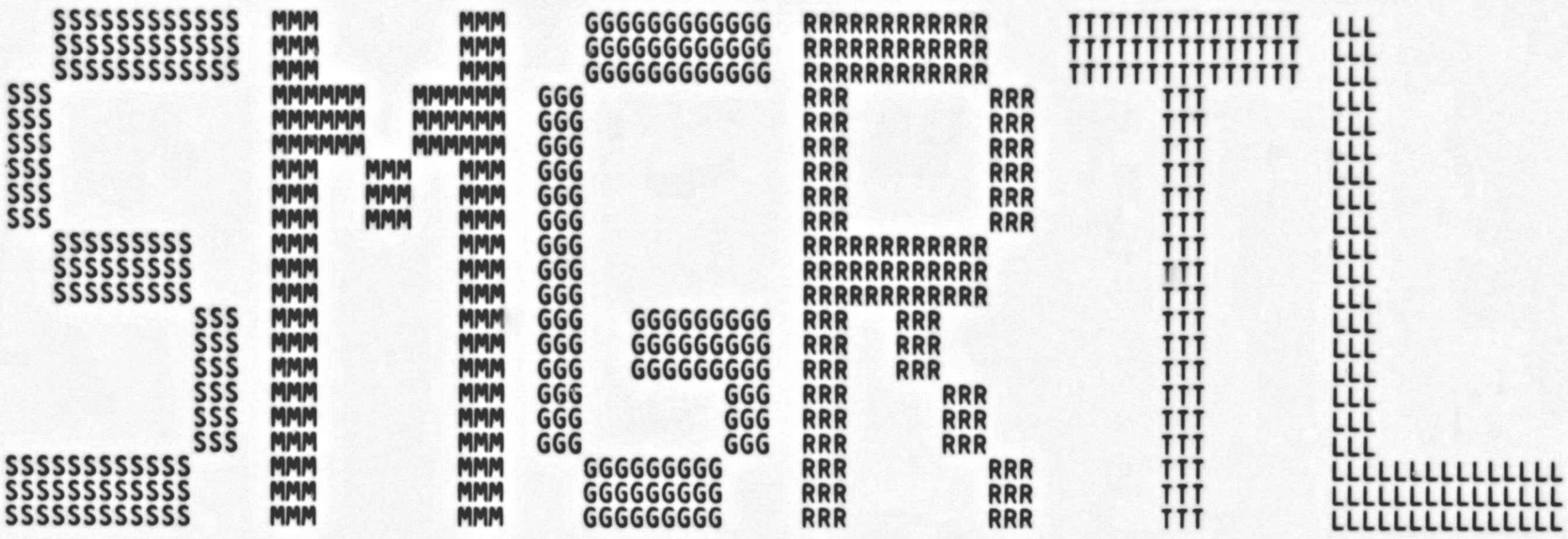

_$2

Val

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

```
  SSSSSSSS  MM      MM   GGGGGGGG   SSSSSSSS  TTTTTTTTTT    AAAAAA    TTTTTTTTTT    AAAAAA    BBBBBBBB
  SSSSSSSS  MM      MM   GGGGGGGG   SSSSSSSS  TTTTTTTTTT    AAAAAA    TTTTTTTTTT    AAAAAA    BBBBBBBB
SS          MMMM  MMMM  GG         SS            TT       AA    AA      TT      AA    AA  BB      BB
SS          MMMM  MMMM  GG         SS            TT       AA    AA      TT      AA    AA  BB      BB
SS          MM  MM  MM  GG         SS            TT       AA    AA      TT      AA    AA  BB      BB
SS          MM  MM  MM  GG         SS            TT       AA    AA      TT      AA    AA  BB      BB
  SSSSSS    MM      MM  GG           SSSSSS      TT       AA    AA      TT      AA    AA  BBBBBBBB
  SSSSSS    MM      MM  GG           SSSSSS      TT       AA    AA      TT      AA    AA  BBBBBBBB
        SS  MM      MM  GG   GGGGG         SS    TT       AAAAAAAAAA    TT      AAAAAAAAAA  BB      BB
        SS  MM      MM  GG   GGGGG         SS    TT       AAAAAAAAAA    TT      AAAAAAAAAA  BB      BB
        SS  MM      MM  GG       GG        SS    TT       AA    AA      TT      AA    AA  BB      BB
        SS  MM      MM  GG       GG        SS    TT       AA    AA      TT      AA    AA  BB      BB
SSSSSSSS    MM      MM    GGGGGG    SSSSSSSS     TT       AA    AA      TT      AA    AA  BBBBBBBB
SSSSSSSS    MM      MM    GGGGGG    SSSSSSSS     TT       AA    AA      TT      AA    AA  BBBBBBBB

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 MODULE SMG$STATEMENT_TABLES( %TITLE 'TPARSE tables for TERMTABLE statements'
;     2          0002  0                      IDENT = '1-004'          ! File: SMGSTATAB.B32 Edit: PLL1004
;     3          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 module contains the LIB$TPARSE state tables used to parse
;    36          0036  1 !       TERMTABLE statements.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT:  User mode - AST reentrant
;    39          0039  1 !
;    40          0040  1 ! AUTHOR: P. Levesque CREATION DATE: 29-Jan-1984
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 ! 1-001 - Original.  PLL 29-Jan-1983
;    45          0045  1 ! 1-002 - Allow logical name in REQUIRE file spec.  PLL 21-Mar-1984
;    46          0046  1 ! 1-003 - Tighten up some error checking in the state tables.  PLL 11-Jul-1984
;    47          0047  1 ! 1-004 - More error checking.  PLL 11-Jul-1984
;    48          0048  1 !--
;    49          0049  1
```

```
;    51          0050  1 %SBTTL 'Declarations'
;    52          0051  1 !
;    53          0052  1 ! SWITCHES:
;    54          0053  1 !
;    55          0054  1
;    56          0055  1 !
;    57          0056  1 ! LINKAGES:
;    58          0057  1 !
;    59          0058  1 !       NONE
;    60          0059  1 !
;    61          0060  1 ! TABLE OF CONTENTS:
;    62          0061  1 !
;    63          0062  1
;    64          0063  1 FORWARD ROUTINE
;    65          0064  1         MISSING_FILE_SPEC,              ! signal error
;    66          0065  1         MISSING_NAME_REQUIRE,           ! signal error
;    67          0066  1         PARSE_BOOLEANS,                 ! compile boolean capabilities
;    68          0067  1         PARSE_NUMERICS,                 ! compile numeric capabilities
;    69          0068  1         PARSE_REQUIRE,                  ! compile caps in REQUIRE file
;    70          0069  1         PARSE_STRINGS,                  ! compile string capabilities
;    71          0070  1         SEARCH_KNOWN_TYPES,             ! search table of known terminals
;    72          0071  1         START_NEW_TERM_DEF,             ! new term, init def fields
;    73          0072  1         UNRECOGNIZED_STATEMENT,         ! signal error
;    74          0073  1         ZERO_CUR_TERM_DEF,              ! mark no current terminal definition
;    75          0074  1
;    76          0075  1     ! The following are used by all the SMG TPARSE tables.
;    77          0076  1
;    78          0077  1         SMG$$BLANKS_OFF,                ! set bit to skip over blanks
;    79          0078  1         SMG$$BLANKS_ON,                 ! set bit to find blanks
;    80          0079  1         SMG$$COPY_CAP,                  ! copy string cap into TERMTABLE
;    81          0080  1         SMG$$FLUSH_ARITHMETIC,          ! flush arithmetic data to TERMTABLE
;    82          0081  1         SMG$$FLUSH_NUMERIC,             ! flush number to data area
;    83          0082  1         SMG$$FLUSH_SAVED_BUFFER,        ! flush saved token to data area
;    84          0083  1         SMG$$NEXT_RECORD,               ! get next record from ascii TERMTABLE
;    85          0084  1         SMG$$SAVE_TOKEN_STRING,         ! save pointer & count of chars
;    86          0085  1                                         ! in current cap string
;    87          0086  1         SMG$$STORE_CAP_MASK,            ! remember cap number
;    88          0087  1         SMG$$SYNTAX_ERROR,              ! signal syntax error
;    89          0088  1         SMG$$WRITE_DATA;                ! block I/O write data to file
;    90          0089  1
;    91          0090  1 !
;    92          0091  1 ! INCLUDE FILES:
;    93          0092  1 !
;    94          0093  1
;    95          0094  1 REQUIRE 'RTLIN:SMGPROLOG';              ! Defines psects, macros, etc.
;    96          0172  1
;    97          0173  1
;    98          0174  1 LIBRARY 'RTLML:SMGTPALIB';              ! Bliss library of definitions &
;    99          0175  1                                         ! macros used for TERMTABLE
;   100          0176  1
;   101          0177  1 LIBRARY 'RTLTPAMAC';                    ! TPARSE library of macros
;   102          0178  1
;   103          0179  1 !
;   104          0180  1 ! EQUATED SYMBOLS:
;   105          0181  1 !
;   106          0182  1 LITERAL
;   107          0183  1         SINGLE_QUOTE = %X'27',
```

```
;  108     0184  1          DOUBLE_QUOTE = %X'22',
;  109     0185  1          FORCED_END = %X'00';
;  110     0186  1
;  111     0187  1
;  112     0188  1 !
;  113     0189  1 ! FIELDS:
;  114     0190  1 !
;  115     0191  1 !       NONE
;  116     0192  1 !
;  117     0193  1 ! PSECTS:
;  118     0194  1 !
;  119     0195  1
;  120     0196  1 !
;  121     0197  1 ! EXTERNAL REFERENCES:
;  122     0198  1 !
;  123     0199  1 EXTERNAL ROUTINE
;  124     0200  1          LIB$TPARSE,                    ! parser
;  125     0201  1          STR$UPCASE,                    ! convert to uppercase chars
;  126     0202  1          STR$COPY_DX;                   ! copy src string to dest string
;  127     0203  1
;  128     0204  1 EXTERNAL
;  129     0205  1          SMG$_ERRLIN,                   ! error at line n
;  130     0206  1          SMG$_ERRAT_LIN,                ! error in line n at or near 'x'
;  131     0207  1          SMG$_FAIOPEFIL,                ! failed to open file x
;  132     0208  1          SMG$_MISTERNAM,                ! missing terminal name
;  133     0209  1          SMG$_SYNERR,                   ! syntax error
;  134     0210  1          SMG$_UNRECSTA,                 ! unrecognized statement
;  135     0211  1          SMG$_MISFILSPE,                ! missing file spec in REQUIRE stmt
;  136     0212  1          SMG$_MISENDSTA,                ! missing END statement
;  137     0213  1          SMG$_MISNAMREQ;                ! expected NAME or REQUIRE
;  138     0214  1
;  139     0215  1 !+
;  140     0216  1 ! The following is the table of DEC terminals used by SET/SHOW TERM.
;  141     0217  1 !-
;  142     0218  1 EXTERNAL
;  143     0219  1          TERM$_TABLE : BLOCKVECTOR [,28,BYTE], !table of known terminals
;  144     0220  1          TERM$_NAME : VECTOR,           ! table of their names
;  145     0221  1          TERM$_NUM;                     ! number of known terminals
;  146     0222  1
;  147     0223  1 !+
;  148     0224  1 ! The following are the keyword and state tables needed by LIB$TPARSE.
;  149     0225  1 !-
;  150     0226  1 EXTERNAL
;  151     0227  1          SMG$$A_BOOLEAN_STATES,
;  152     0228  1          SMG$$A_BOOLEAN_KEYWDS,
;  153     0229  1          SMG$$A_NUMERIC_STATES,
;  154     0230  1          SMG$$A_NUMERIC_KEYWDS,
;  155     0231  1          SMG$$A_STRING_STATES,
;  156     0232  1          SMG$$A_STRING_KEYWDS;
;  157     0233  1 !
;  158     0234  1 ! OWN STORAGE:
;  159     0235  1 !
;  160     0236  1
;  161     0237  1 GLOBAL
;  162     0238  1          SMG$$MASK_ADR : INITIAL (0),           ! used by TPARSE action routines
;  163     0239  1          SMG$$NEXT_NEGATIVE_NUMBER : INITIAL (-2),
;  164     0240  1                                         ! used to assign terminal numbers
```

```
;  165     0241  1                                          ! (start with -2 cause SET TERM
;  166     0242  1                                          !  views -1 as failed to find name)
;  167     0243  1          SMG$$CURRENT_LINE : INITIAL (0),! current input line - maintained
;  168     0244  1                                          ! for error messages
;  169     0245  1          SMG$$CURRENT_BLOCK : INITIAL (2 + SMG$K_CAP_PTRS_SIZE/512),
;  170     0246  1                                          ! current virtual block in TERMTABLE.EXE
;  171     0247  1                                          ! (data skips over ptrs)
;  172     0248  1          SMG$$CURRENT_DEF_BLOCK : INITIAL (2),
;  173     0249  1                                          ! start of terminal definition
;  174     0250  1                                          ! (0 is reserved for a header)
;  175     0251  1          SMG$$DATA_OFFSET : INITIAL (SMG$K_CAP_PTRS_SIZE);
;  176     0252  1                                          ! starting offset for terminal data
;  177     0253  1  !
;  178     0254  1  ! MACROS:
;  179     0255  1  !
;  180     0256  1  !       NONE
;  181     0257  1
```

```
%SBTTL 'SMG$$STATEMENT_TABLES - TPARSE tables for TERMTABLE statements'
!++
! FUNCTIONAL DESCRIPTION:
!
!       The following are the state tables used to parse an ascii
!       TERMTABLE.TXT (a file which defines terminal characteristics).
!       This set of state tables looks for valid statements and then
!       calls LIB$TPARSE with the state tables for those statements.
!--

$INIT_STATE (SMG$$A_STMT_STATES, SMG$$A_STMT_KEYWDS);
                                        ! set up state tables, key words

!+
! Begin scanning loop.  Look for the start of a statement.
! Skip over blanks and comments.
!-

$STATE (BEGIN_SCAN,
    ((END_OF_LINE), BEGIN_SCAN, SMG$$NEXT_RECORD),
    ('!', BEGIN_SCAN, SMG$$NEXT_RECORD),
    ((STATEMENT), BEGIN_SCAN, SMG$$BLANKS_OFF),
    (TPA$_LAMBDA, TPA$_EXIT)
    );

!+
! This state indicates the end of a line.  A comment also signals the
! end of a line.
!-

$STATE (END_OF_LINE,
    (TPA$_EOS, TPA$_EXIT),
    ('!', TPA$_EXIT),
    (TPA$_LAMBDA, TPA$_FAIL)
    );

!+
! Look for the beginning of a known statement type.
!-
$STATE (STATEMENT,
    ('REQUIRE', REQUIRE_FILE),
    ('NAME', EQUALS, , SMG$K_NAME, SMG$$MASK_ADR),
    (FORCED_END, TPA$_FAIL),
    (TPA$_SYMBOL, , MISSING_NAME_REQUIRE),
    (TPA$_ANY, , MISSING_NAME_REQUIRE)
    );

!+
! NAME = 'string' is the format we expect.  Found NAME, now look for '='.
!-
$STATE (EQUALS,
    ((END_OF_LINE), EQUALS, SMG$$NEXT_RECORD),
    ('=', NAME_VALUE, SMG$$STORE_CAP_MASK),
    (TPA$_SYMBOL, , SMG$$SYNTAX_ERROR),
    (TPA$_ANY, , SMG$$SYNTAX_ERROR)
    );
```

```
;   240          0315  1 !+
;   241          0316  1 ! The value of NAME should be a string.
;   242          0317  1 !-
;   243        P 0318  1 $STATE (NAME_VALUE,
;   244        P 0319  1     ((END_OF_LINE), NAME_VALUE, SMG$$NEXT_RECORD),
;   245        P 0320  1     ((STRING), SCAN_DEFINITION),
;   246        P 0321  1     (TPA$_SYMBOL, , SMG$$SYNTAX_ERROR),
;   247        P 0322  1     (TPA$_ANY, , SMG$$SYNTAX_ERROR)
;   248          0323  1     );
;   249          0324  1
;   250          0325  1 !+
;   251          0326  1 ! A string can be bounded by single or double quotes.
;   252          0327  1 !-
;   253        P 0328  1 $STATE (STRING,
;   254        P 0329  1     (SINGLE_QUOTE, SINGLE_QUOTE_STRING),
;   255        P 0330  1     (DOUBLE_QUOTE, DOUBLE_QUOTE_STRING),
;   256        P 0331  1     (TPA$_LAMBDA, TPA$_FAIL)
;   257          0332  1     );
;   258          0333  1
;   259          0334  1 !+
;   260          0335  1 ! The name should be plain text - no escape, control, or expressions allowed.
;   261          0336  1 !-
;   262        P 0337  1 $STATE (SINGLE_QUOTE_STRING,
;   263        P 0338  1     (SINGLE_QUOTE, TPA$_EXIT, SMG$$COPY_CAP),
;   264        P 0339  1     (TPA$_SYMBOL, SINGLE_QUOTE_STRING, SMG$$SAVE_TOKEN_STRING),
;   265        P 0340  1     (DOUBLE_QUOTE, , SMG$$SYNTAX_ERROR),
;   266        P 0341  1     (TPA$_LAMBDA, TPA$_FAIL)
;   267          0342  1     );
;   268          0343  1
;   269        P 0344  1 $STATE (DOUBLE_QUOTE_STRING,
;   270        P 0345  1     (DOUBLE_QUOTE, TPA$_EXIT, SMG$$COPY_CAP),
;   271        P 0346  1     (TPA$_SYMBOL, DOUBLE_QUOTE_STRING, SMG$$SAVE_TOKEN_STRING),
;   272        P 0347  1     (SINGLE_QUOTE, , SMG$$SYNTAX_ERROR),
;   273        P 0348  1     (TPA$_LAMBDA, TPA$_FAIL)
;   274          0349  1     );
;   275          0350  1
;   276          0351  1 !+
;   277          0352  1 ! Found a REQUIRE statement, now find the file name that should follow it.
;   278          0353  1 !-
;   279        P 0354  1 $STATE (REQUIRE_FILE,
;   280        P 0355  1     ((END_OF_LINE), REQUIRE_FILE, SMG$$NEXT_RECORD),
;   281        P 0356  1     ((FILE_SPEC), BEGIN_SCAN, PARSE_REQUIRE),
;   282        P 0357  1     (TPA$_LAMBDA, TPA$_FAIL, MISSING_FILE_SPEC)
;   283          0358  1     );
;   284          0359  1
;   285          0360  1 !+
;   286          0361  1 ! Assume the file name is between single quotes.  Pass the name to $OPEN
;   287          0362  1 ! and let it decide whether it's a valid filespec.
;   288          0363  1 !-
;   289        P 0364  1 $STATE (FILE_SPEC,
;   290        P 0365  1     (SINGLE_QUOTE),
;   291        P 0366  1     (TPA$_LAMBDA, TPA$_FAIL)
;   292          0367  1     );
;   293          0368  1
;   294        P 0369  1 $STATE (GET_FILE_SPEC,
;   295        P 0370  1     (SINGLE_QUOTE, TPA$_EXIT),
;   296        P 0371  1     (TPA$_ANY, GET_FILE_SPEC, SMG$$SAVE_TOKEN_STRING),
```

```
;  297          P 0372  1      (TPA$_LAMBDA, TPA$_FAIL)
;  298            0373  1      );
;  299            0374  1
;  300            0375  1  !+
;  301            0376  1  ! When we get here, we have found a terminal name.  At this point we are
;  302            0377  1  ! expecting some capabilities to be defined.
;  303            0378  1  !-
;  304          P 0379  1  $STATE (SCAN_DEFINITION,
;  305          P 0380  1      ((END_OF_LINE), SCAN_DEFINITION, SMG$$NEXT_RECORD),
;  306          P 0381  1      ('BOOLEAN', SCAN_DEFINITION, PARSE_BOOLEANS),
;  307          P 0382  1      ('NUMERIC', SCAN_DEFINITION, PARSE_NUMERICS),
;  308          P 0383  1      ('STRING', SCAN_DEFINITION, PARSE_STRINGS),
;  309          P 0384  1      ('END', BEGIN_SCAN, ZERO_CUR_TERM_DEF),
;  310          P 0385  1      (TPA$_SYMBOL, , UNRECOGNIZED_STATEMENT),
;  311          P 0386  1      (TPA$_ANY, , UNRECOGNIZED_STATEMENT)
;  312            0387  1      );
;  313            0388  1
```

```
1 %SBTTL 'MISSING_FILE_SPEC - signal missing file specification'
1 ROUTINE MISSING_FILE_SPEC =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       We come here if there is no file spec in the REQUIRE statement.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       MISSING_FILE_SPEC ();
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     SIGNAL_STOP (SMG$_ERRLIN,
2                  1, .SMG$$CURRENT_LINE,
2                  SMG$_MISFILSPE)
2
1     END;                                  ! end of routine MISSING_FILE_SPEC


                                                  .TITLE  SMG$STATEMENT_TABLES TPARSE tables for TERMTABL
                                                                                E statements
                                                  .IDENT  \1-004\

                                                  .PSECT  _LIB$KEY1$,NOWRT,  SHR,  PIC,1

                                        00000 ;TPA$KEYST0
                                              U.23:   .BLKB   0
        45  52  49  55  51  45  52      00000 ;TPA$KEYST
                                              U.25:   .ASCII  \REQUIRE\
                                    FF  00007         .BYTE   -1
                                        00008 ;TPA$KEYST0
```

```
                                                U.29:   .BLKB   0
                    45  4D  41  4E  00008 ;TPA$KEYST
                                                U.31:   .ASCII  \NAME\                                  ;
                                FF  0000C           .BYTE   -1                                      ;
                                FF  0000D ;TPA$KEYFILL
                                                U.43:   .BYTE   -1                                      ;
                                    0000E ;TPA$KEYSTO
                                                U.123:  .BLKB   0
            4E  41  45  4C  4F  4F  42  0000E ;TPA$KEYST
                                                U.125:  .ASCII  \BOOLEAN\                               ;
                                FF  00015           .BYTE   -1                                      ;
                                    00016 ;TPA$KEYSTO
                                                U.129:  .BLKB   0
            43  49  52  45  4D  55  4E  00016 ;TPA$KEYST
                                                U.131:  .ASCII  \NUMERIC\                               ;
                                FF  0001D           .BYTE   -1                                      ;
                                    0001E ;TPA$KEYSTO
                                                U.135:  .BLKB   0
                47  4E  49  52  54  53  0001E ;TPA$KEYST
                                                U.137:  .ASCII  \STRING\                                ;
                                FF  00024           .BYTE   -1                                      ;
                                    00025 ;TPA$KEYSTO
                                                U.141:  .BLKB   0
                        44  4E  45  00025 ;TPA$KEYST
                                                U.143:  .ASCII  \END\                                   ;
                                FF  00028           .BYTE   -1                                      ;
                                FF  00029 ;TPA$KEYFILL
                                                U.151:  .BYTE   -1                                      ;

                                                        .PSECT  _LIB$STATE$,NOWRT,  SHR,  PIC,1

                                    00000 SMG$$A_STMT_STATES::
                                                        .BLKB   0
                                    00000 BEGIN_SCAN:
                                                        .BLKB   0
                            99F8  00000 ;TPA$TYPE
                                                U.2:    .WORD   -26120                                  ;
                            0000* 00002 ;TPA$SUBEXP
                                                U.4:    .WORD   <<U.3-U.4>-2>                           ;
                        00000000V 00004 ;TPA$ACTION
                                                U.5:    .LONG   <<SMG$$NEXT_RECORD-U.5>-4>              ;
                            0000* 00008 ;TPA$TARGET
                                                U.6:    .WORD   <<BEGIN_SCAN-U.6>-2>                    ;
                            9021  0000A ;TPA$TYPE
                                                U.7:    .WORD   -28639                                  ;
                        00000000V 0000C ;TPA$ACTION
                                                U.8:    .LONG   <<SMG$$NEXT_RECORD-U.8>-4>              ;
                            0000* 00010 ;TPA$TARGET
                                                U.9:    .WORD   <<BEGIN_SCAN-U.9>-2>                    ;
                            99F8  00012 ;TPA$TYPE
                                                U.10:   .WORD   -26120                                  ;
                            0000* 00014 ;TPA$SUBEXP
                                                U.12:   .WORD   <<U.11-U.12>-2>                         ;
                        00000000V 00016 ;TPA$ACTION
                                                U.13:   .LONG   <<SMG$$BLANKS_OFF-U.13>-4>              ;
                            0000* 0001A ;TPA$TARGET
                                                U.14:   .WORD   <<BEGIN_SCAN-U.14>-2>                   ;
```

```
                    15F6  0001C ;TPA$TYPE
                                U.15:   .WORD   5622                                    ;
                    FFFF  0001E ;TPA$TARGET
                                U.16:   .WORD   -1                                      ;
                          00020 ;END_OF_LINE
                                U.3:    .BLKB   0
                    11F7  00020 ;TPA$TYPE
                                U.17:   .WORD   4599                                    ;
                    FFFF  00022 ;TPA$TARGET
                                U.18:   .WORD   -1                                      ;
                    1021  00024 ;TPA$TYPE
                                U.19:   .WORD   4129                                    ;
                    FFFF  00026 ;TPA$TARGET
                                U.20:   .WORD   -1                                      ;
                    15F6  00028 ;TPA$TYPE
                                U.21:   .WORD   5622                                    ;
                    FFFE  0002A ;TPA$TARGET
                                U.22:   .WORD   -2                                      ;
                          0002C ;STATEMENT
                                U.11:   .BLKB   0
                    1100  0002C ;TPA$TYPE
                                U.26:   .WORD   4352                                    ;
                    0000* 0002E ;TPA$TARGET
                                U.28:   .WORD   <<U.27-U.28>-2>                         ;
                    7101  00030 ;TPA$TYPE
                                U.32:   .WORD   28929                                   ;
               00000000* 00032 ;TPA$ADDR
                                U.33:   .LONG   <<SMG$$MASK_ADR-U.33>-4>                ;
               0000022B  00036 ;TPA$MASK
                                U.34:   .LONG   555                                     ;
                    0000* 0003A ;TPA$TARGET
                                U.36:   .WORD   <<U.35-U.36>-2>                         ;
                    1000  0003C ;TPA$TYPE
                                U.37:   .WORD   4096                                    ;
                    FFFE  0003E ;TPA$TARGET
                                U.38:   .WORD   -2                                      ;
                    81F1  00040 ;TPA$TYPE
                                U.39:   .WORD   -32271                                  ;
               00000000V 00042 ;TPA$ACTION
                                U.40:   .LONG   <<MISSING_NAME_REQUIRE-U.40>-4>         ;
                    85ED  00046 ;TPA$TYPE
                                U.41:   .WORD   -31251                                  ;
               00000000V 00048 ;TPA$ACTION
                                U.42:   .LONG   <<MISSING_NAME_REQUIRE-U.42>-4>         ;
                          0004C ;EQUALS
                                U.35:   .BLKB   0
                    99F8  0004C ;TPA$TYPE
                                U.44:   .WORD   -26120                                  ;
                    0000* 0004E ;TPA$SUBEXP
                                U.45:   .WORD   <<U.3-U.45>-2>                          ;
               00000000V 00050 ;TPA$ACTION
                                U.46:   .LONG   <<SMG$$NEXT_RECORD-U.46>-4>             ;
                    0000* 00054 ;TPA$TARGET
                                U.47:   .WORD   <<U.35-U.47>-2>                         ;
                    903D  00056 ;TPA$TYPE
                                U.48:   .WORD   -28611                                  ;
               00000000V 00058 ;TPA$ACTION
```

```
                                                U.49:    .LONG   <<SMG$$STORE_CAP_MASK-U.49>-4>          ;
                                    0000* 0005C ;TPA$TARGET
                                                U.51:    .WORD   <<U.50-U.51>-2>                          ;
                                      81F1  0005E ;TPA$TYPE
                                                U.52:    .WORD   -32271                                   ;
                                00000000V 00060 ;TPA$ACTION
                                                U.53:    .LONG   <<SMG$$SYNTAX_ERROR-U.53>-4>             ;
                                      85ED  00064 ;TPA$TYPE
                                                U.54:    .WORD   -31251                                   ;
                                00000000V 00066 ;TPA$ACTION
                                                U.55:    .LONG   <<SMG$$SYNTAX_ERROR-U.55>-4>             ;
                                            0006A ;NAME_VALUE
                                                U.50:    .BLKB   0
                                      99F8  0006A ;TPA$TYPE
                                                U.56:    .WORD   -26120                                   ;
                                    0000* 0006C ;TPA$SUBEXP
                                                U.57:    .WORD   <<U.3-U.57>-2>                           ;
                                00000000V 0006E ;TPA$ACTION
                                                U.58:    .LONG   <<SMG$$NEXT_RECORD-U.58>-4>              ;
                                    0000* 00072 ;TPA$TARGET
                                                U.59:    .WORD   <<U.50-U.59>-2>                          ;
                                      19F8  00074 ;TPA$TYPE
                                                U.60:    .WORD   6648                                     ;
                                    0000* 00076 ;TPA$SUBEXP
                                                U.62:    .WORD   <<U.61-U.62>-2>                          ;
                                    0000* 00078 ;TPA$TARGET
                                                U.64:    .WORD   <<U.63-U.64>-2>                          ;
                                      81F1  0007A ;TPA$TYPE
                                                U.65:    .WORD   -32271                                   ;
                                00000000V 0007C ;TPA$ACTION
                                                U.66:    .LONG   <<SMG$$SYNTAX_ERROR-U.66>-4>             ;
                                      85ED  00080 ;TPA$TYPE
                                                U.67:    .WORD   -31251                                   ;
                                00000000V 00082 ;TPA$ACTION
                                                U.68:    .LONG   <<SMG$$SYNTAX_ERROR-U.68>-4>             ;
                                            00086 ;STRING
                                                U.61:    .BLKB   0
                                      1027  00086 ;TPA$TYPE
                                                U.69:    .WORD   4135                                     ;
                                    0000* 00088 ;TPA$TARGET
                                                U.71:    .WORD   <<U.70-U.71>-2>                          ;
                                      1022  0008A ;TPA$TYPE
                                                U.72:    .WORD   4130                                     ;
                                    0000* 0008C ;TPA$TARGET
                                                U.74:    .WORD   <<U.73-U.74>-2>                          ;
                                      15F6  0008E ;TPA$TYPE
                                                U.75:    .WORD   5622                                     ;
                                      FFFE  00090 ;TPA$TARGET
                                                U.76:    .WORD   -2                                       ;
                                            00092 ;SINGLE_QUOTE_STRING
                                                U.70:    .BLKB   0
                                      9027  00092 ;TPA$TYPE
                                                U.77:    .WORD   -28633                                   ;
                                00000000V 00094 ;TPA$ACTION
                                                U.78:    .LONG   <<SMG$$COPY_CAP-U.78>-4>                 ;
                                      FFFF  00098 ;TPA$TARGET
                                                U.79:    .WORD   -1                                       ;
```

```
                    91F1  0009A ;TPA$TYPE
                                U.80:   .WORD   -28175                                  ;
                00000000V 0009C ;TPA$ACTION
                                U.81:   .LONG   <<SMG$$SAVE_TOKEN_STRING-U.81>-4>       ;
                    0000* 000A0 ;TPA$TARGET
                                U.82:   .WORD   <<U.70-U.82>-2>                         ;
                    8022  000A2 ;TPA$TYPE
                                U.83:   .WORD   -32734                                  ;
                00000000V 000A4 ;TPA$ACTION
                                U.84:   .LONG   <<SMG$$SYNTAX_ERROR-U.84>-4>            ;
                    15F6  000A8 ;TPA$TYPE
                                U.85:   .WORD   5622                                    ;
                    FFFE  000AA ;TPA$TARGET
                                U.86:   .WORD   -2                                      ;
                          000AC ;DOUBLE_QUOTE_STRING
                                U.73:   .BLKB   0
                    9022  000AC ;TPA$TYPE
                                U.87:   .WORD   -28638                                  ;
                00000000V 000AE ;TPA$ACTION
                                U.88:   .LONG   <<SMG$$COPY_CAP-U.88>-4>                ;
                    FFFF  000B2 ;TPA$TARGET
                                U.89:   .WORD   -1                                      ;
                    91F1  000B4 ;TPA$TYPE
                                U.90:   .WORD   -28175                                  ;
                00000000V 000B6 ;TPA$ACTION
                                U.91:   .LONG   <<SMG$$SAVE_TOKEN_STRING-U.91>-4>       ;
                    0000* 000BA ;TPA$TARGET
                                U.92:   .WORD   <<U.73-U.92>-2>                         ;
                    8027  000BC ;TPA$TYPE
                                U.93:   .WORD   -32729                                  ;
                00000000V 000BE ;TPA$ACTION
                                U.94:   .LONG   <<SMG$$SYNTAX_ERROR-U.94>-4>            ;
                    15F6  000C2 ;TPA$TYPE
                                U.95:   .WORD   5622                                    ;
                    FFFE  000C4 ;TPA$TARGET
                                U.96:   .WORD   -2                                      ;
                          000C6 ;REQUIRE_FILE
                                U.27:   .BLKB   0
                    99F8  000C6 ;TPA$TYPE
                                U.97:   .WORD   -26120                                  ;
                    0000* 000C8 ;TPA$SUBEXP
                                U.98:   .WORD   <<U.3-U.98>-2>                          ;
                00000000V 000CA ;TPA$ACTION
                                U.99:   .LONG   <<SMG$$NEXT_RECORD-U.99>-4>             ;
                    0000* 000CE ;TPA$TARGET
                                U.100:  .WORD   <<U.27-U.100>-2>                        ;
                    99F8  000D0 ;TPA$TYPE
                                U.101:  .WORD   -26120                                  ;
                    0000* 000D2 ;TPA$SUBEXP
                                U.103:  .WORD   <<U.102-U.103>-2>                       ;
                00000000V 000D4 ;TPA$ACTION
                                U.104:  .LONG   <<PARSE_REQUIRE-U.104>-4>               ;
                    0000* 000D8 ;TPA$TARGET
                                U.105:  .WORD   <<BEGIN_SCAN-U.105>-2>                  ;
                    95F6  000DA ;TPA$TYPE
                                U.106:  .WORD   -27146                                  ;
                00000000* 000DC ;TPA$ACTION
```

```
                                                U.107:  .LONG   <<MISSING_FILE_SPEC-U.107>-4>           ;
                                FFFE    000E0 ;TPA$TARGET
                                                U.108:  .WORD   -2                                      ;
                                        000E2 ;FILE_SPEC
                                                U.102:  .BLKB   0
                                0027    000E2 ;TPA$TYPE
                                                U.109:  .WORD   39                                      ;
                                15F6    000E4 ;TPA$TYPE
                                                U.110:  .WORD   5622                                    ;
                                FFFE    000E6 ;TPA$TARGET
                                                U.111:  .WORD   -2                                      ;
                                        000E8 GET_FILE_SPEC:
                                                .BLKB   0
                                1027    000E8 ;TPA$TYPE
                                                U.112:  .WORD   4135                                    ;
                                FFFF    000EA ;TPA$TARGET
                                                U.113:  .WORD   -1                                      ;
                                91ED    000EC ;TPA$TYPE
                                                U.114:  .WORD   -28179                                  ;
                            00000000V   000EE ;TPA$ACTION
                                                U.115:  .LONG   <<SMG$$SAVE_TOKEN_STRING-U.115>-4>      ;
                                0000*   000F2 ;TPA$TARGET
                                                U.116:  .WORD   <<GET_FILE_SPEC-U.116>-2>               ;
                                15F6    000F4 ;TPA$TYPE
                                                U.117:  .WORD   5622                                    ;
                                FFFE    000F6 ;TPA$TARGET
                                                U.118:  .WORD   -2                                      ;
                                        000F8 ;SCAN_DEFINITION
                                                U.63:   .BLKB   0
                                99F8    000F8 ;TPA$TYPE
                                                U.119:  .WORD   -26120                                  ;
                                0000*   000FA ;TPA$SUBEXP
                                                U.120:  .WORD   <<U.3-U.120>-2>                         ;
                            00000000V   000FC ;TPA$ACTION
                                                U.121:  .LONG   <<SMG$$NEXT_RECORD-U.121>-4>            ;
                                0000*   00100 ;TPA$TARGET
                                                U.122:  .WORD   <<U.63-U.122>-2>                        ;
                                9102    00102 ;TPA$TYPE
                                                U.126:  .WORD   -28414                                  ;
                            00000000V   00104 ;TPA$ACTION
                                                U.127:  .LONG   <<PARSE_BOOLEANS-U.127>-4>              ;
                                0000*   00108 ;TPA$TARGET
                                                U.128:  .WORD   <<U.63-U.128>-2>                        ;
                                9103    0010A ;TPA$TYPE
                                                U.132:  .WORD   -28413                                  ;
                            00000000V   0010C ;TPA$ACTION
                                                U.133:  .LONG   <<PARSE_NUMERICS-U.133>-4>              ;
                                0000*   00110 ;TPA$TARGET
                                                U.134:  .WORD   <<U.63-U.134>-2>                        ;
                                9104    00112 ;TPA$TYPE
                                                U.138:  .WORD   -28412                                  ;
                            00000000V   00114 ;TPA$ACTION
                                                U.139:  .LONG   <<PARSE_STRINGS-U.139>-4>               ;
                                0000*   00118 ;TPA$TARGET
                                                U.140:  .WORD   <<U.63-U.140>-2>                        ;
                                9105    0011A ;TPA$TYPE
                                                U.144:  .WORD   -28411                                  ;
```

```
                                00000000V 0011C ;TPA$ACTION
                                          U.145:  .LONG   <<ZERO_CUR_TERM_DEF-U.145>-4>           ;
                                    0000* 00120 ;TPA$TARGET
                                          U.146:  .WORD   <<BEGIN_SCAN-U.146>-2>                  ;
                                    81F1  00122 ;TPA$TYPE
                                          U.147:  .WORD   -32271                                  ;
                                00000000V 00124 ;TPA$ACTION
                                          U.148:  .LONG   <<UNRECOGNIZED_STATEMENT-U.148>-4>      ;
                                    85ED  00128 ;TPA$TYPE
                                          U.149:  .WORD   -31251                                  ;
                                00000000V 0012A ;TPA$ACTION
                                          U.150:  .LONG   <<UNRECOGNIZED_STATEMENT-U.150>-4>      ;

                                                  .PSECT  _LIB$KEY0$,NOWRT,  SHR,  PIC,1

                                          00000 SMG$$A_STMT_KEYWDS::
                                                  .BLKB   0
                                          00000 ;TPA$KEY0
                                          U.1:    .BLKB   0
                                    0000* 00000 ;TPA$KEY
                                          U.24:   .WORD   <U.23-U.1>                              ;
                                    0000* 00002 ;TPA$KEY
                                          U.30:   .WORD   <U.29-U.1>                              ;
                                    0000* 00004 ;TPA$KEY
                                          U.124:  .WORD   <U.123-U.1>                             ;
                                    0000* 00006 ;TPA$KEY
                                          U.130:  .WORD   <U.129-U.1>                             ;
                                    0000* 00008 ;TPA$KEY
                                          U.136:  .WORD   <U.135-U.1>                             ;
                                    0000* 0000A ;TPA$KEY
                                          U.142:  .WORD   <U.141-U.1>                             ;

                                                  .PSECT  _SMG$DATA,NOEXE,  PIC,2

                                00000000  00000 SMG$$MASK_ADR::
                                                  .LONG   0                                       ;
                                FFFFFFFE  00004 SMG$$NEXT_NEGATIVE_NUMBER::
                                                  .LONG   -2                                      ;
                                00000000  00008 SMG$$CURRENT_LINE::
                                                  .LONG   0                                       ;
                                00000005  0000C SMG$$CURRENT_BLOCK::
                                                  .LONG   5                                       ;
                                00000002  00010 SMG$$CURRENT_DEF_BLOCK::
                                                  .LONG   2                                       ;
                                00000600  00014 SMG$$DATA_OFFSET::
                                                  .LONG   1536                                    ;

                                                  .EXTRN  LIB$TPARSE, STR$UPCASE
                                                  .EXTRN  STR$COPY_DX, SMG$_ERRLIN
                                                  .EXTRN  SMG$_ERRAT_LIN, SMG$_FAIOPEFIL
                                                  .EXTRN  SMG$_MISTERNAM, SMG$_SYNERR
                                                  .EXTRN  SMG$_UNRECSTA, SMG$_MISFILSPE
                                                  .EXTRN  SMG$_MISENDSTA, SMG$_MISNAMREQ
                                                  .EXTRN  TERM$_TABLE, TERM$_NAME
                                                  .EXTRN  TERM$_NUM, SMG$$A_BOOLEAN_STATES
                                                  .EXTRN  SMG$$A_BOOLEAN_KEYWDS
                                                  .EXTRN  SMG$$A_NUMERIC_STATES
```

```
                                                        .EXTRN  SMG$$A_NUMERIC_KEYWDS
                                                        .EXTRN  SMG$$A_STRING_STATES
                                                        .EXTRN  SMG$$A_STRING_KEYWDS

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

                                        0000 00000 MISSING_FILE_SPEC:
                                                        .WORD   Save nothing                    ; 0390
                                00000000G  00   9F 00002        PUSHAB  SMG$_MISFILSPE          ; 0427
                                00000000'  EF   DD 00008        PUSHL   SMG$$CURRENT_LINE       ; 0428
                                           01   DD 0000E        PUSHL   #1                      ; 0427
                                00000000G  00   9F 00010        PUSHAB  SMG$_ERRLIN             ;
                    00000000G   00              04   FB 00016  CALLS   #4, LIB$STOP            ;
                                                04   0001D    RET                             ; 0431

; Routine Size:  30 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
1 %SBTTL 'MISSING_NAME_REQUIRE - missing expected NAME or REQUIRE'
1 ROUTINE MISSING_NAME_REQUIRE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       We come here if there is some statement other than NAME or REQUIRE
1 !       starting a definition.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       MISSING_NAME_REQUIRE ();
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     SIGNAL_STOP (SMG$_ERRAT_LIN,
2                  3, .SMG$$CURRENT_LINE,
2                  .AP [TPA$L_TOKENCNT],
2                  .AP [TPA$L_TOKENPTR],
2                  SMG$_MISNAMREQ)
2
1     END;                                  ! end of routine MISSING_FILE_SPEC


                                 0000 00000 MISSING_NAME_REQUIRE:
                                                .WORD   Save nothing                    ; 0433
          00000000G  00  9F 00002               PUSHAB  SMG$_MISNAMREQ                  ; 0471
      7E        10   AC  7D 00008               MOVQ    16(AP), -(SP)                   ; 0473
          00000000'  EF  DD 0000C               PUSHL   SMG$$CURRENT_LINE               ; 0472
                     03  DD 00012               PUSHL   #3                              ; 0471
```

```
                                00000000G  00  9F 00014        PUSHAB  SMG$_ERRAT_LIN                          ;
                    00000000G  00          06  FB 0001A        CALLS   #6, LIB$STOP                            ;
                                               04 00021        RET                                             ; 0477

; Routine Size:  34 bytes,    Routine Base:  _SMG$CODE + 001E
```

```
;  406     0478  1 %SBTTL 'PARSE_BOOLEANS - parse boolean capabilities'
;  407     0479  1 ROUTINE PARSE_BOOLEANS =
;  408     0480  1
;  409     0481  1 !++
;  410     0482  1 ! FUNCTIONAL DESCRIPTION:
;  411     0483  1 !
;  412     0484  1 !       Parse boolean capabilities by calling LIB$TPARSE with the
;  413     0485  1 !       boolean state tables and keywords.  We use separate tables
;  414     0486  1 !       in order to avoid bumping into the 220 keyword limit.
;  415     0487  1 !
;  416     0488  1 ! CALLING SEQUENCE:
;  417     0489  1 !
;  418     0490  1 !       status = PARSE_BOOLEANS ();
;  419     0491  1 !
;  420     0492  1 ! FORMAL PARAMETERS:
;  421     0493  1 !
;  422     0494  1 !       NONE
;  423     0495  1 !
;  424     0496  1 ! IMPLICIT INPUTS:
;  425     0497  1 !
;  426     0498  1 !       AP      Points to TPARSE parameter block
;  427     0499  1 !
;  428     0500  1 ! IMPLICIT OUTPUTS:
;  429     0501  1 !
;  430     0502  1 !       NONE
;  431     0503  1 !
;  432     0504  1 ! COMPLETION STATUS:
;  433     0505  1 !
;  434     0506  1 !        SS$_NORMAL
;  435     0507  1 !
;  436     0508  1 ! SIDE EFFECTS:
;  437     0509  1 !
;  438     0510  1 !--
;  439     0511  1
;  440     0512  2     BEGIN
;  441     0513  2     BUILTIN
;  442     0514  2         CALLG,
;  443     0515  2         AP;
;  444     0516  2     MAP
;  445     0517  2         AP : REF BLOCK [,BYTE];
;  446     0518  2     LOCAL
;  447     0519  2         STATUS;
;  448     0520  2
;  449     0521  2 !+
;  450     0522  2 ! Should have seen a NAME statement by the time we reach here.
;  451     0523  2 !-
;  452     0524  2     IF .AP [PARAM_L_CUR_TERM_DEF] EQL 0
;  453     0525  2     THEN
;  454     0526  2         SIGNAL_STOP (SMG$_ERRLIN,
;  455     0527  2                      1, .SMG$$CURRENT_LINE,
;  456     0528  2                      SMG$_MISTERNAM);
;  457     0529  2
;  458     0530  2 !+
;  459     0531  2 ! We reuse the same TPARSE parameter block.
;  460     0532  2 !-
;  461     0533  3     IF NOT (STATUS = LIB$TPARSE (.AP, SMG$$A_BOOLEAN_STATES,
;  462     0534  3                                  SMG$$A_BOOLEAN_KEYWDS))
```

```
;  463          0535  2      THEN
;  464          0536  2          SIGNAL_STOP (.STATUS);
;  465          0537  2
;  466          0538  2      RETURN (SS$_NORMAL);
;  467          0539  1      END;                                  ! end of routine PARSE_BOOLEANS



                                                  0004 00000 PARSE_BOOLEANS:
                                                                  .WORD   Save R2                                  ; 0479
                          52 00000000G  00  9E 00002              MOVAB   LIB$STOP, R2                             ;
                                   48   AC  D5 00009              TSTL    72(AP)                                   ; 0524
                                        17  12 0000C              BNEQ    1$                                       ;
                             00000000G  00  9F 0000E              PUSHAB  SMG$_MISTERNAM                           ; 0526
                             00000000'  EF  DD 00014              PUSHL   SMG$$CURRENT_LINE                        ; 0527
                                        01  DD 0001A              PUSHL   #1                                       ; 0526
                             00000000G  00  9F 0001C              PUSHAB  SMG$_ERRLIN                              ;
                          62            04  FB 00022              CALLS   #4, LIB$STOP                             ;
                             00000000G  00  9F 00025 1$:          PUSHAB  SMG$$A_BOOLEAN_KEYWDS                    ; 0533
                             00000000G  00  9F 0002B              PUSHAB  SMG$$A_BOOLEAN_STATES                    ;
                                        5C  DD 00031              PUSHL   AP                                       ;
                00000000G  00           03  FB 00033              CALLS   #3, LIB$TPARSE                           ;
                           05           50  E8 0003A              BLBS    STATUS, 2$                               ;
                                        50  DD 0003D              PUSHL   STATUS                                   ; 0536
                          62            01  FB 0003F              CALLS   #1, LIB$STOP                             ;
                          50            01  D0 00042 2$:          MOVL    #1, R0                                   ; 0538
                                            04 00045              RET                                              ; 0539

; Routine Size:  70 bytes,    Routine Base:  _SMG$CODE + 0040
```

```
;  469          0540  1 %SBTTL 'PARSE_NUMERICS - parse numeric capabilities'
;  470          0541  1 ROUTINE PARSE_NUMERICS =
;  471          0542  1
;  472          0543  1 !++
;  473          0544  1 ! FUNCTIONAL DESCRIPTION:
;  474          0545  1 !
;  475          0546  1 !       Parse numeric capabilities by calling LIB$TPARSE with the
;  476          0547  1 !       numeric state and keyword tables.  We use separate tables
;  477          0548  1 !       in order to avoid bumping into the 220 keyword limit.
;  478          0549  1 !
;  479          0550  1 ! CALLING SEQUENCE:
;  480          0551  1 !
;  481          0552  1 !       status = PARSE_NUMERICS ();
;  482          0553  1 !
;  483          0554  1 ! FORMAL PARAMETERS:
;  484          0555  1 !
;  485          0556  1 !       NONE
;  486          0557  1 !
;  487          0558  1 ! IMPLICIT INPUTS:
;  488          0559  1 !
;  489          0560  1 !       AP      Points to TPARSE parameter block
;  490          0561  1 !
;  491          0562  1 ! IMPLICIT OUTPUTS:
;  492          0563  1 !
;  493          0564  1 !       NONE
;  494          0565  1 !
;  495          0566  1 ! COMPLETION STATUS:
;  496          0567  1 !
;  497          0568  1 !        SS$_NORMAL
;  498          0569  1 !
;  499          0570  1 ! SIDE EFFECTS:
;  500          0571  1 !
;  501          0572  1 !--
;  502          0573  1
;  503          0574  2     BEGIN
;  504          0575  2     BUILTIN
;  505          0576  2         AP;
;  506          0577  2     MAP
;  507          0578  2         AP : REF BLOCK [,BYTE];
;  508          0579  2     LOCAL
;  509          0580  2         STATUS;
;  510          0581  2
;  511          0582  2 !+
;  512          0583  2 ! Should have seen a NAME statement by the time we reach here.
;  513          0584  2 !-
;  514          0585  2     IF .AP [PARAM_L_CUR_TERM_DEF] EQL 0
;  515          0586  2     THEN
;  516          0587  2         SIGNAL_STOP (SMG$_ERRLIN,
;  517          0588  2                      1, .SMG$$CURRENT_LINE,
;  518          0589  2                      SMG$_MISTERNAM);
;  519          0590  2
;  520          0591  2 !+
;  521          0592  2 ! Reuse the same TPARSE parameter block.
;  522          0593  2 !-
;  523          0594  3     IF NOT (STATUS = LIB$TPARSE (.AP, SMG$$A_NUMERIC_STATES,
;  524          0595  3                                  SMG$$A_NUMERIC_KEYWDS))
;  525          0596  2     THEN
```

```
;  526        0597  2          SIGNAL_STOP (.STATUS);
;  527        0598  2
;  528        0599  2      RETURN (SS$_NORMAL);
;  529        0600  1      END;                                  ! end of routine PARSE_NUMERICS



                                                  0004 00000 PARSE_NUMERICS:
                                                                        .WORD   Save R2                                  ; 0541
                                  52 00000000G  00  9E 00002            MOVAB   LIB$STOP, R2                             ;
                                            48  AC  D5 00009            TSTL    72(AP)                                   ; 0585
                                                17  12 0000C            BNEQ    1$                                       ;
                                     00000000G  00  9F 0000E            PUSHAB  SMG$_MISTERNAM                           ; 0587
                                     00000000'  EF  DD 00014            PUSHL   SMG$$CURRENT_LINE                        ; 0588
                                                01  DD 0001A            PUSHL   #1                                       ; 0587
                                     00000000G  00  9F 0001C            PUSHAB  SMG$_ERRLIN                              ;
                                  62            04  FB 00022            CALLS   #4, LIB$STOP                             ;
                                     00000000G  00  9F 00025 1$:        PUSHAB  SMG$$A_NUMERIC_KEYWDS                    ; 0594
                                     00000000G  00  9F 0002B            PUSHAB  SMG$$A_NUMERIC_STATES                    ;
                                                5C  DD 00031            PUSHL   AP                                       ;
                          00000000G  00         03  FB 00033            CALLS   #3, LIB$TPARSE                           ;
                                  05            50  E8 0003A            BLBS    STATUS, 2$                               ;
                                                50  DD 0003D            PUSHL   STATUS                                   ; 0597
                                  62            01  FB 0003F            CALLS   #1, LIB$STOP                             ;
                                  50            01  D0 00042 2$:        MOVL    #1, R0                                   ; 0599
                                                    04 00045            RET                                              ; 0600

; Routine Size:  70 bytes,    Routine Base:  _SMG$CODE + 0086
```

```
;   531          0601  1 %SBTTL 'PARSE_REQUIRE - parse definitions in REQUIRE file'
;   532          0602  1 ROUTINE PARSE_REQUIRE =
;   533          0603  1
;   534          0604  1 !++
;   535          0605  1 ! FUNCTIONAL DESCRIPTION:
;   536          0606  1 !
;   537          0607  1 !       In order to parse the definitions contained in a REQUIRE file,
;   538          0608  1 !       we call ourselve recursively.  Notice that the REQUIRE file may
;   539          0609  1 !       contain another REQUIRE statement, etc.
;   540          0610  1 !
;   541          0611  1 ! CALLING SEQUENCE:
;   542          0612  1 !
;   543          0613  1 !       status = PARSE_REQUIRE ();
;   544          0614  1 !
;   545          0615  1 ! FORMAL PARAMETERS:
;   546          0616  1 !
;   547          0617  1 !       NONE
;   548          0618  1 !
;   549          0619  1 ! IMPLICIT INPUTS:
;   550          0620  1 !
;   551          0621  1 !       AP      Points to TPARSE parameter block
;   552          0622  1 !
;   553          0623  1 ! IMPLICIT OUTPUTS:
;   554          0624  1 !
;   555          0625  1 !       NONE
;   556          0626  1 !
;   557          0627  1 ! COMPLETION STATUS:
;   558          0628  1 !
;   559          0629  1 !        SS$_NORMAL
;   560          0630  1 !
;   561          0631  1 ! SIDE EFFECTS:
;   562          0632  1 !
;   563          0633  1 !--
;   564          0634  1
;   565          0635  2     BEGIN
;   566          0636  2     BUILTIN
;   567          0637  2         CALLG,
;   568          0638  2         AP;
;   569          0639  2     MAP
;   570          0640  2         AP : REF BLOCK [,BYTE];
;   571          0641  2     LOCAL
;   572          0642  2         REQUIRE_FAB : $FAB_DECL,          ! FAB for required file
;   573          0643  2         REQUIRE_RAB : $RAB_DECL,          ! RAB for required file
;   574          0644  2         SAVED_FAB,                        ! addr of original FAB
;   575          0645  2         SAVED_RAB,                        ! addr of original FAB
;   576          0646  2         SAVED_CUR_LINE,                   ! cur line in original file
;   577          0647  2         TEMP_BUFFER : VECTOR [255,BYTE],  ! buffer to hold records
;   578          0648  2         OPEN_STATUS,                      ! status retd by $OPEN
;   579          0649  2         CLOSE_STATUS,                     ! status retd by $CLOSE
;   580          0650  2         STATUS;                           ! status retd by called routines
;   581          0651  2
;   582          0652  2 !+
;   583          0653  2 ! In order to parse records from a different file, we need to reset the
;   584          0654  2 ! FAB and RAB fields stored in the TPARSE parameter block.  We save the
;   585          0655  2 ! addresses of the original FAB and RAB so that we can continue reading
;   586          0656  2 ! those records when we are done here.
;   587          0657  2 !-
```

```
;  588          0658  2
;  589          0659  2      SAVED_FAB = .AP [PARAM_A_TXT_FAB];
;  590          0660  2      SAVED_RAB = .AP [PARAM_A_TXT_RAB];
;  591          0661  2      SAVED_CUR_LINE = .SMG$$CURRENT_LINE;
;  592          0662  2                                      ! remember where we are in first file
;  593        P 0663  2      $FAB_INIT (FAB = REQUIRE_FAB, FNS = .AP [PARAM_L_SAVED_TOKENCNT],
;  594        P 0664  2                 FNA = .AP [PARAM_L_SAVED_TOKENSTR], DNM = '.TXT',
;  595          0665  2                 FAC = GET, SHR = GET, ORG = SEQ);
;  596          0666  2
;  597          0667  3      IF NOT (OPEN_STATUS = $OPEN (FAB = REQUIRE_FAB))
;  598          0668  2      THEN
;  599          0669  2          SIGNAL_STOP (SMG$_FAIOPEFIL,
;  600          0670  2                       2, .AP [TPA$L_TOKENCNT],
;  601          0671  2                       .AP [TPA$L_TOKENPTR],
;  602          0672  2                       .OPEN_STATUS);
;  603          0673  2
;  604          0674  2      AP [PARAM_L_SAVED_TOKENCNT] = 0;
;  605          0675  2      AP [PARAM_L_SAVED_TOKENSTR] = 0;
;  606          0676  2                                      ! re-init
;  607          0677  2
;  608          0678  2  !+
;  609          0679  2  ! Connect a RAB to the FAB so we can access the file.
;  610          0680  2  !-
;  611          0681  2
;  612          0682  2      $RAB_INIT (RAB = REQUIRE_RAB, FAB = REQUIRE_FAB);
;  613          0683  2
;  614          0684  2      $CONNECT (RAB = REQUIRE_RAB);
;  615          0685  2
;  616          0686  2      REQUIRE_RAB [RAB$W_USZ] = 255;
;  617          0687  2      REQUIRE_RAB [RAB$L_UBF] = TEMP_BUFFER [0];
;  618          0688  2                                      ! set up buffer for RMS to use
;  619          0689  2  !+
;  620          0690  2  ! Put the new FAB and RAB into the parameter block.
;  621          0691  2  !-
;  622          0692  2
;  623          0693  2      AP [PARAM_A_TXT_FAB] = REQUIRE_FAB;
;  624          0694  2      AP [PARAM_A_TXT_RAB] = REQUIRE_RAB;
;  625          0695  2
;  626          0696  2  !+
;  627          0697  2  ! Call ourself recursively, reusing the current TPARSE block.
;  628          0698  2  !-
;  629          0699  2
;  630          0700  3      IF NOT (STATUS = LIB$TPARSE (.AP, SMG$$A_STMT_STATES,
;  631          0701  3                                   SMG$$A_STMT_KEYWDS))
;  632          0702  2      THEN
;  633          0703  2          SIGNAL_STOP (.STATUS);
;  634          0704  2
;  635          0705  2  !+
;  636          0706  2  ! Restore the original FAB and RAB addresses to the parameter block.
;  637          0707  2  !-
;  638          0708  2
;  639          0709  2      AP [PARAM_A_TXT_FAB] = .SAVED_FAB;
;  640          0710  2      AP [PARAM_A_TXT_RAB] = .SAVED_RAB;
;  641          0711  2
;  642          0712  2      SMG$$CURRENT_LINE = .SAVED_CUR_LINE;
;  643          0713  2
;  644          0714  2  !+
```

```
;  645          0715  2 ! Done with the required file, close it.
;  646          0716  2 !-
;  647          0717  2
;  648          0718  3     IF NOT (CLOSE_STATUS = $CLOSE (FAB = REQUIRE_FAB))
;  649          0719  2     THEN
;  650          0720  2         SIGNAL_STOP (.CLOSE_STATUS);
;  651          0721  2
;  652          0722  2 !+
;  653          0723  2 ! Get the next record from the original definition file.
;  654          0724  2 !-
;  655          0725  2
;  656          0726  2     CALLG (.AP, SMG$$NEXT_RECORD);
;  657          0727  2
;  658          0728  2     RETURN (SS$_NORMAL);
;  659          0729  1     END;                                  ! end of routine PARSE_REQUIRE


                                    54  58  54  2E  000CC P.AAA:  .ASCII  \.TXT\                                  ;

                                                                  .EXTRN  SYS$OPEN, SYS$CONNECT
                                                                  .EXTRN  SYS$CLOSE

                                                 07FC 00000 PARSE_REQUIRE:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10          ; 0602
                                 5A 00000000'  EF  9E 00002       MOVAB   SMG$$CURRENT_LINE, R10                    ;
                                 59 00000000G  00  9E 00009       MOVAB   LIB$STOP, R9                              ;
                                 5E     FE6C   CE  9E 00010       MOVAB   -404(SP), SP                              ;
                                 58       24   AC  D0 00015       MOVL    36(AP), SAVED_FAB                         ; 0659
                                 57       28   AC  D0 00019       MOVL    40(AP), SAVED_RAB                         ; 0660
                                 56            6A  D0 0001D       MOVL    SMG$$CURRENT_LINE, SAVED_CUR_LINE         ; 0661
   0050   8F          00         6E            00  2C 00020       MOVC5   #0, (SP), #0, #80, $RMS_PTR               ; 0665
                                          B0   AD     00027                                                        ;
                          B0   AD     5003   8F  B0 00029       MOVW    #20483, $RMS_PTR                          ;
                          C6   AD     0202   8F  B0 0002F       MOVW    #514, $RMS_PTR+22                         ;
                                          CD   AD  94 00035       CLRB    $RMS_PTR+29                               ;
                          CF   AD            02  90 00038       MOVB    #2, $RMS_PTR+31                           ;
                          DC   AD       58   AC  D0 0003C       MOVL    88(AP), $RMS_PTR+44                       ;
                          E0   AD       B8   AF  9E 00041       MOVAB   P.AAA, $RMS_PTR+48                        ;
                          E4   AD       54   AC  90 00046       MOVB    84(AP), $RMS_PTR+52                       ;
                          E5   AD            04  90 0004B       MOVB    #4, $RMS_PTR+53                           ;
                                          B0   AD  9F 0004F       PUSHAB  REQUIRE_FAB                               ; 0667
                     00000000G   00         01  FB 00052       CALLS   #1, SYS$OPEN                              ;
                                 11         50  E8 00059       BLBS    OPEN_STATUS, 1$                           ;
                                            50  DD 0005C       PUSHL   OPEN_STATUS                               ; 0672
                                 7E       10   AC  7D 0005E       MOVQ    16(AP), -(SP)                             ; 0670
                                            02  DD 00062       PUSHL   #2                                        ; 0669
                                 00000000G   00  9F 00064       PUSHAB  SMG$_FAIOPEFIL                            ;
                                 69         05  FB 0006A       CALLS   #5, LIB$STOP                              ;
                                          54   AC  7C 0006D 1$:  CLRQ    84(AP)                                    ; 0674
   0044   8F          00         6E            00  2C 00070       MOVC5   #0, (SP), #0, #68, $RMS_PTR               ; 0682
                                       FF6C   CD     00077                                                        ;
                          FF6C   CD     4401   8F  B0 0007A       MOVW    #17409, $RMS_PTR                          ;
                          A8   AD       B0   AD  9E 00081       MOVAB   REQUIRE_FAB, $RMS_PTR+60                  ;
                                       FF6C   CD  9F 00086       PUSHAB  REQUIRE_RAB                               ; 0684
                     00000000G   00         01  FB 0008A       CALLS   #1, SYS$CONNECT                           ;
                          8C   AD       FF   8F  9B 00091       MOVZBW  #255, REQUIRE_RAB+32                      ; 0686
```

```
                    90   AD              6E  9E 00096          MOVAB   TEMP_BUFFER, REQUIRE_RAB+36          ;0687
                    24   AC        B0    AD  9E 0009A          MOVAB   REQUIRE_FAB, 36(AP)                  ;0693
                    28   AC      FF6C    CD  9E 0009F          MOVAB   REQUIRE_RAB, 40(AP)                  ;0694
                            0000'   CF    9F 000A5          PUSHAB  SMG$$A_STMT_KEYWDS                   ;0700
                            0000'   CF    9F 000A9          PUSHAB  SMG$$A_STMT_STATES                   ;
                                    5C    DD 000AD          PUSHL   AP                                   ;
                00000000G   00            03  FB 000AF          CALLS   #3, LIB$TPARSE                       ;
                            05            50  E8 000B6          BLBS    STATUS, 2$                           ;
                                          50  DD 000B9          PUSHL   STATUS                               ;0703
                            69            01  FB 000BB          CALLS   #1, LIB$STOP                         ;
                    24   AC               58  D0 000BE 2$:      MOVL    SAVED_FAB, 36(AP)                    ;0709
                    28   AC               57  D0 000C2          MOVL    SAVED_RAB, 40(AP)                    ;0710
                            6A            56  D0 000C6          MOVL    SAVED_CUR_LINE, SMG$$CURRENT_LINE    ;0712
                              B0          AD  9F 000C9          PUSHAB  REQUIRE_FAB                          ;0718
                00000000G   00            01  FB 000CC          CALLS   #1, SYS$CLOSE                        ;
                            05            50  E8 000D3          BLBS    CLOSE_STATUS, 3$                     ;
                                          50  DD 000D6          PUSHL   CLOSE_STATUS                         ;0720
                            69            01  FB 000D8          CALLS   #1, LIB$STOP                         ;
                     0000V  CF            6C  FA 000DB 3$:      CALLG   (AP), SMG$$NEXT_RECORD               ;0726
                            50            01  D0 000E0          MOVL    #1, R0                               ;0728
                                          04  000E3          RET                                          ;0729

; Routine Size:  228 bytes,    Routine Base:  _SMG$CODE + 00D0
```

```
;  661     0730  1 %SBTTL 'PARSE_STRINGS - parse string capabilities'
;  662     0731  1 ROUTINE PARSE_STRINGS =
;  663     0732  1
;  664     0733  1 !++
;  665     0734  1 ! FUNCTIONAL DESCRIPTION:
;  666     0735  1 !
;  667     0736  1 !       Parse string capabilities by calling LIB$TPARSE with the string
;  668     0737  1 !       state and keyword tables.  We use separate tables to avoid bumping
;  669     0738  1 !       into the 220 keyword limit.
;  670     0739  1 !
;  671     0740  1 ! CALLING SEQUENCE:
;  672     0741  1 !
;  673     0742  1 !       status = PARSE_STRINGS ();
;  674     0743  1 !
;  675     0744  1 ! FORMAL PARAMETERS:
;  676     0745  1 !
;  677     0746  1 !       NONE
;  678     0747  1 !
;  679     0748  1 ! IMPLICIT INPUTS:
;  680     0749  1 !
;  681     0750  1 !       AP      Points to TPARSE parameter block
;  682     0751  1 !
;  683     0752  1 ! IMPLICIT OUTPUTS:
;  684     0753  1 !
;  685     0754  1 !       NONE
;  686     0755  1 !
;  687     0756  1 ! COMPLETION STATUS:
;  688     0757  1 !
;  689     0758  1 !        SS$_NORMAL
;  690     0759  1 !
;  691     0760  1 ! SIDE EFFECTS:
;  692     0761  1 !
;  693     0762  1 !--
;  694     0763  1
;  695     0764  2     BEGIN
;  696     0765  2     BUILTIN
;  697     0766  2         AP;
;  698     0767  2     MAP
;  699     0768  2         AP : REF BLOCK [,BYTE];
;  700     0769  2     LOCAL
;  701     0770  2         STATUS;
;  702     0771  2
;  703     0772  2 !+
;  704     0773  2 ! Should have seen a NAME statement by the time we reach here.
;  705     0774  2 !-
;  706     0775  2     IF .AP [PARAM_L_CUR_TERM_DEF] EQL 0
;  707     0776  2     THEN
;  708     0777  2         SIGNAL_STOP (SMG$_ERRLIN,
;  709     0778  2                      1, .SMG$$CURRENT_LINE,
;  710     0779  2                      SMG$_MISTERNAM);
;  711     0780  2
;  712     0781  2 !+
;  713     0782  2 ! Reuse the same TPARSE block.
;  714     0783  2 !-
;  715     0784  3     IF NOT (STATUS = LIB$TPARSE (.AP, SMG$$A_STRING_STATES,
;  716     0785  3                                  SMG$$A_STRING_KEYWDS))
;  717     0786  2     THEN
```

```
;  718          0787  2          SIGNAL_STOP (.STATUS);
;  719          0788  2
;  720          0789  2      RETURN (SS$_NORMAL);
;  721          0790  1      END;                                  ! end of routine PARSE_STRINGS



                                                  0004 00000 PARSE_STRINGS:
                                                                  .WORD   Save R2                                  ; 0731
                          52 00000000G  00  9E 00002              MOVAB   LIB$STOP, R2                             ;
                                   48   AC  D5 00009              TSTL    72(AP)                                   ; 0775
                                        17  12 0000C              BNEQ    1$                                       ;
                             00000000G  00  9F 0000E              PUSHAB  SMG$_MISTERNAM                           ; 0777
                             00000000'  EF  DD 00014              PUSHL   SMG$$CURRENT_LINE                        ; 0778
                                        01  DD 0001A              PUSHL   #1                                       ; 0777
                             00000000G  00  9F 0001C              PUSHAB  SMG$_ERRLIN                              ;
                          62            04  FB 00022              CALLS   #4, LIB$STOP                             ;
                             00000000G  00  9F 00025 1$:          PUSHAB  SMG$$A_STRING_KEYWDS                     ; 0784
                             00000000G  00  9F 0002B              PUSHAB  SMG$$A_STRING_STATES                     ;
                                        5C  DD 00031              PUSHL   AP                                       ;
                  00000000G  00         03  FB 00033              CALLS   #3, LIB$TPARSE                           ;
                             05         50  E8 0003A              BLBS    STATUS, 2$                               ;
                                        50  DD 0003D              PUSHL   STATUS                                   ; 0787
                          62            01  FB 0003F              CALLS   #1, LIB$STOP                             ;
                          50            01  D0 00042 2$:          MOVL    #1, R0                                   ; 0789
                                            04 00045              RET                                              ; 0790

; Routine Size:  70 bytes,    Routine Base:  _SMG$CODE + 01B4
```

```
;  723          0791  1 %SBTTL 'SEARCH_KNOWN_TYPES - search table of known terminal types'
;  724          0792  1 ROUTINE SEARCH_KNOWN_TYPES =
;  725          0793  1
;  726          0794  1 !++
;  727          0795  1 ! FUNCTIONAL DESCRIPTION:
;  728          0796  1 !
;  729          0797  1 !       Search a table of DEC terminals to see if the user is using a
;  730          0798  1 !       known terminal name in his definition.  (SMG will define DEC
;  731          0799  1 !       terminals - users shouldn't but just in case...)  We want to
;  732          0800  1 !       assign the positive DT$_xx type instead of a negative number
;  733          0801  1 !       for DEC terminals.
;  734          0802  1 !
;  735          0803  1 ! CALLING SEQUENCE:
;  736          0804  1 !
;  737          0805  1 !       status = SEARCH_KNOWN_TYPES ();
;  738          0806  1 !
;  739          0807  1 ! FORMAL PARAMETERS:
;  740          0808  1 !
;  741          0809  1 !       NONE
;  742          0810  1 !
;  743          0811  1 ! IMPLICIT INPUTS:
;  744          0812  1 !
;  745          0813  1 !       AP      Points to TPARSE parameter block
;  746          0814  1 !
;  747          0815  1 ! IMPLICIT OUTPUTS:
;  748          0816  1 !
;  749          0817  1 !       NONE
;  750          0818  1 !
;  751          0819  1 ! COMPLETION STATUS:
;  752          0820  1 !
;  753          0821  1 !         SS$_NORMAL
;  754          0822  1 !
;  755          0823  1 ! SIDE EFFECTS:
;  756          0824  1 !
;  757          0825  1 !--
;  758          0826  1
;  759          0827  2     BEGIN
;  760          0828  2     BUILTIN
;  761          0829  2         CALLG,
;  762          0830  2         AP;
;  763          0831  2     MAP
;  764          0832  2         AP : REF BLOCK [,BYTE];
;  765          0833  2     LOCAL
;  766          0834  2         TYPE : INITIAL (-1),            ! terminal type
;  767          0835  2         PTR_UPCASED_NAME,               ! addr of upcased terminal name
;  768          0836  2         TXT_RAB : REF $RAB_DECL;        ! ptr to RAB for TERMTABLE.TXT
;  769          0837  2     MACRO
;  770          0838  2         TERM$B_TYPE = 1, 0, 8, 0%;
;  771          0839  2
;  772          0840  2 !+
;  773          0841  2 ! The saved tokenptr points to the user's original text.  For comparison
;  774          0842  2 ! purposes, we need to look at the upcased version.  So index into upcased
;  775          0843  2 ! buffer.
;  776          0844  2 !-
;  777          0845  2     TXT_RAB = .AP [PARAM_A_TXT_RAB];
;  778          0846  2
;  779          0847  2     PTR_UPCASED_NAME = .TXT_RAB [RAB$L_UBF] +
```

```
2                                   (.AP [PARAM_L_SAVED_TOKENSTR] - .AP [PARAM_L_ORIG_TXT]);
2
2 !+
2 ! Search table of names indexed by type.
2 !-
2     INCR INDEX FROM 0 TO TERM$_NUM -1 DO
3         BEGIN
3         BIND
3             NAME = .TERM$_NAME [.INDEX] : $BBLOCK;
3         IF CH$EQL (.AP [PARAM_L_SAVED_TOKENCNT], .PTR_UPCASED_NAME,
3                    .AP [PARAM_L_SAVED_TOKENCNT], .NAME [DSC$A_POINTER])
3         THEN
4             BEGIN
4             TYPE = .TERM$_TABLE [.INDEX, TERM$B_TYPE];
4             EXITLOOP;
3             END;
2         END;                                  ! end of INCR loop
2
2 !+
2 ! If we found the name, then use its type number.  If we exit this routine
2 ! without a type, the caller (START_NEW_TERM_DEF) will assign a negative
2 ! terminal number.
2 !-
2
2     IF .TYPE NEQ -1
2     THEN
3         BEGIN
3         AP [TPA$L_NUMBER] = .TYPE;
3         CALLG (.AP, SMG$$FLUSH_NUMERIC);        ! put type into data buffer
3
3         RETURN (SS$_NORMAL);
3         END
2     ELSE
2         RETURN (0);                           ! tell caller to assign negative number
2
1     END;                                      ! end of routine SEARCH_KNOWN_TYPES


                                  00FC 00000 SEARCH_KNOWN_TYPES:
                                                        .WORD   Save R2,R3,R4,R5,R6,R7                          ; 0792
                    57          01  CE 00002            MNEGL   #1, TYPE                                        ; 0827
                    50      28  AC  D0 00005            MOVL    40(AP), TXT_RAB                                 ; 0845
        51      58  AC      5C  AC  C3 00009            SUBL3   92(AP), 88(AP), R1                              ; 0848
        56          51      24  A0  C1 0000F            ADDL3   36(TXT_RAB), R1, PTR_UPCASED_NAME               ;
                    55 00000000G 00  9E 00014           MOVAB   TERM$_NUM-1, R5                                 ; 0853
                    54          01  CE 0001B            MNEGL   #1, INDEX                                       ; 0857
                                1E  11 0001E            BRB     2$                                              ;
                    50 00000000G0044  D0 00020 1$:      MOVL    TERM$_NAME[INDEX], R0                           ; 0856
    04  B0          66      54  AC  29 00028            CMPC3   84(AP), (PTR_UPCASED_NAME), @4(R0)              ; 0857
                                0E  12 0002E            BNEQ    2$                                              ;
        50          54          1C  C5 00030            MULL3   #28, INDEX, R0                                  ; 0861
                    57 00000000G0040  9A 00034          MOVZBL  TERM$_TABLE+1[R0], TYPE                         ;
                                04  11 0003C            BRB     3$                                              ; 0860
        DE          54          55  F3 0003E 2$:        AOBLEQ  R5, INDEX, 1$                                   ; 0853
```

```
                    FFFFFFFF   8F          57  D1 00042 3$:    CMPL    TYPE, #-1                          ; 0872
                                           0D  13 00049        BEQL    4$                                 ;
                          1C   AC          57  D0 0004B        MOVL    TYPE, 28(AP)                       ; 0875
                      0000V    CF          6C  FA 0004F        CALLG   (AP), SMG$$FLUSH_NUMERIC           ; 0876
                           50             01  D0 00054        MOVL    #1, R0                             ; 0881
                                              04 00057        RET                                        ;
                                          50  D4 00058 4$:    CLRL    R0                                 ;
                                              04 0005A        RET                                        ; 0883

; Routine Size:  91 bytes,    Routine Base:  _SMG$CODE + 01FA
```

```
;  817     0884  1 %SBTTL 'START_NEW_TERM_DEF - Start of a term def, init fields'
;  818     0885  1 ROUTINE START_NEW_TERM_DEF =
;  819     0886  1
;  820     0887  1 !++
;  821     0888  1 ! FUNCTIONAL DESCRIPTION:
;  822     0889  1 !
;  823     0890  1 !       We are at the beginning of a terminal definition (ie. we just
;  824     0891  1 !       encountered a NAME field).  We need to begin a new block, make
;  825     0892  1 !       an entry into the terminal definition index, and initialize
;  826     0893  1 !       some fields preceeding the capability pointers.
;  827     0894  1 !
;  828     0895  1 ! CALLING SEQUENCE:
;  829     0896  1 !
;  830     0897  1 !       status = START_NEW_TERM_DEF ();
;  831     0898  1 !
;  832     0899  1 ! FORMAL PARAMETERS:
;  833     0900  1 !
;  834     0901  1 !       NONE
;  835     0902  1 !
;  836     0903  1 ! IMPLICIT INPUTS:
;  837     0904  1 !
;  838     0905  1 !       AP      Points to TPARSE parameter block
;  839     0906  1 !
;  840     0907  1 ! IMPLICIT OUTPUTS:
;  841     0908  1 !
;  842     0909  1 !       NONE
;  843     0910  1 !
;  844     0911  1 ! COMPLETION STATUS:
;  845     0912  1 !
;  846     0913  1 !        SS$_NORMAL
;  847     0914  1 !
;  848     0915  1 ! SIDE EFFECTS:
;  849     0916  1 !
;  850     0917  1 !--
;  851     0918  1
;  852     0919  2     BEGIN
;  853     0920  2     BUILTIN
;  854     0921  2         CALLG,
;  855     0922  2         AP;
;  856     0923  2     MAP
;  857     0924  2         AP : REF BLOCK [,BYTE];
;  858     0925  2     LOCAL
;  859     0926  2         TERM_INDEX : REF VECTOR [,BYTE], ! addr of terminal def index
;  860     0927  2         CUR_INDEX_BYTE;                  ! first available byte
;  861     0928  2
;  862     0929  2     TERM_INDEX = .AP [PARAM_A_TERM_INDEX];
;  863     0930  2     CUR_INDEX_BYTE = .AP [PARAM_L_TERM_INDEX_SIZE];
;  864     0931  2
;  865     0932  2 !+
;  866     0933  2 ! If there is a current terminal definition, then there must be two
;  867     0934  2 ! NAMEs with no intervening END.
;  868     0935  2 !-
;  869     0936  2     IF .AP [PARAM_L_CUR_TERM_DEF] NEQ 0
;  870     0937  2     THEN
;  871     0938  2         SIGNAL_STOP (SMG$_ERRLIN,
;  872     0939  2                      1, .SMG$$CURRENT_LINE,
;  873     0940  2                      SMG$_MISENDSTA);
```

```
;  874          0941  2
;  875          0942  2 !+
;  876          0943  2 ! Make an entry in the terminal definition index.  Since each
;  877          0944  2 ! terminal definition begins on a block boundary, all we need to
;  878          0945  2 ! store is the terminal name (a counted string) and the block number.
;  879          0946  2 !-
;  880          0947  2
;  881          0948  2     TERM_INDEX [.CUR_INDEX_BYTE] = .AP [PARAM_L_SAVED_TOKENCNT];
;  882          0949  2     CUR_INDEX_BYTE = .CUR_INDEX_BYTE + 1;
;  883          0950  2
;  884          0951  2     CH$MOVE (.AP [PARAM_L_SAVED_TOKENCNT],
;  885          0952  2              .AP [PARAM_L_SAVED_TOKENSTR],
;  886          0953  2              TERM_INDEX [.CUR_INDEX_BYTE]);
;  887          0954  2     CUR_INDEX_BYTE = .CUR_INDEX_BYTE + .AP [PARAM_L_SAVED_TOKENCNT];
;  888          0955  2                                 ! copy counted name string into index
;  889          0956  2     TERM_INDEX [.CUR_INDEX_BYTE] = .SMG$$CURRENT_DEF_BLOCK;
;  890          0957  2                                 ! store block number
;  891          0958  2
;  892          0959  2     AP [PARAM_L_TERM_INDEX_SIZE] = .CUR_INDEX_BYTE + 1;
;  893          0960  2                                 ! incr size of terminal index
;  894          0961  2
;  895          0962  2 !+
;  896          0963  2 ! Update pointer into the current terminal definition.
;  897          0964  2 !-
;  898          0965  2
;  899          0966  2     AP [PARAM_L_CUR_TERM_DEF] = .AP [PARAM_A_CAP_PTRS];
;  900          0967  2                                 ! point to beginning of buffer
;  901          0968  2
;  902          0969  2 !+
;  903          0970  2 ! Ready to fill in control fields which preceed the capability offsets.
;  904          0971  2 !-
;  905          0972  2
;  906          0973  2     BEGIN
;  907          0974  2     LOCAL
;  908          0975  2         CUR_TERM_DEF : REF VECTOR [,WORD],
;  909          0976  2         SAVED_CAP_NUMBER,
;  910          0977  2         TYPE_FOUND;
;  911          0978  2
;  912          0979  2     CUR_TERM_DEF = .AP [PARAM_L_CUR_TERM_DEF];
;  913          0980  2
;  914          0981  2     CUR_TERM_DEF [SMG$K_TERM_DEF_ID_OFFSET] = SMG$K_TERM_DEF_ID;
;  915          0982  2                                 ! identify as a terminal definition
;  916          0983  2     !+
;  917          0984  2     ! If this is a foreign terminal, then VMS will not assign any
;  918          0985  2     ! DT$_xx number to it.  All the negative numbers are unused by
;  919          0986  2     ! the terminal driver, so assigning a unique negative number to
;  920          0987  2     ! the terminal identifies it as one that is defined in TERMTABLE.
;  921          0988  2     ! The DCL commands SET and SHOW will make use of this number.
;  922          0989  2     !-
;  923          0990  2     SAVED_CAP_NUMBER = .AP [PARAM_L_CUR_CAP_NUMBER];
;  924          0991  2     AP [PARAM_L_CUR_CAP_NUMBER] = SMG$K_VMS_TERMINAL_NUMBER;
;  925          0992  3                                 ! replace NAME with type for writing data
;  926          0993  4     IF NOT (TYPE_FOUND = CALLG (.AP, SEARCH_KNOWN_TYPES))
;  927          0994  3     THEN
;  928          0995  4         BEGIN
;  929          0996  4         AP [TPA$L_NUMBER] = .SMG$$NEXT_NEGATIVE_NUMBER;
;  930          0997  4         CALLG (.AP, SMG$$FLUSH_NUMERIC);
```

```
;  931        0998  4          SMG$$NEXT_NEGATIVE_NUMBER = .SMG$$NEXT_NEGATIVE_NUMBER - 1;
;  932        0999  3          END;
;  933        1000  3
;  934        1001  3      AP [PARAM_L_CUR_CAP_NUMBER] = .SAVED_CAP_NUMBER;
;  935        1002  3                                          ! restore NAME
;  936        1003  2      END;
;  937        1004  2
;  938        1005  2      RETURN (SS$_NORMAL);
;  939        1006  2
;  940        1007  1      END;                                ! end of routine START_NEW_TERM_DEF



                                                  01FC 00000 START_NEW_TERM_DEF:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8                          ; 0885
                        58 00000000'  EF  9E 00002                MOVAB   SMG$$NEXT_NEGATIVE_NUMBER, R8                      ;
                        56        44  AC  D0 00009                MOVL    68(AP), TERM_INDEX                                 ; 0929
                        57        40  AC  D0 0000D                MOVL    64(AP), CUR_INDEX_BYTE                             ; 0930
                                  48  AC  D5 00011                TSTL    72(AP)                                             ; 0936
                                      18  13 00014                BEQL    1$                                                 ;
                           00000000G  00  9F 00016                PUSHAB  SMG$_MISENDSTA                                     ; 0938
                                  04  A8  DD 0001C                PUSHL   SMG$$CURRENT_LINE                                  ; 0939
                                      01  DD 0001F                PUSHL   #1                                                 ; 0938
                           00000000G  00  9F 00021                PUSHAB  SMG$_ERRLIN                                        ;
                00000000G  00         04  FB 00027                CALLS   #4, LIB$STOP                                       ;
                      8746        54  AC  90 0002E 1$:            MOVB    84(AP), (CUR_INDEX_BYTE)+[TERM_INDEX]              ; 0948
          6746        58  BC        54  AC  28 00033                MOVC3   84(AP), @88(AP), (CUR_INDEX_BYTE)-                 ; 0953
                                                                          [TERM_INDEX]                                       ;
                        57        54  AC  C0 0003A                ADDL2   84(AP), CUR_INDEX_BYTE                             ; 0954
                      6746        0C  A8  90 0C03E                MOVB    SMG$$CURRENT_DEF_BLOCK, (CUR_INDEX_BYTE)-          ; 0956
                                                                          [TERM_INDEX]                                       ;
                  40  AC        01  A7  9E 00043                MOVAB   1(R7), 64(AP)                                      ; 0959
                  48  AC        34  AC  D0 00048                MOVL    52(AP), 72(AP)                                     ; 0966
                      50        48  AC  D0 0004D                MOVL    72(AP), CUR_TERM_DEF                               ; 0979
                      60        89  8F  9B 00051                MOVZBW  #137, (CUR_TERM_DEF)                               ; 0981
                      52        50  AC  D0 00055                MOVL    80(AP), SAVED_CAP_NUMBER                           ; 0990
                  50  AC        E3  8F  9A 00059                MOVZBL  #227, 80(AP)                                       ; 0991
                FF42  CF            6C  FA 0005E                CALLG   (AP), SEARCH_KNOWN_TYPES                           ; 0993
                      51            50  D0 00063                MOVL    R0, TYPE_FOUND                                     ;
                      0B            50  E8 00066                BLBS    R0, 2$                                             ;
                  1C  AC            68  D0 00069                MOVL    SMG$$NEXT_NEGATIVE_NUMBER, 28(AP)                  ; 0996
                0000V CF            6C  FA 0006D                CALLG   (AP), SMG$$FLUSH_NUMERIC                           ; 0997
                                    68  D7 00072                DECL    SMG$$NEXT_NEGATIVE_NUMBER                          ; 0998
                  50  AC            52  D0 00074 2$:            MOVL    SAVED_CAP_NUMBER, 80(AP)                           ; 1001
                      50            01  D0 00078                MOVL    #1, R0                                             ; 1005
                                        04 0007B                RET                                                        ; 1007

; Routine Size:  124 bytes,    Routine Base:  _SMG$CODE + 0255
```

```
1 %SBTTL 'UNRECOGNIZED_STATEMENT - signal UNRECSTA error'
1 ROUTINE UNRECOGNIZED_STATEMENT =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       We just found something that isn't one of the statements we know -
1 !       complain.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       UNRECOGNIZED_STATEMENT ();
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     SIGNAL_STOP (SMG$_ERRLIN,
2                  1,
2                  .SMG$$CURRENT_LINE,
2                 SMG$_UNRECSTA)
2
1     END;                                  ! end of routine UNRECOGNIZED_STATEMENT


                                 0000 00000 UNRECOGNIZED_STATEMENT:
                                                    .WORD   Save nothing                    ; 1009
                  00000000G  00  9F 00002           PUSHAB  SMG$_UNRECSTA                   ; 1047
                  00000000'  EF  DD 00008           PUSHL   SMG$$CURRENT_LINE               ; 1049
                             01  DD 0000E           PUSHL   #1                              ; 1047
                  00000000G  00  9F 00010           PUSHAB  SMG$_ERRLIN                     ;
      00000000G  00          04  FB 00016           CALLS   #4, [IB$STOP                    ;
```

```
                                                    04 0001D        RET                                                  ; 1052

; Routine Size:  30 bytes,    Routine Base:  _SMG$CODE + 02D1
```

```
:  988          1053  1 %SBTTL 'ZERO_CUR_TERM_DEF - zero the current terminal definition pointer'
:  989          1054  1 ROUTINE ZERO_CUR_TERM_DEF =
:  990          1055  1
:  991          1056  1 !++
:  992          1057  1 ! FUNCTIONAL DESCRIPTION:
:  993          1058  1 !
:  994          1059  1 !       We just encountered an END statement - mark that there is no
:  995          1060  1 !       current terminal definition so that capabilities will not be
:  996          1061  1 !       accepted before a new NAME statement is encountered.
:  997          1062  1 !
:  998          1063  1 ! CALLING SEQUENCE:
:  999          1064  1 !
: 1000          1065  1 !       status = ZERO_CUR_TERM_DEF ();
: 1001          1066  1 !
: 1002          1067  1 ! FORMAL PARAMETERS:
: 1003          1068  1 !
: 1004          1069  1 !       NONE
: 1005          1070  1 !
: 1006          1071  1 ! IMPLICIT INPUTS:
: 1007          1072  1 !
: 1008          1073  1 !       AP      Points to TPARSE parameter block
: 1009          1074  1 !
: 1010          1075  1 ! IMPLICIT OUTPUTS:
: 1011          1076  1 !
: 1012          1077  1 !       NONE
: 1013          1078  1 !
: 1014          1079  1 ! COMPLETION STATUS:
: 1015          1080  1 !
: 1016          1081  1 !        SS$_NORMAL
: 1017          1082  1 !
: 1018          1083  1 ! SIDE EFFECTS:
: 1019          1084  1 !
: 1020          1085  1 !--
: 1021          1086  1
: 1022          1087  2     BEGIN
: 1023          1088  2     BUILTIN
: 1024          1089  2         CALLG,
: 1025          1090  2         AP;
: 1026          1091  2     MAP
: 1027          1092  2         AP : REF BLOCK [,BYTE];
: 1028          1093  2     LOCAL
: 1029          1094  2         BINARY_RAB : REF $RAB_DECL;     ! point to RAB
: 1030          1095  2
: 1031          1096  2 !+
: 1032          1097  2 ! Move the previous definition to TERMTABLE.EXE.  First write
: 1033          1098  2 ! out the capability pointers.
: 1034          1099  2 !-
: 1035          1100  2
: 1036          1101  2     BINARY_RAB = .AP [PARAM_A_BINARY_RAB];
: 1037          1102  2
: 1038          1103  2     BINARY_RAB [RAB$W_RSZ] = SMG$K_CAP_PTRS_SIZE;
: 1039          1104  2     BINARY_RAB [RAB$L_RBF] = .AP [PARAM_A_CAP_PTRS];
: 1040          1105  2                                     ! set up size & buffer addr
: 1041          1106  2     BINARY_RAB [RAB$L_BKT] = .SMG$$CURRENT_DEF_BLOCK;
: 1042          1107  2                                     ! set virtual block number
: 1043          1108  2
: 1044          1109  2     $WRITE (RAB = .BINARY_RAB);
```

```
; 1045          1110  2
; 1046          1111  2 !+
; 1047          1112  2 ! See if any data needs to be written out.
; 1048          1113  2 !-
; 1049          1114  2
; 1050          1115  2     CALLG (.AP, SMG$$WRITE_DATA);
; 1051          1116  2
; 1052          1117  2 !+
; 1053          1118  2 ! Update the next available block number.
; 1054          1119  2 !-
; 1055          1120  2
; 1056          1121  2     SMG$$CURRENT_DEF_BLOCK = .SMG$$CURRENT_BLOCK;
; 1057          1122  2     SMG$$CURRENT_BLOCK = .SMG$$CURRENT_DEF_BLOCK + (SMG$K_CAP_PTRS_SIZE/512);
; 1058          1123  2                                         ! next data skips over pointers
; 1059          1124  2
; 1060          1125  2 !+
; 1061          1126  2 ! Mark that there is no current definition.  Init some things that will be
; 1062          1127  2 ! reused by the next definition.
; 1063          1128  2 !-
; 1064          1129  2
; 1065          1130  2     AP [PARAM_L_CUR_TERM_DEF] = 0;
; 1066          1131  2
; 1067          1132  2     CH$FILL (0, SMG$K_CAP_PTRS_SIZE, .AP [PARAM_A_CAP_PTRS]);
; 1068          1133  2                                         ! zero data offsets
; 1069          1134  2     SMG$$DATA_OFFSET = SMG$K_CAP_PTRS_SIZE;
; 1070          1135  2                                         ! start with 1st offset again
; 1071          1136  2     CH$FILL (0, SMG$K_CAP_DATA_SIZE, .AP [PARAM_A_CAP_DATA]);
; 1072          1137  2                                         ! zero the data buffer
; 1073          1138  2     AP [PARAM_L_CUR_DATA_BYTE] = .AP [PARAM_A_CAP_DATA];
; 1074          1139  2                                         ! current data byte is 1st byte
; 1075          1140  2
; 1076          1141  2     RETURN (SS$_NORMAL);
; 1077          1142  1     END;                                ! end of routine ZERO_CUR_TERM_DEF


                                                                .EXTRN  SYS$WRITE

                                                  007C 00000 ZERO_CUR_TERM_DEF:
                                                                .WORD   Save R2,R3,R4,R5,R6                                  ; 1054
                                 56 00000000'  EF  9E 00002     MOVAB   SMG$$CURRENT_DEF_BLOCK, R6                           ;
                                 50       30   AC  D0 00009     MOVL    48(AP), BINARY_RAB                                   ; 1101
                           22    A0     0600   8F  B0 0000D     MOVW    #1536, 34(BINARY_RAB)                                ; 1103
                           28    A0       34   AC  D0 00013     MOVL    52(AP), 40(BINARY_RAB)                               ; 1104
                           38    A0            66  D0 00018     MOVL    SMG$$CURRENT_DEF_BLOCK, 56(BINARY_RAB)               ; 1106
                                               50  DD 0001C     PUSHL   BINARY_RAB                                           ; 1109
                     00000000G   00            01  FB 0001E     CALLS   #1, SYS$WRITE                                        ;
                         0000V   CF            6C  FA 00025     CALLG   (AP), SMG$$WRITE_DATA                                ; 1115
                                 66       FC   A6  D0 0002A     MOVL    SMG$$CURRENT_BLOCK, SMG$$CURRENT_DEF_BLOCK           ; 1121
                FC    A6         66            03  C1 0002E     ADDL3   #3, SMG$$CURRENT_DEF_BLOCK, -                        ; 1122
                                                                         SMG$$CURRENT_BLOCK                                   ;
                                          48   AC  D4 00033     CLRL    72(AP)                                               ; 1130
   0600    8F          00         6E            00  2C 00036     MOVC5   #0, (SP), #0, #1536, @52(AP)                         ; 1132
                                          34   BC     0003D                                                                    ;
                           04    A6     0600   8F  3C 0003F     MOVZWL  #1536, SMG$$DATA_OFFSET                              ; 1134
   1400    8F          00         6E            00  2C 00045     MOVC5   #0, (SP), #0, #5120, @56(AP)                         ; 1136
                                          38   BC     0004C                                                                    ;
```

```
                    4C   AC      38   AC  D0 0004E        MOVL    56(AP), 76(AP)                    ; 1138
                         50           01  D0 00053        MOVL    #1, R0                            ; 1141
                                          04 00056        RET                                       ; 1142

; Routine Size:  87 bytes,    Routine Base:  _SMG$CODE + 02EF
```

```
1 %SBTTL 'SMG$$BLANKS_OFF - Turn OFF bit to skip over blanks'
1 GLOBAL ROUTINE SMG$$BLANKS_OFF =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Turns OFF a bit in the TPARSE parameter block which controls
1 !       whether blanks are significant.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       status = SMG$$BLANKS_OFF ();
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     AP [TPA$V_BLANKS] = 0;
2
2     RETURN (SS$_NORMAL);
2
1     END;                                    ! end of routine SMG$$BLANKS_OFF


                                  0000 00000        .ENTRY  SMG$$BLANKS_OFF, Save nothing          ; 1144
          04    AC            01   8A 00002        BICB2   #1, 4(AP)                              ; 1182
                50            01   D0 00006        MOVL    #1, R0                                 ; 1184
                                   04 00009        RET                                            ; 1186

; Routine Size:  10 bytes,    Routine Base:  _SMG$CODE + 0346
```

```
1 %SBTTL 'SMG$$BLANKS_ON - Set bit to look at blanks'
1 GLOBAL ROUTINE SMG$$BLANKS_ON =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Sets a bit in the TPARSE parameter block that causes blanks
1 !       and tabs to be processed.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       status = SMG$$BLANKS_ON ();
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     AP [TPA$V_BLANKS] = 1;
2
2     RETURN (SS$_NORMAL);
2
1     END;                                    ! end of routine SMG$$BLANKS_ON


                                  0000 00000        .ENTRY  SMG$$BLANKS_ON, Save nothing                ; 1188
          04    AC              01   88 00002        BISB2   #1, 4(AP)                                   ; 1226
                50              01   D0 00006        MOVL    #1, R0                                      ; 1228
                                     04 00009        RET                                                 ; 1230

; Routine Size:  10 bytes,    Routine Base:  _SMG$CODE + 0350
```

```
;  1169          1231  1 %SBTTL 'SMG$$COPY_CAP - Copy a string capability into TERMTABLE.EXE'
;  1170          1232  1 GLOBAL ROUTINE SMG$$COPY_CAP =
;  1171          1233  1
;  1172          1234  1 !++
;  1173          1235  1 ! FUNCTIONAL DESCRIPTION:
;  1174          1236  1 !
;  1175          1237  1 !       Copies a string capability into TERMTABLE.EXE.  It uses the
;  1176          1238  1 !       current capability number to determine where to copy the string.
;  1177          1239  1 !
;  1178          1240  1 ! CALLING SEQUENCE:
;  1179          1241  1 !
;  1180          1242  1 !       status = SMG$$COPY_CAP ()
;  1181          1243  1 !
;  1182          1244  1 ! FORMAL PARAMETERS:
;  1183          1245  1 !
;  1184          1246  1 !       NONE
;  1185          1247  1 !
;  1186          1248  1 ! IMPLICIT INPUTS:
;  1187          1249  1 !
;  1188          1250  1 !       AP      Points to TPARSE parameter block
;  1189          1251  1 !
;  1190          1252  1 ! IMPLICIT OUTPUTS:
;  1191          1253  1 !
;  1192          1254  1 !       NONE
;  1193          1255  1 !
;  1194          1256  1 ! COMPLETION STATUS:
;  1195          1257  1 !
;  1196          1258  1 !        SS$_NORMAL
;  1197          1259  1 !
;  1198          1260  1 ! SIDE EFFECTS:
;  1199          1261  1 !
;  1200          1262  1 !--
;  1201          1263  1
;  1202          1264  2     BEGIN
;  1203          1265  2     BUILTIN
;  1204          1266  2         CALLG,
;  1205          1267  2         AP;
;  1206          1268  2     MAP
;  1207          1269  2         AP : REF BLOCK [,BYTE];
;  1208          1270  2
;  1209          1271  2 !+
;  1210          1272  2 ! If this is the name capability, then we are starting a new terminal
;  1211          1273  2 ! definition.  Start a new block and set up some pointers.
;  1212          1274  2 !-
;  1213          1275  2
;  1214          1276  2     IF .AP [PARAM_L_CUR_CAP_NUMBER] EQL SMG$K_NAME
;  1215          1277  2     THEN
;  1216          1278  2         CALLG (.AP, START_NEW_TERM_DEF);
;  1217          1279  2
;  1218          1280  2 !+
;  1219          1281  2 ! If this is not the NAME capability and we have no pointers set up
;  1220          1282  2 ! for the terminal definition, then NAME was not the first capability
;  1221          1283  2 ! in the definition.  Complain.
;  1222          1284  2 !-
;  1223          1285  2
;  1224          1286  3     BEGIN
;  1225          1287  3     BIND
```

```
3          CAP_PTRS = .AP [PARAM_L_CUR_TERM_DEF] : VECTOR [,WORD];
3
3      IF CAP_PTRS EQL 0
3      THEN
3          SIGNAL_STOP (SMG$_MISTERNAM);
3
3  !+
3  ! Move the capability data.  The byte count is in the first byte and
3  ! the actual data follows.
3  !
3  ! Part of the string may already be copied - append in this part so
3  ! as not to overwrite it.  (This can happen if escape or control are
3  ! part of the sequence.)
3  !-
3
3      IF .AP [PARAM_L_SAVED_TOKENCNT] NEQ 0
3      THEN
3          CALLG (.AP, SMG$$FLUSH_SAVED_BUFFER);
3                                                  ! copy string into data area
3
2      END;                                         ! end of BINDs scope
2
2      AP [TPA$V_BLANKS] = 0;                       ! out of string cap, skip over blanks
2
2      RETURN (SS$_NORMAL);
1      END;                                         ! end of routine SMG$$COPY_CAP


                                  0000 00000          .ENTRY  SMG$$COPY_CAP, Save nothing                  ; 1232
0000022B   8F          50    AC   D1 00002          CMPL    80(AP), #555                                 ; 1276
                             05   12 0000A          BNEQ    1$                                           ;
    FEEA   CF                6C   FA 0000C          CALLG   (AP), START_NEW_TERM_DEF                     ; 1278
                       48    AC   D5 00011 1$:      TSTL    72(AP)                                       ; 1290
                             0D   12 00014          BNEQ    2$                                           ;
                 00000000G   00   9F 00016          PUSHAB  SMG$_MISTERNAM                               ; 1292
00000000G  00                01   FB 0001C          CALLS   #1, LIB$STOP                                 ;
                       54    AC   D5 00023 2$:      TSTL    84(AP)                                       ; 1303
                             05   13 00026          BEQL    3$                                           ;
    0000V  CF                6C   FA 00028          CALLG   (AP), SMG$$FLUSH_SAVED_BUFFER                ; 1305
      04   AC                01   8A 0002D 3$:      BICB2   #1, 4(AP)                                    ; 1310
           50                01   D0 00031          MOVL    #1, R0                                       ; 1312
                                  04 00034          RET                                                  ; 1313

; Routine Size:  53 bytes,    Routine Base:  _SMG$CODE + 035A
```

```
;  1253          1314  1 %SBTTL 'SMG$$FLUSH_ARITHMETIC - flush arithmetic data to TERMTABLE'
;  1254          1315  1 GLOBAL ROUTINE SMG$$FLUSH_ARITHMETIC =
;  1255          1316  1
;  1256          1317  1 !++
;  1257          1318  1 ! FUNCTIONAL DESCRIPTION:
;  1258          1319  1 !
;  1259          1320  1 !       Because expressions don't follow the rules for numeric or string
;  1260          1321  1 !       data, this is a special routine to copy expression 'tokens' into
;  1261          1322  1 !       the data area.  Usually we want to just write to the current data
;  1262          1323  1 !       byte without regard for a size or type.
;  1263          1324  1 !
;  1264          1325  1 ! CALLING SEQUENCE:
;  1265          1326  1 !
;  1266          1327  1 !       status = SMG$$FLUSH_ARITHMETIC ()
;  1267          1328  1 !
;  1268          1329  1 ! FORMAL PARAMETERS:
;  1269          1330  1 !
;  1270          1331  1 !       NONE
;  1271          1332  1 !
;  1272          1333  1 ! IMPLICIT INPUTS:
;  1273          1334  1 !
;  1274          1335  1 !       AP      Points to TPARSE parameter block
;  1275          1336  1 !
;  1276          1337  1 ! IMPLICIT OUTPUTS:
;  1277          1338  1 !
;  1278          1339  1 !       NONE
;  1279          1340  1 !
;  1280          1341  1 ! COMPLETION STATUS:
;  1281          1342  1 !
;  1282          1343  1 !        SS$_NORMAL
;  1283          1344  1 !
;  1284          1345  1 ! SIDE EFFECTS:
;  1285          1346  1 !
;  1286          1347  1 !--
;  1287          1348  1
;  1288          1349  2     BEGIN
;  1289          1350  2     BUILTIN
;  1290          1351  2         CALLG,
;  1291          1352  2         AP;
;  1292          1353  2     MAP
;  1293          1354  2         AP : REF BLOCK [,BYTE];
;  1294          1355  2     LOCAL
;  1295          1356  2         START_COPY : REF VECTOR [,BYTE,SIGNED];
;  1296          1357  2
;  1297          1358  2     $CHECK_BUFFER_OVERFLOW (.AP [PARAM_L_SAVED_TOKENCNT]);
;  1298          1359  2                                 ! make sure this will fit in the buffer
;  1299          1360  2     START_COPY = .AP [PARAM_L_CUR_DATA_BYTE];
;  1300          1361  2
;  1301          1362  2     CH$MOVE (.AP [PARAM_L_SAVED_TOKENCNT],
;  1302          1363  2              .AP [PARAM_L_SAVED_TOKENSTR],
;  1303          1364  2              START_COPY [0]);
;  1304          1365  2                                 ! move the data
;  1305          1366  2     AP [PARAM_L_CUR_DATA_BYTE] = .AP [PARAM_L_CUR_DATA_BYTE] +
;  1306          1367  2                                  .AP [PARAM_L_SAVED_TOKENCNT];
;  1307          1368  2                                 ! update next available byte
;  1308          1369  2     SMG$$DATA_OFFSET = .SMG$$DATA_OFFSET + .AP [PARAM_L_SAVED_TOKENCNT];
;  1309          1370  2                                 ! update next available data offset
```

```
; 1310          1371  2     $UPDATE_ARITH_STRING_SIZE (.AP [PARAM_L_SAVED_TOKENCNT]);
; 1311          1372  2                                                  ! update size of expression
; 1312          1373  2     AP [PARAM_L_SAVED_TOKENCNT] = 0;
; 1313          1374  2     AP [PARAM_L_SAVED_TOKENSTR] = 0;
; 1314          1375  2                                                  ! re-init
; 1315          1376  2     RETURN (SS$_NORMAL);
; 1316          1377  2
; 1317          1378  1     END;                                         ! end of routine SMG$$FLUSH_ARITHMETIC



                                                  003C 00000          .ENTRY  SMG$$FLUSH_ARITHMETIC, Save R2,R3,R4,R5      ; 1315
                50        38   AC        4C   AC   C3 00002          SUBL3   76(AP), 56(AP), R0                           ; 1358
                                50      1400   C0   9E 00008          MOVAB   5120(R0), REMAINING_BYTES                    ;
                                50        54   AC   D1 0000D          CMPL    84(AP), REMAINING_BYTES                      ;
                                                05   15 00011          BLEQ    1$                                           ;
                      0000V   CF              6C   FA 00013          CALLG   (AP), SMG$$WRITE_DATA                        ;
                                50        4C   AC   D0 00018 1$:      MOVL    76(AP), START_COPY                           ; 1360
                60        58   BC        54   AC   28 0001C          MOVC3   84(AP), @88(AP), (START_COPY)                ; 1364
                          4C   AC        54   AC   C0 00022          ADDL2   84(AP), 76(AP)                               ; 1367
                00000000'   EF        54   AC   C0 00027          ADDL2   84(AP), SMG$$DATA_OFFSET                     ; 1369
                                50        50   AC   D0 0002F          MOVL    80(AP), R0                                   ; 1371
                                50        48 BC40   3C 00033          MOVZWL  @72(AP)[R0], CAP_SIZE                        ;
                                50        48   AC   C0 00038          ADDL2   72(AP), CAP_SIZE                             ;
                                60        54   AC   82 0003C          SUBB2   84(AP), (CAP_SIZE)                           ;
                                          54   AC   7C 00040          CLRQ    84(AP)                                       ; 1373
                                50              01   D0 00043          MOVL    #1, R0                                       ; 1376
                                                     04 00046          RET                                                  ; 1378

; Routine Size:  71 bytes,    Routine Base:  _SMG$CODE + 038F
```

```
;  1319          1379  1 %SBTTL 'SMG$$FLUSH_NUMERIC - flush number to data area'
;  1320          1380  1 GLOBAL ROUTINE SMG$$FLUSH_NUMERIC =
;  1321          1381  1
;  1322          1382  1 !++
;  1323          1383  1 ! FUNCTIONAL DESCRIPTION:
;  1324          1384  1 !
;  1325          1385  1 !       Copies a boolean or numeric value into TERMTABLE.EXE.  It uses the
;  1326          1386  1 !       current capability number to determine where to copy it.
;  1327          1387  1 !
;  1328          1388  1 ! CALLING SEQUENCE:
;  1329          1389  1 !
;  1330          1390  1 !       status = SMG$$FLUSH_NUMERIC ()
;  1331          1391  1 !
;  1332          1392  1 ! FORMAL PARAMETERS:
;  1333          1393  1 !
;  1334          1394  1 !       NONE
;  1335          1395  1 !
;  1336          1396  1 ! IMPLICIT INPUTS:
;  1337          1397  1 !
;  1338          1398  1 !       AP      Points to TPARSE parameter block
;  1339          1399  1 !
;  1340          1400  1 ! IMPLICIT OUTPUTS:
;  1341          1401  1 !
;  1342          1402  1 !       NONE
;  1343          1403  1 !
;  1344          1404  1 ! COMPLETION STATUS:
;  1345          1405  1 !
;  1346          1406  1 !        SS$_NORMAL
;  1347          1407  1 !
;  1348          1408  1 ! SIDE EFFECTS:
;  1349          1409  1 !
;  1350          1410  1 !--
;  1351          1411  1
;  1352          1412  2     BEGIN
;  1353          1413  2     BUILTIN
;  1354          1414  2         CALLG,
;  1355          1415  2         AP;
;  1356          1416  2     MAP
;  1357          1417  2         AP : REF BLOCK [,BYTE];
;  1358          1418  2     LOCAL
;  1359          1419  2         CUR_TERM_DEF : REF VECTOR [,WORD],
;  1360          1420  2         START_CAP_STRING : REF VECTOR [,BYTE,SIGNED];
;  1361          1421  2
;  1362          1422  2     CUR_TERM_DEF = .AP [PARAM_L_CUR_TERM_DEF];
;  1363          1423  2     IF .CUR_TERM_DEF EQL 0
;  1364          1424  2     THEN
;  1365          1425  2         SIGNAL_STOP (SMG$_MISTERNAM);   ! complain if no current def
;  1366          1426  2
;  1367          1427  2     $CHECK_BUFFER_OVERFLOW (6);         ! make sure there's room in buffer
;  1368          1428  2
;  1369          1429  2     CUR_TERM_DEF [.AP [PARAM_L_CUR_CAP_NUMBER]] = .SMG$$DATA_OFFSET;
;  1370          1430  2                                         ! set cap ptr to data offset
;  1371          1431  2     SMG$$DATA_OFFSET = .SMG$$DATA_OFFSET + 6;
;  1372          1432  2                                         ! update to next offset
;  1373          1433  2     START_CAP_STRING = .AP [PARAM_L_CUR_DATA_BYTE];
;  1374          1434  2                                         ! point to next available data byte
;  1375          1435  2     START_CAP_STRING [0] = 4;           ! length
```

```
;  1376          1436  2     CH$MOVE (4, AP [TPA$L_NUMBER], START_CAP_STRING [2]);
;  1377          1437  2                                                 ! data
;  1378          1438  2     AP [PARAM_L_CUR_DATA_BYTE] = .AP [PARAM_L_CUR_DATA_BYTE] + 6;
;  1379          1439  2                                                 ! update next available data byte
;  1380          1440  2
;  1381          1441  2     RETURN (SS$_NORMAL);
;  1382          1442  2
;  1383          1443  1     END;                                        ! end of routine SMG$$FLUSH_NUMERIC


                                        000C 00000     .ENTRY  SMG$$FLUSH_NUMERIC, Save R2,R3                  ; 1380
                          53 00000000'  EF   9E 00002  MOVAB   SMG$$DATA_OFFSET, R3                            ;
                          52       48   AC   D0 00009  MOVL    72(AP), CUR_TERM_DEF                           ; 1422
                                        0D   12 0000D  BNEQ    1$                                             ; 1423
                             00000000G  00   9F 0000F  PUSHAB  SMG$_MISTERNAM                                 ; 1425
                 00000000G   00         01   FB 00015  CALLS   #1, LIB$STOP                                  ;
                50       38   AC       4C   AC   C3 0001C  1$:  SUBL3   76(AP), 56(AP), R0                    ; 1427
                          50     1400   C0   9E 00022  MOVAB   5120(R0), REMAINING_BYTES                      ;
                          06            50   D1 00027  CMPL    REMAINING_BYTES, #6                            ;
                                        05   18 0002A  BGEQ    2$                                             ;
                    0000V CF            6C   FA 0002C  CALLG   (AP), SMG$$WRITE_DATA                         ;
                          50       50   AC   D0 00031  2$:  MOVL    80(AP), R0                                 ; 1429
                        6240           63   B0 00035  MOVW    SMG$$DATA_OFFSET, (CUR_TERM_DEF)[R0]           ;
                          63            06   C0 00039  ADDL2   #6, SMG$$DATA_OFFSET                          ; 1431
                          50       4C   AC   D0 0003C  MOVL    76(AP), START_CAP_STRING                       ; 1433
                          60            04   90 00040  MOVB    #4, (START_CAP_STRING)                         ; 1435
                       02 A0       1C   AC   D0 00043  MOVL    28(AP), 2(START_CAP_STRING)                    ; 1436
                       4C AC            06   C0 00048  ADDL2   #6, 76(AP)                                     ; 1438
                          50            01   D0 0004C  MOVL    #1, R0                                         ; 1441
                                        04   0004F     RET                                                    ; 1443

; Routine Size:  80 bytes,    Routine Base:  _SMG$CODE + 03D6
```

```
;  1385          1444  1 %SBTTL 'SMG$$FLUSH_SAVED_BUFFER - flush saved token to data buffer'
;  1386          1445  1 GLOBAL ROUTINE SMG$$FLUSH_SAVED_BUFFER =
;  1387          1446  1
;  1388          1447  1 !++
;  1389          1448  1 ! FUNCTIONAL DESCRIPTION:
;  1390          1449  1 !
;  1391          1450  1 ! Append the saved token into the data area for the current capability.
;  1392          1451  1 ! We must append the data rather than just copying it because tokens are
;  1393          1452  1 ! inserted as they are discovered, so this may not be the first.
;  1394          1453  1 !
;  1395          1454  1 ! CALLING SEQUENCE:
;  1396          1455  1 !
;  1397          1456  1 !
;  1398          1457  1 !       status = SMG$$FLUSH_SAVED_BUFFER ()
;  1399          1458  1 !
;  1400          1459  1 ! FORMAL PARAMETERS:
;  1401          1460  1 !
;  1402          1461  1 !       NONE
;  1403          1462  1 !
;  1404          1463  1 ! IMPLICIT INPUTS:
;  1405          1464  1 !
;  1406          1465  1 !       AP      Points to TPARSE parameter block
;  1407          1466  1 !
;  1408          1467  1 ! IMPLICIT OUTPUTS:
;  1409          1468  1 !
;  1410          1469  1 !       NONE
;  1411          1470  1 !
;  1412          1471  1 ! COMPLETION STATUS:
;  1413          1472  1 !
;  1414          1473  1 !        SS$_NORMAL
;  1415          1474  1 !
;  1416          1475  1 ! SIDE EFFECTS:
;  1417          1476  1 !
;  1418          1477  1 !--
;  1419          1478  1
;  1420          1479  2     BEGIN
;  1421          1480  2     BUILTIN
;  1422          1481  2         CALLG,
;  1423          1482  2         AP;
;  1424          1483  2     MAP
;  1425          1484  2         AP : REF BLOCK [,BYTE];
;  1426          1485  2
;  1427          1486  2     IF .AP [PARAM_L_SAVED_TOKENCNT] EQL 0
;  1428          1487  2     THEN
;  1429          1488  3         RETURN (SS$_NORMAL)                 ! no string to copy
;  1430          1489  2     ELSE
;  1431          1490  3         BEGIN                               ! copy saved string
;  1432          1491  3         BIND
;  1433          1492  3             CAP_PTRS = .AP [PARAM_L_CUR_TERM_DEF] : VECTOR [,WORD];
;  1434          1493  3         LOCAL
;  1435          1494  3             FIRST : INITIAL (0),
;  1436          1495  3             START_CAP_STRING : REF VECTOR [,BYTE,SIGNED],
;  1437          1496  3             SIZE_TYPE : INITIAL (0);
;  1438          1497  3
;  1439          1498  3         $CHECK_BUFFER_OVERFLOW (.AP [PARAM_L_SAVED_TOKENCNT]);
;  1440          1499  3                                             ! make sure there's room in the buffer
;  1441          1500  3         IF .CAP_PTRS [.AP [PARAM_L_CUR_CAP_NUMBER]] NEQ 0
```

```
;  1442          1501  3          THEN
;  1443          1502  4              BEGIN
;  1444          1503  4              !+
;  1445          1504  4              ! Not start of the capability string.
;  1446          1505  4              !-
;  1447          1506  4              START_CAP_STRING = .AP [PARAM_L_CUR_TERM_DEF] +
;  1448          1507  4                                 .CAP_PTRS [.AP [PARAM_L_CUR_CAP_NUMBER]];
;  1449          1508  4              END
;  1450          1509  3          ELSE
;  1451          1510  3              !+
;  1452          1511  3              ! First byte of capability string.  Set offset pointer.
;  1453          1512  3              !-
;  1454          1513  4              BEGIN
;  1455          1514  4              FIRST = 1;
;  1456          1515  4              START_CAP_STRING = .AP [PARAM_L_CUR_DATA_BYTE];
;  1457          1516  4              CAP_PTRS [.AP [PARAM_L_CUR_CAP_NUMBER]] = .SMG$$DATA_OFFSET;
;  1458          1517  3              END;
;  1459          1518  3
;  1460          1519  3          IF .START_CAP_STRING [1] NEQ SMG$K_ARITH_STRING
;  1461          1520  3          THEN
;  1462          1521  4              BEGIN
;  1463          1522  4              CH$MOVE (.AP [PARAM_L_SAVED_TOKENCNT],
;  1464          1523  4                       .AP [PARAM_L_SAVED_TOKENSTR],
;  1465          1524  4                       START_CAP_STRING [2] + ABS (.START_CAP_STRING [0]));
;  1466          1525  4              END
;  1467          1526  3          ELSE
;  1468          1527  4              BEGIN
;  1469          1528  4              BIND
;  1470          1529  5                  EMBEDDED_STRING = (START_CAP_STRING [2] +
;  1471          1530  5                                     ABS (.START_CAP_STRING [0]) - 2)
;  1472          1531  4                                  : VECTOR [,BYTE];
;  1473          1532  4                                  ! subtract arith size & type
;  1474          1533  4              EMBEDDED_STRING [0] = .AP [PARAM_L_SAVED_TOKENCNT];
;  1475          1534  4              EMBEDDED_STRING [1] = 0;              ! positive count indicates string
;  1476          1535  4              CH$MOVE (.AP [PARAM_L_SAVED_TOKENCNT],
;  1477          1536  4                       .AP [PARAM_L_SAVED_TOKENSTR],
;  1478          1537  4                       EMBEDDED_STRING [2]);
;  1479          1538  4              SIZE_TYPE = 2;
;  1480          1539  3              END;
;  1481          1540  3
;  1482        P 1541  3          $INCR_CAP_STRING_SIZE ((.AP [PARAM_L_SAVED_TOKENCNT] + .SIZE_TYPE),
;  1483          1542  3                                 START_CAP_STRING [0]);
;  1484          1543  3                                                  ! update # bytes in string
;  1485          1544  3
;  1486          1545  3          $INCR_CUR_DATA_BYTE ((.AP [PARAM_L_SAVED_TOKENCNT] + .SIZE_TYPE), .FIRST);
;  1487          1546  3                                                  ! macro to update next available
;  1488          1547  3                                                  ! data byte, data offset
;  1489          1548  3
;  1490          1549  3          AP [PARAM_L_SAVED_TOKENCNT] = 0;
;  1491          1550  3          AP [PARAM_L_SAVED_TOKENSTR] = 0;
;  1492          1551  3                                                  ! re-init to no saved string
;  1493          1552  3
;  1494          1553  3          IF .SMG$$MASK_ADR EQL SMG$K_FAO_STRING
;  1495          1554  3          THEN
;  1496          1555  4              BEGIN
;  1497          1556  4              !+
;  1498          1557  4              ! If this is an FAO string, then we should set the type
```

```
; 1499          1558  4              ! byte and negate the size byte.
; 1500          1559  4              !-
; 1501          1560  4              IF .START_CAP_STRING [0] GTR 0
; 1502          1561  4              THEN
; 1503          1562  4                  START_CAP_STRING [0] = -.START_CAP_STRING [0];
; 1504          1563  4                                          ! don't negate twice
; 1505          1564  4              START_CAP_STRING [1] = .SMG$$MASK_ADR;
; 1506          1565  4                                          ! set type byte
; 1507          1566  4              SMG$$MASK_ADR = 0;          ! reset mask
; 1508          1567  3              END;
; 1509          1568  3
; 1510          1569  2          END;                            ! copy saved string
; 1511          1570  2
; 1512          1571  2
; 1513          1572  2      RETURN (SS$_NORMAL);
; 1514          1573  2
; 1515          1574  1      END;                                ! end of routine SMG$$FLUSH_SAVED_BUFFER



                                        0FFC 00000          .ENTRY  SMG$$FLUSH_SAVED_BUFFER, Save R2,R3,R4,R5,-  ; 1445
                                                                    R6,R7,R8,R9,R10,R11                          ;
                      59        54    AC   9E 00002          MOVAB   84(AP), R9                                   ; 1486
                                      69   D5 00006          TSTL    (R9)                                         ;
                                      03   12 00008          BNEQ    1$                                           ;
                                    00E5   31 0000A          BRW     16$                                          ;
                      52        48    AC   D0 0000D 1$:      MOVL    72(AP), R2                                   ; 1492
                                      5A   7C 00011          CLRQ    SIZE_TYPE                                    ;
          50        38    AC        4C   AC   C3 00013          SUBL3   76(AP), 56(AP), R0                           ; 1498
                      50      1400    C0   9E 00019          MOVAB   5120(R0), REMAINING_BYTES                    ;
                      50              69   D1 0001E          CMPL    (R9), REMAINING_BYTES                        ;
                                      05   15 00021          BLEQ    2$                                           ;
                0000V CF              6C   FA 00023          CALLG   (AP), SMG$$WRITE_DATA                        ;
                      50        50    AC   D0 00028 2$:      MOVL    80(AP), R0                                   ; 1500
                                    6240   B5 0002C          TSTW    (R2)[R0]                                     ;
                                      0A   13 0002F          BEQL    3$                                           ;
                      56            6240   3C 00031          MOVZWL  (R2)[R0], START_CAP_STRING                   ; 1507
                      56        48    AC   C0 00035          ADDL2   72(AP), START_CAP_STRING                     ;
                                      0F   11 00039          BRB     4$                                           ; 1500
                      5B              01   D0 0003B 3$:      MOVL    #1, FIRST                                    ; 1514
                      56        4C    AC   D0 0003E          MOVL    76(AP), START_CAP_STRING                     ; 1515
                    6240 00000000'    EF   B0 00042          MOVW    SMG$$DATA_OFFSET, (R2)[R0]                   ; 1516
                FE    8F        01    A6   91 0004A 4$:      CMPB    1(START_CAP_STRING), #-2                     ; 1519
                                      17   13 0004F          BEQL    6$                                           ;
                      58              69   D0 00051          MOVL    (R9), R8                                     ; 1522
                      57              66   98 00054          CVTBL   (START_CAP_STRING), R7                       ; 1524
                      50              57   D0 00057          MOVL    R7, R0                                       ;
                                      03   18 0005A          BGEQ    5$                                           ;
                      50              50   CE 0005C          MNEGL   R0, R0                                       ;
      02 A046     58    BC              58   28 0005F 5$:      MOVC3   R8, @88(AP), 2(R0)[START_CAP_STRING]         ;
                                      1D   11 00066          BRB     8$                                           ; 1519
                      57              66   98 00068 6$:      CVTBL   (START_CAP_STRING), R7                       ; 1530
                      50              57   D0 0006B          MOVL    R7, R0                                       ;
                                      03   18 0006E          BGEQ    7$                                           ;
                      50              50   CE 00070          MNEGL   R0, R0                                       ;
```

```
                                50          56  C0 00073 7$:     ADDL2   START_CAP_STRING, R0                          ; 1529
                                58          69  D0 00076         MOVL    (R9), R8                                      ; 1533
                                60          58  9B 00079         MOVZBW  R8, (R0)                                      ;
          02    A0        58    BC          58  28 0007C         MOVC3   R8, @88(AP), 2(R0)                            ; 1537
                                5A          02  D0 00082         MOVL    #2, SIZE_TYPE                                 ; 1538
                                            57  D5 00085 8$:     TSTL    R7                                            ; 1542
                                            0A  18 00087         BGEQ    9$                                            ;
                50              58          5A  C1 00089         ADDL3   SIZE_TYPE, R8, R0                             ;
                66              57          50  83 0008D         SUBB3   R0, R7, (START_CAP_STRING)                    ;
                                            08  11 00091         BRB     10$                                           ;
                50              58          5A  C1 00093 9$:     ADDL3   SIZE_TYPE, R8, R0                             ;
                66              57          50  81 00097         ADDB3   R0, R7, (START_CAP_STRING)                    ;
                                50      4C  AC  C0 0009B 10$:    ADDL2   76(AP), R0                                    ; 1545
                                05          5B  E9 0009F         BLBC    FIRST, 11$                                    ;
                                51          02  D0 000A2         MOVL    #2, R1                                        ;
                                            02  11 000A5         BRB     12$                                           ;
                                            51  D4 000A7 11$:    CLRL    R1                                            ;
          4C    AC              50          51  C1 000A9 12$:    ADDL3   R1, R0, 76(AP)                                ;
                51              69          5A  C1 000AE         ADDL3   SIZE_TYPE, (R9), R1                           ;
                                51 00000000' EF  C0 000B2        ADDL2   SMG$$DATA_OFFSET, R1                          ;
                                05          5B  E9 000B9         BLBC    FIRST, 13$                                    ;
                                50          02  D0 000BC         MOVL    #2, R0                                        ;
                                            02  11 000BF         BRB     14$                                           ;
                                            50  D4 000C1 13$:    CLRL    R0                                            ;
     00000000'  EF              51          50  C1 000C3 14$:    ADDL3   R0, R1, SMG$$DATA_OFFSET                      ;
                                            69  D4 000CB         CLRL    (R9)                                          ; 1549
                                        58  AC  D4 000CD         CLRL    88(AP)                                        ; 1550
                   FFFFFFFF     8F 00000000' EF  D1 000D0        CMPL    SMG$$MASK_ADR, #-1                            ; 1553
                                            15  12 000DB         BNEQ    16$                                           ;
                                            66  95 000DD         TSTB    (START_CAP_STRING)                            ; 1560
                                            03  15 000DF         BLEQ    15$                                           ;
                                66          66  8E 000E1         MNEGB   (START_CAP_STRING), (START_CAP_STRING)        ; 1562
                          01    A6 00000000' EF  90 000E4 15$:    MOVB    SMG$$MASK_ADR, 1(START_CAP_STRING)            ; 1564
                                00000000'   EF  D4 000EC         CLRL    SMG$$MASK_ADR                                 ; 1566
                                50          01  D0 000F2 16$:    MOVL    #1, R0                                        ; 1572
                                                04 000F5         RET                                                   ; 1574

; Routine Size:  246 bytes,    Routine Base:  _SMG$CODE + 0426
```

```
1 %SBTTL 'SMG$$MISSING_END - signal missing end statement'
1 GLOBAL ROUTINE SMG$$MISSING_END =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Signal that some other statement occurred where there should have
1 !       been an END statement.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !
1 !       SMG$$MISSING_END ()
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     SIGNAL_STOP (SMG$_ERRLIN,
2                  1, .SMG$$CURRENT_LINE,
2                  SMG$_MISENDSTA)
2
1     END;                                  ! end of routine SMG$$MISSING_END
```

```
                                  0000 00000        .ENTRY  SMG$$MISSING_END, Save nothing          ; 1576
                  00000000G  00   9F 00002        PUSHAB  SMG$_MISENDSTA                          ; 1615
                  00000000'  EF   DD 00008        PUSHL   SMG$$CURRENT_LINE                       ; 1616
                             01   DD 0000E        PUSHL   #1                                      ; 1615
                  00000000G  00   9F 00010        PUSHAB  SMG$_ERRLIN                             ;
     00000000G  00           04   FB 00016        CALLS   #4, [IB$STOP                            ;
                                  04 0001D        RET                                             ; 1619
```

; Routine Size: 30 bytes,   Routine Base: _SMG$CODE + 051C

```
;  1563          1620  1 %SBTTL 'SMG$$NEXT_RECORD - Get next record from TERMTABLE.TXT'
;  1564          1621  1 GLOBAL ROUTINE SMG$$NEXT_RECORD =
;  1565          1622  1
;  1566          1623  1 !++
;  1567          1624  1 ! FUNCTIONAL DESCRIPTION:
;  1568          1625  1 !
;  1569          1626  1 !       Gets the next record to be parsed from TERMTABLE.TXT.
;  1570          1627  1 !
;  1571          1628  1 ! CALLING SEQUENCE:
;  1572          1629  1 !
;  1573          1630  1 !       status = SMG$$NEXT_RECORD ()
;  1574          1631  1 !
;  1575          1632  1 ! FORMAL PARAMETERS:
;  1576          1633  1 !
;  1577          1634  1 !       NONE
;  1578          1635  1 !
;  1579          1636  1 ! IMPLICIT INPUTS:
;  1580          1637  1 !
;  1581          1638  1 !       AP      Points to TPARSE parameter block
;  1582          1639  1 !
;  1583          1640  1 ! IMPLICIT OUTPUTS:
;  1584          1641  1 !
;  1585          1642  1 !       the string pointer and string count in the TPARSE
;  1586          1643  1 !       parameter block are updated to point to the new record
;  1587          1644  1 !
;  1588          1645  1 ! COMPLETION STATUS:
;  1589          1646  1 !
;  1590          1647  1 !        SS$_NORMAL
;  1591          1648  1 !
;  1592          1649  1 ! SIDE EFFECTS:
;  1593          1650  1 !
;  1594          1651  1 !--
;  1595          1652  1
;  1596          1653  2     BEGIN
;  1597          1654  2     BUILTIN
;  1598          1655  2         AP;
;  1599          1656  2     MAP
;  1600          1657  2         AP : REF BLOCK [,BYTE];
;  1601          1658  2     LOCAL
;  1602          1659  2         TXT_RAB : REF $RAB_DECL,
;  1603          1660  2         RMS_STATUS;
;  1604          1661  2
;  1605          1662  2 !+
;  1606          1663  2 ! Get a record from TERMTABLE.TXT.  Use the private parameters
;  1607          1664  2 ! appended to the TPARSE parameter block to find the RAB.
;  1608          1665  2 !-
;  1609          1666  2
;  1610          1667  2     TXT_RAB = .AP [PARAM_A_TXT_RAB];
;  1611          1668  2
;  1612          1669  2     DO
;  1613          1670  3         BEGIN
;  1614          1671  3         RMS_STATUS = $GET (RAB = .TXT_RAB);
;  1615          1672  3         END
;  1616          1673  2     UNTIL .RMS_STATUS NEQ RMS$_RSA;
;  1617          1674  2
;  1618          1675  2     IF NOT (.RMS_STATUS) AND
;  1619          1676  2        .RMS_STATUS NEQ RMS$_EOF
```

```
;  1620          1677  2     THEN
;  1621          1678  2         SIGNAL_STOP (.RMS_STATUS);
;  1622          1679  2
;  1623          1680  2 !+
;  1624          1681  2 ! Set TPARSE pointer to the new record.
;  1625          1682  2 !-
;  1626          1683  2
;  1627          1684  2     IF .RMS_STATUS NEQ RMS$_EOF
;  1628          1685  2     THEN
;  1629          1686  3         BEGIN
;  1630          1687  3         LOCAL
;  1631          1688  3             UP_STATUS,
;  1632          1689  3             SRC_STR : BLOCK [8,BYTE],
;  1633          1690  3             ORIG_STRING : BLOCK [8,BYTE],
;  1634          1691  3             COPY_STATUS;
;  1635          1692  3
;  1636          1693  3         SMG$$CURRENT_LINE = .SMG$$CURRENT_LINE + 1;
;  1637          1694  3                                         ! maintain line number for errors
;  1638          1695  3         !+
;  1639          1696  3         ! We need to parse an upcased string, but copy from
;  1640          1697  3         ! the user's original string, preserving lower case
;  1641          1698  3         ! escape sequences.  So save away the original string.
;  1642          1699  3         !-
;  1643          1700  3
;  1644          1701  3         ORIG_STRING [DSC$B_CLASS] = DSC$K_CLASS_S;
;  1645          1702  3         ORIG_STRING [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1646          1703  3         ORIG_STRING [DSC$W_LENGTH] = 255;
;  1647          1704  3         ORIG_STRING [DSC$A_POINTER] = .AP [PARAM_L_ORIG_TXT];
;  1648          1705  3
;  1649          1706  3         SRC_STR [DSC$B_CLASS] = DSC$K_CLASS_S;
;  1650          1707  3         SRC_STR [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1651          1708  3         SRC_STR [DSC$W_LENGTH] = .TXT_RAB [RAB$W_RSZ];
;  1652          1709  3         SRC_STR [DSC$A_POINTER] = .TXT_RAB [RAB$L_RBF];
;  1653          1710  3
;  1654          1711  4         IF NOT (COPY_STATUS = STR$COPY_DX (ORIG_STRING, SRC_STR))
;  1655          1712  3         THEN
;  1656          1713  3             SIGNAL_STOP (.COPY_STATUS);
;  1657          1714  3
;  1658          1715  4         IF NOT (UP_STATUS = STR$UPCASE (SRC_STR, SRC_STR))
;  1659          1716  3         THEN
;  1660          1717  3             SIGNAL_STOP (.UP_STATUS);
;  1661          1718  3
;  1662          1719  3         AP [TPA$L_STRINGPTR] = .SRC_STR [DSC$A_POINTER];
;  1663          1720  3         AP [TPA$L_STRINGCNT] = .SRC_STR [DSC$W_LENGTH];
;  1664          1721  3         END
;  1665          1722  2     ELSE
;  1666          1723  3         BEGIN
;  1667          1724  3         !+
;  1668          1725  3         ! Tell LIB$TPARSE there's no more to parse.
;  1669          1726  3         !-
;  1670          1727  3         AP [TPA$L_STRINGPTR] = UPLIT (BYTE (NULL));
;  1671          1728  3         AP [TPA$L_STRINGCNT] = 1;
;  1672          1729  2         END;
;  1673          1730  2
;  1674          1731  2     RETURN (SS$_NORMAL);
;  1675          1732  2
;  1676          1733  1     END;                                    ! end of routine SMG$$NEXT_RECORD
```

```
                                                  0053A          .BLKB   2
                                            00    0053C P.AAB:   .BYTE   0                                          ;

                                                                 .EXTRN  SYS$GET

                                            001C 00000           .ENTRY  SMG$$NEXT_RECORD, Save R2,R3,R4           ; 1621
                  54 00000000G   00   9E 00002                   MOVAB   LIB$STOP, R4                              ;
                  5E             10   C2 00009                   SUBL2   #16, SP                                   ;
                  52        28   AC   D0 0000C                   MOVL    40(AP), TXT_RAB                           ; 1667
                                 52   DD 00010 1$:               PUSHL   TXT_RAB                                   ; 1671
        00000000G 00             01   FB 00012                   CALLS   #1, SYS$GET                               ;
                  53             50   D0 00019                   MOVL    R0, RMS_STATUS                            ;
        000182DA  8F             53   D1 0001C                   CMPL    RMS_STATUS, #99034                        ; 1673
                                 EB   13 00023                   BEQL    1$                                        ;
                  0E             53   E8 00025                   BLBS    RMS_STATUS, 2$                            ; 1675
        0001827A  8F             53   D1 00028                   CMPL    RMS_STATUS, #98938                        ; 1676
                                 05   13 0002F                   BEQL    2$                                        ;
                                 53   DD 00031                   PUSHL   RMS_STATUS                                ; 1678
                  64             01   FB 00033                   CALLS   #1, LIB$STOP                              ;
        0001827A  8F             53   D1 00036 2$:               CMPL    RMS_STATUS, #98938                        ; 1684
                                 58   13 0003D                   BEQL    5$                                        ;
                     00000000'   EF   D6 0003F                   INCL    SMG$$CURRENT_LINE                         ; 1693
                  6E 010E00FF    8F   D0 00045                   MOVL    #17694975, ORIG_STRING                    ; 1703
              04  AE       5C   AC   D0 0004C                   MOVL    92(AP), ORIG_STRING+4                     ; 1704
              0A  AE     010E   8F   B0 00051                   MOVW    #270, SRC_STR+2                           ; 1707
              08  AE       22   A2   B0 00057                   MOVW    34(TXT_RAB), SRC_STR                      ; 1708
              0C  AE       28   A2   D0 0005C                   MOVL    40(TXT_RAB), SRC_STR+4                    ; 1709
                           08   AE   9F 00061                   PUSHAB  SRC_STR                                   ; 1711
                           04   AE   9F 00064                   PUSHAB  ORIG_STRING                               ;
        00000000G 00             02   FB 00067                   CALLS   #2, STR$COPY_DX                           ;
                  05             50   E8 0006E                   BLBS    COPY_STATUS, 3$                           ;
                                 50   DD 00071                   PUSHL   COPY_STATUS                               ; 1713
                  64             01   FB 00073                   CALLS   #1, LIB$STOP                              ;
                           08   AE   9F 00076 3$:               PUSHAB  SRC_STR                                   ; 1715
                           0C   AE   9F 00079                   PUSHAB  SRC_STR                                   ;
        00000000G 00             02   FB 0007C                   CALLS   #2, STR$UPCASE                            ;
                  05             50   E8 00083                   BLBS    UP_STATUS, 4$                             ;
                                 50   DD 00086                   PUSHL   UP_STATUS                                 ; 1717
                  64             01   FB 00088                   CALLS   #1, LIB$STOP                              ;
              0C  AC       0C   AE   D0 0008B 4$:               MOVL    SRC_STR+4, 12(AP)                         ; 1719
              08  AC       08   AE   3C 00090                   MOVZWL  SRC_STR, 8(AP)                            ; 1720
                                 0A   11 00095                   BRB     6$                                        ; 1684
              0C  AC     FF64   CF   9E 00097 5$:               MOVAB   P.AAB, 12(AP)                             ; 1727
              08  AC             01   D0 0009D                   MOVL    #1, 8(AP)                                 ; 1728
                  50             01   D0 000A1 6$:               MOVL    #1, R0                                    ; 1731
                                      04 000A4                   RET                                               ; 1733

; Routine Size:  165 bytes,    Routine Base:  _SMG$CODE + 053D
```

```
;  1678          1734  1 %SBTTL 'SMG$$SAVE_TOKEN_STRING - Save count & pointer to current token'
;  1679          1735  1 GLOBAL ROUTINE SMG$$SAVE_TOKEN_STRING =
;  1680          1736  1
;  1681          1737  1 !++
;  1682          1738  1 ! FUNCTIONAL DESCRIPTION:
;  1683          1739  1 !
;  1684          1740  1 !       Save a pointer to the start of the capability string, and
;  1685          1741  1 !       maintain a count of the characters.
;  1686          1742  1 !
;  1687          1743  1 ! CALLING SEQUENCE:
;  1688          1744  1 !
;  1689          1745  1 !       status = SAVE_TOKEN_STRING ()
;  1690          1746  1 !
;  1691          1747  1 ! FORMAL PARAMETERS:
;  1692          1748  1 !
;  1693          1749  1 !       NONE
;  1694          1750  1 !
;  1695          1751  1 ! IMPLICIT INPUTS:
;  1696          1752  1 !
;  1697          1753  1 !       AP      Points to TPARSE parameter block
;  1698          1754  1 !
;  1699          1755  1 ! IMPLICIT OUTPUTS:
;  1700          1756  1 !
;  1701          1757  1 !       NONE
;  1702          1758  1 !
;  1703          1759  1 ! COMPLETION STATUS:
;  1704          1760  1 !
;  1705          1761  1 !        SS$_NORMAL
;  1706          1762  1 !
;  1707          1763  1 ! SIDE EFFECTS:
;  1708          1764  1 !
;  1709          1765  1 !--
;  1710          1766  1
;  1711          1767  2     BEGIN
;  1712          1768  2     BUILTIN
;  1713          1769  2         AP;
;  1714          1770  2     MAP
;  1715          1771  2         AP : REF BLOCK [,BYTE];
;  1716          1772  2
;  1717          1773  2 !+
;  1718          1774  2 ! TPARSE maintains a pointer to the current character but what we need
;  1719          1775  2 ! is all the characters between quotes - they are the current capability
;  1720          1776  2 ! string.  So we maintain our own pointer into the TPARSE input string.
;  1721          1777  2 ! (COPY_CAP will use this information to copy the current capability.
;  1722          1778  2 ! COPY_CAP also re-initializes the count & pointer to zero after it does
;  1723          1779  2 ! its job.)
;  1724          1780  2 !
;  1725          1781  2 ! Note that we store a pointer to the user's original string, not the
;  1726          1782  2 ! string that we parsed.  This is because the parse string has been
;  1727          1783  2 ! upcased but we need to preserve lower case letters in an escape sequence.
;  1728          1784  2 !
;  1729          1785  2 ! For a numeric string, there are no quotes delimiting the string - we
;  1730          1786  2 ! will be storing all digits.
;  1731          1787  2 !-
;  1732          1788  2
;  1733          1789  2     IF .AP [PARAM_L_SAVED_TOKENSTR] EQL 0
;  1734          1790  2     THEN
```

```
;  1735          1791  2          !+
;  1736          1792  2          ! Only save the pointer the first time we're called.
;  1737          1793  2          !-
;  1738          1794  3          BEGIN
;  1739          1795  3          LOCAL
;  1740          1796  3              TXT_RAB : REF $RAB_DECL,
;  1741          1797  3              OFFSET;
;  1742          1798  3
;  1743          1799  3          TXT_RAB = .AP [PARAM_A_TXT_RAB];
;  1744          1800  3          OFFSET = .AP [TPA$L_TOKENPTR] - .TXT_RAB [RAB$L_RBF];
;  1745          1801  3                                        ! compute offset into parsed buffer
;  1746          1802  3          AP [PARAM_L_SAVED_TOKENSTR] = .AP [PARAM_L_ORIG_TXT] + .OFFSET;
;  1747          1803  2          END;
;  1748          1804  2
;  1749          1805  2      AP [PARAM_L_SAVED_TOKENCNT] = .AP [PARAM_L_SAVED_TOKENCNT] +
;  1750          1806  2                                    .AP [TPA$L_TOKENCNT];
;  1751          1807  2
;  1752          1808  2      RETURN (SS$_NORMAL);
;  1753          1809  2
;  1754          1810  1      END;                                ! end of routine SMG$$SAVE_TOKEN_STRING


                                        0000 00000          .ENTRY  SMG$$SAVE_TOKEN_STRING, Save nothing          ;  1735
                                  58    AC  D5 00002          TSTL    88(AP)                                        ;  1789
                                        10  12 00005          BNEQ    1$                                            ;
                              50        28  AC  D0 00007          MOVL    40(AP), TXT_RAB                           ;  1799
                        50        14    28  A0  C3 0000B          SUBL3   40(TXT_RAB), 20(AP), OFFSET               ;  1800
                                  58    AC  5C BC40  9E 00011          MOVAB   @92(AP)[OFFSET], 88(AP)               ;  1802
                                  54    AC  10  AC  C0 00017 1$:       ADDL2   16(AP), 84(AP)                        ;  1806
                                        50      01  D0 0001C          MOVL    #1, R0                                ;  1808
                                                    04 0001F          RET                                           ;  1810

; Routine Size:  32 bytes,    Routine Base:  _SMG$CODE + 05E2
```

```
;  1756          1811  1 %SBTTL 'SMG$$STORE_CAP_MASK - Store the capability mask'
;  1757          1812  1 GLOBAL ROUTINE SMG$$STORE_CAP_MASK =
;  1758          1813  1
;  1759          1814  1 !++
;  1760          1815  1 ! FUNCTIONAL DESCRIPTION:
;  1761          1816  1 !
;  1762          1817  1 !       The capability mask is actually the capability number.  We
;  1763          1818  1 !       need to remember this so that we store the data in the correct
;  1764          1819  1 !       location.
;  1765          1820  1 !
;  1766          1821  1 ! CALLING SEQUENCE:
;  1767          1822  1 !
;  1768          1823  1 !       status = SMG$$STORE_CAP_MASK ()
;  1769          1824  1 !
;  1770          1825  1 ! FORMAL PARAMETERS:
;  1771          1826  1 !
;  1772          1827  1 !       NONE
;  1773          1828  1 !
;  1774          1829  1 ! IMPLICIT INPUTS:
;  1775          1830  1 !
;  1776          1831  1 !       AP      Points to TPARSE parameter block
;  1777          1832  1 !
;  1778          1833  1 ! IMPLICIT OUTPUTS:
;  1779          1834  1 !
;  1780          1835  1 !       NONE
;  1781          1836  1 !
;  1782          1837  1 ! COMPLETION STATUS:
;  1783          1838  1 !
;  1784          1839  1 !        SS$_NORMAL
;  1785          1840  1 !
;  1786          1841  1 ! SIDE EFFECTS:
;  1787          1842  1 !
;  1788          1843  1 !--
;  1789          1844  1
;  1790          1845  1
;  1791          1846  2     BEGIN
;  1792          1847  2     BUILTIN
;  1793          1848  2         AP;
;  1794          1849  2     MAP
;  1795          1850  2         AP : REF BLOCK [,BYTE];
;  1796          1851  2
;  1797          1852  2     !+
;  1798          1853  2     ! Remember the capability number so that when we find the associated
;  1799          1854  2     ! value, we know where to store it.  The capability number is used as
;  1800          1855  2     ! an index.
;  1801          1856  2     !-
;  1802          1857  2
;  1803          1858  2     AP [PARAM_L_CUR_CAP_NUMBER] = .SMG$$MASK_ADR;
;  1804          1859  2     SMG$$MASK_ADR = 0;                      ! clear for next cap number
;  1805          1860  2
;  1806          1861  2     RETURN (SS$_NORMAL);
;  1807          1862  2
;  1808          1863  1     END;                                    ! end of routine SMG$$STORE_CAP_MASK
```

```
                                  0004 00000        .ENTRY  SMG$$STORE_CAP_MASK, Save R2                ; 1812
          52 00000000'   EF   9E 00002        MOVAB   SMG$$MASK_ADR, R2                           ;
     50   AC             62   D0 00009        MOVL    SMG$$MASK_ADR, 80(AP)                       ; 1858
                         62   D4 0000D        CLRL    SMG$$MASK_ADR                               ; 1859
          50             01   D0 0000F        MOVL    #1, R0                                      ; 1861
                              04 00012        RET                                                 ; 1863

; Routine Size:  19 bytes,    Routine Base:  _SMG$CODE + 0602
```

```
1 %SBTTL 'SMG$$SYNTAX_ERROR - Signal a syntax error'
1 GLOBAL ROUTINE SMG$$SYNTAX_ERROR =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       Signal that a syntax error occurred while parsing TERMTABLE.TXT.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !
1 !       SMG$$SYNTAX_ERROR ()
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       AP      Points to TPARSE parameter block
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !        SS$_NORMAL
1 !
1 ! SIDE EFFECTS:
1 !
1 !--
1
2     BEGIN
2     BUILTIN
2         AP;
2     MAP
2         AP : REF BLOCK [,BYTE];
2
2     SIGNAL_STOP (SMG$_ERRAT_LIN,
2                  3, .SMG$$CURRENT_LINE,
2                  .AP [TPA$L_TOKENCNT],
2                  .AP [TPA$L_TOKENPTR],
2                  SMG$_SYNERR)
2
1     END;                                  ! end of routine SMG$$SYNTAX_ERROR


                                0000 00000        .ENTRY  SMG$$SYNTAX_ERROR, Save nothing          ; 1865
                 00000000G  00   9F 00002        PUSHAB  SMG$_SYNERR                              ; 1903
              7E        10  AC   7D 00008        MOVQ    16(AP), -(SP)                            ; 1905
                 00000000'  EF   DD 0000C        PUSHL   SMG$$CURRENT_LINE                        ; 1904
                            03   DD 00012        PUSHL   #3                                       ; 1903
                 00000000G  00   9F 00014        PUSHAB  SMG$_ERRAT_LIN                           ;
```

```
                        00000000G  00          06  FB 0001A          CALLS   #6, LIB$STOP                        ;
                                                   04 00021          RET                                        ; 1909

; Routine Size:  34 bytes,    Routine Base:  _SMG$CODE + 0615
```

```
;  1857          1910  1 %SBTTL 'SMG$$WRITE_DATA - perform block I/O write of data to file'
;  1858          1911  1 GLOBAL ROUTINE SMG$$WRITE_DATA =
;  1859          1912  1
;  1860          1913  1 !++
;  1861          1914  1 ! FUNCTIONAL DESCRIPTION:
;  1862          1915  1 !
;  1863          1916  1 !       This routine writes the terminal data to TERMTABLE.EXE using
;  1864          1917  1 !       block I/O.
;  1865          1918  1 !
;  1866          1919  1 ! CALLING SEQUENCE:
;  1867          1920  1 !
;  1868          1921  1 !
;  1869          1922  1 !       SMG$$WRITE_DATA ()
;  1870          1923  1 !
;  1871          1924  1 ! FORMAL PARAMETERS:
;  1872          1925  1 !
;  1873          1926  1 !       NONE
;  1874          1927  1 !
;  1875          1928  1 ! IMPLICIT INPUTS:
;  1876          1929  1 !
;  1877          1930  1 !       AP      Points to TPARSE parameter block
;  1878          1931  1 !
;  1879          1932  1 ! IMPLICIT OUTPUTS:
;  1880          1933  1 !
;  1881          1934  1 !       NONE
;  1882          1935  1 !
;  1883          1936  1 ! COMPLETION STATUS:
;  1884          1937  1 !
;  1885          1938  1 !        SS$_NORMAL
;  1886          1939  1 !
;  1887          1940  1 ! SIDE EFFECTS:
;  1888          1941  1 !
;  1889          1942  1 !--
;  1890          1943  1
;  1891          1944  2     BEGIN
;  1892          1945  2     BUILTIN
;  1893          1946  2         AP;
;  1894          1947  2     MAP
;  1895          1948  2         AP : REF BLOCK [,BYTE];
;  1896          1949  2     LOCAL
;  1897          1950  2         BINARY_RAB : REF $RAB_DECL,      ! point to RAB
;  1898          1951  2         NUM_BYTES,                       ! number of bytes to write
;  1899          1952  2         NUM_BLOCKS : INITIAL (0);        ! number of blocks used
;  1900          1953  2
;  1901          1954  2     BINARY_RAB = .AP [PARAM_A_BINARY_RAB];
;  1902          1955  2
;  1903          1956  2     NUM_BYTES = .AP [PARAM_L_CUR_DATA_BYTE] - .AP [PARAM_A_CAP_DATA];
;  1904          1957  2     BINARY_RAB [RAB$W_RSZ] = .NUM_BYTES;
;  1905          1958  2     BINARY_RAB [RAB$L_RBF] = .AP [PARAM_A_CAP_DATA];
;  1906          1959  2                                          ! set up size & buffer addr
;  1907          1960  2     BINARY_RAB [RAB$L_BKT] = .SMG$$CURRENT_BLOCK;
;  1908          1961  2                                          ! set virtual block number
;  1909          1962  2
;  1910          1963  2     $WRITE (RAB = .BINARY_RAB);
;  1911          1964  2
;  1912          1965  2 !+
;  1913          1966  2 ! Update the next available block number.
```

```
; 1914          1967  2 !-
; 1915          1968  2
; 1916          1969  2     WHILE .NUM_BYTES GEQ 1 DO
; 1917          1970  3     BEGIN
; 1918          1971  3         NUM_BYTES = .NUM_BYTES - 512;
; 1919          1972  3         NUM_BLOCKS = .NUM_BLOCKS + 1;
; 1920          1973  2     END;
; 1921          1974  2
; 1922          1975  2     SMG$$CURRENT_BLOCK = .SMG$$CURRENT_BLOCK + .NUM_BLOCKS;
; 1923          1976  2
; 1924          1977  2     RETURN (SS$_NORMAL);
; 1925          1978  1     END;                                  ! end of routine SMG$$WRITE_BLOCK


                                        001C 00000          .ENTRY  SMG$$WRITE_DATA, Save R2,R3,R4                  ; 1911
                        54 00000000'  EF   9E 00002          MOVAB   SMG$$CURRENT_BLOCK, R4                          ;
                                      53   D4 00009          CLRL    NUM_BLOCKS                                      ; 1944
                        50       30   AC   D0 0000B          MOVL    48(AP), BINARY_RAB                              ; 1954
          52       4C   AC       38   AC   C3 0000F          SUBL3   56(AP), 76(AP), NUM_BYTES                       ; 1956
                   22   A0            52   B0 00015          MOVW    NUM_BYTES, 34(BINARY_RAB)                       ; 1957
                   28   A0       38   AC   D0 00019          MOVL    56(AP), 40(BINARY_RAB)                          ; 1958
                   38   A0            64   D0 0001E          MOVL    SMG$$CURRENT_BLOCK, 56(BINARY_RAB)              ; 1960
                                      50   DD 00022          PUSHL   BINARY_RAB                                      ; 1963
          00000000G     00            01   FB 00024          CALLS   #1, SYS$WRITE                                   ;
                                      52   D5 0002B 1$:      TSTL    NUM_BYTES                                       ; 1969
                                      09   15 0002D          BLEQ    2$                                              ;
                        52     FE00   C2   9E 0002F          MOVAB   -512(R2), NUM_BYTES                             ; 1971
                                      53   D6 00034          INCL    NUM_BLOCKS                                      ; 1972
                                      F3   11 00036          BRB     1$                                              ; 1969
                        64            53   C0 00038 2$:      ADDL2   NUM_BLOCKS, SMG$$CURRENT_BLOCK                  ; 1975
                        50            01   D0 0003B          MOVL    #1, R0                                          ; 1977
                                           04 0003E          RET                                                     ; 1978

; Routine Size:  63 bytes,    Routine Base:  _SMG$CODE + 0637


; 1926          1979  1
; 1927          1980  1 !<BLF/PAGE>
```

```
; 1929          1981  1 END                                    ! End of module SMG$STATEMENT_TABLES
; 1930          1982  1
; 1931          1983  0 ELUDOM



                                                                        .EXTRN  LIB$STOP

;                                   PSECT SUMMARY
;
;       Name                    Bytes                       Attributes
;
;   _SMG$DATA                      24  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   _LIB$KEY0$                     12  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;   _LIB$STATE$                   302  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;   _LIB$KEY1$                     42  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(1)
;   _SMG$CODE                    1654  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)



;                              Library Statistics
;
;                                          -------- Symbols --------      Pages       Processing
;       File                               Total    Loaded   Percent      Mapped      Time
;
;   _$255$DUA28:[SYSLIB]STARLET.L32;1       9776        90         0        581         00:01.0
;   _$255$DUA28:[SMGRTL.OBJ]RTLLIB.L32;1      36         0         0          8         00:00.1
;   _$255$DUA28:[SMGRTL.OBJ]SMGLIB.L32;1     469         1         0         38         00:00.4
;   _$255$DUA28:[SMGRTL.OBJ]SMGTPALIB.L32;1
;                                            41        23        56         10         00:00.1
;   _$255$DUA28:[SYSLIB]TPAMAC.L32;1         42        28        66         14         00:00.1



;                                   COMMAND QUALIFIERS

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

; Size:          1647 code + 387 data bytes
; Run Time:         00:55.2
; Elapsed Time:     02:46.4
; Lines/CPU Min:     2156
; Lexemes/CPU-Min: 53016
; Memory Used:  249 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SMGTPALIB
LIS

SMGSTATAB
LIS

SMGSTRTAB
LIS