          DESIRED_ROW_START,
;  3658     3901  1                                          DESIRED_ROW_END
;  3659     3902  1                                         ) =
;  3660     3903  1 !++
;  3661     3904  1 ! FUNCTIONAL DESCRIPTION:
;  3662     3905  1 !
;  3663     3906  1 !       This routine should be called before calling a procedure which
;  3664     3907  1 !       may perform output to the screen without using the SMG$
;  3665     3908  1 !       This procedure saves the state of the screen so that it can be
;  3666     3909  1 !       restored via a later call to SMG$RESTORE_PHYSICAL_SCREEN.
;  3667     3910  1 !
;  3668     3911  1 !       This routine performs 4 functions:
;  3669     3912  1 !       It:
;  3670     3913  1 !           a). Creates a virtual display which is as wide as the
;  3671     3914  1 !               physical screen and is as high indicated by the
;  3672     3915  1 !               desired_row_start and desired_row_end.
;  3673     3916  1 !               The resulting virtual display id is returned
;  3674     3917  1 !               to the caller.
;  3675     3918  1 !
;  3676     3919  1 !           b). Pastes this virtual display to cover the screen at a
;  3677     3920  1 !               position corresponding to column 1 of desired_row_start.
;  3678     3921  1 !
;  3679     3922  1 !           c). Set the physical cursor to (1,1) in the virtual display.
;  3680     3923  1 !               This corresponds to (desired_row_start, 1) on the
;  3681     3924  1 !               physical screen.
;  3682     3925  1 !
;  3683     3926  1 !           d). Set the physical scrolling region to be the height
;  3684     3927  1 !               of the resulting virtual display.
;  3685     3928  1 !
;  3686     3929  1 !       If either desired_row_start or desired_row_end are omitted,
;  3687     3930  1 !       the first row of the physical display and the last row of the
;  3688     3931  1 !       physical display are used, respectively, in calculating the
;  3689     3932  1 !       height of the virtual display.
;  3690     3933  1 !
;  3691     3934  1 !       The effects of this routine can be reversed by doing a
;  3692     3935  1 !       SMG$RESTORE_PHYSICAL_SCREEN (Display_id.rl.r, Pasteboard_id.rl.r),
;  3693     3936  1 !       supplying the display_id returned by this routine.
;  3694     3937  1 !
;  3695     3938  1 ! CALLING SEQUENCE:
;  3696     3939  1 !
;  3697     3940  1 !       ret_status.wlc.v = SMG$SAVE_PHYSICAL_SCREEN (
;  3698     3941  1 !                                          PASTEBOARD_ID.rl.r,
;  3699     3942  1 !                                          DISPLAY_ID.wl.r
;  3700     3943  1 !                                          [,DESIRED_ROW_START.rl.r]
;  3701     3944  1 !                                          [,DESIRED_ROW_END.rl.r])
;  3702     3945  1 !
;  3703     3946  1 ! FORMAL PARAMETERS:
;  3704     3947  1 !
;  3705     3948  1 !       PASTEBOARD_ID.rl.r      Address of a pasteboard id which is to
;  3706     3949  1 !                               be "saved".
;  3707     3950  1 !
;  3708     3951  1 !       DISPLAY_ID.wl.r         Returned display id invented to
;  3709     3952  1 !                               perform requested function.
```

```
;  3710     3953  1 !
;  3711     3954  1 !       DESIRED_ROW_START.rl.r  Optional.  The address of the 1st row
;  3712     3955  1 !                               to be "saved".  If omitted, row 1 of
;  3713     3956  1 !                               the physical display is used.
;  3714     3957  1 !
;  3715     3958  1 !       DESIRED_ROW_END.rl.r    Optional.  The address of the last row
;  3716     3959  1 !                               to be "saved".  If omitted, the last
;  3717     3960  1 !                               row of the physical display is used.
;  3718     3961  1 ! IMPLICIT INPUTS:
;  3719     3962  1 !
;  3720     3963  1 !       NONE
;  3721     3964  1 !
;  3722     3965  1 ! IMPLICIT OUTPUTS:
;  3723     3966  1 !
;  3724     3967  1 !       NONE
;  3725     3968  1 !
;  3726     3969  1 ! COMPLETION STATUS:
;  3727     3970  1 !
;  3728     3971  1 !               SS$_NORMAL      Normal successful completion
;  3729     3972  1 !
;  3730     3973  1 ! From: SMG$CREATE_VIRTUAL_DISPLAY
;  3731     3974  1 !               LIB$_INSVIRMEM  Insufficient virtual memory
;  3732     3975  1 !
;  3733     3976  1 ! From: SMG$PASTE_VIRTUAL_DISPLAY
;  3734     3977  1 !               SMG$_INVPAS_ID  Invalid Pasteboard Id.
;  3735     3978  1 !
;  3736     3979  1 !
;  3737     3980  1 ! SIDE EFFECTS:
;  3738     3981  1 !
;  3739     3982  1 !       The appropriate part of the physical screen will be blanked,
;  3740     3983  1 !       scrolling region will be full height of the past to be "saved",
;  3741     3984  1 !       and cursor will be at (desired_row_start,1) on screen.
;  3742     3985  1 !--
;  3743     3986  1
;  3744     3987  2     BEGIN
;  3745     3988  2     BUILTIN
;  3746     3989  2         NULLPARAMETER;
;  3747     3990  2
;  3748     3991  2     LOCAL
;  3749     3992  2         ROW1,                       ! Resulting 1st row
;  3750     3993  2         ROWN,                       ! Resulting last row
;  3751     3994  2         FULL_SCREEN,                ! Logical indicating that we
;  3752     3995  2                                     ! are saving the whole screen.
;  3753     3996  2         PBCB : REF $PBCB_DECL,      ! Address of pasteboard control
;  3754     3997  2                                     ! block
;  3755     3998  2
;  3756     3999  2         NEW_DCB : REF $DCB_DECL,    ! Address of a display control
;  3757     4000  2                                     ! block.  This will also
;  3758     4001  2                                     ! become the display_id
;  3759     4002  2                                     ! returned.
;  3760     4003  2
;  3761     4004  2         STATUS;                     ! Status of subr. calls
;  3762     4005  2
;  3763     4006  2 !+
;  3764     4007  2 ! Validate number of arguments and get the PBCB that goes with the
;  3765     4008  2 ! Pasteboard id.
;  3766     4009  2 !-
```

```
;  3767     4010  2     $SMG$VALIDATE_ARGCOUNT( 2,4);
;  3768     4011  2
;  3769     4012  2     $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);
;  3770     4013  2
;  3771     4014  2 !+
;  3772     4015  2 ! Assume full screen case and intialize accordingly
;  3773     4016  2 !-
;  3774     4017  2     FULL_SCREEN = 1;    ! Assume full screen
;  3775     4018  2     ROW1 = 1;
;  3776     4019  2     ROWN = .PBCB [PBCB_B_ROWS];
;  3777     4020  2
;  3778     4021  2 !+
;  3779     4022  2 ! See which optional parameters were supplied and re-adjust assumptions.
;  3780     4023  2 !-
;  3781     4024  2     IF NOT NULLPARAMETER (DESIRED_ROW_START)
;  3782     4025  2     THEN
;  3783     4026  3         BEGIN   ! Desired_row_start specified
;  3784     4027  3         FULL_SCREEN = 0;
;  3785     4028  3         ROW1 = ..DESIRED_ROW_START;
;  3786     4029  3         END;    ! Desired_row_start specified
;  3787     4030  2
;  3788     4031  2     IF NOT NULLPARAMETER (DESIRED_ROW_END)
;  3789     4032  2     THEN
;  3790     4033  3         BEGIN   ! Desired_row_end specified
;  3791     4034  3         FULL_SCREEN = 0;
;  3792     4035  3         ROWN = ..DESIRED_ROW_END;
;  3793     4036  3         END;    ! Desired_row_end specified
;  3794     4037  2 !+
;  3795     4038  2 ! If either of the optional row parameters were supplied, make sure
;  3796     4039  2 ! we got a consistant range.
;  3797     4040  2 !-
;  3798     4041  2     IF NOT .FULL_SCREEN
;  3799     4042  2     THEN
;  3800     4043  3         BEGIN   ! Validity check on rows
;  3801     4044  3         IF .ROW1 LSS 1                              OR ! Start off top
;  3802     4045  3            .ROW1 GEQ .PBCB [PBCB_B_ROWS] -1         OR ! need 2 lines to scroll
;  3803     4046  3            .ROWN LSS 1                              OR ! End off top
;  3804     4047  3            .ROWN GTR .PBCB [PBCB_B_ROWS]            OR ! End off bottom
;  3805     4048  3            .ROWN - .ROW1 LSS 1                        ! Wrong order
;  3806     4049  3         THEN
;  3807     4050  3             RETURN SMG$_INVROW;
;  3808     4051  3
;  3809     4052  3         END;    ! Validity check on rows
;  3810     4053  2
;  3811     4054  2 !+
;  3812     4055  2 ! Create a virtual display the same width as the physical screen and
;  3813     4056  2 ! as high as desired.
;  3814     4057  2 !-
;  3815     4058  2     IF NOT (STATUS = SMG$$CREATE_VIRTUAL_DISPLAY (
;  3816     4059  2                         %REF ( .ROWN - .ROW1 +1),      ! # rows
;  3817     4060  2                         %REF (.PBCB [PBCB_W_WIDTH]),   ! # columns
;  3818     4061  2                         NEW_DCB,                       ! new disp. id
;  3819     4062  2                         %REF(0),                       ! default display attr
;  3820     4063  2                         %REF(0),                       ! default video attr
;  3821     4064  2                         %REF(0)                        ! default alt char set
;  3822     4065  2                         ))
;  3823     4066  2     THEN
```

```
;  3824     4067  2          RETURN (.STATUS);
;  3825     4068  2
;  3826     4069  2      !+
;  3827     4070  2      ! Paste newly-create virtual display to (desired_row_start,1) of
;  3828     4071  2      ! pasteboard.
;  3829     4072  2      !-
;  3830     4073  3          IF NOT (STATUS = SMG$$PASTE_VIRTUAL_DISPLAY (
;  3831     4074  3                              .NEW_DCB,                   ! DCB address
;  3832     4075  3                              .PBCB,                      ! Pasteboard control block
;  3833     4076  3                              ROW1,                       ! Row
;  3834     4077  3                              %REF (1)))                  ! Col 1
;  3835     4078  2          THEN
;  3836     4079  2              RETURN (.STATUS);
;  3837     4080  2
;  3838     4081  2
;  3839     4082  2      !+
;  3840     4083  2      ! Set physical scrolling region to be full height of screen.
;  3841     4084  2      !-
;  3842     4085  3          IF NOT (STATUS = SMG$$FORCE_SCROLL_REG ( .PBCB,      ! Pasteboard
;  3843     4086  3                                                   .ROW1,      ! Top row
;  3844     4087  3                                                   .ROWN))     ! Bottom row
;  3845     4088  2          THEN
;  3846     4089  2              RETURN (.STATUS);
;  3847     4090  2
;  3848     4091  2      !+
;  3849     4092  2      ! Return id of newly-create virtual display to caller.
;  3850     4093  2      !-
;  3851     4094  2          .DISPLAY_ID = .NEW_DCB;
;  3852     4095  2
;  3853     4096  2          RETURN (SS$_NORMAL);
;  3854     4097  2
;  3855     4098  1          END;                    ! End of routine SMG$SAVE_PHYSICAL_SCREEN


                                                001C 00000          .ENTRY  SMG$SAVE_PHYSICAL_SCREEN, Save R2,R3,R4      ; 3897
                                    54 00000000'  EF   9E 00002     MOVAB   PBD_L_COUNT, R4                              ;
                                                  5E   C2 00009     SUBL2   #28, SP                                      ;
                                50                6C   83 0000C     SUBB3   #2, (AP), DIFF                               ; 4010
                                                  02   91 00010     CMPB    DIFF, #2                                     ;
                                                       1B 00013     BLEQU   1$                                           ;
                                    50 00000000G  8F   DD 00015     MOVL    #SMG$_WRONUMARG, R0                          ;
                                                       04 0001C     RET                                                  ;
                                    50        04  BC   D0 0001D 1$: MOVL    @PASTEBOARD_ID, R0                           ; 4012
                                                  0A   19 00021     BLSS    2$                                           ;
                                    64            50   D1 00023     CMPL    R0, PBD_L_COUNT                              ;
                                                  05   14 00026     BGTR    2$                                           ;
                                08        44  A4   50   E0 00028     BBS     R0, PBD_V_PB_AVAIL, 3$                       ;
                                    50 00000000G  8F   D0 0002D 2$: MOVL    #SMG$_INVPAS_ID, R0                          ;
                                                       04 00034     RET                                                  ;
                                    53        04 A440  D0 00035 3$: MOVL    PBD_A_PBCB[R0], PBCB                         ;
                                    50            01   D0 0003A     MOVL    #1, FULL_SCREEN                              ; 4017
                                              18  AE   01   D0 0003D     MOVL    #1, ROW1                                     ; 4018
                                    52        5F  A3   9A 00041     MOVZBL  95(PBCB), ROWN                               ; 4019
                                                  03   6C   91 00045     CMPB    (AP), #3                                     ; 4024
```

```
                                                 0C  1F 00048          BLSSU   4$                                          ;
                                              0C AC  D5 0004A          TSTL    12(AP)                                      ;
                                                 07  13 0004D          BEQL    4$                                          ;
                                                 50  D4 0004F          CLRL    FULL_SCREEN                                 ; 4027
                                           18 AE 0C  BC  D0 00051          MOVL    @DESIRED_ROW_START, ROW1                    ; 4028
                                           04        6C  91 00056 4$:      CMPB    (AP), #4                                    ; 4031
                                                 0B  1F 00059          BLSSU   5$                                          ;
                                              10 AC  D5 0005B          TSTL    16(AP)                                      ;
                                                 06  13 0005E          BEQL    5$                                          ;
                                                 50  D4 00060          CLRL    FULL_SCREEN                                 ; 4034
                                           52    10  BC  D0 00062          MOVL    @DESIRED_ROW_END, ROWN                      ; 4035
                                           2F        50  E8 00066 5$:      BLBS    FULL_SCREEN, 7$                             ; 4041
                                              18 AE  D5 00069          TSTL    ROW1                                        ; 4044
                                                 22  15 0006C          BLEQ    6$                                          ;
                                           50    5F  A3  9A 0006E          MOVZBL  95(PBCB), R0                                ; 4045
                                                 50  D7 00072          DECL    R0                                          ;
                                           50    18 AE  D1 00074          CMPL    ROW1, R0                                    ;
                                                 16  18 00078          BGEQ    6$                                          ;
                                                 52  D5 0007A          TSTL    ROWN                                        ; 4046
                                                 12  15 0007C          BLEQ    6$                                          ;
52          5F  A3                           08        00  ED 0007E          CMPZV   #0, #8, 95(PBCB), ROWN                      ; 4047
                                                 0A  19 00084          BLSS    6$                                          ;
            50                 18 AE             01  C1 00086          ADDL3   #1, ROW1, R0                                ; 4048
                                           50        52  D1 0008B          CMPL    ROWN, R0                                    ;
                                                 08  18 0008E          BGEQ    7$                                          ;
                                           50 00000000G  8F  D0 00090 6$:      MOVL    #SMG$_INVROW, R0                            ; 4050
                                                 04  00097          RET                                                 ;
                                              10 AE  D4 00098 7$:      CLRL    16(SP)                                      ; 4064
                                              10 AE  9F 0009B          PUSHAB  16(SP)                                      ;
                                              10 AE  D4 0009E          CLRL    16(SP)                                      ; 4063
                                              10 AE  9F 000A1          PUSHAB  16(SP)                                      ;
                                              10 AE  D4 000A4          CLRL    16(SP)                                      ; 4062
                                              10 AE  9F 000A7          PUSHAB  16(SP)                                      ;
                                              20 AE  9F 000AA          PUSHAB  NEW_DCB                                     ; 4058
                                        14 AE 5A  A3  3C 000AD          MOVZWL  90(PBCB), 20(SP)                            ; 4060
                                              14 AE  9F 000B2          PUSHAB  20(SP)                                      ;
50                                 52    2C  AE  C3 000B5          SUBL3   ROW1, ROWN, R0                              ; 4059
                                        14 AE 01  A0  9E 000BA          MOVAB   1(R0), 20(SP)                               ;
                                              14 AE  9F 000BF          PUSHAB  20(SP)                                      ;
                                     0000V CF        06  FB 000C2          CALLS   #6, SMG$$CREATE_VIRTUAL_DISPLAY             ;
                                           30        50  E9 000C7          BLBC    STATUS, 8$                                  ; 4058
                                        10 AE        01  D0 000CA          MOVL    #1, 16(SP)                                  ; 4077
                                              10 AE  9F 000CE          PUSHAB  16(SP)                                      ;
                                              1C AE  9F 000D1          PUSHAB  ROW1                                        ; 4073
                                                 53  DD 000D4          PUSHL   PBCB                                        ; 407?
                                              20 AE  DD 000D6          PUSHL   NEW_DCB                                     ; 4074
                                     0000V CF        04  FB 000D9          CALLS   #4, SMG$$PASTE_VIRTUAL_DISPLAY              ;
                                           19        50  E9 000DE          BLBC    STATUS, 8$                                  ; 4073
                                                 52  DD 000E1          PUSHL   ROWN                                        ; 4087
                                              1C AE  DD 000E3          PUSHL   ROW1                                        ; 4086
                                                 53  DD 000E6          PUSHL   PBCB                                        ; 4085
                              00000000G    00        03  FB 000E8          CALLS   #3, SMG$$FORCE_SCROLL_REG                   ;
                                           08        50  E9 000EF          BLBC    STATUS, 8$                                  ;
                                        08 BC 14 AE  D0 000F2          MOVL    NEW_DCB, @DISPLAY_ID                        ; 4094
                                           50        01  D0 000F7          MOVL    #1, R0                                      ; 4096
                                                 04  000FA 8$:      RET                                                 ; 4098
```

```
; Routine Size:  251 bytes,    Routine Base:  _SMG$CODE + 1485

; 3856          4099  1 !<BLF/PAGE>
```

```
;  3858     4100  1 %SBTTL 'SMG$SET_DISPLAY_SCROLL_REGION - Set scrolling region in a virtual display'
;  3859     4101  1 GLOBAL ROUTINE SMG$SET_DISPLAY_SCROLL_REGION (
;  3860     4102  1                                               DISPLAY_ID,
;  3861     4103  1                                               TOP_LINE_OF_REGION,
;  3862     4104  1                                               BOTTOM_LINE_OF_REGION
;  3863     4105  1                                              ) =
;  3864     4106  1 !++
;  3865     4107  1 ! FUNCTIONAL DESCRIPTION:
;  3866     4108  1 !
;  3867     4109  1 !       This routine sets the top and bottom lines of a 'scrolling region'
;  3868     4110  1 !       in a virtual display.  The scrolling region limits are used by
;  3869     4111  1 !       output routines which scroll (SMG$PUT_WITH_SCROLL and
;  3870     4112  1 !       SMG$PUT_LINE with line advancing).  If this routine is called
;  3871     4113  1 !       with only a display_id, the scrolling region defaults to the
;  3872     4114  1 !       entire display.
;  3873     4115  1 !
;  3874     4116  1 !       If a top and bottom line are passed, they must be within the
;  3875     4117  1 !       display bounds.  Scrolling can not occur outside the bounds of
;  3876     4118  1 !       a display.
;  3877     4119  1 !
;  3878     4120  1 !       This routine does not change the appearance of the screen or the
;  3879     4121  1 !       cursor position.
;  3880     4122  1 !
;  3881     4123  1 ! CALLING SEQUENCE:
;  3882     4124  1 !
;  3883     4125  1 !       ret_status.wlc.v = SMG$SET_DISPLAY_SCROLL_REGION (
;  3884     4126  1 !                                                         DISPLAY_ID.rl.r
;  3885     4127  1 !                                                         [,TOP_LINE_OF_REGION.rl.r]
;  3886     4128  1 !                                                         [,BOTTOM_LINE_OF_REGION.rl.r])
;  3887     4129  1 !
;  3888     4130  1 ! FORMAL PARAMETERS:
;  3889     4131  1 !
;  3890     4132  1 !       DISPLAY_ID.rl.r         Display id of desired display.
;  3891     4133  1 !
;  3892     4134  1 !       TOP_LINE_OF_REGION.rl.r Optional.  The top line of a scrolling
;  3893     4135  1 !                               region.  Defaults to line 1 of the display.
;  3894     4136  1 !
;  3895     4137  1 !       BOTTOM_LINE_OF_REGION.rl.r
;  3896     4139  1 !                               Optional.  The bottom line of a scrolling
;  3897     4139  1 !                               region.  Defaults to the bottom line of the
;  3898     4140  1 !                               display.
;  3899     4141  1 !
;  3900     4142  1 !
;  3901     4143  1 ! IMPLICIT INPUTS:
;  3902     4144  1 !
;  3903     4145  1 !       NONE
;  3904     4146  1 !
;  3905     4147  1 ! IMPLICIT OUTPUTS:
;  3906     4148  1 !
;  3907     4149  1 !       NONE
;  3908     4150  1 !
;  3909     4151  1 ! COMPLETION STATUS:
;  3910     4152  1 !
;  3911     4153  1 !       SS$_NORMAL      Normal successful completion
;  3912     4154  1 !       SMG$_INVARG     Bottom line is less than or equal to top line.
;  3913     4155  1 !       SMG$_INVROW     Row number is negative or too large
;  3914     4156  1 !       SMG$_WRONUMARG  Wrong number arguments.
```

```
;  3915          4157  1 !
;  3916          4158  1 ! SIDE EFFECTS:
;  3917          4159  1 !
;  3918          4160  1 !       NONE
;  3919          4161  1 !--
;  3920          4162  2     BEGIN
;  3921          4163  2     BUILTIN
;  3922          4164  2         NULLPARAMETER;
;  3923          4165  2
;  3924          4166  2     LOCAL
;  3925          4167  2         TOP_LINE,                       ! working top line
;  3926          4168  2         BOTTOM_LINE,                    ! working bottom line
;  3927          4169  2         DCB : REF $DCB_DECL;            ! Addr. of display control block
;  3928          4170  2
;  3929          4171  2     $SMG$VALIDATE_ARGCOUNT (1,3);
;  3930          4172  2
;  3931          4173  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);   ! Get address of display control
;  3932          4174  2                                         ! block
;  3933          4175  2
;  3934          4176  2 !+
;  3935          4177  2 ! Validate optional arguments.
;  3936          4178  2 !-
;  3937          4179  2
;  3938          4180  2     TOP_LINE = .DCB [DCB_W_ROW_START];  ! init to default
;  3939          4181  2
;  3940          4182  2     IF NOT NULLPARAMETER (TOP_LINE_OF_REGION)
;  3941          4183  2     THEN
;  3942          4184  3         BEGIN
;  3943          4185  3         IF ..TOP_LINE_OF_REGION GEQ .DCB [DCB_W_ROW_START] AND
;  3944          4186  3             ..TOP_LINE_OF_REGION LEQ .DCB [DCB_W_NO_ROWS]
;  3945          4187  3         THEN
;  3946          4188  3             TOP_LINE = ..TOP_LINE_OF_REGION
;  3947          4189  3         ELSE
;  3948          4190  3             RETURN (SMG$_INVROW);       ! can't be outside display
;  3949          4191  2         END;
;  3950          4192  2
;  3951          4193  2     BOTTOM_LINE = .DCB [DCB_W_NO_ROWS]; ! init to default
;  3952          4194  2
;  3953          4195  2     IF NOT NULLPARAMETER (BOTTOM_LINE_OF_REGION)
;  3954          4196  2     THEN
;  3955          4197  3         BEGIN
;  3956          4198  3         IF ..BOTTOM_LINE_OF_REGION GEQ .DCB [DCB_W_ROW_START] AND
;  3957          4199  3             ..BOTTOM_LINE_OF_REGION LEQ .DCB [DCB_W_NO_ROWS]
;  3958          4200  3         THEN
;  3959          4201  3             BOTTOM_LINE = ..BOTTOM_LINE_OF_REGION
;  3960          4202  3         ELSE
;  3961          4203  3             RETURN (SMG$_INVROW);       ! can't be outside display
;  3962          4204  2         END;
;  3963          4205  2
;  3964          4206  2     IF .BOTTOM_LINE LEQ .TOP_LINE
;  3965          4207  2     THEN
;  3966          4208  2         RETURN (SMG$_INVARG);           ! can't go backwards or
;  3967          4209  2                                         ! overlap
;  3968          4210  2
;  3969          4211  2 !+
;  3970          4212  2 ! If we get here, we have a valid scrolling region.  Store it.
;  3971          4213  2 !-
```

```
; 3972      4214  2
; 3973      4215  2      DCB [DCB_W_TOP_OF_SCRREG] = .TOP_LINE;
; 3974      4216  2      DCB [DCB_W_BOTTOM_OF_SCRREG] = .BOTTOM_LINE;
; 3975      4217  2
; 3976      4218  2      RETURN (SS$_NORMAL);
; 3977      4219  1      END;                    ! end of routine SMG$SET_DISPLAY_SCROLL_REGION


                                        0004 00000      .ENTRY  SMG$SET_DISPLAY_SCROLL_REGION, Save R2      ; 4101
                 50          6C      01  83 00002       SUBB3   #1, (AP), DIFF                              ; 4171
                             02      50  91 00006       CMPB    DIFF, #2                                    ;
                                     08  1B 00009       BLEQU   1$                                          ;
                             50 00000000G 8F D0 0000B   MOVL    #SMG$_WRONUMARG, R0                         ;
                                     04 00012           RET                                                 ;
                             50    04  BC  D0 00013 1$: MOVL    @DISPLAY_ID, R0                             ; 4173
                     04  BC    38  A0  D1 00017         CMPL    56(R0), @DISPLAY_ID                         ;
                                     06  12 0001C       BNEQ    2$                                          ;
                             11    44  A0  91 0001E     CMPB    68(R0), #17                                 ;
                                     08  13 00022       BEQL    3$                                          ;
                             50 00000000G 8F D0 00024 2$: MOVL  #SMG$_INVDIS_ID, R0                         ;
                                     04 0002B           RET                                                 ;
                             50    04  BC  D0 0002C 3$: MOVL    @DISPLAY_ID, DCB                            ;
                             52        60  3C 00030     MOVZWL  (DCB), TOP_LINE                             ; 4180
                             02        6C  91 00033     CMPB    (AP), #2                                    ; 4182
                                     1A  1F 00036       BLSSU   4$                                          ;
                                   08  AC  D5 00038     TSTL    8(AP)                                       ;
                                     15  13 0003B       BEQL    4$                                          ;
   08   BC          60       10      00  ED 0003D       CMPZV   #0, #16, (DCB), @TOP_LINE_OF_REGION         ; 4185
                                     32  14 00043       BGTR    5$                                          ;
   08   BC     02   A0       10      00  ED 00045       CMPZV   #0, #16, 2(DCB), @TOP_LINE_OF_REGION        ; 4186
                                     29  19 0004C       BLSS    5$                                          ;
                             52    08  BC  D0 0004E     MOVL    @TOP_LINE_OF_REGION, TOP_LINE               ; 4188
                             51    02  A0  3C 00052 4$: MOVZWL  2(DCB), BOTTOM_LINE                         ; 4193
                             03        6C  91 00056     CMPB    (AP), #3                                    ; 4195
                                     24  1F 00059       BLSSU   6$                                          ;
                                   0C  AC  D5 0005B     TSTL    12(AP)                                      ;
                                     1F  13 0005E       BEQL    6$                                          ;
   0C   BC          60       10      00  ED 00060       CMPZV   #0, #16, (DCB), @BOTTOM_LINE_OF_REGION      ; 4198
                                     0F  14 00066       BGTR    5$                                          ;
   0C   BC     02   A0       10      00  ED 00068       CMPZV   #0, #16, 2(DCB), @BOTTOM_LINE_OF_REGION     ; 4199
                                     06  19 0006F       BLSS    5$                                          ;
                             51    0C  BC  D0 00071     MOVL    @BOTTOM_LINE_OF_REGION, BOTTOM_LINE         ; 4201
                                     08  11 00075       BRB     6$                                          ;
                             50 00000000G 8F D0 00077 5$: MOVL  #SMG$_INVROW, R0                            ; 4203
                                     04 0007E           RET                                                 ;
                             52        51  D1 0007F 6$: CMPL    BOTTOM_LINE, TOP_LINE                       ; 4206
                                     08  14 00082       BGTR    7$                                          ;
                             50 00000000G 8F D0 00084   MOVL    #SMG$_INVARG, R0                            ; 4208
                                     04 0008B           RET                                                 ;
                     48  A0        52  B0 0008C 7$:     MOVW    TOP_LINE, 72(DCB)                           ; 4215
                     4A  A0        51  B0 00090         MOVW    BOTTOM_LINE, 74(DCB)                        ; 4216
                             50        01  D0 00094     MOVL    #1, R0                                      ; 4218
                                     04 00097           RET                                                 ; 4219
```

```
; Routine Size:  152 bytes,    Routine Base:  _SMG$CODE + 1580

; 3978          4220  1 !<BLF/PAGE>
```

```
;  3980          4221  1 %SBTTL 'SMG$UNPASTE_VIRTUAL_DISPLAY - Unpaste virtual display from pasteboard'
;  3981          4222  1 GLOBAL ROUTINE SMG$UNPASTE_VIRTUAL_DISPLAY (
;  3982          4223  1                                             DISPLAY_ID,
;  3983          4224  1                                             PASTEBOARD_ID
;  3984          4225  1                                            ) =
;  3985          4226  1 !++
;  3986          4227  1 ! FUNCTIONAL DESCRIPTION:
;  3987          4228  1 !
;  3988          4229  1 !       The specified virtual display is "unpasted" from a pasteboard.
;  3989          4230  1 !       Unpasting does not destroy the virtual display or its contents.
;  3990          4231  1 !       It merely removes its mapping to a particular pasteboard and
;  3991          4232  1 !       hence its visibility on that pasteboard.
;  3992          4233  1 !
;  3993          4234  1 ! CALLING SEQUENCE:
;  3994          4235  1 !
;  3995          4236  1 !       ret_status.wlc.v = SMG$UNPASTE_VIRTUAL_DISPLAY (
;  3996          4237  1 !                                                       DISPLAY_ID.rl.r,
;  3997          4238  1 !                                                       PASTEBOARD_ID.rl.r)
;  3998          4239  1 !
;  3999          4240  1 ! FORMAL PARAMETERS:
;  4000          4241  1 !
;  4001          4242  1 !       DISPLAY_ID.rl.r         Id of virtual display to be unpasted.
;  4002          4243  1 !
;  4003          4244  1 !       PASTEBOARD_ID.rl.r      The pasteboard id of the pasteboard from
;  4004          4245  1 !                               which the unpasting is to take place.
;  4005          4246  1 !
;  4006          4247  1 ! IMPLICIT INPUTS:
;  4007          4248  1 !
;  4008          4249  1 !       None
;  4009          4250  1 !
;  4010          4251  1 ! IMPLICIT OUTPUTS:
;  4011          4252  1 !
;  4012          4253  1 !       None
;  4013          4254  1 !
;  4014          4255  1 ! COMPLETION STATUS:
;  4015          4256  1 !
;  4016          4257  1 !       SS$_NORMAL      Normal successful completion
;  4017          4258  1 !       SMG$_INVDIS_ID  Invalid virtual display id.
;  4018          4259  1 !       SMG$_INVPAS_ID  Invalid pasteboard id.
;  4019          4260  1 !       SMG$_WRONUMARG  Wrong number of arguments.
;  4020          4261  1 !       SMG$_NOTPASTED  Specified virtual display is not currently
;  4021          4262  1 !                       pasted to the specified pasteboard.
;  4022          4263  1 !       SMG$_ILLBATFNC  Display is batched.
;  4023          4264  1 !
;  4024          4265  1 ! SIDE EFFECTS:
;  4025          4266  1 !
;  4026          4267  1 !       NONE
;  4027          4268  1 !--
;  4028          4269  2     BEGIN
;  4029          4270  2     LOCAL
;  4030          4271  2         STATUS,                         ! Status of subroutine call
;  4031          4272  2
;  4032          4273  2         DCB     : REF $DCB_DECL,        ! Addr of display control block
;  4033          4274  2         PBCB    : REF $PBCB_DECL;       ! Addr of pasteboard control block
;  4034          4275  2
;  4035          4276  2     $SMG$VALIDATE_ARGCOUNT (2, 2);      ! Test for right no. of args
;  4036          4277  2
```

```
;  4037     4278  2 !+
;  4038     4279  2 ! Get addresses of virtual display control block and pasteboard control
;  4039     4280  2 ! block and validate display id and pasteboard id.
;  4040     4281  2 !-
;  4041     4282  2     $SMG$GET_PBCB ( .PASTEBOARD_ID, PBCB);
;  4042     4283  2                                         ! Get addr of pasteboard control
;  4043     4284  2                                         ! block
;  4044     4285  2
;  4045     4286  2     $SMG$GET_DCB ( .DISPLAY_ID, DCB);   ! Get addr of virtual display
;  4046     4287  2                                         ! control block
;  4047     4288  2
;  4048     4289  2 !+
;  4049     4290  2 ! Give an error if the display is batched.
;  4050     4291  2 !-
;  4051     4292  2
;  4052     4293  2     IF .DCB[DLB_L_BATCH_LEVEL] NEQ 0
;  4053     4294  2     THEN
;  4054     4295  2         RETURN  SMG$_ILLBATFNC;
;  4055     4296  2
;  4056     4297  2     RETURN SMG$$UNPASTE_VIRTUAL_DISPLAY(.DCB,.PBCB)
;  4057     4298  2
;  4058     4299  1     END;                          ! Routine SMG$UNPASTE_VIRTUAL_DISPLAY



                                0004 00000          .ENTRY  SMG$UNPASTE_VIRTUAL_DISPLAY, Save R2        ; 4222
                    52 00000000'   EF   9E 00002    MOVAB   PBD_L_COUNT, R2
                    02             6C   91 00009    CMPB    (AP), #2                                    ; 4276
                                   08   13 0000C    BEQL    1$
                    50 00000000G   8F   D0 0000E    MOVL    #SMG$_WRONUMARG, R0
                                   04   00015      RET
                    50        08   BC   D0 00016 1$: MOVL    @PASTEBOARD_ID, R0                          ; 4282
                                   0A   19 0001A    BLSS    2$
                    62             50   D1 0001C    CMPL    R0, PBD_L_COUNT
                                   05   14 0001F    BGTR    2$
08        44        A2             50   E0 00021    BBS     R0, PBD_V_PB_AVAIL, 3$
                    50 00000000G   8F   D0 00026 2$: MOVL    #SMG$_INVPAS_ID, R0
                                   04   0002D      RET
                    51        04 A240   D0 0002E 3$: MOVL    PBD_A_PBCB[R0], PBCB
                    50        04   BC   D0 00033    MOVL    @DISPLAY_ID, R0                             ; 4286
              04    BC        38   A0   D1 00037    CMPL    56(R0), @DISPLAY_ID
                                   06   12 0003C    BNEQ    4$
                    11        44   A0   91 0003E    CMPB    68(R0), #17
                                   08   13 00042    BEQL    5$
                    50 00000000G   8F   D0 00044 4$: MOVL    #SMG$_INVDIS_ID, R0
                                   04   0004B      RET
                    50        04   BC   D0 0004C 5$: MOVL    @DISPLAY_ID, DCB
                              1C   A0   D5 00050    TSTL    28(DCB)                                     ; 4293
                                   08   13 00053    BEQL    6$
                    50 00000000G   8F   D0 00055    MOVL    #SMG$_ILLBATFNC, R0                         ; 4295
                                   04   0005C      RET
                                   03   BB 0005D 6$: PUSHR   #^M<R0,R1>                                 ; 4297
          0000V     CF             02   FB 00060    CALLS   #2, SMG$$UNPASTE_VIRTUAL_DISPLAY
                                   04   00064      RET                                                 ; 4299
```

```
; Routine Size:  101 bytes,    Routine Base:  _SMG$CODE + 1618

; 4059          4300  1 !<BLF/PAGE>
```

```
;  4061          4301  1 %SBTTL 'SMG$$CALC_PASTE_TRANSF - Calculate pasting transformation constants'
;  4062          4302  1 GLOBAL ROUTINE SMG$$CALC_PASTE_TRANSF ( PP  : REF $PP_DECL ) =
;  4063          4303  1 !++
;  4064          4304  1 ! FUNCTIONAL DESCRIPTION:
;  4065          4305  1 !
;  4066          4306  1 !       This procedure precalculates the constants needed to efficiently
;  4067          4307  1 !       copy portions of the text and attributes from the virtual
;  4068          4308  1 !       display buffers located in the DCB to the window buffer located
;  4069          4309  1 !       in the WCB.
;  4070          4310  1 !       This data is derived from the pasting relationship between
;  4071          4311  1 !       the given virtual display and the pasteboard to which it is
;  4072          4312  1 !       pasted.  The calculated constants are stored in the pasting
;  4073          4313  1 !       packet that reflects this pasting.
;  4074          4314  1 !
;  4075          4315  1 !
;  4076          4316  1 ! CALLING SEQUENCE:
;  4077          4317  1 !
;  4078          4318  1 !       ret_status.wlc.v = SMG$$CALC_PASTE_TRANSF ( PP.mab.r)
;  4079          4319  1 !
;  4080          4320  1 ! FORMAL PARAMETERS:
;  4081          4321  1 !
;  4082          4322  1 !       PP.mab.r        Address of pasting packet.
;  4083          4323  1 !
;  4084          4324  1 ! IMPLICIT INPUTS:
;  4085          4325  1 !
;  4086          4326  1 !       NONE
;  4087          4327  1 !
;  4088          4328  1 ! IMPLICIT OUTPUTS:
;  4089          4329  1 !
;  4090          4330  1 !       NONE
;  4091          4331  1 !
;  4092          4332  1 ! COMPLETION STATUS:
;  4093          4333  1 !
;  4094          4334  1 !       SS$_NORMAL      Normal successful completion
;  4095          4335  1 !
;  4096          4336  1 ! SIDE EFFECTS:
;  4097          4337  1 !
;  4098          4338  1 !       NONE
;  4099          4339  1 !--
;  4100          4340  1
;  4101          4341  2     BEGIN
;  4102          4342  2     LOCAL
;  4103          4343  2         TEMP    : BLOCK [8,BYTE],       ! Temporary representation of
;  4104          4344  2                                         ! display buffer area as
;  4105          4345  2                                         ! projected on window buffer.
;  4106          4346  2         DCB     : REF $DCB_DECL,        ! Addr of DCB involved.
;  4107          4347  2         WCB     : REF $WCB_DECL,        ! Addr of WCB involved.
;  4108          4348  2         PBCB    : REF $PBCB_DECL,       ! Addr of PBCB involved.
;  4109          4349  2         OVERLAP : BLOCK [8,BYTE];       ! Describes area of overlap
;  4110          4350  2                                         ! between virtual display and
;  4111          4351  2                                         ! window buffer.
;  4112          4352  2
;  4113          4353  2     PBCB = .PP [PP_A_PBCB_ADDR] ;
;  4114          4354  2     WCB = .PBCB [PBCB_A_WCB] ;
;  4115          4355  2     DCB = .PP [PP_A_DCB_ADDR] ;
;  4116          4356  2
;  4117          4357  2 !+
```

```
; 4118          4358  2 ! Mark the border label as being invisible until it proves otherwise.
; 4119          4359  2 !-
; 4120          4360  2     PP [PP_W_LABEL_BYTES_TO_MOVE] = 0;
; 4121          4361  2     PP [PP_W_SRC_LABEL_OFF]       = 0;
; 4122          4362  2     PP [PP_W_DST_LABEL_OFF]       = 0;
; 4123          4363  2
; 4124          4364  2
; 4125          4365  2     TEMP [DCB_W_ROW_START] = .DCB [DCB_W_ROW_START] + .PP [PP_W_ROW]-1;
; 4126          4366  2     TEMP [DCB_W_NO_ROWS]   = .DCB [DCB_W_NO_ROWS];
; 4127          4367  2     TEMP [DCB_W_COL_START] = .DCB [DCB_W_COL_START] + .PP [PP_W_COL]-1;
; 4128          4368  2     TEMP [DCB_W_NO_COLS]   = .DCB [DCB_W_NO_COLS];
; 4129          4369  2
; 4130          4370  2 !+
; 4131          4371  2 ! Check to see what part (if any) of this virtual display maps onto
; 4132          4372  2 ! the viewable part of the pasteboard -- i.e., the area that goes into
; 4133          4373  2 ! the window control block buffer.
; 4134          4374  2 !-
; 4135          4375  2     IF NOT SMG$$OCCLUDE (
; 4136          4376  2                          WCB [WCB_Q_COORD],    ! Area of window buffer
; 4137          4377  2                          TEMP,                 ! Area of display buffer
; 4138          4378  2                          OVERLAP )             ! Area of overlap
; 4139          4379  2                                                ! (if any)
; 4140          4380  2     THEN
; 4141          4381  3         BEGIN   ! No overlap
; 4142          4382  3         PP [PP_W_ROWS_TO_MOVE] = 0 ; ! There are no rows to move
; 4143          4383  3         !+
; 4144          4384  3         ! If the display isn't visible, the border label isn't visible
; 4145          4385  3         ! either.    **** Not really true -- clean this up later ****
; 4146          4386  3         !-
; 4147          4387  3         END     ! No overlap
; 4148          4388  2     ELSE
; 4149          4389  3         BEGIN   ! Overlap
; 4150          4390  3         LOCAL
; 4151          4391  3             DCB_START_ROW,      ! 1st row of display buffer that lands
; 4152          4392  3                                 ! in window buffer.
; 4153          4393  3             DCB_START_COL;      ! 1st column of display buffer that
; 4154          4394  3                                 ! lands in window buffer.
; 4155          4395  3
; 4156          4396  3         PP [PP_W_ROWS_TO_MOVE] = .OVERLAP [DCB_W_NO_ROWS];
; 4157          4397  3         PP [PP_W_MOVE_LENGTH]  = .OVERLAP [DCB_W_NO_COLS];
; 4158          4398  3
; 4159          4399  3         PP [PP_W_FIRST_WCB_ROW] = .OVERLAP [DCB_W_ROW_START];
; 4160          4400  3         PP [PP_W_LAST_WCB_ROW]  = .OVERLAP [DCB_W_ROW_START] +
; 4161          4401  3                                   .OVERLAP [DCB_W_NO_ROWS] - 1;
; 4162          4402  3
; 4163          4403  3         PP [PP_W_FIRST_WCB_COL] = .OVERLAP [DCB_W_COL_START];
; 4164          4404  3         PP [PP_W_LAST_WCB_COL]  = .OVERLAP [DCB_W_COL_START] +
; 4165          4405  3                                   .OVERLAP [DCB_W_NO_COLS] - 1;
; 4166          4406  3
; 4167          4407  3         PP [PP_L_MOVE_SIZE] = .OVERLAP [DCB_W_NO_ROWS] *
; 4168          4408  3                               .OVERLAP [DCB_W_NO_COLS];
; 4169          4409  3
; 4170          4410  3         DCB_START_ROW = .OVERLAP [DCB_W_ROW_START] - .PP [PP_W_ROW] +1;
; 4171          4411  3         DCB_START_COL = .OVERLAP [DCB_W_COL_START] - .PP [PP_W_COL] +1;
; 4172          4412  3
; 4173          4413  3         PP [PP_W_FROM_INDEX] = (.DCB_START_ROW -1) *.DCB [DCB_W_NO_COLS]
; 4174          4414  3                              + .DCB_START_COL - 1;
```

```
;  4175          4415  3
;  4176          4416  3          PP [PP_W_TO_INDEX] = (.OVERLAP [DCB_W_ROW_START] -1) *
;  4177          4417  3                               .WCB [WCB_W_NO_COLS] +
;  4178          4418  3                               .OVERLAP [DCB_W_COL_START] -1;
;  4179          4419  3
;  4180          4420  3          IF .DCB [DCB_V_BORDERED]
;  4181          4421  3          THEN
;  4182          4422  4              BEGIN       ! Bordered display
;  4183          4423  4              LOCAL
;  4184          4424  4                  UPPER_ROW,      ! Row above top row of pasted display
;  4185          4425  4                  LOWER_ROW,      ! Row below bottom row of pasted display
;  4186          4426  4                  LEFT_COL,               ! Col. to left of pasted display
;  4187          4427  4                  RIGHT_COL,      ! Col. to right of pasted display
;  4188          4428  4                  LDES : REF BLOCK [,BYTE];! Address of dynamic descr. in
;  4189          4429  4                                          ! DCB that points to label
;  4190          4430  4                                          ! string.
;  4191          4431  4
;  4192          4432  4              LDES = DCB [DCB_Q_LABEL_DESC];
;  4193          4433  4
;  4194          4434  4              !+
;  4195          4435  4              ! Compute the row and column numbers where the borders fall.
;  4196          4436  4              ! Note these rows and columns may not map into the buffer
;  4197          4437  4              ! and need to be validated before use.
;  4198          4438  4              !-
;  4199          4439  4              UPPER_ROW = .PP [PP_W_ROW] - 1 ;
;  4200          4440  4              LOWER_ROW = .PP [PP_W_ROW] + .DCB [DCB_W_NO_ROWS];
;  4201          4441  4              LEFT_COL  = .PP [PP_W_COL] - 1 ;
;  4202          4442  4              RIGHT_COL = .PP [PP_W_COL] + .DCB [DCB_W_NO_COLS];
;  4203          4443  4
;  4204          4444  4              IF .LDES [DSC$W_LENGTH] NEQ 0
;  4205          4445  4              THEN
;  4206          4446  5                  BEGIN   ! Label position computation
;  4207          4447  5                  CASE .DCB [DCB_B_LABEL_POS] FROM SMG$K_TOP TO SMG$K_RIGHT OF
;  4208          4448  5                  SET
;  4209          4449  5                      [SMG$K_TOP]:
;  4210          4450  6                          BEGIN   ! Label in top row
;  4211          4451  6                          IF .UPPER_ROW GEQ 1
;  4212          4452  6                          THEN
;  4213          4453  7                              BEGIN       ! Top row in buffer
;  4214          4454  7                              LOCAL
;  4215          4455  7                                  DCOLS : SIGNED; ! Dest. col. start
;  4216          4456  7
;  4217          4457  7                              DCOLS =         .PP [PP_W_COL] +
;  4218          4458  7                                              .DCB [DCB_W_LABEL_UNITS] -2 ;
;  4219          4459  7
;  4220          4460  7                              IF .DCOLS LEQ .WCB [WCB_W_NO_COLS]
;  4221          4461  7                              THEN
;  4222          4462  8                                  BEGIN   ! partially on screen
;  4223          4463  8                                  LOCAL
;  4224          4464  8                                      DCOLE : SIGNED; ! Dest. col end
;  4225          4465  8
;  4226          4466  8                                  DCOLE = MIN ( (.LDES [DSC$W_LENGTH] + .DCOLS -1),
;  4227          4467  9                                                (.PP [PP_W_COL] +
;  4228          4468  8                                                 .DCB [DCB_W_NO_COLS] ),
;  4229          4469  8                                                 .WCB [WCB_W_NO_COLS]);
;  4230          4470  8
;  4231          4471  8                                  PP [PP_W_LABEL_BYTES_TO_MOVE] =
```

```
;  4232          4472  8                                    MAX ( 0, .DCOLE +1 -
;  4233          4473  8                                            MAX (0, .DCOLS) );
;  4234          4474  8
;  4235          4475  8                                IF .PP [PP_W_COL] LEQ 0
;  4236          4476  8                                THEN
;  4237          4477  9                                    BEGIN       ! Using tail end of label
;  4238          4478  9                                    PP [PP_W_SRC_LABEL_OFF] =
;  4239          4479  9                                        .LDES [DSC$W_LENGTH] -
;  4240          4480  9                                        .PP [PP_W_LABEL_BYTES_TO_MOVE];
;  4241          4481  9                                    END ! Using tail end of label
;  4242          4482  8                                ELSE
;  4243          4483  9                                    BEGIN       ! Using front end of label
;  4244          4484  9                                    PP [PP_W_SRC_LABEL_OFF] = 0;
;  4245          4485  8                                    END;        ! Using front end of label
;  4246          4486  8
;  4247          4487  8                                PP [PP_W_DST_LABEL_OFF] = (.UPPER_ROW -1) *
;  4248          4488  8                                                  .WCB [WCB_W_NO_COLS] +
;  4249          4489  8                                                 MAX(0, .DCOLS - 1);
;  4250          4490  7                                END; ! Partially on screen
;  4251          4491  7
;  4252          4492  6                            END;        ! Top row in buffer
;  4253          4493  5                        END;    ! Label in top row
;  4254          4494  5
;  4255          4495  5                    [SMG$K_BOTTOM]:
;  4256          4496  6                        BEGIN   ! Label in bottom row
;  4257          4497  6                        IF .LOWER_ROW LEQ .WCB [WCB_W_NO_ROWS]
;  4258          4498  6                        THEN
;  4259          4499  7                            BEGIN       ! Bottom row in buffer
;  4260          4500  7                            LOCAL
;  4261          4501  7                                DCOLS : SIGNED; ! Dest. col. start
;  4262          4502  7
;  4263          4503  7                            DCOLS =         .PP [PP_W_COL] +
;  4264          4504  7                                            .DCB [DCB_W_LABEL_UNITS] - 2;
;  4265          4505  7
;  4266          4506  7                            IF .DCOLS LEQ .WCB [WCB_W_NO_COLS]
;  4267          4507  7                            THEN
;  4268          4508  8                                BEGIN ! Partially visible
;  4269          4509  8                                LOCAL
;  4270          4510  8                                    DCOLE : SIGNED; ! Dest. col. end
;  4271          4511  8
;  4272          4512  8                                DCOLE = MIN ( (.LDES [DSC$W_LENGTH] + .DCOLS -1),
;  4273          4513  9                                        (.PP [PP_W_COL] +
;  4274          4514  8                                         .DCB [DCB_W_NO_COLS] ),
;  4275          4515  8                                         .WCB [WCB_W_NO_COLS]);
;  4276          4516  8
;  4277          4517  8                                PP [PP_W_LABEL_BYTES_TO_MOVE] =
;  4278          4518  8                                        MAX ( 0, .DCOLE + 1 -
;  4279          4519  8                                                 MAX (0, .DCOLS) );
;  4280          4520  8
;  4281          4521  8
;  4282          4522  8                                IF .PP [PP_W_COL] LEQ 0
;  4283          4523  8                                THEN
;  4284          4524  9                                    BEGIN       ! Using tail end of label
;  4285          4525  9                                    PP [PP_W_SRC_LABEL_OFF] =
;  4286          4526  9                                        .LDES [DSC$W_LENGTH] -
;  4287          4527  9                                        .PP [PP_W_LABEL_BYTES_TO_MOVE];
;  4288          4528  9                                    END ! Using tail end of label
```

```
;  4289          4529  8                                ELSE
;  4290          4530  9                                    BEGIN       ! Using front end of label
;  4291          4531  9                                    PP [PP_W_SRC_LABEL_OFF] = 0;
;  4292          4532  8                                    END;        ! Using front end of label
;  4293          4533  8                                PP [PP_W_DST_LABEL_OFF] = (.LOWER_ROW -1) *
;  4294          4534  8                                                         .WCB [WCB_W_NO_COLS] +
;  4295          4535  8                                                        MAX(0, .DCOLS - 1);
;  4296          4536  8
;  4297          4537  7                                END ;! Partially visible
;  4298          4538  6                            END;        ! Bottom row in buffer
;  4299          4539  5                        END;    ! Label in bottom row
;  4300          4540  5
;  4301          4541  5                    [SMG$K_LEFT]:
;  4302          4542  6                        BEGIN   !Label in left column
;  4303          4543  6                        IF .LEFT_COL GEQ 1
;  4304          4544  6                        THEN
;  4305          4545  7                            BEGIN       ! Left column in buffer
;  4306          4546  7                            LOCAL
;  4307          4547  7                                DROWS : SIGNED; ! Dest. row start
;  4308          4548  7
;  4309          4549  7                            DROWS =       .PP [PP_W_ROW] +
;  4310          4550  7                                          .DCB [DCB_W_LABEL_UNITS] - 2;
;  4311          4551  7
;  4312          4552  7                            IF .DROWS LEQ .WCB [WCB_W_NO_ROWS]
;  4313          4553  7                            THEN
;  4314          4554  8                                BEGIN ! Partially visible
;  4315          4555  8                                LOCAL
;  4316          4556  8                                    DROWE : SIGNED ; ! Dest. row end
;  4317          4557  8
;  4318          4558  8                                DROWE = MIN ( (.LDES [DSC$W_LENGTH] + .DROWS -1),
;  4319          4559  9                                        (.PP [PP_W_ROW] +
;  4320          4560  8                                         .DCB [DCB_W_NO_ROWS] ),
;  4321          4561  8                                         .WCB [WCB_W_NO_ROWS]);
;  4322          4562  8
;  4323          4563  8                                PP [PP_W_LABEL_BYTES_TO_MOVE] =
;  4324          4564  8                                        MAX ( 0, .DROWE + 1 -
;  4325          4565  8                                                 MAX (0, .DROWS) );
;  4326          4566  8
;  4327          4567  8                                IF .PP [PP_W_ROW] LEQ 0
;  4328          4568  8                                THEN
;  4329          4569  9                                    BEGIN       ! Using tail end of label
;  4330          4570  9                                    PP [PP_W_SRC_LABEL_OFF] =
;  4331          4571  9                                        .LDES [DSC$W_LENGTH] -
;  4332          4572  9                                        .PP [PP_W_LABEL_BYTES_TO_MOVE];
;  4333          4573  9                                    END ! Using tail end of label
;  4334          4574  8                                ELSE
;  4335          4575  9                                    BEGIN       ! Using front end of label
;  4336          4576  9                                    PP [PP_W_SRC_LABEL_OFF] = 0;
;  4337          4577  8                                    END;        ! Using front end of label
;  4338          4578  8
;  4339          4579  8                                PP [PP_W_DST_LABEL_OFF] = (.DROWS -1) *
;  4340          4580  8                                                         .WCB [WCB_W_NO_COLS] +
;  4341          4581  8                                                        MAX(0, .LEFT_COL - 1);
;  4342          4582  7                                END; ! Partially visible
;  4343          4583  6                            END;        ! Left column in buffer
;  4344          4584  5                        END;    ! Label in left column
;  4345          4585  5
```

```
; 4346     4586  5                    [SMG$K_RIGHT]:
; 4347     4587  6                        BEGIN   ! Label in right column
; 4348     4588  6                        IF .RIGHT_COL LEQ .WCB [WCB_W_NO_COLS]
; 4349     4589  6                        THEN
; 4350     4590  7                            BEGIN       ! Right column in buffer
; 4351     4591  7                            LOCAL
; 4352     4592  7                                DROWS : SIGNED; ! Dest. row start
; 4353     4593  7
; 4354     4594  7                            DROWS =         .PP [PP_W_ROW] +
; 4355     4595  7                                            .DCB [DCB_W_LABEL_UNITS] - 2;
; 4356     4596  7
; 4357     4597  7                            IF .DROWS LEQ .WCB [WCB_W_NO_ROWS]
; 4358     4598  7                            THEN
; 4359     4599  8                                BEGIN ! Partially visible
; 4360     4600  8                                LOCAL
; 4361     4601  8                                  DROWE : SIGNED ; ! Dest. row end
; 4362     4602  8
; 4363     4603  8                                DROWE = MIN ( (.LDES [DSC$W_LENGTH] + .DROWS -1),
; 4364     4604  9                                        (.PP [PP_W_ROW] +
; 4365     4605  8                                         .DCB [DCB_W_NO_ROWS] ),
; 4366     4606  8                                         .WCB [WCB_W_NO_ROWS]);
; 4367     4607  8
; 4368     4608  8                                PP [PP_W_LABEL_BYTES_TO_MOVE] =
; 4369     4609  8                                        MAX ( 0, .DROWE + 1 -
; 4370     4610  8                                                 MAX (0, .DROWS) );
; 4371     4611  8
; 4372     4612  8
; 4373     4613  8                                IF .PP [PP_W_ROW] LEQ 0
; 4374     4614  8                                THEN
; 4375     4615  9                                    BEGIN       ! Using tail end of label
; 4376     4616  9                                    PP [PP_W_SRC_LABEL_OFF] =
; 4377     4617  9                                        .LDES [DSC$W_LENGTH] -
; 4378     4618  9                                        .PP [PP_W_LABEL_BYTES_TO_MOVE];
; 4379     4619  9                                    END ! Using tail end of label
; 4380     4620  8                                ELSE
; 4381     4621  9                                    BEGIN       ! Using front end of label
; 4382     4622  9                                    PP [PP_W_SRC_LABEL_OFF] = 0;
; 4383     4623  8                                    END;        ! Using front end of label
; 4384     4624  8
; 4385     4625  8                                PP [PP_W_DST_LABEL_OFF] = (.DROWS -1) *
; 4386     4626  8                                                          .WCB [WCB_W_NO_COLS] +
; 4387     4627  8                                                          MAX(0, .RIGHT_COL - 1);
; 4388     4628  8
; 4389     4629  7                                END; ! Partially visible
; 4390     4630  6                            END;        ! Right column in buffer
; 4391     4631  5                        END;    ! Label in right column
; 4392     4632  5
; 4393     4633  5                    [OUTRANGE]:
; 4394     4634  5                        RETURN (SMG$_FATERRLIB);
; 4395     4635  5
; 4396     4636  5                TES;
; 4397     4637  4                END;    ! Label position computation
; 4398     4638  3            END;        ! Bordered display
; 4399     4639  2        END;    ! Overlap
; 4400     4640  2
; 4401     4641  2 !+
; 4402     4642  2 ! If the virtual display width matches the window control block width,
```

```
2 ! and the width of the data to be move during a mapping operation is
2 ! this same width, then both the source area and the destination are
2 ! contiguous sets of bytes and can be moved with a single CH$MOVE.
2 ! If not, they have to be moved a row at a time since that is how many
2 ! are piece-wise contiguous.
2 !-
2     PP [PP_V_CONTIG] = 0; ! Assume not contiguous until so proven.
2     IF .DCB [DCB_W_NO_COLS]    EQL .WCB [WCB_W_NO_COLS]     AND
2        .PP [PP_W_MOVE_LENGTH] EQL .DCB [DCB_W_NO_COLS]
2     THEN
2        PP [PP_V_CONTIG] = 1;    ! Contiguous
2
2     RETURN (SS$_NORMAL);
1     END;                    ! End of routine SMG$$CALC_PASTE_TRANSF
```

```
                                  0FFC 00000          .ENTRY  SMG$$CALC_PASTE_TRANSF, Save R2,R3,R4,R5,-  ; 4302
                                                              R6,R7,R8,R9,R10,R11
                      5E        2C  C2 00002          SUBL2   #44, SP
                      57    04  AC  D0 00005          MOVL    PP, R7                                      ; 4353
                      50    14  A7  D0 00009          MOVL    20(R7), PBCB
                      59    08  A0  D0 0000D          MOVL    8(PBCB), WCB                                ; 4354
                      52    10  A7  D0 00011          MOVL    16(R7), DCB                                 ; 4355
                 18   AE    24  A7  9E 00015          MOVAB   36(R7), 24(SP)                              ; 4360
                            18  BE  B4 0001A          CLRW    @24(SP)
                 14   AE    26  A7  9E 0001D          MOVAB   38(R7), 20(SP)                              ; 4361
                            14  BE  B4 00022          CLRW    @20(SP)
                 10   AE    28  A7  9E 00025          MOVAB   40(R7), 16(SP)                              ; 4362
                            10  BE  B4 0002A          CLRW    @16(SP)
                 0C   AE    18  A7  9E 0002D          MOVAB   24(R7), 12(SP)                              ; 4365
                      50        62  3C 00032          MOVZWL  (DCB), R0
                      51    0C  BE  32 00035          CVTWL   @12(SP), R1
                      50        51  C0 00039          ADDL2   R1, R0
        24   AE       50        01  A3 0003C          SUBW3   #1, R0, TEMP
                 26   AE    02  A2  B0 00041          MOVW    2(DCB), TEMP+2                              ; 4366
                 08   AE    1A  A7  9E 00046          MOVAB   26(R7), 8(SP)                               ; 4367
                      50    04  A2  3C 0004B          MOVZWL  4(DCB), R0
                      51    08  BE  32 0004F          CVTWL   @8(SP), R1
                      50        51  C0 00053          ADDL2   R1, R0
        28   AE       50        01  A3 00056          SUBW3   #1, R0, TEMP+4
                 04   AE    06  A2  3C 0005B          MOVZWL  6(DCB), 4(SP)                               ; 4368
                 2A   AE    04  AE  B0 00060          MOVW    4(SP), TEMP+6
                            1C  AE  9F 00065          PUSHAB  OVERLAP                                     ; 4376
                            28  AE  9F 00068          PUSHAB  TEMP
                                59  DD 0006B          PUSHL   WCB
          00000000G   00        03  FB 0006D          CALLS   #3, SMG$$OCCLUDE
                      06        50  E8 00074          BLBS    R0, 2$
                            1C  A7  B4 00077          CLRW    28(R7)                                      ; 4382
                              0258  31 0007A 1$:      BRW     42$                                         ; 4375
                 1C   A7    1E  AE  B0 0007D 2$:      MOVW    OVERLAP+2, 28(R7)                           ; 4396
                 22   A7    22  AE  B0 00082          MOVW    OVERLAP+6, 34(R7)                           ; 4397
                 2F   A7    1C  AE  B0 00087          MOVW    OVERLAP, 47(R7)                             ; 4399
                      50    1C  AE  3C 0008C          MOVZWL  OVERLAP, R0                                 ; 4401
                      51    1E  AE  3C 00090          MOVZWL  OVERLAP+2, R1
```

```
                                50            51  C0 00094          ADDL2   R1, R0                          ;
          31    A7              50            01  A3 00097          SUBW3   #1, R0, 49(R7)                  ;
                                53      20    AE  3C 0009C          MOVZWL  OVERLAP+4, R3                   ; 4403
                        33      A7            53  B0 000A0          MOVW    R3, 51(R7)                      ;
                                50      22    AE  3C 000A4          MOVZWL  OVERLAP+6, R0                   ; 4405
                                51      FF  A340  9E 000A8          MOVAB   -1(R3)[R0], R1                  ;
                        35      A7            51  B0 000AD          MOVW    R1, 53(R7)                      ;
                                50      1E    AE  3C 000B1          MOVZWL  OVERLAP+2, R0                   ; 4408
                                51      22    AE  3C 000B5          MOVZWL  OVERLAP+6, R1                   ;
          2B    A7              50            51  C5 000B9          MULL3   R1, R0, 43(R7)                  ;
                                51      1C    AE  3C 000BE          MOVZWL  OVERLAP, R1                     ;
                                50      0C    BE  32 000C2          CVTWL   @12(SP), R0                     ; 4410
                                51            50  C2 000C6          SUBL2   R0, R1                          ;
                                50      08    BE  32 000C9          CVTWL   @8(SP), R0                      ; 4411
                50              53            50  C3 000CD          SUBL3   R0, R3, R0                      ;
                                51      04    AE  C4 000D1          MULL2   4(SP), R1                       ; 4413
                                54          8041  9E 000D5          MOVAB   (DCB_START_COL)+[R1], R4        ; 4414
                        1E      A7            54  B0 000D9          MOVW    R4, 30(R7)                      ;
                                50      1C    AE  3C 000DD          MOVZWL  OVERLAP, R0                     ; 4416
                                              50  D7 000E1          DECL    R0                              ;
                                5A      06    A9  3C 000E3          MOVZWL  6(WCB), R10                     ; 4417
                                50            5A  C4 000E7          MULL2   R10, R0                         ;
                                51      FF  A340  9E 000EA          MOVAB   -1(R3)[R0], R1                  ; 4418
                        20      A7            51  B0 000EF          MOVW    R1, 32(R7)                      ;
                                83      2F    A2  E9 000F3          BLBC    47(DCB), 1$                     ; 4420
                                50      08    A2  9E 000F7          MOVAB   8(R2), LDES                     ; 4432
                                55      0C    BE  32 000FB          CVTWL   @12(SP), UPPER_ROW              ; 4439
                                              55  D7 000FF          DECL    UPPER_ROW                       ;
                                5B      0C    BE  32 00101          CVTWL   @12(SP), R11                    ; 4440
                                51      02    A2  3C 00105          MOVZWL  2(DCB), R1                      ;
                                5B            51  C0 00109          ADDL2   R1, R11                         ;
                                51            5B  D0 0010C          MOVL    R11, LOWER_ROW                  ;
                                54      08    BE  32 0010F          CVTWL   @8(SP), LEFT_COL                ; 4441
                                              54  D7 00113          DECL    LEFT_COL                        ;
                                58      08    BE  32 00115          CVTWL   @8(SP), R8                      ; 4442
                                58      04    AE  C0 00119          ADDL2   4(SP), R8                       ;
                                53            58  D0 0011D          MOVL    R8, RIGHT_COL                   ;
                                56            60  3C 00120          MOVZWL  (LDES), R6                      ; 4444
                                              74  13 00123          BEQL    12$                             ;
                03              00      31    A2  8F 00125          CASEB   49(DCB), #0, #3                 ; 4447
  0141        00D6            0071          0010     0012A 3$:      .WORD   4$-3$,-                         ;
                                                                            13$-3$,-                        ;
                                                                            23$-3$,-                        ;
                                                                            33$-3$                          ;
                                50 00000000G  8F  D0 00132          MOVL    #SMG$_FATERRLIB, R0             ; 4634
                                              04  00139             RET                                     ;
                                              55  D5 0013A 4$:      TSTL    UPPER_ROW                       ; 4451
                                              5B  15 0013C          BLEQ    12$                             ;
                                50      08    BE  32 0013E          CVTWL   @8(SP), R0                      ; 4458
                                6E      2C    A2  3C 00142          MOVZWL  44(DCB), (SP)                   ;
                                50            6E  C0 00146          ADDL2   (SP), R0                        ;
                                50            02  C2 00149          SUBL2   #2, DCOLS                       ;
                                5A            50  D1 0014C          CMPL    DCOLS, R10                      ; 4460
                                              63  14 0014F          BGTR    14$                             ;
                                52      FF  A046  9E 00151          MOVAB   -1(DCOLS)[R6], R2               ; 4466
                                58            52  D1 00156          CMPL    R2, R8                          ; 4467
                                              03  15 00159          BLEQ    5$                              ;
```

```
                     52         58  D0 0015B        MOVL    R8, R2                              ;
                     5A         52  D1 0015E 5$:    CMPL    R2, R10                             ; 4469
                                03  15 00161        BLEQ    6$                                  ;
                     52         5A  D0 00163        MOVL    R10, R2                             ;
                     58         50  D0 00166 6$:    MOVL    DCOLS, R8                           ; 4473
                                02  18 00169        BGEQ    7$                                  ;
                                58  D4 0016B        CLRL    R8                                  ;
                     52         58  C2 0016D 7$:    SUBL2   R8, R2                              ;
                                52  D6 00170        INCL    R2                                  ; 4472
                                02  18 00172        BGEQ    8$                                  ;
                                52  D4 00174        CLRL    R2                                  ;
                18   BE         52  B0 00176 8$:    MOVW    R2, @24(SP)                         ;
                          08    BE  B5 0017A        TSTW    @8(SP)                              ; 4475
                          08        14 0017D        BGTR    9$                                  ;
           14   BE   56   18    BE  A3 0017F        SUBW3   @24(SP), R6, @20(SP)                ; 4480
                                03  11 00185        BRB     10$                                 ; 4475
                          14    BE  B4 00187 9$:    CLRW    @20(SP)                             ; 4484
                                55  D7 0018A 10$:   DECL    R5                                  ; 4487
                     55         5A  C4 0018C        MULL2   R10, R5                             ; 4488
                     02         50  F4 0018F        SOBGEQ  R0, 11$                             ; 4489
                                50  D4 00192        CLRL    R0                                  ;
           10   BE   55         50  A1 00194 11$:   ADDW3   R0, R5, @16(SP)                     ;
                                63  11 00199 12$:   BRB     22$                                 ; 4447
      51   02   A9   10         00  ED 0019B 13$:   CMPZV   #0, #16, 2(WCB), LOWER_ROW          ; 4497
                                76  19 001A1        BLSS    24$                                 ;
                     50   08    BE  32 001A3        CVTWL   @8(SP), R0                          ; 4504
                     55   2C    A2  3C 001A7        MOVZWL  44(DCB), R5                         ;
                     50         55  C0 001AB        ADDL2   R5, R0                              ;
                     50         02  C2 001AE        SUBL2   #2, DCOLS                           ;
                     5A         50  D1 001B1        CMPL    DCOLS, R10                          ; 4506
                                48  14 001B4 14$:   BGTR    22$                                 ;
                     52   FF A046   9E 001B6        MOVAB   -1(DCOLS)[R6], R2                   ; 4512
                     58         52  D1 001BB        CMPL    R2, R8                              ; 4513
                                03  15 001BE        BLEQ    15$                                 ;
                     52         58  D0 001C0        MOVL    R8, R2                              ;
                     5A         52  D1 001C3 15$:   CMPL    R2, R10                             ; 4515
                                03  15 001C6        BLEQ    16$                                 ;
                     52         5A  D0 001C8        MOVL    R10, R2                             ;
                     55         50  D0 001CB 16$:   MOVL    DCOLS, R5                           ; 4519
                                02  18 001CE        BGEQ    17$                                 ;
                                55  D4 001D0        CLRL    R5                                  ;
                     52         55  C2 001D2 17$:   SUBL2   R5, R2                              ;
                                52  D6 001D5        INCL    R2                                  ; 4518
                                02  18 001D7        BGEQ    18$                                 ;
                                52  D4 001D9        CLRL    R2                                  ;
                18   BE         52  B0 001DB 18$:   MOVW    R2, @24(SP)                         ;
                          08    BE  B5 001DF        TSTW    @8(SP)                              ; 4522
                                08  14 001E2        BGTR    19$                                 ;
           14   BE   56   18    BE  A3 001E4        SUBW3   @24(SP), R6, @20(SP)                ; 4527
                                03  11 001EA        BRB     20$                                 ; 4522
                          14    BE  B4 001EC 19$:   CLRW    @20(SP)                             ; 4531
                                51  D7 001EF 20$:   DECL    R1                                  ; 4533
                     51         5A  C4 001F1        MULL2   R10, R1                             ; 4534
                     02         50  F4 001F4        SOBGEQ  R0, 21$                             ; 4535
                                50  D4 001F7        CLRL    R0                                  ;
           10   BE   51         50  A1 001F9 21$:   ADDW3   R0, R1, @16(SP)                     ;
                                69  11 001FE 22$:   BRB     32$                                 ; 4447
```

```
                                                54  D5 00200 23$:   TSTL    LEFT_COL                                ; 4543
                                                65  15 00202        BLEQ    32$                                     ;
                                          51    0C  BE 32 00204        CVTWL   @12(SP), R1                             ; 4550
                                          50    2C  A2 3C 00208        MOVZWL  44(DCB), R0                             ;
                                          51        50 C0 0020C        ADDL2   R0, R1                                  ;
                                          50    FE  A1 9E 0020F        MOVAB   -2(R1), DROWS                           ;
     50   02   A9                        10        00 ED 00213        CMPZV   #0, #16, 2(WCB), DROWS                  ; 4552
                                                6A  19 00219 24$:       BLSS    34$                                     ;
                                          51    FF A046 9E 0021B        MOVAB   -1(DROWS)[R6], R1                       ; 4558
                                          5B        51 D1 00220        CMPL    R1, R11                                 ; 4559
                                                    03 15 00223        BLEQ    25$                                     ;
                                          51        5B D0 00225        MOVL    R11, R1                                 ;
     51   02   A9                        10        00 ED 00228 25$:   CMPZV   #0, #16, 2(WCB), R1                     ; 4561
                                                    04 18 0022E        BGEQ    26$                                     ;
                                          51    02  A9 3C 00230        MOVZWL  2(WCB), R1                              ;
                                          51        50 7D 00234 26$:   MOVQ    DROWS, R1                               ; 4565
                                                    51 D5 00237        TSTL    R1                                      ;
                                                    02 18 00239        BGEQ    27$                                     ;
                                                    51 D4 0023B        CLRL    R1                                      ;
                                          52        51 C2 0023D 27$:   SUBL2   R1, R2                                  ;
                                                    52 D6 00240        INCL    R2                                      ; 4564
                                                    02 18 00242        BGEQ    28$                                     ;
                                                    52 D4 00244        CLRL    R2                                      ;
                          18   BE                   52 B0 00246 28$:   MOVW    R2, @24(SP)                             ;
                                          0C    BE  B5 0024A        TSTW    @12(SP)                                 ; 4567
                                                08  14 0024D        BGTR    29$                                     ;
               14   BE                   56   18   BE A3 0024F        SUBW3   @24(SP), R6, @20(SP)                    ; 4572
                                                03  11 00255        BRB     30$                                     ; 4567
                                          14    BE  B4 00257 29$:   CLRW    @20(SP)                                 ; 4576
                                                50  D7 0025A 30$:   DECL    R0                                      ; 4579
                                          50        5A C4 0025C        MULL2   R10, R0                                 ; 4580
                                          02        54 F4 0025F        SOBGEQ  R4, 31$                                 ; 4581
                                                    54 D4 00262        CLRL    R4                                      ;
          10   BE                        50        54 A1 00264 31$:   ADDW3   R4, R0, @16(SP)                         ;
                                                6A  11 00269 32$:   BRB     42$                                     ; 4447
                                          5A        53 D1 0026B 33$:   CMPL    RIGHT_COL, R10                          ; 4588
                                                65  14 0026E        BGTR    42$                                     ;
                                          51    0C  BE 32 00270        CVTWL   @12(SP), R1                             ; 4595
                                          50    2C  A2 3C 00274        MOVZWL  44(DCB), R0                             ;
                                          51        50 C0 00278        ADDL2   R0, R1                                  ;
                                          50    FE  A1 9E 0027B        MOVAB   -2(R1), DROWS                           ;
     50   02   A9                        10        00 ED 0027F        CMPZV   #0, #16, 2(WCB), DROWS                  ; 4597
                                                4E  19 00285 34$:       BLSS    42$                                     ;
                                          51    FF A046 9E 00287        MOVAB   -1(DROWS)[R6], R1                       ; 4603
                                          5B        51 D1 0028C        CMPL    R1, R11                                 ; 460
                                                    03 15 0028F        BLEQ    35$                                     ;
                                          51        5B D0 00291        MOVL    R11, R1                                 ;
     51   02   A9                        10        00 ED 00294 35$:   CMPZV   #0, #16, 2(WCB), R1                     ; 4606
                                                    04 18 0029A        BGEQ    36$                                     ;
                                          51    02  A9 3C 0029C        MOVZWL  2(WCB), R1                              ;
                                          51        50 7D 002A0 36$:   MOVQ    DROWS, R1                               ; 4610
                                                    51 D5 002A3        TSTL    R1                                      ;
                                                    02 18 002A5        BGEQ    37$                                     ;
                                                    51 D4 002A7        CLRL    R1                                      ;
                                          52        51 C2 002A9 37$:   SUBL2   R1, R2                                  ;
                                                    52 D6 002AC        INCL    R2                                      ; 4609
                                                    02 18 002AE        BGEQ    38$                                     ;
```

```
                                  52  D4 002B0          CLRL    R2                                ;
           18   BE                52  B0 002B2 38$:     MOVW    R2, @24(SP)                       ;
                        0C        BE  B5 002B6          TSTW    @12(SP)                           ; 4613
                                  08  14 002B9          BGTR    39$                               ;
  14   BE         56    18        BE  A3 002BB          SUBW3   @24(SP), R6, @20(SP)              ; 4618
                                  03  11 002C1          BRB     40$                               ; 4613
                        14        BE  B4 002C3 39$:     CLRW    @20(SP)                           ; 4622
                                  50  D7 002C6 40$:     DECL    R0                                ; 4625
                  50              5A  C4 002C8          MULL2   R10, R0                           ; 4626
                  02              53  F4 002CB          SOBGEQ  R3, 41$                           ; 4627
                                  53  D4 002CE          CLRL    R3                                ;
  10   BE         50              53  A1 002D0 41$:     ADDW3   R3, R0, @16(SP)                   ;
           2A     A7              02  8A 002D5 42$:     BICB2   #2, 42(R7)                        ; 4649
           04     AE    06        A9  B1 002D9          CMPW    6(WCB), 4(SP)                     ; 4650
                                  0B  12 002DE          BNEQ    43$                               ;
           04     AE    22        A7  B1 002E0          CMPW    34(R7), 4(SP)                     ; 4651
                                  04  12 002E5          BNEQ    43$                               ;
           2A     A7              02  88 002E7          BISB2   #2, 42(R7)                        ; 4653
                  50              01  D0 002EB 43$:     MOVL    #1, R0                            ; 4655
                                      04 002EE          RET                                       ; 4656

; Routine Size:  751 bytes,    Routine Base:  _SMG$CODE + 167D

; 4417         4657  1 !<BLF/PAGE>
```

```
;  4419     4658  1 %SBTTL 'SMG$$CHECK_OCCLUSION - Check pastings for occlusion'
;  4420     4659  1 GLOBAL ROUTINE SMG$$CHECK_OCCLUSION (
;  4421     4660  1                                      PBCB : REF $PBCB_DECL
;  4422     4661  1                                      ) =
;  4423     4662  1 !++
;  4424     4663  1 ! FUNCTIONAL DESCRIPTION:
;  4425     4664  1 !
;  4426     4665  1 !       This procedure checks the overlap in all the pastings to
;  4427     4666  1 !       a given pasteboard -- setting a bit in the respective pasting
;  4428     4667  1 !       packets to record whether each pasting is occluded by
;  4429     4668  1 !       higher (more-recently pasted) pastings.  This procedure is
;  4430     4669  1 !       invoked whenever there is a change in the pasting order or
;  4431     4670  1 !       pasting position of any virtual display.
;  4432     4671  1 !       If we can determine at pasting time that a particular virtual
;  4433     4672  1 !       display (as pasted) and with a given succession of higher-pasted
;  4434     4673  1 !       virtual displays is not occluded by any of them,
;  4435     4674  1 !       we can make mapping run faster since a change to a virtual
;  4436     4675  1 !       display which is not occluded causes only the changed virtual
;  4437     4676  1 !       display to be remapped.  Higher pasted virtual displays do not
;  4438     4677  1 !       have to be remapped since it is known that they do not occlude
;  4439     4678  1 !       the changed one.
;  4440     4679  1 !
;  4441     4680  1 ! CALLING SEQUENCE:
;  4442     4681  1 !
;  4443     4682  1 !       ret_status.wlc.v = SMG$$CHECK_OCCLUSION ( PBCB.rab.r)
;  4444     4683  1 !
;  4445     4684  1 ! FORMAL PARAMETERS:
;  4446     4685  1 !
;  4447     4686  1 !       PBCB.rab.r              Address of pasteboard control block
;  4448     4687  1 !                               which has something new or different
;  4449     4688  1 !                               pasted to it.
;  4450     4689  1 !
;  4451     4690  1 ! IMPLICIT INPUTS:
;  4452     4691  1 !
;  4453     4692  1 !       NONE
;  4454     4693  1 !
;  4455     4694  1 ! IMPLICIT OUTPUTS:
;  4456     4695  1 !
;  4457     4696  1 !       NONE
;  4458     4697  1 !
;  4459     4698  1 ! COMPLETION STATUS:
;  4460     4699  1 !
;  4461     4700  1 !       SS$_NORMAL      Normal successful completion
;  4462     4701  1 !
;  4463     4702  1 ! SIDE EFFECTS:
;  4464     4703  1 !
;  4465     4704  1 !       NONE
;  4466     4705  1 !--
;  4467     4706  1
;  4468     4707  2     BEGIN
;  4469     4708  2     LOCAL
;  4470     4709  2         THIS_PP : REF $PP_DECL,
;  4471     4710  2                   ! Addr of pasting packet for upper-most pasted
;  4472     4711  2                   ! virtual display.
;  4473     4712  2
;  4474     4713  2         THIS_Q_HEAD : REF BLOCK [,BYTE];
;  4475     4714  2                   ! Addr of 2 longwords that form queue header in
```

```
;  4476     4715  2                          ! PP currently under inspection.
;  4477     4716  2
;  4478     4717  2              !+
;  4479     4718  2              ! To initialize for the rest of the algorithm, run through whole pasting
;  4480     4719  2              ! list marking all packets not occluded.
;  4481     4720  2              !-
;  4482     4721  2                  THIS_Q_HEAD = .PBCB [PBCB_A_PP_NEXT];       ! 1st (more recent pasting)
;  4483     4722  2                  WHILE .THIS_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
;  4484     4723  2                  DO
;  4485     4724  2                      BEGIN   ! Init. pass
;  4486     4725  2                      THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET; ! To top of packet
;  4487     4726  2                      THIS_PP [ PP_V_OCCLUDED] = 0;   ! Init to not occluded
;  4488     4727  2                      THIS_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];    ! To queue header in
;  4489     4728  2                                                                  !  next packet
;  4490     4729  2                      END;    ! Init. pass
;  4491     4730  2
;  4492     4731  2                  THIS_Q_HEAD = .PBCB [PBCB_A_PP_NEXT];
;  4493     4732  2                  THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4494     4733  2                          .
;  4495     4734  2              !+
;  4496     4735  2              ! Loop for all pasting packets starting with most-recently pasted one.
;  4497     4736  2              !-
;  4498     4737  2                  WHILE .THIS_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
;  4499     4738  2                  DO
;  4500     4739  2                      BEGIN   ! For all displays from top to bottom
;  4501     4740  2                      LOCAL
;  4502     4741  2                          NEXT_PP         : REF $PP_DECL,
;  4503     4742  2                                            ! Addr of pasting packet currently under
;  4504     4743  2                                            ! inspection.
;  4505     4744  2
;  4506     4745  2                          NEXT_PP_Q_HEAD : REF BLOCK [,BYTE],
;  4507     4746  2                                            ! Addr of 2 longwords that form queue
;  4508     4747  2                                            ! header in PP currently under
;  4509     4748  2                                            ! inspection.
;  4510     4749  2
;  4511     4750  2                          TEMP_THIS : BLOCK [8,BYTE],
;  4512     4751  2                                            ! Area of projection of THIS virtual
;  4513     4752  2                                            ! display on pasteboard
;  4514     4753  2
;  4515     4754  2                          TEMP_NEXT : BLOCK [8,BYTE],
;  4516     4755  2                                            ! Area of projection of NEXT virtual
;  4517     4756  2                                            ! display on pasteboard
;  4518     4757  2
;  4519     4758  2                          THIS_DCB : REF BLOCK [,BYTE];
;  4520     4759  2                                            ! Addr of virtual display currently
;  4521     4760  2                                            ! under inspection.
;  4522     4761  2
;  4523     4762  2                  !+
;  4524     4763  2                  ! Recalculate pasting packet address and DCB address for this
;  4525     4764  2                  ! iteration.
;  4526     4765  2                  !-
;  4527     4766  2                  THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4528     4767  2                  THIS_DCB = .THIS_PP [PP_A_DCB_ADDR];
;  4529     4768  2
;  4530     4769  2                  !+
;  4531     4770  2                  ! It is safe to assume that there is at least one virtual
;  4532     4771  2                  ! display pasted to this pasteboard -- but there may not be more
```

```
;  4533          4772  3          ! than one.  Be careful about reaching ahead to a packet that
;  4534          4773  3          ! may not be a packet.  If doesn't exist, pointer will be
;  4535          4774  3          ! pointing back into PBCB -- and inner loop will not be
;  4536          4775  3          ! executed.
;  4537          4776  3          !-
;  4538          4777  3          NEXT_PP_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];
;  4539          4778  3
;  4540          4779  3          IF .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
;  4541          4780  3          THEN
;  4542          4781  4              BEGIN       ! NEXT exists
;  4543          4782  4              NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4544          4783  4              !+
;  4545          4784  4              ! Form a representation of the projection of THIS virtual
;  4546          4785  4              ! display onto pasteboard coordinate system.
;  4547          4786  4              !-
;  4548          4787  4              TEMP_THIS [DCB_W_ROW_START] = .THIS_DCB [DCB_W_ROW_START] +
;  4549          4788  4                                            .THIS_PP [PP_W_ROW] - 1;
;  4550          4789  4              TEMP_THIS [DCB_W_NO_ROWS]   = .THIS_DCB [DCB_W_NO_ROWS];
;  4551          4790  4              TEMP_THIS [DCB_W_COL_START] = .THIS_DCB [DCB_W_COL_START] +
;  4552          4791  4                                            .THIS_PP [PP_W_COL] - 1;
;  4553          4792  4              TEMP_THIS [DCB_W_NO_COLS]   = .THIS_DCB [DCB_W_NO_COLS];
;  4554          4793  4
;  4555          4794  4              !+
;  4556          4795  4              ! If this virtual display is bordered, its projection is
;  4557          4796  4              ! bigger than if it were not.  Adjust its projection
;  4558          4797  4              ! representation.
;  4559          4798  4              !-
;  4560          4799  4              IF .THIS_DCB [DCB_V_BORDERED]
;  4561          4800  4              THEN
;  4562          4801  5                  BEGIN   ! Border adjustment
;  4563          4802  5                  TEMP_THIS [DCB_W_ROW_START] = .TEMP_THIS [DCB_W_ROW_START] - 1;
;  4564          4803  5                  TEMP_THIS [DCB_W_NO_ROWS]   = .TEMP_THIS [DCB_W_NO_ROWS] + 2;
;  4565          4804  5                  TEMP_THIS [DCB_W_COL_START] = .TEMP_THIS [DCB_W_COL_START] - 1;
;  4566          4805  5                  TEMP_THIS [DCB_W_NO_COLS]   = .TEMP_THIS [DCB_W_NO_COLS] + 2;
;  4567          4806  4                  END;    ! Border adjustment
;  4568          4807  4
;  4569          4808  3              END;        ! Next exists
;  4570          4809  3
;  4571          4810  3          WHILE .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
;  4572          4811  3          DO
;  4573          4812  4              BEGIN       ! For all displays from current to bottom
;  4574          4813  4              LOCAL
;  4575          4814  4                  NEXT_DCB : REF $DCB_DECL,
;  4576          4815  4                                 ! Addr of DCB associated with NEXT_PP
;  4577          4816  4                  OVERLAP : BLOCK [8,BYTE];
;  4578          4817  4                                 ! Returned by SMG$$OCCLUDE, but not
;  4579          4818  4                                 ! used in this context
;  4580          4819  4
;  4581          4820  4              NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4582          4821  4              NEXT_DCB = .NEXT_PP [PP_A_DCB_ADDR];
;  4583          4822  4
;  4584          4823  4              !+
;  4585          4824  4              ! Form a representation of the projection of NEXT virtual
;  4586          4825  4              ! display onto pasteboard coordinate system.
;  4587          4826  4              !-
;  4588          4827  4              TEMP_NEXT [DCB_W_ROW_START] = .NEXT_DCB [DCB_W_ROW_START] +
;  4589          4828  4                                            .NEXT_PP [PP_W_ROW] - 1;
```

```
;  4590          4829  4                TEMP_NEXT [DCB_W_NO_ROWS]   = .NEXT_DCB [DCB_W_NO_ROWS];
;  4591          4830  4                TEMP_NEXT [DCB_W_COL_START] = .NEXT_DCB [DCB_W_COL_START] +
;  4592          4831  4                                              .NEXT_PP [PP_W_COL] - 1;
;  4593          4832  4                TEMP_NEXT [DCB_W_NO_COLS]   = .NEXT_DCB [DCB_W_NO_COLS];
;  4594          4833  4
;  4595          4834  4                !+
;  4596          4835  4                ! If this next virtual display is bordered, its projection is
;  4597          4836  4                ! bigger than if it were not.  Adjust its projection
;  4598          4837  4                ! representation.
;  4599          4838  4                !-
;  4600          4839  4                IF .NEXT_DCB [DCB_V_BORDERED]
;  4601          4840  4                THEN
;  4602          4841  5                    BEGIN   ! Border adjustment
;  4603          4842  5                    TEMP_NEXT [DCB_W_ROW_START] = .TEMP_NEXT [DCB_W_ROW_START] - 1;
;  4604          4843  5                    TEMP_NEXT [DCB_W_NO_ROWS]   = .TEMP_NEXT [DCB_W_NO_ROWS] + 2;
;  4605          4844  5                    TEMP_NEXT [DCB_W_COL_START] = .TEMP_NEXT [DCB_W_COL_START] - 1;
;  4606          4845  5                    TEMP_NEXT [DCB_W_NO_COLS]   = .TEMP_NEXT [DCB_W_NO_COLS] + 2;
;  4607          4846  4                    END;    ! Border adjustment
;  4608          4847  4
;  4609          4848  4                !+
;  4610          4849  4                ! Check to see if THIS virtual display occludes NEXT
;  4611          4850  4                ! vitual display and if so set occlusion bit of NEXT.
;  4612          4851  4                !-
;  4613          4852  4                IF SMG$$OCCLUDE ( TEMP_NEXT, TEMP_THIS, OVERLAP)
;  4614          4853  4                THEN
;  4615          4854  4                    NEXT_PP [PP_V_OCCLUDED] = 1;
;  4616          4855  4
;  4617          4856  4                !+
;  4618          4857  4                ! Walk chain in direction of earlier pasted packets.
;  4619          4858  4                !-
;  4620          4859  4                NEXT_PP_Q_HEAD = .NEXT_PP [PP_A_NEXT_PBCB];
;  4621          4860  4
;  4622          4861  3                END;        ! For all displays from current to bottom
;  4623          4862  3
;  4624          4863  3            !+
;  4625          4864  3            ! Walk chain in direction of earlier pasted packets.
;  4626          4865  3            !-
;  4627          4866  3            THIS_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];
;  4628          4867  3
;  4629          4868  2            END;    ! For all displays from top to bottom
;  4630          4869  2
;  4631          4870  2        RETURN (SS$_NORMAL);
;  4632          4871  2
;  4633          4872  1        END;                ! End of routine SMG$$CHECK_OCCLUSION


                                 00FC 00000          .ENTRY  SMG$$CHECK_OCCLUSION, Save R2,R3,R4,R5,R6,- ; 4659
                                                             R7                                           ;
                           5E    18   C2 00002              SUBL2   #24, SP                              ;
                           56 04 AC   D0 00005              MOVL    PBCB, R6                             ; 4721
                           54    66   D0 00009              MOVL    (R6), THIS_Q_HEAD                    ;
                           56    54   D1 0000C 1$:          CMPL    THIS_Q_HEAD, R6                      ; 4722
                                 0E   13 0000F              BEQL    2$                                   ;
                           52 F8 A4   9E 00011              MOVAB   -8(R4), THIS_PP                      ; 4725
```

```
           2A   A2          01  8A 00015       BICB2    #1, 42(THIS_PP)                        ; 4726
                54     08   A2  D0 00019       MOVL     8(THIS_PP), THIS_Q_HEAD                ; 4727
                            ED  11 0001D       BRB      1$                                     ; 4722
                54          66  D0 0001F 2$:   MOVL     (R6), THIS_Q_HEAD                      ; 4731
                52     F8   A4  9E 00022       MOVAB    -8(R4), THIS_PP                        ; 4732
                56          54  D1 00026 3$:   CMPL     THIS_Q_HEAD, R6                        ; 4737
                            03  12 00029       BNEQ     4$
                          00BB  31 0002B       BRW      9$
                52     F8   A4  9E 0002E 4$:   MOVAB    -8(R4), THIS_PP                        ; 4766
                50     10   A2  D0 00032       MOVL     16(THIS_PP), THIS_DCB                  ; 4767
                55     08   A2  D0 00036       MOVL     8(THIS_PP), NEXT_PP_Q_HEAD             ; 4777
                56          55  D1 0003A       CMPL     NEXT_PP_Q_HEAD, R6                     ; 4779
                            3F  13 0003D       BEQL     5$
                53     F8   A5  9E 0003F       MOVAB    -8(R5), NEXT_PP                        ; 4782
                51          60  3C 00043       MOVZWL   (THIS_DCB), R1                         ; 4788
                57     18   A2  32 00046       CVTWL    24(THIS_PP), R7
                51          57  C0 0004A       ADDL2    R7, R1
  10   AE       51          01  A3 0004D       SUBW3    #1, R1, TEMP_THIS
           12   AE     02   A0  B0 00052       MOVW     2(THIS_DCB), TEMP_THIS+2               ; 4789
                51     04   A0  3C 00057       MOVZWL   4(THIS_DCB), R1                        ; 4791
                57     1A   A2  32 0005B       CVTWL    26(THIS_PP), R7
                51          57  C0 0005F       ADDL2    R7, R1
  14   AE       51          01  A3 00062       SUBW3    #1, R1, TEMP_THIS+4
           16   AE     06   A0  B0 00067       MOVW     6(THIS_DCB), TEMP_THIS+6               ; 4792
                0E     2F   A0  E9 0006C       BLBC     47(THIS_DCB), 5$                       ; 4799
                       10   AE  B7 00070       DECW     TEMP_THIS                              ; 4802
           12   AE          02  A0 00073       ADDW2    #2, TEMP_THIS+2                        ; 4803
                       14   AE  B7 00077       DECW     TEMP_THIS+4                            ; 4804
           16   AE          02  A0 0007A       ADDW2    #2, TEMP_THIS+6                        ; 4805
                56          55  D1 0007E 5$:   CMPL     NEXT_PP_Q_HEAD, R6                     ; 4810
                            5F  13 00081       BEQL     8$
                53     F8   A5  9E 00083       MOVAB    -8(R5), NEXT_PP                        ; 4820
                50     10   A3  D0 00087       MOVL     16(NEXT_PP), NEXT_DCB                  ; 4821
                51          60  3C 0008B       MOVZWL   (NEXT_DCB), R1                         ; 4828
                57     18   A3  32 0008E       CVTWL    24(NEXT_PP), R7
                51          57  C0 00092       ADDL2    R7, R1
  08   AE       51          01  A3 00095       SUBW3    #1, R1, TEMP_NEXT
           0A   AE     02   A0  B0 0009A       MOVW     2(NEXT_DCB), TEMP_NEXT+2               ; 4829
                51     04   A0  3C 0009F       MOVZWL   4(NEXT_DCB), R1                        ; 4831
                57     1A   A3  32 000A3       CVTWL    26(NEXT_PP), R7
                51          57  C0 000A7       ADDL2    R7, R1
  0C   AE       51          01  A3 000AA       SUBW3    #1, R1, TEMP_NEXT+4
           0E   AE     06   A0  B0 000AF       MOVW     6(NEXT_DCB), TEMP_NEXT+6               ; 4832
                0E     2F   A0  E9 000B4       BLBC     47(NEXT_DCB), 6$                       ; 4839
                       08   AE  B7 000B8       DECW     TEMP_NEXT                              ; 4842
           0A   AE          02  A0 000BB       ADDW2    #2, TEMP_NEXT+2                        ; 4843
                       0C   AE  B7 000BF       DECW     TEMP_NEXT+4                            ; 4844
           0E   AE          02  A0 000C2       ADDW2    #2, TEMP_NEXT+6                        ; 4845
                            5E  DD 000C6 6$:   PUSHL    SP                                     ; 4852
                       14   AE  9F 000C8       PUSHAB   TEMP_THIS
                       10   AE  9F 000CB       PUSHAB   TEMP_NEXT
    00000000G   00          03  FB 000CE       CALLS    #3, SMG$$OCCLUDE
                04          50  E9 000D5       BLBC     R0, 7$
           2A   A3          01  88 000D8       BISB2    #1, 42(NEXT_PP)                        ; 4854
                55     08   A3  D0 000DC 7$:   MOVL     8(NEXT_PP), NEXT_PP_Q_HEAD             ; 4859
                            9C  11 000E0       BRB      5$                                     ; 4810
                54     08   A2  D0 000E2 8$:   MOVL     8(THIS_PP), THIS_Q_HEAD                ; 4866
```

```
                                    FF3D  31 000E6          BRW     3$                                      ; 4737
                              50      01  D0 000E9 9$:      MOVL    #1, R0                                  ; 4870
                                          04 000EC          RET                                             ; 4872

; Routine Size:  237 bytes,    Routine Base:  _SMG$CODE + 196C

; 4634          4873  1 !<BLF/PAGE>
```

```
;  4636          4874  1 %SBTTL 'SMG$$CHECK_OCCLUSION_FIRST - Check pastings for occlusion'
;  4637          4875  1 GLOBAL ROUTINE SMG$$CHECK_OCCLUSION_FIRST (
;  4638          4876  1                                         PBCB : REF $PBCB_DECL
;  4639          4877  1                                         ) =
;  4640          4878  1 !++
;  4641          4879  1 ! FUNCTIONAL DESCRIPTION:
;  4642          4880  1 !
;  4643          4881  1 !       This procedure updates the overlap bit in all the pastings to
;  4644          4882  1 !       a given pasteboard -- setting a bit in the respective pasting
;  4645          4883  1 !       packets to record whether each pasting is occluded by
;  4646          4884  1 !       the highest (most-recently pasted) pasting.  This procedure is
;  4647          4885  1 !       invoked whenever a virtual display is freshly pasted.
;  4648          4886  1 !
;  4649          4887  1 !       This routine differs from SMG$$CHECK_OCCLUSION it that that
;  4650          4888  1 !       routine compares all pastings to all other pasting.  This
;  4651          4889  1 !       routine only compares the top pasting to all the others since
;  4652          4890  1 !       the addition of this new top one can only add an occlusion to
;  4653          4891  1 !       a lower one.  Any occlusions already existing at a low level
;  4654          4892  1 !       cannot have been modified by pasting one more on top and there
;  4655          4893  1 !       is no reason to recalculate their relationship to each other.
;  4656          4894  1 !
;  4657          4895  1 !       If we can determine at pasting time that a particular virtual
;  4658          4896  1 !       display (as pasted) and with a given succession of higher-pasted
;  4659          4897  1 !       virtual displays is not occluded by any of them,
;  4660          4898  1 !       we can make napping run faster since a change to a virtual
;  4661          4899  1 !       display which is not occluded causes only the changed virtual
;  4662          4900  1 !       display to be remapped.  Higher pasted virtual displays do not
;  4663          4901  1 !       have to be remapped since it is known that they do not occlude
;  4664          4902  1 !       the changed one.
;  4665          4903  1 !
;  4666          4904  1 ! CALLING SEQUENCE:
;  4667          4905  1 !
;  4668          4906  1 !       ret_status.wlc.v = SMG$$CHECK_OCCLUSION_FIRST ( PBCB.rab.r)
;  4669          4907  1 !
;  4670          4908  1 ! FORMAL PARAMETERS:
;  4671          4909  1 !
;  4672          4910  1 !       PBCB.rab.r              Address of pasteboard control block
;  4673          4911  1 !                               which has a new virtual display pasted
;  4674          4912  1 !                               to it.
;  4675          4913  1 !
;  4676          4914  1 ! IMPLICIT INPUTS:
;  4677          4915  1 !
;  4678          4916  1 !       NONE
;  4679          4917  1 !
;  4680          4918  1 ! IMPLICIT OUTPUTS:
;  4681          4919  1 !
;  4682          4920  1 !       NONE
;  4683          4921  1 !
;  4684          4922  1 ! COMPLETION STATUS:
;  4685          4923  1 !
;  4686          4924  1 !       SS$_NORMAL      Normal successful completion
;  4687          4925  1 !
;  4688          4926  1 ! SIDE EFFECTS:
;  4689          4927  1 !
;  4690          4928  1 !       NONE
;  4691          4929  1 !--
;  4692          4930  1
```

```
;  4693          4931  2     BEGIN
;  4694          4932  2     LOCAL
;  4695          4933  2         THIS_PP : REF $PP_DECL,
;  4696          4934  2                   ! Addr of pasting packet for upper-most pasted
;  4697          4935  2                   ! virtual display.
;  4698          4936  2
;  4699          4937  2         THIS_Q_HEAD : REF BLOCK [,BYTE],
;  4700          4938  2                   ! Addr of 2 longwords that form queue header in
;  4701          4939  2                   ! PP currently under inspection.
;  4702          4940  2
;  4703          4941  2         NEXT_PP : REF $PP_DECL,
;  4704          4942  2                   ! Addr of pasting packet currently under
;  4705          4943  2                   ! inspection.
;  4706          4944  2
;  4707          4945  2         NEXT_PP_Q_HEAD : REF BLOCK [,BYTE],
;  4708          4946  2                   ! Addr of 2 longwords that form queue
;  4709          4947  2                   ! header in PP currently under
;  4710          4948  2                   ! inspection.
;  4711          4949  2
;  4712          4950  2         TEMP_THIS : BLOCK [8,BYTE],
;  4713          4951  2                   ! Area of projection of THIS virtual
;  4714          4952  2                   ! display on pasteboard
;  4715          4953  2
;  4716          4954  2         TEMP_NEXT : BLOCK [8,BYTE],
;  4717          4955  2                   ! Area of projection of NEXT virtual
;  4718          4956  2                   ! display on pasteboard
;  4719          4957  2
;  4720          4958  2         THIS_DCB : REF $DCB_DECL;
;  4721          4959  2
;  4722          4960  2                   ! Addr of virtual display currently
;  4723          4961  2                   ! under inspection.
;  4724          4962  2
;  4725          4963  2     THIS_Q_HEAD = .PBCB [PBCB_A_PP_NEXT];       ! Most recent pasting
;  4726          4964  2     THIS_PP = .THIS_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4727          4965  2
;  4728          4966  2     THIS_DCB = .THIS_PP [PP_A_DCB_ADDR];
;  4729          4967  2
;  4730          4968  2 !+
;  4731          4969  2 ! It is safe to assume that there is at least one virtual
;  4732          4970  2 ! display pasted to this pasteboard -- but there may not be more than
;  4733          4971  2 ! one.  Be careful about reaching ahead to a packet that may not be a
;  4734          4972  2 ! packet.  If doesn't exist, pointer will be pointing back into PBCB
;  4735          4973  2 ! -- and inner loop will not be executed.
;  4736          4974  2 !-
;  4737          4975  2     NEXT_PP_Q_HEAD = .THIS_PP [PP_A_NEXT_PBCB];
;  4738          4976  2
;  4739          4977  2     IF .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
;  4740          4978  2     THEN
;  4741          4979  3         BEGIN   ! NEXT exists
;  4742          4980  3         NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4743          4981  3         !+
;  4744          4982  3         ! Form a representation of the projection of THIS virtual
;  4745          4983  3         ! display onto pasteboard coordinate system.
;  4746          4984  3         !-
;  4747          4985  3         TEMP_THIS [DCB_W_ROW_START] = .THIS_DCB [DCB_W_ROW_START] +
;  4748          4986  3                                       .THIS_PP [PP_W_ROW] - 1;
;  4749          4987  3         TEMP_THIS [DCB_W_NO_ROWS]   = .THIS_DCB [DCB_W_NO_ROWS];
```

```
;  4750          4988  3          TEMP_THIS [DCB_W_COL_START] = .THIS_DCB [DCB_W_COL_START] +
;  4751          4989                                        .THIS_PP [PP_W_COL] - 1;
;  4752          4990             TEMP_THIS [DCB_W_NO_COLS]   = .THIS_DCB [DCB_W_NO_COLS];
;  4753          4991
;  4754          4992             !+
;  4755          4993             ! If this virtual display is bordered, its projection is bigger
;  4756          4994             ! than if it were not.  Adjust its projection representation.
;  4757          4995             !-
;  4758          4996             IF .THIS_DCB [DCB_V_BORDERED]
;  4759          4997             THEN
;  4760          4998  4              BEGIN       ! Border adjustment
;  4761          4999  4              TEMP_THIS [DCB_W_ROW_START] = .TEMP_THIS [DCB_W_ROW_START] - 1;
;  4762          5000  4              TEMP_THIS [DCB_W_NO_ROWS]   = .TEMP_THIS [DCB_W_NO_ROWS] + 2;
;  4763          5001  4              TEMP_THIS [DCB_W_COL_START] = .TEMP_THIS [DCB_W_COL_START] - 1;
;  4764          5002  4              TEMP_THIS [DCB_W_NO_COLS]   = .TEMP_THIS [DCB_W_NO_COLS] + 2;
;  4765          5003  4              END;        ! Border adjustment
;  4766          5004
;  4767          5005             END;    ! Next exists
;  4768          5006
;  4769          5007         WHILE .NEXT_PP_Q_HEAD NEQ PBCB [PBCB_A_PP_NEXT]
;  4770          5008         DO
;  4771          5009             BEGIN   ! For all displays from current to bottom
;  4772          5010             LOCAL
;  4773          5011                 NEXT_DCB : REF $DCB_DECL,
;  4774          5012                                     ! Addr of DCB associated with NEXT_PP
;  4775          5013                 OVERLAP : BLOCK [8,BYTE];
;  4776          5014                                     ! Returned by SMG$$OCCLUDE, but not
;  4777          5015                                     ! used in this context
;  4778          5016
;  4779          5017             NEXT_PP = .NEXT_PP_Q_HEAD - PP_PBCB_QUEUE_OFFSET;
;  4780          5018             NEXT_DCB = .NEXT_PP [PP_A_DCB_ADDR];
;  4781          5019
;  4782          5020             !+
;  4783          5021             ! Form a representation of the projection of NEXT virtual
;  4784          5022             ! display onto pasteboard coordinate system.
;  4785          5023             !-
;  4786          5024             TEMP_NEXT [DCB_W_ROW_START] = .NEXT_DCB [DCB_W_ROW_START] +
;  4787          5025                                           .NEXT_PP [PP_W_ROW] - 1;
;  4788          5026             TEMP_NEXT [DCB_W_NO_ROWS]   = .NEXT_DCB [DCB_W_NO_ROWS];
;  4789          5027             TEMP_NEXT [DCB_W_COL_START] = .NEXT_DCB [DCB_W_COL_START] +
;  4790          5028                                           .NEXT_PP [PP_W_COL] - 1;
;  4791          5029             TEMP_NEXT [DCB_W_NO_COLS]   = .NEXT_DCB [DCB_W_NO_COLS];
;  4792          5030
;  4793          5031                 !+
;  4794          5032                 ! If this next virtual display is bordered, its projection is
;  4795          5033                 ! bigger than if it were not.  Adjust its projection
;  4796          5034                 ! representation.
;  4797          5035                 !-
;  4798          5036                 IF .NEXT_DCB [DCB_V_BORDERED]
;  4799          5037                 THEN
;  4800          5038  4                  BEGIN   ! Border adjustment
;  4801          5039  4                  TEMP_NEXT [DCB_W_ROW_START] = .TEMP_NEXT [DCB_W_ROW_START] - 1;
;  4802          5040  4                  TEMP_NEXT [DCB_W_NO_ROWS]   = .TEMP_NEXT [DCB_W_NO_ROWS] + 2;
;  4803          5041  4                  TEMP_NEXT [DCB_W_COL_START] = .TEMP_NEXT [DCB_W_COL_START] - 1;
;  4804          5042  4                  TEMP_NEXT [DCB_W_NO_COLS]   = .TEMP_NEXT [DCB_W_NO_COLS] + 2;
;  4805          5043  4                  END;    ! Border adjustment
;  4806          5044  3
```

```
3               !+
3               ! Check to see if THIS virtual display occludes NEXT
3               ! vitual display and if so set occlusion bit of NEXT.
3               !-
3               IF SMG$$OCCLUDE ( TEMP_NEXT, TEMP_THIS, OVERLAP)
3               THEN
3                   NEXT_PP [PP_V_OCCLUDED] = 1;
3
3               !+
3               ! Walk chain in direction of earlier pasted packets.
3               !-
3               NEXT_PP_Q_HEAD = .NEXT_PP [PP_A_NEXT_PBCB];
3
2               END;     ! For all displays from current to bottom
2
2           RETURN (SS$_NORMAL);
2
1           END;                ! End of routine SMG$$CHECK_OCCLUSION_FIRST
```

```
                        003C 00000          .ENTRY  SMG$$CHECK_OCCLUSION_FIRST, Save R2,R3,R4,- ; 4875
                                                    R5                                          ;
              5E     18   C2 00002          SUBL2   #24, SP                                     ;
              51  04 BC   D0 00005          MOVL    @PBCB, THIS_Q_HEAD                          ; 4963
              51     08   C2 00009          SUBL2   #8, THIS_PP                                 ; 4964
              50  10 A1   D0 0000C          MOVL    16(THIS_PP), THIS_DCB                       ; 4966
              53  08 A1   D0 00010          MOVL    8(THIS_PP), NEXT_PP_Q_HEAD                  ; 4975
        04    AC     53   D1 00014          CMPL    NEXT_PP_Q_HEAD, PBCB                        ; 4977
                     3F   13 00018          BEQL    1$                                          ;
              52  F8 A3   9E 0001A          MOVAB   -8(R3), NEXT_PP                             ; 4980
              54     60   3C 0001E          MOVZWL  (THIS_DCB), R4                              ; 4986
              55  18 A1   32 00021          CVTWL   24(THIS_PP), R5                             ;
              54     55   C0 00025          ADDL2   R5, R4                                      ;
 10   AE      54     01   A3 00028          SUBW3   #1, R4, TEMP_THIS                           ;
        12    AE  02 A0   B0 0002D          MOVW    2(THIS_DCB), TEMP_THIS+2                    ; 4987
              54  04 A0   3C 00032          MOVZWL  4(THIS_DCB), R4                             ; 4989
              51  1A A1   32 00036          CVTWL   26(THIS_PP), R1                             ;
              51     54   C0 0003A          ADDL2   R4, R1                                      ;
 14   AE      51     01   A3 0003D          SUBW3   #1, R1, TEMP_THIS+4                         ;
        16    AE  06 A0   B0 00042          MOVW    6(THIS_DCB), TEMP_THIS+6                    ; 4990
              0E  2F A0   E9 00047          BLBC    47(THIS_DCB), 1$                            ; 4996
                  10 AE   B7 0004B          DECW    TEMP_THIS                                   ; 4999
        12    AE  02 A0      0004E          ADDW2   #2, TEMP_THIS+2                             ; 5000
                  14 AE   B7 00052          DECW    TEMP_THIS+4                                 ; 5001
        16    AE     02   A0 00055          ADDW2   #2, TEMP_THIS+6                             ; 5002
        04    AC     53   D1 00059  1$:     CMPL    NEXT_PP_Q_HEAD, PBCB                        ; 5007
                     5F   13 0005D          BEQL    4$                                          ;
              52  F8 A3   9E 0005F          MOVAB   -8(R3), NEXT_PP                             ; 5017
              50  10 A2   D0 00063          MOVL    16(NEXT_PP), NEXT_DCB                       ; 5018
              51     60   3C 00067          MOVZWL  (NEXT_DCB), R1                              ; 5025
              54  18 A2   32 0006A          CVTWL   24(NEXT_PP), R4                             ;
              51     54   C0 0006E          ADDL2   R4, R1                                      ;
 08   AE      51     01   A3 00071          SUBW3   #1, R1, TEMP_NEXT                           ;
```

```
                          0A  AE      02  A0  B0 00076          MOVW    2(NEXT_DCB), TEMP_NEXT+2          ; 5026
                              51      04  A0  3C 0007B          MOVZWL  4(NEXT_DCB), R1                   ; 5028
                              54      1A  A2  32 0007F          CVTWL   26(NEXT_PP), R4                   ;
                              51          54  C0 00083          ADDL2   R4, R1                            ;
                    0C    AE  51          01  A3 00086          SUBW3   #1, R1, TEMP_NEXT+4               ;
                          0E  AE      06  A0  B0 0008B          MOVW    6(NEXT_DCB), TEMP_NEXT+6          ; 5029
                              0E      2F  A0  E9 00090          BLBC    47(NEXT_DCB), 2$                  ; 5036
                                      08  AE  B7 00094          DECW    TEMP_NEXT                         ; 5039
                          0A  AE          02  A0 00097          ADDW2   #2, TEMP_NEXT+2                   ; 5040
                                      0C  AE  B7 0009B          DECW    TEMP_NEXT+4                       ; 5041
                          0E  AE          02  A0 0009E          ADDW2   #2, TEMP_NEXT+6                   ; 5042
                                          5E  DD 000A2 2$:      PUSHL   SP                                ; 5049
                                      14  AE  9F 000A4          PUSHAB  TEMP_THIS                         ;
                                      10  AE  9F 000A7          PUSHAB  TEMP_NEXT                         ;
                 00000000G        00          03  FB 000AA          CALLS   #3, SMG$$OCCLUDE              ;
                              04              50  E9 000B1          BLBC    R0, 3$                        ;
                          2A  A2              01  88 000B4          BISB2   #1, 42(NEXT_PP)               ; 5051
                              53      08  A2  D0 000B8 3$:      MOVL    8(NEXT_PP), NEXT_PP_Q_HEAD        ; 5056
                                      9B  11 000BC          BRB     1$                                ; 5007
                              50          01  D0 000BE 4$:      MOVL    #1, R0                            ; 5061
                                          04 000C1          RET                                       ; 5063

; Routine Size:  194 bytes,    Routine Base:  _SMG$CODE + 1A59

; 4826          5064  1 !<BLF/PAGE>
```

```
;  4828          5065  1 %SBTTL 'SMG$$CREATE_PASTEBOARD - Create Pasteboard Control Block (PBCB)'
;  4829          5066  1 GLOBAL ROUTINE SMG$$CREATE_PASTEBOARD (
;  4830          5067  1                                       ROWS,
;  4831          5068  1                                       COLS,
;  4832          5069  1                                       PBCB_ADDR
;  4833          5070  1                                            ) =
;  4834          5071  1 !++
;  4835          5072  1 ! FUNCTIONAL DESCRIPTION:
;  4836          5073  1 !
;  4837          5074  1 !       This routine allocates space for a pasteboard control block.
;  4838          5075  1 !       It also allocates a buffer called the output buffer
;  4839          5076  1 !       which is used to buffer output to this terminal
;  4840          5077  1 !       (if buffering is enabled).
;  4841          5078  1 !
;  4842          5079  1 ! CALLING SEQUENCE:
;  4843          5080  1 !
;  4844          5081  1 !       ret_status.wlc.v = SMG$$CREATE_PASTEBOARD (
;  4845          5082  1 !                                                  ROWS.rl.r,
;  4846          5083  1 !                                                  COLS.rl.r,
;  4847          5084  1 !                                                  PBCB.wl.r)
;  4848          5085  1 !
;  4849          5086  1 ! FORMAL PARAMETERS:
;  4850          5087  1 !
;  4851          5088  1 !       ROWS.rl.r       Max. number of rows that a window onto this
;  4852          5089  1 !                       pasteboard will have.
;  4853          5090  1 !
;  4854          5091  1 !       COLS.rl.r       Max. number of columns that a window onto this
;  4855          5092  1 !                       pasteboard will have.
;  4856          5093  1 !
;  4857          5094  1 !       PBCB_ADDR.wl.r  Address of the newly-created PBCB -- returned to
;  4858          5095  1 !                       caller.
;  4859          5096  1 !
;  4860          5097  1 ! IMPLICIT INPUTS:
;  4861          5098  1 !
;  4862          5099  1 !       NONE
;  4863          5100  1 !
;  4864          5101  1 ! IMPLICIT OUTPUTS:
;  4865          5102  1 !
;  4866          5103  1 !       items in PBCB get filled in.
;  4867          5104  1 !       in particular, an output buffer is allocated.
;  4868          5105  1 !
;  4869          5106  1 ! COMPLETION STATUS:
;  4870          5107  1 !
;  4871          5108  1 !       SS$_NORMAL      Normal successful completion
;  4872          5109  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
;  4873          5110  1 !                       buffers.
;  4874          5111  1 !
;  4875          5112  1 ! SIDE EFFECTS:
;  4876          5113  1 !
;  4877          5114  1 !       NONE
;  4878          5115  1 !--
```

```
;  4880          5116  2     BEGIN
;  4881          5117  2     LOCAL
;  4882          5118  2         PBCB : REF $PBCB_DECL, ! Address of PBCB allocated.
;  4883          5119  2
;  4884          5120  2         STATUS;          ! Status of subroutine calls
;  4885          5121  2
;  4886          5122  2     LITERAL
;  4887          5123  2
;  4888          5124  2         PBCB_K_OUTBUF_DEFAULT_SIZE
;  4889          5125  2
;  4890          5126  2             = 256;       ! Default size for output buffer
;  4891          5127  2                          ! (if all other algorithms fail)
;  4892          5128  2
;  4893          5129  2 !+
;  4894          5130  2 ! Allocate space for the PBCB itself.
;  4895          5131  2 !-
;  4896          5132  2     IF NOT (STATUS = LIB$GET_VM (%REF (PBCB_K_SIZE), PBCB))
;  4897          5133  2     THEN
;  4898          5134  2         RETURN (.STATUS);
;  4899          5135  2
;  4900          5136  2     CH$FILL (0, PBCB_K_SIZE, .PBCB);    ! Clear all fields to default 0
;  4901          5137  2
;  4902          5138  2 !+
;  4903          5139  2 ! Allocate the window control block that goes along with this
;  4904          5140  2 ! pasteboard, returning failure if we can't.
;  4905          5141  2 !-
;  4906          5142  2     IF NOT (STATUS = SMG$$CREATE_WCB (.ROWS, .COLS, PBCB [PBCB_A_WCB]))
;  4907          5143  2     THEN
;  4908          5144  3         BEGIN   ! No more space
;  4909          5145  3         !+
;  4910          5146  3         ! If we can't get space for WCB, we might as well give back
;  4911          5147  3         ! the PBCB space itself.
;  4912          5148  3         !-
;  4913          5149  3         LIB$FREE_VM (%REF (PBCB_K_SIZE), PBCB);
;  4914          5150  3         RETURN (.STATUS);
;  4915          5151  2         END;    ! No more space
;  4916          5152  2
;  4917          5153  2 !+
;  4918          5154  2 ! Allocate output buffer that goes along with this pasteboard, returning
;  4919          5155  2 ! failure if we can't.
;  4920          5156  2 ! This buffer is used (if buffering is enabled) to buffer all output to
;  4921          5157  2 ! this terminal.
;  4922          5158  2 ! When V3B comes out, we should do a better job in figuring
;  4923          5159  2 ! out a good size for this buffer by looking at sysgen paramaters
;  4924          5160  2 ! and user quotas, etc.  For now, we just allocate a fixed space.
;  4925          5161  2 !-
;  4926          5162  2
;  4927          5163  2     STATUS = LIB$GET_VM (%REF (PBCB_K_OUTBUF_DEFAULT_SIZE),
;  4928          5164  2                          PBCB [PBCB_A_OUTPUT_BUFFER]);
;  4929          5165  2     IF NOT .STATUS
;  4930          5166  2     THEN
;  4931          5167  3         BEGIN
;  4932          5168  3         !+
;  4933          5169  3         ! If we can't get space for the output buffer, we might as well
;  4934          5170  3         ! give back the PBCB space itself as well as the WCB space.
;  4935          5171  3         ! Ignore any errors that occur while trying to free this space.
;  4936          5172  3         !-
```

```
; 4937     5173  3          SMG$$DEALLOCATE_WCB( .PBCB [PBCB_A_WCB] );
; 4938     5174  3          LIB$FREE_VM (%REF (PBCB_K_SIZE), PBCB);
; 4939     5175  3          RETURN (.STATUS);
; 4940     5176  2          END;
; 4941     5177  2
; 4942     5178  2      PBCB [PBCB_W_OUTPUT_BUFSIZ] = PBCB_K_OUTBUF_DEFAULT_SIZE;    ! allocation                    !
; 4943     5179  2
; 4944     5180  2  !+
; 4945     5181  2  ! Initialize pasting queue header to self.
; 4946     5182  2  !-
; 4947     5183  2      PBCB [PBCB_A_PP_NEXT] = PBCB [PBCB_A_PP_NEXT];
; 4948     5184  2      PBCB [PBCB_A_PP_PREV] = PBCB [PBCB_A_PP_NEXT];
; 4949     5185  2
; 4950     5186  2  !+
; 4951     5187  2  ! Initialize mode settings to default.
; 4952     5188  2  !-
; 4953     5189  2      PBCB [PBCB_L_MODE_SETTINGS] = PBCB_K_DEF_MODE_SETTINGS;
; 4954     5190  2
; 4955     5191  2  !+
; 4956     5192  2  ! Return the address of the PBCB we've built.
; 4957     5193  2  !-
; 4958     5194  2      .PBCB_ADDR = .PBCB;
; 4959     5195  2
; 4960     5196  2      RETURN (SS$_NORMAL);
; 4961     5197  1      END;                                ! Routine SMG$$CREATE_PASTEBOARD



                                                00FC 00000          .ENTRY  SMG$$CREATE_PASTEBOARD, Save R2,R3,R4,R5,-      ; 5066
                                                                            R6,R7
                                            57 00000000G    00  9E 00002     MOVAB   LIB$GET_VM, R7
                                            5E                  08  C2 00009     SUBL2   #8, SP
                                                            04      AE 9F 0000C     PUSHAB  PBCB                            ; 5132
                                        04  AE              014C    8F 3C 0000F     MOVZWL  #332, 4(SP)
                                                            04      AE 9F 00015     PUSHAB  4(SP)
                                            67                      02 FB 00018     CALLS   #2, LIB$GET_VM
                                            50                      50 D0 0001B     MOVL    R0, STATUS
                                            51                      56 E9 0001E     BLBC    STATUS, 2$
014C    8F          00                      6E                      00 2C 00021     MOVC5   #0, (SP), #0, #332, @PBCB       ; 5136
                                                            04      BE      00028
                    7E      04      AE                      08  C1 0002A     ADDL3   #8, PBCB, -(SP)                 ; 5142
                                            7E              04      AC 7D 0002F     MOVQ    R0WS, -(SP)
                                0000V       CF                      03 FB 00033     CALLS   #3, SMG$$CREATE_WCB
                                            50                      50 D0 00038     MOVL    R0, STATUS
                                            21                      56 E9 00038     BLBC    STATUS, 1$
                                            52              04      AE D0 0003E     MOVL    PBCB, R2                        ; 5164
                                                            6C      A2 9F 00042     PUSHAB  108(R2)
                                        04  AE              0100    8F 3C 00045     MOVZWL  #256, 4(SP)                     ; 5163
                                                            04      AE 9F 0004B     PUSHAB  4(SP)
                                            67                      02 FB 0004E     CALLS   #2, LIB$GET_VM                  ; 5164
                                            50                      50 D0 00051     MOVL    R0, STATUS
                                            1F                      56 E8 00054     BLBS    STATUS, 3$                      ; 5165
                                                            08      A2 DD 00057     PUSHL   8(R2)                           ; 5173
                                0000V       CF                      01 FB 0005A     CALLS   #1, SMG$$DEALLOCATE_WCB
                                                            04      AE 9F 0005F 1$: PUSHAB  PBCB                            ; 5174
```

```
                            04   AE      014C   8F   3C 00062          MOVZWL  #332, 4(SP)                                  ;
                                 04      AE   9F   00068               PUSHAB  4(SP)                                       ;
                    00000000G    00           02   FB 0006B          CALLS   #2, LIB$FREE_VM                             ;
                                 50           56   D0 00072 2$:      MOVL    STATUS, R0                                  ; 5175
                                              04   00075             RET                                                 ;
                                 50      04   AE   D0 00076 3$:      MOVL    PBCB, R0                                    ; 5178
                              70 A0      0100 8F   B0 0007A          MOVW    #256, 112(R0)                               ;
                                 60           50   D0 00080          MOVL    R0, (R0)                                    ; 5183
                              04 A0           50   D0 00083          MOVL    R0, 4(R0)                                   ; 5184
                              0C AC           02   D0 00087          MOVL    #2, 12(R0)                                  ; 5189
                              0C BC           50   D0 0008B          MOVL    R0, @PBCB_ADDR                              ; 5194
                                 50           01   D0 0008F          MOVL    #1, R0                                      ; 5196
                                              04   00092             RET                                                 ; 5197

; Routine Size:  147 bytes,    Routine Base:  _SMG$CODE + 1B1B

; 4962          5198  1 !<BLF/PAGE>
```

```
;  4964          5199  1 %SBTTL 'SMG$$CREATE_VIRTUAL_DISPLAY - Create Virtual Display'
;  4965          5200  1 GLOBAL ROUTINE SMG$$CREATE_VIRTUAL_DISPLAY (
;  4966          5201  1                                             NUM_ROWS,      ! height
;  4967          5202  1                                             NUM_COLS,      ! width
;  4968          5203  1                                             NEW_DISPLAY_ID,
;  4969          5204  1                                             DISPLAY_ATTRIBUTES,
;  4970          5205  1                                             VIDEO_ATTRIBUTES,
;  4971          5206  1                                             CHAR_SET
;  4972          5207  1                                             ) =
;  4973          5208  1 !++
;  4974          5209  1 ! FUNCTIONAL DESCRIPTION:
;  4975          5210  1 !
;  4976          5211  1 !       This routine creates a new virtual display -- returning its
;  4977          5212  1 !       assigned display_id.  Its initial contents are blanks with
;  4978          5213  1 !       video attributes set to those specified.  The cursor
;  4979          5214  1 !       will be at row 1 column 1.
;  4980          5215  1 !       This is the inner-most create_virtual_display routine.  It
;  4981          5216  1 !       assumes all of its parameters are present.
;  4982          5217  1 !
;  4983          5218  1 ! CALLING SEQUENCE:
;  4984          5219  1 !
;  4985          5220  1 !       ret_status.wlc.v = SMG$$CREATE_VIRTUAL_DISPLAY (
;  4986          5221  1 !                                       NUM_ROWS.rl.r,          ! Height
;  4987          5222  1 !                                       NUM_COLS.rl.r,          ! Width
;  4988          5223  1 !                                       NEW_DISPLAY_ID.wl.r,
;  4989          5224  1 !                                       DISPLAY_ATTRIBUTES.rl.r,
;  4990          5225  1 !                                       VIDEO_ATTRIBUTES.rl.r,
;  4991          5226  1 !                                       CHAR_SET.rl.r)
;  4992          5227  1 !
;  4993          5228  1 ! FORMAL PARAMETERS:
;  4994          5229  1 !
;  4995          5230  1 !       NUM_ROWS.rl.r     Number of rows in new virtual display.
;  4996          5231  1 !
;  4997          5232  1 !       NUM_COLS.rl.r     Number of columns in new virtual display.
;  4998          5233  1 !
;  4999          5234  1 !       NEW_DISPLAY_ID.wl.r     Virtual display id of newly-created
;  5000          5235  1 !                               virtual display.
;  5001          5236  1 !
;  5002          5237  1 !       DISPLAY_ATTRIBUTES.rl.r The default display attributes.
;  5003          5238  1 !
;  5004          5239  1 !                               SMG$M_BORDER if virtual display is to be
;  5005          5240  1 !                                       displayed with a border.
;  5006          5241  1 !
;  5007          5242  1 !                               SMG$M_TRUNC_ICON if an icon should be displayed
;  5008          5243  1 !                                       when text overflows the display
;  5009          5244  1 !                                       bounds.
;  5010          5245  1 !
;  5011          5246  1 !                               SMG$M_DISPLAY_CONTROLS if carriage controls (CR, LF,
;  5012          5247  1 !                                       FF, VT, HT) should be displayed instead
;  5013          5248  1 !                                       of executed.
;  5014          5249  1 !
;  5015          5250  1 !       VIDEO_ATTRIBUTES.rl.r   The default rendition code to be
;  5016          5251  1 !                               applied to all output to this display unless
;  5017          5252  1 !                               overridden on a particular output call.
;  5018          5253  1 !
;  5019          5254  1 !                               Values:
;  5020          5255  1 !
```

```
;  5021          5256  1 !                                SMG$M_BLINK     displays characters blinking.
;  5022          5257  1 !
;  5023          5258  1 !                                SMG$M_BOLD      displays characters in
;  5024          5259  1 !                                                higher-than-normal intensity.
;  5025          5260  1 !
;  5026          5261  1 !                                SMG$M_REVERSE   displays characters in reverse
;  5027          5262  1 !                                                video -- that is, using the
;  5028          5263  1 !                                                opposite default rendition of
;  5029          5264  1 !                                                the virtual display.
;  5030          5265  1 !
;  5031          5266  1 !                                SMG$M_UNDERLINE displays characters underlined.
;  5032          5267  1 !
;  5033          5268  1 !                      CHAR_SET.rb.r   Specifies the default character set to be used
;  5034          5269  1 !                                      for this display.
;  5035          5270  1 !                                      Recognized values are:
;  5036          5271  1 !                                                      SMG$C_UNITED_KINGDOM
;  5037          5272  1 !                                                      SMG$C_ASCII (default)
;  5038          5273  1 !                                                      SMG$C_SPEC_GRAPHICS
;  5039          5274  1 !                                                      SMG$C_ALT_CHAR
;  5040          5275  1 !                                                      SMG$C_ALT_GRAPHICS
;  5041          5276  1 !
;  5042          5277  1 ! IMPLICIT INPUTS:
;  5043          5278  1 !
;  5044          5279  1 !       NONE
;  5045          5280  1 !
;  5046          5281  1 ! IMPLICIT OUTPUTS:
;  5047          5282  1 !
;  5048          5283  1 !       NONE
;  5049          5284  1 !
;  5050          5285  1 ! COMPLETION STATUS:
;  5051          5286  1 !
;  5052          5287  1 !       SS$_NORMAL      Normal successful completion
;  5053          5288  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
;  5054          5289  1 !                       buffer.
;  5055          5290  1 !       SMG$_INVARG     Unrecognized Video Attributes
;  5056          5291  1 !                    or Unrecognized Display Attributes
;  5057          5292  1 !
;  5058          5293  1 ! SIDE EFFECTS:
;  5059          5294  1 !
;  5060          5295  1 !       NONE
;  5061          5296  1 !--
;  5062          5297  2     BEGIN
;  5063          5298  2     LOCAL
;  5064          5299  2         STATUS,                     ! Status of subroutine calls
;  5065          5300  2         DCB : REF $DCB_DECL,        ! Addr of display control block
;  5066          5301  2         DESC : REF BLOCK [8,BYTE]; ! Pointer to dynamic descriptor in
;  5067          5302  2                                     ! DCB for border label
;  5068          5303  2
;  5069          5304  2 !+
;  5070          5305  2 ! Allocate space for DCB itself.  Quit if we can't get it.
;  5071          5306  2 !-
;  5072          5307  3     IF NOT (STATUS = LIB$GET_VM ( %REF (DCB_K_SIZE), DCB))
;  5073          5308  2     THEN
;  5074          5309  2         RETURN (.STATUS);
;  5075          5310  2
;  5076          5311  2     CH$FILL (0, DCB_K_SIZE, .DCB);      ! set all fields to default of 0
;  5077          5312  2
```

```
;  5078          5313  2 !+
;  5079          5314  2 ! Set up dimensions of display
;  5080          5315  2 !-
;  5081          5316  2     DCB [DCB_W_ROW_START] = 1;
;  5082          5317  2     DCB [DCB_W_NO_ROWS]   = ..NUM_ROWS;
;  5083          5318  2     DCB [DCB_W_COL_START] = 1;
;  5084          5319  2     DCB [DCB_W_NO_COLS]   = ..NUM_COLS;
;  5085          5320  2     DCB [DCB_L_BUFSIZE]   = ..NUM_ROWS * ..NUM_COLS;
;  5086          5321  2
;  5087          5322  2 !+
;  5088          5323  2 ! Record default display attributes, default video attributes and
;  5089          5324  2 ! default character set.
;  5090          5325  2 !-
;  5091          5326  2     DCB [DCB_B_DEF_DISPLAY_ATTR] = ..DISPLAY_ATTRIBUTES;
;  5092          5327  2     DCB [DCB_B_DEF_VIDEO_ATTR]   = ..VIDEO_ATTRIBUTES;
;  5093          5328  2     DCB [DCB_B_DEF_CHAR_SET]     = ..CHAR_SET;
;  5094          5329  2
;  5095          5330  2 !+
;  5096          5331  2 ! Set up various fields in the DCB
;  5097          5332  2 !-
;  5098          5333  2     DCB [DCB_L_DID] = .DCB;                 ! Display id itself -- the
;  5099          5334  2                                             ! address of the DCB
;  5100          5335  2
;  5101          5336  2     DCB [DCB_W_CURSOR_ROW] = 1;             ! Cursor row and column to home
;  5102          5337  2     DCB [DCB_W_CURSOR_COL] = 1;
;  5103          5338  2
;  5104          5339  2     DCB [DCB_B_STRUCT_TYPE]  = DCB_K_STRUCT_TYPE; ! Mark as being a DCB
;  5105          5340  2     DCB [DCB_W_DCB_LENGTH]   = DCB_K_SIZE;        ! Size of structure
;  5106          5341  2
;  5107          5342  2     DCB [DCB_W_TOP_OF_SCRREG] = .DCB [DCB_W_ROW_START];
;  5108          5343  2     DCB [DCB_W_BOTTOM_OF_SCRREG] = .DCB [DCB_W_NO_ROWS];
;  5109          5344  2                                             ! init scrolling region
;  5110          5345  2 !+
;  5111          5346  2 ! Allocate enough space for both the text buffer and the attribute
;  5112          5347  2 ! buffer.
;  5113          5348  2 !-
;  5114          5349  3     IF NOT (STATUS = LIB$GET_VM (%REF (.DCB [DCB_L_BUFSIZE] *2),
;  5115          5350  3                                  DCB [DCB_A_TEXT_BUF]))
;  5116          5351  2     THEN
;  5117          5352  3         BEGIN   ! Ran out of space
;  5118          5353  3         !+
;  5119          5354  3         ! If we can't get enough space for the buffers, we might as well
;  5120          5355  3         ! give back the DCB space itself.
;  5121          5356  3         !-
;  5122          5357  3         LIB$FREE_VM ( %REF (DCB_K_SIZE), DCB); ! Return DCB space
;  5123          5358  3         RETURN (.STATUS);                      ! Return the LIB$_INSVIRMEM from the
;  5124          5359  3                                                ! buffer allocation attempt
;  5125          5360  2         END;    ! Ran out of space
;  5126          5361  2
;  5127          5362  2 !+
;  5128          5363  2 ! Use upper half of space allocated as the attribute buffer.
;  5129          5364  2 !-
;  5130          5365  2     DCB [DCB_A_ATTR_BUF] = .DCB [DCB_A_TEXT_BUF] + .DCB [DCB_L_BUFSIZE];
;  5131          5366  2
;  5132          5367  2 !+
;  5133          5368  2 ! Initialize text and attribute buffers.
;  5134          5369  2 !-
```

```
;  5135          5370  2     CH$FILL (%C' ', .DCB [DCB_L_BUFSIZE], .DCB [DCB_A_TEXT_BUF]);
;  5136          5371  2     CH$FILL (.DCB [DCB_B_DEF_VIDEO_ATTR], .DCB [DCB_L_BUFSIZE],
;  5137          5372  2              .DCB [DCB_A_ATTR_BUF]);
;  5138          5373  2
;  5139          5374  2 !+
;  5140          5375  2 ! If we are dealing with a non-standard character set, allocate the
;  5141          5376  2 ! char_set buffer.  If we can't, bail out, giving back all the space
;  5142          5377  2 ! allocated on this transaction.
;  5143          5378  2 !-
;  5144          5379  2     IF .DCB [DCB_B_DEF_CHAR_SET] NEQ 0
;  5145          5380  2     THEN
;  5146          5381  3         BEGIN   ! Will need char_set buffer
;  5147          5382  4         IF NOT (STATUS = LIB$GET_VM ( DCB [DCB_L_BUFSIZE],
;  5148          5383  4                                       DCB [DCB_A_CHAR_SET_BUF]))
;  5149          5384  3         THEN
;  5150          5385  4             BEGIN       ! Bailout
;  5151          5386  4             !+
;  5152          5387  4             ! If we can't get space for buffer we need, give back the
;  5153          5388  4             ! the text and attribute buffer, and DCB itself before
;  5154          5389  4             ! quitting.
;  5155          5390  4             !-
;  5156          5391  4             LIB$FREE_VM (%REF (2 * .DCB [DCB_L_BUFSIZE]),
;  5157          5392  4                          DCB [DCB_A_TEXT_BUF]);
;  5158          5393  4             LIB$FREE_VM (%REF (DCB_K_SIZE), DCB);
;  5159          5394  4             RETURN (.STATUS);
;  5160          5395  3             END;        ! Bailout
;  5161          5396  3
;  5162          5397  3         CH$FILL (.DCB [DCB_B_DEF_CHAR_SET], .DCB [DCB_L_BUFSIZE],
;  5163          5398  3                  .DCB [DCB_A_CHAR_SET_BUF]);
;  5164          5399  3
;  5165          5400  2         END;    ! Will need char_set buffer
;  5166          5401  2
;  5167          5402  2
;  5168          5403  2 !+
;  5169          5404  2 ! Allocate and clear the line characteristics vector.
;  5170          5405  2 !-
;  5171          5406  3     IF NOT (STATUS = LIB$GET_VM ( %REF ( .DCB [DCB_W_NO_ROWS] +1),
;  5172          5407  3                                   DCB [DCB_A_LINE_CHAR]))
;  5173          5408  2     THEN
;  5174          5409  3         BEGIN   ! Error path
;  5175          5410  3         !+
;  5176          5411  3         ! Give back all space accumulated on this trans. before
;  5177          5412  3         ! bailing out.
;  5178          5413  3         !-
;  5179          5414  3         LIB$FREE_VM (%REF (2 * .DCB [DCB_L_BUFSIZE]),
;  5180          5415  3                      DCB [DCB_A_TEXT_BUF]);
;  5181          5416  3
;  5182          5417  3         IF .DCB [DCB_A_CHAR_SET_BUF] NEQ 0
;  5183          5418  3         THEN
;  5184          5419  3             LIB$FREE_VM ( DCB [DCB_L_BUFSIZE], DCB [DCB_A_CHAR_SET_BUF]);
;  5185          5420  3
;  5186          5421  3         LIB$FREE_VM (%REF (DCB_K_SIZE), DCB);
;  5187          5422  3
;  5188          5423  3         RETURN (.STATUS);
;  5189          5424  2         END;    ! Error path
;  5190          5425  2
;  5191          5426  2     CH$FILL ( 0, .DCB [DCB_W_NO_ROWS] + 1, .DCB [DCB_A_LINE_CHAR]);
```

```
;  5192          5427  2
;  5193          5428  2 !+
;  5194          5429  2 ! Initialize pasting queue headers to point to self.
;  5195          5430  2 !-
;  5196          5431  2     DCB [DCB_A_PP_NEXT] = DCB [DCB_A_PP_NEXT];
;  5197          5432  2     DCB [DCB_A_PP_PREV] = DCB [DCB_A_PP_NEXT];
;  5198          5433  2
;  5199          5434  2 !+
;  5200          5435  2 ! Initialize border label descriptor to virgin dynamic string descriptor
;  5201          5436  2 !-
;  5202          5437  2     DESC = DCB [DCB_Q_LABEL_DESC];
;  5203          5438  2
;  5204          5439  2     DESC [DSC$B_CLASS]          = DSC$K_CLASS_D;
;  5205          5440  2     DESC [DSC$B_DTYPE]          = DSC$K_DTYPE_T;
;  5206          5441  2
;  5207          5442  2 !+
;  5208          5443  2 ! Return the new display id to caller
;  5209          5444  2 !-
;  5210          5445  2     .NEW_DISPLAY_ID = .DCB;
;  5211          5446  2
;  5212          5447  2     RETURN (SS$_NORMAL);
;  5213          5448  1     END;                            ! Routine SMG$$CREATE_VIRTUAL_DISPLAY



                                                03FC 00000          .ENTRY  SMG$$CREATE_VIRTUAL_DISPLAY, Save R2,R3,R4,-; 5200
                                                                            R5,R6,R7,R8,R9
                                      59 00000000G   00   9E 00002  MOVAB   LIB$FREE_VM, R9                                ;
                                      58 00000000G   00   9E 00009  MOVAB   LIB$GET_VM, R8                                 ;
                                      5E             08   C2 00010  SUBL2   #8, SP                                       ;
                                               04    AE   9F 00013  PUSHAB  DCB                                          ; 5307
                                   04  AE          70   8F 9A 00016  MOVZBL  #112, 4(SP)                                  ;
                                               04    AE   9F 0001B  PUSHAB  4(SP)                                        ;
                                      68           02   FB 0001E  CALLS   #2, LIB$GET_VM                                ;
                                      57           50   D0 00021  MOVL    R0, STATUS                                    ;
                                      03           57   E8 00024  BLBS    STATUS, 1$                                    ;
                                                00F5   31 00027  BRW     7$                                            ;
                                      56           04   AE D0 0002A 1$:  MOVL    DCB, R6                                 ; 5311
0070  8F                 00           6E           00   2C 0002E  MOVC5   #0, (SP), #0, #112, (R6)                       ;
                                                   66        00035                                                       ;
                                      66           01   B0 00036  MOVW    #1, (R6)                                      ; 5316
                                   02  A6          04   BC B0 00039  MOVW    @NUM_ROWS, 2(R6)                              ; 5317
                                   04  A6          01   B0 0003E  MOVW    #1, 4(R6)                                     ; 5318
                                   06  A6          08   BC B0 00042  MOVW    @NUM_COLS, 6(R6)                              ; 5319
                              3C   A6          04   BC   08 BC C5 00047  MULL3   @NUM_COLS, @NUM_ROWS, 60(R6)           ; 5320
                                   2F  A6          10   BC 90 0004E  MOVB    @DISPLAY_ATTRIBUTES, 47(R6)                   ; 5326
                                   2E  A6          14   BC 90 00053  MOVB    @VIDEO_ATTRIBUTES, 46(R6)                     ; 5327
                                   30  A6          18   BC 90 00058  MOVB    @CHAR_SET, 48(R6)                             ; 5328
                                   38  A6               56   D0 0005D  MOVL    R6, 56(R6)                                ; 5333
                                   28  A6  00010001    8F   D0 00061  MOVL    #65537, 40(R6)                            ; 5336
                                   44  A6               11   90 00069  MOVB    #17, 68(R6)                               ; 5339
                                   45  A6          70   8F   9B 0006D  MOVZBW  #112, 69(R6)                               ; 5340
                                   48  A6               66   D0 00072  MOVL    (R6), 72(R6)                              ; 5342
                                               10    A6   9F 00076  PUSHAB  16(R6)                                       ; 5350
                              04   AE          3C   A6        01 78 00079  ASHL    #1, 60(R6), 4(SP)                     ; 5349
```

```
                                        04    AE   9F 0007F        PUSHAB  4(SP)                                          ;
                                  68          02   FB 00082        CALLS   #2, LIB$GET_VM                                 ; 5350
                                  57          50   D0 00085        MOVL    R0, STATUS                                     ;
                                  03          57   E8 00088        BLBS    STATUS, 2$                                     ;
                                      0083         31 0008B        BRW     6$                                             ;
                                  56    04    AE   D0 0008E 2$:    MOVL    DCB, R6                                        ; 5365
                   14    A6    10 A6    3C    A6   C1 00092        ADDL3   60(R6), 16(R6), 20(R6)                         ;
     3C    A6          20         6E    00         2C 00099        MOVC5   #0, (SP), #32, 60(R6), @16(R6)                 ; 5370
                                        10    B6      0009F                                                               ;
     3C    A6    2E    A6         6E    00         2C 000A1        MOVC5   #0, (SP), 46(R6), 60(R6), @20(R6)              ; 5372
                                        14    B6      000A8                                                               ;
                                        30    A6   95 000AA        TSTB    48(R6)                                         ; 5379
                                              2A   13 000AD        BEQL    4$                                             ;
                                        18    A6   9F 000AF        PUSHAB  24(R6)                                         ; 5383
                                        3C    A6   9F 000B2        PUSHAB  60(R6)                                         ; 5382
                                  68          02   FB 000B5        CALLS   #2, LIB$GET_VM                                 ; 5383
                                  57          50   D0 000B8        MOVL    R0, STATUS                                     ;
                                  0E          57   E8 000BB        BLBS    STATUS, 3$                                     ;
                                        10    A6   9F 000BE        PUSHAB  16(R6)                                         ; 5392
                   04    AE    3C A6    01         78 000C1        ASHL    #1, 60(R6), 4(SP)                              ; 5391
                                        04    AE   9F 000C7        PUSHAB  4(SP)                                          ;
                                              42   11 000CA        BRB     5$                                             ; 5392
                                  50    04    AE   D0 000CC 3$:    MOVL    DCB, R0                                        ; 5397
     3C    A0    30    A0         6E    00         2C 000D0        MOVC5   #0, (SP), 48(R0), 60(R0), @24(R0)              ; 5398
                                        18    B0      000D7                                                               ;
                                  52    04    AE   D0 000D9 4$:    MOVL    DCB, R2                                        ; 5407
                                        4C    A2   9F 000DD        PUSHAB  76(R2)                                         ;
                            04    AE    02    A2   3C 000E0        MOVZWL  2(R2), 4(SP)                                   ; 5406
                                        04    AE   D6 000E5        INCL    4(SP)                                          ;
                                        04    AE   9F 000E8        PUSHAB  4(SP)                                          ;
                                  68          02   FB 000EB        CALLS   #2, LIB$GET_VM                                 ; 5407
                                  57          50   D0 000EE        MOVL    R0, STATUS                                     ;
                                  2F          57   E8 000F1        BLBS    STATUS, 8$                                     ;
                                        10    A2   9F 000F4        PUSHAB  16(R2)                                         ; 5415
                   04    AE    3C A2    01         78 000F7        ASHL    #1, 60(R2), 4(SP)                              ; 5414
                                        04    AE   9F 000FD        PUSHAB  4(SP)                                          ;
                                  69          02   FB 00100        CALLS   #2, LIB$FREE_VM                                ; 5415
                                        18    A2   D5 00103        TSTL    24(R2)                                         ; 5417
                                              09   13 00106        BEQL    6$                                             ;
                                        18    A2   9F 00108        PUSHAB  24(R2)                                         ; 5419
                                        3C    A2   9F 0010B        PUSHAB  60(R2)                                         ;
                                  69          02   FB 0010E 5$:    CALLS   #2, LIB$FREE_VM                                ;
                                        04    AE   9F 00111 6$:    PUSHAB  DCB                                            ; 5421
                            04    AE    70    8F   9A 00114        MOVZBL  #112, 4(SP)                                    ;
                                        04    AE   9F 00119        PUSHAB  4(SP)                                          ;
                                  69          02   FB 0011C        CALLS   #2, LIB$FREE_VM                                ;
                                  50          57   D0 0011F 7$:    MOVL    STATUS, R0                                     ; 5423
                                                   04 00122        RET                                                    ;
                                  56    04    AE   D0 00123 8$:    MOVL    DCB, R6                                        ; 5426
                                  50    02    A6   3C 00127        MOVZWL  2(R6), R0                                      ;
                                              50   D6 0012B        INCL    R0                                             ;
           50          00         6E    00         2C 0012D        MOVC5   #0, (SP), #0, R0, @76(R6)                      ;
                                        4C    B6      00132                                                               ;
                            20    A6    20    A6   9E 00134        MOVAB   32(R6), 32(R6)                                 ; 5431
                            24    A6    20    A6   9E 00139        MOVAB   32(R6), 36(R6)                                 ; 5432
                                  50    08    A6   9E 0013E        MOVAB   8(R6), DESC                                    ; 5437
                            02    A0  020E    8F   B0 00142        MOVW    #526, 2(DESC)                                  ; 5440
```

```
                          0C   BC           56  D0 00148        MOVL    R6, @NEW_DISPLAY_ID                     ; 5445
                               50           01  D0 0014C        MOVL    #1, R0                                  ; 5447
                                            04 0014F          RET                                             ; 5448

; Routine Size:  336 bytes,    Routine Base:  _SMG$CODE + 1BAE

; 5214          5449  1 !<BLF/PAGE>
```

```
;  5216          5450  1 %SBTTL 'SMG$$CREATE_WCB - Create WCB and its buffers'
;  5217          5451  1 GLOBAL ROUTINE SMG$$CREATE_WCB (
;  5218          5452  1                                 ROWS,
;  5219          5453  1                                 COLS,
;  5220          5454  1                                 WCB_ADDR
;  5221          5455  1                                       ) =
;  5222          5456  1 !++
;  5223          5457  1 ! FUNCTIONAL DESCRIPTION:
;  5224          5458  1 !
;  5225          5459  1 !       This routine allocates space for the window control block and
;  5226          5460  1 !       its window text and attribute buffers and initializes them.
;  5227          5461  1 !       Two sets of these two buffers are built -- one to reflect what
;  5228          5462  1 !       is currently on the screen and one to build up what the next
;  5229          5463  1 !       screen image should look like.
;  5230          5464  1 !
;  5231          5465  1 ! CALLING SEQUENCE:
;  5232          5466  1 !
;  5233          5467  1 !       ret_status.wlc.v = SMG$$CREATE_WCB (    ROWS.rl.r,
;  5234          5468  1 !                                               COLS.rl.r,
;  5235          5469  1 !                                               WCB_ADDR.wl.r)
;  5236          5470  1 !
;  5237          5471  1 ! FORMAL PARAMETERS:
;  5238          5472  1 !
;  5239          5473  1 !       ROWS.rl.r       No. of rows in each of the buffers
;  5240          5474  1 !
;  5241          5475  1 !       COLS.rl.r       No. of columns in each of the buffers
;  5242          5476  1 !
;  5243          5477  1 !       WCB_ADDR.wl.r   Address of the newly-created WCB -- returned to
;  5244          5478  1 !                       caller.
;  5245          5479  1 !
;  5246          5480  1 !
;  5247          5481  1 !
;  5248          5482  1 ! IMPLICIT INPUTS:
;  5249          5483  1 !
;  5250          5484  1 !       NONE
;  5251          5485  1 !
;  5252          5486  1 ! IMPLICIT OUTPUTS:
;  5253          5487  1 !
;  5254          5488  1 !       NONE
;  5255          5489  1 !
;  5256          5490  1 ! COMPLETION STATUS:
;  5257          5491  1 !
;  5258          5492  1 !       SS$_NORMAL      Normal successful completion
;  5259          5493  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
;  5260          5494  1 !                       buffer.
;  5261          5495  1 !
;  5262          5496  1 ! SIDE EFFECTS:
;  5263          5497  1 !
;  5264          5498  1 !       NONE
;  5265          5499  1 !--
;  5266          5500  2     BEGIN
;  5267          5501  2     LOCAL
;  5268          5502  2         WCB : REF $WCB_DECL,    ! Address of WCB allocated.
;  5269          5503  2         STATUS;                 ! Status of subroutine calls
;  5270          5504  2
;  5271          5505  2 !+
;  5272          5506  2 ! Allocate space for the WCB itself.
```

```
;  5273          5507  2 !-
;  5274          5508  3     IF NOT (STATUS = LIB$GET_VM (%REF (WCB_K_SIZE), WCB))
;  5275          5509  2     THEN
;  5276          5510  2         RETURN (.STATUS);
;  5277          5511  2
;  5278          5512  2     CH$FILL (0, WCB_K_SIZE, .WCB);      ! Clear all fields to default 0
;  5279          5513  2
;  5280          5514  2     WCB [WCB_L_BUFSIZE] = ..ROWS * ..COLS; ! Overall size of each buffer
;  5281          5515  2
;  5282          5516  2 !+
;  5283          5517  2 ! Attempt to get space for all 4 buffers at once, returning an error if
;  5284          5518  2 ! we can't.
;  5285          5519  2 !-
;  5286          5520  3     IF NOT (STATUS = LIB$GET_VM ( %REF (4 * .WCB [WCB_L_BUFSIZE]),
;  5287          5521  3                                   WCB [WCB_A_TEXT_BUF]))
;  5288          5522  2     THEN
;  5289          5523  3         BEGIN   ! No more space
;  5290          5524  3         !+
;  5291          5525  3         ! If we can't get space for buffers, we might as well give back
;  5292          5526  3         ! the WCB space itself.
;  5293          5527  3         !-
;  5294          5528  3         LIB$FREE_VM (%REF (WCB_K_SIZE), WCB);
;  5295          5529  3         RETURN (.STATUS);
;  5296          5530  3         END;    ! No more space
;  5297          5531  2
;  5298          5532  2 !+
;  5299          5533  2 ! Carve up the space gotten into the 4 buffers we need.
;  5300          5534  2 !-
;  5301          5535  2     WCB [WCB_A_ATTR_BUF]     = .WCB [WCB_A_TEXT_BUF] +     .WCB [WCB_L_BUFSIZE];
;  5302          5536  2     WCB [WCB_A_SCR_TEXT_BUF] = .WCB [WCB_A_TEXT_BUF] + 2 * .WCB [WCB_L_BUFSIZE];
;  5303          5537  2     WCB [WCB_A_SCR_ATTR_BUF] = .WCB [WCB_A_TEXT_BUF] + 3 * .WCB [WCB_L_BUFSIZE];
;  5304          5538  2
;  5305          5539  2 !+
;  5306          5540  2 ! Initialize the working buffers.
;  5307          5541  2 !-
;  5308          5542  2     CH$FILL (%C' ', .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_TEXT_BUF]);
;  5309          5543  2     CH$FILL (0,     .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_ATTR_BUF]);
;  5310          5544  2
;  5311          5545  2 !+
;  5312          5546  2 ! Initialize the buffers representing what's on the screen to non-
;  5313          5547  2 ! matchable text as an initial state.  This means the first time
;  5314          5548  2 ! minimum screen update looks at it it will cause the entire window
;  5315          5549  2 ! to be repainted.
;  5316          5550  2 !-
;  5317          5551  2     CH$FILL (-1, .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_SCR_TEXT_BUF]);
;  5318          5552  2     CH$FILL (0,  .WCB [WCB_L_BUFSIZE], .WCB [WCB_A_SCR_ATTR_BUF]);
;  5319          5553  2
;  5320          5554  2 !+
;  5321          5555  2 ! Allocate the line characteristic vectors.  There are two of them --
;  5322          5556  2 ! one for the text buffer and one for the screen text buffer.  We
;  5323          5557  2 ! allocate and initialize them together for efficiency.
;  5324          5558  2 !-
;  5325          5559  3     IF NOT (STATUS = LIB$GET_VM ( %REF (2 * (..ROWS + 1)),
;  5326          5560  3                                   WCB [WCB_A_LINE_CHAR]))
;  5327          5561  2     THEN
;  5328          5562  3         BEGIN   ! Error path
;  5329          5563  3         !+
```

```
;  5330          5564  3          ! Give back all space accumulated on this transaction.
;  5331          5565  3          !-
;  5332          5566  3          LIB$FREE_VM ( %REF (4 * .WCB [WCB_L_BUFSIZE]),
;  5333          5567  3                        WCB [WCB_A_TEXT_BUF]);
;  5334          5568  3
;  5335          5569  3          LIB$FREE_VM ( %REF (WCB_K_SIZE), WCB);
;  5336          5570  2          END;    ! Error path
;  5337          5571  2      !+
;  5338          5572  2      ! Clear both buffer to zero at once
;  5339          5573  2      !-
;  5340          5574  2          CH$FILL (0, 2 * (..ROWS + 1), .WCB [WCB_A_LINE_CHAR]);
;  5341          5575  2
;  5342          5576  2      !+
;  5343          5577  2      ! Use upper half of space just allocated and cleared as the line
;  5344          5578  2      ! characteristics vector for the screen text buffer.
;  5345          5579  2      !-
;  5346          5580  2          WCB [WCB_A_SCR_LINE_CHAR] = .WCB [WCB_A_LINE_CHAR] + ..ROWS + 1;
;  5347          5581  2
;  5348          5582  2      !+
;  5349          5583  2      ! Fill in rest of WCB and return the address of the WCB we've built.
;  5350          5584  2      !-
;  5351          5585  2          WCB [WCB_W_NO_ROWS] = ..ROWS;
;  5352          5586  2          WCB [WCB_W_ROW_START] = 1;
;  5353          5587  2          WCB [WCB_W_NO_COLS] = ..COLS;
;  5354          5588  2          WCB [WCB_W_COL_START]  = 1;
;  5355          5589  2
;  5356          5590  2          .WCB_ADDR = .WCB;
;  5357          5591  2
;  5358          5592  2          RETURN (SS$_NORMAL);
;  5359          5593  1          END;                              ! Routine SMG$$CREATE_WCB


                                                  0FFC 00000      .ENTRY  SMG$$CREATE_WCB, Save R2,R3,R4,R5,R6,R7,R8,-  ; 5451
                                                                          R9,R10,R11
                                        5B 00000000G   00  9E 00002  MOVAB   LIB$FREE_VM, R11
                                        5E             08  C2 00009  SUBL2   #8, SP
                                                 04    AE  9F 0000C  PUSHAB  WCB                                         ; 5508
                                              04       AE  34 D0 0000F  MOVL    #52, 4(SP)
                                                 04    AE  9F 00013  PUSHAB  4(SP)
                                     00000000G   00       02 FB 00016  CALLS   #2, LIB$GET_VM
                                                 5A       50 D0 0001D  MOVL    R0, STATUS
                                                 3A       5A E9 00020  BLBC    STATUS, 1$
                                                 56    04 AE D0 00023  MOVL    WCB, R6                                   ; 5512
                34                    00         6E       00 2C 00027  MOVC5   #0, (SP), #0, #52, (R6)
                                                                66 0002C
                                                 59    04 BC D0 0002D  MOVL    @ROWS, R9                                 ; 5514
                                28    A6         59    08 BC C5 00031  MULL3   @COLS, R9, 40(R6)
                                                 08    A6 9F 00037  PUSHAB  8(R6)                                     ; 5521
                                04    AE   28    A6       02 78 0003A  ASHL    #2, 40(R6), 4(SP)                         ; 5520
                                                 04    AE  9F 00040  PUSHAB  4(SP)
                                     00000000G   00       02 FB 00043  CALLS   #2, LIB$GET_VM                             ; 5521
                                                 5A       50 D0 0004A  MOVL    R0, STATUS
                                                 11       5A E8 0004D  BLBS    STATUS, 2$
                                                 04    AE 9F 00050  PUSHAB  WCB                                       ; 5528
```

```
                          04   AE          34   D0 00053           MOVL    #52, 4(SP)                               :
                                     04    AE   9F 00057           PUSHAB  4(SP)                                    :
                               6B          02   FB 0005A           CALLS   #2, LIB$FREE_VM                          :
                               50          5A   D0 0005D 1$:       MOVL    STATUS, R0                               : 5529
                                                04 00060           RET                                              :
                               56    04    AE   D0 00061 2$:       MOVL    WCB, R6                                  : 5535
                               57    08    A6   9E 00065           MOVAB   8(R6), R7                                :
                               58    28    A6   9E 00069           MOVAB   40(R6), R8                               :
                 0C   A6       67          68   C1 0006D           ADDL3   (R8), (R7), 12(R6)                       :
                               50          68   D0 00072           MOVL    (R8), R0                                 : 5536
                          14   A6    00 B740    3E 00075           MOVAW   @0(R7)[R0], 20(R6)                       :
                      50       68          03   C5 0007B           MULL3   #3, (R8), R0                             : 5537
                 18   A6       50          67   C1 0007F           ADDL3   (R7), R0, 24(R6)                         :
          68          20       6E          00   2C 00084           MOVC5   #0, (SP), #32, (R8), @0(R7)              : 5542
                                     00    B7      00089                                                            :
          68          00       6E          00   2C 0008B           MOVC5   #0, (SP), #0, (R8), @12(R6)              : 5543
                                     0C    B6      00090                                                            :
          68     FF   8F       6E          00   2C 00092           MOVC5   #0, (SP), #-1, (R8), @20(R6)             : 5551
                                     14    B6      00098                                                            :
          68          00       6E          00   2C 0009A           MOVC5   #0, (SP), #0, (R8), @24(R6)              : 5552
                                     18    B6      0009F                                                            :
                                     2C    A6   9F 000A1           PUSHAB  44(R6)                                   : 5560
                      52       59          01   78 000A4           ASHL    #1, R9, R2                               : 5559
                               52          02   C0 000A8           ADDL2   #2, R2                                   :
                          04   AE          52   D0 000AB           MOVL    R2, 4(SP)                                :
                                     04    AE   9F 000AF           PUSHAB  4(SP)                                    :
                 00000000G     00          02   FB 000B2           CALLS   #2, LIB$GET_VM                           : 5560
                               5A          50   D0 000B9           MOVL    R0, STATUS                               :
                               1A          5A   E8 000BC           BLBS    STATUS, 3$                               :
                                           57   DD 000BF           PUSHL   R7                                       : 5567
                 04   AE       68          02   78 000C1           ASHL    #2, (R8), 4(SP)                          : 5566
                                     04    AE   9F 000C6           PUSHAB  4(SP)                                    :
                               6B          02   FB 000C9           CALLS   #2, LIB$FREE_VM                          : 5567
                                     04    AE   9F 000CC           PUSHAB  WCB                                      : 5569
                          04   AE          34   D0 000CF           MOVL    #52, 4(SP)                               :
                                     04    AE   9F 000D3           PUSHAB  4(SP)                                    :
                               6B          02   FB 000D6           CALLS   #2, LIB$FREE_VM                          :
                               57    04    AE   D0 000D9 3$:       MOVL    WCB, R7                                  : 5574
          52          00       6E          00   2C 000DD           MOVC5   #0, (SP), #0, R2, @44(R7)                :
                                     2C    B7      000E2                                                            :
                      50       59    2C    A7   C1 000E4           ADDL3   44(R7), R9, R0                           : 5580
                          30   A7    01    A0   9E 000E9           MOVAB   1(R0), 48(R7)                            :
                          02   A7          59   B0 000EE           MOVW    R9, 2(R7)                                : 5585
                               67          01   B0 000F2           MOVW    #1, (R7)                                 : 5586
                          06   A7    08    BC   B0 000F5           MOVW    @COLS, 6(R7)                             : 5587
                          04   A7          01   B0 000FA           MOVW    #1, 4(R7)                                : 5588
                          0C   BC          57   D0 000FE           MOVL    R7, @WCB_ADDR                            : 5590
                               50          01   D0 00102           MOVL    #1, R0                                   : 5592
                                                04 00105           RET                                              : 5593

; Routine Size:  262 bytes,    Routine Base:  _SMG$CODE + 1CFE


; 5360          5594  1 !<BLF/PAGE>
```

```
;  5362          5595  1 %SBTTL 'SMG$$DEALLOCATE_WCB - Get rid of WCB and its buffers'
;  5363          5596  1 GLOBAL ROUTINE SMG$$DEALLOCATE_WCB (WCB : REF $WCB_DECL) =
;  5364          5597  1 !++
;  5365          5598  1 ! FUNCTIONAL DESCRIPTION:
;  5366          5599  1 !
;  5367          5600  1 !       This routine deallocates space for the window control block and
;  5368          5601  1 !       its window text and attribute buffers.
;  5369          5602  1 !
;  5370          5603  1 ! CALLING SEQUENCE:
;  5371          5604  1 !
;  5372          5605  1 !       ret_status.wlc.v = SMG$$CREATE_WCB ( WCB.wl.r)
;  5373          5606  1 !
;  5374          5607  1 ! FORMAL PARAMETERS:
;  5375          5608  1 !
;  5376          5609  1 !       WCB.wl.r        Address of the previously-created WCB.
;  5377          5610  1 !
;  5378          5611  1 ! IMPLICIT INPUTS:
;  5379          5612  1 !
;  5380          5613  1 !       contents of WCB
;  5381          5614  1 !
;  5382          5615  1 ! IMPLICIT OUTPUTS:
;  5383          5616  1 !
;  5384          5617  1 !       NONE
;  5385          5618  1 !
;  5386          5619  1 ! COMPLETION STATUS:
;  5387          5620  1 !
;  5388          5621  1 !       SS$_NORMAL      Normal successful completion
;  5389          5622  1 !       LIB$_xxx        Errors from LIB$FREE_VM
;  5390          5623  1 !
;  5391          5624  1 ! SIDE EFFECTS:
;  5392          5625  1 !
;  5393          5626  1 !       NONE
;  5394          5627  1 !--
;  5395          5628  2     BEGIN
;  5396          5629  2     LOCAL
;  5397          5630  2         RET_STATUS,             ! Status to be returned to caller
;  5398          5631  2         STATUS;                 ! Status of subroutine calls
;  5399          5632  2
;  5400          5633  2 !+
;  5401          5634  2 ! Attempt to deallocate the space for all 4 buffers (text and attr) at
;  5402          5635  2 ! once.
;  5403          5636  2 !-
;  5404          5637  2     RET_STATUS = LIB$FREE_VM ( %REF(4 * .WCB [WCB_L_BUFSIZE]),
;  5405          5638  2                                WCB [WCB_A_TEXT_BUF]);
;  5406          5639  2
;  5407          5640  2 !+
;  5408          5641  2 ! Attempt to deallocate the alternate character set buffers if they
;  5409          5642  2 ! exist.
;  5410          5643  2 !-
;  5411          5644  2     IF .WCB [WCB_A_CHAR_SET_BUF] NEQ 0
;  5412          5645  2     THEN
;  5413          5646  3         BEGIN   ! Free alt char set buffers
;  5414          5647  3         ! NOTE: Right now we free them separately.  If it turns out
;  5415          5648  3         ! they are allocated as a adjacent pair, we can deallocate
;  5416          5649  3         ! them with a single call.
;  5417          5650  3
;  5418          5651  3         STATUS = LIB$FREE_VM ( WCB [WCB_L_BUFSIZE],
```

```
;  5419     5652  3                                  WCB [WCB_A_CHAR_SET_BUF]);
;  5420     5653  3
;  5421     5654  3          IF NOT .STATUS THEN RET_STATUS = .STATUS ; ! Propagate an error
;  5422     5655  3                                                    ! status
;  5423     5656  3
;  5424     5657  3          STATUS = LIB$FREE_VM ( WCB [WCB_L_BUFSIZE],
;  5425     5658  3                                 WCB [WCB_A_SCR_CHAR_SET_BUF]);
;  5426     5659  3
;  5427     5660  3          IF NOT .STATUS THEN RET_STATUS = .STATUS ; ! Propagate an error
;  5428     5661  3                                                    ! status
;  5429     5662  2          END;     ! Free alt char set buffers
;  5430     5663  2
;  5431     5664  2      !+
;  5432     5665  2      ! Deallocate the line characteristics vectors.  These were allocated
;  5433     5666  2      ! as a pair so can be deallocated as a pair.
;  5434     5667  2      !-
;  5435     5668  2          STATUS = LIB$FREE_VM ( %REF ( 2 * (.WCB [WCB_W_NO_ROWS] + 1)),
;  5436     5669  2                                 WCB [WCB_A_LINE_CHAR]);
;  5437     5670  2
;  5438     5671  2          IF NOT .STATUS THEN RET_STATUS = .STATUS;    ! Propagate an error
;  5439     5672  2                                                       ! status
;  5440     5673  2
;  5441     5674  2      !+
;  5442     5675  2      ! Deallocate the WCB itself.
;  5443     5676  2      !-
;  5444     5677  2          STATUS = LIB$FREE_VM (%REF(WCB_K_SIZE), WCB);
;  5445     5678  2          IF NOT .STATUS THEN RET_STATUS = .STATUS ; ! Propagate an error status
;  5446     5679  2
;  5447     5680  2          RETURN (.RET_STATUS);
;  5448     5681  2
;  5449     5682  1          END;                          ! Routine SMG$$DEALLOCATE_WCB
```

```
                                    001C 00000          .ENTRY  SMG$$DEALLOCATE_WCB, Save R2,R3,R4      ; 5596
                54 00000000G    00   9E 00002          MOVAB   LIB$FREE_VM, R4                         ;
                5E              04   C2 00009          SUBL2   #4, SP                                  ;
                52                04 AC 00000C         MOVL    WCB, R2                                 ; 5638
                                  08 A2 9F 00010       PUSHAB  8(R2)                                   ;
    04    AE          28 A2           02 78 00013     ASHL    #2, 40(R2), 4(SP)                       ; 5637
                                  04 AE 9F 00019       PUSHAB  4(SP)                                   ;
                64                  02 FB 0001C      CALLS   #2, LIB$FREE_VM                         ; 5638
                53                  50 D0 0001F      MOVL    R0, RET_STATUS                          ;
                                  10 A2 D5 00022       TSTL    16(R2)                                  ; 5644
                                     1E 13 00025       BEQL    2$                                      ;
                                  10 A2 9F 00027       PUSHAB  16(R2)                                  ; 5652
                                  28 A2 9F 0002A       PUSHAB  40(R2)                                  ; 5651
                64                  02 FB 0002D      CALLS   #2, LIB$FREE_VM                         ; 5652
                03                  50 E8 00030      BLBS    STATUS, 1$                              ; 5654
                53                  50 D0 00033      MOVL    STATUS, RET_STATUS                      ;
                                  1C A2 9F 00036 1$:    PUSHAB  28(R2)                                  ; 5658
                                  28 A2 9F 00039       PUSHAB  40(R2)                                  ; 5657
                64                  02 FB 0003C      CALLS   #2, LIB$FREE_VM                         ; 5658
                03                  50 E8 0003F      BLBS    STATUS, 2$                              ; 5660
                53                  50 D0 00042      MOVL    STATUS, RET_STATUS                      ;
```

```
                                                                 2C  A2  9F  00045 2$:     PUSHAB  44(R2)                          ; 5669
                                                             51  02  A2  3C  00048         MOVZWL  2(R2), R1                       ; 5668
                                                         04  AE  51      01  78  0004C     ASHL    #1, R1, 4(SP)                   ;
                                                             04  AE      02  C0  00051     ADDL2   #2, 4(SP)                       ;
                                                                 04  AE  9F  00055         PUSHAB  4(SP)                           ;
                                                             64          02  FB  00058     CALLS   #2, LIB$FREE_VM                 ; 5669
                                                             03          50  E8  0005B     BLBS    STATUS, 3$                      ; 5671
                                                             53          50  D0  0005E     MOVL    STATUS, RET_STATUS              ;
                                                                 04  AC  9F  00061 3$:     PUSHAB  WCB                             ; 5677
                                                         04  AE          34  D0  00064     MOVL    #52, 4(SP)                      ;
                                                                 04  AE  9F  00068         PUSHAB  4(SP)                           ;
                                                             64          02  FB  0006B     CALLS   #2, LIB$FREE_VM                 ;
                                                             03          50  E8  0006E     BLBS    STATUS, 4$                      ; 5678
                                                             53          50  D0  00071     MOVL    STATUS, RET_STATUS              ;
                                                             50          53  D0  00074 4$:     MOVL    RET_STATUS, R0          ; 5680
                                                                         04  00077         RET                                     ; 5682

; Routine Size:  120 bytes,    Routine Base:  _SMG$CODE + 1E04

; 5450          5683  1 !<BLF/PAGE>
```

```
;  5452     5684  1 %SBTTL 'SMG$$DUPL_VIRTUAL_DISPLAY - Duplicate a virtual display'
;  5453     5685  1 GLOBAL ROUTINE SMG$$DUPL_VIRTUAL_DISPLAY (
;  5454     5686  1                                           CURR_DISPLAY_ID,
;  5455     5687  1                                           NEW_DISPLAY_ID
;  5456     5688  1                                         ) =
;  5457     5689  1 !++
;  5458     5690  1 ! FUNCTIONAL DESCRIPTION:
;  5459     5691  1 !
;  5460     5692  1 !       This routine makes a copy of an existing virtual display,
;  5461     5693  1 !       assigning it a new virtual display number.  The new virtual
;  5462     5694  1 !       will not be pasted anywhere -- even if the virtual display from
;  5463     5695  1 !       which it was created was.
;  5464     5696  1 !
;  5465     5697  1 ! CALLING SEQUENCE:
;  5466     5698  1 !
;  5467     5699  1 !       ret_status.wlc.v = SMG$$DUPL_VIRTUAL_DISPLAY (CURR_DISPLAY_ID,
;  5468     5700  1 !                                                     NEW_DISPLAY_ID)
;  5469     5701  1 !
;  5470     5702  1 ! FORMAL PARAMETERS:
;  5471     5703  1 !
;  5472     5704  1 !       CURR_DISPLAY_ID.rl.r    Display id of virtual display to be
;  5473     5705  1 !                               replicated.
;  5474     5706  1 !
;  5475     5707  1 !       NEW_DISPLAY_ID.wl.r     Display id of newly-created virtual
;  5476     5708  1 !                               display.
;  5477     5709  1 !
;  5478     5710  1 ! IMPLICIT INPUTS:
;  5479     5711  1 !
;  5480     5712  1 !       NONE
;  5481     5713  1 !
;  5482     5714  1 ! IMPLICIT OUTPUTS:
;  5483     5715  1 !
;  5484     5716  1 !       NONE
;  5485     5717  1 !
;  5486     5718  1 ! COMPLETION STATUS:
;  5487     5719  1 !
;  5488     5720  1 !       SS$_NORMAL      Normal successful completion
;  5489     5721  1 !       LIB$_INSVIRMEM  Insufficient virtual memory to allocate needed
;  5490     5722  1 !                       buffer.
;  5491     5723  1 !
;  5492     5724  1 ! SIDE EFFECTS:
;  5493     5725  1 !
;  5494     5726  1 !       NONE
;  5495     5727  1 !--
;  5496     5728  2     BEGIN
;  5497     5729  2     LOCAL
;  5498     5730  2         DCB     : REF $DCB_DECL,        ! Address of current DCB.
;  5499     5731  2         DCB_NEW : REF $DCB_DECL,        ! Address of new DCB
;  5500     5732  2         STATUS;         ! Status of subroutine calls
;  5501     5733  2
;  5502     5734  2     $SMG$GET_DCB (.CURR_DISPLAY_ID, DCB); ! Get addr of DCB for current
;  5503     5735  2                                         ! display
;  5504     5736  2
;  5505     5737  2 !+
;  5506     5738  2 ! If a backup DCB does not yet exist, allocate one.
;  5507     5739  2 ! Make a new virtual display using the sizes and attributes of the old
;  5508     5740  2 ! one.  Quit if we can't.
```

```
;  5509          5741  2 !-
;  5510          5742  2
;  5511          5743  2     IF .DCB [DCB_A_BACKUP_DCB] EQL 0
;  5512          5744  2     THEN
;  5513          5745  3         BEGIN   ! 1st time, create the backup
;  5514          5746  4         IF NOT (STATUS = SMG$$CREATE_VIRTUAL_DISPLAY (
;  5515          5747  4                         %REF (.DCB [DCB_W_NO_ROWS]),              ! #rows
;  5516          5748  4                         %REF (.DCB [DCB_W_NO_COLS]),              ! #cols
;  5517          5749  4                         .NEW_DISPLAY_ID,                          ! new id
;  5518          5750  4                         %REF (.DCB [DCB_B_DEF_DISPLAY_ATTR]),     ! disp
;  5519          5751  4                         %REF (.DCB [DCB_B_DEF_VIDEO_ATTR]),       ! video
;  5520          5752  3                         %REF (.DCB [DCB_B_DEF_CHAR_SET])))        ! alt char set
;  5521          5753  3         THEN
;  5522          5754  3             RETURN (.STATUS);
;  5523          5755  3
;  5524          5756  3         $SMG$GET_DCB (.NEW_DISPLAY_ID, DCB_NEW); ! Get DCB address of new
;  5525          5757  3
;  5526          5758  3         !+
;  5527          5759  3         ! Store the new display id in the new DCB.
;  5528          5760  3         !-
;  5529          5761  3         DCB_NEW [DCB_L_DID] = ..NEW_DISPLAY_ID;
;  5530          5762  3
;  5531          5763  3         END     ! 1st time, create the backup
;  5532          5764  2
;  5533          5765  2     ELSE
;  5534          5766  3         BEGIN   ! Backup already exists
;  5535          5767  3         .NEW_DISPLAY_ID = .DCB [DCB_A_BACKUP_DCB]; ! Return id of existing
;  5536          5768  3         DCB_NEW =         .DCB [DCB_A_BACKUP_DCB];
;  5537          5769  3         END;    ! Backup already exists
;  5538          5770  2
;  5539          5771  2     !+
;  5540          5772  2     ! Now need to copy over the current text and attribute buffers from
;  5541          5773  2     ! the current to the new.
;  5542          5774  2     !-
;  5543          5775  2
;  5544          5776  2         CH$MOVE ( .DCB [DCB_L_BUFSIZE],                           ! #bytes
;  5545          5777  2                   .DCB [DCB_A_TEXT_BUF],                          ! from
;  5546          5778  2                   .DCB_NEW [DCB_A_TEXT_BUF]);                     ! to
;  5547          5779  2
;  5548          5780  2         CH$MOVE ( .DCB [DCB_L_BUFSIZE],                           ! #bytes
;  5549          5781  2                   .DCB [DCB_A_ATTR_BUF],                          ! from
;  5550          5782  2                   .DCB_NEW [DCB_A_ATTR_BUF]);                     ! to
;  5551          5783  2
;  5552          5784  2     !+
;  5553          5785  2     ! Copy over the line characteristics vector.
;  5554          5786  2     !-
;  5555          5787  2         CH$MOVE ( .DCB [DCB_W_NO_ROWS] + 1,
;  5556          5788  2                   .DCB [DCB_A_LINE_CHAR],
;  5557          5789  2                   .DCB_NEW [DCB_A_LINE_CHAR]);
;  5558          5790  2
;  5559          5791  2     !+
;  5560          5792  2     ! Copy over stuff relating to borders and labels.
;  5561          5793  2     !-
;  5562          5794  2         IF .DCB_NEW [DCB_V_BORDERED]
;  5563          5795  2         THEN
;  5564          5796  3             BEGIN   ! Bordered
;  5565          5797  3             LOCAL
                                        DESC : REF BLOCK [8,BYTE];  ! Pointer to dynamic string
```

```
;  5566     5798  3                                          ! descriptor in DCB for border
;  5567     5799  3                                          ! label
;  5568     5800  3
;  5569     5801  3          DESC = DCB [DCB_Q_LABEL_DESC];
;  5570     5802  3
;  5571     5803  3          !+
;  5572     5804  3          ! If label exists, make a copy.
;  5573     5805  3          !-
;  5574     5806  3          IF .DESC [DSC$A_POINTER] NEQ 0
;  5575     5807  3          THEN
;  5576     5808  4              BEGIN       ! Labeled
;  5577     5809  5              IF NOT (STATUS = LIB$SCOPY_DXDX ( .DESC,
;  5578     5810  5                                                DCB_NEW [DCB_Q_LABEL_DESC] ))
;  5579     5811  4              THEN
;  5580     5812  4                  RETURN (.STATUS);
;  5581     5813  4
;  5582     5814  4              DCB_NEW [DCB_W_LABEL_UNITS]    = .DCB [DCB_W_LABEL_UNITS];
;  5583     5815  4              DCB_NEW [DCB_B_LABEL_POS]      = .DCB [DCB_B_LABEL_POS];
;  5584     5816  4              DCB_NEW [DCB_B_LABEL_CHAR_SET] = .DCB [DCB_B_LABEL_CHAR_SET];
;  5585     5817  4              DCB_NEW [DCB_V_LABEL_CENTER]   = .DCB [DCB_V_LABEL_CENTER];
;  5586     5818  4
;  5587     5819  3              END;        ! Labeled
;  5588     5820  2          END;    ! Bordered
;  5589     5821  2
;  5590     5822  2      !+
;  5591     5823  2      ! If alternate character set buffer involved, copy it over as well.
;  5592     5824  2      !-
;  5593     5825  2          IF .DCB_NEW [DCB_A_CHAR_SET_BUF] NEQ 0
;  5594     5826  2          THEN
;  5595     5827  3              BEGIN   ! Alt char set buffer involved
;  5596     5828  4              IF NOT (STATUS = LIB$GET_VM (DCB [DCB_L_BUFSIZE],
;  5597     5829  4                                           DCB_NEW [DCB_A_CHAR_SET_BUF]))
;  5598     5830  3              THEN
;  5599     5831  3                  RETURN (.STATUS);
;  5600     5832  3
;  5601     5833  3              CH$MOVE (.DCB [DCB_L_BUFSIZE],                  ! Num.
;  5602     5834  3                       .DCB [DCB_A_CHAR_SET_BUF],             ! From
;  5603     5835  3                       .DCB_NEW [DCB_A_CHAR_SET_BUF]);        ! To
;  5604     5836  3
;  5605     5837  2              END;    ! Alt char set buffer involved
;  5606     5838  2
;  5607     5839  2      !+
;  5608     5840  2      ! Also preserve the current cursor postion.
;  5609     5841  2      !-
;  5610     5842  2          DCB_NEW [DCB_W_CURSOR_ROW] = .DCB [DCB_W_CURSOR_ROW];
;  5611     5843  2          DCB_NEW [DCB_W_CURSOR_COL] = .DCB [DCB_W_CURSOR_COL];
;  5612     5844  2
;  5613     5845  2          RETURN (SS$_NORMAL);
;  5614     5846  1          END;                            ! Routine SMG$$DUPL_VIRTUAL_DISPLAY



                                    03FC 00000          .ENTRY  SMG$$DUPL_VIRTUAL_DISPLAY, Save R2,R3,R4,-  ; 5685
                                                                R5,R6,R7,R8,R9                                  ;
                         59 00000000G   8F   D0 00002   MOVL    #SMG$_INVDIS_ID, R9                          ;
```

```
                         5E        14  C2 00009          SUBL2    #20, SP                                      :
                         50    04  BC  D0 0000C          MOVL     @CURR_DISPLAY_ID, R0                         : 5734
                 04      BC    38  A0  D1 00010          CMPL     56(R0), @CURR_DISPLAY_ID                     :
                               06  12 00015              BNEQ     1$                                           :
                         11    44  A0  91 00017          CMPB     68(R0), #17                                  :
                               04  13 0001B              BEQL     2$                                           :
                         50        59  D0 0001D  1$:     MOVL     R9, R0                                       :
                                   04 00020              RET                                                   :
                         56    04  BC  D0 00021  2$:     MOVL     @CURR_DISPLAY_ID, DCB                        :
                         50    40  A6  D0 00025          MOVL     64(DCB), R0                                  : 5742
                               56  12 00029              BNEQ     5$                                           :
                 10      AE    30  A6  9A 0002B          MOVZBL   48(DCB), 16(SP)                              : 5751
                               10  AE  9F 00030          PUSHAB   16(SP)                                       :
                 10      AE    2E  A6  9A 00033          MOVZBL   46(DCB), 16(SP)                              : 5750
                               10  AE  9F 00038          PUSHAB   16(SP)                                       :
                 10      AE    2F  A6  9A 0003B          MOVZBL   47(DCB), 16(SP)                              : 5749
                               10  AE  9F 00040          PUSHAB   16(SP)                                       :
                               08  AC  DD 00043          PUSHL    NEW_DISPLAY_ID                               : 5748
                 14      AE    06  A6  3C 00046          MOVZWL   6(DCB), 20(SP)                               : 5747
                               14  AE  9F 0004B          PUSHAB   20(SP)                                       :
                 14      AE    02  A6  3C 0004E          MOVZWL   2(DCB), 20(SP)                               : 5746
                               14  AE  9F 00053          PUSHAB   20(SP)                                       :
               FCD7      CF        06  FB 00056          CALLS    #6, SMG$$CREATE_VIRTUAL_DISPLAY              :
                         58        50  D0 0005B          MOVL     R0, STATUS                                   :
                         5D        58  E9 0005E          BLBC     STATUS, 7$                                   : 5745
                         50    08  BC  D0 00061          MOVL     @NEW_DISPLAY_ID, R0                          : 5755
                 08      BC    38  A0  D1 00065          CMPL     56(R0), @NEW_DISPLAY_ID                      :
                               06  12 0006A              BNEQ     3$                                           :
                         11    44  A0  91 0006C          CMPB     68(R0), #17                                  :
                               04  13 00070              BEQL     4$                                           :
                         50        59  D0 00072  3$:     MOVL     R9, R0                                       :
                                   04 00075              RET                                                   :
                         57    08  BC  D0 00076  4$:     MOVL     @NEW_DISPLAY_ID, DCB_NEW                     :
                 38      A7    08  BC  D0 0007A          MOVL     @NEW_DISPLAY_ID, 56(DCB_NEW)                 : 5760
                               07  11 0007F              BRB      6$                                           : 5742
                 08      BC        50  D0 00081  5$:     MOVL     R0, @NEW_DISPLAY_ID                          : 5766
                         57        50  D0 00085          MOVL     R0, DCB_NEW                                  : 5767
     10   B7     10      B6    3C  A6  28 00088  6$:     MOVC3    60(DCB), @16(DCB), @16(DCB_NEW)              : 5777
     14   B7     14      B6    3C  A6  28 0008F          MOVC3    60(DCB), @20(DCB), @20(DCB_NEW)              : 5781
                         50    02  A6  3C 00096          MOVZWL   2(DCB), R0                                   : 5786
                                   50  D6 0009A          INCL     R0                                           :
     4C   B7     4C      B6        50  28 0009C          MOVC3    R0, @76(DCB), @76(DCB_NEW)                   : 5788
                         31    2F  A7  E9 000A2          BLBC     47(DCB_NEW), 8$                              : 5793
                         50    08  A6  9E 000A6          MOVAB    8(R6), DESC                                  : 5801
                               04  A0  D5 000AA          TSTL     4(DESC)                                      : 5806
                               28  13 000AD              BEQL     8$                                           :
                               08  A7  9F 000AF          PUSHAB   8(DCB_NEW)                                   : 5810
                                   50  DD 000B2          PUSHL    DESC                                         :
          00000000G      00        02  FB 000B4          CALLS    #2, LIB$SCOPY_DXDX                           :
                         58        50  D0 000BB          MOVL     R0, STATUS                                   :
                         2E        58  E9 000BE  7$:     BLBC     STATUS, 9$                                   :
                 2C      A7    2C  A6  B0 000C1          MOVW     44(DCB), 44(DCB_NEW)                         : 5814
                 31      A7    31  A6  B0 000C6          MOVW     49(DCB), 49(DCB_NEW)                         : 5815
     50          34      A6    01  02  EF 000CB          EXTZV    #2, #1, 52(DCB), R0                          : 5817
 34  A7                  01    02  50  F0 000D1          INSV     R0, #2, #1, 52(DCB_NEW)                      :
                               18  A7  D5 000D7  8$:     TSTL     24(DCB_NEW)                                  : 5825
                               1E  13 000DA              BEQL     11$                                          :
```

```
                                                  18   A7  9F 000DC          PUSHAB  24(DCB_NEW)                                ; 5829
                                                  3C   A6  9F 000DF          PUSHAB  60(DCB)                                    ; 5828
                          00000000G   00                  02  FB 000E2          CALLS   #2, LIB$GET_VM                             ; 5829
                                      58                  50  D0 000E9          MOVL    R0, STATUS
                                      04                  58  E8 000EC          BLBS    STATUS, 10$
                                      50                  58  D0 000EF 9$:      MOVL    STATUS, R0                                 ; 5831
                                                          04 000F2              RET
                    18   B7       18   B6        3C   A6  28 000F3 10$:      MOVC3   60(DCB), @24(DCB), @24(DCB_NEW)            ; 5835
                                  28   A7        28   A6  D0 000FA 11$:      MOVL    40(DCB), 40(DCB_NEW)                       ; 5842
                                      50                  01  D0 000FF          MOVL    #1, R0                                     ; 5845
                                                          04 00102              RET                                                ; 5846

; Routine Size:  259 bytes,    Routine Base:  _SMG$CODE + 1E7C

; 5615          5847  1 !<BLF/PAGE>
```

```
;  5617          5848  1 %SBTTL 'SMG$$LOCATE_PP - Locate Pasting packet for given display and pasteboard'
;  5618          5849  1 GLOBAL ROUTINE SMG$$LOCATE_PP ( DCB  : REF $DCB_DECL,
;  5619          5850  1                                 PBCB : REF $PBCB_DECL,
;  5620          5851  1                                 PP ) =
;  5621          5852  1 !++
;  5622          5853  1 ! FUNCTIONAL DESCRIPTION:
;  5623          5854  1 !
;  5624          5855  1 !       Locate the address of the pasting packet that joins this
;  5625          5856  1 !       virtual display to this pasteboard.
;  5626          5857  1 !
;  5627          5858  1 ! CALLING SEQUENCE:
;  5628          5859  1 !
;  5629          5860  1 !       ret_status.wlc.v = SMG$$LOCATE_PP (     DCB.rab.r,
;  5630          5861  1 !                                               PBCB.rab.r,
;  5631          5862  1 !                                               PP.wl.r)
;  5632          5863  1 !
;  5633          5864  1 ! FORMAL PARAMETERS:
;  5634          5865  1 !
;  5635          5866  1 !       DCB.rab.r       Address of a virtual display control block.
;  5636          5867  1 !
;  5637          5868  1 !       PBCB.rab.r      Address of a pasteboard control block.
;  5638          5869  1 !
;  5639          5870  1 !       PP.wl.r         Return address of the pasting packet that
;  5640          5871  1 !                       represents the pasting of the given virtual
;  5641          5872  1 !                       display to the given pasteboard control block.
;  5642          5873  1 !
;  5643          5874  1 ! IMPLICIT INPUTS:
;  5644          5875  1 !
;  5645          5876  1 !       None
;  5646          5877  1 !
;  5647          5878  1 ! IMPLICIT OUTPUTS:
;  5648          5879  1 !
;  5649          5880  1 !       None
;  5650          5881  1 !
;  5651          5882  1 ! COMPLETION STATUS:
;  5652          5883  1 !
;  5653          5884  1 !       SS$_NORMAL      Normal successful completion
;  5654          5885  1 !       SMG$_NOTPASTED  Given display is not pasted to given pasteboard
;  5655          5886  1 !
;  5656          5887  1 ! SIDE EFFECTS:
;  5657          5888  1 !
;  5658          5889  1 !       NONE
;  5659          5890  1 !--
;  5660          5891  2     BEGIN
;  5661          5892  2     LOCAL
;  5662          5893  2         SEARCH_DCB : REF $DCB_DECL,     ! Addr of the DCB we'll actually
;  5663          5894  2                                         ! search for
;  5664          5895  2
;  5665          5896  2         CURR_PP : REF $PP_DECL;         ! Addr of pasting packet being
;  5666          5897  2                                         ! inspected.
;  5667          5898  2
;  5668          5899  2     CURR_PP = .DCB [DCB_A_PP_NEXT];     ! Start with 1st PP in chain
;  5669          5900  2
;  5670          5901  2 !+
;  5671          5902  2 ! If the virtual display is currently batched, the batch level will be non-zero.
;  5672          5903  2 ! This means a match needs to be found on the backup DCB address instead of the
;  5673          5904  2 ! DCB address.
```

```
;  5674          5905  2 !-
;  5675          5906  2     SEARCH_DCB = .DCB;
;  5676          5907  2
;  5677          5908  2     IF .DCB [DCB_L_BATCH_LEVEL] NEQ 0           ! Currently batched
;  5678          5909  2     THEN
;  5679          5910  2         SEARCH_DCB = .DCB [DCB_A_BACKUP_DCB];
;  5680          5911  2
;  5681          5912  2
;  5682          5913  2     WHILE .CURR_PP NEQ DCB [DCB_A_PP_NEXT]      ! While any remain
;  5683          5914  2     DO
;  5684          5915  3         BEGIN   ! Search for packet with matching PBCB addr
;  5685          5916  3         IF .CURR_PP [PP_A_DCB_ADDR]  EQL .SEARCH_DCB    AND
;  5686          5917  3            .CURR_PP [PP_A_PBCB_ADDR] EQL .PBCB
;  5687          5918  3         THEN
;  5688          5919  4             BEGIN       ! Desired packet found
;  5689          5920  4             .PP = .CURR_PP;
;  5690          5921  4             RETURN (SS$_NORMAL);        ! Return success
;  5691          5922  3             END;        ! Desired packet found
;  5692          5923  3
;  5693          5924  3         CURR_PP = .CURR_PP [PP_A_NEXT_DCB]; ! Otherwise step along DCB
;  5694          5925  3                                           ! side of chain
;  5695          5926  2         END;    ! Search for packet with matching PBCB addr
;  5696          5927  2
;  5697          5928  2 !+
;  5698          5929  2 ! If we fall out of the while loop, this virtual display is not pasted
;  5699          5930  2 ! to the specified pasteboard -- according to the pasting packets.
;  5700          5931  2 !-
;  5701          5932  2     .PP = 0;    ! To reduce liklihood someone will try to use it
;  5702          5933  2                 ! and disregard status.
;  5703          5934  2     RETURN (SMG$_NOTPASTED);    ! Return failure
;  5704          5935  1     END;                        ! Routine SMG$$LOCATE_PP


                                    000C 00000         .ENTRY  SMG$$LOCATE_PP, Save R2,R3                      ;  5849
                50          04  AC  D0 00002         MOVL    DCB, R0                                         ;  5899
                51          20  A0  D0 00006         MOVL    32(R0), CURR_PP                                 ;
                53              50  D0 0000A         MOVL    R0, SEARCH_DCB                                  ;  5906
                            1C  A0  D5 0000D         TSTL    28(R0)                                          ;  5908
                                04  13 00010         BEQL    1$                                              ;
                53          40  A0  D0 00012         MOVL    64(R0), SEARCH_DCB                              ;  5910
                52          20  A0  9E 00016 1$:     MOVAB   32(R0), R2                                      ;  5913
                52                  51  D1 0001A     CMPL    CURR_PP, R2                                     ;
                                    1A  13 0001D     BEQL    3$                                              ;
                53          10  A1  D1 0001F         CMPL    16(CURR_PP), SEARCH_DCB                         ;  5916
                                    0F  12 00023     BNEQ    2$                                              ;
             08 AC          14  A1  D1 00025         CMPL    20(CURR_PP), PBCB                               ;  5917
                                    08  12 0002A     BNEQ    2$                                              ;
             0C BC                  51  D0 0002C     MOVL    CURR_PP, @PP                                    ;  5920
                50                  01  D0 00030     MOVL    #1, R0                                          ;  5921
                                        04 00033     RET                                                     ;
                51                  61  D0 00034 2$:     MOVL    (CURR_PP), CURR_PP                          ;  5924
                                    DD  11 00037     BRB     1$                                              ;  5913
                            0C  BC  D4 00039 3$:     CLRL    @PP                                             ;  5932
                50 00000000G    8F  D0 0003C         MOVL    #SMG$_NOTPASTED, R0                             ;  5934
```

```
                                        04 00043        RET                                             ; 5935

; Routine Size:  68 bytes,    Routine Base:  _SMG$CODE + 1F7F

; 5705          5936  1 !<BLF/PAGE>
```

```
;  5707          5937  1 %SBTTL 'SMG$$PASTE_VIRTUAL_DISPLAY - Paste virtual display to pasteboard'
;  5708          5938  1 GLOBAL ROUTINE SMG$$PASTE_VIRTUAL_DISPLAY (
;  5709          5939  1                                           DCB : REF $DCB_DECL,
;  5710          5940  1                                           PBCB : REF $PBCB_DECL,
;  5711          5941  1                                           PASTEBOARD_ROW,
;  5712          5942  1                                           PASTEBOARD_COL
;  5713          5943  1                                         ) =
;  5714          5944  1 !++
;  5715          5945  1 ! FUNCTIONAL DESCRIPTION:
;  5716          5946  1 !
;  5717          5947  1 !       The specified virtual display is "pasted" (oriented
;  5718          5948  1 !       with respect to) a pasteboard.  This makes the display visible.
;  5719          5949  1 !       This is the inner paste routine.  It assumes input parameters
;  5720          5950  1 !       are all present and valid.  Further assumes that display
;  5721          5951  1 !       specified by DCB is not already pasted to pasteboard specified
;  5722          5952  1 !       by PBCB.
;  5723          5953  1 !
;  5724          5954  1 ! CALLING SEQUENCE:
;  5725          5955  1 !
;  5726          5956  1 !       ret_status.wlc.v = SMG$$PASTE_VIRTUAL_DISPLAY (
;  5727          5957  1 !                                                       DCB.rab.r,
;  5728          5958  1 !                                                       PBCB.rab.r,
;  5729          5959  1 !                                                       PASTEBOARD_ROW.rl.r,
;  5730          5960  1 !                                                       PASTEBOARD_COL.rl.r)
;  5731          5961  1 !
;  5732          5962  1 ! FORMAL PARAMETERS:
;  5733          5963  1 !
;  5734          5964  1 !       DCB.rab.r               Address of virtual display to be pasted.
;  5735          5965  1 !
;  5736          5966  1 !       PBCB.rab.r              Address of the pasteboard on
;  5737          5967  1 !                               which the pasting is to take place.
;  5738          5968  1 !
;  5739          5969  1 !       PASTEBOARD_ROW.rl.r     Row on pasteboard which is to contain
;  5740          5970  1 !                               row 1 of the specified virtual display.
;  5741          5971  1 !
;  5742          5972  1 !       PASTEBOARD_COL.rl.r     Column on pasteboard which is to contain
;  5743          5973  1 !                               column 1 of the specified virtual
;  5744          5974  1 !                               display.
;  5745          5975  1 !
;  5746          5976  1 ! IMPLICIT INPUTS:
;  5747          5977  1 !
;  5748          5978  1 !       None
;  5749          5979  1 !
;  5750          5980  1 ! IMPLICIT OUTPUTS:
;  5751          5981  1 !
;  5752          5982  1 !       None
;  5753          5983  1 !
;  5754          5984  1 ! COMPLETION STATUS:
;  5755          5985  1 !
;  5756          5986  1 !       SS$_NORMAL      Normal successful completion
;  5757          5987  1 !
;  5758          5988  1 ! SIDE EFFECTS:
;  5759          5989  1 !
;  5760          5990  1 !       NONE
;  5761          5991  1 !--
;  5762          5992  2     BEGIN
;  5763          5993  2     LOCAL
```

```
;  5764          5994  2          STATUS,                         ! Status of subroutine calls
;  5765          5995  2
;  5766          5996  2          PP      : REF $PP_DECL,         ! Addr of the pasting packet
;  5767          5997  2                                          ! being created.
;  5768          5998  2
;  5769          5999  2          WCB     : REF $WCB_DECL;        ! Addr. of window control block
;  5770          6000  2
;  5771          6001  2      !+
;  5772          6002  2      ! Get space for pasting packet.
;  5773          6003  2      !-
;  5774          6004  2          IF NOT (STATUS = LIB$GET_VM ( %REF (PP_K_SIZE), PP))
;  5775          6005  2          THEN
;  5776          6006  2              RETURN (.STATUS);
;  5777          6007  2
;  5778          6008  2          CH$FILL (0, PP_K_SIZE, .PP);        ! Clear all fields to default 0
;  5779          6009  2
;  5780          6010  2      !+
;  5781          6011  2      ! Initialize pasting packet
;  5782          6012  2      !-
;  5783          6013  2          PP [PP_A_DCB_ADDR]  = .DCB;
;  5784          6014  2          PP [PP_A_PBCB_ADDR] = .PBCB;
;  5785          6015  2          PP [PP_W_ROW]       = ..PASTEBOARD_ROW;
;  5786          6016  2          PP [PP_W_COL]       = ..PASTEBOARD_COL;
;  5787          6017  2
;  5788          6018  2      !+
;  5789          6019  2      ! Plug this packet onto both queues.
;  5790          6020  2      !-
;  5791          6021  2          $SMG$INSERT_AT_HEAD ( PP [PP_A_NEXT_DCB],  DCB [DCB_A_PP_NEXT]);
;  5792          6022  2          $SMG$INSERT_AT_HEAD ( PP [PP_A_NEXT_PBCB], PBCB [PBCB_A_PP_NEXT]);
;  5793          6023  2
;  5794          6024  2      !+
;  5795          6025  2      ! If the display is batched, we want the backpointer in the PP to be
;  5796          6026  2      ! pointing to our backup DCB.
;  5797          6027  2      !-
;  5798          6028  2          IF .DCB [DCB_L_BATCH_LEVEL] NEQ 0
;  5799          6029  2          THEN
;  5800          6030  2              PP [PP_A_DCB_ADDR] = .DCB [DCB_A_BACKUP_DCB];
;  5801          6031  2
;  5802          6032  2      !+
;  5803          6033  2      ! Recalc. occlusions introduced by this new pasting.
;  5804          6034  2      !-
;  5805          6035  2          IF NOT ( STATUS = SMG$$CHECK_OCCLUSION_FIRST ( .PBCB))
;  5806          6036  2          THEN                                                (
;  5807          6037  2              RETURN (.STATUS);
;  5808          6038  2
;  5809          6039  2      !+
;  5810          6040  2      ! Calculate the transformation constants needed to copy this display's
;  5811          6041  2      ! buffers into the associated window's buffers.
;  5812          6042  2      !-
;  5813          6043  2          IF NOT ( STATUS = SMG$$CALC_PASTE_TRANSF (.PP))
;  5814          6044  2          THEN
;  5815          6045  2              RETURN (.STATUS);
;  5816          6046  2
;  5817          6047  2      !+
;  5818          6048  2      ! If pasteboard batching enabled, quit here.
;  5819          6049  2      !-
;  5820          6050  2          IF .PBCB [PBCB_L_BATCH_LEVEL] NEQ 0
```

```
2      THEN
2          RETURN ( SMG$_BATWAS_ON);
2
2  !+
2  ! Force physical display's cursor to be where this virtual display's
2  ! cursor is.
2  ! Chose between current DCB and backup DCB.
2  !-
2      WCB = .PBCB [PBCB_A_WCB];
2
2      IF .DCB [DCB_L_BATCH_LEVEL] EQL 0
2      THEN
3          BEGIN   ! Get from current DCB
3          WCB [WCB_W_CURR_CUR_ROW] = .DCB [DCB_W_CURSOR_ROW] - 1 + .PP [PP_W_ROW];
3          WCB [WCB_W_CURR_CUR_COL] = .DCB [DCB_W_CURSOR_COL] - 1 + .PP [PP_W_COL];
3          END     ! Get from current DCB
3
2      ELSE
2
3          BEGIN   ! Get from backup DCB
3          LOCAL
3              BACK_DCB : REF $DCB_DECL; ! Addr of backup DCB
3
3          BACK_DCB = .DCB [DCB_A_BACKUP_DCB];
3          WCB [WCB_W_CURR_CUR_ROW] = .BACK_DCB [DCB_W_CURSOR_ROW] - 1 + .PP [PP_W_ROW];
3          WCB [WCB_W_CURR_CUR_COL] = .BACK_DCB [DCB_W_CURSOR_COL] - 1 + .PP [PP_W_COL];
2          END;    ! Get from backup DCB
2
2  !+
2  ! Move stuff from virtual display to pasteboard buffer and caused it
2  ! to be output if pasteboard is not batched.
2  !-
2      IF .PBCB [PBCB_L_BATCH_LEVEL] EQL 0
2      THEN
3          BEGIN
3          PBCB [PBCB_W_FIRST_CHANGED_ROW] = 1;
3          PBCB [PBCB_W_LAST_CHANGED_ROW]  = .PBCB [PBCB_B_ROWS];
3          PBCB [PBCB_W_FIRST_CHANGED_COL] = 1;
3          PBCB [PBCB_W_LAST_CHANGED_COL]  = .PBCB [PBCB_W_WIDTH];
3          RETURN (SMG$$FILL_WINDOW_BUFFER (.PP));
2          END;
2
2  !+
2  ! Else just return Batch-was-On status.
2  !-
2      RETURN ( SMG$_BATWAS_ON);
2
1      END;                            ! Routine SMG$$PASTE_VIRTUAL_DISPLAY


                     00FC 00000        .ENTRY  SMG$$PASTE_VIRTUAL_DISPLAY, Save R2,R3,R4,- ; 5938
                                               R5,R6,R7                                    ;
        5E        08  C2 00002        SUBL2   #8, SP                                      ;
             04   AE  9F 00005        PUSHAB  PP                                          ; 6004
```

```
                    04    AE          37  D0 00008       MOVL    #55, 4(SP)                              ;
                                04    AE  9F 0000C       PUSHAB  4(SP)                                   ;
            00000000G    00           02  FB 0000F       CALLS   #2, LIB$GET_VM                          ;
                         57           50  D0 00016       MOVL    R0, STATUS                              ;
                         5B           57  E9 00019       BLBC    STATUS, 2$                              ;
                         56     04    AE  D0 0001C       MOVL    PP, R6                                  ; 6008
     37          00      6E           00  2C 00020       MOVC5   #0, (SP), #0, #55, (R6)                 ;
                                      66     00025                                                       ;
                    10    A6    04    AC  7D 00026       MOVQ    DCB, 16(R6)                             ; 6013
                    18    A6    0C    BC  B0 0002B       MOVW    @PASTEBOARD_ROW, 24(R6)                 ; 6015
                    1A    A6    10    BC  B0 00030       MOVW    @PASTEBOARD_COL, 26(R6)                 ; 6016
                 50 04    AC          20  C1 00035       ADDL3   #32, DCB, R0                            ; 6021
                          60          66  0E 0003A       INSQUE  (R6), (R0)                              ;
                          50    04    AE  D0 0003D       MOVL    PP, R0                                  ; 6022
                    08    BC    08    A0  0E 00041       INSQUE  8(R0), @PBCB                            ;
                          52    04    AC  D0 00046       MOVL    DCB, R2                                 ; 6028
                                1C    A2  D5 0004A       TSTL    28(R2)                                  ;
                                      09  13 0004D       BEQL    1$                                      ;
                          50    04    AE  D0 0004F       MOVL    PP, R0                                  ; 6030
                    10    A0    40    A2  D0 00053       MOVL    64(R2), 16(R0)                          ;
                          53    08    AC  D0 00058 1$:   MOVL    PBCB, R3                                ; 6035
                                      53  DD 0005C       PUSHL   R3                                      ;
                  FA33    CF          01  FB 0005E       CALLS   #1, SMG$$CHECK_OCCLUSION_FIRST          ;
                          57          50  D0 00063       MOVL    R0, STATUS                              ;
                          0E          57  E9 00066       BLBC    STATUS, 2$                              ;
                                04    AE  DD 00069       PUSHL   PP                                      ; 6043
                  F649    CF          01  FB 0006C       CALLS   #1, SMG$$CALC_PASTE_TRANSF              ;
                          57          50  D0 00071       MOVL    R0, STATUS                              ;
                          04          57  E8 00074       BLBS    STATUS, 3$                              ;
                          50          57  D0 00077 2$:   MOVL    STATUS, R0                              ; 6045
                                      04     0007A       RET                                             ;
                          56  00A4    C3  D0 0007B 3$:   MOVL    164(R3), R6                             ; 6050
                                      7C  12 00080       BNEQ    6$                                      ;
                          51    08    A3  D0 00082       MOVL    8(R3), WCB                              ; 6059
                          55    04    AE  D0 00086       MOVL    PP, R5                                  ; 6064
                          54    04    AE  D0 0008A       MOVL    PP, R4                                  ; 6065
                                1C    A2  D5 0008E       TSTL    28(R2)                                  ; 6061
                                      22  12 00091       BNEQ    4$                                      ;
                          50    28    A2  3C 00093       MOVZWL  40(R2), R0                              ; 6064
                          57    18    A5  32 00097       CVTWL   24(R5), R7                              ;
                          50          57  C0 0009B       ADDL2   R7, R0                                  ;
          20     A1       50          01  A3 0009E       SUBW3   #1, R0, 32(WCB)                         ;
                          52    2A    A2  3C 000A3       MOVZWL  42(R2), R2                              ; 6065
                          50    1A    A4  32 000A7       CVTWL   26(R4), R0                              ;
                          52          50  C0 000AB       ADDL2   R0, R2                                  ;
          22     A1       52          01  A3 000AE       SUBW3   #1, R2, 34(WCB)                         ;
                                      24  11 000B3       BRB     5$                                      ; 6061
                          50    40    A2  D0 000B5 4$:   MOVL    64(R2), BACK_DCB                        ; 6074
                          52    28    A0  3C 000B9       MOVZWL  40(BACK_DCB), R2                        ; 6075
                          57    18    A5  32 000BD       CVTWL   24(R5), R7                              ;
                          52          57  C0 000C1       ADDL2   R7, R2                                  ;
          20     A1       52          01  A3 000C4       SUBW3   #1, R2, 32(WCB)                         ;
                          50    2A    A0  3C 000C9       MOVZWL  42(BACK_DCB), R0                        ; 6076
                          52    1A    A4  32 000CD       CVTWL   26(R4), R2                              ;
                          50          52  C0 000D1       ADDL2   R2, R0                                  ;
          22     A1       50          01  A3 000D4       SUBW3   #1, R0, 34(WCB)                         ;
                                      56  D5 000D9 5$:   TSTL    R6                                      ; 6083
```

```
                                21  12 000DB       BNEQ    6$                                  ;
               00A8    C3        01  B0 000DD       MOVW    #1, 168(R3)                         ; 6086
               00AA    C3   5F   A3  9B 000E2       MOVZBW  95(R3), 170(R3)                     ; 6087
               00AC    C3        01  B0 000E8       MOVW    #1, 172(R3)                         ; 6088
               00AE    C3   5A   A3  B0 000ED       MOVW    90(R3), 174(R3)                     ; 6089
                            04   AE  DD 000F3       PUSHL   PP                                  ; 6090
          00000000G    00        01  FB 000F6       CALLS   #1, SMG$$FILL_WINDOW_BUFFER         ;
                                 04  000FD          RET                                         ;
                    50 00000000G 8F  D0 000FE 6$:   MOVL    #SMG$_BATWAS_ON, R0                 ; 6096
                                 04  00105          RET                                         ; 6098

; Routine Size:  262 bytes,    Routine Base:  _SMG$CODE + 1FC3

; 5869        6099  1 !<BLF/PAGE>
```

```
;  5871     6100  1 %SBTTL 'SMG$$RECALC_PP_FIELDS - Recalc. Pasting Packet fields'
;  5872     6101  1 GLOBAL ROUTINE SMG$$RECALC_PP_FIELDS (
;  5873     6102  1                                       DCB  : REF $DCB_DECL
;  5874     6103  1                                     )=
;  5875     6104  1 !++
;  5876     6105  1 ! FUNCTIONAL DESCRIPTION:
;  5877     6106  1 !
;  5878     6107  1 !       This routine recalculates fields in the pasting packet that
;  5879     6108  1 !       need to change.
;  5880     6109  1 !       It walks the chain of pasting packets associated with the
;  5881     6110  1 !       given Display Control Block, updating each.
;  5882     6111  1 !
;  5883     6112  1 ! CALLING SEQUENCE:
;  5884     6113  1 !
;  5885     6114  1 !       ret_status.wlc.v = SMG$$RECALC_PP_FIELDS ( DCB.rab.r )
;  5886     6115  1 !
;  5887     6116  1 ! FORMAL PARAMETERS:
;  5888     6117  1 !
;  5889     6118  1 !       DCB.rab.r       Address of a virtual display control block.
;  5890     6119  1 !
;  5891     6120  1 ! IMPLICIT INPUTS:
;  5892     6121  1 !
;  5893     6122  1 !       None
;  5894     6123  1 !
;  5895     6124  1 ! IMPLICIT OUTPUTS:
;  5896     6125  1 !
;  5897     6126  1 !       None
;  5898     6127  1 !
;  5899     6128  1 ! COMPLETION STATUS:
;  5900     6129  1 !
;  5901     6130  1 !       SS$_NORMAL      Normal successful completion
;  5902     6131  1 !       Statuses returned by SMG$$CHECK_OCCLUSION and
;  5903     6132  1 !                           SMG$$CALC_PASTE_TRANF
;  5904     6133  1 !
;  5905     6134  1 ! SIDE EFFECTS:
;  5906     6135  1 !
;  5907     6136  1 !       NONE
;  5908     6137  1 !--
;  5909     6138  2     BEGIN
;  5910     6139  2     LOCAL
;  5911     6140  2         PP : REF $PP_DECL;              ! Addr. of a pasting packet
;  5912     6141  2
;  5913     6142  2 !+
;  5914     6143  2 ! Step through all associated pasting packets, updating each.
;  5915     6144  2 !-
;  5916     6145  2     PP = .DCB [DCB_A_PP_NEXT];  ! get 1st packet in DCB-oriented chain
;  5917     6146  2     WHILE .PP NEQ DCB [DCB_A_PP_NEXT]   ! While any remain...
;  5918     6147  2     DO
;  5919     6148  3         BEGIN
;  5920     6149  3         LOCAL
;  5921     6150  3             STATUS,                     ! Status of subroutine calls
;  5922     6151  3             PBCB : REF $PBCB_DECL;      ! Addr of pasteboard control blk
;  5923     6152  3
;  5924     6153  3         PBCB = .PP [PP_A_PBCB_ADDR];
;  5925     6154  3         !+
;  5926     6155  3         ! Calculate who occludes who in current pasting chain.
;  5927     6156  3         !-
```

```
;  5928          6157  4          IF NOT (STATUS =SMG$$CHECK_OCCLUSION ( .PBCB))  ! Recalc. occlusion
;  5929          6158  4          THEN
;  5930          6159  4              RETURN (.STATUS);
;  5931          6160  4
;  5932          6161  4          !+
;  5933          6162  4          ! Calculate critical constants used to map virtual displays
;  5934          6163  4          ! to their correct position within the pasteboard buffer.
;  5935          6164  4          !-
;  5936          6165  4          IF NOT ( STATUS = SMG$$CALC_PASTE_TRANSF ( .PP)) ! Clean up packet
;  5937          6166  4          THEN
;  5938          6167  4              RETURN (.STATUS);
;  5939          6168  4
;  5940          6169  4          PP = .PP [PP_A_NEXT_DCB];       ! Step to next packet
;  5941          6170  2          END;
;  5942          6171  2
;  5943          6172  2      RETURN ( SS$_NORMAL);
;  5944          6173  1      END;                            ! Routine SMG$$RECALC_PP_FIELDS



                                        000C 00000          .ENTRY  SMG$$RECALC_PP_FIELDS, Save R2,R3                   ; 6101
                                52      AC   D0 00002       MOVL    DCB, R2                                             ; 6145
                                53  20  A2   D0 00006       MOVL    32(R2), PP
                                50  20  A2   9E 0000A 1$:    MOVAB   32(R2), R0                                          ; 6146
                                50          53   D1 0000E       CMPL    PP, R0
                                            1D   13 00011       BEQL    2$
                                50      14  A3   D0 00013       MOVL    20(PP), PBCB                                    ; 6153
                                            50   DD 00017       PUSHL   PBCB                                            ; 6157
                            F885    CF          01   FB 00019       CALLS   #1, SMG$$CHECK_OCCLUSION
                                    12          50   E9 0001E       BLBC    STATUS, 3$
                                                53   DD 00021       PUSHL   PP                                          ; 6165
                            F58C    CF          01   FB 00023       CALLS   #1, SMG$$CALC_PASTE_TRANSF
                                    08          50   E9 00028       BLBC    STATUS, 3$
                                    53          63   D0 0002B       MOVL    (PP), PP                                   ; 6169
                                                DA   11 0002E       BRB     1$                                         ; 6146
                                    50          01   D0 00030 2$:    MOVL    #1, R0                                     ; 6172
                                                     04 00033 3$:    RET                                                ; 6173

; Routine Size:  52 bytes,    Routine Base:  _SMG$CODE + 20C9

;  5945          6174  1 !<BLF/PAGE>
```

```
;  5947     6175  1 %SBTTL 'SMG$$UNPASTE_VIRTUAL_DISPLAY - Unpaste virtual display from pasteboard'
;  5948     6176  1 GLOBAL ROUTINE SMG$$UNPASTE_VIRTUAL_DISPLAY (
;  5949     6177  1                                                DCB :  REF $DCB_DECL,
;  5950     6178  1                                                PBCB : REF $PBCB_DECL
;  5951     6179  1                                              ) =
;  5952     6180  1 !++
;  5953     6181  1 ! FUNCTIONAL DESCRIPTION:
;  5954     6182  1 !
;  5955     6183  1 !       The specified virtual display is "unpasted" from a pasteboard
;  5956     6184  1 !       if a pasting packet can be found.
;  5957     6185  1 !       This is the inner-most unpasting routine.  It assumes both
;  5958     6186  1 !       parameters are present and that they are valid (not necessarily
;  5959     6187  1 !       that the pasting packet which joins these two exists).
;  5960     6188  1 !
;  5961     6189  1 ! CALLING SEQUENCE:
;  5962     6190  1 !
;  5963     6191  1 !       ret_status.wlc.v = SMG$$UNPASTE_VIRTUAL_DISPLAY (
;  5964     6192  1 !                                                DCB.rab.r,
;  5965     6193  1 !                                                PBCB.rab.r)
;  5966     6194  1 !
;  5967     6195  1 ! FORMAL PARAMETERS:
;  5968     6196  1 !
;  5969     6197  1 !       DCB.rab.r               Address of DCB of virtual display to be
;  5970     6198  1 !                               unpasted.
;  5971     6199  1 !
;  5972     6200  1 !       PBCB.rab.r              Address of the pasteboard from
;  5973     6201  1 !                               which the unpasting is to take place.
;  5974     6202  1 !
;  5975     6203  1 ! IMPLICIT INPUTS:
;  5976     6204  1 !
;  5977     6205  1 !       None
;  5978     6206  1 !
;  5979     6207  1 ! IMPLICIT OUTPUTS:
;  5980     6208  1 !
;  5981     6209  1 !       None
;  5982     6210  1 !
;  5983     6211  1 ! COMPLETION STATUS:
;  5984     6212  1 !
;  5985     6213  1 !       SS$_NORMAL      Normal successful completion
;  5986     6214  1 !       SMG$_NOTPASTED  Specified virtual display is not currently
;  5987     6215  1 !                       pasted to the specified pasteboard.
;  5988     6216  1 !
;  5989     6217  1 ! SIDE EFFECTS:
;  5990     6218  1 !
;  5991     6219  1 !       NONE
;  5992     6220  1 !--
;  5993     6221  2     BEGIN
;  5994     6222  2     LOCAL
;  5995     6223  2         STATUS,                         ! Status of subroutine call
;  5996     6224  2         PP      : REF $PP_DECL;         ! Addr of pasting packet being
;  5997     6225  2                                         ! inspected.
;  5998     6226  2
;  5999     6227  2 !+
;  6000     6228  2 ! Try to find the pasting packet joining this DCB and PBCB.
;  6001     6229  2 ! Exit with SMG$_NOTPASTED if we can't.
;  6002     6230  2 !-
;  6003     6231  2
```

```
;  6004          6232  3     IF NOT (STATUS = SMG$$LOCATE_PP ( .DCB, .PBCB, PP))
;  6005          6233  2     THEN
;  6006          6234  2         RETURN (.STATUS);   ! No common pasting packet exists
;  6007          6235  2
;  6008          6236  2 !+
;  6009          6237  2 ! Located desired packet.  Remove it from both queues.
;  6010          6238  2 !-
;  6011          6239  2     $SMG$REMOVE_FROM_QUEUE ( PP [PP_A_NEXT_DCB] );
;  6012          6240  2     $SMG$REMOVE_FROM_QUEUE ( PP [PP_A_NEXT_PBCB] );
;  6013          6241  2
;  6014          6242  2 !+
;  6015          6243  2 ! Give back the pasting packet space
;  6016          6244  2 !-
;  6017          6245  2     IF NOT (STATUS = LIB$FREE_VM ( %REF(PP_K_SIZE), PP))
;  6018          6246  2     THEN
;  6019          6247  2         RETURN (.STATUS);
;  6020          6248  2
;  6021          6249  2 !+
;  6022          6250  2 ! If other virtual displays are still pasted to this pasteboard, we need
;  6023          6251  2 ! to recalculate their occlusion bits since they may have changed by
;  6024          6252  2 ! removing this virtual display.
;  6025          6253  2 !-
;  6026          6254  2     IF .PBCB [ PBCB_A_PP_NEXT] NEQ PBCB [ PBCB_A_PP_NEXT]
;  6027          6255  2     THEN
;  6028          6256  3         IF NOT ( STATUS = SMG$$CHECK_OCCLUSION ( .PBCB ))
;  6029          6257  2         THEN
;  6030          6258  2             RETURN (.STATUS);
;  6031          6259  2
;  6032          6260  2 !+
;  6033          6261  2 ! Cause pasteboard to reflect this change.
;  6034          6262  2 !-
;  6035          6263  2
;  6036          6264  2     RETURN ( SMG$$CHECK_FOR_OUTPUT_PBCB ( .PBCB ));
;  6037          6265  2
;  6038          6266  1     END;                            ! Routine SMG$$UNPASTE_VIRTUAL_DISPLAY


                                        0004 00000     .ENTRY  SMG$$UNPASTE_VIRTUAL_DISPLAY, Save R2      ; 6176
                                  5E    08   C2 00002     SUBL2   #8, SP                                     ;
                                04 AE   9F   00005     PUSHAB  PP                                         ; 6232
                                  7E    04   AC 7D 00008     MOVQ    DCB, -(SP)                            ;
                              FE71      CF   03   FB 0000C     CALLS   #3, SMG$$LOCATE_PP                    ;
                                  3C        50   E9 00011     BLBC    STATUS, 2$                            ;
                                  51    04   BE 0F 00014     REMQUE  @PP, FOO                              ; 6239
                           51     04 AE      08   C1 00018     ADDL3   #8, PP, R1                            ; 6240
                                  52        61   0F 0001D     REMQUE  (R1), FOO                             ;
                                04 AE   9F   00020     PUSHAB  PP                                         ; 6245
                                04 AE      37   D0 00023     MOVL    #55, 4(SP)                            ;
                                04 AE   9F   00027     PUSHAB  4(SP)                                      ;
                          00000000G     00   02   FB 0002A     CALLS   #2, LIB$FREE_VM                       ;
                                  1C        50   E9 00031     BLBC    STATUS, 2$                            ;
                                08 AC   08   BC   D1 00034     CMPL    @PBCB, PBCB                           ; 6254
                                        0B   13   00039     BEQL    1$                                    ;
                                08 AC   DD   0003B     PUSHL   PBCB                                       ; 6256
```

```
                          F82C   CF            01  FB 0003E          CALLS   #1, SMG$$CHECK_OCCLUSION                ;
                                 0A            50  E9 00043          BLBC    STATUS, 2$                               ;
                                          08   AC  DD 00046 1$:      PUSHL   PBCB                                     ; 6264
                      00000000G  00            01  FB 00049          CALLS   #1, SMG$$CHECK_FOR_OUTPUT_PBCB           ;
                                               04 00050 2$:         RET                                              ; 6266

; Routine Size:  81 bytes,    Routine Base:  _SMG$CODE + 20FD

; 6039          6267  1 !<BLF/PAGE>
```

```
; 6041          6268  1 END                       ! End of module SMG$DISPLAY_LINKS
; 6042          6269  1
; 6043          6270  0 ELUDOM


                                                    .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;   _SMG$DATA                     80  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   _SMG$CODE                   8526  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                           Library Statistics
;
;                                       --------  Symbols  --------       Pages       Processing
;       File                            Total     Loaded   Percent      Mapped      Time
;
;   _$255$DUA18:[SYSLIB]STARLET.L32;1     9776        101         1         581       00:01.0
;   _$255$DUA18:[SMGRTL.OBJ]RTLLIB.L32;1    36          0         0           8       00:00.1
;   _$255$DUA18:[SMGRTL.OBJ]SMGLIB.L32;1   469        152        32          38       00:00.4


;                                   COMMAND QUALIFIERS

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

; Size:         8488 code + 118 data bytes
; Run Time:       02:45.4
; Elapsed Time:   03:15.9
; Lines/CPU Min:   2274
; Lexemes/CPU-Min: 20334
; Memory Used:  435 pages
; Compilation Complete
```

0445 AH-EF71A-SE
VAX/VMS V4.1 SRC LST MCRF UPD

SMGDISLIN
LIS

SMGRTL

SMGSHR
MAP

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

SMGMIN
LIS

SMGDISOUT
LIS