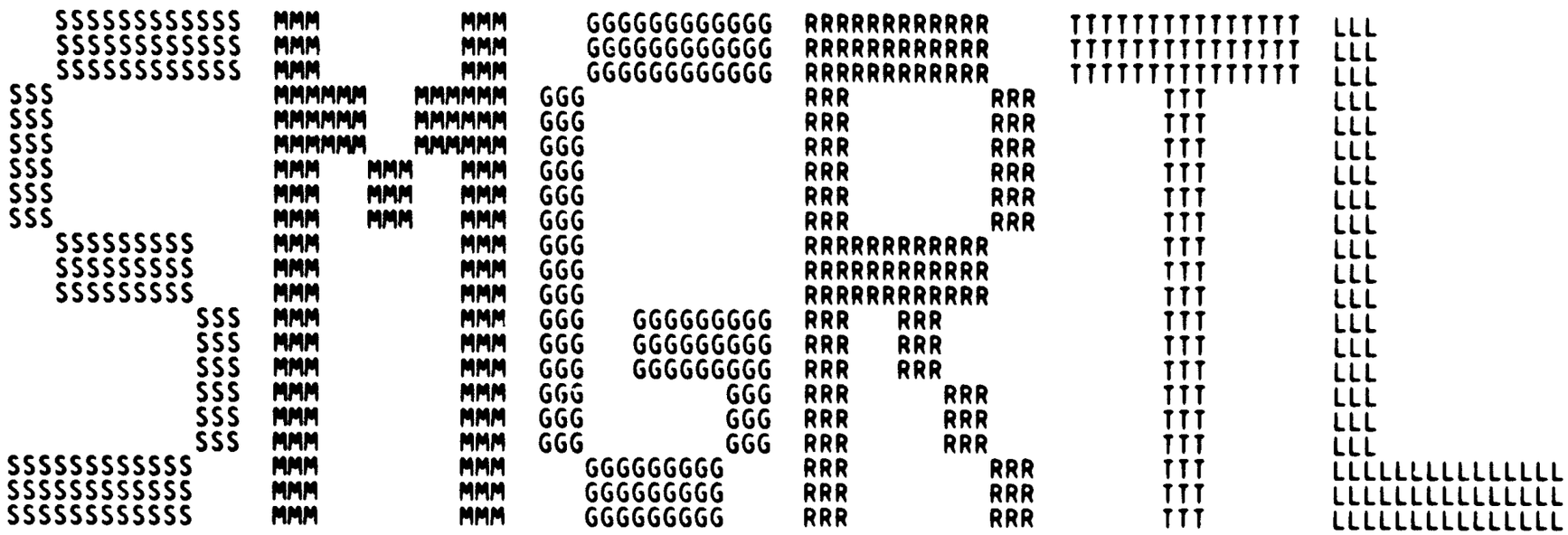


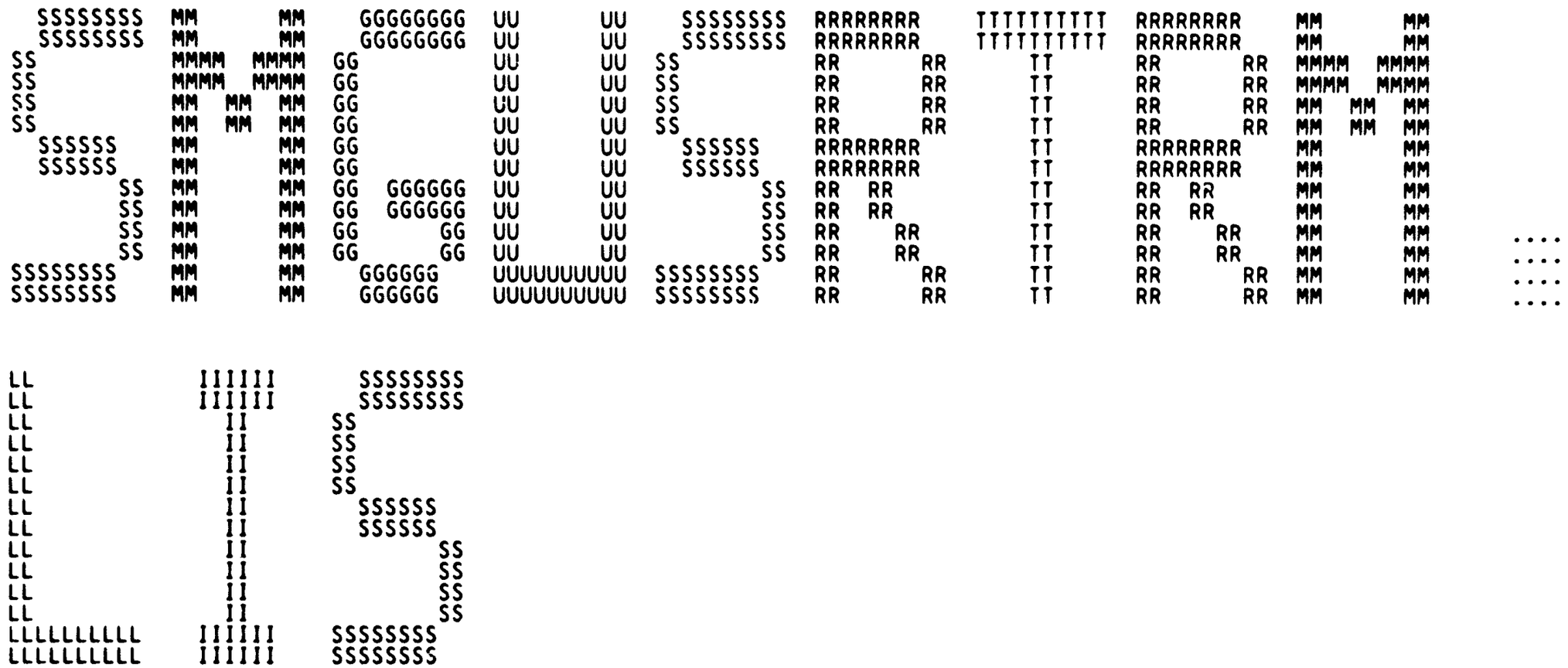
_$2

Val

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

FILEID**SMGUSRTRM

C 1



```
;     1          0001  0 %TITLE 'SMG$INTERFACE_TERM_TABLE - User Interface to Termtable'
;     2          0002  0 MODULE SMG$INTERFACE_TERM_TABLE (
;     3          0003  0                 IDENT = 'T-005' ! File: SMGUSRTRM.B32 Edit: STAN1005
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1 !
;     7          0007  1 !*****************************************************************************
;     8          0008  1 !*                                                                           *
;     9          0009  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    10          0010  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    11          0011  1 !*  ALL RIGHTS RESERVED.                                                     *
;    12          0012  1 !*                                                                           *
;    13          0013  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    14          0014  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    15          0015  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    16          0016  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    17          0017  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    18          0018  1 !*  TRANSFERRED.                                                             *
;    19          0019  1 !*                                                                           *
;    20          0020  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    21          0021  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    22          0022  1 !*  CORPORATION.                                                             *
;    23          0023  1 !*                                                                           *
;    24          0024  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    25          0025  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*                                                                           *
;    28          0028  1 !*****************************************************************************
;    29          0029  1 !
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 ! FACILITY:     Screen Management
;    33          0033  1 !
;    34          0034  1 ! ABSTRACT:
;    35          0035  1 !
;    36          0036  1 !       This module contains routines which return device-specific
;    37          0037  1 !       information about terminals, including, but not
;    38          0038  1 !       limited to, the proper character sequences to perform
;    39          0039  1 !       sundry functions.
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT:  User mode, Shared library routines.
;    42          0042  1 !
;    43          0043  1 ! AUTHOR: P. Levesque, CREATION DATE: 9-Nov-1983
;    44          0044  1 !
;    45          0045  1 ! MODIFIED BY:
;    46          0046  1 !
;    47          0047  1 ! 1-005 - Remove use of FNM= construct which produces code that
;    48          0048  1 !         is not appropriate for use within a shared image.
;    49          0049  1 !         STAN 18-Mar-1984.
;    50          0050  1 ! 1-004 - When fetching the terminal type from a private section,
;    51          0051  1 !         'unbias' the number.  PLL 22-Feb-1984
;    52          0052  1 ! 1-003 - Allow shared use of private sections. STAN. 14-Feb-1984.
;    53          0053  1 ! 1-002 - Allow positive terminal types. STAN. 8-Feb-1984.
;    54          0054  1 ! 1-001 - Original.  PLL 9-Nov-1983
;    55          0055  1 !--
```

```
;    57          0056  1 %SBTTL 'Declarations'
;    58          0057  1 !
;    59          0058  1 ! SWITCHES:
;    60          0059  1 !
;    61          0060  1
;    62          0061  1 !
;    63          0062  1 ! LINKAGES:
;    64          0063  1 !
;    65          0064  1 !       NONE
;    66          0065  1 !
;    67          0066  1 ! INCLUDE FILES:
;    68          0067  1 !
;    69          0068  1
;    70          0069  1 REQUIRE 'SRC$:SMGPROLOG';               ! defines psects, macros, etc.
;    71          0147  1
;    72          0148  1 LIBRARY 'RTLML:SMGTPALIB';              ! TERMTABLE macros, defs, etc
;    73          0149  1
;    74          0150  1 !
;    75          0151  1 ! TABLE OF CONTENTS:
;    76          0152  1 !
;    77          0153  1
;    78          0154  1 FORWARD ROUTINE
;    79          0155  1
;    80          0156  1     SMG$INIT_TERM_TABLE,                ! Setup term table for future calls
;    81          0157  1     SMG$INIT_TERM_TABLE_BY_TYPE,        ! Same as above except input is
;    82          0158  1                                         ! device type instead of device name
;    83          0159  1     SMG$GET_TERM_DATA,                  ! Get terminal characteristics
;    84          0160  1     SMG$DEL_TERM_TABLE,                 ! Delete term table
;    85          0161  1
;    86          0162  1     MAP_PRIV_SECTION,                   ! Map private TERMTABLE.EXE as a
;    87          0163  1                                         ! temporary section
;    88          0164  1     MAP_GBL_SECTION,                    ! Map to system TERMTABLE.EXE in
;    89          0165  1                                         ! global section
;    90          0166  1     SEARCH_FOR_TERM_DEF,                ! Search an index for a given terminal
;    91          0167  1                                         ! name
;    92          0168  1     SEARCH_FOR_TERM_DEF_BY_TYPE,        ! Search an index for a given terminal
;    93          0169  1                                         ! type
;    94          0170  1     DECODE_ARITH_STRING;                ! Decode arithmetic data
;    95          0171  1
;    96          0172  1 !
;    97          0173  1 ! MACROS:
;    98          0174  1 !
;    99          0175  1 !
;   100          0176  1
;   101        M 0177  1 MACRO $BIAS_TYPE_NUMBER (TERM_TAB, TYPE, LENGTH, BUFFER) =
;   102        M 0178  1 !+
;   103        M 0179  1 !       If this definition is from a private table, terminal type
;   104        M 0180  1 !       numbers must be biased so that they are in the range -65 to -127.
;   105        M 0181  1 !       (Note that this bias can't be applied at compile time because it's
;   106        M 0182  1 !       not known whether the table will be mapped privately or globally.)
;   107        M 0183  1 !
;   108        M 0184  1 !       Since we can't modify the type number in the definition (the section
;   109        M 0185  1 !       is read only), we must copy the type number directly into the user's
;   110        M 0186  1 !       buffer.
;   111        M 0187  1 !-
;   112        M 0188  1 BEGIN
;   113        M 0189  1 LOCAL
```

```
;  114        M 0190  1     BIAS_TYPE;
;  115        M 0191  1
;  116        M 0192  1 !+
;  117        M 0193  1 ! Compare the terminal definition address with the starting address of
;  118        M 0194  1 ! the private section.  If there is no private section or the terminal
;  119        M 0195  1 ! definition is at a lower address, then this must be a global definition.
;  120        M 0196  1 !-
;  121        M 0197  1     BIAS_TYPE = TYPE;                   ! init to no bias
;  122        M 0198  1
;  123        M 0199  1     IF .PRIV_SECTION_ADDR NEQ 0 AND
;  124        M 0200  1        TERM_TAB GEQ .PRIV_SECTION_ADDR
;  125        M 0201  1     THEN
;  126        M 0202  1         !+
;  127        M 0203  1         ! Don't bias the type of a DEC terminal (which will have
;  128        M 0204  1         ! a positive number).
;  129        M 0205  1         !-
;  130        M 0206  1         IF TYPE LSS 0
;  131        M 0207  1         THEN
;  132        M 0208  1             BIAS_TYPE = .BIAS_TYPE + K_PRIV_TYPE;
;  133        M 0209  1
;  134        M 0210  1     CH$MOVE (LENGTH, BIAS_TYPE, BUFFER);
;  135        M 0211  1                                 ! put in the user's buffer
;  136          0212  1 END%;
;  137          0213  1
;  138        M 0214  1 MACRO $UNBIAS_TYPE_NUMBER (TERM_TAB, TYPE) =
;  139        M 0215  1 BEGIN
;  140        M 0216  1 !+
;  141        M 0217  1 ! Compare the terminal definition address with the starting address of
;  142        M 0218  1 ! the private section.  If there is no private section or the terminal
;  143        M 0219  1 ! definition is at a lower address, then this must be a global definition.
;  144        M 0220  1 !-
;  145        M 0221  1     IF .PRIV_SECTION_ADDR NEQ 0 AND
;  146        M 0222  1        TERM_TAB GEQ .PRIV_SECTION_ADDR
;  147        M 0223  1     THEN
;  148        M 0224  1         !+
;  149        M 0225  1         ! Don't bias the type of a DEC terminal (which will have
;  150        M 0226  1         ! a positive number).
;  151        M 0227  1         !-
;  152        M 0228  1         IF .TYPE LSS 0
;  153        M 0229  1         THEN
;  154        M 0230  1             TYPE = .TYPE - K_PRIV_TYPE;
;  155        M 0231  1
;  156          0232  1 END;%;
;  157          0233  1
;  158          0234  1 !+
;  159          0235  1 ! Some string capabilities expect parameters.  If the caller does not
;  160          0236  1 ! supply any parameters, we default them to 1's.  The caller must supply
;  161          0237  1 ! all or none of the parameters.
;  162          0238  1 !-
;  163        M 0239  1 MACRO $CHECK_SUFFICIENT_USER_ARGS =
;  164        M 0240  1 BEGIN
;  165        M 0241  1     IF NOT NULLPARAMETER (INPUT_ARG_VECTOR)
;  166        M 0242  1     THEN
;  167        M 0243  1         IF .INPUT_ARG_VECTOR [0] NEQ 0
;  168        M 0244  1         THEN
;  169        M 0245  1             BEGIN
;  170        M 0246  1             IF .INPUT_ARG_VECTOR [0] NEQ .SMG$$NUM_PARAMS [..REQUEST_CODE]
```

```
;  171        M 0247  1                THEN
;  172        M 0248  1                    RETURN (SMG$_NO_ARGS);  ! require all or none
;  173        M 0249  1
;  174        M 0250  1                ! Caller provided args - overwrite defaults
;  175        M 0251  1
;  176        M 0252  1                INCR CTR FROM 0 TO .SMG$$NUM_PARAMS [..REQUEST_CODE] - 1 DO
;  177        M 0253  1                    LOCAL_ARGS [.CTR] = .INPUT_ARG_VECTOR [.CTR+1];
;  178        M 0254  1                END;
;  179          0255  1 END;%;
;  180          0256  1
;  181          0257  1 !
;  182          0258  1 ! EQUATED SYMBOLS:
;  183          0259  1 !
;  184          0260  1
;  185          0261  1 !
;  186          0262  1 ! FIELDS:
;  187          0263  1 !
;  188          0264  1 !       NONE
;  189          0265  1 !
;  190          0266  1 ! PSECTS:
;  191          0267  1 !
;  192          0268  1
;  193          0269  1 !
;  194          0270  1 ! STRUCTURES:
;  195          0271  1 !
;  196          0272  1
;  197          0273  1
;  198          0274  1 !
;  199          0275  1 ! OWN STORAGE:
;  200          0276  1 !
;  201          0277  1
;  202          0278  1 !+
;  203          0279  1 ! The following are used to avoid mapping a section twice.
;  204          0280  1 !-
;  205          0281  1
;  206          0282  1 OWN
;  207          0283  1     GBL_SECTION_ADDR : INITIAL (0) VOLATILE,
;  208          0284  1     PRIV_SECTION_ADDR : INITIAL (0) VOLATILE,
;  209          0285  1     PRIV_SECTION_ADDR_END : INITIAL (0) VOLATILE,
;  210          0286  1     PRIV_SECTION_CHANNEL : INITIAL (0) VOLATILE;
;  211          0287  1
;  212          0288  1 !+
;  213          0289  1 ! We need the byte equivalents of some longword constants.
;  214          0290  1 ! Bliss doesn't think it's possible for a byte to be equal
;  215          0291  1 ! to a longword and it is much more efficient for us to
;  216          0292  1 ! deal with bytes.
;  217          0293  1 !-
;  218          0294  1 OWN
;  219          0295  1     K_OPERAND : BYTE INITIAL (SMG$K_OPERAND <0,8>),
;  220          0296  1     K_SUBSTITUTE : BYTE INITIAL (SMG$K_SUBSTITUTE <0,8>),
;  221          0297  1     K_ADD : BYTE INITIAL (SMG$K_ADD <0,8>),
;  222          0298  1     K_SUBTRACT : BYTE INITIAL (SMG$K_SUBTRACT <0,8>),
;  223          0299  1     K_MULTIPLY : BYTE INITIAL (SMG$K_MULTIPLY <0,8>),
;  224          0300  1     K_DIVIDE : BYTE INITIAL (SMG$K_DIVIDE <0,8>),
;  225          0301  1     K_STORE : BYTE INITIAL (SMG$K_STORE <0,8>);
;  226          0302  1
;  227          0303  1 !
```

```
;  228          0304  1 ! Literals used in various routines
;  229          0305  1 !
;  230          0306  1
;  231          0307  1 LITERAL
;  232          0308  1     K_MATCH = 0,
;  233          0309  1     K_OFF = 0,
;  234          0310  1     K_ON = 1,
;  235          0311  1     K_PRIV_TYPE = -65;          ! private types are -65 to -127,
;  236          0312  1                                 ! system types are -2 to -64
;  237          0313  1
;  238          0314  1 !
;  239          0315  1 ! EXTERNAL REFERENCES:
;  240          0316  1 !
;  241          0317  1
;  242          0318  1 EXTERNAL ROUTINE
;  243          0319  1     SMG$$NUMBER_PARAMETERS,             ! init # params for caps
;  244          0320  1     STR$COPY_DX,                        ! copy source string to destination
;  245          0321  1     STR$CASE_BLIND_COMPARE;             ! compare 2 strings without
;  246          0322  1                                         ! regard for upper/lower case
;  247          0323  1
;  248          0324  1 EXTERNAL
;  249          0325  1     SMG$_WRONUMARG,     ! wrong number of arguments
;  250          0326  1     SMG$_TABID_MIS,     ! Termtable Id Mismatch
;  251          0327  1     SMG$_UNDTERNAM,     ! Undefined terminal name
;  252          0328  1     SMG$_UNDTERTYP,     ! Undefined terminal type
;  253          0329  1     SMG$_PRISECMAP,     ! Private terminal table used
;  254          0330  1     SMG$_GBLSECMAP,     ! System terminal table used
;  255          0331  1     SMG$_UNDTERNOP,     ! Undefined terminal - no private termtable
;  256          0332  1     SMG$_UNDTERNOS,     ! Undefined terminal - no system termtable
;  257          0333  1     SMG$_INVTERTAB,     ! Invalid Termtable
;  258          0334  1     SMG$_INVREQCOD,     ! Invalid request code
;  259          0335  1     SMG$_FATERRLIB,     ! Fatal error in library
;  260          0336  1     SMG$_NO_ARGS;       ! No arguments to substitute
;  261          0337  1
;  262          0338  1 EXTERNAL
;  263          0339  1     SMG$$NUM_PARAMS : VECTOR [,BYTE];
;  264          0340  1                         ! number of parameters for each string cap
```

```
;  266          0341  1 %SBTTL 'SMG$INIT_TERM_TABLE  - Initialize terminal table'
;  267          0342  1 GLOBAL ROUTINE SMG$INIT_TERM_TABLE (
;  268          0343  1                 P_TERM_NAME,
;  269          0344  1                 P_TERM_TABLE     ) =
;  270          0345  1
;  271          0346  1 !++
;  272          0347  1 ! FUNCTIONAL DESCRIPTION:
;  273          0348  1 !
;  274          0349  1 !       This routine points to a specific entry in the terminal capabilities
;  275          0350  1 !       file, TERMTABLE.EXE.  TERM$TABLOC is searched first for a private
;  276          0351  1 !       copy of TERMTABLE.EXE, and if this fails the system definition file
;  277          0352  1 !       (in global section SMG$TERMTABLE) is used.  TERMTABLE.EXE must
;  278          0353  1 !       previously have been created by compiling TERMTABLE.TXT.  The address
;  279          0354  1 !       returned by this routine is used as an input to SMG$GET_TERM_DATA,
;  280          0355  1 !       which fetches capability data.
;  281          0356  1 !
;  282          0357  1 ! CALLING SEQUENCE:
;  283          0358  1 !
;  284          0359  1 !       ret_status.wlc.v = SMG$INIT_TERM_TABLE (P_TERM_NAME.rt.dx,
;  285          0360  1 !                                               P_TERM_TABLE.wa.r)
;  286          0361  1 !
;  287          0362  1 ! FORMAL PARAMETERS:
;  288          0363  1 !
;  289          0364  1 !       P_TERM_NAME             Name of the terminal - must be
;  290          0365  1 !                               an entry in TERMTABLE.EXE
;  291          0366  1 !
;  292          0367  1 !       P_TERM_TABLE            Address of the table created
;  293          0368  1 !
;  294          0369  1 ! IMPLICIT INPUTS:
;  295          0370  1 !
;  296          0371  1 !       Contents of general terminal capabilities file.
;  297          0372  1 !
;  298          0373  1 ! IMPLICIT OUTPUTS:
;  299          0374  1 !
;  300          0375  1 !       NONE
;  301          0376  1 !
;  302          0377  1 ! COMPLETION STATUS:
;  303          0378  1 !
;  304          0379  1 !       SMG$_PRISECMAP          success - definition found in private TERMTABLE
;  305          0380  1 !       SMG$_GBLSECMAP          success - definition found in global TERMTABLE
;  306          0381  1 !       SMG$_UNDTERNAM          undefined terminal name
;  307          0382  1 !       SMG$_UNDTERNOP          undefined terminal - no private terminal table
;  308          0383  1 !       SMG$_UNDTERNOS          undefined terminal - no system terminal table
;  309          0384  1 !       RMS errors
;  310          0385  1 !
;  311          0386  1 ! SIDE EFFECTS:
;  312          0387  1 !
;  313          0388  1 !       NONE
;  314          0389  1 !--
;  315          0390  2     BEGIN
;  316          0391  2     LOCAL
;  317          0392  2         FOUND : INITIAL (1),            ! flag to indicate term def found
;  318          0393  2         TMP_TERM_DEF,                   ! temp. place to store def addr
;  319          0394  2         PREV_AST_STATE,                 ! retd by $SETAST to indicate AST
;  320          0395  2                                         ! delivery was enabled or disabled
;  321          0396  2         PRIV_STATUS : INITIAL (1),      ! status retd BY MAP_PRIV_SECTION
;  322          0397  2         GBL_STATUS : INITIAL (1),       ! status retd by MAP_GBL_SECTION
```

```
;  323          0398  2          RET_STATUS : INITIAL (1);       ! status to return to caller
;  324          0399  2
;  325          0400  2      $SMG$VALIDATE_ARGCOUNT (2,2);
;  326          0401  2
;  327          0402  2  !+
;  328          0403  2  ! First we look for a private copy of TERMTABLE.EXE.  If
;  329          0404  2  ! PRIV_SECTION_ADDR is already set, then we don't need to map
;  330          0405  2  ! the section again.
;  331          0406  2  !-
;  332          0407  2
;  333          0408  2      PREV_AST_STATE = $SETAST (ENBFLG = K_OFF);  ! disable ASTs while setting addr
;  334          0409  2
;  335          0410  2      IF .PRIV_SECTION_ADDR EQL 0
;  336          0411  2      THEN
;  337          0412  2          PRIV_STATUS = MAP_PRIV_SECTION ();
;  338          0413  2                                          ! map priv section if possible
;  339          0414  2
;  340          0415  2      IF .PREV_AST_STATE EQL SS$_WASSET
;  341          0416  2      THEN
;  342          0417  2          $SETAST (ENBFLG = K_ON);        ! re-enable ASTs
;  343          0418  2
;  344          0419  2
;  345          0420  2  !+
;  346          0421  2  ! See if the requested terminal definition is in the private section.
;  347          0422  2  !-
;  348          0423  2
;  349          0424  2      IF .PRIV_SECTION_ADDR NEQ 0
;  350          0425  2      THEN
;  351          0426  3          BEGIN
;  352          0427  3          !+
;  353          0428  3          ! This routine will search the entire terminal index.
;  354          0429  3          !-
;  355          0430  3          FOUND = SEARCH_FOR_TERM_DEF (.PRIV_SECTION_ADDR, .P_TERM_NAME,
;  356          0431  3                                       TMP_TERM_DEF);
;  357          0432  3          RET_STATUS = SMG$_PRISECMAP;
;  358          0433  2          END;
;  359          0434  2
;  360          0435  2  !+
;  361          0436  2  ! If there was no private copy of Termtable or the requested terminal definition
;  362          0437  2  ! wasn't defined there, then map to the global system copy.
;  363          0438  2  !-
;  364          0439  2
;  365          0440  2      IF .FOUND NEQ K_MATCH
;  366          0441  2      THEN
;  367          0442  3          BEGIN
;  368          0443  3
;  369          0444  3          PREV_AST_STATE = $SETAST (ENBFLG = K_OFF);      ! disable ASTs
;  370          0445  3
;  371          0446  3          IF .GBL_SECTION_ADDR EQL 0
;  372          0447  3          THEN
;  373          0448  3              GBL_STATUS = MAP_GBL_SECTION ();
;  374          0449  3
;  375          0450  3          IF .PREV_AST_STATE EQL SS$_WASSET
;  376          0451  3          THEN
;  377          0452  3              $SETAST (ENBFLG = K_ON);                    ! re-enable ASTs
;  378          0453  3
;  379          0454  3          IF .GBL_SECTION_ADDR NEQ 0
```

```
;   380     0455  3          THEN
;   381     0456  4              BEGIN
;   382     0457  4              FOUND = SEARCH_FOR_TERM_DEF (.GBL_SECTION_ADDR, .P_TERM_NAME,
;   383     0458  4                                           TMP_TERM_DEF);
;   384     0459  4              RET_STATUS = SMG$_GBLSECMAP;
;   385     0460  3              END;
;   386     0461  2          END;
;   387     0462  2
;   388     0463  2      !+
;   389     0464  2      ! When we get here, we have looked in both the private and global copies
;   390     0465  2      ! of Termtable.exe.  If we failed to find the requested terminal definition,
;   391     0466  2      ! return an error.
;   392     0467  2      !-
;   393     0468  2
;   394     0469  2          IF .FOUND NEQ K_MATCH
;   395     0470  2          THEN
;   396     0471  3              BEGIN
;   397     0472  3              IF NOT .PRIV_STATUS
;   398     0473  3              THEN
;   399     0474  4                  RETURN (SMG$_UNDTERNOP)
;   400     0475  3              ELSE
;   401     0476  3                  IF NOT .GBL_STATUS
;   402     0477  3                  THEN
;   403     0478  4                      RETURN (SMG$_UNDTERNOS)
;   404     0479  3                  ELSE
;   405     0480  4                      RETURN (SMG$_UNDTERNAM)
;   406     0481  3                  END
;   407     0482  2          ELSE
;   408     0483  2              .P_TERM_TABLE = .TMP_TERM_DEF;
;   409     0484  2
;   410     0485  2          SMG$$NUMBER_PARAMETERS ();          ! init expected # params for caps
;   411     0486  2                                              ! for future get_term_data calls
;   412     0487  2
;   413     0488  2          RETURN (.RET_STATUS);               ! return success which also reveals
;   414     0489  2                                              ! where we found the defintition
;   415     0490  2
;   416     0491  1          END;                                ! end of routine SMG$INIT_TERM_TABLE


                                        .TITLE  SMG$INTERFACE_TERM_TABLE SMG$INTERFACE_TERM_TAB
;                                                                       LE - User Interface to
;                                                                        Te
                                        .IDENT  \1-005\

                                        .PSECT  _SMG$DATA,NOEXE,  PIC,2

                        00000000  00000 GBL_SECTION_ADDR:
                                        .LONG   0                                               ;
                        00000000  00004 PRIV_SECTION_ADDR:
                                        .LONG   0                                               ;
                        00000000  00008 PRIV_SECTION_ADDR_END:
                                        .LONG   0                                               ;
                        00000000  0000C PRIV_SECTION_CHANNEL:
                                        .LONG   0                                               ;
                              FD  00010 K_OPERAND:
                                        .BYTE   -3                                              ;
                              FC  00011 K_SUBSTITUTE:
```

```
                                                        .BYTE   -4                                      ;
                                FB  00012 K_ADD:        .BYTE   -5                                      ;
                                FA  00013 K_SUBTRACT:
                                                        .BYTE   -6                                      ;
                                F9  00014 K_MULTIPLY:
                                                        .BYTE   -7                                      ;
                                F8  00015 K_DIVIDE:
                                                        .BYTE   -8                                      ;
                                F7  00016 K_STORE:      .BYTE   -9                                      ;

                                                        .EXTRN  SMG$$NUMBER_PARAMETERS
                                                        .EXTRN  STR$COPY_DX, STR$CASE_BLIND_COMPARE
                                                        .EXTRN  SMG$_WRONUMARG, SMG$_TABID_MIS
                                                        .EXTRN  SMG$_UNDTERNAM, SMG$_UNDTERTYP
                                                        .EXTRN  SMG$_PRISECMAP, SMG$_GBLSECMAP
                                                        .EXTRN  SMG$_UNDTERNOP, SMG$_UNDTERNOS
                                                        .EXTRN  SMG$_INVTERTAB, SMG$_INVREQCOD
                                                        .EXTRN  SMG$_FATERRLIB, SMG$_NO_ARGS
                                                        .EXTRN  SMG$$NUM_PARAMS
                                                        .EXTRN  SYS$SETAST

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

                                    01FC 00000          .ENTRY  SMG$INIT_TERM_TABLE, Save R2,R3,R4,R5,R6,-      ; 0342
                                                                R7,R8                                           ;
                    58 00000000G    00   9E 00002        MOVAB   SYS$SETAST, R8                                  ;
                    57 00000000'    EF   9E 00009        MOVAB   PRIV_SECTION_ADDR, R7                           ;
                    5E              04   C2 00010        SUBL2   #4, SP                                          ;
                    54              01   D0 00013        MOVL    #1, FOUND                                       ; 0390
                    56              01   D0 00016        MOVL    #1, PRIV_STATUS                                 ;
                    55              01   D0 00019        MOVL    #1, GBL_STATUS                                  ;
                    52              01   D0 0001C        MOVL    #1, RET_STATUS                                  ;
                    02              6C   91 0001F        CMPB    (AP), #2                                        ; 0400
                                    08   13 00022        BEQL    1$                                              ;
                    50 00000000G    8F   D0 00024        MOVL    #SMG$_WRONUMARG, R0                             ;
                                         04 0002B        RET                                                     ;
                                    7E   D4 0002C 1$:    CLRL    -(SP)                                           ; 0408
                    68              01   FB 0002E        CALLS   #1, SYS$SETAST                                  ;
                    53              50   D0 00031        MOVL    R0, PREV_AST_STATE                              ;
                                    67   D5 00034        TSTL    PRIV_SECTION_ADDR                               ; 0410
                                    08   12 00036        BNEQ    2$                                              ;
            0000V   CF              00   FB 00038        CALLS   #0, MAP_PRIV_SECTION                            ; 0412
                    56              50   D0 0003D        MOVL    R0, PRIV_STATUS                                 ;
                    09              53   D1 00040 2$:    CMPL    PREV_AST_STATE, #9                              ; 0415
                                    05   12 00043        BNEQ    3$                                              ;
                                    01   DD 00045        PUSHL   #1                                              ; 0417
                    68              01   FB 00047        CALLS   #1, SYS$SETAST                                  ;
                                    67   D5 0004A 3$:    TSTL    PRIV_SECTION_ADDR                               ; 0424
                                    16   13 0004C        BEQL    4$                                              ;
                                    5E   DD 0004E        PUSHL   SP                                              ; 0430
                                04  AC   DD 00050        PUSHL   P_TERM_NAME                                     ;
                                    67   DD 00053        PUSHL   PRIV_SECTION_ADDR                               ;
            0000V   CF              03   FB 00055        CALLS   #3, SEARCH_FOR_TERM_DEF                         ;
                    54              50   D0 0005A        MOVL    R0, FOUND                                       ;
                    52 00000000G    00   9E 0005D        MOVAB   SMG$_PRISECMAP, RET_STATUS                      ; 0432
                                    54   D5 00064 4$:    TSTL    FOUND                                           ; 0440
                                    3B   13 00066        BEQL    7$                                              ;
```

```
                                                7E  D4 00068          CLRL    -(SP)                                                   ; 0444
                                            68  01  FB 0006A          CALLS   #1, SYS$SETAST                                          ;
                                            53  50  D0 0006D          MOVL    R0, PREV_AST_STATE                                      ;
                                                FC  A7  D5 00070          TSTL    GBL_SECTION_ADDR                                        ; 0446
                                                    08  12 00073          BNEQ    5$                                                      ;
                                      0000V     CF      00  FB 00075          CALLS   #0, MAP_GBL_SECTION                             ; 0448
                                            55      50  D0 0007A          MOVL    R0, GBL_STATUS                                          ;
                                            09      53  D1 0007D 5$:       CMPL    PREV_AST_STATE, #9                              ; 0450
                                                    05  12 00080          BNEQ    6$                                                      ;
                                                    01  DD 00082          PUSHL   #1                                              ; 0452
                                            68      01  FB 00084          CALLS   #1, SYS$SETAST                                          ;
                                                FC  A7  D5 00087 6$:       TSTL    GBL_SECTION_ADDR                                ; 0454  ;
                                                    17  13 0008A          BEQL    7$                                                      ;
                                                    5E  DD 0008C          PUSHL   SP                                              ; 0457
                                                04  AC  DD 0008E          PUSHL   P_TERM_NAME                                             ;
                                                FC  A7  DD 00091          PUSHL   GBL_SECTION_ADDR                                        ;
                                      0000V     CF      03  FB 00094          CALLS   #3, SEARCH_FOR_TERM_DEF                         ;
                                            54      50  D0 00099          MOVL    R0, FOUND                                               ;
                                            52  00000000G   00  9E 0009C          MOVAB   SMG$_GBLSECMAP, RET_STATUS              ; 0459
                                                    54  D5 000A3 7$:       TSTL    FOUND                                           ; 0469
                                                    1E  13 000A5          BEQL    10$                                                     ;
                                            08      56  E8 000A7          BLBS    PRIV_STATUS, 8$                                 ; 0472
                                            50  00000000G   00  9E 000AA          MOVAB   SMG$_UNDTERNOP, R0                      ; 0474
                                                        04  000B1          RET                                                     ; 0476
                                            08      55  E8 000B2 8$:       BLBS    GBL_STATUS, 9$                                          ;
                                            50  00000000G   00  9E 000B5          MOVAB   SMG$_UNDTERNOS, R0                      ; 0478
                                                        04  000BC          RET                                                             ;
                                            50  00000000G   00  9E 000E  9$:       MOVAB   SMG$_UNDTERNAM, R0                      ; 0480
                                                        04  00C           RET                                                     ; 0476
                                                08      BC      6E  D0 00C5 10$:       MOVL    TMP_TERM_DEF, @P_TERM_TABLE             ; 0483
                                      00000000G     00      00  FB 000C9          CALLS   #0, SMG$$NUMBER_PARAMETERS              ; 0485
                                            50      52  D0 000D0          MOVL    RET_STATUS, R0                                  ; 0488
                                                        04  000D3          RET                                                     ; 0491

; Routine Size:  212 bytes,    Routine Base:  _SMG$CODE + 0000
```

```
;  418          0492  1 %SBTTL 'SMG$INIT_TERM_TABLE_BY_TYPE - Init terminal def address given type'
;  419          0493  1 GLOBAL ROUTINE SMG$INIT_TERM_TABLE_BY_TYPE (
;  420          0494  1                                             P_TERM_TYPE,
;  421          0495  1                                             P_TERM_TABLE,
;  422          0496  1                                             P_TERM_NAME ) =
;  423          0497  1 !++
;  424          0498  1 ! FUNCTIONAL DESCRIPTION:
;  425          0499  1 !
;  426          0500  1 !       This routine points to a specific entry in the terminal capabilities
;  427          0501  1 !       file, TERMTABLE.EXE.  TERM$TABLOC is searched first for a private
;  428          0502  1 !       copy of TERMTABLE.EXE, and if this fails the system definition file
;  429          0503  1 !       (in global section SMG$TERMTABLE) is used.  TERMTABLE.EXE must
;  430          0504  1 !       previously have been created by compiling TERMTABLE.TXT.  The address
;  431          0505  1 !       returned by this routine is used as an input to SMG$GET_TERM_DATA,
;  432          0506  1 !       which fetches capability data.
;  433          0507  1 !
;  434          0508  1 !       This routine is similar to SMG$INIT_TERM_TABLE except that it
;  435          0509  1 !       accepts a device type rather than a device name as input.
;  436          0510  1 !
;  437          0511  1 ! CALLING SEQUENCE:
;  438          0512  1 !
;  439          0513  1 !       ret_status.wlc.v = SMG$INIT_TERM_TABLE_BY_TYPE (P_TERM_TYPE.rb.r,
;  440          0514  1 !                                                       P_TERM_TABLE.wa.r
;  441          0515  1 !                                                      [,P_TERM_NAME.wt.dx])
;  442          0516  1 !
;  443          0517  1 ! FORMAL PARAMETERS:
;  444          0518  1 !
;  445          0519  1 !       P_TERM_TYPE             Device type of the terminal - must be
;  446          0520  1 !                               an entry in TERMTABLE.EXE.  (The device
;  447          0521  1 !                               type can be obtained via a call to $GETDVIW.)
;  448          0522  1 !                               This is the address of a signed byte.
;  449          0523  1 !
;  450          0524  1 !       P_TERM_TABLE            Address of the table created
;  451          0525  1 !
;  452          0526  1 !       P_TERM_NAME             Optional descriptor to contain the name
;  453          0527  1 !                               associated with this device type
;  454          0528  1 !
;  455          0529  1 ! IMPLICIT INPUTS:
;  456          0530  1 !
;  457          0531  1 !       Contents of general terminal capabilities file.
;  458          0532  1 !
;  459          0533  1 ! IMPLICIT OUTPUTS:
;  460          0534  1 !
;  461          0535  1 !       NONE
;  462          0536  1 !
;  463          0537  1 ! COMPLETION STATUS:
;  464          0538  1 !
;  465          0539  1 !       SMG$_PRISECMAP          success - definition found in private TERMTABLE
;  466          0540  1 !       SMG$_GBLSECMAP          success - definition found in global TERMTABLE
;  467          0541  1 !       SMG$_UNDTERTYP          undefined terminal type
;  468          0542  1 !       SMG$_UNDTERNOP          undefined terminal - no private termtable
;  469          0543  1 !       SMG$_UNDTERNOS          undefined terminal - no system termtable
;  470          0544  1 !
;  471          0545  1 ! SIDE EFFECTS:
;  472          0546  1 !
;  473          0547  1 !       NONE
;  474          0548  1 !--
```

```
;  475          0549  2     BEGIN
;  476          0550  2     BUILTIN
;  477          0551  2         NULLPARAMETER;
;  478          0552  2     LOCAL
;  479          0553  2         FOUND : INITIAL (1),                ! flag to indicate term def found
;  480          0554  2         TMP_TERM_DEF,                       ! temp. place to store def addr
;  481          0555  2         TMP_TERM_NAME : REF VECTOR [,BYTE], ! temp. place to store name addr
;  482          0556  2         PREV_AST_STATE,                     ! retd by $SETAST to indicate ASTs
;  483          0557  2                                             ! previously enabled or disabled
;  484          0558  2         PRIV_STATUS : INITIAL (1),          ! status retd by MAP_PRIV_SECTION
;  485          0559  2         GBL_STATUS : INITIAL (1),           ! status retd by MAP_GBL_SECTION
;  486          0560  2         RET_STATUS : INITIAL (1);
;  487          0561  2     BIND
;  488          0562  2         TERM_TYPE = .P_TERM_TYPE : SIGNED BYTE;
;  489          0563  2
;  490          0564  2     $SMG$VALIDATE_ARGCOUNT (2,3);
;  491          0565  2
;  492          0566  2     .P_TERM_TABLE = 0;                      ! init
;  493          0567  2
;  494          0568  2 !+
;  495          0569  2 ! First we look for a private copy of TERMTABLE.EXE.  If
;  496          0570  2 ! PRIV_SECTION_ADDR is already set, then we don't need to map
;  497          0571  2 ! the section again.
;  498          0572  2 !-
;  499          0573  2
;  500          0574  2     PREV_AST_STATE = $SETAST (ENBFLG = K_OFF);  ! disable ASTs while setting addr
;  501          0575  2
;  502          0576  2     IF .PRIV_SECTION_ADDR EQL 0
;  503          0577  2     THEN
;  504          0578  2         PRIV_STATUS = MAP_PRIV_SECTION ();      ! map priv section if possible
;  505          0579  2
;  506          0580  2     IF .PREV_AST_STATE EQL SS$_WASSET
;  507          0581  2     THEN
;  508          0582  2         $SETAST (ENBFLG = K_ON);                ! re-enable ASTs
;  509          0583  2
;  510          0584  2
;  511          0585  2 !+
;  512          0586  2 ! See if the requested terminal definition is in the private section.
;  513          0587  2 ! If this terminal type number is negative but not in the private range, skip this
;  514          0588  2 ! and proceed to check the global table.
;  515          0589  2 !-
;  516          0590  2
;  517          0591  2     IF .PRIV_SECTION_ADDR NEQ 0 AND
;  518          0592  3        (.TERM_TYPE LSS K_PRIV_TYPE  OR .TERM_TYPE GEQ 0)
;  519          0593  2     THEN
;  520          0594  3         BEGIN
;  521          0595  3         !+
;  522          0596  3         ! This routine will search the entire terminal index.
;  523          0597  3         !-
;  524          0598  3         FOUND = SEARCH_FOR_TERM_DEF_BY_TYPE (.PRIV_SECTION_ADDR, .TERM_TYPE,
;  525          0599  3                                              TMP_TERM_DEF, TMP_TERM_NAME);
;  526          0600  3         RET_STATUS = SMG$_PRISECMAP;
;  527          0601  2         END;
;  528          0602  2
;  529          0603  2 !+
;  530          0604  2 ! If there was no private copy of Termtable or the requested terminal definition
;  531          0605  2 ! wasn't defined there, then map to the global system copy.
```

```
;  532          0606  2 !
;  533          0607  2 ! To prevent private and global Termtable definitions from having the same
;  534          0608  2 ! terminal type numbers assigned, we always bias private ones by -65.  No
;  535          0609  2 ! need to check the global definition for a private number.
;  536          0610  2 ! Positive types and negative types larger than or equal to K_PRIV_TYPE are ok.
;  537          0611  2 !-
;  538          0612  2
;  539          0613  2     IF .FOUND NEQ K_MATCH AND
;  540          0614  2        .TERM_TYPE GEQ K_PRIV_TYPE
;  541          0615  2     THEN
;  542          0616  3         BEGIN
;  543          0617  3
;  544          0618  3         PREV_AST_STATE = $SETAST (ENBFLG = K_OFF);      ! disable ASTs
;  545          0619  3
;  546          0620  3         IF .GBL_SECTION_ADDR EQL 0
;  547          0621  3         THEN
;  548          0622  3             GBL_STATUS = MAP_GBL_SECTION ();
;  549          0623  3
;  550          0624  3         IF .PREV_AST_STATE EQL SS$_WASSET
;  551          0625  3         THEN
;  552          0626  3             $SETAST (ENBFLG = K_ON);                    ! re-enable ASTs
;  553          0627  3
;  554          0628  3         IF .GBL_SECTION_ADDR NEQ 0
;  555          0629  3         THEN
;  556          0630  4             BEGIN
;  557          0631  4             FOUND = SEARCH_FOR_TERM_DEF_BY_TYPE (.GBL_SECTION_ADDR,
;  558          0632  4                                                  .TERM_TYPE,
;  559          0633  4                                                  TMP_TERM_DEF,
;  560          0634  4                                                  TMP_TERM_NAME);
;  561          0635  4             RET_STATUS = SMG$_GBLSECMAP;
;  562          0636  3             END;
;  563          0637  2         END;
;  564          0638  2
;  565          0639  2 !+
;  566          0640  2 ! When we get here, we have looked in both the private and global copies
;  567          0641  2 ! of Termtable.exe.  If we failed to find the requested terminal definition,
;  568          0642  2 ! return an error.
;  569          0643  2 !-
;  570          0644  2
;  571          0645  2     IF .FOUND NEQ K_MATCH
;  572          0646  2     THEN
;  573          0647  3         BEGIN
;  574          0648  3         IF NOT .PRIV_STATUS
;  575          0649  3         THEN
;  576          0650  4             RETURN (SMG$_UNDTERNOP)
;  577          0651  3         ELSE
;  578          0652  3             IF NOT .GBL_STATUS
;  579          0653  3             THEN
;  580          0654  4                 RETURN (SMG$_UNDTERNOS)
;  581          0655  3         ELSE
;  582          0656  4             RETURN (SMG$_UNDTERNAM)
;  583          0657  3         END
;  584          0658  2     ELSE
;  585          0659  3         BEGIN
;  586          0660  3         .P_TERM_TABLE = .TMP_TERM_DEF;
;  587          0661  3                                         ! return addr of terminal definition
;  588          0662  3         IF NOT NULLPARAMETER (P_TERM_NAME)
```

```
;  589          0663  3          THEN
;  590          0664  4              BEGIN                       ! return term name if requested
;  591          0665  4              LOCAL
;  592          0666  4                  TERM_NAME_DESC : BLOCK [8,BYTE];
;  593          0667  4
;  594          0668  4              TERM_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  595          0669  4              TERM_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  596          0670  4              TERM_NAME_DESC [DSC$W_LENGTH] = .TMP_TERM_NAME [0];
;  597          0671  4              TERM_NAME_DESC [DSC$A_POINTER] = TMP_TERM_NAME [2];
;  598          0672  4
;  599          0673  4              STR$COPY_DX (.P_TERM_NAME, TERM_NAME_DESC);
;  600          0674  3              END;
;  601          0675  2          END;
;  602          0676  2
;  603          0677  2      SMG$$NUMBER_PARAMETERS ();          ! init expected # params for caps
;  604          0678  2                                          ! for future get_term_data calls
;  605          0679  2
;  606          0680  2      RETURN (.RET_STATUS);          ! success indicating where we found
;  607          0681  2                                     ! the definition
;  608          0682  2
;  609          0683  1      END;                           ! end of routine SMG$INIT_TERM_TABLE_BY_TYPE



                                    01FC 00000          .ENTRY  SMG$INIT_TERM_TABLE_BY_TYPE, Save R2,R3,R4,-; 0493
                                                                R5,R6,R7,R8                                   ;
                          58 00000000G   00  9E 00002   MOVAB   SYS$SETAST, R8                               ;
                          57 00000000'   EF  9E 00009   MOVAB   PRIV_SECTION_ADDR, R7                        ;
                          5E             10  C2 00010   SUBL2   #16, SP                                     ;
                          53             01  D0 00013   MOVL    #1, FOUND                                   ; 0549
                          56             01  D0 00016   MOVL    #1, PRIV_STATUS                             ;
                          55             01  D0 00019   MOVL    #1, GBL_STATUS                              ;
                          54             01  D0 0001C   MOVL    #1, RET_STATUS                              ;
                50        6C             02  83 0001F   SUBB3   #2, (AP), DIFF                              ; 0564
                          01             50  91 00023   CMPB    DIFF, #1                                    ;
                                         08  1B 00026   BLEQU   1$                                          ;
                          50 00000000G   8F  D0 00028   MOVL    #SMG$_WRONUMARG, R0                         ;
                                         04  0002F     RET                                                 ;
                                   08    BC  D4 00030 1$: CLRL    @P_TERM_TABLE                               ; 0566
                                         7E  D4 00033     CLRL    -(SP)                                       ; 0574
                          68             01  FB 00035     CALLS   #1, SYS$SETAST                              ;
                          52             50  D0 00038     MOVL    R0, PREV_AST_STATE                          ;
                                         67  D5 0003B     TSTL    PRIV_SECTION_ADDR                           ; 0576
                                         08  12 0003D     BNEQ    2$                                          ;
                    0000V CF             00  FB 0003F     CALLS   #0, MAP_PRIV_SECTION                        ; 0578
                          56             50  D0 00044     MOVL    R0, PRIV_STATUS                             ;
                          09             52  D1 00047 2$: CMPL    PREV_AST_STATE, #9                          ; 0580
                                         05  12 0004A     BNEQ    3$                                          ;
                                         01  DD 0004C     PUSHL   #1                                          ; 0582
                          68             01  FB 0004E     CALLS   #1, SYS$SETAST                              ;
                                         67  D5 00051 3$: TSTL    PRIV_SECTION_ADDR                           ; 0591
                                         26  13 00053     BEQL    5$                                          ;
                       BF 8F       04    BC  91 00055     CMPB    @P_TERM_TYPE, #-65                          ; 0592
                                         05  19 0005A     BLSS    4$                                          ;
                                   04    BC  95 0005C     TSTB    @P_TERM_TYPE                                ;
```

```
                                                 1A  19 0005F             BLSS    5$                                                      ;
                                                 5E  DD 00061 4$:        PUSHL   SP                                                      ; 0598
                                           08    AE  9F 00063            PUSHAB  TMP_TERM_DEF                                            ;
                                     7E    04    BC  98 00066            CVTBL   @P_TERM_TYPE, -(SP)                                     ;
                                                 67  DD 00069            PUSHL   PRIV_SECTION_ADDR                                       ;
                                0000V    CF          04  FB 0006C            CALLS   #4, SEARCH_FOR_TERM_DEF_BY_TYPE                         ;
                                     53              50  D0 00071            MOVL    R0, FOUND                                               ;
                                     54 00000000G    00  9E 00074            MOVAB   SMG$_PRISECMAP, RET_STATUS                              ; 0600
                                                     53  D5 0007B 5$:        TSTL    FOUND                                                   ; 0613
                                                     46  13 0007D            BEQL    8$                                                      ;
                                  BF 8F          04  BC  91 0007F            CMPB    @P_TERM_TYPE, #-65                                      ; 0614
                                                     3F  19 00084            BLSS    8$                                                      ;
                                                     7E  D4 00086            CLRL    -(SP)                                                   ; 0618
                                     68              01  FB 00088            CALLS   #1, SYS$SETAST                                          ;
                                     52              50  D0 0008B            MOVL    R0, PREV_AST_STATE                                      ;
                                                 FC  A7  D5 0008E            TSTL    GBL_SECTION_ADDR                                        ; 0620
                                                     08  12 00091            BNEQ    6$                                                      ;
                                0000V    CF          00  FB 00093            CALLS   #0, MAP_GBL_SECTION                                     ; 0622
                                     55              50  D0 00098            MOVL    R0, GBL_STATUS                                          ;
                                     09              52  D1 0009B 6$:        CMPL    PREV_AST_STATE, #9                                      ; 0624
                                                     05  12 0009E            BNEQ    7$                                                      ;
                                                     01  DD 000A0            PUSHL   #1                                                      ; 0626
                                     68              01  FB 000A2            CALLS   #1, SYS$SETAST                                          ;
                                                 FC  A7  D5 000A5 7$:        TSTL    GBL_SECTION_ADDR                                        ; 0628
                                                     1B  13 000A8            BEQL    8$                                                      ;
                                                     5E  DD 000AA            PUSHL   SP                                                      ; 0631
                                               08    AE  9F 000AC            PUSHAB  TMP_TERM_DEF                                            ;
                                     7E        04    BC  98 000AF            CVTBL   @P_TERM_TYPE, -(SP)                                     ; 0632
                                               FC    A7  DD 000B3            PUSHL   GBL_SECTION_ADDR                                        ; 0631
                                0000V    CF          04  FB 000B6            CALLS   #4, SEARCH_FOR_TERM_DEF_BY_TYPE                         ;
                                     53              50  D0 000BB            MOVL    R0, FOUND                                               ;
                                     54 00000000G    00  9E 000BE            MOVAB   SMG$_GBLSECMAP, RET_STATUS                              ; 0635
                                                     53  D5 000C5 8$:        TSTL    FOUND                                                   ; 0645
                                                     1E  13 000C7            BEQL    11$                                                     ;
                                     08              56  E8 000C9            BLBS    PRIV_STATUS, 9$                                         ; 0648
                                     50 00000000G    00  9E 000CC            MOVAB   SMG$_UNDTERNOP, R0                                      ; 0650
                                                     04  000D3             RET                                                             ; 0652
                                     08              55  E8 000D4 9$:        BLBS    GBL_STATUS, 10$                                         ;
                                     50 00000000G    00  9E 000D7            MOVAB   SMG$_UNDTERNOS, R0                                      ; 0654
                                                     04  000DE             RET                                                             ;
                                     50 00000000G    00  9E 000DF 10$:       MOVAB   SMG$_UNDTERNAM, R0                                      ; 0656
                                                     04  000E6             RET                                                             ; 0652
                                08   BC      04     AE  D0 000E7 11$:       MOVL    TMP_TERM_DEF, @P_TERM_TABLE                             ; 0660
                                     03              6C  91 000EC            CMPB    (AP), #3                                                ; 0662
                                                     22  1F 000EF            BLSSU   12$                                                     ;
                                                 0C  AC  D5 000F1            TSTL    12(AP)                                                  ;
                                                     1D  13 000F4            BEQL    12$                                                     ;
                                0A   AE      010E   8F  B0 000F6            MOVW    #270, TERM_NAME_DESC+2                                  ; 0668
                                08   AE        00   BE  9B 000FC            MOVZBW  @TMP_TERM_NAME, TERM_NAME_DESC                          ; 0670
                          0C    AE              6E  02  C1 00101            ADDL3   #2, TMP_TERM_NAME, TERM_NAME_DESC+4                     ; 0671
                                               08   AE  9F 00106            PUSHAB  TERM_NAME_DESC                                          ; 0673
                                               0C   AC  DD 00109            PUSHL   P_TERM_NAME                                             ;
                           00000000G    00          02  FB 0010C            CALLS   #2, STR$COPY_DX                                         ;
                           00000000G    00          00  FB 00113 12$:       CALLS   #0, SMG$$NUMBER_PARAMETERS                              ; 0677
                                     50              54  D0 0011A            MOVL    RET_STATUS, R0                                          ; 0680
                                                     04  0011D             RET                                                             ; 0683
```

; Routine Size: 286 bytes,   Routine Base: _SMG$CODE + 00D4

```
;  611          0684  1 %SBTTL 'SMG$GET_TERM_DATA  - Get terminal data'
;  612          0685  1 GLOBAL ROUTINE SMG$GET_TERM_DATA (
;  613          0686  1                                   P_TERM_TABLE,
;  614          0687  1                                   REQUEST_CODE,
;  615          0688  1                                   MAX_BUFFER_LENGTH,
;  616          0689  1                                   RETURN_LENGTH,
;  617          0690  1                                   BUFFER_ADDRESS,
;  618          0691  1                                   INPUT_ARG_VECTOR : REF VECTOR [,LONG]
;  619          0692  1                                   ) =
;  620          0693  1
;  621          0694  1 !++
;  622          0695  1 ! FUNCTIONAL DESCRIPTION:
;  623          0696  1 !
;  624          0697  1 !       This routine extracts the requested information from
;  625          0698  1 !       the specified terminal table.  The terminal table address must
;  626          0699  1 !       have been obtained by a call to SMG$INIT_TERM_TABLE or
;  627          0700  1 !       SMG$INIT_TERM_TABLE_BY_TYPE prior to calling this routine.
;  628          0701  1 !
;  629          0702  1 ! CALLING SEQUENCE:
;  630          0703  1 !
;  631          0704  1 !       ret_status.wlc.v = SMG$GET_TERM_DATA (P_TERM_TABLE.ra.r,
;  632          0705  1 !                                             REQUEST_CODE.rl.r,
;  633          0706  1 !                                             MAX_BUFFER_LENGTH.rl.r,
;  634          0707  1 !                                             RETURN_LENGTH.wl.r,
;  635          0708  1 !                                             BUFFER_ADDRESS.wa.r
;  636          0709  1 !                                             [,INPUT_ARG_VECTOR.rl.r]
;  637          0710  1 !
;  638          0711  1 ! FORMAL PARAMETERS:
;  639          0712  1 !
;  640          0713  1 !       P_TERM_TABLE            Address of terminal capabilities table
;  641          0714  1 !                               for desired terminal type.  (See
;  642          0715  1 !                               SMG$INIT_TERM_TABLE to obtain this address.)
;  643          0716  1 !
;  644          0717  1 !       REQUEST_CODE            Longword constant of the form
;  645          0718  1 !                               SMG$K_xyz (defined in Digital-supplied
;  646          0719  1 !                               library).  Specifies desired capability.
;  647          0720  1 !                               Xyz corresponds to a keyword in the terminal
;  648          0721  1 !                               definition (for instance ANSI_CRT).
;  649          0722  1 !
;  650          0723  1 !       MAX_BUFFER_LENGTH       Longword number of bytes of the buffer
;  651          0724  1 !                               in which the requested capability data will be
;  652          0725  1 !                               returned.  Most capabilities will fit in
;  653          0726  1 !                               20 bytes or less - you will probably want
;  654          0727  1 !                               to allocate a little extra.
;  655          0728  1 !
;  656          0729  1 !       RETURN_LENGTH           Longword length of the data returned.
;  657          0730  1 !                               It indicates how many bytes of the buffer are
;  658          0731  1 !                               actually used.
;  659          0732  1 !
;  660          0733  1 !       BUFFER_ADDRESS          Longword address of the buffer in which
;  661          0734  1 !                               to return the capability data.
;  662          0735  1 !
;  663          0736  1 !       INPUT_ARG_VECTOR        Optional argument used only for a
;  664          0737  1 !                               'variable' capability.  Address of a longword
;  665          0738  1 !                               vector.  The number of longwords which follow
;  666          0739  1 !                               must be in the first longword.  This vector
;  667          0740  1 !                               is used with capability string containing
```

```
;  668          0741  1 !                                  ! or %.  You may provide the correct number of
;  669          0742  1 !                                  values to be used with each ! and % in the
;  670          0743  1 !                                  string or you may default all of the values.
;  671          0744  1 !                                  SMG$GET_TERM_DATA will supply a default value
;  672          0745  1 !                                  of 1.
;  673          0746  1 !
;  674          0747  1 !
;  675          0748  1 ! IMPLICIT INPUTS:
;  676          0749  1 !
;  677          0750  1 !       Contents of system terminal capabilities file.
;  678          0751  1 !
;  679          0752  1 ! IMPLICIT OUTPUTS:
;  680          0753  1 !
;  681          0754  1 !       NONE
;  682          0755  1 !
;  683          0756  1 ! COMPLETION STATUS:
;  684          0757  1 !
;  685          0758  1 !       SMG$_INVTERTAB          invalid terminal table address
;  686          0759  1 !       SMG$_INVREQCOD          invalid request code
;  687          0760  1 !       SS$_NORMAL              success
;  688          0761  1 !
;  689          0762  1 ! SIDE EFFECTS:
;  690          0763  1 !
;  691          0764  1 !       NONE
;  692          0765  1 !--
;  693          0766  2     BEGIN
;  694          0767  2     BUILTIN
;  695          0768  2         NULLPARAMETER;
;  696          0769  2     LOCAL
;  697          0770  2         DATA : REF VECTOR [,BYTE,SIGNED],! terminal data
;  698          0771  2         DATA_LENGTH;                     ! length of terminal data
;  699          0772  2     BIND
;  700          0773  2         TERM_DEF = ..P_TERM_TABLE,       ! start of terminal definition
;  701          0774  2         TERM_DEF_PTRS = TERM_DEF : VECTOR [,WORD],
;  702          0775  2                                          ! start of data pointers
;  703          0776  2         START_DEF_DATA = TERM_DEF_PTRS + SMG$K_CAP_PTRS_SIZE;
;  704          0777  2                                          ! start of actual data
;  705          0778  2
;  706          0779  2     $SMG$VALIDATE_ARGCOUNT (5, 6);
;  707          0780  2
;  708          0781  2 !+
;  709          0782  2 ! Validate that we have a valid TERMTABLE.  The terminal
;  710          0783  2 ! definition should contain an id in the first byte.
;  711          0784  2 !-
;  712          0785  2
;  713          0786  2     IF .TERM_DEF <0,8> NEQ SMG$K_TERM_DEF_ID
;  714          0787  2     THEN
;  715          0788  2         RETURN (SMG$_INVTERTAB);
;  716          0789  2
;  717          0790  2 !+
;  718          0791  2 ! Make sure this is a valid request code.
;  719          0792  2 !-
;  720          0793  2
;  721          0794  2     IF ..REQUEST_CODE LSS SMG$K_MIN_REQUEST_CODE OR
;  722          0795  2        ..REQUEST_CODE GTR SMG$K_MAX_REQUEST_CODE
;  723          0796  2     THEN
;  724          0797  2         RETURN (SMG$_INVREQCOD);
```

```
;  725          0798  2
;  726          0799  2 !+
;  727          0800  2 ! Find the requested terminal data.
;  728          0801  2 !-
;  729          0802  2
;  730          0803  2     .RETURN_LENGTH = 0;                 ! init
;  731          0804  2
;  732          0805  2     IF .TERM_DEF_PTRS [..REQUEST_CODE] EQL 0
;  733          0806  2     THEN
;  734          0807  2         RETURN (SS$_NORMAL);
;  735          0808  2
;  736          0809  2     DATA = .TERM_DEF_PTRS [..REQUEST_CODE] + TERM_DEF;
;  737          0810  2                             ! offset + start of data area
;  738          0811  2     DATA_LENGTH = .DATA [0];
;  739          0812  2
;  740          0813  2     IF .DATA_LENGTH EQL 0
;  741          0814  2     THEN
;  742          0815  2         RETURN (SS$_NORMAL);    ! requested data not available
;  743          0816  2
;  744          0817  2 !+
;  745          0818  2 ! Branch on whether additional processing is required.
;  746          0819  2 !-
;  747          0820  2
;  748          0821  2     IF .DATA_LENGTH GTR 0
;  749          0822  2     THEN
;  750          0823  3         BEGIN                       ! static data - copy into user's buffer
;  751          0824  3         LOCAL
;  752          0825  3             RET_LEN;
;  753          0826  3
;  754          0827  3         RET_LEN = MIN (..MAX_BUFFER_LENGTH, .DATA_LENGTH);
;  755          0828  3
;  756          0829  3         IF ..REQUEST_CODE EQL SMG$K_VMS_TERMINAL_NUMBER
;  757          0830  3         THEN
;  758          0831  4             $BIAS_TYPE_NUMBER (..P_TERM_TABLE, .DATA [2], .RET_LEN, .BUFFER_ADDRESS)
;  759          0832  4                             ! private numbers must be -65 to -127
;  760          0833  3         ELSE
;  761          0834  3             CH$MOVE (.RET_LEN, DATA [2], .BUFFER_ADDRESS);
;  762          0835  3         !+
;  763          0836  3         ! Return length to caller.
;  764          0837  3         !-
;  765          0838  3         IF .RET_LEN NEQ 0
;  766          0839  3         THEN
;  767          0840  3             .RETURN_LENGTH = .RET_LEN;
;  768          0841  3         END                         ! static data - copy into user's buffer
;  769          0842  3
;  770          0843  2     ELSE
;  771          0844  2
;  772          0845  3         BEGIN                       ! dynamic - additional processing needed
;  773          0846  3         LOCAL
;  774          0847  3             LOCAL_ARGS : VECTOR [10,LONG] INITIAL (REP 10 OF (1)),
;  775          0848  3             KIND;
;  776          0849  3         KIND = .DATA [1];               ! make this a longword for comparison
;  777          0850  3
;  778          0851  3         !+
;  779          0852  3         ! Use either the defaults or the parameters passed by the caller.
;  780          0853  3         !-
;  781          0854  3         $CHECK_SUFFICIENT_USER_ARGS;
```

```
;  782          0855  3
;  783          0856  3          IF .KIND EQL SMG$K_FAO_STRING
;  784          0857  3          THEN
;  785          0858  4              BEGIN                           ! FAO processing
;  786          0859  4              !+
;  787          0860  4              ! Process FAO string.
;  788          0861  4              !-
;  789          0862  4              LOCAL
;  790          0863  4                  FAO_STATUS,
;  791          0864  4                  CONTROL_DSC : BLOCK [8,BYTE],
;  792          0865  4                  RETURN_DSC : BLOCK [8,BYTE];
;  793          0866  4
;  794          0867  4              CONTROL_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  795          0868  4              CONTROL_DSC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  796          0869  4              CONTROL_DSC [DSC$W_LENGTH] = ABS (.DATA_LENGTH);
;  797          0870  4              CONTROL_DSC [DSC$A_POINTER] = DATA [2];
;  798          0871  4
;  799          0872  4              RETURN_DSC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  800          0873  4              RETURN_DSC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  801          0874  4              RETURN_DSC [DSC$W_LENGTH] = ..MAX_BUFFER_LENGTH;
;  802          0875  4              RETURN_DSC [DSC$A_POINTER] = .BUFFER_ADDRESS;
;  803          0876  4
;  804        P 0877  4              FAO_STATUS = $FAOL (CTRSTR = CONTROL_DSC, OUTLEN = .RETURN_LENGTH,
;  805          0878  4                                  OUTBUF = RETURN_DSC, PRMLST = LOCAL_ARGS [0]);
;  806          0879  4              IF NOT .FAO_STATUS
;  807          0880  4              THEN
;  808          0881  4                  RETURN (.FAO_STATUS);
;  809          0882  4
;  810          0883  4              END                             ! end of FAO processing
;  811          0884  4
;  812          0885  3          ELSE
;  813          0886  3
;  814          0887  3          IF .KIND EQL SMG$K_ARITH_STRING
;  815          0888  3          THEN
;  816          0889  4              BEGIN                                   ! arithmetic processing
;  817          0890  4              LOCAL
;  818          0891  4                  RET_LEN,
;  819          0892  4                  DECODED_ARITH_STRING : BLOCK [255,BYTE],
;  820          0893  4                  DECODED_LENGTH,
;  821          0894  4                  STATUS;
;  822          0895  4
;  823          0896  5              IF NOT (STATUS = DECODE_ARITH_STRING (DATA [0],
;  824          0897  5                                                    LOCAL_ARGS [0],
;  825          0898  5                                                     DECODED_ARITH_STRING,
;  826          0899  5                                                     DECODED_LENGTH))
;  827          0900  4              THEN
;  828          0901  4                  RETURN (.STATUS);
;  829          0902  4
;  830          0903  4              RET_LEN = MIN (..MAX_BUFFER_LENGTH, .DECODED_LENGTH);
;  831          0904  4              CH$MOVE (.RET_LEN, DECODED_ARITH_STRING, .BUFFER_ADDRESS);
;  832          0905  4                                              ! copy decoded stuff to user buffer
;  833          0906  4              IF .RET_LEN NEQ 0
;  834          0907  4              THEN
;  835          0908  4                  .RETURN_LENGTH = .RET_LEN;
;  836          0909  4                                              ! return length to caller
;  837          0910  4              END                             ! end of arithmetic processing
;  838          0911  4
```

```
;  839          0912  3          ELSE
;  840          0913  3              RETURN (SMG$_FATERRLIB);    ! unknown type code
;  841          0914  3
;  842          0915  2          END;                        ! dynamic - additional processing needed
;  843          0916  2
;  844          0917  2      RETURN (SS$_NORMAL);
;  845          0918  1      END,                            ! end of SMG$GET_TERM_DATA


                                                  001F2                .BLKB   2
                                        00000001# 001F4 P.AAA:         .LONG   1[10]                                                   ;

                                                                       .EXTRN  SYS$FAOL

                                                  07FC 00000           .ENTRY  SMG$GET_TERM_DATA, Save R2,R3,R4,R5,R6,R7,-             ; 0685
                                                                                R8,R9,R10
                                  5A 00000000'    EF   9E 00002        MOVAB   PRIV_SECTION_ADDR, R10                                  ;
                                  5E      FED0    CE   9E 00009        MOVAB   -304(SP), SP                                            ;
                                  51        04    BC   D0 0000E        MOVL    @P_TERM_TABLE, R1                                       ; 0773
                            50    6C              05   83 00012        SUBB3   #5, (AP), DIFF                                          ; 0779
                                  01              50   91 00016        CMPB    DIFF, #1                                                ;
                                                  08   1B 00019        BLEQU   1$                                                      ;
                                  50 00000000G    8F   D0 0001B        MOVL    #SMG$_WRONUMARG, R0                                     ;
                                                  04   00022           RET                                                             ;
                                        89        8F             61   91 00023 1$:     CMPB    (R1), #137                              ; 0786
                                                                 08   13 00027        BEQL    2$                                      ;
                                  50 00000000G    00   9E 00029        MOVAB   SMG$_INVTERTAB, R0                                      ; 0788
                                                  04   00030           RET                                                             ;
                                  57        08    BC   D0 00031 2$:     MOVL    @REQUEST_CODE, R7                                       ; 0794
                                                  09   15 00035        BLEQ    3$                                                      ;
                             00000294     8F             57   D1 00037        CMPL    R7, #660                                        ; 0795
                                                                 08   15 0003E        BLEQ    4$                                      ;
                                  50 00000000G    00   9E 00040 3$:     MOVAB   SMG$_INVREQCOD, R0                                      ; 0797
                                                  04   00047           RET                                                             ;
                                           10     BC   D4 00048 4$:     CLRL    @RETURN_LENGTH                                          ; 0803
                                              6147 B5   0004B           TSTW    (R1)[R7]                                                ; 0805
                                                 55 13  0004E           BEQL    9$                                                      ;
                                  58             6147 3C   00050        MOVZWL  (R1)[R7], DATA                                          ; 0809
                                  58                51 C0   00054        ADDL2   R1, DATA                                                ;
                                  56                68 98   00057        CVTBL   (DATA), DATA_LENGTH                                     ; 0811
                                                    49 13   0005A        BEQL    9$                                                      ; 0813
                                                    4A 15   0005C        BLEQ    10$                                                     ; 0821
                                  50        0C      BC D0   0005E        MOVL    @MAX_BUFFER_LENGTH, R0                                  ; 0827
                                  56                50 D1   00062        CMPL    R0, DATA_LENGTH                                         ;
                                                    03 15   00065        BLEQ    5$                                                      ;
                                  50                56 D0   00067        MOVL    DATA_LENGTH, R0                                         ;
                                  59                50 D0   0006A 5$:     MOVL    R0, RET_LEN                                             ;
                             000000E3     8F             57   D1 0006D        CMPL    R7, #227                                        ; 0829
                                                                 21   12 00074        BNEQ    7$                                      ;
                                  6E        02      A8 98   00076        CVTBL   2(DATA), BIAS_TYPE                                      ; 0831
                                                    6A D5   0007A        TSTL    PRIV_SECTION_ADDR                                       ;
                                                    12 13   0007C        BEQL    6$                                                      ;
                                  6A        04      BC D1   0007E        CMPL    @P_TERM_TABLE, PRIV_SECTION_ADDR                        ;
                                                    0C 19   00082        BLSS    6$                                                      ;
                                           02     A8 95   00084        TSTB    2(DATA)                                                 ;
                                                    07 18   00087        BGEQ    6$                                                      ;
```

```
                    6E 00000041    8F  C2 00089         SUBL2   #65, BIAS_TYPE                              :
      14   BC               6E         59  28 00090 6$:     MOVC3   RET_LEN, BIAS_TYPE, @BUFFER_ADDRESS         :
                                       06  11 00095         BRB     8$                                          : 0829
      14   BC     02        A8         59  28 00097 7$:     MOVC3   RET_LEN, 2(DATA), @BUFFER_ADDRESS           : 0834
                                       59  D5 0009D 8$:     TSTL    RET_LEN                                     : 0838
                                       04  13 0009F         BEQL    9$                                          :
           10     BC                   59  D0 000A1         MOVL    RET_LEN, @RETURN_LENGTH                     : 0840
                                     00CF  31 000A5 9$:     BRW     19$                                         : 0821
      D8   AD   FF2B        CF         28  28 000A8 10$:    MOVC3   #40, P.AAA, LOCAL_ARGS                      : 0847
                            53    01   A8  98 000AF         CVTBL   1(DATA), KIND                               : 0849
                            06         6C  91 000B3         CMPB    (AP), #6                                    :
                                       34  1F 000B6         BLSSU   14$                                         :
                                  18   AC  D5 000B8         TSTL    24(AP)                                      :
                                       2F  13 000BB         BEQL    14$                                         :
                                  18   BC  D5 000BD         TSTL    @INPUT_ARG_VECTOR                           :
                                       2A  13 000C0         BEQL    14$                                         :
                    52 00000000G0047       9A 000C2         MOVZBL  SMG$$NUM_PARAMS[R7], R2                     :
                    52            18   BC  D1 000CA         CMPL    @INPUT_ARG_VECTOR, R2                       :
                                       08  13 000CE         BEQL    11$                                         :
                    50 00000000G       00  9E 000D0         MOVAB   SMG$_NO_ARGS, R0                            :
                                           04 000D7         RET                                                 :
                    50            01   CE  000D8 11$:       MNEGL   #1, CTR                                     :
                                       0B  11 000DB         BRB     13$                                         :
                    51          18 BC40    DE 000DD 12$:    MOVAL   @INPUT_ARG_VECTOR[CTR], R1                  :
              D8 AD40             04   A1  D0 000E2         MOVL    4(R1), LOCAL_ARGS[CTR]                      :
         F1         50                 52  F2 000E8 13$:    AOBLSS  R2, CTR, 12$                                :
           FFFFFFFF 8F                 53  D1 000EC 14$:    CMPL    KIND, #-1                                   : 0856
                                       3D  12 000F3         BNEQ    16$                                         :
           D2     AD      010E         8F  B0 000F5         MOVW    #270, CONTROL_DSC+2                         : 0867
                                       56  D5 000FB         TSTL    R6                                          : 0869
                                       03  18 000FD         BGEQ    15$                                         :
                    56                 56  CE 000FF         MNEGL   R6, R6                                      :
           D0     AD                   56  B0 00102 15$:    MOVW    R6, CONTROL_DSC                             :
           D4     AD        02         A8  9E 00106         MOVAB   2(R8), CONTROL_DSC+4                        : 0870
           CA     AD      010E         8F  B0 0010B         MOVW    #270, RETURN_DSC+2                          : 0872
           C8     AD        0C         BC  B0 00111         MOVW    @MAX_BUFFER_LENGTH, RETURN_DSC              : 0874
           CC     AD        14         AC  D0 00116         MOVL    BUFFER_ADDRESS, RETURN_DSC+4                : 0875
                            D8         AD  9F 0011B         PUSHAB  LOCAL_ARGS                                  : 0878
                            C8         AD  9F 0011E         PUSHAB  RETURN_DSC                                  :
                            10         AC  DD 00121         PUSHL   RETURN_LENGTH                               :
                            D0         AD  9F 00124         PUSHAB  CONTROL_DSC                                 :
           00000000G  00               04  FB 00127         CALLS   #4, SYS$FAOL                                :
                      46               50  E8 0012E         BLBS    FAO_STATUS, 19$                             : 0879
                                           04 00131         RET                                                 : 0881
           FFFFFFFE   8F               53  D1 00132 16$:    CMPL    KIND, #-2                                   : 0887
                                       34  12 00139         BNEQ    18$                                         :
                            04         AE  9F 0013B         PUSHAB  DECODED_LENGTH                              : 0896
                            0C         AE  9F 0013E         PUSHAB  DECODED_ARITH_STRING                        :
                            D8         AD  9F 00141         PUSHAB  LOCAL_ARGS                                  : 0897
                                       58  DD 00144         PUSHL   DATA                                        : 0896
              0000V   CF               04  FB 00146         CALLS   #4, DECODE_ARITH_STRING                     :
                      2C               50  E9 0014B         BLBC    STATUS, 20$                                 :
                      50    0C         BC  D0 0014E         MOVL    @MAX_BUFFER_LENGTH, R0                      : 0903
           04         AE               50  D1 00152         CMPL    R0, DECODED_LENGTH                          :
                                       04  15 00156         BLEQ    17$                                         :
                      50    04         AE  D0 00158         MOVL    DECODED_LENGTH, R0                          :
                      56               50  D0 0015C 17$:    MOVL    R0, RET_LEN                                 :
```

```
                     14   BC       08    AE             56  28 0015F          MOVC3   RET_LEN, DECODED_ARITH_STRING, -                ; 0904
                                                                                      @BUFFER_ADDRESS
                                                        56  D5 00165          TSTL    RET_LEN                                         ; 0906
                                                        0E  13 00167          BEQL    19$
                                 10    BC               56  D0 00169          MOVL    RET_LEN, @RETURN_LENGTH                         ; 0908
                                                        08  11 0016D          BRB     19$                                             ; 0887
                                       50 00000000G     00  9E 0016F 18$:     MOVAB   SMG$_FATERRLIB, R0                              ; 0913
                                                            04 00176          RET
                                       50               01  D0 00177 19$:     MOVL    #1, R0                                          ; 0917
                                                            04 0017A 20$:     RET                                                     ; 0918

; Routine Size:  379 bytes,    Routine Base:  _SMG$CODE + 021C
```

```
;  847          0919  1 %SBTTL 'SMG$DEL_TERM_TABLE - Delete terminal table'
;  848          0920  1 GLOBAL ROUTINE SMG$DEL_TERM_TABLE =
;  849          0921  1
;  850          0922  1 !++
;  851          0923  1 ! FUNCTIONAL DESCRIPTION:
;  852          0924  1 !
;  853          0925  1 !       This routine terminates access to TERMTABLE.EXE.
;  854          0926  1 !
;  855          0927  1 !       If a private TERMTABLE.EXE was used, then we created a
;  856          0928  1 !       temporary section to map it into the user's virtual address
;  857          0929  1 !       space.  This routine will free up that virtual space so that
;  858          0930  1 !       the user can re-use it.
;  859          0931  1 !
;  860          0932  1 !       If the system TERMTABLE.EXE was used, then we used the system
;  861          0933  1 !       global section, and we do not have to free any virtual address
;  862          0934  1 !       space.  In this case we simply return SS$_NORMAL.
;  863          0935  1 !
;  864          0936  1 !       In both cases (temporary and global section), this call is
;  865          0937  1 !       optional.  Even for a temporary section, the virtual memory
;  866          0938  1 !       would be automatically freed when the image terminated.
;  867          0939  1 !       This routine is useful only in the case where a caller may
;  868          0940  1 !       need to re-use the virtual address space.
;  869          0941  1 !
;  870          0942  1 ! CALLING SEQUENCE:
;  871          0943  1 !
;  872          0944  1 !       ret_status.wlc.v = SMG$DEL_TERM_TABLE
;  873          0945  1 !
;  874          0946  1 ! FORMAL PARAMETERS:
;  875          0947  1 !
;  876          0948  1 !       NONE
;  877          0949  1 !
;  878          0950  1 ! IMPLICIT INPUTS:
;  879          0951  1 !
;  880          0952  1 !       NONE
;  881          0953  1 !
;  882          0954  1 ! IMPLICIT OUTPUTS:
;  883          0955  1 !
;  884          0956  1 !       NONE
;  885          0957  1 !
;  886          0958  1 ! COMPLETION STATUS:
;  887          0959  1 !
;  888          0960  1 !       SS$_NORMAL
;  889          0961  1 !
;  890          0962  1 ! SIDE EFFECTS:
;  891          0963  1 !
;  892          0964  1 !       NONE
;  893          0965  1 !--
;  894          0966  2     BEGIN
;  895          0967  2     LOCAL
;  896          0968  2         PREV_AST_STATE;
;  897          0969  2
;  898          0970  2 !+
;  899          0971  2 ! If the terminal definition came from the system global section,
;  900          0972  2 ! then we simply return.  If the definition came from a private
;  901          0973  2 ! temporary section, then we free up the virtual address space.
;  902          0974  2 !-
;  903          0975  2
```

```
;  904          0976  2     PREV_AST_STATE = $SETAST (ENBFLG = K_OFF);  ! disable ASTs
;  905          0977  2
;  906          0978  2     IF .PRIV_SECTION_ADDR NEQ 0
;  907          0979  2     THEN
;  908          0980  3         BEGIN                       ! free va - temp section
;  909          0981  3         LOCAL
;  910          0982  3             FREE_ADDR : VECTOR [2],
;  911          0983  3             STATUS;
;  912          0984  3
;  913          0985  3         FREE_ADDR [0] = .PRIV_SECTION_ADDR;
;  914          0986  3         FREE_ADDR [1] = .PRIV_SECTION_ADDR_END;
;  915          0987  4         IF NOT (STATUS = $DELTVA (INADR = FREE_ADDR))
;  916          0988  3         THEN
;  917          0989  3             RETURN (.STATUS);
;  918          0990  3
;  919          0991  4         IF NOT (STATUS = $DASSGN (CHAN = .PRIV_SECTION_CHANNEL))
;  920          0992  3         THEN
;  921          0993  3             RETURN (.STATUS);
;  922          0994  3
;  923          0995  3         PRIV_SECTION_ADDR = 0;
;  924          0996  3         PRIV_SECTION_CHANNEL = 0;
;  925          0997  2         END;                                ! free va - temp section
;  926          0998  2
;  927          0999  2     IF .PREV_AST_STATE EQL SS$_WASSET
;  928          1000  2     THEN
;  929          1001  2         $SETAST (ENBFLG = K_ON);                ! re-enable ASTs
;  930          1002  2
;  931          1003  2     RETURN (SS$_NORMAL);
;  932          1004  1     END;                                ! end of SMG$DEL_TERM_TABLE


                                                .EXTRN  SYS$DELTVA, SYS$DASSGN

                                    001C 00000      .ENTRY  SMG$DEL_TERM_TABLE, Save R2,R3,R4       ; 0920
                54 00000000G    00   9E 00002      MOVAB   SYS$SETAST, R4                          ;
                53 00000000'    EF   9E 00009      MOVAB   PRIV_SECTION_ADDR, R3                   ;
                5E                   08   C2 00010      SUBL2   #8, SP                          ;
                                     7E   D4 00013      CLRL    -(SP)                           ; 0976
                64                   01   FB 00015      CALLS   #1, SYS$SETAST                  ;
                52                   50   D0 00018      MOVL    R0, PREV_AST_STATE              ;
                                     63   D5 0001B      TSTL    PRIV_SECTION_ADDR               ; 0978
                                     29   13 0001D      BEQL    1$                              ;
                6E                   63   D0 0001F      MOVL    PRIV_SECTION_ADDR, FREE_ADDR    ; 0985
          04    AE             04    A3   D0 00022      MOVL    PRIV_SECTION_ADDR_END, FREE_ADDR+4 ; 0986
                                     7E   7C 00027      CLRQ    -(SP)                           ; 0987
                               08    AE   9F 00029      PUSHAB  FREE_ADDR                       ;
00000000G       00                   03   FB 0002C      CALLS   #3, SYS$DELTVA                  ;
                1F                   50   E9 00033      BLBC    STATUS, 3$                      ;
                               08    A3   DD 00036      PUSHL   PRIV_SECTION_CHANNEL            ; 0991
00000000G       00                   01   FB 00039      CALLS   #1, SYS$DASSGN                  ;
                12                   50   E9 00040      BLBC    STATUS, 3$                      ;
                                     63   D4 00043      CLRL    PRIV_SECTION_ADDR               ; 0995
                               08    A3   D4 00045      CLRL    PRIV_SECTION_CHANNEL            ; 0996
                09                   52   D1 00048 1$:  CMPL    PREV_AST_STATE, #9              ; 0999
                                     05   12 0004B      BNEQ    2$                              ;
                                     01   DD 0004D      PUSHL   #1                              ; 1001
```

```
                                64          01  FB 0004F         CALLS   #1, SYS$SETAST                          ;
                                50          01  D0 00052 2$:     MOVL    #1, R0                                 ; 1003
                                                04 00055 3$:     RET                                            ; 1004

; Routine Size:  86 bytes,    Routine Base:  _SMG$CODE + 0397
```

```
:  934          1005  1 %SBTTL 'MAP_PRIV_SECTION - Map private section'
:  935          1006  1 ROUTINE MAP_PRIV_SECTION =
:  936          1007  1
:  937          1008  1 !++
:  938          1009  1 ! FUNCTIONAL DESCRIPTION:
:  939          1010  1 !
:  940          1011  1 !       This routine attempts to open a user's private copy of TERMTABLE.EXE.
:  941          1012  1 !       If successful, TERMTABLE.EXE is mapped into the user's virtual address
:  942          1013  1 !       space as a temporary section.
:  943          1014  1 !
:  944          1015  1 ! CALLING SEQUENCE:
:  945          1016  1 !
:  946          1017  1 !       ret_status.wlc.v = MAP_PRIV_SECTION ()
:  947          1018  1 !
:  948          1019  1 ! FORMAL PARAMETERS:
:  949          1020  1 !
:  950          1021  1 !       NONE
:  951          1022  1 !
:  952          1023  1 ! IMPLICIT INPUTS:
:  953          1024  1 !
:  954          1025  1 !       NONE
:  955          1026  1 !
:  956          1027  1 ! IMPLICIT OUTPUTS:
:  957          1028  1 !
:  958          1029  1 !       NONE
:  959          1030  1 !
:  960          1031  1 ! COMPLETION STATUS:
:  961          1032  1 !
:  962          1033  1 !       status returned by $OPEN
:  963          1034  1 !       status returned by $CRMPSC
:  964          1035  1 !       SS$_NORMAL
:  965          1036  1 !
:  966          1037  1 ! SIDE EFFECTS:
:  967          1038  1 !
:  968          1039  1 !       NONE
:  969          1040  1 !--
:  970          1041  1
:  971          1042  2     BEGIN
:  972          1043  2
:  973          1044  2     OWN
:  974          1045  2
:  975          1046  2         TABLE_NAME : VECTOR[25,BYTE]
:  976          1047  2                      INITIAL(BYTE('TERM$TABLOC:TERMTABLE.EXE'));
:  977          1048  2
:  978          1049  2     LOCAL
:  979          1050  2         REQ_ADDR : VECTOR [2] INITIAL (%X'200',%X'200'),
:  980          1051  2                                    ! input address to $MGBLSC
:  981          1052  2         ACTUAL_ADDR : VECTOR [2],  ! retd address from $MGBLSC
:  982          1053  2         PRIVATE_FAB : $FAB (FOP = UFO, FAC=<GET>,SHR=<GET,UPI>),
:  983          1054  2         OPEN_STATUS,
:  984          1055  2         SECTION_STATUS;
:  985          1056  2
:  986          1057  2 !+
:  987          1058  2 ! Initialize some fields.  This can't be done at link time because
:  988          1059  2 ! it would require fixup vectors which would make the PSECT read/write
:  989          1060  2 ! which would prevent sharing in our shared image.
:  990          1061  2 !-
```

```
;  991          1062  2     PRIVATE_FAB[FAB$L_FNA]=TABLE_NAME;
;  992          1063  2     PRIVATE_FAB[FAB$B_FNS]=%ALLOCATION(TABLE_NAME);
;  993          1064  2
;  994          1065  2 !+
;  995          1066  2 ! Look in TERM$TABLOC for private copy of TERMTABLE.EXE.
;  996          1067  2 !-
;  997          1068  3     IF (OPEN_STATUS = $OPEN (FAB = PRIVATE_FAB))
;  998          1069  2     THEN
;  999          1070  3         BEGIN
; 1000          1071  3         !+
; 1001          1072  3         ! Found a private termtable.  Map to it so we can determine
; 1002          1073  3         ! if the requested terminal is defined there.
; 1003          1074  3         !-
; 1004        P 1075  4         IF NOT (SECTION_STATUS = $CRMPSC (INADR = REQ_ADDR,
; 1005        P 1076  4                                           CHAN = .PRIVATE_FAB [FAB$L_STV],
; 1006        P 1077  4                                           FLAGS = SEC$M_EXPREG,
; 1007          1078  4                                           RETADR = ACTUAL_ADDR))
; 1008          1079  3         THEN
; 1009          1080  3             RETURN (.SECTION_STATUS);
; 1010          1081  3
; 1011          1082  3         PRIV_SECTION_ADDR = .ACTUAL_ADDR [0];
; 1012          1083  3                                           ! save mapped addr
; 1013          1084  3         PRIV_SECTION_ADDR_END = .ACTUAL_ADDR [1];
; 1014          1085  3                                           ! also save end addr
; 1015          1086  3         PRIV_SECTION_CHANNEL = .PRIVATE_FAB [FAB$L_STV];
; 1016          1087  3                                           ! save channel for $DASSGN
; 1017          1088  3
; 1018          1089  3         END
; 1019          1090  2     ELSE
; 1020          1091  2         RETURN (.OPEN_STATUS);
; 1021          1092  2
; 1022          1093  2     RETURN (SS$_NORMAL);
; 1023          1094  1     END;                              ! end of routine MAP_PRIV_SECTION


                                                            .PSECT  _SMG$DATA,NOEXE,  PIC,2

                                                    00017           .BLKB   1
52  45  54  3A  43  4F  4C  42  41  54  24  4D  52  45  54  00018 TABLE_NAME:
                                                            .ASCII  \TERM$TABLOC:TERMTABLE.EXE\
                    45  58  45  2E  45  4C  42  41  54  4D  00027

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

                                                    003ED           .BLKB   3
                                                 03 003F0 P.AAB:    .BYTE   3
                                                 50 003F1           .BYTE   80
                                               0000 003F2           .WORD   0
                                         00020000 003F4             .LONG   131072
                                         00000000 003F8             .LONG   0
                                         00000000 003FC             .LONG   0
                                         00000000 00400             .LONG   0
                                               0000 00404           .WORD   0
                                                 02 00406           .BYTE   2
                                                 42 00407           .BYTE   66
                                         00000000 00408             .LONG   0
```

```
                                00  0040C          .BYTE   0                                                       ;
                                00  0040D          .BYTE   0                                                       ;
                                00  0040E          .BYTE   0                                                       ;
                                02  0040F          .BYTE   2                                                       ;
                          00000000  00410          .LONG   0                                                       ;
                          00000000  00414          .LONG   0                                                       ;
                          00000000  00418          .LONG   0                                                       ;
                          00000000  0041C          .LONG   0                                                       ;
                          00000000  00420          .LONG   0                                                       ;
                                00  00424          .BYTE   0                                                       ;
                                00  00425          .BYTE   0                                                       ;
                              0000  00426          .WORD   0                                                       ;
                          00000000  00428          .LONG   0                                                       ;
                              0000  0042C          .WORD   0                                                       ;
                                00  0042E          .BYTE   0                                                       ;
                                00  0042F          .BYTE   0                                                       ;
                          00000000  00430          .LONG   0                                                       ;
                          00000000  00434          .LONG   0                                                       ;
                              0000  00438          .WORD   0                                                       ;
                                00  0043A          .BYTE   0                                                       ;
                                00  0043B          .BYTE   0                                                       ;
                          00000000  0043C          .LONG   0                                                       ;

                                                   .EXTRN  SYS$OPEN, SYS$CRMPSC

                                    007C 00000 MAP_PRIV_SECTION:
                                                   .WORD   Save R2,R3,R4,R5,R6                                     ; 1006
                    56 00000000'    EF   9E 00002  MOVAB   TABLE_NAME, R6
                    5E          A0  AE   9E 00009  MOVAB   -96(SP), SP
                 58     AE      0200 8F   3C 0000D  MOVZWL  #512, REQ_ADDR                                         ; 1042
                 5C     AE      0200 8F   3C 00013  MOVZWL  #512, REQ_ADDR+4
              6E 91     AF      0050 8F   28 00019  MOVC3   #80, P.AAB, PRIVATE_FAB                                ; 1053
                 2C     AE           66   9E 00020  MOVAB   TABLE_NAME, PRIVATE_FAB+44                            ; 1062
                 34     AE           19   90 00024  MOVB    #25, PRIVATE_FAB+52                                   ; 1063
                                     5E   DD 00028  PUSHL   SP                                                    ; 1068
                    00000000G    00  01   FB 0002A  CALLS   #1, SYS$OPEN
                                 35  50   E9 00031  BLBC    OPEN_STATUS, 1$
                                     7E   7C 00034  CLRQ    -(SP)                                                 ; 1078
                                     7E   7C 00036  CLRQ    -(SP)
                                  1C AE   DD 00038  PUSHL   PRIVATE_FAB+12
                                     7E   7C 0003B  CLRQ    -(SP)
                                     7E   D4 0003D  CLRL    -(SP)
                                00020000 8F   DD 0003F  PUSHL   #131072
                                     7E   D4 00045  CLRL    -(SP)
                                  78 AE   9F 00047  PUSHAB  ACTUAL_ADDR
                                  F8 AD   9F 0004A  PUSHAB  REQ_ADDR
                    00000000G    00  0C   FB 0004D  CALLS   #12, SYS$CRMPSC
                                 12  50   E9 00054  BLBC    SECTION_STATUS, 1$
                       EC AE     50 AE   D0 00057  MOVL    ACTUAL_ADDR, PRIV_SECTION_ADDR                        ; 1082
                       F0 AE     54 AE   D0 0005C  MOVL    ACTUAL_ADDR+4, PRIV_SECTION_ADDR_END                  ; 1084
                       F4 AE     0C AE   D0 00061  MOVL    PRIVATE_FAB+12, PRIV_SECTION_CHANNEL                  ; 1086
                                 50      01 D0 00066  MOVL    #1, R0                                                ; 1093
                                             04 00069 1$:   RET                                                   ; 1094

; Routine Size:  106 bytes,    Routine Base:  _SMG$CODE + 0440
```

; 1024 1095 1

```
;  1026          1096  1 %SBTTL 'MAP_GBL_SECTION - Map global section'
;  1027          1097  1 ROUTINE MAP_GBL_SECTION =
;  1028          1098  1
;  1029          1099  1 !++
;  1030          1100  1 ! FUNCTIONAL DESCRIPTION:
;  1031          1101  1 !
;  1032          1102  1 !       This routine maps the global section SMG$TERMTABLE into
;  1033          1103  1 !       the caller's virtual address space.
;  1034          1104  1 !
;  1035          1105  1 ! CALLING SEQUENCE:
;  1036          1106  1 !
;  1037          1107  1 !       ret_status.wlc.v = MAP_GBL_SECTION ()
;  1038          1108  1 !
;  1039          1109  1 ! FORMAL PARAMETERS:
;  1040          1110  1 !
;  1041          1111  1 !       NONE
;  1042          1112  1 !
;  1043          1113  1 ! IMPLICIT INPUTS:
;  1044          1114  1 !
;  1045          1115  1 !       NONE
;  1046          1116  1 !
;  1047          1117  1 ! IMPLICIT OUTPUTS:
;  1048          1118  1 !
;  1049          1119  1 !       NONE
;  1050          1120  1 !
;  1051          1121  1 ! COMPLETION STATUS:
;  1052          1122  1 !
;  1053          1123  1 !       status returned by $MGBLSC
;  1054          1124  1 !       SS$_NORMAL
;  1055          1125  1 !
;  1056          1126  1 ! SIDE EFFECTS:
;  1057          1127  1 !
;  1058          1128  1 !       NONE
;  1059          1129  1 !--
;  1060          1130  1
;  1061          1131  2     BEGIN
;  1062          1132  2     LOCAL
;  1063          1133  2         REQ_ADDR : VECTOR [2] INITIAL (%X'200',%X'200'),
;  1064          1134  2                                     ! input address to $MGBLSC
;  1065          1135  2         ACTUAL_ADDR : VECTOR [2],   ! retd address from $MGBLSC
;  1066          1136  2         MAP_STATUS,                 ! status retd by $MGBLSC
;  1067          1137  2         GBL_NAME_DESC : BLOCK [8,BYTE]; ! desc for global section name                                  ; R
;  1068          1138  2
;  1069          1139  2     GBL_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1070          1140  2     GBL_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  1071          1141  2     GBL_NAME_DESC [DSC$W_LENGTH] = %CHARCOUNT ('SMG$TERMTABLE');
;  1072          1142  2     GBL_NAME_DESC [DSC$A_POINTER] = UPLIT ('SMG$TERMTABLE');
;  1073          1143  2
;  1074        P 1144  2     MAP_STATUS = $MGBLSC (INADR = REQ_ADDR,
;  1075        P 1145  2                           RETADR = ACTUAL_ADDR,
;  1076        P 1146  2                           FLAGS = SEC$M_SYSGBL + SEC$M_PERM + SEC$M_EXPREG,
;  1077          1147  2                           GSDNAM = GBL_NAME_DESC);
;  1078          1148  2     IF NOT .MAP_STATUS
;  1079          1149  2     THEN
;  1080          1150  2         RETURN .MAP_STATUS;
;  1081          1151  2
;  1082          1152  2     GBL_SECTION_ADDR = .ACTUAL_ADDR [0];
```

```
; 1083          1153  2
; 1084          1154  2     RETURN (SS$_NORMAL);
; 1085          1155  1     END;                                    . end of routine MAP_GBL_SECTION


                                                          004AA             .BLKB   2
00  00  45  4C  42  41  54  4D  52  45  54  24  47  4D  53  004AC P.AAC:     .ASCII  \SMG$TERMTABLE\<0><0><0>                    ;
                                                        00  004BB                                                             ;

                                                                            .EXTRN  SYS$MGBLSC

                                                    0000 00000 MAP_GBL_SECTION:
                                                                            .WORD   Save nothing                                ; 1097
                                        5E                  14  C2 00002   SUBL2   #20, SP                                    ;
                                    0C  AE          0200    8F  3C 00005   MOVZWL  #512, REQ_ADDR                             ; 1131
                                    10  AE          0200    8F  3C 0000B   MOVZWL  #512, REQ_ADDR+4                           ;
                                            010E0C0D    8F  DD 00011   PUSHL   #17694733                                   ; 1141
                                    04  AE            D6    AF  9E 00017   MOVAB   P.AAC, GBL_NAME_DESC+4                     ; 1142
                                                        7E  7C 0001C   CLRQ    -(SP)                                       ; 1147
                                                  08    AE  9F 0001E   PUSHAB  GBL_NAME_DESC                               ;
                                             0002C000   8F  DD 00021   PUSHL   #180224                                     ;
                                                        7E  D4 00027   CLRL    -(SP)                                       ;
                                                  1C    AE  9F 00029   PUSHAB  ACTUAL_ADDR                                 ;
                                                  28    AE  9F 0002C   PUSHAB  REQ_ADDR                                    ;
                                00000000G   00          07  FB 0002F   CALLS   #7, SYS$MGBLSC                             ;
                                            0B          50  E9 00036   BLBC    MAP_STATUS, 1$                              ; 1148
                                00000000'   EF    08    AE  D0 00039   MOVL    ACTUAL_ADDR, GBL_SECTION_ADDR               ; 1152
                                            50            01  D0 00041   MOVL    #1, R0                                      ; 1154
                                                        04  00044 1$:     RET                                                 ; 1155

; Routine Size:  69 bytes,    Routine Base:  _SMG$CODE + 04BC

; 1086          1156  1
```

```
;  1088          1157  1 %SBTTL 'SEARCH_FOR_TERM_DEF - Search for terminal definition'
;  1089          1158  1 ROUTINE SEARCH_FOR_TERM_DEF(
;  1090          1159  1                            TERM_TAB : REF BLOCK [,BYTE],
;  1091          1160  1                            TERM_NAME : REF BLOCK [,BYTE],
;  1092          1161  1                            RET_TERM_DEF     ) =
;  1093          1162  1
;  1094          1163  1 !++
;  1095          1164  1 ! FUNCTIONAL DESCRIPTION:
;  1096          1165  1 !
;  1097          1166  1 !       This routine searches a Termtable.exe for a particular
;  1098          1167  1 !       terminal definition.
;  1099          1168  1 !
;  1100          1169  1 ! CALLING SEQUENCE:
;  1101          1170  1 !
;  1102          1171  1 !       found.wl.v = SEARCH_FOR_TERM_DEF (TERM_TAB.rl.v,
;  1103          1172  1 !                                         TERM_NAME.rt.dx,
;  1104          1173  1 !                                         RET_TERM_DEF.wl.r)
;  1105          1174  1 !
;  1106          1175  1 ! FORMAL PARAMETERS:
;  1107          1176  1 !
;  1108          1177  1 !       TERM_TAB                address where TERMTABLE.EXE starts
;  1109          1178  1 !
;  1110          1179  1 !       TERM_NAME               name of the terminal for which we
;  1111          1180  1 !                               will search
;  1112          1181  1 !
;  1113          1182  1 !       RET_TERM_DEF            returned terminal definition address,
;  1114          1183  1 !                               if found
;  1115          1184  1 !
;  1116          1185  1 ! IMPLICIT INPUTS:
;  1117          1186  1 !
;  1118          1187  1 !       NONE
;  1119          1188  1 !
;  1120          1189  1 ! IMPLICIT OUTPUTS:
;  1121          1190  1 !
;  1122          1191  1 !       NONE
;  1123          1192  1 !
;  1124          1193  1 ! COMPLETION STATUS:
;  1125          1194  1 !
;  1126          1195  1 !       1 = terminal definition not found
;  1127          1196  1 !       0 = terminal definition was found
;  1128          1197  1 !
;  1129          1198  1 ! SIDE EFFECTS:
;  1130          1199  1 !
;  1131          1200  1 !       NONE
;  1132          1201  1 !--
;  1133          1202  1
;  1134          1203  2     BEGIN
;  1135          1204  2     LOCAL
;  1136          1205  2         FOUND : INITIAL (1),            ! flag to indicate end of search
;  1137          1206  2         BLOCK_NUMBER : BYTE,
;  1138          1207  2         INDEX_NAME_DESC : BLOCK [8,BYTE],
;  1139          1208  2         INDEX_NAME_COUNT : REF VECTOR [,BYTE],
;  1140          1209  2         INDEX_NAME_STRING : REF VECTOR [,BYTE];
;  1141          1210  2
;  1142          1211  2
;  1143          1212  2 !+
;  1144          1213  2 ! Each TERMTABLE.EXE contains an index of all the terminals defined.
```

```
;  1145          1214  2 ! Search the terminal index for the requested name.
;  1146          1215  2 !-
;  1147          1216  2
;  1148          1217  2     INDEX_NAME_COUNT = .TERM_TAB + .TERM_TAB [TTB_L_INDEX_OFFSET];
;  1149          1218  2                                       ! point to 1st count
;  1150          1219  2                                       ! (address plus offset to get to
;  1151          1220  2                                       !  start of index - count is 1st byte)
;  1152          1221  2     INDEX_NAME_STRING = .INDEX_NAME_COUNT + 1;
;  1153          1222  2                                       ! point to 1st name string in index
;  1154          1223  2
;  1155          1224  2     IF .TERM_TAB [TTB_W_IDENT] NEQ 1
;  1156          1225  2     THEN
;  1157          1226  2         RETURN (SMG$_TABID_MIS);        ! Termtable Id Mismatch
;  1158          1227  2
;  1159          1228  2     INDEX_NAME_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
;  1160          1229  2     INDEX_NAME_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
;  1161          1230  2
;  1162          1231  2     WHILE .FOUND NEQ K_MATCH DO
;  1163          1232  3         BEGIN
;  1164          1233  3         !+
;  1165          1234  3         ! Loop through all terminal names in the section
;  1166          1235  3         ! index.  Exit if we find a match.
;  1167          1236  3         !-
;  1168          1237  3         INDEX_NAME_DESC [DSC$A_POINTER] = INDEX_NAME_STRING [0];
;  1169          1238  3         INDEX_NAME_DESC [DSC$W_LENGTH] = .INDEX_NAME_COUNT [0];
;  1170          1239  3
;  1171          1240  3         FOUND = STR$CASE_BLIND_COMPARE (.TERM_NAME, INDEX_NAME_DESC);
;  1172          1241  3
;  1173          1242  3         IF .FOUND EQL K_MATCH
;  1174          1243  3         THEN
;  1175          1244  3             EXITLOOP
;  1176          1245  3         ELSE
;  1177          1246  4             BEGIN
;  1178          1247  4             INDEX_NAME_COUNT = INDEX_NAME_STRING [0] + .INDEX_NAME_COUNT [0] + 1;
;  1179          1248  4                                 ! point past the previous string & block number
;  1180          1249  4             IF .INDEX_NAME_COUNT [0] EQL 0
;  1181          1250  4             THEN
;  1182          1251  4                 EXITLOOP;       ! return - no more terminals
;  1183          1252  4             INDEX_NAME_STRING = .INDEX_NAME_COUNT + 1;
;  1184          1253  4                                 ! point to next name string
;  1185          1254  3             END;
;  1186          1255  3
;  1187          1256  2         END;                                ! end of WHILE loop
;  1188          1257  2
;  1189          1258  2     !+
;  1190          1259  2     ! If we found a match, use the block number stored with the index
;  1191          1260  2     ! name to calculate the address of where the terminal definition
;  1192          1261  2     ! starts.
;  1193          1262  2     !-
;  1194          1263  2
;  1195          1264  2     IF .FOUND EQL K_MATCH
;  1196          1265  2     THEN
;  1197          1266  3         BEGIN
;  1198          1267  3         BLOCK_NUMBER = .(INDEX_NAME_STRING [0] + .INDEX_NAME_COUNT [0]);
;  1199          1268  3                                     ! block # follows counted name string
;  1200          1269  3         .RET_TERM_DEF = .TERM_TAB + ((.BLOCK_NUMBER - 1) * 512);
;  1201          1270  2     END;                                ! end of search for requested name
```

```
;  1202          1271  2
;  1203          1272  2
;  1204          1273  2     RETURN (.FOUND);
;  1205          1274  1     END;                                ! end of SEARCH_FOR_TERM_DEF


                                        003C 00000 SEARCH_FOR_TERM_DEF:
                                                   .WORD   Save R2,R3,R4,R5                          ; 1158
                              5E                08  C2 00002          SUBL2   #8, SP                  ;
                              50                01  D0 00005          MOVL    #1, FOUND               ; 1203
                              55          04    AC  D0 00008          MOVL    TERM_TAB, R5            ; 1217
                    53        55          04    A5  C1 0000C          ADDL3   4(R5), R5, INDEX_NAME_COUNT
                              52          01    A3  9E 00011          MOVAB   1(R3), INDEX_NAME_STRING ; 1221
                              01                65  B1 00015          CMPW    (R5), #1                ; 1224
                                                0B  13 00018          BEQL    1$
                              51 00000000G      00  9E 0001A          MOVAB   SMG$_TABID_MIS, R1      ; 1226
                              50                51  D0 00021          MOVL    R1, R0
                                                    04 00024          RET                             ;
                        02    AE        010E    8F  B0 00025 1$:      MOVW    #270, INDEX_NAME_DESC+2 ; 1228
                                                50  D5 0002B 2$:      TSTL    FOUND                   ; 1231
                                                29  13 0002D          BEQL    3$                      ;
                        04    AE                52  D0 0002F          MOVL    INDEX_NAME_STRING, INDEX_NAME_DESC+4 ; 1237
                              54                63  9A 00033          MOVZBL  (INDEX_NAME_COUNT), R4  ; 1238
                              6E                54  B0 00036          MOVW    R4, INDEX_NAME_DESC
                                                5E  DD 00039          PUSHL   SP                      ; 1240
                                          08    AC  DD 0003B          PUSHL   TERM_NAME               ;
                  00000000G   00                02  FB 0003E          CALLS   #2, STR$CASE_BLIND_COMPARE
                                                50  D5 00045          TSTL    FOUND                   ; 1242
                                                0F  13 00047          BEQL    3$                      ;
                              53          01 A442  9E 00049          MOVAB   1(R4)[INDEX_NAME_STRING], INDEX_NAME_COUNT ; 1247
                                                63  95 0004E          TSTB    (INDEX_NAME_COUNT)      ; 1249
                                                06  13 00050          BEQL    3$                      ;
                              52          01    A3  9E 00052          MOVAB   1(R3), INDEX_NAME_STRING ; 1252
                                                D3  11 00056          BRB     2$                      ; 1231
                                                50  D5 00058 3$:      TSTL    FOUND                   ; 1264
                                                15  12 0005A          BNEQ    4$                      ;
                              51                63  9A 0005C          MOVZBL  (INDEX_NAME_COUNT), R1  ; 1267
                              54              6142  90 0005F          MOVB    (R1)[INDEX_NAME_STRING], BLOCK_NUMBER
                              51                54  9A 00063          MOVZBL  BLOCK_NUMBER, R1        ; 1269
                    51        51                09  78 00066          ASHL    #9, R1, R1              ;
                        0C    BC      FE00 C145  9E 0006A          MOVAB   -512(R1)[R5], @RET_TERM_DEF
                                                04 00071 4$:       RET                             ; 1274

; Routine Size:  114 bytes,    Routine Base:  _SMG$CODE + 0501
```

```
;  1207          1275  1 %SBTTL 'SEARCH_FOR_TERM_DEF_BY_TYPE - search for term def given device type'
;  1208          1276  1 ROUTINE SEARCH_FOR_TERM_DEF_BY_TYPE (
;  1209          1277  1                         TERM_TAB : REF BLOCK [,BYTE],
;  1210          1278  1                         TERM_TYPE : SIGNED,
;  1211          1279  1                         RET_TERM_DEF,
;  1212          1280  1                         RET_TERM_NAME    ) =
;  1213          1281  1
;  1214          1282  1 !++
;  1215          1283  1 ! FUNCTIONAL DESCRIPTION:
;  1216          1284  1 !
;  1217          1285  1 !       This routine searches a Termtable.exe for a particular
;  1218          1286  1 !       terminal definition.  It will look in all terminal definitions
;  1219          1287  1 !       for the given device type.
;  1220          1288  1 !
;  1221          1289  1 ! CALLING SEQUENCE:
;  1222          1290  1 !
;  1223          1291  1 !       found.wl.v = SEARCH_FOR_TERM_DEF_BY_TYPE (TERM_TAB.rl.v,
;  1224          1292  1 !                                                 TERM_TYPE.rl.v,
;  1225          1293  1 !                                                 RET_TERM_DEF.wl.r,
;  1226          1294  1 !                                                 RET_TERM_NAME.wl.r)
;  1227          1295  1 !
;  1228          1296  1 ! FORMAL PARAMETERS:
;  1229          1297  1 !
;  1230          1298  1 !       TERM_TAB                address where TERMTABLE.EXE starts
;  1231          1299  1 !
;  1232          1300  1 !       TERM_TYPE               type of the terminal for which we
;  1233          1301  1 !                               will search (signed longword value)
;  1234          1302  1 !
;  1235          1303  1 !       RET_TERM_DEF            returned terminal definition address,
;  1236          1304  1 !                               if found
;  1237          1305  1 !
;  1238          1306  1 !       RET_TERM_NAME           address of name associated with this device type
;  1239          1307  1 !                               (from the definition)
;  1240          1308  1 !
;  1241          1309  1 ! IMPLICIT INPUTS:
;  1242          1310  1 !
;  1243          1311  1 !       NONE
;  1244          1312  1 !
;  1245          1313  1 ! IMPLICIT OUTPUTS:
;  1246          1314  1 !
;  1247          1315  1 !       NONE
;  1248          1316  1 !
;  1249          1317  1 ! COMPLETION STATUS:
;  1250          1318  1 !
;  1251          1319  1 !       1 = terminal definition not found
;  1252          1320  1 !       0 = terminal definition was found
;  1253          1321  1 !
;  1254          1322  1 ! SIDE EFFECTS:
;  1255          1323  1 !
;  1256          1324  1 !       NONE
;  1257          1325  1 !--
;  1258          1326  1
;  1259          1327  2     BEGIN
;  1260          1328  2     LOCAL
;  1261          1329  2         FOUND : INITIAL (1),            ! flag to indicate end of search
;  1262          1330  2         LOCAL_TYPE : SIGNED,            ! working terminal type
;  1263          1331  2         CUR_TERM_DEF : REF VECTOR [,WORD], ! offset pointers are actually at
```

```
;  1264          1332  2                                          ! beginning of definition
;  1265          1333  2          INDEX_NAME_COUNT : REF VECTOR [,BYTE],
;  1266          1334  2          INDEX_NAME_STRING : REF VECTOR [,BYTE];
;  1267          1335  2
;  1268          1336  2
;  1269          1337  2      !+
;  1270          1338  2      ! Each TERMTABLE.EXE contains an index of all the terminals defined.
;  1271          1339  2      ! Look at the definition of every terminal in the index, until we
;  1272          1340  2      ! find one with the requested device type.
;  1273          1341  2      !-
;  1274          1342  2
;  1275          1343  2          INDEX_NAME_COUNT = .TERM_TAB + .TERM_TAB [TTB_L_INDEX_OFFSET];
;  1276          1344  2                                          ! point to 1st count
;  1277          1345  2                                          ! (address plus offset to get to
;  1278          1346  2                                          !  start of index - count is 1st byte)
;  1279          1347  2          INDEX_NAME_STRING = .INDEX_NAME_COUNT + 1;
;  1280          1348  2                                          ! point to 1st name string in index
;  1281          1349  2
;  1282          1350  2          IF .TERM_TAB [TTB_W_IDENT] NEQ 1
;  1283          1351  2          THEN
;  1284          1352  2              RETURN (SMG$_TABID_MIS);        ! Termtable Id Mismatch
;  1285          1353  2
;  1286          1354  2      !+
;  1287          1355  2      ! If this definition is in a private section, the type number was biased.
;  1288          1356  2      ! Since the unbiased number is what is actually in the file, remove the
;  1289          1357  2      ! bias before doing any comparisons.
;  1290          1358  2      !-
;  1291          1359  2
;  1292          1360  2          LOCAL_TYPE = .TERM_TYPE;
;  1293          1361  2          $UNBIAS_TYPE_NUMBER (.TERM_TAB, LOCAL_TYPE);
;  1294          1362  2
;  1295          1363  2          WHILE .FOUND NEQ K_MATCH DO
;  1296          1364  3              BEGIN
;  1297          1365  3              LOCAL
;  1298          1366  3                  BLOCK_NUMBER : BYTE,
;  1299          1367  3                  DEF_DEVICE : REF VECTOR [,BYTE,SIGNED];
;  1300          1368  3                                          ! points to term number cap
;  1301          1369  3              !+
;  1302          1370  3              ! Loop through all terminal names in the section
;  1303          1371  3              ! index.  Exit if we find a match.
;  1304          1372  3              !-
;  1305          1373  3              BLOCK_NUMBER = .(INDEX_NAME_STRING [0] + .INDEX_NAME_COUNT [0]);
;  1306          1374  3                                          ! block # follows counted name string
;  1307          1375  3              CUR_TERM_DEF = .TERM_TAB + ((.BLOCK_NUMBER - 1) * 512);
;  1308          1376  3                                          ! get def for this index entry
;  1309          1377  3              DEF_DEVICE = .CUR_TERM_DEF [SMG$K_VMS_TERMINAL_NUMBER] +
;  1310          1378  3                           .CUR_TERM_DEF;
;  1311          1379  3                                          ! point to device type in definition
;  1312          1380  3              !+
;  1313          1381  3              ! Device type is stored as a signed longword.  Skip past the
;  1314          1382  3              ! size and type bytes to the actual data.
;  1315          1383  3              !-
;  1316          1384  3              IF .DEF_DEVICE [2] EQL .LOCAL_TYPE
;  1317          1385  3              THEN
;  1318          1386  4                  BEGIN
;  1319          1387  4                  FOUND = K_MATCH;
;  1320          1388  4                  EXITLOOP
```

```
: 1321          1389  4              END
: 1322          1390  3          ELSE
: 1323          1391  4              BEGIN
: 1324          1392  4              INDEX_NAME_COUNT = INDEX_NAME_STRING [0] + .INDEX_NAME_COUNT [0] +
: 1325          1393  4                                 1;
: 1326          1394  4                                 ! point past the previous string & block number
: 1327          1395  4              IF .INDEX_NAME_COUNT [0] EQL 0
: 1328          1396  4              THEN
: 1329          1397  4                  EXITLOOP;      ! return - no more terminals
: 1330          1398  4              INDEX_NAME_STRING = .INDEX_NAME_COUNT + 1;
: 1331          1399  4                                 ! point to next name string
: 1332          1400  3              END;
: 1333          1401  3
: 1334          1402  2          END;                              ! end of WHILE loop
: 1335          1403  2
: 1336          1404  2 !+
: 1337          1405  2 ! If we found a match, set the return name and return definition address.
: 1338          1406  2 !-
: 1339          1407  2
: 1340          1408  2      IF .FOUND EQL K_MATCH
: 1341          1409  2      THEN
: 1342          1410  3          BEGIN
: 1343          1411  3          .RET_TERM_DEF = .CUR_TERM_DEF;
: 1344          1412  3          .RET_TERM_NAME = .CUR_TERM_DEF + .CUR_TERM_DEF [SMG$K_NAME];
: 1345          1413  3                                 ! address of counted string
: 1346          1414  2          END;
: 1347          1415  2
: 1348          1416  2      RETURN (.FOUND);
: 1349          1417  1      END;                        ! end of SEARCH_FOR_TERM_DEF_BY_TYPE


                                01FC 00000 SEARCH_FOR_TERM_DEF_BY_TYPE:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8                    : 1276
                58 00000000'    EF   9E 00002      MOVAB   PRIV_SECTION_ADDR, R8                :
                57              01   D0 00009      MOVL    #1, FOUND                            : 1327
                55        04    AC   D0 0000C      MOVL    TERM_TAB, R5                         : 1343
        51      55        04    A5   C1 00010      ADDL3   4(R5), R5, INDEX_NAME_COUNT          :
                50        01    A1   9E 00015      MOVAB   1(R1), INDEX_NAME_STRING             : 1347
                01              65   B1 00019      CMPW    (R5), #1                             : 1350
                                0B   13 0001C      BEQL    1$                                   :
                52 00000000G    00   9E 0001E      MOVAB   SMG$_TABID_MIS, R2                   : 1352
                50              52   D0 00025      MOVL    R2, R0                               :
                                04   0C028        RET                                          :
                56        08    AC   D0 00029 1$:  MOVL    TERM_TYPE, LOCAL_TYPE                : 1360
                                68   D5 0002D      TSTL    PRIV_SECTION_ADDR                    : 1361
                                0D   13 0002F      BEQL    2$                                   :
                68              55   D1 00031      CMPL    R5, PRIV_SECTION_ADDR                :
                                08   19 00034      BLSS    2$                                   :
                                56   D5 00036      TSTL    LOCAL_TYPE                           :
                                04   18 00038      BGEQ    2$                                   :
                56        41    A6   9E 0003A      MOVAB   65(R6), LOCAL_TYPE                   :
                                57   D5 0003E 2$:  TSTL    FOUND                                : 1363
                                37   13 00040      BEQL    4$                                   :
                54              61   9A 00042      MOVZBL  (INDEX_NAME_COUNT), R4               : 1373
```

```
                                    52          6440  90 00045          MOVB    (R4)[INDEX_NAME_STRING], BLOCK_NUMBER
                                    52            52  9A 00049          MOVZBL  BLOCK_NUMBER, R2                                    ; 1375
                              52    52            09  78 0004C          ASHL    #9, R2, R2
                                    53      FE00 C245  9E 00050          MOVAB   -512(R2)[R5], CUR_TERM_DEF
                                    52      01C6   C3  3C 00056          MOVZWL  454(CUR_TERM_DEF), DEF_DEVICE                       ; 1378
                                    52            53  C0 0005B          ADDL2   CUR_TERM_DEF, DEF_DEVICE
                 56       02   A2  08            00  EC 0005E          CMPV    #0, #8, 2(DEF_DEVICE), LOCAL_TYPE                   ; 1384
                                                04  12 00064          BNEQ    3$
                                                57  D4 00066          CLRL    FOUND                                               ; 1387
                                                0F  11 00068          BRB     4$                                                  ; 1386
                                    51        01 A440  9E 0006A 3$:     MOVAB   1(R4)[INDEX_NAME_STRING], INDEX_NAME_COUNT          ; 1392
                                                61  95 0006F          TSTB    (INDEX_NAME_COUNT)                                  ; 1395
                                                06  13 00071          BEQL    4$
                                    50        01   A1  9E 00073          MOVAB   1(R1), INDEX_NAME_STRING                            ; 1398
                                                C5  11 00077          BRB     2$                                                  ; 1363
                                                57  D5 00079 4$:     TSTL    FOUND                                               ; 1408
                                                0E  12 0007B          BNEQ    5$
                          0C    BC            53  D0 0007D          MOVL    CUR_TERM_DEF, @RET_TERM_DEF                         ; 1411
                                    50      0456   C3  3C 00081          MOVZWL  1110(CUR_TERM_DEF), R0                              ; 1412
                    10    BC      53            50  C1 00086          ADDL3   R0, CUR_TERM_DEF, @RET_TERM_NAME
                                    50            57  D0 0008B 5$:     MOVL    FOUND, R0                                           ; 1416
                                                    04 0008E          RET                                                         ; 1417

; Routine Size:  143 bytes,    Routine Base:  _SMG$CODE + 0573
```

```
;1351     1418  1 %SBTTL 'DECODE_ARITH_STRING - Decode arithmetic data'
;1352     1419  1 ROUTINE DECODE_ARITH_STRING (ARITH_STRING : REF VECTOR [,BYTE],
;1353     1420  1                             USER_ARGS : REF VECTOR,
;1354     1421  1                             DECODED_STRING : REF VECTOR [,BYTE],
;1355     1422  1                             DECODED_LENGTH) =
;1356     1423  1
;1357     1424  1 !++
;1358     1425  1 ! FUNCTIONAL DESCRIPTION:
;1359     1426  1 !
;1360     1427  1 !       Arithmetic strings are stored as encoded data bytes.  Codes
;1361     1428  1 !       are used to indicate where an operand is, what operator to apply,
;1362     1429  1 !       etc.  The general format is number, op, number, op, number, ... etc.
;1363     1430  1 !
;1364     1431  1 !       In the midst of binary information, there may be text that must
;1365     1432  1 !       be included in the final string.
;1366     1433  1 !
;1367     1434  1 ! CALLING SEQUENCE:
;1368     1435  1 !
;1369     1436  1 !       status.rl.v = DECODE_ARITH_STRING (ARITH_STRING.rl.r,
;1370     1437  1 !                                          USER_ARGS.rl.r,
;1371     1438  1 !                                          DECODED_STRING.wl.r,
;1372     1439  1 !                                          DECODED_LENGTH.wl.r)
;1373     1440  1 !
;1374     1441  1 !
;1375     1442  1 ! FORMAL PARAMETERS:
;1376     1443  1 !
;1377     1444  1 !       ARITH_STRING.rl.r               Address of string to decode
;1378     1445  1 !
;1379     1446  1 !       USER_ARGS.rl.r                  Address of a vector containing
;1380     1447  1 !                                       args provided by caller
;1381     1448  1 !
;1382     1449  1 !       DECODED_STRING.wl.r             Address where decoded string
;1383     1450  1 !                                       should be placed
;1384     1451  1 !
;1385     1452  1 !       DECODED_LENGTH.wl.r             Length of final string
;1386     1453  1 !
;1387     1454  1 ! IMPLICIT INPUTS:
;1388     1455  1 !
;1389     1456  1 !       NONE
;1390     1457  1 !
;1391     1458  1 ! IMPLICIT OUTPUTS:
;1392     1459  1 !
;1393     1460  1 !       NONE
;1394     1461  1 !
;1395     1462  1 ! COMPLETION STATUS:
;1396     1463  1 !
;1397     1464  1 !       SUCCESS         String was decoded
;1398     1465  1 !       FAILURE         String could not be decoded
;1399     1466  1 !
;1400     1467  1 ! SIDE EFFECTS:
;1401     1468  1 !
;1402     1469  1 !       NONE
;1403     1470  1 !--
;1404     1471  2     BEGIN
;1405     1472  2     LOCAL
;1406     1473  2         ACCUMULATOR : INITIAL (0),      ! place to hold current result
;1407     1474  2         INDEX,                          ! index into arithmetic string
```

```
; 1408          1475  2         CURRENT_BYTE : INITIAL (0),     ! pointer into return string
; 1409          1476  2         ARG_COUNT : INITIAL (0),        ! number of user args that have
; 1410          1477  2                                         ! been used
; 1411          1478  2         ARITH_CAP_SIZE,                 ! length of entire capability string
; 1412          1479  2         SEEN : INITIAL (0);             ! flag to indicate 1st operand found
; 1413          1480  2
; 1414          1481  2     !+
; 1415          1482  2     ! Initialize return length.
; 1416          1483  2     !-
; 1417          1484  2
; 1418          1485  2         .DECODED_LENGTH = 0;
; 1419          1486  2
; 1420          1487  2     !+
; 1421          1488  2     ! Start with the 3rd byte of the string (skipping over the size and
; 1422          1489  2     ! type of the entire string).  This byte should be the data type of
; 1423          1490  2     ! the first item to process.  It could be one of the following:
; 1424          1491  2     !       positive number         text
; 1425          1492  2     !       k_operand               longword operand
; 1426          1493  2     !       k_<operator>            add, subtract, multiply, divide
; 1427          1494  2     !       k_store                 calculation complete, store result
; 1428          1495  2     !-
; 1429          1496  2
; 1430          1497  2         ARITH_CAP_SIZE = .ARITH_STRING [0] XOR %X'FFFFFF00';
; 1431          1498  2                                         ! sign extend byte
; 1432          1499  2         ARITH_CAP_SIZE = ABS (.ARITH_CAP_SIZE);
; 1433          1500  2                                         ! get length (which is stored negative)
; 1434          1501  2
; 1435          1502  2         INDEX = 2;                      ! start with 3rd byte
; 1436          1503  2
; 1437          1504  2         WHILE .INDEX LSS .ARITH_CAP_SIZE DO
; 1438          1505  3             BEGIN
; 1439          1506  3             LOCAL
; 1440          1507  3                 DATA_TYPE : BYTE,       ! code for kind of data to follow
; 1441          1508  3                 DATA_SIZE : BYTE,       ! relevant only for string data
; 1442          1509  3                 TEMP : REF VECTOR;      ! addr of operand awaiting operator
; 1443          1510  3
; 1444          1511  3             DATA_TYPE = .ARITH_STRING [.INDEX],
; 1445          1512  3
; 1446          1513  3             SELECTONE .DATA_TYPE OF
; 1447          1514  3             SET
; 1448          1515  3
; 1449          1516  3                 [.K_OPERAND]:           ! longword operand
; 1450          1517  4                     BEGIN
; 1451          1518  4                     IF NOT .SEEN
; 1452          1519  4                     THEN
; 1453          1520  5                         BEGIN           ! 1st operand seen goes in accum
; 1454          1521  5                         CH$MOVE (4, ARITH_STRING [.INDEX + 2], ACCUMULATOR);
; 1455          1522  5                         SEEN = 1;
; 1456          1523  5                         END
; 1457          1524  4                     ELSE
; 1458          1525  4                         TEMP = ARITH_STRING [.INDEX + 2];
; 1459          1526  4                                         ! remember location of operand
; 1460          1527  4                     INDEX = .INDEX + 6; ! get next code
; 1461          1528  3                     END;
; 1462          1529  3
; 1463          1530  3                 [.K_SUBSTITUTE]:        ! substitute argument from user
; 1464          1531  4                     BEGIN
```

```
;  1465          1532  4                  LOCAL
;  1466          1533  4                      ARG_NUMBER;
;  1467          1534  4                  !+
;  1468          1535  4                  ! Number following substitution symbol will tell us
;  1469          1536  4                  ! which argument from the vector to use.
;  1470          1537  4                  !-
;  1471          1538  4                  ARG_NUMBER = .ARITH_STRING [.INDEX+1];
;  1472          1539  4                  IF NOT .SEEN
;  1473          1540  4                  THEN
;  1474          1541  5                      BEGIN                   ! 1st operand found goes into accum
;  1475          1542  5                      ACCUMULATOR = .USER_ARGS [.ARG_NUMBER-1];
;  1476          1543  5                      SEEN = 1;
;  1477          1544  5                      END
;  1478          1545  4                  ELSE
;  1479          1546  4                      TEMP = USER_ARGS [.ARG_NUMBER-1];
;  1480          1547  4                                              ! remember location of operand
;  1481          1548  4                  INDEX = .INDEX + 2;         ! next code
;  1482          1549  3                  END;
;  1483          1550  3
;  1484          1551  3              [.K_ADD]:
;  1485          1552  4                  BEGIN                       ! add temp to accumulator
;  1486          1553  4                  ACCUMULATOR = .TEMP [0] + .ACCUMULATOR;
;  1487          1554  4                  TEMP = 0;                   ! don't use this operand again
;  1488          1555  4                  INDEX = .INDEX + 1;         ! next code
;  1489          1556  3                  END;
;  1490          1557  3
;  1491          1558  3              [.K_SUBTRACT]:
;  1492          1559  4                  BEGIN                       ! subtract temp from accumulator
;  1493          1560  4                  ACCUMULATOR = .ACCUMULATOR - .TEMP [0];
;  1494          1561  4                  TEMP = 0;                   ! don't reuse this operand
;  1495          1562  4                  INDEX = .INDEX + 1;         ! next code
;  1496          1563  3                  END;
;  1497          1564  3
;  1498          1565  3              [.K_MULTIPLY]:
;  1499          1566  4                  BEGIN                       ! multiply temp by accumulator
;  1500          1567  4                  ACCUMULATOR = .ACCUMULATOR * .TEMP [0];
;  1501          1568  4                  TEMP = 0;                   ! don't reuse this operand
;  1502          1569  4                  INDEX = .INDEX + 1;         ! next code
;  1503          1570  3                  END;
;  1504          1571  3
;  1505          1572  3              [.K_DIVIDE]:
;  1506          1573  4                  BEGIN                       ! divide accumulator by temp
;  1507          1574  4                  ACCUMULATOR = .ACCUMULATOR / .TEMP [0];
;  1508          1575  4                  TEMP = 0;                   ! don't reuse this operand
;  1509          1576  4                  INDEX = .INDEX + 1;         ! next code
;  1510          1577  3                  END;
;  1511          1578  3
;  1512          1579  3              [.K_STORE]:
;  1513          1580  4                  BEGIN                       ! store a calculation
;  1514          1581  4                  DECODED_STRING [.CURRENT_BYTE] = .ACCUMULATOR <0,8>;
;  1515          1582  4                                              ! store only as a byte
;  1516          1583  4                  CURRENT_BYTE = .CURRENT_BYTE + 1;
;  1517          1584  4                  ACCUMULATOR = 0;
;  1518          1585  4                  SEEN = 0;
;  1519          1586  4                  INDEX = .INDEX + 1;         ! next code
;  1520          1587  3                  END;
;  1521          1588  3
```

```
;  1522          1589  3              [OTHERWISE]:                    ! plain vanilla text
;  1523          1590  4                  BEGIN
;  1524          1591  4                  CH$MOVE (.ARITH_STRING [.INDEX],        ! size of text
;  1525          1592  4                           ARITH_STRING [.INDEX + 2],     ! skip over type byte
;  1526          1593  4                           DECODED_STRING [.CURRENT_BYTE]);
;  1527          1594  4                  CURRENT_BYTE = .CURRENT_BYTE + .ARITH_STRING [.INDEX];
;  1528          1595  4                  INDEX = .INDEX + .ARITH_STRING [.INDEX] + 2;
;  1529          1596  4                                                  ! move past string, count & type bytes
;  1530          1597  3                  END;
;  1531          1598  3          TES;
;  1532          1599  3
;  1533          1600  2          END;                                    ! end of WHILE LOOP
;  1534          1601  2
;  1535          1602  2      .DECODED_LENGTH = .CURRENT_BYTE;            ! return length of string
;  1536          1603  2
;  1537          1604  2      RETURN (SS$_NORMAL);
;  1538          1605  2
;  1539          1606  1      END;                                        ! end of routine DECODE_ARITH_STRING
;  INFO#250                 L1:1553
;  Referenced LOCAL symbol TEMP is probably not initialized



                                        0FFC 00000 DECODE_ARITH_STRING:
                                                   .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1419
                                             7E D4 00002   CLRL    ACCUMULATOR                           ; 1471
                                             58 D4 00004   CLRL    CURRENT_BYTE                          ;
                                             50 D4 00006   CLRL    ARG_COUNT                             ;
                                             5B D4 00008   CLRL    SEEN                                  ;
                                          10 BC D4 0000A   CLRL    @DECODED_LENGTH                       ; 1485
                                       56 04 AC D0 0000D   MOVL    ARITH_STRING, R6                      ; 1497
                                       5A    66 9A 00011   MOVZBL  (R6), ARITH_CAP_SIZE                  ;
                                       5A FFFFFF00 8F CC 00014   XORL2   #-256, ARITH_CAP_SIZE           ;
                                             03 18 0001B   BGEQ    1$                                    ; 1499
                                       5A    5A CE 0001D   MNEGL   ARITH_CAP_SIZE, ARITH_CAP_SIZE        ;
                                       57    02 D0 00020 1$: MOVL    #2, INDEX                           ; 1502
                                       5A    57 D1 00023 2$: CMPL    INDEX, ARITH_CAP_SIZE               ; 1504
                                             03 19 00026   BLSS    3$                                    ;
                                          00B5 31 00028   BRW     19$                                   ;
                                       59    6746 9A 0002B 3$: MOVZBL  (INDEX)[R6], R9                   ; 1511
                                       52    59 90 0002F   MOVB    R9, DATA_TYPE                         ;
00000000'                              EF    52 91 00032   CMPB    DATA_TYPE, K_OPERAND                  ; 1516
                                             19 12 00039   BNEQ    7$                                    ;
                                       0C    5B E8 0003B   BLBS    SEEN, 4$                              ; 1518
                                          02 A746 9F 0003E   PUSHAB  2(INDEX)[R6]                        ; 1521
                                       6E    9E D0 00042   MOVL    @(SP)+, ACCUMULATOR                   ;
                                       5B    01 D0 00045   MOVL    #1, SEEN                              ; 1522
                                             05 11 00048   BRB     5$                                    ; 1518
                                       51    02 A746 9E 0004A 4$: MOVAB   2(INDEX)[R6], TEMP            ; 1525
                                       57    06 C0 0004F 5$: ADDL2   #6, INDEX                          ; 1527
                                             CF 11 00052 6$: BRB     2$                                 ; 1513
00000000'                              EF    52 91 00054 7$: CMPB    DATA_TYPE, K_SUBSTITUTE            ; 1530
                                             1F 12 0005B   BNEQ    11$                                   ;
                                       50    01 A746 9A 0005D   MOVZBL  1(INDEX)[R6], ARG_NUMBER         ; 1538
                                       50    08 BC40 DE 00062   MOVAL   @USER_ARGS[ARG_NUMBER], R0       ; 1542
```

```
                                09              5B  E8 00067          BLBS    SEEN, 8$                                        ; 1539
                                6E        FC    A0  D0 0006A          MOVL    -4(R0), ACCUMULATOR                             ; 1542
                                5B              01  D0 0006E          MOVL    #1, SEEN                                        ; 1543
                                                04  11 00071          BRB     9$                                              ; 1539
                                51        FC    A0  9E 00073 8$:      MOVAB   -4(R0), TEMP                                    ; 1546
                                57              02  C0 00077 9$:      ADDL2   #2, INDEX                                       ; 1548
                                                A7  11 0007A 10$:     BRB     2$                                              ; 1513
                    00000000'   EF              52  91 0007C 11$:     CMPB    DATA_TYPE, K_ADD                                ; 1551
                                                05  12 00083          BNEQ    12$                                             ;
                                6E              61  C0 00085          ADDL2   (TEMP), ACCUMULATOR                             ; 1553
                                                28  11 00088          BRB     15$                                             ; 1554
                    00000000'   EF              52  91 0008A 12$:     CMPB    DATA_TYPE, K_SUBTRACT                           ; 1558
                                                05  12 00091          BNEQ    13$                                             ;
                                6E              61  C2 00093          SUBL2   (TEMP), ACCUMULATOR                             ; 1560
                                                1A  11 00096          BRB     15$                                             ; 1561
                    00000000'   EF              52  91 00098 13$:     CMPB    DATA_TYPE, K_MULTIPLY                           ; 1565
                                                05  12 0009F          BNEQ    14$                                             ;
                                6E              61  C4 000A1          MULL2   (TEMP), ACCUMULATOR                             ; 1567
                                                0C  11 000A4          BRB     15$                                             ; 1568
                    00000000'   EF              52  91 000A6 14$:     CMPB    DATA_TYPE, K_DIVIDE                             ; 1572
                                                07  12 000AD          BNEQ    16$                                             ;
                                6E              61  C6 000AF          DIVL2   (TEMP), ACCUMULATOR                             ; 1574
                                                51  D4 000B2 15$:     CLRL    TEMP                                            ; 1575
                                                14  11 000B4          BRB     17$                                             ; 1576
                    00000000'   EF              52  91 000B6 16$:     CMPB    DATA_TYPE, K_STORE                              ; 1579
                                                0F  12 000BD          BNEQ    18$                                             ;
                          0C BC48               6E  90 000BF          MOVB    ACCUMULATOR, @DECODED_STRING[CURRENT_BYTE]      ; 1581
                                                58  D6 000C4          INCL    CURRENT_BYTE                                    ; 1583
                                                6E  D4 000C6          CLRL    ACCUMULATOR                                     ; 1584
                                                5B  D4 000C8          CLRL    SEEN                                            ; 1585
                                                57  D6 000CA 17$:     INCL    INDEX                                           ; 1586
                                                84  11 000CC          BRB     6$                                              ; 1513
                    0C BC48         02 A746             59  28 000CE 18$:     MOVC3   R9, 2(INDEX)[R6], @DECODED_STRING-      ; 1593
                                                                                      [CURRENT_BYTE]                            ;
                                58              59  C0 000D6          ADDL2   R9, CURRENT_BYTE                                ; 1594
                                57        02 A947   9E 000D9          MOVAB   2(R9)[INDEX], INDEX                             ; 1595
                                                9A  11 000DE          BRB     10$                                             ; 1504
                                10    BC        58  D0 000E0 19$:     MOVL    CURRENT_BYTE, @DECODED_LENGTH                   ; 1602
                                50              01  D0 000E4          MOVL    #1, R0                                          ; 1604
                                                    04 000E7          RET                                                     ; 1606

; Routine Size:  232 bytes,    Routine Base:  _SMG$CODE + 0602
```

```
; 1541          1607  1     END                                     . end of module SMG$INTERFACE_TERM_TABLE
; 1542          1608  0     ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _SMG$DATA                      49  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;  _SMG$CODE                    1770  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        46         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        15        36          10       00:00.1
```

```
; Information:  1
; Warnings:     0
; Errors:       0
```

```
;                                       COMMAND QUALIFIERS

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

; Size:          1627 code + 192 data bytes
; Run Time:        00:33.8
; Elapsed Time:    01:40.2
; Lines/CPU Min:    2858
; Lexemes/CPU-Min: 18090
; Memory Used:  173 pages
; Compilation Complete
```

COM
REQ

SMGUSRTRM
LIS

SORLIB
REQ

SORTSHR
MAP

DEFS0
REQ

DKS
REQ

SORT32

SRTSPC
REQ

CDDTYPE
R32

SORTMERGE
MAP

SMGVECTOR
LIS

CHKPNT
REQ

CDDMAC
R32

SFKEYWRD
REQ

RECSYM
R32

SRTTRN
MAR

OPCODES
REQ