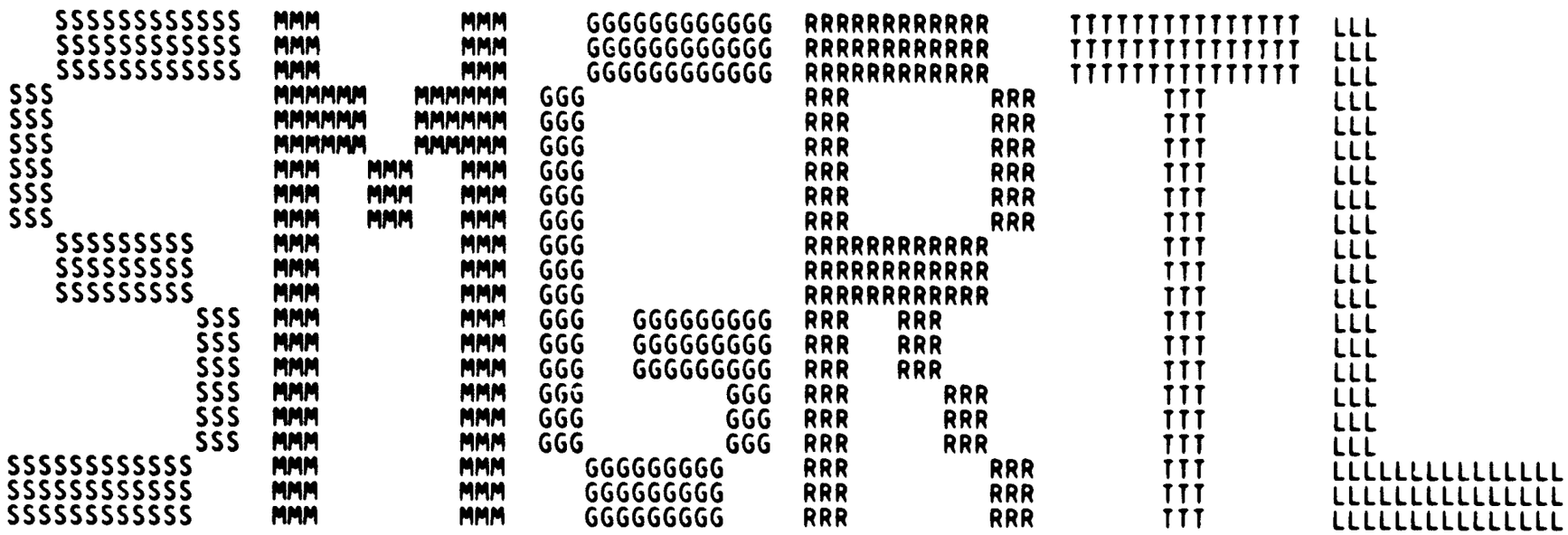
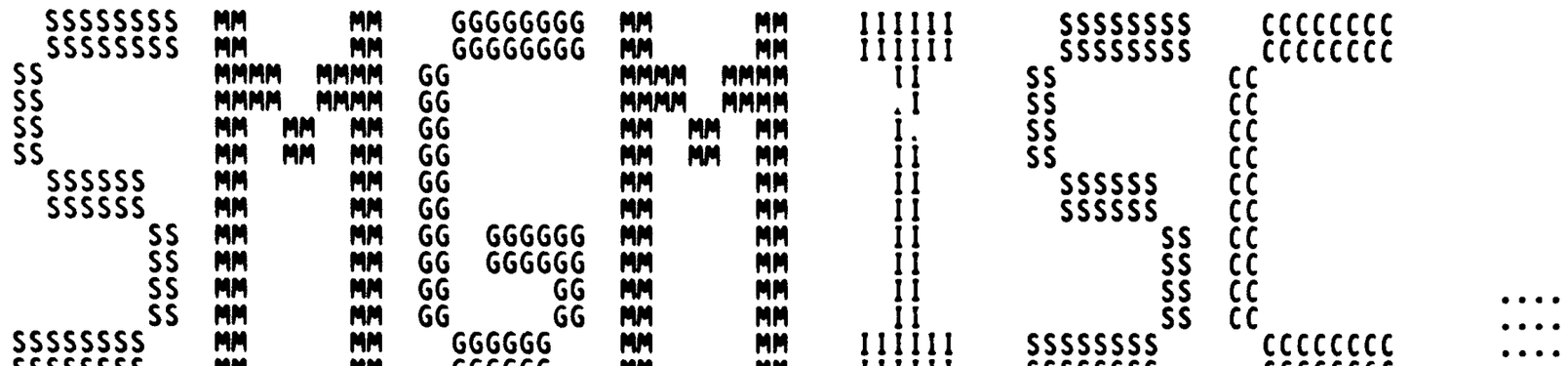


_$2

Val

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


SSSSSSSS MM MM GGGGGGGG MM MM IIIIII SSSSSSSS CCCCCCCC
SSSSSSSS MM MM GGGGGGGG MM MM IIIIII SSSSSSSS CCCCCCCC
SS MMMM MMMM GG MMMM MMMM II SS CC
SS MMMM MMMM GG MMMM MMMM .I SS CC
SS MM MM MM GG MM MM MM I. SS CC
SS MM MM MM GG MM MM MM II SS CC
SSSSSS MM MM GG MM MM II SSSSSS CC
SSSSSS MM MM GG MM MM II SSSSSS CC
SS MM MM GG GGGGGG MM MM II SS CC
SS MM MM GG GGGGGG MM MM II SS CC
SS MM MM GG GG MM MM II SS CC
SS MM MM GG GG MM MM II SS CC
SSSSSSSS MM MM GGGGGG MM MM IIIIII SSSSSSSS CCCCCCCC
SSSSSSSS MM MM GGGGGG MM MM IIIIII SSSSSSSS CCCCCCCC
LL IIIIII SSSSSSSS
LL IIIIII SSSSSSSS
LL II SS
LL II SS
LL II SS
LL II SS
LL II SSSSSS
LL II SSSSSS
LL II SS
LL II SS
LL II SS
LL II SS
LLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLL IIIIII SSSSSSSS

```
;     1          0001  0 %TITLE 'SMG$MISC - Miscellaneous routines for screen mgmt'
;     2          0002  0 MODULE SMG$MISC (
;     3          0003  0                 IDENT = '1-012' ! EDIT: STAN1013
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                      *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    18          0018  1 !*  TRANSFERRED.                                                              *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    22          0022  1 !*  CORPORATION.                                                              *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
```

```
;    31          0030  1 !++
;    32          0031  1 ! FACILITY:     Screen Management
;    33          0032  1 !
;    34          0033  1 ! ABSTRACT:
;    35          0034  1 !
;    36          0035  1 !       This module contains miscellaneous routines
;    37          0036  1 !       having to do with asynchronous terminal operation.
;    38          0037  1 !
;    39          0038  1 ! ENVIRONMENT:  User mode, Shared library routines.
;    40          0039  1 !
;    41          0040  1 ! AUTHOR: Stanley Rabinowitz, CREATION DATE: 3-May-1983
;    42          0041  1 !
;    43          0042  1 ! MODIFIED BY:
;    44          0043  1 !
;    45          0044  1 ! 1-013 Stan    24-Jul-1984     Remove informational errors.
;    46          0045  1 ! 1-012 Stan     4-Apr-1984     Add error message for non-ANSI terminal.
;    47          0046  1 !                               Tighten up error checking in unsolicted input.
;    48          0047  1 ! 1-011 Stan    18-Mar-1984     Allow asking terminal for physical cursor location
;    49          0048  1 !                               Rename some routines.
;    50          0049  1 ! 1-010 Stan    17-Mar-1984     More fixes to documentation
;    51          0050  1 !                               Fix bug in initializing mailbox.
;    52          0051  1 ! 1-009 Stan    21-Feb-1984     Fix documentation
;    53          0052  1 ! 1-008 Stan    27-Nov-1983     Unsolicited input.
;    54          0053  1 ! 1-007 Stan    27-Sep-1983     Get background color.
;    55          0054  1 ! 1-006 Stan    17-Aug-1983     Add items to GET_PASTEBOARD_ATTRIBUTES.
;    56          0055  1 ! 1-005 Stan     7-Jul-1983     Call user's AST routine.
;    57          0056  1 ! 1-004 Stan    13-Jun-1983     Finish mailbox stuff.
;    58          0057  1 ! 1-003 Stan     2-Jun-1983     Add SMG$GET_PASTEBOARD_ATTRIBUTES.
;    59          0058  1 ! 1-002 Stan     1-Jun-1983     Set characteristics
;    60          0059  1 ! 1-001 Stan     3-May-1983     Original version.
;    61          0060  1 !--
```

```
;    63          0061  1 %SBTTL 'Declarations'
;    64          0062  1 !
;    65          0063  1 ! TABLE OF CONTENTS:
;    66          0064  1 !
;    67          0065  1
;    68          0066  1 FORWARD ROUTINE
;    69          0067  1
;    70          0068  1         SMG$SET_OUT_OF_BAND_ASTS,       ! Enable out-of-band ASTs
;    71          0069  1         SMG$SET_BROADCAST_TRAPPING,     ! Enable trapping of broadcast messages
;    72          0070  1         SMG$ENABLE_UNSOLICITED_INPUT,   ! Enable unsolicited input
;    73          0071  1         SMG$GET_BROADCAST_MESSAGE,      ! Gets a broadcast message
;    74          0072  1         SMG$GET_PASTEBOARD_ATTRIBUTES,  ! Get information about pasteboard
;    75          0073  1         SMG$GET_PHYSICAL_CURSOR,        ! Get physical cursor position
;    76          0074  1
;    77          0075  1         SMG$$OUT_OF_BAND_HANDLER,       ! Generic out-of-band AST handler
;    78          0076  1         SMG$$SET_TERM_CHARACTERISTICS,  ! Set characteristics
;    79          0077  1                                         ! in a modular fashion
;    80          0078  1
;    81          0079  1         MAILBOX_AST_ROUTINE,            ! Handles messages to our mailbox
;    82          0080  1         INITIALIZE_MAILBOX,             ! Sets up mailbox and queues
;    83          0081  1         CHAR_EXIT_HANDLER;              ! Handler to reset terminal
;    84          0082  1                                         ! characteristics
;    85          0083  1
;    86          0084  1 !
;    87          0085  1 ! SWITCHES:
;    88          0086  1 !
;    89          0087  1 !       In the include files.
;    90          0088  1 !
;    91          0089  1
;    92          0090  1 !
;    93          0091  1 ! LINKAGES:
;    94          0092  1 !
;    95          0093  1 !       In the include files.
;    96          0094  1 !
;    97          0095  1
;    98          0096  1 !
;    99          0097  1 ! INCLUDE FILES:
;   100          0098  1 !
;   101          0099  1
;   102          0100  1 REQUIRE 'SRC$:SMGPROLOG';               ! defines psects, macros,
;   103          0178  1                                         ! structures and terminal defs
;   104          0179  1 REQUIRE 'RTLIN:STRLNK';                 ! linkages
;   105          0364  1
;   106          0365  1 !
;   107          0366  1 ! MACROS:
;   108          0367  1 !
;   109          0368  1
;   110          0369  1 !
;   111          0370  1 ! EQUATED SYMBOLS:
;   112          0371  1 !
;   113          0372  1
;   114          0373  1 LITERAL
;   115          0374  1
;   116          0375  1         SMG$S_CHAR_EXIT_BLOCK   = 9*4;  ! Size of exit block used by
;   117          0376  1                                         ! SMG$$SET_TERMINAL_CHARACTERISTICS
;   118          0377  1                                         ! (in bytes).
;   119          0378  1
```

```
;  120     0379  1 !
;  121     0380  1 ! FIELDS:
;  122     0381  1 !
;  123     0382  1
;  124     0383  1 FIELD
;  125     0384  1
;  126     0385  1         SMGMBX_FIELDS =
;  127     0386  1
;  128     0387  1 SET
;  129     0388  1
;  130     0389  1 SMGMBX_L_FLINK  = [ 0,0,32,0],  ! Forward  link in queue of mailbox blocks
;  131     0390  1 SMGMBX_L_BLINK  = [ 4,0,32,0],  ! Backward link in queue of mailbox blocks
;  132     0391  1 SMGMBX_A_BUFFER = [ 8,0,32,0],  ! Addres of buffer containing received message
;  133     0392  1 SMGMBX_Q_IOSB   = [12,0, 0,0],  ! Copy of 4-word I/O status block
;  134     0393  1 SMGMBX_W_STATUS = [12,0,16,0],  ! Status of mailbox read (from I/O status block)
;  135     0394  1 SMGMBX_W_MSGLEN = [14,0,16,0],  ! Size of message received
;  136     0395  1 SMGMBX_L_SENDER = [16,0,32,0],  ! PID of process that sent message
;  137     0396  1 SMGMBX_W_BUFSIZ = [20,0,16,0],  ! Allocated size of buffer
;  138     0397  1 SMGMBX_W_UNUSED = [22,0,16,0]   ! Reserved for future use
;  139     0398  1
;  140     0399  1 TES;
;  141     0400  1
;  142     0401  1 LITERAL
;  143     0402  1
;  144     0403  1         SMGMBX_S_SIZE   = 24;   ! Size of structure in bytes
;  145     0404  1
;  146     0405  1 MACRO
;  147     0406  1
;  148     0407  1         $SMGMBX_DECL    = BLOCK[SMGMBX_S_SIZE,BYTE] FIELD(SMGMBX_FIELDS) %;
;  149     0408  1
;  150     0409  1 !
;  151     0410  1 ! PSECTS:
;  152     0411  1 !
;  153     0412  1
;  154     0413  1 ! OWN STORAGE:
;  155     0414  1 !
;  156     0415  1 !       NONE    (and shouldn't have any)
;  157     0416  1
;  158     0417  1 LITERAL
;  159     0418  1
;  160     0419  1         MAX_MSG = 300,  ! *****
;  161     0420  1         BUFQUO  = 300;  ! *****
;  162     0421  1
;  163     0422  1 !
;  164     0423  1 ! EXTERNAL REFERENCES:
;  165     0424  1 !
;  166     0425  1
;  167     0426  1 !EXTERNAL ROUTINE
;  168     0427  1 !
;  169     0428  1 !       NONE
```

```
;  171     0429  1 %SBTTL 'SMG$GET_PASTEBOARD_ATTRIBUTES'
;  172     0430  1 GLOBAL ROUTINE SMG$GET_PASTEBOARD_ATTRIBUTES (
;  173     0431  1     PASTEBOARD_ID,P_PIT,P_PIT_SIZE) =
;  174     0432  1
;  175     0433  1 !++
;  176     0434  1 ! FUNCTIONAL DESCRIPTION:
;  177     0435  1 !
;  178     0436  1 !       This routine gets information about a pasteboard and
;  179     0437  1 !       leaves it in a user-supplied area (the PIT), the pasteboard
;  180     0438  1 !       information table.
;  181     0439  1 !
;  182     0440  1 ! CALLING SEQUENCE:
;  183     0441  1 !
;  184     0442  1 !       RET_STATUS.wlc.v = SMG$GET_PASTEBOARD_ATTRIBUTES (
;  185     0443  1 !                                  PASTEBOARD_ID.rl.r,
;  186     0444  1 !                                  P_PIT.wab.r,
;  187     0445  1 !                                  P_PIT_SIZE.rl.r)
;  188     0446  1 !
;  189     0447  1 ! FORMAL PARAMETERS:
;  190     0448  1 !
;  191     0449  1 !       PASTEBOARD_ID.rl.r      Address of pasteboard id
;  192     0450  1 !
;  193     0451  1 !       P_PIT.wab.r             Address of a pasteboard information table.
;  194     0452  1 !                               This is a byte block whose size and field
;  195     0453  1 !                               references are described in SMGDEF.SDL.
;  196     0454  1 !                               It is the caller's responsibility to allocate
;  197     0455  1 !                               the correct size block and pass it's address
;  198     0456  1 !                               to this routine.
;  199     0457  1 !
;  200     0458  1 !       P_PIT_SIZE.rl.r         Address of a longword containing the size of
;  201     0459  1 !                               the PIT (in bytes).  For now, this size must
;  202     0460  1 !                               be exactly correct.  This will allow the
;  203     0461  1 !                               structure to grow in future releases.
;  204     0462  1 !
;  205     0463  1 ! IMPLICIT INPUTS:
;  206     0464  1 !
;  207     0465  1 !       contents of PBCB
;  208     0466  1 !
;  209     0467  1 ! IMPLICIT OUTPUTS:
;  210     0468  1 !
;  211     0469  1 !       PIT gets filled in.
;  212     0470  1 !
;  213     0471  1 ! COMPLETION STATUS:
;  214     0472  1 !
;  215     0473  1 !       SMG$_INVARG     PIT is the wrong size
;  216     0474  1 !       SS$_NORMAL      Normal successful completion
;  217     0475  1 !
;  218     0476  1 ! SIDE EFFECTS:
;  219     0477  1 !
;  220     0478  1 !       NONE
;  221     0479  1 !
;  222     0480  1 !--
```

```
;  224     0481  2 BEGIN
;  225     0482  2
;  226     0483  2 LOCAL
;  227     0484  2
;  228     0485  2         STATUS,
;  229     0486  2         WCB             : REF BLOCK[,BYTE],     ! Window control block
;  230     0487  2         PBCB            : REF BLOCK[,BYTE];     ! Pasteboard control block
;  231     0488  2
;  232     0489  2 BIND
;  233     0490  2
;  234     0491  2         PIT             = .P_PIT        : BLOCK[,BYTE], ! pasteboard info block
;  235     0492  2         PIT_SIZE        = .P_PIT_SIZE;                  ! size of user's PIT
;  236     0493  2
;  237     0494  2 EXTERNAL LITERAL
;  238     0495  2
;  239     0496  2         SMG$_INVARG;            ! PIT is the wrong size
;  240     0497  2
;  241     0498  2 EXTERNAL ROUTINE
;  242     0499  2
;  243     0500  2         SMG$FIND_CURSOR_DISPLAY;
```

```
;  245          0501  2 $SMG$VALIDATE_ARGCOUNT(3,3);
;  246          0502  2
;  247          0503  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  248          0504  2
;  249          0505  2 !+
;  250          0506  2 ! Make sure that the PIT is the correct size.
;  251          0507  2 !-
;  252          0508  2
;  253          0509  2 IF .PIT_SIZE NEQ SMG$S_PASTEBOARD_INFO_BLOCK
;  254          0510  2   THEN RETURN SMG$_INVARG;
;  255          0511  2
;  256          0512  2 WCB=.PBCB[PBCB_A_WCB];
;  257          0513  2
;  258          0514  2 !+
;  259          0515  2 ! Move arguments from the PBCB to the PIT.
;  260          0516  2 !-
;  261          0517  2
;  262          0518  2 PIT[SMG$L_DEVCHAR]      = .PBCB[PBCB_L_DEVCHAR];
;  263          0519  2 PIT[SMG$L_DEVDEPEND]    = .PBCB[PBCB_L_DEVDEPEND];
;  264          0520  2 PIT[SMG$L_DEVDEPEND2]   = .PBCB[PBCB_L_DEVDEPEND2];
;  265          0521  2 PIT[SMG$B_DEVCLASS]     = .PBCB[PBCB_B_CLASS];
;  266          0522  2 PIT[SMG$B_SMG_DEVTYPE]  = .PBCB[PBCB_B_DEVTYPE];
;  267          0523  2 PIT[SMG$B_PHY_DEVTYPE]  = .PBCB[PBCB_B_PHY_DEV_TYPE];
;  268          0524  2 PIT[SMG$B_ROWS]         = .PBCB[PBCB_B_ROWS];
;  269          0525  2 PIT[SMG$W_WIDTH]        = .PBCB[PBCB_W_WIDTH];
;  270          0526  2 PIT[SMG$B_PARITY]       = .PBCB[PBCB_B_PARITY];
;  271          0527  2 PIT[SMG$W_SPEED]        = .PBCB[PBCB_W_SPEED];
;  272          0528  2 PIT[SMG$W_FILL]         = .PBCB[PBCB_W_FILL];
;  273          0529  2 PIT[SMG$B_COLOR]        = .PBCB[PBCB_B_BACKGROUND_COLOR];
;  274          0530  2
;  275          0531  2 !+
;  276          0532  2 ! Move arguments from the WCB into the PIT.
;  277          0533  2 !-
;  278          0534  2
;  279          0535  2 PIT[SMG$W_CURSOR_ROW]   = .WCB[WCB_W_CURR_CUR_ROW];
;  280          0536  2 PIT[SMG$W_CURSOR_COL]   = .WCB[WCB_W_CURR_CUR_COL];
;  281          0537  2
;  282          0538  2 !+
;  283          0539  2 ! Find out which virtual display the cursor is in and
;  284          0540  2 ! leave that in the PIT.
;  285          0541  2 !-
;  286          0542  2
;  287          0543  2 STATUS=SMG$FIND_CURSOR_DISPLAY(.PASTEBOARD_ID,PIT[SMG$L_CURSOR_DID]);
;  288          0544  2 IF NOT .STATUS THEN RETURN .STATUS;
;  289          0545  2
;  290          0546  2 RETURN  SS$_NORMAL
;  291          0547  2
;  292          0548  1 END;


                                                        .TITLE  SMG$MISC SMG$MISC - Miscellaneous routines for
;                                                                       screen mg
                                                        .IDENT  \1-012\

                                                        .EXTRN  SMG$_INVARG, SMG$FIND_CURSOR_DISPLAY
                                                        .EXTRN  SMG$_WRONUMARG, SMG$_INVPAS_ID
                                                        .EXTRN  PBD_C_COUNT, PBD_A_PBCB
```

```
                                                        .EXTRN  PBD_V_PB_AVAIL

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

                                        0004 00000      .ENTRY  SMG$GET_PASTEBOARD_ATTRIBUTES, Save R2      ; 0430
                    50       08     AC  D0 00002       MOVL    P_PIT, R0                                   ; 0491
                    03              6C  91 00006       CMPB    (AP), #3                                    ; 0501
                                    08  13 00009       BEQL    1$                                          ;
                    50 00000000G    8F  D0 0000B       MOVL    #SMG$_WRONUMARG, R0                         ;
                                        04 00012       RET                                                 ;
                    51       04     BC  D0 00013 1$:   MOVL    @PASTEBOARD_ID, R1                          ; 0503
                                    11  19 00017       BLSS    2$                                          ;
              00000000G     00        51  D1 00019       CMPL    R1, PBD_L_COUNT                             ;
                                    08  14 00020       BGTR    2$                                          ;
          08 00000000G      00        51  E0 00022       BBS     R1, PBD_V_PB_AVAIL, 3$                      ;
                    50 00000000G    8F  D0 0002A 2$:   MOVL    #SMG$_INVPAS_ID, R0                         ;
                                        04 00031       RET                                                 ;
                    51 00000000G0041    D0 00032 3$:   MOVL    PBD_A_PBCB[R1], PBCB                        ;
                    20       0C     BC  D1 0003A       CMPL    @P_PIT_SIZE, #32                            ; 0509
                                    08  13 0003E       BEQL    4$                                          ;
                    50 00000000G    8F  D0 00040       MOVL    #SMG$_INVARG, R0                            ; 0510
                                        04 00047       RET                                                 ;
                    52       08     A1  D0 00048 4$:   MOVL    8(PBCB), WCB                                ; 0512
                    60       58     A1  7D 0004C       MOVQ    88(PBCB), (R0)                              ; 0518
              08    A0       60     A1  D0 00050       MOVL    96(PBCB), 8(R0)                             ; 0520
              0C    A0       58     A1  90 00055       MOVB    88(PBCB), 12(R0)                            ; 0521
              0D    A0       10     A1  90 0005A       MOVB    16(PBCB), 13(R0)                            ; 0522
              0E    A0       59     A1  90 0005F       MOVB    89(PBCB), 14(R0)                            ; 0523
              0F    A0       5F     A1  90 00064       MOVB    95(PBCB), 15(R0)                            ; 0524
              10    A0       5A     A1  B0 00069       MOVW    90(PBCB), 16(R0)                            ; 0525
              13    A0       11     A1  90 0006E       MOVB    17(PBCB), 19(R0)                            ; 0526
              14    A0     00B4     C1  D0 00073       MOVL    180(PBCB), 20(R0)                           ; 0527
              12    A0     00F9     C1  90 00079       MOVB    249(PBCB), 18(R0)                           ; 0529
              18    A0       20     A2  D0 0007F       MOVL    32(WCB), 24(R0)                             ; 0535
                             1C     A0  9F 00084       PUSHAB  28(R0)                                      ; 0543
                             04     AC  D0 00087       PUSHL   PASTEBOARD_ID                               ;
              00000000G     00        02  FB 0008A       CALLS   #2, SMG$FIND_CURSOR_DISPLAY                 ;
                            03        50  E9 00091       BLBC    STATUS, 5$                                  ; 0544
                            50        01  D0 00094       MOVL    #1, R0                                      ; 0546
                                        04 00097 5$:   RET                                                 ; 0548

; Routine Size:  152 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
;  294          0549  1 %SBTTL 'SMG$GET_PHYSICAL_CURSOR'
;  295          0550  1 GLOBAL ROUTINE SMG$GET_PHYSICAL_CURSOR (
;  296          0551  1     PASTEBOARD_ID,P_ROW,P_COL,P_PHYS_ROW,P_PHYS_COL) =
;  297          0552  1
;  298          0553  1 !++
;  299          0554  1 ! FUNCTIONAL DESCRIPTION:
;  300          0555  1 !
;  301          0556  1 !       This routine returns information about where
;  302          0557  1 !       the physical cursor is on the pasteboard.
;  303          0558  1 !       In case SMG is unsure of the cursor location (for example
;  304          0559  1 !       if the pasteboard is created without clearing the screen),
;  305          0560  1 !       there is also an option to inquire of the terminal
;  306          0561  1 !       finding out where it thinks the cursor is.
;  307          0562  1 !
;  308          0563  1 ! CALLING SEQUENCE:
;  309          0564  1 !
;  310          0565  1 !       RET_STATUS.wlc.v = SMG$GET_PHYSICAL_CURSOR (
;  311          0566  1 !                               PASTEBOARD_ID.rl.r,
;  312          0567  1 !                               [,P_ROW.wl.r]
;  313          0568  1 !                               [,P_COL.wl.r]
;  314          0569  1 !                               [,P_PHYS_ROW.wl.r]
;  315          0570  1 !                               [,P_PHYS_COL.wl.r])
;  316          0571  1 !
;  317          0572  1 ! FORMAL PARAMETERS:
;  318          0573  1 !
;  319          0574  1 !       PASTEBOARD_ID.rl.r      Address of pasteboard id
;  320          0575  1 !
;  321          0576  1 !       P_ROW.wl.r              Gets SMG's idea of what row cursor is in
;  322          0577  1 !                               0 means it does not know.
;  323          0578  1 !
;  324          0579  1 !       P_COL.wl.r              Gets SMG's idea of what col cursor is in
;  325          0580  1 !                               0 means it does not know.
;  326          0581  1 !
;  327          0582  1 !       P_PHYS_ROW.wl.r         Gets terminal's idea of what row cursor is in
;  328          0583  1 !
;  329          0584  1 !       P_PHYS_COL.wl.r         Gets terminal's idea of what col cursor is in
;  330          0585  1 !                               If this and the previous arguments are omitted,
;  331          0586  1 !                               then no inquiry to the terminal is made
;  332          0587  1 !                               and no typeahead is purged.
;  333          0588  1 !
;  334          0589  1 ! IMPLICIT INPUTS:
;  335          0590  1 !
;  336          0591  1 !       contents of PBCB
;  337          0592  1 !
;  338          0593  1 ! IMPLICIT OUTPUTS:
;  339          0594  1 !
;  340          0595  1 !       SMG's idea of physical cursor location may get updated
;  341          0596  1 !
;  342          0597  1 ! COMPLETION STATUS:
;  343          0598  1 !
;  344          0599  1 !       SMG$_TRMNOTANS  Terminal is not ansi
;  345          0600  1 !       SS$_NORMAL      Normal successful completion
;  346          0601  1 !                       (includes case where position is not known)
;  347          0602  1 !
;  348          0603  1 ! SIDE EFFECTS:
;  349          0604  1 !
;  350          0605  1 !       Type ahead may be cancelled.
```

```
;  351          0606  1 !
;  352          0607  1 !--
```

```
;  354     0608  2 BEGIN
;  355     0609  2
;  356     0610  2 LOCAL
;  357     0611  2
;  358     0612  2         STATUS,
;  359     0613  2         WCB             : REF BLOCK[,BYTE],     ! Window control block
;  360     0614  2         PBCB            : REF BLOCK[,BYTE];     ! Pasteboard control block
;  361     0615  2
;  362     0616  2 BUILTIN
;  363     0617  2
;  364     0618  2         NULLPARAMETER;
;  365     0619  2
;  366     0620  2 BIND
;  367     0621  2
;  368     0622  2         ROW             = .P_ROW,
;  369     0623  2         COL             = .P_COL,
;  370     0624  2         PHYS_ROW        = .P_PHYS_ROW,
;  371     0625  2         PHYS_COL        = .P_PHYS_COL;
;  372     0626  2
;  373     0627  2 EXTERNAL LITERAL
;  374     0628  2
;  375     0629  2         SMG$_TRMNOTANS, ! Terminal is not ANSI
;  376     0630  2         SMG$_INVARG;
;  377     0631  2
;  378     0632  2 OWN
;  379     0633  2
;  380     0634  2         INQUIRY         : VECTOR[4,BYTE] INITIAL( BYTE(27,'[6n') );
;  381     0635  2
;  382     0636  2                         ! ANSI Cursor position report
```

```
;  384          0637  2 $SMG$VALIDATE_ARGCOUNT(1,5);
;  385          0638  2
;  386          0639  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  387          0640  2
;  388          0641  2 WCB=.PBCB[PBCB_A_WCB];
;  389          0642  2
;  390          0643  2 !+
;  391          0644  2 ! Move physical cursor row and column from the WCB to the user, if requested.
;  392          0645  2 !-
;  393          0646  2
;  394          0647  2 IF NOT NULLPARAMETER(P_ROW)
;  395          0648  2   THEN  ROW = .WCB[WCB_W_CURR_CUR_ROW];
;  396          0649  2 IF NOT NULLPARAMETER(P_COL)
;  397          0650  2   THEN  COL = .WCB[WCB_W_CURR_CUR_COL];
;  398          0651  2
;  399          0652  2 !+
;  400          0653  2 ! If the user requested that we interrogate the terminal, then go do that now.
;  401          0654  2 !-
;  402          0655  2
;  403          0656  2 IF NOT NULLPARAMETER(P_PHYS_ROW)
;  404          0657  2 OR NOT NULLPARAMETER(P_PHYS_COL)
;  405          0658  3   THEN  BEGIN   ! Interrogate terminal
;  406          0659  3
;  407          0660  3         LOCAL
;  408          0661  3
;  409          0662  3                 PTR,
;  410          0663  3                 SEQ             : REF VECTOR[,BYTE],
;  411          0664  3                 ROW_START,
;  412          0665  3                 ROW_LEN,
;  413          0666  3                 COL_START,
;  414          0667  3                 COL_LEN,
;  415          0668  3                 RESP_BUFFER     : VECTOR[SMG$K_LONGEST_SEQUENCE,BYTE],
;  416          0669  3                 IOSB            : VECTOR[4,WORD],
;  417          0670  3                 TERM_ROW : VOLATILE,    ! physical row as reported by terminal
;  418          0671  3                 TERM_COL : VOLATILE;    ! physical col as reported by terminal
;  419          0672  3
;  420          0673  3         !+
;  421          0674  3         ! Check that the terminal is really ANSI and that it supports
;  422          0675  3         ! the cursor position report.
;  423          0676  3         !-
;  424          0677  3
;  425          0678  4         BEGIN
;  426          0679  4         BIND TT2=PBCB[PBCB_L_DEVDEPEND2] : $BBLOCK;
;  427          0680  4         IF NOT  TT2[TT2$V_ANSICRT] THEN RETURN SMG$_TRMNOTANS
;  428          0681  3         END;
;  429          0682  3
;  430          0683  3         !+
;  431          0684  3         ! Issue a QIO to the terminal, sending it the standard
;  432          0685  3         ! ANSI sequence asking for the row and column of the cursor,
;  433          0686  3         ! and getting back a respnse in RESP_BUFFER.
;  434          0687  3         ! Purge type-ahead first, and allow the read to time out.
;  435          0688  3         !-
;  436          0689  3
;  437        P 0690  3         STATUS=$QIOW(   FUNC    = IO$_READPROMPT OR IO$M_TIMED OR
;  438        P 0691  3                           IO$M_PURGE     OR IO$M_ESCAPE,
;  439        P 0692  3                        CHAN    = .PBCB[PBCB_W_CHAN],
;  440        P 0693  3                        EFN     = .PBCB[PBCB_B_EFN],
```

```
;  441        P 0694  3                        IOSB    = IOSB,
;  442        P 0695  3                        P1      = RESP_BUFFER,                  ! gets reply
;  443        P 0696  3                        P2      = %ALLOCATION(RESP_BUFFER),
;  444        P 0697  3                        P3      = 4,                            ! wait 4 seconds
;  445        P 0698  3                        P5      = INQUIRY,                      ! inquire sequence
;  446          0699  3                        P6      = %ALLOCATION(INQUIRY) );
;  447          0700  3
;  448          0701  3              IF .STATUS THEN STATUS=.IOSB[0];
;  449          0702  3              IF NOT .STATUS THEN RETURN .STATUS;
;  450          0703  3
;  451          0704  3              !+
;  452          0705  3              ! If the read succeeds, then IOSB[3] has the number of
;  453          0706  3              ! characters in the escape sequence read, and
;  454          0707  3              ! IOSB[1] points to the beginning of this escape sequence.
;  455          0708  3              ! The format of the answer is "$ [ r ; c R" where r and c
;  456          0709  3              ! are the row and column in decimal-coded ASCII.
;  457          0710  3              ! Start by pointing past the escape and the "[" and then parse a number.
;  458          0711  3              !-
;  459          0712  3
;  460          0713  3              ROW_START=RESP_BUFFER[.IOSB[1]+2];
;  461          0714  3
;  462          0715  3              PTR=CH$FIND_CH(.IOSB[3]-1,.ROW_START,';');
;  463          0716  3              IF CH$FAIL(.PTR)
;  464          0717  3                THEN  RETURN  0;
;  465          0718  3              ROW_LEN=.PTR-.ROW_START;
;  466          0719  3
;  467          0720  3              COL_START=.PTR+1;
;  468          0721  3              PTR=CH$FIND_CH(.IOSB[3]-1,.COL_START,'R');
;  469          0722  3              IF CH$FAIL(.PTR)
;  470          0723  3                THEN  RETURN  0;
;  471          0724  3              COL_LEN=.PTR-.COL_START;
;  472          0725  3
;  473          0726  3              !+
;  474          0727  3              ! Convert the decimal-coded ASCII to a number.
;  475          0728  3              !-
;  476          0729  3
;  477          0730  3              SEQ=.ROW_START;
;  478          0731  3              PHYS_ROW=0;
;  479          0732  3              INCR I FROM 0 TO .ROW_LEN-1 DO
;  480          0733  3                      PHYS_ROW=.PHYS_ROW*10+.SEQ[.I]-%C'0';
;  481          0734  3
;  482          0735  3              SEQ=.COL_START;
;  483          0736  3              PHYS_COL=0;
;  484          0737  3              INCR I FROM 0 TO .COL_LEN-1 DO
;  485          0738  3                      PHYS_COL=.PHYS_COL*10+.SEQ[.I]-%C'0';
;  486          0739  3
;  487          0740  3              !+
;  488          0741  3              ! Pass the row and column back to the user, if requested.
;  489          0742  3              !-
;  490          0743  3
;  491          0744  3              IF NOT NULLPARAMETER(PHYS_ROW)
;  492          0745  3                THEN  PHYS_ROW = .TERM_ROW;
;  493          0746  3              IF NOT NULLPARAMETER(PHYS_COL)
;  494          0747  3                THEN  PHYS_COL = .TERM_COL;
;  495          0748  3
;  496          0749  2              END;    ! Interrogate terminal
;  497          0750  2
```

```
;  498          0751  2 RETURN  SS$_NORMAL
;  499          0752  2
;  500          0753  1 END;


                                                        .PSECT  _SMG$DATA,NOEXE,  PIC,2

                                                 1B  00000 INQUIRY:.BYTE   27
                                              6E 36  5B  00001        .ASCII  \[6n\                                   ;

                                                        .EXTRN  SMG$_TRMNOTANS, SYS$QIOW

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

                                                 01FC 00000        .ENTRY  SMG$GET_PHYSICAL_CURSOR, Save R2,R3,R4,R5,- ; 0550
                                                                           R6,R7,R8
                                        5E     FEF0  CE  9E 00002        MOVAB   -272(SP), SP                            ;
                                        56       10  AC  D0 00007        MOVL    P_PHYS_ROW, R6                          ; 0624
                                        55       14  AC  D0 0000B        MOVL    P_PHYS_COL, R5                          ; 0625
                                50      6C           01  83 0000F        SUBB3   #1, (AP), DIFF                          ; 0637
                                        04           50  91 00013        CMPB    DIFF, #4                                ;
                                                     08  1B 00016        BLEQU   1$                                      ;
                                        50 00000000G 8F  D0 00018        MOVL    #SMG$_WRONUMARG, R0                     ;
                                                     04  0001F        RET                                             ;
                                        50       04  BC  D0 00020 1$:    MOVL    @PASTEBOARD_ID, R0                      ; 0639
                                                     11  19 00024        BLSS    2$                                      ;
                                   00000000G     00       50  D1 00026        CMPL    R0, PBD_L_COUNT                         ;
                                                     08  14 0002D        BGTR    2$                                      ;
                                08 00000000G     00       50  E0 0002F        BBS     R0, PBD_V_PB_AVAIL, 3$                  ;
                                        50 00000000G 8F  D0 00037 2$:    MOVL    #SMG$_INVPAS_ID, R0                     ;
                                                     04  0003E        RET                                             ;
                                        51 00000000C0040 D0 0003F 3$:    MOVL    PBD_A_PBCB[R0], PBCB                    ;
                                        50       08  A1  D0 00047        MOVL    8(PBCB), WCB                            ; 0641
                                        02           6C  91 0004B        CMPB    (AP), #2                                ; 0647
                                                     0A  1F 0004E        BLSSU   4$                                      ;
                                                 08  AC  D5 00050        TSTL    8(AP)                                   ;
                                                     05  13 00053        BEQL    4$                                      ;
                                08   BC          20  A0  32 00055        CVTWL   32(WCB), @P_ROW                         ; 0648
                                        03           6C  91 0005A 4$:    CMPB    (AP), #3                                ; 0649
                                                     0A  1F 0005D        BLSSU   5$                                      ;
                                                 0C  AC  D5 0005F        TSTL    12(AP)                                  ;
                                                     05  13 00062        BEQL    5$                                      ;
                                0C   BC          22  A0  32 00064        CVTWL   34(WCB), @P_COL                         ; 0650
                                        04           6C  91 00069 5$:    CMPB    (AP), #4                                ; 0656
                                                     05  1F 0006C        BLSSU   6$                                      ;
                                                 10  AC  D5 0006E        TSTL    16(AP)                                  ;
                                                     0D  12 00071        BNEQ    9$                                      ;
                                        05           6C  91 00073 6$:    CMPB    (AP), #5                                ; 0657
                                                     03  1E 00076        BGEQU   8$                                      ;
                                                 00D2  31 00078 7$:    BRW     20$                                     ;
                                                 14  AC  D5 0007B 8$:    TSTL    20(AP)                                  ;
                                                     F8  13 0007E        BEQL    7$                                      ;
                                        08       63  A1  E8 00080 9$:    BLBS    99(PBCB), 10$                           ; 0680
                                        50 00000000G 8F  D0 00084        MOVL    #SMG$_TRMNOTANS, R0                     ;
                                                     04  0008B        RET                                             ;
                                                     04  DD 0008C 10$:    PUSHL   #4                                      ; 0699
```

J 2

```
                 00000000'  EF  9F 0008E          PUSHAB  INQUIRY                          ;
              7E            04  7D 00094          MOVQ    #4, -(SP)                        ;
              7E       FF   8F  9A 00097          MOVZBL  #255, -(SP)                      ;
                       24   AE  9F 0009B          PUSHAB  RESP_BUFFER                      ;
                            7E  7C 0009E          CLRQ    -(SP)                            ;
                       28   AE  9F 000A0          PUSHAB  IOSB                             ;
              7E     48B7   8F  3C 000A3          MOVZWL  #18615, -(SP)                    ;
              7E       64   A1  3C 000A8          MOVZWL  100(PBCB), -(SP)                 ;
              7E       66   A1  9A 000AC          MOVZBL  102(PBCB), -(SP)                 ;
      00000000G  00             0C  FB 000B0          CALLS   #12, SYS$QIOW                    ;
              04            50  E9 000B7          BLBC    STATUS, 11$                      ; 0701
              50       08   AE  3C 000BA          MOVZWL  IOSB, STATUS                     ;
              01            50  E8 000BE  11$:    BLBS    STATUS, 12$                      ; 0702
                                04 000C1          RET                                      ;
              50       12   AE  9E 000C2  12$:    MOVAB   RESP_BUFFER+2, R0                ; 0713
              52       0A   AE  3C 000C6          MOVZWL  IOSB+2, ROW_START                ;
              52            50  C0 000CA          ADDL2   R0, ROW_START                    ;
              53       0E   AE  3C 000CD          MOVZWL  IOSB+6, R3                       ; 0715
                            53  D7 000D1          DECL    R3                               ;
         62   53            3B  3A 000D3          LOCC    #59, R3, (ROW_START)             ;
                            02  12 000D7          BNEQ    13$                              ;
                            51  D4 000D9          CLRL    R1                               ;
                            51  D5 000DB  13$:    TSTL    PTR                              ; 0716
                            72  13 000DD          BEQL    21$                              ;
         58   51            52  C3 000DF          SUBL3   ROW_START, PTR, ROW_LEN          ; 0718
              54       01   A1  9E 000E3          MOVAB   1(R1), COL_START                 ; 0720
         64   53       52   8F  3A 000E7          LOCC    #82, R3, (COL_START)             ; 0721
                            02  12 000EC          BNEQ    14$                              ;
                            51  D4 000EE          CLRL    R1                               ;
                            51  D5 000F0  14$:    TSTL    PTR                              ; 0722
                            5D  13 000F2          BEQL    21$                              ;
         57   51            54  C3 000F4          SUBL3   COL_START, PTR, COL_LEN          ; 0724
              51            52  D0 000F8          MOVL    ROW_START, SEQ                   ; 0730
                            66  D4 000FB          CLRL    (R6)                             ; 0731
              53            01  CE 000FD          MNEGL   #1, I                            ; 0733
                            0D  11 00100          BRB     16$                              ;
         50   66            0A  C5 00102  15$:    MULL3   #10, (R6), R0                    ;
              52     6341       9A 00106          MOVZBL  (I)[SEQ], R2                     ;
              66  D0 A240       9E 0010A          MOVAB   -48(R2)[R0], (R6)                ;
         EF   53            58  F2 0010F  16$:    AOBLSS  ROW_LEN, I, 15$                  ;
              51            54  D0 00113          MOVL    COL_START, SEQ                   ; 0735
                            65  D4 00116          CLRL    (R5)                             ; 0736
              50            01  CE 00118          MNEGL   #1, I                            ; 0738
                            0D  11 0011B          BRB     18$                              ;
         54   65            0A  C5 0011D  17$:    MULL3   #10, (R5), R4                    ;
              53     6041       9A 00121          MOVZBL  (I)[SEQ], R3                     ;
              65  D0 A344       9E 00125          MOVAB   -48(R3)[R4], (R5)                ;
         EF   50            57  F2 0012A  18$:    AOBLSS  COL_LEN, I, 17$                  ;
  56     6C   08            00  ED 0012E          CMPZV   #0, #8, (AP), R6                 ; 0744
                            09  19 00133          BLSS    19$                              ;
                          6C46  D5 00135          TSTL    (AP)[R6]                         ;
                            04  13 00138          BEQL    19$                              ;
              66       04   AE  D0 0013A          MOVL    TERM_ROW, (R6)                   ; 0745
  55     6C   08            00  ED 0013E  19$:    CMPZV   #0, #8, (AP), R5                 ; 0746
                            08  19 00143          BLSS    20$                              ;
                          6C45  D5 00145          TSTL    (AP)[R5]                         ;
                            03  13 00148          BEQL    20$                              ;
```

```
                    65      6E  D0 0014A          MOVL    TERM_COL, (R5)                  ; 0747
                    50      01  D0 0014D 20$:     MOVL    #1, R0                          ; 0751
                                04 00150          RET                                     ;
                                50  D4 00151 21$:  CLRL    R0                              ; 0753
                                    04 00153      RET                                     ;

; Routine Size:  340 bytes,    Routine Base:  _SMG$CODE + 0098
```

```
;  502     0754  1 %SBTTL 'SMG$SET_BROADCAST_TRAPPING'
;  503     0755  1 GLOBAL ROUTINE SMG$SET_BROADCAST_TRAPPING(
;  504     0756  1     PASTEBOARD_ID,AST_RTN,AST_ARG) =
;  505     0757  1
;  506     0758  1 !++
;  507     0759  1 ! FUNCTIONAL DESCRIPTION:
;  508     0760  1 !
;  509     0761  1 !       This routine enables trapping of messages broadcast
;  510     0762  1 !       to the specified terminal.  If an AST routine is specified
;  511     0763  1 !       the AST routine will be called whenever a broadcast
;  512     0764  1 !       message comes in.  The AST routine will be called
;  513     0765  1 !       with the first argument being the user's AST argument
;  514     0766  1 !       (passed by value).  This can be the address of a structure
;  515     0767  1 !       if the user needs to pass more than 32 bits of information.
;  516     0768  1 !       If more than one terminal is involved, the user may wish
;  517     0769  1 !       to pass along the pasteboard id as his AST argument.
;  518     0770  1 !       The AST routine can get at the message by calling
;  519     0771  1 !       SMG$GET_BROADCAST_MESSAGE.
;  520     0772  1 !       If no AST routine is specified, operation is synchronous and
;  521     0773  1 !       the user can get at broadcast messages by calling
;  522     0774  1 !       SMG$GET_BROADCAST_MESSAGE whenever he wants to check for
;  523     0775  1 !       receipt of a broadcast message.
;  524     0776  1 !
;  525     0777  1 ! CALLING SEQUENCE:
;  526     0778  1 !
;  527     0779  1 !       RET_STATUS.wlc.v = SMG$ENABLE_BROADCAST_TRAPPING (
;  528     0780  1 !                                  PASTEBOARD_ID.rl.r
;  529     0781  1 !                                  [, AST_RTN.szem.r]
;  530     0782  1 !                                  [, AST_ARG.rz.v])
;  531     0783  1 !
;  532     0784  1 ! FORMAL PARAMETERS:
;  533     0785  1 !
;  534     0786  1 !       PASTEBAORD_ID.rl.r      Address of pasteboard id
;  535     0787  1 !
;  536     0788  1 !       AST_RTN.szem.r          Address of AST routine to be called
;  537     0789  1 !                               if a broadcast message comes in
;  538     0790  1 !                               (optional)
;  539     0791  1 !                               If omitted, and if the next argument is
;  540     0792  1 !                               omitted, then no AST routine will
;  541     0793  1 !                               get called.
;  542     0794  1 !
;  543     0795  1 !       AST_ARG.rz.v            optional value to be passed along to
;  544     0796  1 !                               the AST routine
;  545     0797  1 !                               If this is specified, but no AST_RTN
;  546     0798  1 !                               argument is specified, then only the AST
;  547     0799  1 !                               argument changes, but the routine remains
;  548     0800  1 !                               the same.
;  549     0801  1 !
;  550     0802  1 ! IMPLICIT INPUTS:
;  551     0803  1 !
;  552     0804  1 !       channel of associated mailbox
;  553     0805  1 !
;  554     0806  1 ! IMPLICIT OUTPUTS:
;  555     0807  1 !
;  556     0808  1 !       associated mailbox channel may be assigned if not
;  557     0809  1 !       previous;y created.
;  558     0810  1 !
```

```
;  559          0811  1 ! COMPLETION STATUS:
;  560          0812  1 !
;  561          0813  1 !       SS$_NORMAL      Normal successful completion
;  562          0814  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  563          0815  1 !       SMG$_NOT_A_TRM  Success, but device is not a terminal
;  564          0816  1 !       LIB$_xyz        Errors from LIB$ASN_WTH_MBX
;  565          0817  1 !       SS$_xyz         Any error possible from SMG$$SET_TERM_CHARACTERISTICS
;  566          0818  1 !
;  567          0819  1 ! SIDE EFFECTS:
;  568          0820  1 !
;  569          0821  1 !       A mailbox may be created.
;  570          0822  1 !       The terminal characteristics get changed to BRDCSTMBOX
;  571          0823  1 !       and NOBROADCAST. (The old values of these characteristics
;  572          0824  1 !       are saved away so that they can be restored later.)
;  573          0825  1 !
;  574          0826  1 !--
```

```
;  576     0827  2 BEGIN
;  577     0828  2
;  578     0829  2 LOCAL
;  579     0830  2
;  580     0831  2         STATUS,
;  581     0832  2         PBCB            : REF BLOCK[,BYTE];     ! Pasteboard control block
;  582     0833  2
;  583     0834  2 BUILTIN
;  584     0835  2
;  585     0836  2         NULLPARAMETER;
;  586     0837  2
;  587     0838  2 EXTERNAL LITERAL
;  588     0839  2
;  589     0840  2         SMG$_NOT_A_TRM;         ! pasteboard device is not a terminal
```

```
;  591     0841  2 $SMG$VALIDATE_ARGCOUNT(1,3);
;  592     0842  2
;  593     0843  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  594     0844  2
;  595     0845  2 !+
;  596     0846  2 ! Make sure that the device is a terminal.
;  597     0847  2 ! If not, return the qualified success, SMG$_NOT_A_TRM.
;  598     0848  2 !-
;  599     0849  2
;  600     0850  2 IF .PBCB[PBCB_B_CLASS] NEQ DC$_TERM
;  601     0851  2   THEN  RETURN  SMG$_NOT_A_TRM;
;  602     0852  2
;  603     0853  2 !+
;  604     0854  2 ! If we don't already have a mailbox established,
;  605     0855  2 ! then create a new channel to the terminal with an
;  606     0856  2 ! associated mailbox.
;  607     0857  2 !-
;  608     0858  2
;  609     0859  2 IF .PBCB[PBCB_W_ASYNC_CHAN] EQL 0
;  610     0860  3   THEN  BEGIN
;  611     0861  3         STATUS=INITIALIZE_MAILBOX(.PBCB);
;  612     0862  3         IF NOT .STATUS THEN RETURN .STATUS
;  613     0863  2         END;
;  614     0864  2
;  615     0865  2 !+
;  616     0866  2 ! Note that broadcasts are enabled.
;  617     0867  2 !-
;  618     0868  2
;  619     0869  2 PBCB[PBCB_V_BROADCAST]=1;
;  620     0870  2
;  621     0871  2 !+
;  622     0872  2 ! Store the (possibly new) AST routine and argument in our PBCB.
;  623     0873  2 !-
;  624     0874  2
;  625     0875  2 IF NULLPARAMETER(2)
;  626     0876  3   THEN  BEGIN    ! no AST routine specified
;  627     0877  3         IF NULLPARAMETER(3)
;  628     0878  4           THEN  BEGIN    ! cancel ASTs
;  629     0879  4                 PBCB[PBCB_A_BROADCAST_RTN]=0;
;  630     0880  4                 PBCB[PBCB_L_BROADCAST_ARG]=0
;  631     0881  4                 END      ! cancel ASTs
;  632     0882  4           ELSE  BEGIN    ! change AST argument
;  633     0883  4                 PBCB[PBCB_L_BROADCAST_ARG]=.AST_ARG
;  634     0884  3                 END;     ! change AST argument
;  635     0885  3         END      ! no AST routine specified
;  636     0886  3   ELSE  BEGIN    ! New AST routine
;  637     0887  3         PBCB[PBCB_A_BROADCAST_RTN]=.AST_RTN;
;  638     0888  3         IF NULLPARAMETER(3)
;  639     0889  3           THEN  PBCB[PBCB_L_BROADCAST_ARG]=0
;  640     0890  3           ELSE  PBCB[PBCB_L_BROADCAST_ARG]=.AST_ARG
;  641     0891  2         END;     ! New AST routine
;  642     0892  2
;  643     0893  2 RETURN  SS$_NORMAL
;  644     0894  2
;  645     0895  1 END;
```

```
                                                        .EXTRN  SMG$_NOT_A_TRM

                                        0004 00000      .ENTRY  SMG$SET_BROADCAST_TRAPPING, Save R2          ; 0755
          50            6C              01  83 00002         SUBB3   #1, (AP), DIFF                          ; 0841
                        02              50  91 00006         CMPB    DIFF, #2
                                        08  1B 00009         BLEQU   1$
                        50 00000000G    8F  D0 0000B         MOVL    #SMG$_WRONUMARG, R0
                                            04 00012         RET
                        50        04    BC  D0 00013 1$:     MOVL    @PASTEBOARD_ID, R0                      ; 0843
                                        11  19 00017         BLSS    2$
              00000000G 00              50  D1 00019         CMPL    R0, PBD_L_COUNT
                                        08  14 00020         BGTR    2$
          08 00000000G 00               50  E0 00022         BBS     R0, PBD_V_PB_AVAIL, 3$
                        50 00000000G    8F  D0 0002A 2$:     MOVL    #SMG$_INVPAS_ID, R0
                                            04 00031         RET
                        52 00000000G0040    D0 00032 3$:     MOVL    PBD_A_PBCB[R0], PBCB
                  42    8F        58    A2  91 0003A         CMPB    88(PBCB), #66                           ; 0850
                                        08  13 0003F         BEQL    4$
                        50 00000000G    8F  D0 00041         MOVL    #SMG$_NOT_A_TRM, R0                     ; 0851
                                            04 00048         RET
                                 00D2   C2  B5 00049 4$:     TSTW    210(PBCB)                               ; 0859
                                        0A  12 0004D         BNEQ    5$
                                        52  DD 0004F         PUSHL   PBCB                                    ; 0861
                 0000V  CF                  01  FB 00051         CALLS   #1, INITIALIZE_MAILBOX
                        3F                  50  E9 00056         BLBC    STATUS, 12$                         ; 0862
                 00D0   C2                  01  88 00059 5$:     BISB2   #1, 208(PBCB)                       ; 0869
                        50        00BC      C2  9E 0005E         MOVAB   188(PBCB), R0                       ; 0880
                        02                  6C  91 00063         CMPB    (AP), #2                            ; 0875
                                            05  1F 00066         BLSSU   6$
                                  08        AC  D5 00068         TSTL    8(AP)
                                            10  12 0006B         BNEQ    8$
                              03            6C  91 0006D 6$:     CMPB    (AP), #3                            ; 0877
                                            05  1F 00070         BLSSU   7$
                                  0C        AC  D5 00072         TSTL    12(AP)
                                            1A  12 00075         BNEQ    10$
                                 00B8       C2  D4 00077 7$:     CLRL    184(PBCB)                           ; 0879
                                            10  11 0007B         BRB     9$                                  ; 0880
                       00B8     C2        08  AC  D0 0007D 8$:     MOVL    AST_RTN, 184(PBCB)                  ; 0887
                                03            6C  91 00083         CMPB    (AP), #3                            ; 0888
                                            05  1F 00086         BLSSU   9$
                                  0C        AC  D5 00088         TSTL    12(AP)
                                            04  12 0008B         BNEQ    10$
                                            60  D4 0008D 9$:     CLRL    (R0)                                ; 0889
                                            04  11 0008F         BRB     11$
                              60        0C  AC  D0 00091 10$:     MOVL    AST_ARG, (R0)                       ; 0890
                              50            01  D0 00095 11$:     MOVL    #1, R0                              ; 0893
                                                04 00098 12$:     RET                                         ; 0895

; Routine Size:  153 bytes,    Routine Base:  _SMG$CODE + 01EC
```

```
;  647          0896  1 %SBTTL 'INITIALIZE_MAILBOX';
;  648          0897  1 ROUTINE INITIALIZE_MAILBOX(P_PBCB) =
;  649          0898  1
;  650          0899  1 !++
;  651          0900  1 ! FUNCTIONAL DESCRIPTION:
;  652          0901  1 !
;  653          0902  1 !       Creates a mailbox and assigns a channel to it.
;  654          0903  1 !       Initializes a queue of mailbox blocks.
;  655          0904  1 !
;  656          0905  1 ! CALLING SEQUENCE:
;  657          0906  1 !
;  658          0907  1 !       RET_STATUS.wlc.v = INITIALIZE_MAILBOX(P_PBCB.rl.v)
;  659          0908  1 !
;  660          0909  1 ! FORMAL PARAMETERS:
;  661          0910  1 !
;  662          0911  1 !       P_PBCB.rl.v             Address of pasteboard control block.
;  663          0912  1 !
;  664          0913  1 ! IMPLICIT INPUTS:
;  665          0914  1 !
;  666          0915  1 !       MAX_MSG
;  667          0916  1 !       BUFQUO
;  668          0917  1 !
;  669          0918  1 ! IMPLICIT OUTPUTS:
;  670          0919  1 !
;  671          0920  1 !       NONE
;  672          0921  1 !
;  673          0922  1 ! COMPLETION STATUS:
;  674          0923  1 !
;  675          0924  1 !       SS$_NORMAL      Normal successful completion
;  676          0925  1 !       SS$_xyz         Any error possible from $DASSGN or $CANCEL
;  677          0926  1 !
;  678          0927  1 ! SIDE EFFECTS:
;  679          0928  1 !
;  680          0929  1 !       NONE
;  681          0930  1 !--
```

```
;  683          0931  2 BEGIN
;  684          0932  2
;  685          0933  2 BIND
;  686          0934  2
;  687          0935  2         PBCB    = .P_PBCB       : BLOCK[,BYTE];
;  688          0936  2
;  689          0937  2 LOCAL
;  690          0938  2
;  691          0939  2         STATUS,
;  692          0940  2         NAME_DESC       : VECTOR[2];
;  693          0941  2
;  694          0942  2 EXTERNAL ROUTINE
;  695          0943  2
;  696          0944  2         LIB$GET_VM,
;  697          0945  2         LIB$ASN_WTH_MBX;
```

```
;  699          0946  2 !+
;  700          0947  2 ! LIB$ASN_WTH_MBX wants the name as a descriptor,
;  701          0948  2 ! so we build one temporarily.
;  702          0949  2 !-
;  703          0950  2
;  704          0951  2 NAME_DESC[0]=.PBCB[PBCB_W_DEVNAM_LEN];
;  705          0952  2 NAME_DESC[1]= PBCB[PBCB_T_DEVNAM];
;  706          0953  2
;  707          0954  2 STATUS=LIB$ASN_WTH_MBX(NAME_DESC,%REF(MAX_MSG),%REF(BUFQUO),
;  708          0955  2         PBCB[PBCB_Q_ASYNC_CHAN],PBCB[PBCB_W_MBX_CHAN]);
;  709          0956  2 IF NOT .STATUS THEN RETURN .STATUS;
;  710          0957  2
;  711          0958  2 !+
;  712          0959  2 ! Change the terminal characteristics to
;  713          0960  2 ! NOBROADCAST and BRDCSTMBX.
;  714          0961  2 ! Establish an exit handler to restore the old
;  715          0962  2 ! characteristics upon exit.
;  716          0963  2 !-
;  717          0964  2
;  718          0965  2 STATUS=SMG$$SET_TERM_CHARACTERISTICS(PBCB[PBCB_L_PBID],
;  719          0966  2         TT$M_NOBRDCST,TT2$M_BRDCSTMBX);
;  720          0967  2 IF NOT .STATUS THEN RETURN .STATUS;
;  721          0968  2
;  722          0969  2 !+
;  723          0970  2 ! Allocate a buffer to handle reads.
;  724          0971  2 !-
;  725          0972  2
;  726          0973  2 PBCB[PBCB_W_SMGMBX_BUFSIZ]=MAX_MSG;
;  727          0974  2 STATUS=LIB$GET_VM(%REF(.PBCB[PBCB_W_SMGMBX_BUFSIZ]),
;  728          0975  2                  PBCB[PBCB_A_SMGMBX_BUFFER]);
;  729          0976  2 IF NOT .STATUS THEN RETURN .STATUS;
;  730          0977  2
;  731          0978  2 !+
;  732          0979  2 ! Initialize our queue of received mailbox buffers to empty.
;  733          0980  2 !-
;  734          0981  2
;  735          0982  2 PBCB[PBCB_L_SMGMBX_FLINK]=PBCB[PBCB_L_SMGMBX_FLINK];
;  736          0983  2 PBCB[PBCB_L_SMGMBX_BLINK]=PBCB[PBCB_L_SMGMBX_FLINK];
;  737          0984  2
;  738          0985  2 !+
;  739          0986  2 ! Invoke the AST routine with our special INIT flag on,
;  740          0987  2 ! so that it knows that no buffer was read
;  741          0988  2 ! and a read gets initiated.
;  742          0989  2 !-
;  743          0990  2
;  744          0991  2 PBCB[PBCB_V_SMGMBX_INIT]=1;
;  745        P 0992  2 STATUS=$DCLAST( ASTADR  = MAILBOX_AST_ROUTINE,
;  746          0993  2                 ASTPRM  = PBCB);
;  747          0994  2 IF NOT .STATUS THEN RETURN .STATUS;
;  748          0995  2
;  749          0996  2 RETURN  SS$_NORMAL
;  750          0997  2
;  751          0998  1 END;


                                                .EXTRN  LIB$GET_VM, LIB$ASN_WTH_MBX
```

```
                                                              .EXTRN  SYS$DCLAST

                                         0004 00000 INITIALIZE_MAILBOX:
                                                              .WORD   Save R2                                   ; 0897
                5E              10  C2 00002          SUBL2   #16, SP                                   ;
                52              04  AC  D0 00005      MOVL    P_PBCB, R2                                ; 0935
          08    AE              12  A2  3C 00009      MOVZWL  18(R2), NAME_DESC                         ; 0951
          0C    AE              18  A2  9E 0000E      MOVAB   24(R2), NAME_DESC+4                       ; 0952
                              00D4  C2  9F 00013      PUSHAB  212(R2)                                   ; 0955
                              00D2  C2  9F 00017      PUSHAB  210(R2)                                   ;
          0C    AE            012C  8F  3C 0001B      MOVZWL  #300, 12(SP)                              ; 0954
                                0C  AE  9F 00021      PUSHAB  12(SP)                                    ;
          0C    AE            012C  8F  3C 00024      MOVZWL  #300, 12(SP)                              ;
                                0C  AE  9F 0002A      PUSHAB  12(SP)                                    ;
                                18  AE  9F 0002D      PUSHAB  NAME_DESC                                 ;
   00000000G    00                  05  FB 00030      CALLS   #5, LIB$ASN_WTH_MBX                       ; 0955
                58                  50  E9 00037      BLBC    STATUS, 1$                                ; 0956
                                    10  DD 0003A      PUSHL   #16                                       ; 0965
                          00020000  8F  DD 0003C      PUSHL   #131072                                   ;
                                14  A2  9F 00042      PUSHAB  20(R2)                                    ;
       0000V    CF                  03  FB 00045      CALLS   #3, SMG$$SET_TERM_CHARACTERISTICS         ;
                45                  50  E9 0004A      BLBC    STATUS, 1$                                ; 0967
        00D6    C2            012C  8F  B0 0004D      MOVW    #300, 214(R2)                             ; 0973
                              00D8  C2  9F 00054      PUSHAB  216(R2)                                   ; 0975
          08    AE            00D6  C2  3C 00058      MOVZWL  214(R2), 8(SP)                            ; 0974
                                08  AE  9F 0005E      PUSHAB  8(SP)                                     ;
   00000000G    00                  02  FB 00061      CALLS   #2, LIB$GET_VM                            ; 0975
                27                  50  E9 00068      BLBC    STATUS, 1$                                ; 0976
                51            00C8  C2  9E 0006B      MOVAB   200(R2), R1                               ; 0982
                61                  51  D0 00070      MOVL    R1, (R1)                                  ;
        00CC    C2                  51  D0 00073      MOVL    R1, 204(R2)                               ; 0983
        00D0    C2                  04  88 00078      BISB2   #4, 208(R2)                               ; 0991
                                    7E  D4 0007D      CLRL    -(SP)                                     ; 0993
                                    52  DD 0007F      PUSHL   R2                                        ;
                              0000V CF  9F 00081      PUSHAB  MAILBOX_AST_ROUTINE                       ;
   00000000G    00                  03  FB 00085      CALLS   #3, SYS$DCLAST                            ;
                03                  50  E9 0008C      BLBC    STATUS, 1$                                ; 0994
                50                  01  D0 0008F      MOVL    #1, R0                                    ; 0996
                                        04 00092 1$:  RET                                               ; 0998

; Routine Size:  147 bytes,    Routine Base:  _SMG$CODE + 0285
```

```
;  753          0999  1 %SBTTL 'MAILOX_AST_ROUTINE';
;  754          1000  1 ROUTINE MAILBOX_AST_ROUTINE(P_PBCB) =
;  755          1001  1
;  756          1002  1 !++
;  757          1003  1 ! FUNCTIONAL DESCRIPTION:
;  758          1004  1 !
;  759          1005  1 !       This routine processes messages that appear in our mailbox.
;  760          1006  1 !       All it does is link the message into a message chain.
;  761          1007  1 !       A user's AST routine may get called at this time.
;  762          1008  1 !       A new block and buffer get created and a read for another
;  763          1009  1 !       message is initiated.
;  764          1010  1 !
;  765          1011  1 ! CALLING SEQUENCE:
;  766          1012  1 !
;  767          1013  1 !       RET_STATUS.wlc.v = MAILBOX_AST_ROUTINE(P_PBCB.rl.v)
;  768          1014  1 !
;  769          1015  1 ! FORMAL PARAMETERS:
;  770          1016  1 !
;  771          1017  1 !       P_PBCB.rl.v             Address of pasteboard control block.
;  772          1018  1 !
;  773          1019  1 ! IMPLICIT INPUTS:
;  774          1020  1 !
;  775          1021  1 !       NONE
;  776          1022  1 !
;  777          1023  1 ! IMPLICIT OUTPUTS:
;  778          1024  1 !
;  779          1025  1 !       NONE
;  780          1026  1 !
;  781          1027  1 ! COMPLETION STATUS:
;  782          1028  1 !
;  783          1029  1 !       SS$_NORMAL      Normal successful completion
;  784          1030  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  785          1031  1 !       SS$_xyz         Any error possible from $DASSGN or $CANCEL
;  786          1032  1 !
;  787          1033  1 ! SIDE EFFECTS:
;  788          1034  1 !
;  789          1035  1 !       NONE
;  790          1036  1 !--
```

```
;  792          1037  2 BEGIN
;  793          1038  2
;  794          1039  2 LOCAL
;  795          1040  2
;  796          1041  2         SMGMBX          : REF $SMGMBX_DECL,
;  797          1042  2         STATUS;
;  798          1043  2
;  799          1044  2 BUILTIN
;  800          1045  2
;  801          1046  2         INSQUE;
;  802          1047  2
;  803          1048  2 BIND
;  804          1049  2
;  805          1050  2         PBCB    = .P_PBCB       : BLOCK[,BYTE];
;  806          1051  2
;  807          1052  2 EXTERNAL ROUTINE
;  808          1053  2
;  809          1054  2         LIB$GET_VM;
```

```
;   811     1055  2 !+
;   812     1056  2 ! A read has just completed (except during initialization).
;   813     1057  2 ! The buffer and I/O status block for this read are
;   814     1058  2 ! in the PBCB.
;   815     1059  2 ! We look at the kind of message it is and deal with it accordingly.
;   816     1060  2 !-
;   817     1061  2
;   818     1062  2 !+
;   819     1063  2 ! Create a new SMGMBX structure and link it in at the end.
;   820     1064  2 !-
;   821     1065  2
;   822     1066  2 IF .PBCB[PBCB_V_SMGMBX_INIT]
;   823     1067  2   THEN  PBCB[PBCB_V_SMGMBX_INIT]=0
;   824     1068  3   ELSE  BEGIN    ! Copy message to our queue
;   825     1069  3         BIND BFR = .PBCB[PBCB_A_SMGMBX_BUFFER] : VECTOR[2,WORD];
;   826     1070  3
;   827     1071  3         !+
;   828     1072  3         ! Do different things depending upon whether this is a
;   829     1073  3         ! broadcast message or an unsolicited input message.
;   830     1074  3         !-
;   831     1075  3
;   832     1076  3         IF .BFR[0] EQL MSG$_TRMUNSOLIC
;   833     1077  4           THEN  BEGIN   ! Unsolicited input
;   834     1078  4
;   835     1079  4                 !+
;   836     1080  4                 ! Call the user's AST routine, if one was specified.
;   837     1081  4                 !-
;   838     1082  4
;   839     1083  4                 IF .PBCB[PBCB_A_UNSOLICIT_RTN] NEQ 0
;   840     1084  5                   THEN  BEGIN
;   841     1085  5                         BIND ROUTINE
;   842     1086  5
;   843     1087  5                                BRTN    = .PBCB[PBCB_A_UNSOLICIT_RTN];
;   844     1088  5
;   845     1089  5                         ! Ignore status from user's AST routine
;   846     1090  5
;   847     1091  5                         BRTN(   .PBCB[PBCB_L_PBID],
;   848     1092  5                                 .PBCB[PBCB_L_UNSOLICIT_ARG]);
;   849     1093  5
;   850     1094  4                         END;
;   851     1095  4
;   852     1096  4           END     ! Unsolicited input
;   853     1097  4     ELSE  BEGIN   ! broadcast message
;   854     1098  4
;   855     1099  4           STATUS=LIB$GET_VM(%REF(SMGMBX_S_SIZE),SMGMBX);
;   856     1100  4           IF NOT .STATUS THEN SIGNAL(.STATUS);
;   857     1101  4
;   858     1102  4           CH$FILL(0,SMGMBX_S_SIZE,.SMGMBX);
;   859     1103  4           INSQUE(.SMGMBX,.PBCB[PBCB_L_SMGMBX_BLINK]);
;   860     1104  4
;   861     1105  4           !+
;   862     1106  4           ! Allocate a new buffer to receive mailbox messages.
;   863     1107  4           !-
;   864     1108  4
;   865     1109  4           STATUS=LIB$GET_VM(%REF(.PBCB[PBCB_W_SMGMBX_BUFSIZ]),SMGMBX[SMGMBX_A_BUFFER]);
;   866     1110  4           IF NOT .STATUS THEN SIGNAL(.STATUS);
;   867     1111  4
```

```
;  868          1112  4                    !+
;  869          1113  4                    ! Remember it's size (in case the default size in the PBCB changes.
;  870          1114  4                    !-
;  871          1115  4
;  872          1116  4                    SMGMBX[SMGMBX_W_BUFSIZ]=.PBCB[PBCB_W_SMGMBX_BUFSIZ];
;  873          1117  4
;  874          1118  4                    !+
;  875          1119  4                    ! Copy the returned message and it's I/O status block into
;  876          1120  4                    ! this structure.
;  877          1121  4                    !-
;  878          1122  4
;  879          1123  4                    CH$MOVE(8,PBCB[PBCB_Q_SMGMBX_IOSB],SMGMBX[SMGMBX_Q_IOSB]);
;  880          1124  4                    CH$MOVE(.SMGMBX[SMGMBX_W_MSG[EN],
;  881          1125  4                            .PBCB[PBCB_A_SMGMBX_BUFFER],
;  882          1126  4                            .SMGMBX[SMGMBX_A_BUFFER]);
;  883          1127  4
;  884          1128  4                    !+
;  885          1129  4                    ! Call the user's AST routine, if one was specified.
;  886          1130  4                    !-
;  887          1131  4
;  888          1132  4                    IF .PBCB[PBCB_A_BROADCAST_RTN] NEQ 0
;  889          1133  5                      THEN  BEGIN
;  890          1134  5                            BIND ROUTINE
;  891          1135  5
;  892          1136  5                                    BRTN    = .PBCB[PBCB_A_BROADCAST_RTN];
;  893          1137  5
;  894          1138  5                            ! Ignore status from user's AST routine
;  895          1139  5
;  896          1140  5                            BRTN(.PBCB[PBCB_L_BROADCAST_ARG]);
;  897          1141  5
;  898          1142  4                            END;
;  899          1143  4
;  900          1144  4                         END      ! Broadcast message
;  901          1145  4
;  902          1146  2                 END;     ! Copy message to our queue
;  903          1147  2
;  904          1148  2 !+
;  905          1149  2 ! Issue a read to the mailbox again so that our AST routine
;  906          1150  2 ! will wake up again the next time something comes in.
;  907          1151  2 !-
;  908          1152  2
;  909        P 1153  2 STATUS=$QIO(     CHAN     = .PBCB[PBCB_W_MBX_CHAN],
;  910        P 1154  2                  EFN      = .PBCB[PBCB_B_ASYNC_EFN],
;  911        P 1155  2                  FUNC     =  IO$_READVBLK,
;  912        P 1156  2                  IOSB     =  PBCB[PBCB_Q_SMGMBX_IOSB],
;  913        P 1157  2                  P1       = .PBCB[PBCB_A_SMGMBX_BUFFER],
;  914        P 1158  2                  P2       = .PBCB[PBCB_W_SMGMBX_BUFSIZ],
;  915        P 1159  2                  ASTADR   =  MAILBOX_AST_ROUTINE,
;  916          1160  2                  ASTPRM   =  PBCB);
;  917          1161  2 IF NOT .STATUS THEN SIGNAL(.STATUS);
;  918          1162  2
;  919          1163  2 RETURN  SS$_NORMAL
;  920          1164  2
;  921          1165  1 END;
```

```
                                                        .EXTRN  SYS$QIO

                                                07FC 00000 MAILBOX_AST_ROUTINE:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10        ; 1000
                                          5A 00000000G  00  9E 00002            MOVAB   LIB$GET_VM, R10
                                          59 00000000G  00  9E 00009            MOVAB   LIB$SIGNAL, R9
                                          5E            08  C2 00010            SUBL2   #8, SP
                                          56      04    AC  D0 00013            MOVL    P_PBCB, R6                    ; 1050
                                07        00D0  C6          02  E1 00017        BBC     #2, 208(R6), 1$               ; 1066
                                        00D0  C6            04  8A 0001D        BICB2   #4, 208(R6)                   ; 1067
                                                            18  11 00022        BRB     2$
                                          01    00D8  D6    B1 00024 1$:        CMPW    @216(R6), #1                  ; 1076
                                                            13  12 00029        BNEQ    3$
                                          50    00C0  C6    D0 0002B            MOVL    192(R6), R0                   ; 1083
                                                            72  13 00030        BEQL    6$
                                              00C4  C6      DD 00032            PUSHL   196(R6)                       ; 1092
                                                14  A6      DD 00036            PUSHL   20(R6)                        ; 1091
                                          60                02  FB 00039        CALLS   #2, (R0)
                                                            66  11 0003C 2$:    BRB     6$                            ; 1076
                                                04  AE      9F C003E 3$:        PUSHAB  SMGMBX                        ; 1099
                                      04  AE                18  D0 00041        MOVL    #24, 4(SP)
                                                04  AE      9F 00045            PUSHAB  4(SP)
                                          6A                02  FB 00048        CALLS   #2, LIB$GET_VM
                                          58                50  D0 0004B        MOVL    R0, STATUS
                                          05                58  E8 0004E        BLBS    STATUS, 4$                    ; 1100
                                                            58  DD 00051        PUSHL   STATUS
                                          69                01  FB 00053        CALLS   #1, LIB$SIGNAL
                  18        00            6E                00  2C 00056 4$:    MOVC5   #0, (SP), #0, #24, @SMGMBX    ; 1102
                                                04  BE              0005B
                                        00CC  D6      04  BE  0E  0005D        INSQUE  @SMGMBX, @204(R6)             ; 1103
                                          57          04  AE  D0  00063        MOVL    SMGMBX, R7                    ; 1109
                                                      08  A7  9F  00067        PUSHAB  8(R7)
                                      04  AE          00D6  C6  3C  0006A        MOVZWL  214(R6), 4(SP)
                                                        04  AE  9F  00070        PUSHAB  4(SP)
                                          6A                    02  FB 00073    CALLS   #2, LIB$GET_VM
                                          58                    50  D0 00076    MOVL    R0, STATUS
                                          05                    58  E8 00079    BLBS    STATUS, 5$                    ; 1110
                                                                58  DD 0007C    PUSHL   STATUS
                                          69                    01  FB 0007E    CALLS   #1, LIB$SIGNAL
                                              14  A7          00D6  C6  B0 00081 5$:  MOVW    214(R6), 20(R7)       ; 1116
                                0C    A7        00DC  C6              08  28 00087 MOVC3   #8, 220(R6), 12(R7)      ; 1123
                                08    B7        00D8  D6        0E    A7  28 0008E MOVC3   14(R7), @216(R6), @8(R7) ; 1126
                                          50          00B8  C6  D0  00096        MOVL    184(R6), R0                   ; 1132
                                                            07  13  0009B        BEQL    6$
                                              00BC  C6      DD 0009D            PUSHL   188(R6)                       ; 1140
                                          60                01  FB 000A1        CALLS   #1, (R0)
                                                            7E  7C 000A4 6$:    CLRQ    -(SP)                         ; 1160
                                                            7E  7C 000A6        CLRQ    -(SP)
                                          7E          00D6  C6  3C 000AB        MOVZWL  214(R6), -(SP)
                                                      00D8  C6  DD 000AD        PUSHL   216(R6)
                                                      56  DD 000B1              PUSHL   R6
                                                    FF49  CF  9F 000B3          PUSHAB  MAILBOX_AST_ROUTINE
                                                      00DC  C6  9F 000B7        PUSHAB  220(R6)
                                                            31  DD 000BB        PUSHL   #49
                                          7E          00D4  C6  3C 000BD        MOVZWL  212(R6), -(SP)
                                          7E            67  A6  9A 000C2        MOVZBL  103(R6), -(SP)
                                00000000G 00                  0C  FB 000C6    CALLS   #12, SY$$QIO
```

```
                                58      50  DO 000CD          MOVL    R0, STATUS                      ;
                                05      58  E8 000D0          BLBS    STATUS, 7$                      ; 1161
                                        58  DD 000D3          PUSHL   STATUS                          ;
                                69      01  FB 000D5          CALLS   #1, LIB$SIGNAL                  ;
                                50      01  D0 000D8 7$:      MOVL    #1, R0                          ; 1163
                                            04 000DB          RET                                     ; 1165

; Routine Size:  220 bytes,    Routine Base:  _SMG$CODE + 0318
```

```
;  923          1166  1 %SBTTL 'SMG$GET_BROADCAST_MESSAGE'
;  924          1167  1 GLOBAL ROUTINE SMG$GET_BROADCAST_MESSAGE (
;  925          1168  1     PASTEBOARD_ID,P_MSG_DESC,P_MSG_LEN,P_SENDER_PID) =
;  926          1169  1
;  927          1170  1 !++
;  928          1171  1 ! FUNCTIONAL DESCRIPTION:
;  929          1172  1 !
;  930          1173  1 !       This routine asks if any broadcast messages have been
;  931          1174  1 !       received.  If none have been received, it returns with
;  932          1175  1 !       the sucess status SMG$_NO_MORMSG.  Normally, when a program
;  933          1176  1 !       desires to process trapped messages, it would call this routine
;  934          1177  1 !       repeatedly, handling each message it provided until
;  935          1178  1 !       there were no more.
;  936          1179  1 !
;  937          1180  1 !       This routine may be called from either AST level or NON-AST level.
;  938          1181  1 !
;  939          1182  1 ! CALLING SEQUENCE:
;  940          1183  1 !
;  941          1184  1 !       RET_STATUS.wlc.v = SMG$GET_BROADCAST_MESSAGE (
;  942          1185  1 !                                  PASTEBOARD_ID.rl.r
;  943          1186  1 !                                  [, P_MSG_DESC.wt.dx,
;  944          1187  1 !                                  [, [P_MSG_LEN.ww.r] ]
;  945          1188  1 !                                  [, [P_SENDER_PID])
;  946          1189  1 !
;  947          1190  1 ! FORMAL PARAMETERS:
;  948          1191  1 !
;  949          1192  1 !       PASTEBOARD_ID           address of pastebaord associated
;  950          1193  1 !                               with terminal in question
;  951          1194  1 !
;  952          1195  1 !       P_MSG_DESC              address of a descriptor for a
;  953          1196  1 !                               buffer to receive the broadcast
;  954          1197  1 !                               message.  If omitted, the broadcast
;  955          1198  1 !                               message is discarded.
;  956          1199  1 !
;  957          1200  1 !       P_MSG_LEN               address of a word to receive the
;  958          1201  1 !                               actual length of the received message.
;  959          1202  1 !                               This is set to 0 if no messages were
;  960          1203  1 !                               available.
;  961          1204  1 !                               The value of this word is unpredictable
;  962          1205  1 !                               if other errors occur.
;  963          1206  1 !
;  964          1207  1 !       P_SENDER_PID            address of a longword to receive the PID
;  965          1208  1 !                               (process id) of the process that sent
;  966          1209  1 !                               this broadcast message.  This may
;  967          1210  1 !                               be 0 for certain system processes.
;  968          1211  1 !                               This longword is not set if an
;  969          1212  1 !                               error occurs.
;  970          1213  1 !
;  971          1214  1 ! IMPLICIT INPUTS:
;  972          1215  1 !
;  973          1216  1 !       internal data storage holding previously received broadcast
;  974          1217  1 !       messages.
;  975          1218  1 !
;  976          1219  1 ! IMPLICIT OUTPUTS:
;  977          1220  1 !
;  978          1221  1 !       NONE
;  979          1222  1 !
```

```
;  980          1223  1 ! COMPLETION STATUS:
;  981          1224  1 !
;  982          1225  1 !       SS$_NORMAL      Normal successful completion
;  983          1226  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  984          1227  1 !       SMG$_NO_MORMSG  No more broadcast messages are available
;  985          1228  1 !       LIB$_xyz        Any error possible from LIB$SCOPY_DX
;  986          1229  1 !       SS$_xyz         Errors from mailbox driver (these errors
;  987          1230  1 !                       may have occurred when we tried to read
;  988          1231  1 !                       a message).
;  989          1232  1 !
;  990          1233  1 ! SIDE EFFECTS:
;  991          1234  1 !
;  992          1235  1 !       NONE
;  993          1236  1 !
;  994          1237  1 !--
```

```
;  996          1238  2 BEGIN
;  997          1239  2
;  998          1240  2 BIND
;  999          1241  2
; 1000          1242  2         MSG_DESC        = .P_MSG_DESC   : BLOCK[,BYTE],
; 1001          1243  2         MSG_LEN         = .P_MSG_LEN    : WORD,
; 1002          1244  2         SENDER_PID      = .P_SENDER_PID : LONG;
; 1003          1245  2
; 1004          1246  2 EXTERNAL LITERAL
; 1005          1247  2
; 1006          1248  2         SMG$_NO_MORMSG;
; 1007          1249  2
; 1008          1250  2 EXTERNAL ROUTINE
; 1009          1251  2
; 1010          1252  2         LIB$SCOPY_R_DX,
; 1011          1253  2         LIB$GET_VM,
; 1012          1254  2         LIB$FREE_VM;
; 1013          1255  2
; 1014          1256  2 LOCAL
; 1015          1257  2
; 1016          1258  2         FOUND_FLAG,
; 1017          1259  2         FINAL_STATUS,
; 1018          1260  2         SMGMBX          : REF $SMGMBX_DECL,
; 1019          1261  2         PBCB            : REF BLOCK[,BYTE],
; 1020          1262  2 !       AST_STATUS,
; 1021          1263  2         STATUS;
; 1022          1264  2
; 1023          1265  2 BUILTIN
; 1024          1266  2
; 1025          1267  2         REMQUE,
; 1026          1268  2         NULLPARAMETER;
```

```
;1028        1269  2 $SMG$VALIDATE_ARGCOUNT(1,4);
;1029        1270  2
;1030        1271  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;1031        1272  2
;1032        1273  2 !+
;1033        1274  2 ! Note that we have not yet found a broadcast message.
;1034        1275  2 !-
;1035        1276  2
;1036        1277  2 FOUND_FLAG=0;
;1037        1278  2
;1038        1279  2 !+
;1039        1280  2 ! Go look in our PBCB to see if any mailbox messages are present.
;1040        1281  2 ! If so, extract them off, one at a time, looking for a
;1041        1282  2 ! broadcast message.  Return it to the user if he supplied a buffer.
;1042        1283  2 !-
;1043        1284  2
;1044        1285  2 !+
;1045        1286  2 ! Note that the SMGMBX message queue must always contain at least
;1046        1287  2 ! one block.  This is the block that the current read is attached to.
;1047        1288  2 ! We must never touch this block.  Note also that we need not
;1048        1289  2 ! disable ASTs at critical times to prevent our queue from
;1049        1290  2 ! getting messed up because INSQUE and REMQUE instructions
;1050        1291  2 ! are non-interruptable.
;1051        1292  2 !-
;1052        1293  2
;1053        1294  2 WHILE 1 DO
;1054        1295  3         BEGIN   ! Look for message in queue
;1055        1296  3
;1056        1297  3 !       !+
;1057        1298  3 !       ! Disable delivery of ASTs.
;1058        1299  3 !       !-
;1059        1300  3 !
;1060        1301  3 !       AST_STATUS=$SETAST(ENBFLG=0);
;1061        1302  3
;1062        1303  3 !       IF .PBCB[PBCB_L_SMGMBX_FLINK] EQL .PBCB[PBCB_L_SMGMBX_BLINK]
;1063        1304  3 !         THEN  EXITLOOP;
;1064        1305  3
;1065        1306  3         !+
;1066        1307  3         ! Attempt to remove the first mailbox block from the queue.
;1067        1308  3         !-
;1068        1309  3
;1069        1310  3         STATUS=REMQUE(.PBCB[PBCB_L_SMGMBX_FLINK],SMGMBX);
;1070        1311  3
;1071        1312  3         !+
;1072        1313  3         ! If the queue was empty, then we merely exit the loop.
;1073        1314  3         ! This means that there are no broadcast messages available.
;1074        1315  3         !-
;1075        1316  3
;1076        1317  3         IF .STATUS THEN EXITLOOP;
;1077        1318  3
;1078        1319  3 !       !+
;1079        1320  3 !       ! Re-enable delivery of ASTs if they had previously been enabled.
;1080        1321  3 !       !-
;1081        1322  3 !
;1082        1323  3 !       IF .AST_STATUS EQL SS$_WASSET
;1083        1324  3 !         THEN  $SETAST(ENBFLG=1);
;1084        1325  3
```

```
;  1085          1326  3          !+
;  1086          1327  3          ! Examine the removed block to see if it is a broadcast mailbox
;  1087          1328  3          ! message.
;  1088          1329  3          !-
;  1089          1330  3
;  1090          1331  3          !+
;  1091          1332  3          ! If the I/O status block shows that there was some error
;  1092          1333  3          ! reading this message, then return that error now.
;  1093          1334  3          ! Who knows? It may have been a broadcast message and we
;  1094          1335  3          ! somehow lost it.
;  1095          1336  3          !-
;  1096          1337  3
;  1097          1338  3          IF NOT .SMGMBX[SMGMBX_W_STATUS]
;  1098          1339  4            THEN  BEGIN   ! I/O error reading message
;  1099          1340  4                  IF NOT NULLPARAMETER(3)
;  1100          1341  4                    THEN  MSG_LEN=0;
;  1101          1342  4
;  1102          1343  4                  FOUND_FLAG=1;
;  1103          1344  4
;  1104          1345  4                  FINAL_STATUS=.SMGMBX[SMGMBX_W_STATUS]
;  1105          1346  4                  END     ! I/O error reading message
;  1106          1347  4            ELSE  BEGIN   ! Valid message
;  1107          1348  4
;  1108          1349  4                  BIND
;  1109          1350  4
;  1110          1351  4                          TYPE    = .SMGMBX[SMGMBX_A_BUFFER]      : WORD,
;  1111          1352  4                          BRDMSG  = TYPE                  : VECTOR[,BYTE];
;  1112          1353  4
;  1113          1354  4                  !+
;  1114          1355  4                  ! Look at the type of the message.
;  1115          1356  4                  ! This type is in the first word of the message.
;  1116          1357  4                  ! If it is not a broadcast message, then we throw it away.
;  1117          1358  4                  !-
;  1118          1359  4
;  1119          1360  4                  IF .TYPE EQL MSG$_TRMBRDCST
;  1120          1361  5                    THEN  BEGIN   ! Handle broadcast message
;  1121          1362  5
;  1122          1363  5                          !+
;  1123          1364  5                          ! I can't find any symbolic offsets for this message.
;  1124          1365  5                          ! It is described in the I/O user's guide
;  1125          1366  5                          ! under the section about terminals.
;  1126          1367  5                          ! The broadcast message starts at offset 22
;  1127          1368  5                          ! in this block.
;  1128          1369  5                          ! Its length is stored in a word at offset 20.
;  1129          1370  5                          !-
;  1130          1371  5
;  1131          1372  5                          BIND
;  1132          1373  5
;  1133          1374  5                                  BRDLEN  = BRDMSG[20]    : WORD,
;  1134          1375  5                                  MESSAGE = BRDMSG[22]    : VECTOR[,BYTE];
;  1135          1376  5
;  1136          1377  5                          !+
;  1137          1378  5                          ! Note that we have found a broadcast message.
;  1138          1379  5                          !-
;  1139          1380  5
;  1140          1381  5                          FOUND_FLAG=1;
;  1141          1382  5                          FINAL_STATUS=.SMGMBX[SMGMBX_W_STATUS];
```

```
;  1142     1383  5
;  1143     1384  5                        !+
;  1144     1385  5                        ! Return to the user the length of
;  1145     1386  5                        ! the message (if he asked for it).
;  1146     1387  5                        !-
;  1147     1388  5
;  1148     1389  5                        IF NOT NULLPARAMETER(3)
;  1149     1390  5                          THEN  MSG_LEN=.BRDLEN;
;  1150     1391  5
;  1151     1392  5                        !+
;  1152     1393  5                        ! Return to the user, the PID of the process
;  1153     1394  5                        ! that broadcast this message,
;  1154     1395  5                        ! if he asked for it.
;  1155     1396  5                        !-
;  1156     1397  5
;  1157     1398  5                        IF NOT NULLPARAMETER(4)
;  1158     1399  5                          THEN  SENDER_PID=.SMGMBX[SMGMBX_L_SENDER];
;  1159     1400  5
;  1160     1401  5                        !+
;  1161     1402  5                        ! Return the message to the user
;  1162     1403  5                        ! (using the semantics of his descriptor)
;  1163     1404  5                        ! if he asked for it.
;  1164     1405  5                        !-
;  1165     1406  5
;  1166     1407  5                        IF NOT NULLPARAMETER(2)
;  1167     1408  6                          THEN  BEGIN   ! Return message
;  1168     1409  6
;  1169     1410  6                                !+
;  1170     1411  6                                ! Copy it to the user's buffer.
;  1171     1412  6                                !-
;  1172     1413  6
;  1173     1414  6                                STATUS=LIB$SCOPY_R_DX(
;  1174     1415  6                                        BRDLEN,
;  1175     1416  6                                        MESSAGE,
;  1176     1417  6                                        MSG_DESC);
;  1177     1418  6                                IF NOT .STATUS THEN RETURN .STATUS
;  1178     1419  6
;  1179     1420  5                                END;    ! Return message
;  1180     1421  5
;  1181     1422  4                        END;    ! Handle broadcast message
;  1182     1423  4                                                                    ;
;  1183     1424  3                END;    ! Valid message
;  1184     1425  3
;  1185     1426  3                !+
;  1186     1427  3                ! Return the buffer to virtual memory.
;  1187     1428  3                !-
;  1188     1429  3
;  1189     1430  3                STATUS=LIB$FREE_VM(%REF(.SMGMBX[SMGMBX_W_BUFSIZ]),
;  1190     1431  3                                   SMGMBX[SMGMBX_A_BUFFER]);
;  1191     1432  3                IF NOT .STATUS THEN RETURN .STATUS;
;  1192     1433  3
;  1193     1434  3                !+
;  1194     1435  3                ! Return the mailbox block to virtual memory.
;  1195     1436  3                !-
;  1196     1437  3
;  1197     1438  3                STATUS=LIB$FREE_VM(%REF(SMGMBX_S_SIZE),SMGMBX);
;  1198     1439  3                IF NOT .STATUS THEN RETURN .STATUS;
```

```
;1199     1440  3
;1200     1441  3          !+
;1201     1442  3          ! If we had found a message, then we can return now.
;1202     1443  3          !-
;1203     1444  3
;1204     1445  3          IF .FOUND_FLAG
;1205     1446  3            THEN  RETURN  .FINAL_STATUS
;1206     1447  3
;1207     1448  2          END;     ! Look for message in queue
;1208     1449  2
;1209     1450  2  !+
;1210     1451  2  ! No messages are left in the queue.
;1211     1452  2  ! Return a status to the user telling him that no
;1212     1453  2  ! broadcast messages are available.
;1213     1454  2  !-
;1214     1455  2
;1215     1456  2  IF NOT NULLPARAMETER(3)
;1216     1457  2    THEN  MSG_LEN=0;
;1217     1458  2
;1218     1459  2  RETURN  SMG$_NO_MORMSG
;1219     1460  2
;1220     1461  1  END;


                                                                .EXTRN  SMG$_NO_MORMSG, LIB$SCOPY_R_DX
                                                                .EXTRN  LIB$FREE_VM

                                                07FC 00000      .ENTRY  SMG$GET_BROADCAST_MESSAGE, Save R2,R3,R4,-  ;1167
                                                                        R5,R6,R7,R8,R9,R10
                                          5A 00000000G  00  9E  00002       MOVAB   LIB$FREE_VM, R10
                                          5E            08  C2  00009       SUBL2   #8, SP
                                          59      08    AC  D0  0000C       MOVL    P_MSG_DESC, R9                  ;1242
                                          57      0C    AC  7D  00010       MOVQ    P_MSG_LEN, R7                   ;1243
                                50        6C            01  83  00014       SUBB3   #1, (AP), DIFF                  ;1269
                                          03            50  91  00018       CMPB    DIFF, #3
                                                        08  1B  0001B       BLEQU   1$
                                          50 00000000G  8F  DD  0001D       MOVL    #SMG$_WRONUMARG, R0
                                                        04  00024           RET
                                          50      04    BC  D0  00025 1$:   MOVL    @PASTEBOARD_ID, R0              ;1271
                                                        11  19  00029       BLSS    2$
                             00000000G    00            50  D1  0002B       CMPL    R0, PBD_L_COUNT
                                                        08  14  00032       BGTR    2$
                          08 00000000G    00            50  E0  00034       BBS     R0, PBD_V_PB_AVAIL, 3$
                                          50 00000000G  8F  DD  0003C 2$:   MOVL    #SMG$_INVPAS_ID, R0
                                                        04  00043           RET
                                          53 00000000G0040  D0  00044 3$:   MOVL    PBD_A_PBCB[R0], PBCB
                                                        54  D4  0004C       CLRL    FOUND_FLAG                      ;1277
                                   04   AE      00C8    D3  0F  0004E 4$:   REMQUE  @200(PBCB), SMGMBX              ;1310
                                                        51  DC  00054       MOVPSL  R1
                51                51            02      01  EF  00056       EXTZV   #1, #2, R1, R1
                                                50      51  D0  0005B       MOVL    R1, STATUS
                                                03      50  E9  0005E       BLBC    STATUS, 5$                      ;1317
                                                      0094  31  00061       BRW     12$
                                                52      04  AE  D0  00064 5$:   MOVL    SMGMBX, R2                  ;1338
                                                15      0C  A2  E8  00068       BLBS    12(R2), 7$
                                                03          6C  91  0006C       CMPB    (AP), #3                    ;1340
```

```
                                   07  1F 0006F         BLSSU   6$                              ;
                          0C       AC  D5 00071         TSTL    12(AP)                          ;
                                   02  13 00074         BEQL    6$                              ;
                                   67  B4 00076         CLRW    (R7)                            ; 1341
                      54           01  D0 00078  6$:    MOVL    #1, FOUND_FLAG                  ; 1343
                      56  0C       A2  3C 0007B         MOVZWL  12(R2), FINAL_STATUS            ; 1345
                                   4C  11 0007F         BRB     10$                             ;
                0053  8F  08       B2  B1 00081  7$:    CMPW    @8(R2), #83                     ; 1360
                                   44  12 00087         BNEQ    10$                             ;
          51      08  A2           14  C1 00089         ADDL3   #20, 8(R2), R1                  ; 1374
          55      08  A2           16  C1 0008E         ADDL3   #22, 8(R2), R5                  ; 1375
                      54           01  D0 00093         MOVL    #1, FOUND_FLAG                  ; 1381
                      56  0C       A2  3C 00096         MOVZWL  12(R2), FINAL_STATUS            ; 1382
                      03           6C  91 0009A         CMPB    (AP), #3                        ; 1389
                                   08  1F 0009D         BLSSU   8$                              ;
                          0C       AC  D5 0009F         TSTL    12(AP)                          ;
                                   03  13 000A2         BEQL    8$                              ;
                      67           61  B0 000A4         MOVW    (R1), (R7)                      ; 1390
                      04           6C  91 000A7  8$:    CMPB    (AP), #4                        ; 1398
                                   09  1F 000AA         BLSSU   9$                              ;
                          10       AC  D5 000AC         TSTL    16(AP)                          ;
                                   04  13 000AF         BEQL    9$                              ;
                      68  10       A2  D0 000B1         MOVL    16(R2), (R8)                    ; 1399
                      02           6C  91 000B5  9$:    CMPB    (AP), #2                        ; 1407
                                   13  1F 000B8         BLSSU   10$                             ;
                          08       AC  D5 000BA         TSTL    8(AP)                           ;
                                   0E  13 000BD         BEQL    10$                             ;
                        0222       8F  BB 000BF         PUSHR   #^M<R1,R5,R9>                   ; 1414
           00000000G  00           03  FB 000C3         CALLS   #3, LIB$SCOPY_R_DX              ;
                      3E           50  E9 000CA         BLBC    STATUS, 14$                     ; 1418
                          08       A2  9F 000CD  10$:   PUSHAB  8(R2)                           ; 1431
                  04  AE  14       A2  3C 000D0         MOVZWL  20(R2), 4(SP)                   ; 1430
                          04       AE  9F 000D5         PUSHAB  4(SP)                           ;
                      6A           02  FB 000D8         CALLS   #2, LIB$FREE_VM                 ; 1431
                      2D           50  E9 000DB         BLBC    STATUS, 14$                     ; 1432
                          04       AE  9F 000DE         PUSHAB  SMGMBX                          ; 1438
                  04  AE           18  D0 000E1         MOVL    #24, 4(SP)                      ;
                          04       AE  9F 000E5         PUSHAB  4(SP)                           ;
                      6A           02  FB 000E8         CALLS   #2, LIB$FREE_VM                 ;
                      1D           50  E9 000EB         BLBC    STATUS, 14$                     ; 1439
                      03           54  E8 000EE         BLBS    FOUND_FLAG, 11$                 ; 1445
                                 FF5A  31 000F1         BRW     4$                              ;
                      50           56  D0 000F4  11$:   MOVL    FINAL_STATUS, R0                ; 1446
                                       04 000F7         RET                                     ;
                      03           6C  91 000F8  12$:   CMPB    (AP), #3                        ; 1456
                                   07  1F 000FB         BLSSU   13$                             ;
                          0C       AC  D5 000FD         TSTL    12(AP)                          ;
                                   02  13 00100         BEQL    13$                             ;
                                   67  B4 00102         CLRW    (R7)                            ; 1457
                      50 00000000G 8F  D0 00104  13$:   MOVL    #SMG$_NO_MORMSG, R0             ; 1459
                                       04 0010B  14$:   RET                                     ; 1461

; Routine Size:  268 bytes,    Routine Base:  _SMG$CODE + 03F4
```

```
;  1222          1462  1 %SBTTL 'SMG$$OUT_OF_BAND_HANDLER - Generic out-of-band handler'
;  1223          1463  1 GLOBAL ROUTINE SMG$$OUT_OF_BAND_HANDLER (
;  1224          1464  1         ARG, R0, R1, PC, PSL ) =
;  1225          1465  1
;  1226          1466  1 !++
;  1227          1467  1 ! FUNCTIONAL DESCRIPTION:
;  1228          1468  1 !
;  1229          1469  1 !       This routine processes an out-of-band AST.
;  1230          1470  1 !       It runs at AST level.  It logically gets called when the
;  1231          1471  1 !       user types an out-of-band character.
;  1232          1472  1 !       Actually, another routine actually gets called first
;  1233          1473  1 !       and then calls this one with the same argument list.
;  1234          1474  1 !
;  1235          1475  1 ! CALLING SEQUENCE:
;  1236          1476  1 !
;  1237          1477  1 !       RET_STATUS.wlc.v = SMG$$OUT_OF_BAND_HANDLER (
;  1238          1478  1 !                                   ARG.rz.v,
;  1239          1479  1 !                                   R0.rl.v
;  1240          1480  1 !                                   R1.rl.v,
;  1241          1481  1 !                                   PC.ra.v,
;  1242          1482  1 !                                   PSL.rl.v)
;  1243          1483  1 !
;  1244          1484  1 ! FORMAL PARAMETERS:
;  1245          1485  1 !
;  1246          1486  1 !       ARG             character that was typed
;  1247          1487  1 !       R0              register 0 at time of AST
;  1248          1488  1 !       R1              register 1 at time of AST
;  1249          1489  1 !       PC              PC when AST occurred
;  1250          1490  1 !       PSL             contents of PSL at time of AST
;  1251          1491  1 !
;  1252          1492  1 ! IMPLICIT INPUTS:
;  1253          1493  1 !
;  1254          1494  1 !       Return PC on the stack.
;  1255          1495  1 !       We use this to determine who called us.
;  1256          1496  1 !       The address of the calling procedure is used to
;  1257          1497  1 !       enable us to locate the PBCB associated with the
;  1258          1498  1 !       terminal that the out-of-band character was typed on.
;  1259          1499  1 !
;  1260          1500  1 ! IMPLICIT OUTPUTS:
;  1261          1501  1 !
;  1262          1502  1 !       NONE
;  1263          1503  1 !
;  1264          1504  1 ! COMPLETION STATUS:
;  1265          1505  1 !
;  1266          1506  1 !       SS$_NORMAL      Normal successful completion
;  1267          1507  1 !
;  1268          1508  1 ! SIDE EFFECTS:
;  1269          1509  1 !
;  1270          1510  1 !--
```

```
;  1272          1511  2 BEGIN
;  1273          1512  2
;  1274          1513  2 LOCAL
;  1275          1514  2
;  1276          1515  2         NEXT_PC,                          ! return PC address
;  1277          1516  2                                           ! should point into PBCB
;  1278          1517  2         PBCB : REF BLOCK[,BYTE],          ! Gets the address of our PBCB
;  1279          1518  2         PBID,                             ! Gets our pasteboard id.
;  1280          1519  2         BAND_INFO_BLOCK : BLOCK[SMG$S_BAND_INFORMATION_TABLE,BYTE],
;  1281          1520  2                                           ! Band information table
;  1282          1521  2         STATUS;                           ! Local status
;  1283          1522  2
;  1284          1523  2 EXTERNAL
;  1285          1524  2
;  1286          1525  2         PBD_A_PBCB      : VOLATILE VECTOR;      ! pasteboard directory
;  1287          1526  2
;  1288          1527  2 BUILTIN
;  1289          1528  2
;  1290          1529  2         FP;                       ! frame pointer
;  1291          1530  2
;  1292          1531  2 EXTERNAL LITERAL
;  1293          1532  2
;  1294          1533  2         SMG$_FATERRLIB;
;  1295          1534  2
;  1296          1535  2 BIND    FRAME   = .FP   : BLOCK[,BYTE];         ! Stack frame
;  1297          1536  2
;  1298          1537  2 !+
;  1299          1538  2 ! Macro $GET_FIELD_OFFSET takes a field reference and replaces
;  1300          1539  2 ! it in the text stream by the offset represented by that reference.
;  1301          1540  2 ! (the first of the 4 items in the block structure reference).
;  1302          1541  2 ! This macro works whether the actual argument is a true
;  1303          1542  2 ! BLISS field name or if it is an SDL macro name.
;  1304          1543  2 ! The way it tells the two apart is this: A field name will
;  1305          1544  2 ! be a single lexeme, while a macro as an actual argument will
;  1306          1545  2 ! be expanded before invocation and will appear as 4 arguments.
;  1307          1546  2 !-
;  1308          1547  2
;  1309          1548  2 MACRO
;  1310          1549  2
;  1311        M 1550  2     $GET_FIELD_OFFSET(FIELD_REFERENCE) =
;  1312        M 1551  2
;  1313        M 1552  2         %IF %LENGTH EQL 1
;  1314        M 1553  2
;  1315        M 1554  2           %THEN %FIELDEXPAND(FIELD_REFERENCE,1)
;  1316        M 1555  2
;  1317        M 1556  2           %ELSE FIELD_REFERENCE
;  1318        M 1557  2
;  1319        M 1558  2         %FI
;  1320        M 1559  2
;  1321          1560  2     %;
;  1322          1561  2
;  1323          1562  2 !+
;  1324          1563  2 ! Find out where this routine is going to return to.
;  1325          1564  2 ! That should be the RET at offset PBCB_B_RET in the AST
;  1326          1565  2 ! routine that we built in our PBCB.
;  1327          1566  2 !-
;  1328          1567  2
```

```
;  1329          1568  2 NEXT_PC=.FRAME[SF$L_SAVE_PC];
;  1330          1569  2
;  1331          1570  2 !+
;  1332          1571  2 ! By subtracting the offset of "PBCB_B_RET" we get the
;  1333          1572  2 ! address of the start of the PBCB for the terminal that
;  1334          1573  2 ! the out-of-band character was typed on.  We have to go through
;  1335          1574  2 ! these machinations because there is no other was to tell
;  1336          1575  2 ! which terminal caused the AST to trigger.
;  1337          1576  2 !-
;  1338          1577  2
;  1339          1578  2 PBCB=.NEXT_PC - $GET_FIELD_OFFSET(PBCB_B_RET);
;  1340          1579  2
;  1341          1580  2 !+
;  1342          1581  2 ! Perform some validity checks to make sure that this is
;  1343          1582  2 ! really the address of a PBCB.  If not, we just return
;  1344          1583  2 ! without ever calling the user's AST routine.
;  1345          1584  2 !-
;  1346          1585  2
;  1347          1586  2 IF .PBCB[PBCB_B_RET] NEQ %X'04' THEN    RETURN  SMG$_FATERRLIB;
;  1348          1587  2 IF .PBCB[PBCB_B_ABS] NEQ %X'9F' THEN    RETURN  SMG$_FATERRLIB;
;  1349          1588  2
;  1350          1589  2 PBID=.PBCB[PBCB_L_PBID];
;  1351          1590  2 IF .PBID GTRU PBD_K_MAX_PB      THEN    RETURN  SMG$_FATERRLIB;
;  1352          1591  2 IF .PBD_A_PBCB[.PBID] NEQ .PBCB THEN    RETURN  SMG$_FATERRLIB;
;  1353          1592  2
;  1354          1593  2 !+
;  1355          1594  2 ! Call the user's AST routine as if the system were calling it
;  1356          1595  2 ! directly.  The difference is, though, that the first
;  1357          1596  2 ! argument will be the address of an SMG BAND INFORMATION BLOCK,
;  1358          1597  2 ! instead of just the character that was typed.
;  1359          1598  2 ! If somehow no user routine was specified, we just return also.
;  1360          1599  2 !-
;  1361          1600  2
;  1362          1601  2 IF .PBCB[PBCB_A_BAND_ROUTINE] EQL 0
;  1363          1602  2   THEN  RETURN  SMG$_FATERRLIB
;  1364          1603  3   ELSE  BEGIN
;  1365          1604  3         BIND ROUTINE
;  1366          1605  3
;  1367          1606  3                 USER_RTN = .PBCB[PBCB_A_BAND_ROUTINE];
;  1368          1607  3
;  1369          1608  3         !+
;  1370          1609  3         ! Set up the band information block.
;  1371          1610  3         !-
;  1372          1611  3
;  1373          1612  3         BAND_INFO_BLOCK[SMG$L_PASTEBOARD_ID] = .PBID;
;  1374          1613  3         BAND_INFO_BLOCK[SMG$L_ARG]           = .PBCB[PBCB_L_BAND_AST_ARG];
;  1375          1614  3         BAND_INFO_BLOCK[SMG$L_CHARACTER]     = '    ';
;  1376          1615  3         BAND_INFO_BLOCK[SMG$B_CHARACTER]     = .ARG;
;  1377          1616  3
;  1378          1617  3         STATUS=USER_RTN( BAND_INFO_BLOCK,.R0,.R1,.PC,.PSL );
;  1379          1618  3         IF NOT .STATUS THEN RETURN .STATUS
;  1380          1619  2         END;
;  1381          1620  2
;  1382          1621  2 RETURN  SS$_NORMAL
;  1383          1622  2
;  1384          1623  1 END;
```

```
                                                        .EXTRN  SMG$_FATERRLIB

                                    0004 00000          .ENTRY  SMG$$OUT_OF_BAND_HANDLER, Save R2                ; 1463
                          5E            0C   C2 00002  SUBL2   #12, SP                                          ;
                          50        1C  AE   D0 00005  MOVL    28(FP), NEXT_PC                                  ; 1568
                          50      FF6B  C0   9E 00009  MOVAB   -149(R0), PBCB                                    ; 1578
                          04      0095  C0   91 0000E  CMPB    149(PBCB), #4                                    ; 1586
                                        22   12 00013  BNEQ    1$                                               ;
                      9F  8F      0090  C0   91 00015  CMPB    144(PBCB), #159                                  ; 1587
                                        1A   12 0001B  BNEQ    1$                                               ;
                          51        14  A0   D0 0001D  MOVL    20(PBCB), PBID                                   ; 1589
                          10            51   D1 00021  CMPL    PBID, #16                                        ; 1590
                                        11   1A 00024  BGTRU   1$                                               ;
                          50 00000000G  0041 D1 00026  CMPL    PBD_A_PBCB[PBID], PBCB                           ; 1591
                                        07   12 0002E  BNEQ    1$                                               ;
                          52      0098  C0   D0 00030  MOVL    152(PBCB), R2                                    ; 1601
                                        08   12 00035  BNEQ    2$                                               ;
                          50 00000000G  8F   D0 00037  1$:      MOVL    #SMG$_FATERRLIB, R0                      ; 1602
                                        04   0003E  RET                                                         ;
                          6E            51   D0 0003F  2$:      MOVL    PBID, BAND_INFO_BLOCK                    ; 1612
                      04  AE      009C  C0   D0 00042  MOVL    156(PBCB), BAND_INFO_BLOCK+4                     ; 1613
                      08  AE  20202020  8F   D0 00048  MOVL    #538976288, BAND_INFO_BLOCK+8                    ; 1614
                      08  AE        04  AC   90 00050  MOVB    ARG, BAND_INFO_BLOCK+8                           ; 1615
                          7E        10  AC   7D 00055  MOVQ    PC, -(SP)                                        ; 1617
                          7E        08  AC   7D 00059  MOVQ    R0, -(SP)                                        ;
                                    10  AE   9F 0005D  PUSHAB  BAND_INFO_BLOCK                                  ;
                          62            05   FB 00060  CALLS   #5, (R2)                                        ;
                          03            50   E9 00063  BLBC    STATUS, 3$                                       ; 1618
                          50            01   D0 00066  MOVL    #1, R0                                           ; 1621
                                        04   00069  3$:      RET                                                ; 1623

; Routine Size:  106 bytes,    Routine Base:  _SMG$CODE + 0500
```

```
;1386     1624  1 %SBTTL 'SMG$SET_OUT_OF_BAND_ASTS'
;1387     1625  1 GLOBAL ROUTINE SMG$SET_OUT_OF_BAND_ASTS (
;1388     1626  1           PASTEBOARD_ID,P_CHAR_MASK,AST_RTN,AST_ARG ) =
;1389     1627  1
;1390     1628  1 !++
;1391     1629  1 ! FUNCTIONAL DESCRIPTION:
;1392     1630  1 !
;1393     1631  1 !       This routine enables new out-of-band characters to
;1394     1632  1 !       be accepted from the specified terminal.  If one of these
;1395     1633  1 !       characters is typed, the specified AST routine gets called.
;1396     1634  1 !       See the RTL manual for conventions on how AST routines get called.
;1397     1635  1 !       The first argument to the AST routine is the address of a
;1398     1636  1 !       band information table.  This table contains things like
;1399     1637  1 !       the pasteboard id and the character that was typed.
;1400     1638  1 !       The format and structure of this table is described in SMGDEF.SDL.
;1401     1639  1 !
;1402     1640  1 ! CALLING SEQUENCE:
;1403     1641  1 !
;1404     1642  1 !       RET_STATUS.wlc.v = SMG$ENABLE_OUT_OF_BAND_ASTS (
;1405     1643  1 !                                  PASTEBOARD_ID.rl.r,
;1406     1644  1 !                                  P_CHAR_MASK.rl.r,
;1407     1645  1 !                                  AST_RTN.szem.r
;1408     1646  1 !                                  [, AST_ARG.rz.v])
;1409     1647  1 !
;1410     1648  1 ! FORMAL PARAMETERS:
;1411     1649  1 !
;1412     1650  1 !       PASTEBOARD_ID   Pasteboard ID for terminal to be affected.
;1413     1651  1 !
;1414     1652  1 !       P_CHAR_MASK     Longword mask of which control characters
;1415     1653  1 !                       to be the new out-of-band control characters.
;1416     1654  1 !
;1417     1655  1 !       AST_RTN         Address of AST routine to be called
;1418     1656  1 !
;1419     1657  1 !       AST_ARG         Optional value to be passed along to the AST routine.
;1420     1658  1 !                       It will appear as the second longword in the
;1421     1659  1 !                       band information table.
;1422     1660  1 !
;1423     1661  1 ! IMPLICIT INPUTS:
;1424     1662  1 !
;1425     1663  1 !       NONE
;1426     1664  1 !
;1427     1665  1 ! IMPLICIT OUTPUTS:
;1428     1666  1 !
;1429     1667  1 !       Structure holding information about out-of-band AST routines
;1430     1668  1 !       gets updated.
;1431     1669  1 !
;1432     1670  1 ! COMPLETION STATUS:
;1433     1671  1 !
;1434     1672  1 !       SS$_NORMAL      Normal successful completion
;1435     1673  1 !       SMG$_WRONUMARG  Wrong number of arguments
;1436     1674  1 !       SS$_xyz         Any error possible from $QIOW
;1437     1675  1 !
;1438     1676  1 ! SIDE EFFECTS:
;1439     1677  1 !
;1440     1678  1 !--
```

```
;  1442          1679  2 BEGIN
;  1443          1680  2
;  1444          1681  2 BIND
;  1445          1682  2
;  1446          1683  2         CHAR_MASK       = .P_CHAR_MASK;
;  1447          1684  2
;  1448          1685  2 LOCAL
;  1449          1686  2
;  1450          1687  2         PBCB            : REF BLOCK[,BYTE],     ! Pasteboard control block
;  1451          1688  2         MASK_BLOCK      : VECTOR[2],            ! Character mask block
;  1452          1689  2                                                 ! See figure 9-4 in I/O manual.
;  1453          1690  2         STATUS;
;  1454          1691  2
;  1455          1692  2 BUILTIN
;  1456          1693  2
;  1457          1694  2         ACTUALCOUNT;
;  1458          1695  2
;  1459          1696  2 $SMG$VALIDATE_ARGCOUNT(3,4);
;  1460          1697  2
;  1461          1698  2 !+
;  1462          1699  2 !       Get the PBCB associated with this id and store it in PBCB.
;  1463          1700  2 !-
;  1464          1701  2
;  1465          1702  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  1466          1703  2
;  1467          1704  2 !+
;  1468          1705  2 !       Store the address of the user's AST routine in the PBCB.
;  1469          1706  2 !-
;  1470          1707  2
;  1471          1708  2 PBCB[PBCB_A_BAND_ROUTINE] = .AST_RTN;
;  1472          1709  2
;  1473          1710  2 !+
;  1474          1711  2 !       Store his argument, if one was specified, or 0 if not.
;  1475          1712  2 !-
;  1476          1713  2
;  1477          1714  2 IF ACTUALCOUNT() EQL 4
;  1478          1715  2   THEN  PBCB[PBCB_L_BAND_AST_ARG]=.AST_ARG
;  1479          1716  2   ELSE  PBCB[PBCB_L_BAND_AST_ARG]= 0;
;  1480          1717  2
;  1481          1718  2 !+
;  1482          1719  2 !       Store away the mask for goodness sake in the PBCB
;  1483          1720  2 !       and in our terminator mask block.
;  1484          1721  2 !       The first longword of the terminator mask block must be 0
;  1485          1722  2 !       indicating that it is a short-form mask.
;  1486          1723  2 !-
;  1487          1724  2
;  1488          1725  2 PBCB[PBCB_M_BAND_MASK]=.CHAR_MASK;
;  1489          1726  2 MASK_BLOCK[0]=0;
;  1490          1727  2 MASK_BLOCK[1]=.CHAR_MASK;
;  1491          1728  2
;  1492          1729  2 !+
;  1493          1730  2 ! Issue the appropriate QIO.
;  1494          1731  2 ! The AST routine to service the out-of-band character has
;  1495          1732  2 ! previously been built in the PBCB at offset PBCB_Z_OUT_OF_BAND_RTN.
;  1496          1733  2 ! This handler actually resides within the PBCB.
;  1497          1734  2 ! It is of the form:
;  1498          1735  2 !
```

```
;  1499          1736  2 !        0000    entry mask
;  1500          1737  2 !          FA    CALLG
;  1501          1738  2 !          6C    (AP)
;  1502          1739  2 !          9F    absolute addressing
;  1503          1740  2 !     address    longword address of SMG$$OUT_OF_BAND_HANDLER
;  1504          1741  2 !          04    RET
;  1505          1742  2 !-
;  1506          1743  2
;  1507        P 1744  2 STATUS=$QIOW(    CHAN    =  .PBCB[PBCB_W_CHAN],
;  1508        P 1745  2                  EFN     =  .PBCB[PBCB_B_ASYNC_EFN],
;  1509        P 1746  2                  FUNC    = IO$_SETMODE OR IO$M_OUTBAND,
;  1510        P 1747  2                  P1      = PBCB[PBCB_Z_OUT_OF_BAND_RTN],
;  1511          1748  2                  P2      = MASK_BLOCK);
;  1512          1749  2
;  1513          1750  2 RETURN .STATUS
;  1514          1751  2
;  1515          1752  1 END;


                                                0000 00000          .ENTRY  SMG$SET_OUT_OF_BAND_ASTS, Save nothing      ; 1625
                                5E              08  C2 00002         SUBL2   #8, SP                                      ;
                             50 6C              03  83 00005         SUBB3   #3, (AP), DIFF                              ; 1696
                                01              50  91 00009         CMPB    DIFF, #1                                    ;
                                                08  1B 0000C         BLEQU   1$                                          ;
                                50 00000000G    8F  D0 0000E         MOVL    #SMG$_WRONUMARG, R0                         ;
                                                    04 00015         RET                                                 ;
                                50        04    BC  D0 00016 1$:     MOVL    @PASTEBOARD_ID, R0                          ; 1702
                                                11  19 0001A         BLSS    2$                                          ;
                          00000000G     00      50  D1 0001C         CMPL    R0, PBD_L_COUNT                             ;
                                                08  14 00023         BGTR    2$                                          ;
                       08 00000000G     00      50  E0 00025         BBS     R0, PBD_V_PB_AVAIL, 3$                      ;
                                50 00000000G    8F  D0 0002D 2$:     MOVL    #SMG$_INVPAS_ID, R0                         ;
                                                    04 00034         RET                                                 ;
                                50 00000000G0040    D0 00035 3$:     MOVL    PBD_A_PBCB[R0], PBCB                        ;
                              0098 C0           0C  AC  D0 0003D     MOVL    AST_RTN, 152(PBCB)                          ; 1708
                                04                  6C  91 00043     CMPB    (AP), #4                                    ; 1714
                                                    08  12 00046     BNEQ    4$                                          ;
                              009C C0           10  AC  D0 00048     MOVL    AST_ARG, 156(PBCB)                          ; 1715
                                                    04  11 0004E     BRB     5$                                          ;
                                          009C  C0  D4  00050 4$:    CLRL    156(PBCB)                                   ; 1716
                              00A0 C0           08  BC  D0 00054 5$: MOVL    @P_CHAR_MASK, 160(PBCB)                     ; 1725
                                                    6E  D4 0005A     CLRL    MASK_BLOCK                                  ; 1726
                                04 AE           08  BC  D0 0005C     MOVL    @P_CHAR_MASK, MASK_BLOCK+4                  ; 1727
                                                    7E  7C 00061     CLRQ    -(SP)                                       ; 1748
                                                    7E  7C 00063     CLRQ    -(SP)                                       ;
                                          10    AE  9F  00065        PUSHAB  MASK_BLOCK                                  ;
                                          008C  C0  9F  00068        PUSHAB  140(PBCB)                                   ;
                                                    7E  7C 0006C     CLRQ    -(SP)                                       ;
                                                    7E  D4 0006E     CLRL    -(SP)                                       ;
                                7E        0423    8F  3C 00070     MOVZWL  #1059, -(SP)                                ;
                                7E          64    A0  3C 00075     MOVZWL  100(PBCB), -(SP)                            ;
                                7E          67    A0  9A 00079     MOVZBL  103(PBCB), -(SP)                            ;
                          00000000G     00      0C  FB 0007D     CALLS   #12, SYS$QIOW                               ;
                                                    04 00084         RET                                                 ; 1752
```

; Routine Size: 133 bytes,   Routine Base: _SMG$CODE + 056A

```
;1517     1753  1 %SBTTL 'SMG$ENABLE_UNSOLICITED_INPUT'
;1518     1754  1 GLOBAL ROUTINE SMG$ENABLE_UNSOLICITED_INPUT(
;1519     1755  1         PASTEBOARD_ID,AST_RTN,AST_ARG) =
;1520     1756  1
;1521     1757  1 !++
;1522     1758  1 ! FUNCTIONAL DESCRIPTION:
;1523     1759  1 !
;1524     1760  1 !       This routine enables calling of an AST routine upon
;1525     1761  1 !       receipt of unsolicited input on the specified terminal.
;1526     1762  1 !       The AST routine gets called with 2 arguments:
;1527     1763  1 !       PASTEBOARD_ID.rl.r and AST_ARG.rz.v.  Note that no input
;1528     1764  1 !       characters are read. It is up to the user to read the
;1529     1765  1 !       unsolicited input using normal SMG input routines
;1530     1766  1 !       (or cancel the type-ahead).
;1531     1767  1 !
;1532     1768  1 ! CALLING SEQUENCE:
;1533     1769  1 !
;1534     1770  1 !       RET_STATUS.wlc.v = SMG$ENABLE_UNSOLICITED_INPUT(
;1535     1771  1 !                                  PASTEBOARD_ID.rl.r,
;1536     1772  1 !                                  AST_RTN.szem.r
;1537     1773  1 !                                  [, AST_ARG.rz.v])
;1538     1774  1 !
;1539     1775  1 ! FORMAL PARAMETERS:
;1540     1776  1 !
;1541     1777  1 !       PASTEBOARD_ID   Address of pasteboard id for terminal to
;1542     1778  1 !                       be affected.
;1543     1779  1 !
;1544     1780  1 !       AST_RTN         Address of AST routine to be called
;1545     1781  1 !
;1546     1782  1 !       AST_ARG         Optional value to be passed to AST routine.
;1547     1783  1 !
;1548     1784  1 ! IMPLICIT INPUTS:
;1549     1785  1 !
;1550     1786  1 !       NONE
;1551     1787  1 !
;1552     1788  1 ! IMPLICIT OUTPUTS:
;1553     1789  1 !
;1554     1790  1 !       Internal tables get updated.
;1555     1791  1 !
;1556     1792  1 ! COMPLETION STATUS:
;1557     1793  1 !
;1558     1794  1 !       SS$_NORMAL      Normal successful completion
;1559     1795  1 !       SMG$_WRONUMARG  Wrong number of arguments
;1560     1796  1 !       SS$_xyz         Any error possible from $QIOW
;1561     1797  1 !
;1562     1798  1 ! SIDE EFFECTS:
;1563     1799  1 !
;1564     1800  1 !--
```

```
;  1566          1801  2 BEGIN
;  1567          1802  2
;  1568          1803  2 BUILTIN
;  1569          1804  2
;  1570          1805  2         NULLPARAMETER;
;  1571          1806  2
;  1572          1807  2 LOCAL
;  1573          1808  2
;  1574          1809  2         PBCB            : REF $PBCB_DECL,
;  1575          1810  2         STATUS;
;  1576          1811  2
;  1577          1812  2 EXTERNAL LITERAL
;  1578          1813  2
;  1579          1814  2         SMG$_NOT_A_TRM;
;  1580          1815  2
;  1581          1816  2 $SMG$VALIDATE_ARGCOUNT(2,3);
;  1582          1817  2
;  1583          1818  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  1584          1819  2
;  1585          1820  2 !+
;  1586          1821  2 ! Make sure that the device is a terminal.
;  1587          1822  2 ! If not, return the qualified success, SMG$_NOT_A_TRM.
;  1588          1823  2 !-
;  1589          1824  2
;  1590          1825  2 IF .PBCB[PBCB_B_CLASS] NEQ DC$_TERM
;  1591          1826  2   THEN  RETURN  SMG$_NOT_A_TRM;
;  1592          1827  2
;  1593          1828  2 !+
;  1594          1829  2 ! If we don't already have a mailbox established,
;  1595          1830  2 ! then create a new channel to the terminal with an
;  1596          1831  2 ! associated mailbox.
;  1597          1832  2 !-
;  1598          1833  2
;  1599          1834  2 IF .PBCB[PBCB_W_ASYNC_CHAN] EQL 0
;  1600          1835  3   THEN  BEGIN
;  1601          1836  3         STATUS=INITIALIZE_MAILBOX(.PBCB);
;  1602          1837  3         IF NOT .STATUS THEN RETURN .STATUS
;  1603          1838  2         END;
;  1604          1839  2
;  1605          1840  2 !+
;  1606          1841  2 ! Note that unsolicted input is enabled.
;  1607          1842  2 !-
;  1608          1843  2
;  1609          1844  2 PBCB[PBCB_V_UNSOLICIT]=1;
;  1610          1845  2
;  1611          1846  2 !+
;  1612          1847  2 ! Store the (possibly new) AST routine and argument in our PBCB.
;  1613          1848  2 !-
;  1614          1849  2
;  1615          1850  2 PBCB[PBCB_A_UNSOLICIT_RTN]=.AST_RTN;
;  1616          1851  2 IF NULLPARAMETER(AST_ARG)
;  1617          1852  2   THEN  PBCB[PBCB_L_UNSOLICIT_ARG]=0
;  1618          1853  2   ELSE  PBCB[PBCB_L_UNSOLICIT_ARG]=.AST_ARG;
;  1619          1854  2
;  1620          1855  2 RETURN  SS$_NORMAL
;  1621          1856  2
;  1622          1857  1 END;
```

```
                                        0004 0000C          .ENTRY  SMG$ENABLE_UNSOLICITED_INPUT, Save R2       ; 1754
50              6C              02      83 00002            SUBB3   #2, (AP), DIFF                              ; 1816
                01              50      91 00006            CMPB    DIFF, #1
                                08      1B 00009            BLEQU   1$
                50 00000000G    8F      D0 0000B            MOVL    #SMG$_WRONUMARG, R0
                                        04 00012            RET
                50      04      BC      D0 00013 1$:        MOVL    @PASTEBOARD_ID, R0                          ; 1818
                                11      19 00017            BLSS    2$
    00000000G   00              50      D1 00019            CMPL    R0, PBD_L_COUNT
                                08      14 00020            BGTR    2$
08 00000000G    00              50      E0 00022            BBS     R0, PBD_V_PB_AVAIL, 3$
                50 00000000G    8F      D0 0002A 2$:        MOVL    #SMG$_INVPAS_ID, R0
                                        04 00031            RET
                52 00000000G0040        D0 00032 3$:        MOVL    PBD_A_PBCB[R0], PBCB
          42    8F      58      A2      91 0003A            CMPB    88(PBCB), #66                               ; 1825
                                08      13 0003F            BEQL    4$
                50 00000000G    8F      D0 00041            MOVL    #SMG$_NOT_A_TRM, R0                         ; 1826
                                        04 00048            RET
                        00D2    C2      B5 00049 4$:        TSTW    210(PBCB)                                   ; 1834
                                0A      12 0004D            BNEQ    5$
                                52      DD 0004F            PUSHL   PBCB                                        ; 1836
          FC40  CF                      01      FB 00051    CALLS   #1, INITIALIZE_MAILBOX
                24                      50      E9 00056    BLBC    STATUS, 9$                                  ; 1837
          00D0  C2                      02      88 00059 5$:BISB2   #2, 208(PBCB)                               ; 1844
          00C0  C2        08            AC      D0 0005E    MOVL    AST_RTN, 192(PBCB)                          ; 1850
                03                      6C      91 00064    CMPB    (AP), #3                                    ; 1851
                                        05      1F 00067    BLSSU   6$
                                0C      AC      D5 00069    TSTL    12(AP)
                                        06      12 0006C    BNEQ    7$
                        00C4    C2      D4      0006E 6$:   CLRL    196(PBCB)                                   ; 1852
                                        06      11 00072    BRB     8$
                00C4    C2      0C      AC      D0 00074 7$:MOVL    AST_ARG, 196(PBCB)                          ; 1853
                50                      01      D0 0007A 8$:MOVL    #1, R0                                      ; 1855
                                                04 0007D 9$:RET                                                 ; 1857

; Routine Size:  126 bytes,    Routine Base:  _SMG$CODE + 05EF
```

```
;  1624          1858  1 %SBTTL 'SMG$DISABLE_UNSOLICITED_INPUT'
;  1625          1859  1 GLOBAL ROUTINE SMG$DISABLE_UNSOLICITED_INPUT(
;  1626          1860  1         PASTEBOARD_ID) =
;  1627          1861  1
;  1628          1862  1 !++
;  1629          1863  1 ! FUNCTIONAL DESCRIPTION:
;  1630          1864  1 !
;  1631          1865  1 !       This routine disables unsolicited input ASTs for the
;  1632          1866  1 !       specified terminal.
;  1633          1867  1 !
;  1634          1868  1 ! CALLING SEQUENCE:
;  1635          1869  1 !
;  1636          1870  1 !       RET_STATUS.wlc.v = SMG$DISABLE_UNSOLICITED_INPUT(
;  1637          1871  1 !                                  PASTEBOARD_ID.rl.r)
;  1638          1872  1 !
;  1639          1873  1 ! FORMAL PARAMETERS:
;  1640          1874  1 !
;  1641          1875  1 !       PASTEBOARD_ID   Pasteboard id for terminal to
;  1642          1876  1 !                       be affected.
;  1643          1877  1 !
;  1644          1878  1 ! IMPLICIT INPUTS:
;  1645          1879  1 !
;  1646          1880  1 !       Internal tables.
;  1647          1881  1 !
;  1648          1882  1 ! IMPLICIT OUTPUTS:
;  1649          1883  1 !
;  1650          1884  1 !       Internal tables get updated.
;  1651          1885  1 !
;  1652          1886  1 ! COMPLETION STATUS:
;  1653          1887  1 !
;  1654          1888  1 !       SS$_NORMAL      Normal successful completion
;  1655          1889  1 !       SMG$_WRONUMARG  Wrong number of arguments
;  1656          1890  1 !       SS$_xyz         Any error possible from $QIOW
;  1657          1891  1 !
;  1658          1892  1 ! SIDE EFFECTS:
;  1659          1893  1 !
;  1660          1894  1 !--
```

```
;  1662          1895  2 BEGIN
;  1663          1896  2
;  1664          1897  2 LOCAL
;  1665          1898  2
;  1666          1899  2         PBCB            : REF $PBCB_DECL,
;  1667          1900  2         STATUS;
;  1668          1901  2
;  1669          1902  2 EXTERNAL LITERAL
;  1670          1903  2
;  1671          1904  2         SMG$_NOT_A_TRM;
;  1672          1905  2
;  1673          1906  2 $SMG$VALIDATE_ARGCOUNT(1,1);
;  1674          1907  2
;  1675          1908  2 $SMG$GET_PBCB(.PASTEBOARD_ID,PBCB);
;  1676          1909  2
;  1677          1910  2 !+
;  1678          1911  2 ! Make sure that the device is a terminal.
;  1679          1912  2 ! If not, return the qualified success, SMG$_NOT_A_TRM.
;  1680          1913  2 !-
;  1681          1914  2
;  1682          1915  2 IF .PBCB[PBCB_B_CLASS] NEQ DC$_TERM
;  1683          1916  2   THEN  RETURN  SMG$_NOT_A_TRM;
;  1684          1917  2
;  1685          1918  2 !+
;  1686          1919  2 ! Note that unsolicted input is disabled.
;  1687          1920  2 !-
;  1688          1921  2
;  1689          1922  2 PBCB[PBCB_V_UNSOLICIT]=0;
;  1690          1923  2 PBCB[PBCB_A_UNSOLICIT_RTN]=0;
;  1691          1924  2 PBCB[PBCB_L_UNSOLICIT_ARG]=0;
;  1692          1925  2
;  1693          1926  2 RETURN  SS$_NORMAL
;  1694          1927  2
;  1695          1928  1 END;


                                                0000 00000         .ENTRY  SMG$DISABLE_UNSOLICITED_INPUT, Save nothing ; 1859
                                  01            6C   91 00002         CMPB    (AP), #1                                    ; 1906
                                                08   13 00005         BEQL    1$                                          ;
                                  50 00000000G  8F   D0 00007         MOVL    #SMG$_WRONUMARG, R0                         ;
                                                     04 0000E         RET                                                 ;
                                  50        04  BC   D0 0000F 1$:      MOVL    @PASTEBOARD_ID, R0                          ; 1908
                                                11   19 00013         BLSS    2$                                          ;
                      00000000G   00            50   D1 00015         CMPL    R0, PBD_L_COUNT                             ;
                                                08   14 0001C         BGTR    2$                                          ;
                   08 00000000G   00            50   E0 0001E         BBS     R0, PBD_V_PB_AVAIL, 3$                      ;
                                  50 00000000G  8F   D0 00026 2$:      MOVL    #SMG$_INVPAS_ID, R0                         ;
                                                     04 0002D         RET                                                 ;
                                  50 00000000G00040  D0 0002E 3$:      MOVL    PBD_A_PBCB[R0], PBCB                        ;
                            42    8F        58   A0   91 00036         CMPB    88(PBCB), #66                               ; 1915
                                                08   13 0003B         BEQL    4$                                          ;
                                  50 00000000G  8F   D0 0003D         MOVL    #SMG$_NOT_A_TRM, R0                         ; 1916
                                                     04 00044         RET                                                 ;
                          00D0    C0            02   8A 00045 4$:      BICB2   #2, 208(PBCB)                               ; 1922
```

```
                                 00C0   C0  7C 0004A          CLRQ    192(PBCB)                                ; 1923
                          50             01  D0 0004E          MOVL    #1, R0                                   ; 1926
                                             04 00051          RET                                              ; 1928

; Routine Size:  82 bytes,    Routine Base:  _SMG$CODE + 066D
```

```
;1697            1929  1 %SBTTL 'SMG$$SET_TERM_CHARACTERISTICS'
;1698            1930  1 GLOBAL ROUTINE SMG$$SET_TERM_CHARACTERISTICS (
;1699            1931  1         PBID, ON_CHARACTERISTICS1, ON_CHARACTERISTICS2,
;1700            1932  1         OFF_CHARACTERISTICS1, OFF_CHARACTERISTICS2,
;1701            1933  1         P_HANDLE,
;1702            1934  1         P_OLD_CHARACTERISTICS1, P_OLD_CHARACTERISTICS2 ) =
;1703            1935  1
;1704            1936  1 !++
;1705            1937  1 ! FUNCTIONAL DESCRIPTION:
;1706            1938  1 !
;1707            1939  1 !       This routine changes the terminal characteristics
;1708            1940  1 !       for a given pasteboard.
;1709            1941  1 !       It also establishes an exit handler that will restore
;1710            1942  1 !       the characteristics back to what they were on image exit.
;1711            1943  1 !
;1712            1944  1 ! CALLING SEQUENCE:
;1713            1945  1 !
;1714            1946  1 !       RET_STATUS.wlc.v = SMG$$SET_TERM_CHARACTERISTICS (
;1715            1947  1 !                                  PBID.rl.r
;1716            1948  1 !                                  [,ON_CHARACTERISTICS1.rl.v]
;1717            1949  1 !                                  [,ON_CHARACTERISTICS2.rl.v]
;1718            1950  1 !                                  [,OFF_CHARACTERISTICS1.rl.v]
;1719            1951  1 !                                  [,OFF_CHARACTERISTICS2.rl.v]
;1720            1952  1 !                                  [,P_HANDLE.wl.r]
;1721            1953  1 !                                  [,P_OLD_CHARACTERISTICS1.wl.r]
;1722            1954  1 !                                  [,P_OLD_CHARACTERISTICS2.wl.r])
;1723            1955  1 !
;1724            1956  1 ! FORMAL PARAMETERS:
;1725            1957  1 !
;1726            1958  1 !       PBID.rl.r                       pasteboard id
;1727            1959  1 !       ON_CHARACTERITSICS1.rl.v        bits to turn on in 1st characteristics
;1728            1960  1 !       ON_CHARACTERITSICS2.rl.v        bits to turn on in 2nd characteristics
;1729            1961  1 !       OFF_CHARACTERITSICS1.rl.v       bits to turn off in 1st characteristics
;1730            1962  1 !       OFF_CHARACTERITSICS2.rl.v       bits to turn off in 2nd characteristics
;1731            1963  1 !       P_HANDLE.wl.r                   longword to receive "handle" for
;1732            1964  1 !                                       this request.  This handle value may
;1733            1965  1 !                                       be used to refer back to this
;1734            1966  1 !                                       request in a subsequent call to
;1735            1967  1 !                                       SMG$$REVERT_CHARACTERISTICS.
;1736            1968  1 !       P_OLD_CHARACTERISTICS1.wl.r     gets old characteristics (part 1)
;1737            1969  1 !       P_OLD_CHARACTERISTICS2.wl.r     gets old characteristics (part 2)
;1738            1970  1 !
;1739            1971  1 ! IMPLICIT INPUTS:
;1740            1972  1 !
;1741            1973  1 !       Terminal characteristics
;1742            1974  1 !
;1743            1975  1 ! IMPLICIT OUTPUTS:
;1744            1976  1 !
;1745            1977  1 !       NONE
;1746            1978  1 !
;1747            1979  1 ! COMPLETION STATUS:
;1748            1980  1 !
;1749            1981  1 !       SS$_NORMAL      Normal successful completion
;1750            1982  1 !       SMG$_NOT_A_TRM  Success - but device is not a terminal
;1751            1983  1 !       LIB$_xyz        Errors from LIB$GET_VM
;1752            1984  1 !       SS$_xyz         Errors from $DCLEXH or $QIOW
;1753            1985  1 !
```

```
; 1754          1986  1 ! SIDE EFFECTS:
; 1755          1987  1 !
; 1756          1988  1 !--
```

```
;  1758          1989  2 BEGIN
;  1759          1990  2
;  1760          1991  2 BIND
;  1761          1992  2
;  1762          1993  2         HANDLE          = .P_HANDLE,
;  1763          1994  2         OLD_CHAR1       = .P_OLD_CHARACTERISTICS1,
;  1764          1995  2         OLD_CHAR2       = .P_OLD_CHARACTERISTICS2;
;  1765          1996  2
;  1766          1997  2 EXTERNAL LITERAL
;  1767          1998  2
;  1768          1999  2         SMG$_NOT_A_TRM;         ! Device is not a terminal (success)
;  1769          2000  2
;  1770          2001  2 LOCAL
;  1771          2002  2
;  1772          2003  2         PBCB    : REF BLOCK[,BYTE],     ! Pasteboard control block
;  1773          2004  2         TTIOSB  : VECTOR[4,WORD],       ! I/O status block
;  1774          2005  2         EXIT_BLOCK                      ! Address of exit block
;  1775          2006  2                 : REF VECTOR[SMG$S_CHAR_EXIT_BLOCK/4],
;  1776          2007  2         OLD_CHARBUF     : BLOCK[12,BYTE],       ! Old characteristics buffer
;  1777          2008  2         CHAR1,                          ! Old characteristics (part 1)
;  1778          2009  2         CHAR2,                          ! Old characteristics (part 2)
;  1779          2010  2         STATUS;
;  1780          2011  2
;  1781          2012  2 BUILTIN
;  1782          2013  2
;  1783          2014  2         NULLPARAMETER;
;  1784          2015  2
;  1785          2016  2 EXTERNAL ROUTINE
;  1786          2017  2
;  1787          2018  2         LIB$GET_VM;
```

```
;  1789          2019  2 !+
;  1790          2020  2 ! Get the pasteboard control block from the pasteboard id.
;  1791          2021  2 !-
;  1792          2022  2
;  1793          2023  2 $SMG$GET_PBCB(.PBID,PBCB);
;  1794          2024  2
;  1795          2025  2 !+
;  1796          2026  2 ! Check that the output device is really a terminal.
;  1797          2027  2 ! If it isn't then, return qualified success.
;  1798          2028  2 !-
;  1799          2029  2
;  1800          2030  2 IF .PBCB[PBCB_B_CLASS] NEQ DC$_TERM
;  1801          2031  3   THEN  BEGIN
;  1802          2032  3         IF NOT NULLPARAMETER(7)
;  1803          2033  3           THEN  OLD_CHAR1=0;
;  1804          2034  3
;  1805          2035  3         IF NOT NULLPARAMETER(8)
;  1806          2036  3           THEN  OLD_CHAR2=0;
;  1807          2037  3
;  1808          2038  3         RETURN  SMG$_NOT_A_TRM
;  1809          2039  3
;  1810          2040  2         END;
;  1811          2041  2
;  1812          2042  2 !+
;  1813          2043  2 ! Save the old characteristics buffer.
;  1814          2044  2 !-
;  1815          2045  2
;  1816          2046  2 CH$MOVE(12,PBCB[PBCB_R_CHARBUF],OLD_CHARBUF);
;  1817          2047  2
;  1818          2048  2 !+
;  1819          2049  2 ! Get the old characteristics and return them to the user
;  1820          2050  2 ! if he asked for them.
;  1821          2051  2 !-
;  1822          2052  2
;  1823          2053  2 CHAR1=.PBCB[PBCB_L_DEVDEPEND];
;  1824          2054  2 CHAR2=.PBCB[PBCB_L_DEVDEPEND2];
;  1825          2055  2
;  1826          2056  2 IF NOT NULLPARAMETER(7)
;  1827          2057  2   THEN  OLD_CHAR1=.CHAR1;
;  1828          2058  2
;  1829          2059  2 IF NOT NULLPARAMETER(8)
;  1830          2060  2   THEN  OLD_CHAR2=.CHAR2;
;  1831          2061  2
;  1832          2062  2 !+
;  1833          2063  2 ! OR in the new characteristic bits requested on.
;  1834          2064  2 !-
;  1835          2065  2
;  1836          2066  2 IF NOT NULLPARAMETER(2)
;  1837          2067  2   THEN  PBCB[PBCB_L_DEVDEPEND]=
;  1838          2068  2                 .PBCB[PBCB_L_DEVDEPEND] OR .ON_CHARACTERISTICS1;
;  1839          2069  2
;  1840          2070  2 IF NOT NULLPARAMETER(3)
;  1841          2071  2   THEN  PBCB[PBCB_L_DEVDEPEND2]=
;  1842          2072  2                 .PBCB[PBCB_L_DEVDEPEND2] OR .ON_CHARACTERISTICS2;
;  1843          2073  2
;  1844          2074  2 !+
;  1845          2075  2 ! AND out the characteristics bits requested off.
```

```
; 1846          2076  2 !-
; 1847          2077  2
; 1848          2078  2 IF NOT NULLPARAMETER(4)
; 1849          2079  2   THEN  PBCB[PBCB_L_DEVDEPEND]=
; 1850          2080  2                 .PBCB[PBCB_L_DEVDEPEND] AND (NOT .OFF_CHARACTERISTICS1);
; 1851          2081  2
; 1852          2082  2 IF NOT NULLPARAMETER(5)
; 1853          2083  2   THEN  PBCB[PBCB_L_DEVDEPEND2]=
; 1854          2084  2                 .PBCB[PBCB_L_DEVDEPEND2] AND (NOT .OFF_CHARACTERISTICS2);
; 1855          2085  2
; 1856          2086  2 !+
; 1857          2087  2 ! Set the terminal to have the desired new characteristics.
; 1858          2088  2 !-
; 1859          2089  2
; 1860        P 2090  2 STATUS=$QIOW(   CHAN    = .PBCB[PBCB_W_CHAN],
; 1861        P 2091  2                 FUNC    = IO$_SETMODE,
; 1862        P 2092  2                 IOSB    = TTIOSB,
; 1863        P 2093  2                 P1      = PBCB[PBCB_R_CHARBUF],
; 1864          2094  2                 P2      = 12);
; 1865          2095  2 IF NOT .STATUS THEN RETURN .STATUS;
; 1866          2096  2 IF NOT .TTIOSB[0] THEN RETURN .TTIOSB[0];
; 1867          2097  2
; 1868          2098  2 !+
; 1869          2099  2 ! Create an exit block to be used to set the characteristics back.
; 1870          2100  2 !-
; 1871          2101  2
; 1872          2102  2 STATUS=LIB$GET_VM(%REF(SMG$S_CHAR_EXIT_BLOCK),EXIT_BLOCK);
; 1873          2103  2 IF NOT .STATUS THEN RETURN .STATUS;
; 1874          2104  2
; 1875          2105  2 !+
; 1876          2106  2 ! Give the user back the address of this exit block as the "handle".
; 1877          2107  2 !-
; 1878          2108  2
; 1879          2109  2 IF NOT NULLPARAMETER(6)
; 1880          2110  2   THEN  HANDLE=.EXIT_BLOCK;
; 1881          2111  2
; 1882          2112  2 !+
; 1883          2113  2 ! Set up the exit block.
; 1884          2114  2 !-
; 1885          2115  2
; 1886          2116  2 EXIT_BLOCK[0]=0;                    ! reserved for use by VMS as forward link
; 1887          2117  2 EXIT_BLOCK[1]=CHAR_EXIT_HANDLER;    ! specify our exit handler
; 1888          2118  2 EXIT_BLOCK[2]=6;                    ! six arguments
; 1889          2119  2 EXIT_BLOCK[3]=EXIT_BLOCK[4];        ! Store reason in next entry in exit block
; 1890          2120  2 EXIT_BLOCK[4]=0;                    ! Gets reason for exit
; 1891          2121  2 EXIT_BLOCK[5]=.PBCB[PBCB_W_CHAN]; ! Terminal channel
; 1892          2122  2 !+
; 1893          2123  2 ! Longwords 6,7,8 form a 12-byte characteristics buffer.
; 1894          2124  2 !-
; 1895          2125  2 CH$MOVE(12,OLD_CHARBUF,EXIT_BLOCK[6]);
; 1896          2126  2
; 1897          2127  2 !+
; 1898          2128  2 !   clare an exit handler using this exit block.
; 1899          2129  2 !-
; 1900          2130  2
; 1901          2131  2 STATUS=$DCLEXH( DESBLK=.EXIT_BLOCK);
; 1902          2132  2 IF NOT .STATUS THEN RETURN .STATUS;
```

```
;1903          2133  2
;1904          2134  2 RETURN SS$_NORMAL
;1905          2135  2
;1906          2136  1 END;


                                                                .EXTRN  SYS$DCLEXH

                                                01FC 00000      .ENTRY  SMG$$SET_TERM_CHARACTERISTICS, Save R2,R3,- ;1930
                                                                        R4,R5,R6,R7,R8
                                5E          1C  C2 00002        SUBL2   #28, SP
                                50      04  BC  D0 00005        MOVL    @PBID, R0                                   ;2023
                                            11  19 00009        BLSS    1$
                     00000000G  00          50  D1 0000B        CMPL    R0, PBD_L_COUNT
                                            08  14 00012        BGTR    1$
                  08 00000000G  00          50  E0 00014        BBS     R0, PBD_V_PB_AVAIL, 2$
                                50 00000000G  8F  D0 0001C 1$:    MOVL    #SMG$_INVPAS_ID, R0
                                              04  00023        RET
                                57 00000000G0040  D0 00024 2$:    MOVL    PBD_A_PBCB[R0], PBCB
                              42  8F      58  A7  91 0002C        CMPB    88(PBCB), #66                               ;2030
                                            22  13 00031        BEQL    5$
                                07          6C  91 00033        CMPB    (AP), #7                                    ;2032
                                            08  1F 00036        BLSSU   3$
                                        1C  AC  D5 00038        TSTL    28(AP)
                                            03  13 0003B        BEQL    3$
                                        1C  BC  D4 0003D        CLRL    @P_OLD_CHARACTERISTICS1                     ;2033
                                08          6C  91 00040 3$:    CMPB    (AP), #8                                    ;2035
                                            08  1F 00043        BLSSU   4$
                                        20  AC  D5 00045        TSTL    32(AP)
                                            03  13 00048        BEQL    4$
                                        20  BC  D4 0004A        CLRL    @P_OLD_CHARACTERISTICS2                     ;2036
                                50 00000000G  8F  D0 0004D 4$:    MOVL    #SMG$_NOT_A_TRM, R0                         ;2038
                                              04  00054        RET
                08    AE        58  A7          0C  28 00055 5$:    MOVC3   #12, 88(PBCB), OLD_CHARBUF                  ;2046
                                50      5C  A7  7D 0005B        MOVQ    92(PBCB), CHAR1                             ;2053
                                07          6C  91 0005F        CMPB    (AP), #7                                    ;2056
                                            09  1F 00062        BLSSU   6$
                                        1C  AC  D5 00064        TSTL    28(AP)
                                            04  13 00067        BEQL    6$
                                1C  BC          50  D0 00069        MOVL    CHAR1, @P_OLD_CHARACTERISTICS1          ;2057
                                08          6C  91 0006D 6$:    CMPB    (AP), #8                                    ;2059
                                            09  1F 00070        BLSSU   7$
                                        20  AC  D5 00072        TSTL    32(AP)
                                            04  13 00075        BEQL    7$
                                20  BC          51  D0 00077        MOVL    CHAR2, @P_OLD_CHARACTERISTICS2          ;2060
                                02          6C  91 0007B 7$:    CMPB    (AP), #2                                    ;2066
                                            0A  1F 0007E        BLSSU   8$
                                        08  AC  D5 00080        TSTL    8(AP)
                                            05  13 00083        BEQL    8$
                                5C  A7      08  AC  C8 00085        BISL2   ON_CHARACTERISTICS1, 92(PBCB)        ;2068
                                03          6C  91 0008A 8$:    CMPB    (AP), #3                                    ;2070
                                            0A  1F 0008D        BLSSU   9$
                                        0C  AC  D5 0008F        TSTL    12(AP)
                                            05  13 00092        BEQL    9$
                                60  A7      0C  AC  C8 00094        BISL2   ON_CHARACTERISTICS2, 96(PBCB)        ;2072
                                04          6C  91 00099 9$:    CMPB    (AP), #4                                    ;2078
```

```
                              0A  1F 0009C          BLSSU   10$                                        ;
                      10      AC  D5 0009E          TSTL    16(AP)                                     ;
                              05  13 000A1          BEQL    10$                                        ;
          5C     A7   10      AC  CA 000A3          BICL2   OFF_CHARACTERISTICS1, 92(PBCB)             ; 2080
                 05           6C  91 000A8 10$:     CMPB    (AP), #5                                   ; 2082
                              0A  1F 000AB          BLSSU   11$                                        ;
                      14      AC  D5 000AD          TSTL    20(AP)                                     ;
                              05  13 000B0          BEQL    11$                                        ;
          60     A7   14      AC  CA 000B2          BICL2   OFF_CHARACTERISTICS2, 96(PBCB)             ; 2084
                              7E  7C 000B7 11$:     CLRQ    -(SP)                                      ; 2094
                              7E  7C 000B9          CLRQ    -(SP)                                      ;
                              0C  DD 000BB          PUSHL   #12                                        ;
                      58      A7  9F 000BD          PUSHAB  88(PBCB)                                   ;
                              7E  7C 000C0          CLRQ    -(SP)                                      ;
                      34      AE  9F 000C2          PUSHAB  TTIOSB                                     ;
                              23  DD 000C5          PUSHL   #35                                        ;
                 7E   64      A7  3C 000C7          MOVZWL  100(PBCB), -(SP)                           ;
                              7E  D4 000CB          CLRL    -(SP)                                      ;
     00000000G   00           0C  FB 000CD          CALLS   #12, SYS$QIOW                              ;
                 58           50  D0 000D4          MOVL    R0, STATUS                                 ;
                 61           58  E9 000D7          BLBC    STATUS, 14$                                ; 2095
                 05   14      AE  E8 000DA          BLBS    TTIOSB, 12$                                ; 2096
                 50   14      AE  3C 000DE          MOVZWL  TTIOSB, R0                                 ;
                              04     000E2          RET                                                ;
                      04      AE  9F 000E3 12$:     PUSHAB  EXIT_BLOCK                                 ; 2102
          04     AE           24  D0 000E6          MOVL    #36, 4(SP)                                 ;
                      04      AE  9F 000EA          PUSHAB  4(SP)                                      ;
     00000000G   00           02  FB 000ED          CALLS   #2, LIB$GET_VM                             ;
                 58           50  D0 000F4          MOVL    R0, STATUS                                 ;
                 41           58  E9 000F7          BLBC    STATUS, 14$                                ; 2103
                 06           6C  91 000FA          CMPB    (AP), #6                                   ; 2109
                              0A  1F 000FD          BLSSU   13$                                        ;
                      18      AC  D5 000FF          TSTL    24(AP)                                     ;
                              05  13 00102          BEQL    13$                                        ;
          18     BC   04      AE  D0 00104          MOVL    EXIT_BLOCK, @P_HANDLE                      ; 2110
                 56   04      AE  D0 00109 13$:     MOVL    EXIT_BLOCK, R6                             ; 2116
                              66  D4 0010D          CLRL    (R6)                                       ;
          04     A6 0000V     CF  9E 0010F          MOVAB   CHAR_EXIT_HANDLER, 4(R6)                   ; 2117
          08     A6           06  D0 00115          MOVL    #6, 8(R6)                                  ; 2118
          0C     A6   10      A6  9E 00119          MOVAB   16(R6), 12(R6)                             ; 2119
                      10      A6  D4 0011E          CLRL    16(R6)                                     ; 2120
          14     A6   64      A7  3C 00121          MOVZWL  100(PBCB), 20(R6)                          ; 2121
  18   A6 08     AE           0C  28 00126          MOVC3   #12, OLD_CHARBUF, 24(R6)                   ; 2125
                              56  DD 0012C          PUSHL   R6                                         ; 2131
     00000000G   00           01  FB 0012E          CALLS   #1, SYS$DCLEXH                             ;
                 58           50  D0 00135          MOVL    R0, STATUS                                 ;
                 04           58  E8 00138          BLBS    STATUS, 15$                                ; 2132
                 50           58  D0 0013B 14$:     MOVL    STATUS, R0                                 ;
                              04     0013E          RET                                                ;
                 50           01  D0 0013F 15$:     MOVL    #1, R0                                     ; 2134
                              04     00142          RET                                                ; 2136

; Routine Size:  323 bytes,    Routine Base:  _SMG$CODE + 06BF
```

```
;1908           2137  1 %SBTTL 'CHAR_EXIT_HANDLER - exit handler to reset terminal characteristics'
;1909           2138  1 ROUTINE CHAR_EXIT_HANDLER(RSN_ADR,RSN,CHAN,CHARBUF : VECTOR[3,LONG]) =
;1910           2139  1
;1911           2140  1 !++
;1912           2141  1 ! FUNCTIONAL DESCRIPTION:
;1913           2142  1 !
;1914           2143  1 !       This routine is established as an exit handler.
;1915           2144  1 !       It restores the terminal to it's previous characteristics.
;1916           2145  1 !
;1917           2146  1 ! CALLING SEQUENCE:
;1918           2147  1 !
;1919           2148  1 !       RET_STATUS.wlc.v = CHAR_EXIT_HANDLER( P_EXIT_BLOCK.rab.r)
;1920           2149  1 !
;1921           2150  1 ! FORMAL PARAMETERS:
;1922           2151  1 !
;1923           2152  1 !       P_EXIT_BLOCK.rab.r      Exit block as set up by
;1924           2153  1 !                               SMG$$SET_TERMINAL_CHARACTERISTICS
;1925           2154  1 !
;1926           2155  1 ! IMPLICIT INPUTS:
;1927           2156  1 !
;1928           2157  1 !       NONE
;1929           2158  1 !
;1930           2159  1 ! IMPLICIT OUTPUTS:
;1931           2160  1 !
;1932           2161  1 !       NONE
;1933           2162  1 !
;1934           2163  1 ! COMPLETION STATUS:
;1935           2164  1 !
;1936           2165  1 !       SS$_NORMAL      Normal successful completion
;1937           2166  1 !
;1938           2167  1 !               Errors are ignored.
;1939           2168  1 !
;1940           2169  1 ! SIDE EFFECTS:
;1941           2170  1 !
;1942           2171  1 !       Previous terminal characteristics are restored
;1943           2172  1 !
;1944           2173  1 !--
```

```
;  1946          2174  2 BEGIN
;  1947          2175  2
;  1948          2176  2 BIND
;  1949          2177  2
;  1950          2178  2         CLASS           = CHARBUF       : BYTE;
;  1951          2179  2
;  1952          2180  2 LOCAL
;  1953          2181  2
;  1954          2182  2         STATUS;
;  1955          2183  2
;  1956          2184  2 !+
;  1957          2185  2 ! Verify that the device class to be restored is DC$_TERM.
;  1958          2186  2 ! If not, just go away.
;  1959          2187  2 ! Give no error, since this is an exit handler.
;  1960          2188  2 ! If it is a terminal, then restore the old characteristics.
;  1961          2189  2 !-
;  1962          2190  2
;  1963          2191  2 IF .CLASS EQL DC$_TERM
;  1964        P 2192  2   THEN  $QIOW(  CHAN    = .CHAN,
;  1965        P 2193  2                 FUNC    = IO$_SETMODE,
;  1966        P 2194  2                 P1      = CHARBUF,
;  1967          2195  2                 P2      = 12);
;  1968          2196  2
;  1969          2197  2 RETURN  SS$_NORMAL
;  1970          2198  2
;  1971          2199  1 END;


                                        0000 00000 CHAN_EXIT_HANDLER:
                                                   .WORD   Save nothing                    ;  2138
                          42   8F       10   AC  91 00002          CMPB    CLASS, #66      ;  2191
                                             1A  12 00007          BNEQ    1$              ;
                                             7E  7C 00009          CLRQ    -(SP)           ;  2195
                                             7E  7C 0000B          CLRQ    -(SP)           ;
                                             0C  DD 0000D          PUSHL   #12             ;
                                         10  AC  9F 0000F          PUSHAB  CHARBUF         ;
                                             7E  7C 00012          CLRQ    -(SP)           ;
                                         7E      23  7D 00014      MOVQ    #35, -(SP)      ;
                                         0C  AC  DD 00017          PUSHL   CHAN            ;
                                             7E  D4 0001A          CLRL    -(SP)           ;
                  00000000G              00       0C  FB 0001C     CALLS   #12, SYS$QIOW   ;
                                         50       01  D0 00023 1$: MOVL    #1, R0          ;  2197
                                                  04  00026       RET                     ;  2199

; Routine Size:  39 bytes,    Routine Base:  _SMG$CODE + 0802
```

```
; 1973          2200  1 END
; 1974          2201  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                               PSECT SUMMARY
;
;       Name                    Bytes                   Attributes
;
;   _SMG$CODE                     2089  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   _SMG$DATA                        4  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)


;                           Library Statistics
;
;                                       -------- Symbols --------      Pages       Processing
;       File                            Total    Loaded   Percent      Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1     9776        44         0         581       00:00.9
;   _$255$DUA28:[SMGRTL.OBJ]RTLLIB.L32;1    36         0         0           8       00:00.1
;   _$255$DUA28:[SMGRTL.OBJ]SMGLIB.L32;1   469        48        10          38       00:00.3


;                                   COMMAND QUALIFIERS

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

; Size:          2089 code + 4 data bytes
; Run Time:        00:43.3
; Elapsed Time:    02:02.0
; Lines/CPU Min:    3051
; Lexemes/CPU-Min: 21217
; Memory Used:  182 pages
; Compilation Complete
```

SMGNUMTAB
LIS

SMGMSGPTR
LIS

SMGSCROLL
LIS

SMGMISC
LIS

SMGMSGTXT
LIS

SMGPUTENC
LIS

SMGPUTTEX
LIS

SMGSIMTRM
LIS

SMGNUMPAR
LIS

SMGPRVINP
LIS