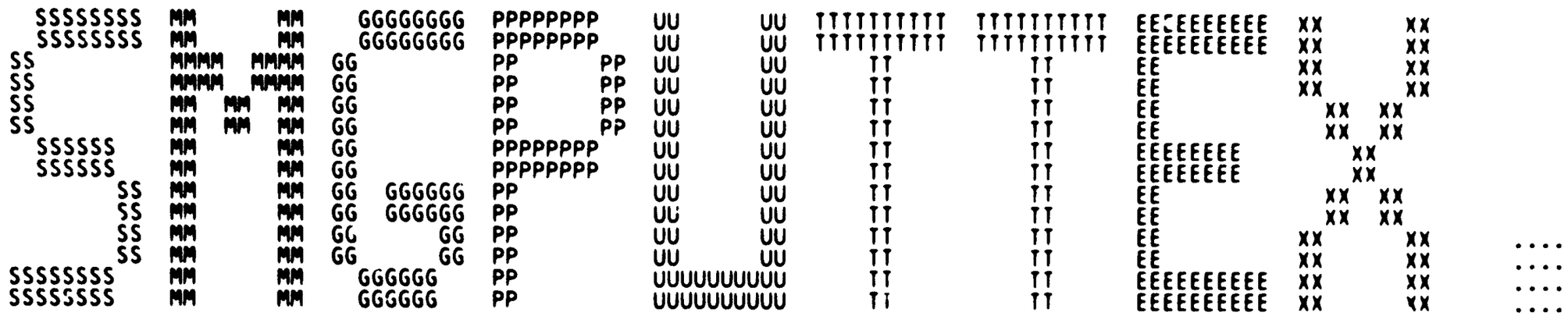
K 7

```
   SSSSSSSSSSSS  MMM         MMM     GGGGGGGGGGGG  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
   SSSSSSSSSSSS  MMM         MMM     GGGGGGGGGGGG  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
   SSSSSSSSSSSS  MMM         MMM     GGGGGGGGGGGG  RRRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
SSS              MMMMMM   MMMMMM  GGG              RRR         RRR        TTT        LLL
SSS              MMMMMM   MMMMMM  GGG              RRR         RRR        TTT        LLL
SSS              MMMMMM   MMMMMM  GGG              RRR         RRR        TTT        LLL
SSS              MMM   MMM   MMM  GGG              RRR         RRR        TTT        LLL
SSS              MMM   MMM   MMM  GGG              RRR         RRR        TTT        LLL
SSS              MMM   MMM   MMM  GGG              RRR         RRR        TTT        LLL
   SSSSSSSSS     MMM         MMM  GGG              RRRRRRRRRRRR           TTT        LLL
   SSSSSSSSS     MMM         MMM  GGG              RRRRRRRRRRRR           TTT        LLL
   SSSSSSSSS     MMM         MMM  GGG              RRRRRRRRRRRR           TTT        LLL
            SSS  MMM         MMM  GGG   GGGGGGGGG  RRR   RRR              TTT        LLL
            SSS  MMM         MMM  GGG   GGGGGGGGG  RRR   RRR              TTT        LLL
            SSS  MMM         MMM  GGG   GGGGGGGGG  RRR   RRR              TTT        LLL
            SSS  MMM         MMM  GGG         GGG  RRR      RRR           TTT        LLL
            SSS  MMM         MMM  GGG         GGG  RRR      RRR           TTT        LLL
            SSS  MMM         MMM  GGG         GGG  RRR      RRR           TTT        LLL
SSSSSSSSSSSS     MMM         MMM     GGGGGGGGG     RRR         RRR        TTT        LLLLLLLLLLLLLLL
SSSSSSSSSSSS     MMM         MMM     GGGGGGGGG     RRR         RRR        TTT        LLLLLLLLLLLLLLL
SSSSSSSSSSSS     MMM         MMM     GGGGGGGGG     RRR         RRR        TTT        LLLLLLLLLLLLLLL
```

FILEID**SMGPUTTEX



```
;     1          0001  0 MODULE SMG$$PUT_TEXT_TO_BUFFER ( %TITLE 'Put text to display buffer'
;001 PLL1013     0002  0                  IDENT = '1-013'          ! File: SMGPUTTEX.B32 Edit: PLL1013
;     3-1        0003  0                  ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !****************************************************************************
;     7          0007  1 !*                                                                          *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                     *
;    11          0011  1 !*                                                                          *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    13          0013  1 !*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE          *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER     *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY      *
;    17          0017  1 !*  TRANSFERRED.                                                             *
;    18          0018  1 !*                                                                          *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE     *
;    20          0020  1 !*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT         *
;    21          0021  1 !*  CORPORATION.                                                             *
;    22          0022  1 !*                                                                          *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS     *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    25          0025  1 !*                                                                          *
;    26          0026  1 !*                                                                          *
;    27          0027  1 !****************************************************************************
;    28          0028  1 !
;    29          0029  1
;    30          0030  1 !++
;    31          0031  1 ! FACILITY:     Screen Management
;    32          0032  1 !
;    33          0033  1 ! ABSTRACT:
;    34          0034  1 !
;    35          0035  1 !       This is an internal routine used by screen management procedures to
;    36          0036  1 !       place user's text into a display buffer.  The text is spanned for
;    37          0037  1 !       special characters.
;    38          0038  1 !
;    39          0039  1 ! ENVIRONMENT:  User mode - AST reentrant
;    40          0040  1 !
;    41          0041  1 ! AUTHOR: P. Levesque, CREATION DATE: 14-Apr-1983
;    42          0042  1 !
;    43          0043  1 ! MODIFIED BY:
;    44          0044  1 !
;    45          0045  1 ! 1-001 - Original.  PLL 14-Apr-1983
;    46          0046  1 ! 1-002 - Finish coding.  PLL 20-Apr-1983
;    47          0047  1 ! 1-003 - Add error message, character set buffer allocation.  PLL 4-May-1983
;    48          0048  1 ! 1-004 - Fix second half of the scan table to agree with actions for
;    49          0049  1 !         DEC Multinational.  PLL 5-May-1983
;    50          0050  1 ! 1-005 - If on the last line and we have found a line feed, scroll.  PLL 11-May-1983
;    51          0051  1 ! 1-006 - If a bell character is found, call SMG$RING_BELL instead of setting
;    52          0052  1 !         a bell bit.  PLL 20-May-1983
;    53          0053  1 ! 1-007 - If a LF is found, scroll according to the new dcb top & bottom of
;    54          0054  1 !         scrolling region fields.  PLL 26-May-1983
;    55          0055  1 ! 1-008 - If an ESC is detected, call the terminal simulator routine to
;    56          0056  1 !         interpret the sequence and perform the correct SMG$ function.
;    57          0057  1 !         PLL 7-Jul-1983
```

```
;    58          0058  1 ! 1-009 - Allow 2 'reserved' positions in upper half of table to pass thru
;    59          0059  1 !           as printable characters.  PLL 17-Aug-1983
;    60          0060  1 ! 1-010 - SMG$$SIM_TERM may set the graphics bit in the DCB's default
;    61          0061  1 !           attributes byte.  Take this into account when copying the attribute
;    62          0062  1 !           bytes for characters into the buffer.  PLL 29-Aug-1983
;    63          0063  1 ! 1-011 - Call SMG$$SIM_TERM when DCB_V_ALLOW_ESC is set.  PLL 2-Sept-1983
;    64          0064  1 ! 1-012 - In order to print carriage control characters instead of execute
;    65          0065  1 !           them, check the DCB_V_DISPLAY_CONTROLS bit and move the ascii rep
;    66          0066  1 !           into the text buffer in a different way.  PLL 23-Sep-1983
;001 PLL1013     0067  1 ! 1-013 - Remove dot on DCB argument in SMG$RING_BELL call.  PLL 10-Oct-1984
;    67          0068  1 !--
;    68          0069  1
```

```
;    70          0070  1 %SBTTL 'Declarations'
;    71          0071  1 !
;    72          0072  1 ! SWITCHES:
;    73          0073  1 !
;    74          0074  1
;    75          0075  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;    76          0076  1
;    77          0077  1 !
;    78          0078  1 ! LINKAGES:
;    79          0079  1 !
;    80          0080  1 !       NONE
;    81          0081  1 !
;    82          0082  1 ! TABLE OF CONTENTS:
;    83          0083  1 !
;    84          0084  1
;    85          0085  1 FORWARD ROUTINE
;    86          0086  1     SMG$$PUT_TEXT_TO_BUFFER;
;    87          0087  1
;    88          0088  1 !
;    89          0089  1 ! INCLUDE FILES:
;    90          0090  1 !
;    91          0091  1
;    92          0092  1 REQUIRE 'RTLIN:SMGPROLOG';              . defines Psects, macros, data base
;    93          0170  1
;    94          0171  1 !
;    95          0172  1 ! MACROS:
;    96          0173  1 !
;    97          0174  1 !       NONE
;    98          0175  1 !
;    99          0176  1 ! EQUATED SYMBOLS:
;   100          0177  1 !
;   101          0178  1 !       NONE
;   102          0179  1 !
;   103          0180  1 ! FIELDS:
;   104          0181  1 !
;   105          0182  1 !       NONE
;   106          0183  1 !
;   107          0184  1 ! PSECTS:
;   108          0185  1 !
;   109          0186  1
;   110          0187  1 !
;   111          0188  1 ! EXTERNAL REFERENCES:
;   112          0189  1 !
;   113          0190  1
;   114          0191  1 EXTERNAL ROUTINE
;   115          0192  1     SMG$$SIM_TERM,
;   116          0193  1     SMG$$SCROLL_AREA,
;   117          0194  1     SMG$RING_BELL;
;   118          0195  1
;   119          0196  1 EXTERNAL LITERAL
;   120          0197  1     SMG$_FATERRLIB,
;   121          0198  1     SMG$_STRTERESC;
;   122          0199  1
;   123          0200  1 ! Some constants needed by reference.
;   124          0201  1 OWN
;   125          0202  1     ALLONES     : BYTE INITIAL (-1);
;   126          0203  1
```

```
;  127          0204  1 ! The following macro is used to move a control character into the
;  128          0205  1 ! text buffer in such a way that output will later convert to the
;  129          0206  1 ! appropriate device dependent graphic character.
;  130          0207  1
;  131          0208  1 MACRO
;  132        M 0209  1         $INSERT_CTRL_CHAR (CHAR) =
;  133        M 0210  1         BEGIN
;  134        M 0211  1         LOCAL
;  135        M 0212  1             INDEX,
;  136        M 0213  1             REMAINING_COLS;
;  137        M 0214  1
;  138        M 0215  1         REMAINING_COLS = .DCB [DCB_W_NO_COLS] - .DCB [DCB_W_CURSOR_ROW];
;  139        M 0216  1         INDEX = $SMG$LINEAR (.DCB [DCB_W_CURSOR_ROW], .DCB [DCB_W_CURSOR_COL]);
;  140        M 0217  1
;  141        M 0218  1         IF 1 GTR .REMAINING_COLS
;  142        M 0219  1         THEN
;  143        M 0220  1             WORK_OVERFLOW = .BYTES_REMAINING
;  144        M 0221  1         ELSE
;  145        M 0222  1             BEGIN                   ! move the low nibble into the high nibble
;  146        M 0223  1             LOCAL
;  147        M 0224  1                 SHIFT_NIBBLE : BYTE,
;  148        M 0225  1                 WORK_ATTR;
;  149        M 0226  1             SHIFT_NIBBLE = (CHAR <0,4>) ^ 4;
;  150        M 0227  1             CH$MOVE (1, SHIFT_NIBBLE, TEXT_BUF [.INDEX]);
;  151        M 0228  1             WORK_ATTR = ATTR_M_USER_GRAPHIC OR .ATTR_CODE;
;  152        M 0229  1             CH$MOVE (1, WORK_ATTR, ATTR_BUF [.INDEX]);
;  153        M 0230  1             END;
;  154        M 0231  1
;  155        M 0232  1         DCB [DCB_W_CURSOR_COL] = .DCB [DCB_W_CURSOR_COL] + 1;
;  156        M 0233  1         IF .DCB [DCB_W_CURSOR_COL] EQL .DCB [DCB_W_NO_COLS]
;  157        M 0234  1         THEN
;  158        M 0235  1             DCB [DCB_W_CURSOR_COL] = .DCB [DCB_W_NO_COLS];
;  159          0236  1         END%;
;  160          0237  1
;  161          0238  1 !<BLF/PAGE>
```

```
;  163     0239  1 !+
;  164     0240  1 ! The table below (CHAR_TABLE) is used with a SCANC instruction to
;  165     0241  1 ! detect characters that have an impact on how text needs to be
;  166     0242  1 ! positioned in a text buffer that models what is on a portion of the
;  167     0243  1 ! screen.  Each character position is occupied by a code indicating
;  168     0244  1 ! the kind of action that this character has on text placement.
;  169     0245  1 ! Characters are grouped into 10 categories based on their impact on
;  170     0246  1 ! the terminal and hence on their impact on what should be placed in
;  171     0247  1 ! the buffer at what position.
;  172     0248  1 !
;  173     0249  1 ! These categories (codes) are:
;  174     0250  1 !
;  175     0251  1 !       Action Code     Action
;  176     0252  1 !       -----------     ------
;  177     0253  1 !            0          Normal processing.  Character occupies next
;  178     0254  1 !                       available slot in buffer.  Cursor column is
;  179     0255  1 !                       advanced by 1 after placement.
;  180     0256  1 !
;  181     0257  1 !            1          Character can be discarded.  Cursor is not
;  182     0258  1 !                       advanced.
;  183     0259  1 !
;  184     0260  1 !            2          Character can be discarded.  Cursor is not
;  185     0261  1 !                       modified, but a note must be made that the
;  186     0262  1 !                       bell needs to be sounded.
;  187     0263  1 !
;  188     0264  1 !            3          Character can be discarded, but cursor must be
;  189     0265  1 !                       backed up one column.  Be careful about cursor
;  190     0266  1 !                       already being in column 1.
;  191     0267  1 !
;  192     0268  1 !            4          Character can be discarded, but cursor must be
;  193     0269  1 !                       advanced to next TAB stop and intervening
;  194     0270  1 !                       character positions in the buffer are
;  195     0271  1 !                       undisturbed.
;  196     0272  1 !
;  197     0273  1 !                       TAB stops are assumed to be set in the following
;  198     0274  1 !                       columns with column numbering starting at 1:
;  199     0275  1 !                       9, 17, 25, 33, 41, 49, 57, 65, 73 ( width=80)
;  200     0276  1 !
;  201     0277  1 !                       9, 17, 25, 33, 41, 49, 57, 65, 73, 81, 89, 97,
;  202     0278  1 !                       105, 113, 121, 129 ( width=132)
;  203     0279  1 !
;  204     0280  1 !            5          Character can be discarded.  Cursor must be
;  205     0281  1 !                       advanced by one line.
;  206     0282  1 !
;  207     0283  1 !            6          Character can be discarded.  Cursor must be
;  208     0284  1 !                       advanced by one line.  (VT treated the same
;  209     0285  1 !                       as #5, FF.)
;  210     0286  1 !
;  211     0287  1 !            7          Character can be discarded.  Effect is
;  212     0288  1 !                       to clear the buffer and reset the cursor to
;  213     0289  1 !                       line 1 column 1.
;  214     0290  1 !
;  215     0291  1 !            8          Character can be discarded.  Effect is to set
;  216     0292  1 !                       cursor to column 1 of current line.
;  217     0293  1 !
;  218     0294  1 !            9          Character can be discarded.  For this version,
;  219     0295  1 !                       ESC terminates the string.  Eventually, subsequent
```

```
;  220     0296  1 ;                    characters need to be inspected to see if they
;  221     0297  1 ;                    constitute a recognized escape sequence whose
;  222     0298  1 ;                    effect must be simulated-- E.g., cursor setting,
;  223     0299  1 ;                    rendition setting.
;  224     0300  1 ;
;  225     0301  1 ;                    Some problems with this are:
;  226     0302  1 ;                        1. What to do about sequences that we don't
;  227     0303  1 ;                           recognize ?
;  228     0304  1 ;                        2. What to do about sequences that we
;  229     0305  1 ;                           recognize as ones that can cause
;  230     0306  1 ;                           confusion later is allowed to be
;  231     0307  1 ;                           sent to terminal -- E.g. select graphics
;  232     0308  1 ;                           rendition, etc ?
;  233     0309  1 ;
;  234     0310  1 ;            10      Character can be discarded.  Character is
;  235     0311  1 ;                    treated as a no-op.  It is broken out separately
;  236     0312  1 ;                    in case we ever need to do something special
;  237     0313  1 ;                    with it.
;  238     0314  1 ;
;  239     0315  1 ; In summary:
;  240     0316  1 ;
;  241     0317  1 ;   Hex Character Codes   ASCII Character          Action Code
;  242     0318  1 ;   ------------------   --------------          -----------
;  243     0319  1 ;       00 to 06         NUL to ACK                    1
;  244     0320  1 ;          07            BEL                           2
;  245     0321  1 ;          08            BS                            3
;  246     0322  1 ;          09            HT                            4
;  247     0323  1 ;          0A            LF                            5
;  248     0324  1 ;          0B            VT                            6
;  249     0325  1 ;          0C            FF                            7
;  250     0326  1 ;          0D            CR                            8
;  251     0327  1 ;       0E to 0F         SO to SI                      9
;  252     0328  1 ;       10 to 1A         DLE to SUB                    1
;  253     0329  1 ;          1B            ESC                           9
;  254     0330  1 ;       1C to 1F         FS to US                      1
;  255     0331  1 ;       20 to 7E         SP to ~                       0
;  256     0332  1 ;          7F            DEL                          10
;  257     0333  1 ;
;  258     0334  1 ;       80 to 9F         control chars                 1
;  259     0335  1 ;          A0            reserved                      1
;  260     0336  1 ;       A1 to FE         printing chars                0
;  261     0337  1 ;          FF            reserved                      1
;  262     0338  1 ;
```

```
;  264     0339  1 GLOBAL
;  265     0340  1     CHAR_TABLE : VECTOR [256, BYTE] INITIAL ( BYTE (
;  266     0341  1                                               ! 1st half is US ASCII
;  267     0342  1                                               ! for DEC Multinational set (default)
;  268     0343  1         1, 1, 1, 1, 1, 1, 1, 2, 3, 4, 5, 6, 7, 8, 9, 9,  ! 00 to 0F
;  269     0344  1         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 9, 1, 1, 1, 1,  ! 10 to 1F
;  270     0345  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! 20 to 2F
;  271     0346  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! 30 to 3F
;  272     0347  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! 40 to 4F
;  273     0348  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! 50 to 5F
;  274     0349  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! 60 to 6F
;  275     0350  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,10,  ! 70 to 7F
;  276     0351  1                                               ! 2nd half is DEC Supplemental Graphics
;  277     0352  1                                               ! for DEC Multinational set (default)
;  278     0353  1         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  ! 80 to 8F
;  279     0354  1         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  ! 90 to 9F
;  280     0355  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! A0 to AF
;  281     0356  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! B0 to BF
;  282     0357  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! C0 to CF
;  283     0358  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! D0 to DF
;  284     0359  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  ! E0 to EF
;  285     0360  1         0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0   ! F0 to FF
;  286     0361  1                                               ));
;  287     0362  1
;  288     0363  1
;  289     0364  1
;  290     0365  1
;  291     0366  1 !<BLF/PAGE>
```

```
;  293          0367  1 %SBTTL 'SMG$$PUT_TEXT_TO_BUFFER - Put text to buffer'
;  294          0368  1 GLOBAL ROUTINE SMG$$PUT_TEXT_TO_BUFFER (
;  295          0369  1                                       DCB : REF BLOCK [,BYTE],
;  296          0370  1                                       ATTR_CODE : BYTE,
;  297          0371  1                                       TEXT_LEN,
;  298          0372  1                                       TEXT_ADDR,
;  299          0373  1                                       CHAR_SET,
;  300          0374  1                                       OVERFLOW
;  301          0375  1                                           ) =
;  302          0376  1 !++
;  303          0377  1 ! FUNCTIONAL DESCRIPTION:
;  304          0378  1 !
;  305          0379  1 !       This procedure places a text string into a buffer given the
;  306          0380  1 !       current row and column in the buffer where output is to go.
;  307          0381  1 !       The input text string is scanned for special characters that
;  308          0382  1 !       prohibit simply moving the text into the buffer.  For example,
;  309          0383  1 !       TABs reposition the maintained cursor position and the text
;  310          0384  1 !       must be deposited at the appropriate tab boundaries as a
;  311          0385  1 !       function of current position in the line.  Escape sequences
;  312          0386  1 !       are not handled; an escape character is treated as a terminator,
;  313          0387  1 !       and a qualified success status will be returned to indicate
;  314          0388  1 !       that truncation occurred.
;  315          0389  1 !
;  316          0390  1 !       Positions in BUFFER that are modified have the corresponding
;  317          0391  1 !       positions in ATTR_BUFFER and CHAR_BUFFER set.
;  318          0392  1 !
;  319          0393  1 !
;  320          0394  1 !
;  321          0395  1 ! CALLING SEQUENCE:
;  322          0396  1 !
;  323          0397  1 !       ret_status.wlc.v = SMG$$PUT_TEXT_TO_BUFFER (
;  324          0398  1 !                                       DCB.mab.r,
;  325          0399  1 !                                       ATTR_CODE.rb.v,
;  326          0400  1 !                                       TEXT_LEN.rl.v,
;  327          0401  1 !                                       TEXT_ADDR.rl.v,
;  328          0402  1 !                                       CHAR_SET.rl.v
;  329          0403  1 !                                       [,OVERFLOW.wl.r])
;  330          0404  1 !
;  331          0405  1 ! FORMAL PARAMETERS:
;  332          0406  1 !
;  333          0407  1 !       DCB.mab.r       Address of virtual display control block.
;  334          0408  1 !                       Various fields from within in this block are
;  335          0409  1 !                       are interrogated and/or updated.
;  336          0410  1 !
;  337          0411  1 !       ATTR_CODE.rb.v  Video rendition attribute code.
;  338          0412  1 !                       Bit 0   Bold
;  339          0413  1 !                       Bit 1   Reverse video
;  340          0414  1 !                       Bit 2   Blinking
;  341          0415  1 !                       Bit 3   Underscored
;  342          0416  1 !
;  343          0417  1 !       TEXT_LEN.rl.v   Length of text string
;  344          0418  1 !
;  345          0419  1 !       TEXT_ADDR.rl.v  Address of text string
;  346          0420  1 !
;  347          0421  1 !       CHAR_SET.rl.v   Character set to use.
;  348          0422  1 !                               SMG$C_UNITED_KINGDOM
;  349          0423  1 !                               SMG$C_ASCII
```

```
;  350     0424  1 !                                   SMG$C_SPEC_GRAPHICS
;  351     0425  1 !                                   SMG$C_ALT_CHAR
;  352     0426  1 !                                   SMG$C_ALT_GRAPHICS
;  353     0427  1 !
;  354     0428  1 !       OVERFLOW.wl.r   Optional.  Address of longword in which
;  355     0429  1 !                       to return the number of characters that
;  356     0430  1 !                       did not fit on the line.
;  357     0431  1 !
;  358     0432  1 ! IMPLICIT INPUTS:
;  359     0433  1 !
;  360     0434  1 !       NONE
;  361     0435  1 !
;  362     0436  1 ! IMPLICIT OUTPUTS:
;  363     0437  1 !
;  364     0438  1 !       NONE
;  365     0439  1 !
;  366     0440  1 ! COMPLETION STATUS:
;  367     0441  1 !
;  368     0442  1 !       SS$_NORMAL      Normal successful completion
;  369     0443  1 !
;  370     0444  1 ! SIDE EFFECTS:
;  371     0445  1 !
;  372     0446  1 !       NONE
;  373     0447  1 !--
;  374     0448  1
;  375     0449  2     BEGIN
;  376     0450  2
;  377     0451  2     BUILTIN
;  378     0452  2         SCANC,
;  379     0453  2         NULLPARAMETER;
;  380     0454  2
;  381     0455  2     LOCAL
;  382     0456  2         TEXT_BUF : REF VECTOR [,BYTE],  ! Addr of text buffer
;  383     0457  2         ATTR_BUF : REF VECTOR [,BYTE],  ! Addr of attr buffer
;  384     0458  2         CHAR_BUF : REF VECTOR [,BYTE],  ! Addr of char set buffer
;  385     0459  2         STATUS,             ! status of subroutine calls
;  386     0460  2         WORK_OVERFLOW : INITIAL (0),    ! no. of overflow chars
;  387     0461  2         BYTES_REMAINING,! No. of bytes in input string yet to be
;  388     0462  2                         ! processed.
;  389     0463  2         IN_POINTER;     ! Current pointer into input string
;  390     0464  2
;  391     0465  2     LITERAL
;  392     0466  2         K_OVERFLOW_ARG = 6;
;  393     0467  2
;  394     0468  2     TEXT_BUF = .DCB [DCB_A_TEXT_BUF];
;  395     0469  2     ATTR_BUF = .DCB [DCB_A_ATTR_BUF];
;  396     0470  2     CHAR_BUF = .DCB [DCB_A_CHAR_SET_BUF];
;  397     0471  2
;  398     0472  2     BYTES_REMAINING = .TEXT_LEN;
;  399     0473  2     IN_POINTER = .TEXT_ADDR;
;  400     0474  2
;  401     0475  2     WHILE .BYTES_REMAINING NEQ 0
;  402     0476  2     DO
;  403     0477  3         BEGIN   ! Overall loop
;  404     0478  3         LOCAL
;  405     0479  3             CHARS_TO_MOVE,          ! No. of characters to move on this
;  406     0480  3                                     ! iteration
```

```
;  407     0481  3              PLACE_TO_MOVE,          ! Place to move from on this iteration
;  408     0482  3              NEW_BYTES_REMAINING,    ! No. of bytes remaining as returned
;  409     0483  3                                      ! by SCANC
;  410     0484  3              ADDR_DIFF;              ! Addr of char in input stream whose
;  411     0485  3                                      ! index into scanc table yields
;  412     0486  3                                      ! non-zero code.
;  413     0487  3          !+
;  414     0488  3          ! See if any of the remaining input characters require special
;  415     0489  3          ! treatment.
;  416     0490  3          !-
;  417     0491  3          SCANC ( BYTES_REMAINING,         ! No. of bytes remaining
;  418     0492  3                  .IN_POINTER,             ! Current pointer to source
;  419     0493  3                  CHAR_TABLE,              ! Address of SCANC table
;  420     0494  3                  ALLONES;                 ! Mask for ANDing
;  421     0495  3                  NEW_BYTES_REMAINING,     ! New remaining no. of bytes
;  422     0496  3                                           ! including the byte which
;  423     0497  3                                           ! caused the instruction to
;  424     0498  3                                           ! halt.  Is zero only if all
;  425     0499  3                                           ! bytes did not satify search.
;  426     0500  3                  ADDR_DIFF);              ! Addr of char in input stream
;  427     0501  3                                           ! whose index into scanc table
;  428     0502  3                                           ! yields non-zero code.
;  429     0503  3
;  430     0504  3          CHARS_TO_MOVE = .BYTES_REMAINING - .NEW_BYTES_REMAINING;
;  431     0505  3          PLACE_TO_MOVE = .IN_POINTER;
;  432     0506  3          IN_POINTER = .IN_POINTER + .CHARS_TO_MOVE;
;  433     0507  3          BYTES_REMAINING = .NEW_BYTES_REMAINING;
;  434     0508  3
;  435     0509  3          !+
;  436     0510  3          ! Copy the appropriate number of characters into the text buffer
;  437     0511  3          ! and the appropriate number of copies of the attribute code
;  438     0512  3          ! into the attribute buffer.
;  439     0513  3          !-
;  440     0514  3          IF .CHARS_TO_MOVE NEQ 0
;  441     0515  3          THEN
;  442     0516  4              BEGIN
;  443     0517  4              LOCAL
;  444     0518  4                  INDEX,  ! 0-based index into BUFFER and ATTR_BUFFER.
;  445     0519  4                  REMAINING_COLS,
;  446     0520  4
;  447     0521  4              INDEX = $SMG$LINEAR ( .DCB [DCB_W_CURSOR_ROW], .DCB [DCB_W_CURSOR_COL]);
;  448     0522  4
;  449     0523  4              REMAINING_COLS = .DCB [DCB_W_NO_COLS] - .DCB [DCB_W_CURSOR_COL] + 1;
;  450     0524  4              IF .CHARS_TO_MOVE GTR .REMAINING_COLS
;  451     0525  4              THEN                                  ! chars will overflow line
;  452     0526  5                  BEGIN
;  453     0527  5                  WORK_OVERFLOW = .BYTES_REMAINING +
;  454     0528  5                                  (.CHARS_TO_MOVE - .REMAINING_COLS);
;  455     0529  5                  CHARS_TO_MOVE = .REMAINING_COLS;
;  456     0530  4                  END;
;  457     0531  4
;  458     0532  4              !+
;  459     0533  4              ! Move text into buffer.
;  460     0534  4              !-
;  461     0535  4              CH$MOVE (.CHARS_TO_MOVE,              ! No. of chars
;  462     0536  4                       .PLACE_TO_MOVE,              ! From
;  463     0537  4                       TEXT_BUF [ .INDEX ] );       ! To
```

```
;  464     0538  4
;  465     0539  4                !+
;  466     0540  4                ! Rewrite attribute bytes.  Normally the attributes are
;  467     0541  4                ! passed to us, but for the 'autobended' case where escape
;  468     0542  4                ! sequences are used, we should look at the default attributes
;  469     0543  4                ! which may have been altered by SMG$$SIM_TERM.
;  470     0544  4                !-
;  471     0545  5                BEGIN
;  472     0546  5                LOCAL
;  473     0547  5                    WORK_ATTR;
;  474     0548  5                WORK_ATTR = .ATTR_CODE;
;  475     0549  5                IF .DCB [DCB_V_ALLOW_ESC]
;  476     0550  5                THEN
;  477     0551  5                    WORK_ATTR = .DCB [DCB_B_DEF_VIDEO_ATTR];
;  478     0552  5                CH$FILL (.WORK_ATTR,                    ! Char. to replicate
;  479     0553  5                         .CHARS_TO_MOVE,                ! No. of times
;  480     0554  5                         ATTR_BUF [ .INDEX ] );         ! Destination
;  481     0555  4                END;
;  482     0556  4
;  483     0557  4                !+
;  484     0558  4                ! Write the character set bytes, if necessary.
;  485     0559  4                !-
;  486     0560  4                IF .CHAR_BUF EQL 0 AND
;  487     0561  4                   .CHAR_SET NEQ SMG$C_ASCII
;  488     0562  4                THEN
;  489     0563  4                    0;      ! first char set - alloc buffer
;  490     0564  4
;  491     0565  4                IF .CHAR_BUF NEQ 0
;  492     0566  4                THEN
;  493     0567  4                    CH$FILL (.CHAR_SET,
;  494     0568  4                             .CHARS_TO_MOVE,
;  495     0569  4                             CHAR_BUF [.INDEX]);
;  496     0570  4
;  497     0571  4                !+
;  498     0572  4                ! Adjust resulting cursor position.  Check for overflow.
;  499     0573  4                !-
;  500     0574  4                DCB [DCB_W_CURSOR_COL] = .DCB [DCB_W_CURSOR_COL] +
;  501     0575  4                                         .CHARS_TO_MOVE;
;  502     0576  4                IF .DCB [DCB_W_CURSOR_COL] GTR .DCB [DCB_W_NO_COLS]
;  503     0577  4                THEN
;  504     0578  4                    DCB [DCB_W_CURSOR_COL] = .DCB [DCB_W_NO_COLS];
;  505     0579  4
;  506     0580  4                IF .WORK_OVERFLOW NEQ 0
;  507     0581  4                THEN
;  508     0582  4                    EXITLOOP;
;  509     0583  3                END;
;  510     0584  3
;  511     0585  3
;  512     0586  3            IF .NEW_BYTES_REMAINING EQL 0
;  513     0587  3            THEN
;  514     0588  3                EXITLOOP;           ! Break out of loop -- we're done
;  515     0589  3
;  516     0590  3            !+
;  517     0591  3            ! Dispatch on the non-zero code located to see what special
;  518     0592  3            ! action is needed.
;  519     0593  3            !-
;  520     0594  3            CASE .CHAR_TABLE [.(.ADDR_DIFF) <0,8>] FROM 1 TO 10 OF
```

```
;  521     0595  3          SET
;  522     0596  3              [1]:
;  523     0597  3              !+
;  524     0598  3              !    Hex Character Codes       ASCII Character
;  525     0599  3              !    ------------------        ---------------
;  526     0600  3              !    00 to 06                   NUL to ACK
;  527     0601  3              !    10 to 1A                   DLE to SUB
;  528     0602  3              !    1C to 1F                   FS to US
;  529     0603  3              !
;  530     0604  3              !  Character can be discarded.  Cursor is not advanced.
;  531     0605  3              !
;  532     0606  3              !  Special case if the user_graphic bit is set.  That indicates
;  533     0607  3              !  a device-independent code which should be placed in the buffer
;  534     0608  3              !  for later interpretation by output.  Notice that we are guaranteed
;  535     0609  3              !  that TEXT_ADDR contains only 1 character since only we call this
;  536     0610  3              !  routine.
;  537     0611  3              !-
;  538     0612  3              IF (.ATTR_CODE AND ATTR_M_USER_GRAPHIC) NEQ 0
;  539     0613  3              THEN
;  540     0614  3                  $INSERT_CTRL_CHAR (.TEXT_ADDR);
;  541     0615  3
;  542     0616  3              [2]:
;  543     0617  3              !+
;  544     0618  3              !    Hex Character Codes       ASCII Character
;  545     0619  3              !    ------------------        ---------------
;  546     0620  3              !       07                       BEL
;  547     0621  3              !
;  548     0622  3              !  Character can be discarded.  Cursor is not modified, and we
;  549     0623  3              !  call a routine to ring the bell now.  (Note that if we had
;  550     0624  3              !  stored the bell in the attribute buffer, the bell would've
;  551     0625  3              !  been rung every time the screen was repainted.)
;  552     0626  3              !-
;001 PLL1013     0627  3              SMG$RING_BELL (DCB [DCB_L_DID]);
;  554-1   0628  3
;  555     0629  3              [3]:
;  556     0630  3              !+
;  557     0631  3              !    Hex Character Codes       ASCII Character
;  558     0632  3              !    ------------------        ---------------
;  559     0633  3              !       08                       BS
;  560     0634  3              !
;  561     0635  3              !  Character can be discarded, but cursor must be backed up
;  562     0636  3              !  one column.  Be careful about cursor already being in
;  563     0637  3              !  column 1.
;  564     0638  3              !-
;  565     0639  4              BEGIN
;  566     0640  4              IF .DCB [DCB_W_CURSOR_COL] NEQ 1
;  567     0641  4              THEN
;  568     0642  4                  DCB [DCB_W_CURSOR_COL] = .DCB [DCB_W_CURSOR_COL] -1;
;  569     0643  4
;  570     0644  3              END;
;  571     0645  3
;  572     0646  3              [4]:
;  573     0647  3              !+
;  574     0648  3              !    Hex Character Codes       ASCII Character
;  575     0649  3              !    ------------------        ---------------
;  576     0650  3              !       09                       HT
;  577     0651  3              !
```

```
;  578     0652  3              ! Character can be discarded, but cursor must be advanced to
;  579     0653  3              ! next TAB stop and intervening character positions in the
;  580     0654  3              ! buffer must be left undisturbed.
;  581     0655  3              !
;  582     0656  3              ! TAB stops are assumed to be set in the following columns:
;  583     0657  3              ! 9, 17, 25, 33, 41, 49, 57, 65, 73 ( width=80)
;  584     0658  3              !
;  585     0659  3              ! 9, 17, 25, 33, 41, 49, 57, 65, 73, 81, 89, 97, 105, 113,
;  586     0660  3              ! 121, 129 ( width=132)
;  587     0661  3              !-
;  588     0662  4              BEGIN
;  589     0663  4              !+
;  590     0664  4              ! Be careful about tabbing off the end of the line or beyond
;  591     0665  4              ! the end of the virtual display line.
;  592     0666  4              !-
;  593     0667  4              IF NOT .DCB [DCB_V_DISPLAY_CONTROLS]
;  594     0668  4              THEN
;  595     0669  5                  BEGIN
;  596     0670  5                  DCB [DCB_W_CURSOR_COL] =
;  597     0671  5                                  (7.DCB [DCB_W_CURSOR_COL]-1)/8+1)*8+1;
;  598     0672  5                  IF .DCB [DCB_W_CURSOR_COL] GTR .DCB [DCB_W_NO_COLS]
;  599     0673  5                  THEN
;  600     0674  5                      DCB [DCB_W_CURSOR_COL] = .DCB [DCB_W_NO_COLS];
;  601     0675  5                  END
;  602     0676  4              ELSE
;  603     0677  4                  $INSERT_CTRL_CHAR (TAB);
;  604     0678  3              END;
;  605     0679  3
;  606     0680  3              [5,6]:
;  607     0681  3              !+
;  608     0682  3              !    Hex Character Codes       ASCII Character
;  609     0683  3              !    -------------------       ---------------
;  610     0684  3              !       0A                       LF
;  611     0685  3              !       0B                       VT
;  612     0686  3              !
;  613     0687  3              ! Character can be discarded.  Cursor must be advanced by
;  614     0688  3              ! one line.  Don't advance beyond last line of display.
;  615     0689  3              !-
;  616     0690  4              BEGIN
;  617     0691  4              !+
;  618     0692  4              ! If cursor not at bottom, advance DCB [DCB_W_CURSOR_ROW]
;  619     0693  4              ! by one.
;  620     0694  4              !-
;  621     0695  4              IF NOT .DCB [DCB_V_DISPLAY_CONTROLS]
;  622     0696  4              THEN
;  623     0697  5                  BEGIN
;  624     0698  5                  IF .DCB [DCB_W_CURSOR_ROW] + 1 LEQ .DCB [DCB_W_BOTTOM_OF_SCRREG]
;  625     0699  5                  THEN
;  626     0700  5                      DCB [DCB_W_CURSOR_ROW] = .DCB [DCB_W_CURSOR_ROW] + 1
;  627     0701  5                  ELSE
;  628     0702  5                      SMG$$SCROLL_AREA (.DCB,
;  629     0703  5                                        .DCB [DCB_W_TOP_OF_SCRREG],
;  630     0704  5                                        .DCB [DCB_W_COL_START],
;  631     0705  6                                        (.DCB [DCB_W_BOTTOM_OF_SCRREG] -
;  632     0706  5                                         .DCB [DCB_W_TOP_OF_SCRREG] + 1),
;  633     0707  5                                        .DCB [DCB_W_NO_COLS],
;  634     0708  5                                        SMG$M_UP,
```

```
;  635     0709  5                                    1);
;  636     0710  5                END
;  637     0711  4            ELSE
;  638     0712  5                BEGIN
;  639     0713  5                LOCAL
;  640     0714  5                    CHAR;
;  641     0715  5                CHAR = .(.ADDR_DIFF)<0,8>;
;  642     0716  5                $INSERT_CTRL_CHAR (.CHAR);
;  643     0717  4                END;
;  644     0718  3            END;
;  645     0719  3
;  646     0720  3            [7]:
;  647     0721  3            !+
;  648     0722  3            !   Hex Character Codes       ASCII Character
;  649     0723  3            !   ------------------        ---------------
;  650     0724  3            !      0C                       FF
;  651     0725  3            !
;  652     0726  3            ! Character can be discarded.  Effect is to clear the buffer
;  653     0727  3            ! and reset the cursor to line 1 column 1.
;  654     0728  3            !-
;  655     0729  4            BEGIN
;  656     0730  4            IF NOT .DCB [DCB_V_DISPLAY_CONTROLS]
;  657     0731  4            THEN
;  658     0732  5                BEGIN
;  659     0733  5                IF .DCB [DCB_W_CURSOR_ROW] + 1 LEQ .DCB [DCB_W_BOTTOM_OF_SCRREG]
;  660     0734  5                THEN
;  661     0735  5                    DCB [DCB_W_CURSOR_ROW] = .DCB [DCB_W_CURSOR_ROW] + 1
;  662     0736  5                ELSE
;  663     0737  5                    SMG$$SCROLL_AREA (.DCB,
;  664     0738  5                                      .DCB [DCB_W_TOP_OF_SCRREG],
;  665     0739  5                                      .DCB [DCB_W_COL_START],
;  666     0740  6                                      (.DCB [DCB_B_BOTTOM_OF_SCRREG] -
;  667     0741  5                                       .DCB [DCB_W_TOP_OF_SCRREG] + 1),
;  668     0742  5                                      .DCB [DCB_B_NO_COLS],
;  669     0743  5                                      SMG$M_UP,
;  670     0744  5                                      1);
;  671     0745  5                END
;  672     0746  4            ELSE
;  673     0747  4                $INSERT_CTRL_CHAR (FF);
;  674     0748  3            END;
;  675     0749  3
;  676     0750  3            [8]:
;  677     0751  3            !+
;  678     0752  3            !   Hex Character Codes       ASCII Character
;  679     0753  3            !   ------------------        ---------------
;  680     0754  3            !      0D                       CR
;  681     0755  3            !
;  682     0756  3            ! Character can be discarded.  Effect is to set cursor to
;  683     0757  3            ! column 1 of current line.
;  684     0758  3            !-
;  685     0759  4            BEGIN
;  686     0760  4            IF NOT .DCB [DCB_V_DISPLAY_CONTROLS]
;  687     0761  4            THEN
;  688     0762  4                DCB [DCB_W_CURSOR_COL] = 1
;  689     0763  4            ELSE
;  690     0764  4                $INSERT_CTRL_CHAR (CR);
;  691     0765  3            END;
```

```
;  692     0766  3
;  693     0767  3              [9]:
;  694     0768  3              !+
;  695     0769  3              !    Hex Character Codes       ASCII Character
;  696     0770  3              !    ------------------        ---------- --
;  697     0771  3              !       1B                      ESC
;  698     0772  3              !       0E                      SO
;  699     0773  3              !       0F                      SI
;  700     0774  3              !
;  701     0775  3              ! Character can be discarded.  Subsequent characters need
;  702     0776  3              ! to be inspected to see if they constitute a recognized
;  703     0777  3              ! escape sequence whose effect must be simulated-- E.g.,
;  704     0778  3              ! cursor setting, rendition setting.
;  705     0779  3              !
;  706     0780  3              ! SMG$$SIM_TERM processes the escape sequence, then returns
;  707     0781  3              ! here to allow any remaining characters to be processed.
;  708     0782  3              !-
;  709     0783  4              BEGIN
;  710     0784  4              IF NOT .DCB [DCB_V_ALLOW_ESC]
;  711     0785  4              THEN
;  712     0786  5                  RETURN (SMG$_STRTERESC) ! error from true SMG$
;  713     0787  4              ELSE
;  714     0788  5                  BEGIN                   ! autobended - attempt to interpret
;  715     0789  5                  LOCAL
;  716     0790  5                      LEN_OF_SEQUENCE,
;  717     0791  5                      STATUS;
;  718     0792  5                  STATUS = SMG$$SIM_TERM (.DCB,
;  719     0793  5                                          .BYTES_REMAINING,
;  720     0794  5                                          .IN_POINTER,  ! pass ptr to esc char
;  721     0795  5                                         LEN_OF_SEQUENCE);
;  722     0796  5                  IF NOT .STATUS THEN RETURN (.STATUS);
;  723     0797  5
;  724     0798  5                  !+
;  725     0799  5                  ! Update the number of bytes processed.  Since there is
;  726     0800  5                  ! an automatic update (by 1 character) at the end of this
;  727     0801  5                  ! loop, don't count the ESC now.
;  728     0802  5                  !-
;  729     0803  5                  BYTES_REMAINING = .BYTES_REMAINING - .LEN_OF_SEQUENCE + 1;
;  730     0804  5                  IN_POINTER = .IN_POINTER + .LEN_OF_SEQUENCE - 1;
;  731     0805  4                  END;                    ! autobended - attempt to interpret
;  732     0806  4
;  733     0807  3              END;
;  734     0808  3
;  735     0809  3              [10]:
;  736     0810  3              !+
;  737     0811  3              !    Hex Character Codes       ASCII Character
;  738     0812  3              !    ---- --------------       ---------------
;  739     0813  3              !       7F                      DEL
;  740     0814  3              !
;  741     0815  3              ! Character can be discarded.
;  742     0816  3              !-
;  743     0817  3              !   ! no special action
;  744     0818  3
;  745     0819  3              [INRANGE, OUTRANGE]:
;  746     0820  3              !+
;  747     0821  3              ! Should never get here -- there are no other codes in
;  748     0822  3              ! CHAR_TABLE.  If we do, we've got a problem.
```

```
;  749          0823  3              !-
;  750          0824  4              BEGIN
;  751          0825  4              RETURN SMG$_FATERRLIB;
;  752          0826  3              END;
;  753          0827  3          TES;
;  754          0828  3
;  755          0829  3
;  756          0830  3          !+
;  757          0831  3          ! Re-adjust pointer and count of bytes left to account for
;  758          0832  3          ! the special character(s) just processed.
;  759          0833  3          !-
;  760          0834  3          IN_POINTER = .IN_POINTER + 1;
;  761          0835  3          BYTES_REMAINING = .BYTES_REMAINING -1;
;  762          0836  2          END;     ! Overall loop
;  763          0837  2
;  764          0838  2      IF .DCB [DCB_W_CURSOR_COL] EQL .DCB [DCB_W_NO_COLS]
;  765          0839  2      THEN
;  766          0840  2          DCB [DCB_V_COL_80] = 1;
;  767          0841  2
;  768          0842  2      IF NOT NULLPARAMETER (K_OVERFLOW_ARG)
;  769          0843  2      THEN
;  770          0844  2          .OVERFLOW = .WORK_OVERFLOW;
;  771          0845  2                                  ! ret overflow chars if requested
;  772          0846  2      RETURN (SS$_NORMAL);
;  773          0847  1      END;                        ! End of routine SMG$$PUT_TEXT_TO_BUFFER


                                                .TITLE  SMG$$PUT_TEXT_TO_BUFFER Put text to display buf
;                                                       fer
                                                .IDENT  \1-013\
                                                .PSECT  _SMG$DATA,NOEXE,  PIC,2

                                        FF  00000 ALLONES:.BYTE   -1                                    ;
                                            00001         .BLKB   3
09  08  07  06  05  04  03  02  01  01  01  01  01  01  01  00004 CHAR_TABLE::
                                                .BYTE   1, 1, 1, 1, 1, 1, 1, 2, 3, 4, 5, 6, 7, 8, -  ;
01  01  09  01  01  01  01  01  01  01  01  01  01  01  09  00013           9, 9, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 9, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  01  01  00022           1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00031           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  00040           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  0004F           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  0005E           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  0006D           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
01  01  01  01  01  01  01  0A  00  00  00  00  00  00  00  0007C           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
01  01  01  01  01  01  01  01  01  01  01  01  01  01  01  0008B           0, 10, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -  ;
00  00  00  00  00  01  01  01  01  01  01  01  01  01  01  0009A           1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  000A9           1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  000B8           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  000C7           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  000D6           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  000E5           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
00  00  00  00  00  00  00  00  00  00  00  00  00  00  00  000F4           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
                                        00  00103           0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, -  ;
                                                            0, 0, 0, 0, 0                                ;

                                                .EXTRN  SMG$$SIM_TERM, SMG$$SCROLL_AREA
```

```
                                                        .EXTRN  SMG$RING_BELL, SMG$_FATERRLIB
                                                        .EXTRN  SMG$_STRTERESC

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

                                                        .ENTRY  SMG$$PUT_TEXT_TO_BUFFER, Save R2,R3,R4,R5,-      ; 0368
                                                                R6,R7,R8,R9,R10,R11
                                                        SUBL2   #32, SP
                                                        CLRL    WORK_OVERFLOW                                    ; 0449
                                                        MOVL    DCB, R9                                          ; 0468
                                                        MOVL    16(R9), TEXT_BUF
                                                        MOVL    20(R9), ATTR_BUF                                 ; 0469
                                                        MOVL    24(R9), CHAR_BUF                                 ; 0470
                                                        MOVQ    TEXT_LEN, BYTES_REMAINING                        ; 0472
                                                1$:     TSTL    BYTES_REMAINING                                  ; 0475
                                                        BNEQ    2$
                                                        BRW     41$
                                                2$:     SCANC   BYTES_REMAINING, (IN_POINTER), CHAR_TABLE, -    ; 0491
                                                                ALLONES
                                                        MOVL    R0, 16(SP)
                                                        MOVL    R1, 12(SP)
                                                        SUBL3   NEW_BYTES_REMAINING, BYTES_REMAINING, -          ; 0504
                                                                CHARS_TO_MOVE
                                                        MOVL    IN_POINTER, PLACE_TO_MOVE                        ; 0505
                                                        ADDL2   CHARS_TO_MOVE, IN_POINTER                        ; 0506
                                                        MOVL    NEW_BYTES_REMAINING, BYTES_REMAINING             ; 0507
                                                        TSTL    CHARS_TO_MOVE                                    ; 0514
                                                        BEQL    8$
                                                        MOVZWL  40(R9), R0                                       ; 0521
                                                        DECL    R0
                                                        MOVZWL  6(R9), R1
                                                        MULL2   R1, R0
                                                        MOVAB   42(R9), 8(SP)
                                                        MOVZWL  @8(SP), R1
                                                        MOVAB   -1(R1)[R0], INDEX
                                                        MOVZWL  6(R9), R0                                        ; 0523
                                                        SUBL2   R1, R0
                                                        INCL    REMAINING_COLS
                                                        CMPL    CHARS_TO_MOVE, REMAINING_COLS                    ; 0524
                                                        BLEQ    3$
                                                        SUBL3   REMAINING_COLS, CHARS_TO_MOVE, R1                ; 0528
                                                        ADDL3   BYTES_REMAINING, R1, WORK_OVERFLOW
                                                        MOVL    REMAINING_COLS, CHARS_TO_MOVE                    ; 0529
                                                3$:     MOVC3   CHARS_TO_MOVE, (PLACE_TO_MOVE), @INDEX-          ; 0537
                                                                [TEXT_BUF]
                                                        MOVZBL  ATTR_CODE, WORK_ATTR                             ; 0548
                                                        BBC     #5, 52(R9), 4$                                   ; 0549
                                                        MOVZBL  46(R9), WORK_ATTR                                ; 0551
                                                4$:     MOVC5   #0, (SP), WORK_ATTR, CHARS_TO_MOVE, @INDEX-      ; 0554
                                                                [ATTR_BUF]
                                                        TSTL    CHAR_BUF                                         ; 0560
                                                        BEQL    5$                                               ; 0565
                                                        ADDL3   CHAR_BUF, INDEX, (SP)                            ; 0569
                                                        MOVC5   #0, (SP), CHAR_SET, CHARS_TO_MOVE, @0(SP)
                                                5$:     ADDW2   CHARS_TO_MOVE, @8(SP)                            ; 0575
                                                        CMPW    @8(SP), 6(R9)                                    ; 0576
```

```
                                            05   1B 000B9        BLEQU   6$                              ;
                  08   BE          06       A9   B0 000BB        MOVW    6(R9), @8(SP)                   ; 0578
                                   18       AE   D5 000C0 6$:    TSTL    WORK_OVERFLOW                   ; 0580
                                            03   13 000C3        BEQL    8$                              ;
                                          022A   31 000C5 7$:    BRW     41$                             ;
                                   10       AE   D5 000C8 8$:    TSTL    NEW_BYTES_REMAINING             ; 0586
                                            F8   13 000CB        BEQL    7$                              ;
                       52          0C       BE   9A 000CD        MOVZBL  @ADDR_DIFF, R2                  ; 0594
              09       01 00000000'EF42     8F 000D1             CASEB   CHAR_TABLE[R2], #1, #9          ;
   006C     0061     0055          001C         000DA 9$:        .WORD   10$-9$,-                        ;
   0186     0118     00DA          00DA         000E2                    12$-9$,-                        ;
                     0211          01E0         000EA                    13$-9$,-                        ;
                                                                         15$-9$,-                        ;
                                                                         21$-9$,-                        ;
                                                                         21$-9$,-                        ;
                                                                         23$-9$,-                        ;
                                                                         31$-9$,-                        ;
                                                                         38$-9$,-                        ;
                                                                         40$-9$                          ;
                       50 00000000G       8F   D0 000EE          MOVL    #SMG$_FATERRLIB, R0             ; 0825
                                                04 000F5         RET                                     ;
              49       08   AC             06   E1 000F6 10$:    BBC     #6, ATTR_CODE, 14$              ; 0612
                            53    06       A9   3C 000FB         MOVZWL  6(R9), REMAINING_COLS           ; 0614
                            50    28       A9   3C 000FF         MOVZWL  40(R9), R0                      ;
                            53             50   C2 00103         SUBL2   R0, REMAINING_COLS              ;
                            50    28       A9   3C 00106         MOVZWL  40(R9), R0                      ;
                                           50   D7 0010A         DECL    R0                              ;
                            51    06       A9   3C 0010C         MOVZWL  6(R9), R1                       ;
                            50             51   C4 00110         MULL2   R1, R0                          ;
                            52    2A       A9   9E 00113         MOVAB   42(R9), R2                      ;
                            51             62   3C 00117         MOVZWL  (R2), R1                        ;
                            51    FF     A140   9E 0011A         MOVAB   -1(R1)[R0], INDEX               ;
                                           53   D5 0011F         TSTL    REMAINING_CO:                   ;
                                           03   14 00121         BGTR    11$                             ;
                                         016C   31 00123         BRW     34$                             ;
  53     10   AC            04             00   EF 00126 11$:    EXTZV   #0, #4, TEXT_ADDR, R3           ;
                                         00BA   31 0012C         BRW     22$                             ;
                                  38       A9   9F 0012F 12$:    PUSHAB  56(R9)                          ; 0627
             00000000G      00             01   FB 00132         CALLS   #1, SMG$RING_BELL               ;
                                           77   11 00139         BRB     20$                             ;
                            01    2A       A9   B1 0013B 13$:    CMPW    42(R9), #1                      ; 0640
                                           71   13 0013F         BEQL    20$                             ;
                                  2A       A9   B7 00141         DECW    42(R9)                          ; 0642
                                           6C   11 00144 14$:    BRB     20$                             ; 0594
                            53    2A       A9   9E 00146 15$:    MOVAB   42(R9), R3                      ; 0670
              18       2F   A9             02   E0 0014A         BBS     #2, 47(R9), 16$                 ; 0667
                            50             63   3C 0014F         MOVZWL  (R3), R0                        ; 0671
                                           50   D7 00152         DECL    R0                              ;
                            50             08   C6 00154         DIVL2   #8, R0                          ;
              51            50             03   78 00157         ASHL    #3, R0, R1                      ;
              63            51             09   A1 0015B         ADDW3   #9, R1, (R3)                    ;
                       06   A9             63   B1 0015F         CMPW    (R3), 6(R9)                     ; 0672
                                           4D   1B 00163         BLEQU   20$                             ;
                                           47   11 00165         BRB     19$                             ; 0674
                            52    06       A9   3C 00167 16$:    MOVZWL  6(R9), REMAINING_COLS           ; 0677
                            50    28       A9   3C 0016B         MOVZWL  40(R9), R0                      ;
                            52             50   C2 0016F         SUBL2   R0, REMAINING_COLS              ;
```

```
                                50      28   A9  3C 00172          MOVZWL  40(R9), R0                                      ;
                                50          D7 00176              DECL    R0                                              ;
                                51      06   A9  3C 00178          MOVZWL  6(R9), R1                                       ;
                                50          51  C4 0017C          MULL2   R1, R0                                          ;
                                51          63  3C 0017F          MOVZWL  (R3), R1                                        ;
                                51      FF A140 9E 00182          MOVAB   -1(R1)[R0], INDEX                               ;
                                            52  D5 00187          TSTL    REMAINING_COLS                                  ;
                                            06  14 00189          BGTR    17$                                             ;
                            18  AE          57  D0 0018B          MOVL    BYTES_REMAINING, WORK_OVERFLOW                  ;
                                            15  11 0018F          BRB     18$                                             ;
                                50      90   8F  90 00191 17$:     MOVB    #-112, SHIFT_NIBBLE                             ;
                              614B          50  90 00195          MOVB    SHIFT_NIBBLE, (INDEX)[TEXT_BUF]                 ;
                                50      08   AC  9A 00199          MOVZBL  ATTR_CODE, WORK_ATTR                            ;
        50          01              06      01  F0 0019D          INSV    #1, #6, #1, WORK_ATTR                           ;
                              614A          50  90 001A2          MOVB    WORK_ATTR, (INDEX)[ATTR_BUF]                    ;
                                            63  B6 001A6 18$:     INCW    (R3)                                            ;
                            06  A9          63  B1 001A8          CMPW    (R3), 6(R9)                                     ;
                                            5A  12 001AC          BNEQ    25$                                             ;
                                63      06   A9  B0 001AE 19$:     MOVW    6(R9), (R3)                                     ;
                                            7D  11 001B2 20$:     BRB     27$                                             ; 0594
                    3E      2F  A9          02  E1 001B4 21$:     BBC     #2, 47(R9), 24$                                 ; 0695
                                54          52  D0 001B9          MOVL    R2, CHAR                                        ; 0715
                                53      06   A9  3C 001BC          MOVZWL  6(R9), REMAINING_COLS                           ; 0716
                                50      28   A9  3C 001C0          MOVZWL  40(R9), R0                                      ;
                                53          50  C2 001C4          SUBL2   R0, REMAINING_COLS                              ;
                                50      28   A9  3C 001C7          MOVZWL  40(R9), R0                                      ;
                                            50  D7 001CB          DECL    R0                                              ;
                                51      06   A9  3C 001CD          MOVZWL  6(R9), R1                                       ;
                                50          51  C4 001D1          MULL2   R1, R0                                          ;
                                52      2A   A9  9E 001D4          MOVAB   42(R9), R2                                      ;
                                51          62  3C 001D8          MOVZWL  (R2), R1                                        ;
                                51      FF A140 9E 001DB          MOVAB   -1(R1)[R0], INDEX                               ;
                                            53  D5 001E0          TSTL    REMAINING_COLS                                  ;
                                            75  15 001E2          BLEQ    29$                                             ;
        53          54              04          00  EF 001E4          EXTZV   #0, #4, CHAR, R3                                ;
                    53              53          04  78 001E9 22$:     ASHL    #4, R3, R3                                      ;
                                50              53  90 001ED          MOVB    R3, SHIFT_NIBBLE                                ;
                                                6C  11 001F0          BRB     30$                                             ;
                    3C      2F  A9              02  E0 001F2 23$:     BBS     #2, 47(R9), 28$                                 ; 0730
                                50      28   A9  3C 001F7 24$:     MOVZWL  40(R9), R0                                      ; 0733
                                            50  D6 001FB          INCL    R0                                              ;
        50      4A  A9              10          00  ED 001FD          CMPZV   #0, #16, 74(R9), R0                             ;
                                            05  19 00203          BLSS    26$                                             ;
                                        28   A9  B6 00205          INCW    40(R9)                                          ; 0735
                                            62  11 00208 25$:     BRB     32$                                             ;
                                            01  DD 0020A 26$:     PUSHL   #1                                              ; 0737
                                            01  DD 0020C          PUSHL   #1                                              ;
                                7E      06   A9  3C 0020E          MOVZWL  6(R9), -(SP)                                    ; 0742
                                50      4A   A9  3C 00212          MOVZWL  74(R9), R0                                      ; 0741
                                51      48   A9  3C 00216          MOVZWL  72(R9), R1                                      ;
                                50          51  C2 0021A          SUBL2   R1, R0                                          ;
                                        01   A0  9F 0021D          PUSHAB  1(R0)                                           ; 0740
                                7E      04   A9  3C 00220          MOVZWL  4(R9), -(SP)                                    ; 0739
                                7E      48   A9  3C 00224          MOVZWL  72(R9), -(SP)                                   ; 0738
                                            59  DD 00228          PUSHL   R9                                              ; 0737
                    00000000G   00          07  FB 0022A          CALLS   #7, SMG$$SCROLL_AREA                            ;
                                            39  11 00231 27$:     BRB     32$                                             ;  50
```

```
                                                53       06   A9   3C 00233 28$:    MOVZWL  6(R9), REMAINING_COLS                ; 0747
                                                50       28   A9   3C 00237        MOVZWL  40(R9), R0
                                                53            50   C2 0023B        SUBL2   R0, REMAINING_COLS
                                                50       28   A9   3C 0023E        MOVZWL  40(R9), R0
                                                             50   D7 00242        DECL    R0
                                                51       06   A9   3C 00244        MOVZWL  6(R9), R1
                                                50            51   C4 00248        MULL2   R1, R0
                                                52       2A   A9   9E 0024B        MOVAB   42(R9), R2
                                                51            62   3C 0024F        MOVZWL  (R2), R1
                                                51       FF A140   9E 00252        MOVAB   -1(R1)[R0], INDEX
                                                             53   D5 00257        TSTL    REMAINING_COLS
                                                             37   15 00259 29$:    BLEQ    34$
                                                50                3F 92 0025B    MCOMB   #63, SHIFT_NIBBLE
                                                             3B   11 0025E 30$:    BRB     36$
                                                52       2A   A9   9E 00260 31$:    MOVAB   42(R9), R2                          ; 0762
                                          05    2F   A9            02   E0 00264    BBS     #2, 47(R9), 33$                     ; 0760
                                                62                 01   B0 00269    MOVW    #1, (R2)                            ; 0762
                                                             7D   11 0026C 32$:    BRB     40$
                                                53       06   A9   3C 0026E 33$:    MOVZWL  6(R9), REMAINING_COLS               ; 0764
                                                50       28   A9   3C 00272        MOVZWL  40(R9), R0
                                                53            50   C2 00276        SUBL2   R0, REMAINING_COLS
                                                50       28   A9   3C 00279        MOVZWL  40(R9), R0
                                                             50   D7 0027D        DECL    R0
                                                51       06   A9   3C 0027F        MOVZWL  6(R9), R1
                                                50            51   C4 00283        MULL2   R1, R0
                                                51            62   3C 00286        MOVZWL  (R2), R1
                                                51       FF A140   9E 00289        MOVAB   -1(R1)[R0], INDEX
                                                             53   D5 0028E        TSTL    REMAINING_COLS
                                                             06   14 00290        BGTR    35$
                                             18 AE                57   D0 00292 34$:    MOVL    BYTES_REMAINING, WORK_OVERFLOW
                                                             14   11 00296        BRB     37$
                                                50                30 8E 00298 35$:    MNEGB   #48, SHIFT_NIBBLE
                                                614B              50   90 0029B 36$:    MOVB    SHIFT_NIBBLE, (INDEX)[TEXT_BUF]
                                                50       08   AC   9A 0029F        MOVZBL  ATTR_CODE, WORK_ATTR
                                          50    01             06   01   F0 002A3    INSV    #1, #6, #1, WORK_ATTR
                                                614A              50   90 002A8    MOVB    WORK_ATTR, (INDEX)[ATTR_BUF]
                                                             62   B6 002AC 37$:    INCW    (R2)
                                                06   A9           62   B1 002AE        CMPW    (R2), 6(R9)
                                                             37   12 002B2        BNEQ    40$
                                                62       06   A9   B0 002B4        MOVW    6(R9), (R2)
                                                             31   11 002B8        BRB     40$                                 ; 0594
                                          08    34   A9            05   E0 002BA 38$:    BBS     #5, 52(R9), 39$             ; 0784
                                                50 00000000G      8F   D0 002BF        MOVL    #SMG$_STRTERESC, R0                 ; 0786
                                                                  04   002C6        RET
                                                         1C   AE   9F 002C7 39$:    PUSHAB  LEN_OF_SEQUENCE                     ; 0792
                                                7E               57   7D 002CA        MOVQ    BYTES_REMAINING, -(SP)              ; 0793
                                                                 59   DD 002CD        PUSHL   R9                                  ; 0792
                                             00000000G 00        04   FB 002CF        CALLS   #4, SMG$$SIM_TERM
                                                36               50   E9 002D6        BLBC    STATUS, 44$                         ; 0796
                                          50    57               1C   AE   C3 002D9    SUBL3   LEN_OF_SEQUENCE, BYTES_REMAINING, R0 ; 0803
                                                57               01   A0   9E 002DE    MOVAB   1(R0), BYTES_REMAINING
                                          50    58               1C   AE   C1 002E2    ADDL3   LEN_OF_SEQUENCE, IN_POINTER, R0     ; 0804
                                                58               FF   A0   9E 002E7    MOVAB   -1(R0), IN_POINTER
                                                                 58   D6 002EB 40$:    INCL    IN_POINTER                          ; 0834
                                                                 57   D7 002ED        DECL    BYTES_REMAINING                     ; 0835
                                                               FD2B   31   002EF        BRW     1$                                  ; 0475
                                                06   A9           2A   A9   B1 002F2 41$:    CMPW    42(R9), 6(R9)               ; 0838
```

```
                                      04  12 002F7           BNEQ    42$                                        ;
                     34    A9         02  88 002F9           BISB2   #2, 52(R9)                                 ; 0840
                           06         6C  91 002FD 42$:      CMPB    (AP), #6                                   ; 0842
                                      0A  1F 00300           BLSSU   43$                                        ;
                                18    AC  D5 00302           TSTL    24(AP)                                     ;
                                      05  13 00305           BEQL    43$                                        ;
                     18    BC   18    AE  D0 00307           MOVL    WORK_OVERFLOW, @OVERFLOW                   ; 0844
                           50         01  D0 0030C 43$:      MOVL    #1, R0                                     ; 0846
                                          04 0030F 44$:      RET                                                ; 0847

; Routine Size:  784 bytes,    Routine Base:  _SMG$CODE + 0000

;  774          0848  1 !<BLF/PAGE>
```

```
;  776          0849  1 END                        ! End of module SMG$$PUT_TEXT_TO_BUFFER
;  777          0850  1
;  778          0851  0 ELUDOM
```

```
;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _SMG$DATA                      260  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _SMG$CODE                      784  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
```

```
;                              Library Statistics
;                                        --------  Symbols  --------      Pages       Processing
;       File                             Total     Loaded   Percent       Mapped      Time
;
;  _$255$DUA18:[SYSLIB]STARLET.L32;1       9776          5         0          581        00:01.1
;  _$255$DUA18:[SMGRTL.OBJ]RTLLIB.L32;1      36          0         0            8        00:00.1
;  _$255$DUA18:[SMGRTL.OBJ]SMGLIB.L32;1     469         19         4           38        00:00.4
```

```
;                                  COMMAND QUALIFIERS
;
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:SMGPUTTEX/OBJ=OBJ$:SMGPUTTEX MSRC$:SMGPUTTEX/UPDATE=(BUG$:SMGPUTTEX
;       )

; Size:          784 code + 260 data bytes
; Run Time:        00:24.2
; Elapsed Time:    00:28.9
; Lines/CPU Min:    2110
; Lexemes/CPU-Min: 17915
; Memory Used:  354 pages
; Compilation Complete
```

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

SYSLOA

SMGMINUPD
LIS

CSP
MAP

SMGPUTTEX
LIS

SYSLOAWS1
MAP