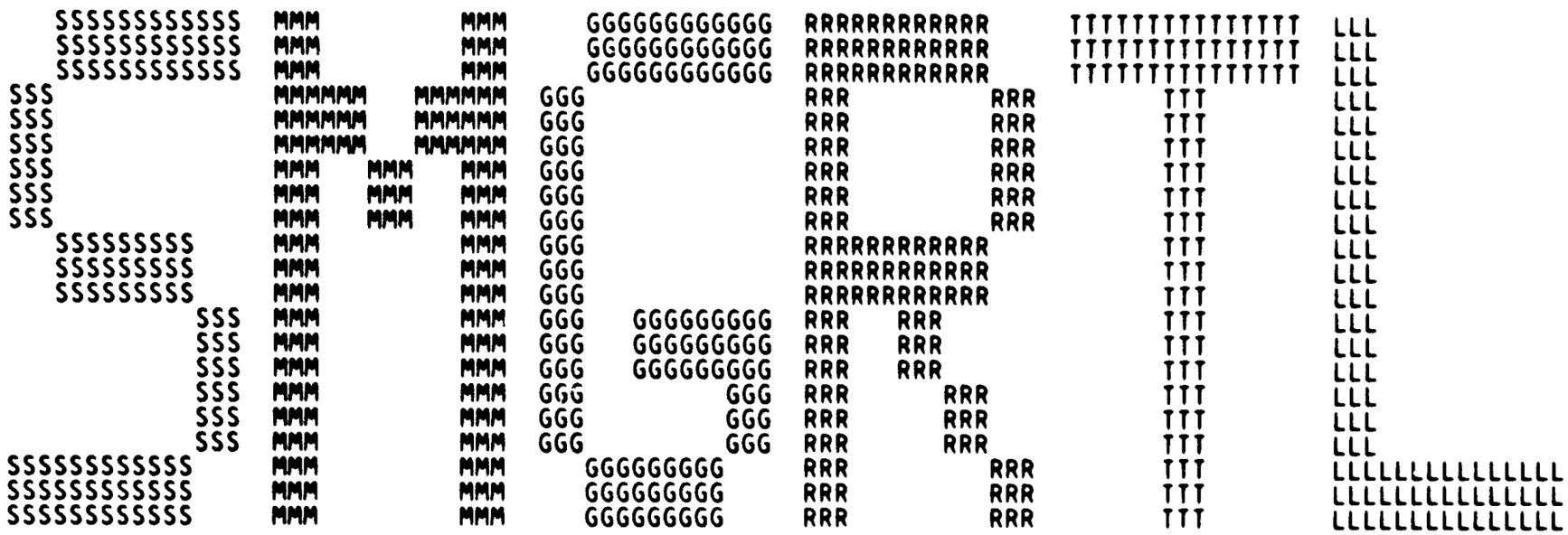

_$2

Val

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

```
  SSSSSSSS  MM      MM    GGGGGGGG  BBBBBBBB    LL          DDDDDDDD    TTTTTTTTTT  RRRRRRRR    MM      MM
  SSSSSSSS  MM      MM    GGGGGGGG  BBBBBBBB    LL          DDDDDDDD    TTTTTTTTTT  RRRRRRRR    MM      MM
SS          MMMM  MMMM  GG          BB      BB  LL          DD      DD      TT      RR      RR  MMMM  MMMM
SS          MMMM  MMMM  GG          BB      BB  LL          DD      DD      TT      RR      RR  MMMM  MMMM
SS          MM  MM  MM  GG          BB      BB  LL          DD      DD      TT      RR      RR  MM  MM  MM
SS          MM  MM  MM  GG          BB      BB  LL          DD      DD      TT      RR      RR  MM  MM  MM
  SSSSSS    MM      MM  GG          BBBBBBBB    LL          DD      DD      TT      RRRRRRRR    MM      MM
  SSSSSS    MM      MM  GG          BBBBBBBB    LL          DD      DD      TT      RRRRRRRR    MM      MM
        SS  MM      MM  GG  GGGGGG  BB      BB  LL          DD      DD      TT      RR  RR      MM      MM
        SS  MM      MM  GG  GGGGGG  BB      BB  LL          DD      DD      TT      RR  RR      MM      MM
        SS  MM      MM  GG      GG  BB      BB  LL          DD      DD      TT      RR    RR    MM      MM      ....
        SS  MM      MM  GG      GG  BB      BB  LL          DD      DD      TT      RR    RR    MM      MM      ....
SSSSSSSS    MM      MM    GGGGGG    BBBBBBBB    LLLLLLLLLL  DDDDDDDD        TT      RR      RR  MM      MM      ....
SSSSSSSS    MM      MM    GGGGGG    BBBBBBBB    LLLLLLLLLL  DDDDDDDD        TT      RR      RR  MM      MM      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE SMG$BUILD_TERM_TABLE( %TITLE 'Build terminal table'
;     2          0002  0                 MAIN = SMG$BUILD_TERM_TABLE,
;     3          0003  0                 IDENT = '1-002'         ! File: SMGBLDTAB.B32 Edit: PLL1002
;     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 the parser for the TERMTABLE.TXT terminal
;    37          0037  1 !       capabilities file.  It accepts an ascii TERMTABLE.TXT file as input and
;    38          0038  1 !       produces a binary capabilities file, TERMTABLE.EXE, as output.
;    39          0039  1 !
;    40          0040  1 !       This module is used in conjunction with the RTL Screen Management
;    41          0041  1 !       routines.
;    42          0042  1 !
;    43          0043  1 ! ENVIRONMENT:  User mode - AST reentrant
;    44          0044  1 !
;    45          0045  1 ! AUTHOR: P. Levesque CREATION DATE: 1-Nov-1983
;    46          0046  1 !
;    47          0047  1 ! MODIFIED BY:
;    48          0048  1 !
;    49          0049  1 ! 1-001 - Original.  PLL 1-Nov-1983
;    50          0050  1 ! 1-002 - For output file, use default extension .EXE and default name
;    51          0051  1 !         of TERMTABLE (so logical name may be used).  PLL 19-Mar-1984
;    52          0052  1 !--
;    53          0053  1
```

```
;    55          0054  1 %SBTTL 'Declarations'
;    56          0055  1 !
;    57          0056  1 ! SWITCHES:
;    58          0057  1 !
;    59          0058  1
;    60          0059  1 !
;    61          0060  1 ! LINKAGES:
;    62          0061  1 !
;    63          0062  1 !       NONE
;    64          0063  1 !
;    65          0064  1 ! TABLE OF CONTENTS:
;    66          0065  1 !
;    67          0066  1
;    68          0067  1 FORWARD ROUTINE
;    69          0068  1         SMG$BUILD_TERM_TABLE : NOVALUE,! mainline interpretter
;    70          0069  1         OPEN_FILES,                    ! open ascii & binary TERMTABLEs
;    71          0070  1         PARSE_TERM_DEFS,               ! parse a terminal definition & insert
;    72          0071  1                                        ! it into the binary file
;    73          0072  1         CLOSE_FILES,                   ! close ascii & binary TERMTABLEs
;    74          0073  1         CLOSE_WITH_DLT;                ! exit handler to close the .exe
;    75          0074  1                                        ! file with the delete bit set
;    76          0075  1
;    77          0076  1 !
;    78          0077  1 ! INCLUDE FILES:
;    79          0078  1 !
;    80          0079  1
;    81          0080  1 REQUIRE 'RTLIN:SMGPROLOG';             ! Defines psects, macros, etc.
;    82          0158  1
;    83          0159  1 LIBRARY 'RTLML:SMGTPALIB';             ! Definitions & macros used
;    84          0160  1                                        ! to construct TERMTABLE.EXE
;    85          0161  1 LIBRARY 'RTLTPAMAC';                   ! TPARSE library of macros
;    86          0162  1
;    87          0163  1 !
;    88          0164  1 ! MACROS:
;    89          0165  1 !
;    90          0166  1 !+
;    91          0167  1 ! If we get interupted by a control Y or control C, we don't want to leave
;    92          0168  1 ! an incomplete TERMTABLE.EXE in the user's directory.  Establish an exit
;    93          0169  1 ! handler that will close the file with the delete bit set.
;    94          0170  1 !-
;    95        M 0171  1 MACRO $DECLARE_EXIT_HANDLER =
;    96        M 0172  1 BEGIN
;    97        M 0173  1     EXIT_BLOCK [1] = CLOSE_WITH_DLT;     ! address of exit handler routine
;    98        M 0174  1     EXIT_BLOCK [2] = 2;                  ! number of args passed
;    99        M 0175  1     EXIT_BLOCK [3] = EXIT_REASON;        ! addr to hold reason
;   100        M 0176  1     EXIT_BLOCK [4] = .PARAM_BLOCK;       ! parameter for exit handler routine
;   101        M 0177  1
;   102        M 0178  1     $DCLEXH (DESBLK = EXIT_BLOCK);               ! establish exit handler
;   103          0179  1 END;%;
;   104          0180  1
;   105          0181  1 !
;   106          0182  1 ! EQUATED SYMBOLS:
;   107          0183  1 !
;   108          0184  1 !       NONE
;   109          0185  1 !
;   110          0186  1 ! FIELDS:
;   111          0187  1 !
```

```
;  112          0188  1 !       NONE
;  113          0189  1 !
;  114          0190  1 ! PSECTS:
;  115          0191  1 !
;  116          0192  1
;  117          0193  1
;  118          0194  1 !
;  119          0195  1 ! OWN STORAGE:
;  120          0196  1 !
;  121          0197  1 OWN
;  122          0198  1     BINARY_NAM : REF $NAM_DECL,          ! name block for binary file
;  123          0199  1     EXIT_REASON : INITIAL (0),           ! addr to put in exit block
;  124          0200  1     EXIT_BLOCK : VECTOR [5] INITIAL (REP 5 OF (0));
;  125          0201  1                                          ! info block for exit handler
;  126          0202  1
;  127          0203  1 !
;  128          0204  1 ! EXTERNAL REFERENCES:
;  129          0205  1 !
;  130          0206  1 EXTERNAL ROUTINE
;  131          0207  1         LIB$TPARSE,                      ! parser
;  132          0208  1         LIB$GET_VM,                      ! allocate virtual memory
;  133          0209  1         LIB$FREE_VM;                     ! deallocate virtual memory
;  134          0210  1
;  135          0211  1 EXTERNAL
;  136          0212  1         SMG$$A_STMT_STATES,              ! TPARSE state table
;  137          0213  1         SMG$$A_STMT_KEYWDS,              ! TPARSE key words
;  138          0214  1         SMG$$DATA_OFFSET,                ! offset into TERMTABLE.EXE
;  139          0215  1         SMG$$CURRENT_DEF_BLOCK;          ! block number of current definition
;  140          0216  1
;  141          0217  1 EXTERNAL LITERAL
;  142          0218  1         SMG$_SYNERR;                     ! syntax error
```

```
;  144     0219  1 %SBTTL 'SMG$BUILD_TERM_TABLE'
;  145     0220  1 GLOBAL ROUTINE SMG$BUILD_TERM_TABLE : NOVALUE =
;  146     0221  1
;  147     0222  1 !++
;  148     0223  1 ! FUNCTIONAL DESCRIPTION:
;  149     0224  1 !
;  150     0225  1 ! This is the mainline logic to build the binary terminal capabilities file.
;  151     0226  1 ! It builds the binary file by:
;  152     0227  1 !       opening the ascii TERMTABLE.TXT for read access
;  153     0228  1 !       opening the binary TERMTABLE.EXE for write access (block I/O)
;  154     0229  1 !       parsing & moving the capabilities from the .txt file to the .exe
;  155     0230  1 !       exits after closing TERMTABLE.TXT and TERMTABLE.EXE
;  156     0231  1 !
;  157     0232  1 ! CALLING SEQUENCE:
;  158     0233  1 !
;  159     0234  1 !       SMG$BUILD_TERM_TABLE ()
;  160     0235  1 !
;  161     0236  1 ! FORMAL PARAMETERS:
;  162     0237  1 !
;  163     0238  1 !       NONE
;  164     0239  1 !
;  165     0240  1 ! IMPLICIT INPUTS:
;  166     0241  1 !
;  167     0242  1 !       NONE
;  168     0243  1 !
;  169     0244  1 ! IMPLICIT OUTPUTS:
;  170     0245  1 !
;  171     0246  1 !       NONE
;  172     0247  1 !
;  173     0248  1 ! COMPLETION STATUS:
;  174     0249  1 !
;  175     0250  1 !       Errors are signalled
;  176     0251  1 !
;  177     0252  1 ! SIDE EFFECTS:
;  178     0253  1 !
;  179     0254  1 !       NONE
;  180     0255  1 !
;  181     0256  1 !--
;  182     0257  1
;  183     0258  2     BEGIN
;  184     0259  2     LOCAL
;  185     0260  2         TPARSE_BLOCK,                   ! address of param block for LIB$TPARSE
;  186     0261  2         STATUS;                         ! status returned by called routines
;  187     0262  2
;  188     0263  2 !+
;  189     0264  2 ! Allocate virtual memory for control blocks.  Our local parameter
;  190     0265  2 ! block will follow the TPARSE parameter block.
;  191     0266  2 !-
;  192     0267  2
;  193     0268  3     IF NOT (STATUS = LIB$GET_VM (%REF (SMG$K_PARAM_BLOCK_SIZE),
;  194     0269  3                                  TPARSE_BLOCK))
;  195     0270  2     THEN
;  196     0271  2         SIGNAL_STOP (.STATUS);
;  197     0272  2
;  198     0273  2     CH$FILL (0, SMG$K_PARAM_BLOCK_SIZE, .TPARSE_BLOCK);
;  199     0274  2                                         ! init to zeroes
;  200     0275  2 !+
```

```
;  201     0276  2 ! OPEN_FILES will fill in the PARAM_BLOCK with the FAB and RAB for the
;  202     0277  2 ! ascii TERMTABLE.TXT, and the channel and virtual addresses of TERMTABLE.EXE
;  203     0278  2 ! which will be mapped as a section.  This data is needed later to access
;  204     0279  2 ! the files, and to close the files.
;  205     0280  2 !-
;  206     0281  2
;  207     0282  3     IF NOT (STATUS = OPEN_FILES (.TPARSE_BLOCK))
;  208     0283  2     THEN
;  209     0284  2         SIGNAL_STOP (.STATUS);
;  210     0285  2
;  211     0286  2 !+
;  212     0287  2 ! Do the real work.  Get records from the ascii file and move and/or
;  213     0288  2 ! convert them to the binary file.
;  214     0289  2 !-
;  215     0290  2
;  216     0291  2     IF NOT (STATUS = PARSE_TERM_DEFS (.TPARSE_BLOCK))
;  217     0292  2     THEN
;  218     0293  2         SIGNAL_STOP (.STATUS);
;  219     0294  2
;  220     0295  2 !+
;  221     0296  2 ! Done.  Close the ascii TERMTABLE and unmap the section from the binary
;  222     0297  2 ! TERMTABLE.
;  223     0298  2 !-
;  224     0299  2
;  225     0300  3     IF NOT (STATUS = CLOSE_FILES (.TPARSE_BLOCK))
;  226     0301  2     THEN
;  227     0302  2         SIGNAL_STOP (.STATUS);
;  228     0303  2
;  229     0304  2 !+
;  230     0305  2 ! Deallocate control blocks.
;  231     0306  2 !-
;  232     0307  2
;  233     0308  3     IF NOT (STATUS = LIB$FREE_VM (%REF (SMG$K_PARAM_BLOCK_SIZE),
;  234     0309  3                                         TPARSE_BLOCK))
;  235     0310  2     THEN
;  236     0311  2         SIGNAL_STOP (.STATUS);
;  237     0312  2
;  238     0313  1     END;                                ! End of routine SMG$BUILD_TERM_TABLE


                                                        .TITLE  SMG$BUILD_TERM_TABLE Build terminal table
                                                        .IDENT  \1-002\

                                                        .PSECT  _SMG$DATA,NOEXE,  PIC,.

                                            00000 BINARY_NAM:
                                                        .BLKB   4
                                   00000000 00004 EXIT_REASON:
                                                        .LONG   0                                       ;
                                   00000000# 00008 EXIT_BLOCK:
                                                        .LONG   0[5]                                    ;

                                                        .EXTRN  LIB$TPARSE, LIB$GET_VM
                                                        .EXTRN  LIB$FREE_VM, SMG$$A_STMT_STATES
                                                        .EXTRN  SMG$$A_STMT_KEYWDS
                                                        .EXTRN  SMG$$DATA_OFFSET
                                                        .EXTRN  SMG$$CURRENT_DEF_BLOCK
```

```
                                                        .EXTRN  SMG$_SYNERR

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

                                        00FC 00000      .ENTRY  SMG$BUILD_TERM_TABLE, Save R2,R3,R4,R5,R6,-      ; 0220
                                                                R7
                                    57 00000000G   00   9E 00002      MOVAB   LIB$STOP, R7
                                    5E             08   C2 00009      SUBL2   #8, SP
                                           04      AE   9F 0000C      PUSHAB  TPARSE_BLOCK                     ; 0268
                                        04  AE     60   8F 9A 0000F   MOVZBL  #96, 4(SP)
                                           04      AE   9F 00014      PUSHAB  4(SP)
                                00000000G  00           02 FB 00017  CALLS   #2, LIB$GET_VM
                                    56             50   D0 0001E      MOVL    R0, STATUS
                                    05             56   E8 00021      BLBS    STATUS, 1$
                                                   56   DD 00024      PUSHL   STATUS                           ; 0271
                                    67             01   FB 00026      CALLS   #1, LIB$STOP
0060    8F      00                  6E             00   2C 00029 1$:  MOVC5   #0, (SP), #0, #96, @TPARSE_BLOCK ; 0273
                                           04      BE      00030
                                           04      AE   DD 00032      PUSHL   TPARSE_BLOCK                     ; 0282
                                0000V      CF           01   FB 00035  CALLS   #1, OPEN_FILES
                                    56             50   D0 0003A      MOVL    R0, STATUS
                                    05             56   E8 0003D      BLBS    STATUS, 2$
                                                   56   DD 00040      PUSHL   STATUS                           ; 0284
                                    67             01   FB 00042      CALLS   #1, LIB$STOP
                                           04      AE   DD 00045 2$:  PUSHL   TPARSE_BLOCK                     ; 0291
                                0000V      CF           01   FB 00048  CALLS   #1, PARSE_TERM_DEFS
                                    56             50   D0 0004D      MOVL    R0, STATUS
                                    05             56   E8 00050      BLBS    STATUS, 3$
                                                   56   DD 00053      PUSHL   STATUS                           ; 0293
                                    67             01   FB 00055      CALLS   #1, LIB$STOP
                                           04      AE   DD 00058 3$:  PUSHL   TPARSE_BLOCK                     ; 0300
                                0000V      CF           01   FB 0005B  CALLS   #1, CLOSE_FILES
                                    56             50   D0 00060      MOVL    R0, STATUS
                                    05             56   E8 00063      BLBS    STATUS, 4$
                                                   56   DD 00066      PUSHL   STATUS                           ; 0302
                                    67             01   FB 00068      CALLS   #1, LIB$STOP
                                           04      AE   9F 0006B 4$:  PUSHAB  TPARSE_BLOCK                     ; 0308
                                        04  AE     60   8F 9A 0006E   MOVZBL  #96, 4(SP)
                                           04      AE   9F 00073      PUSHAB  4(SP)
                                00000000G  00           02 FB 00076  CALLS   #2, LIB$FREE_VM
                                    56             50   D0 0007D      MOVL    R0, STATUS
                                    05             56   E8 00080      BLBS    STATUS, 5$
                                                   56   DD 00083      PUSHL   STATUS                           ; 0311
                                    67             01   FB 00085      CALLS   #1, LIB$STOP
                                                   04   00088 5$:     RET                                      ; 0313

; Routine Size:  137 bytes,    Routine Base.  _SMG$CODE + 0000
```

```
;  240     0314  1 %SBTTL 'OPEN_FILES'
;  241     0315  1 ROUTINE OPEN_FILES ( PARAM_BLOCK : REF BLOCK [,BYTE]
;  242     0316  1     ) =
;  243     0317  1
;  244     0318  1 !++
;  245     0319  1 ! FUNCTIONAL DESCRIPTION:
;  246     0320  1 !
;  247     0321  1 !       This routine opens TERMTABLE.TXT for read access, and creates a file
;  248     0322  1 !       named TERMTABLE.EXE.  TERMTABLE.EXE is accessed via block I/O.
;  249     0323  1 !
;  250     0324  1 !
;  251     0325  1 ! CALLING SEQUENCE:
;  252     0326  1 !
;  253     0327  1 !       ret_status = OPEN_FILES (PARAM_BLOCK.mz.r)
;  254     0328  1 !
;  255     0329  1 !
;  256     0330  1 ! FORMAL PARAMETERS:
;  257     0331  1 !
;  258     0332  1 !       PARAM_BLOCK.mz.r        Parameter block in which to place information
;  259     0333  1 !                               needed to access files
;  260     0334  1 !
;  261     0335  1 !
;  262     0336  1 ! IMPLICIT INPUTS:
;  263     0337  1 !
;  264     0338  1 !       NONE
;  265     0339  1 !
;  266     0340  1 ! IMPLICIT OUTPUTS:
;  267     0341  1 !
;  268     0342  1 !       NONE
;  269     0343  1 !
;  270     0344  1 ! COMPLETION STATUS:
;  271     0345  1 !
;  272     0346  1 !       SS$_NORMAL
;  273     0347  1 !       Errors returned from $OPEN, $CREATE
;  274     0348  1 !
;  275     0349  1 ! SIDE EFFECTS:
;  276     0350  1 !
;  277     0351  1 !       NONE                                                            ; I
;  278     0352  1 !
;  279     0353  1 !--
;  280     0354  1
;  281     0355  2     BEGIN
;  282     0356  2
;  283     0357  2 !+
;  284     0358  2 ! Set up pointers to RMS structures.  The FAB and RAB for the
;  285     0359  2 ! ascii file will be needed later to do $GETs and $CLOSE.
;  286     0360  2 !-
;  287     0361  3     BEGIN
;  288     0362  3     LOCAL
;  289     0363  3         ASCII_FAB : REF $FAB_DECL,      ! ptr to FAB for ascii file
;  290     0364  3         ASCII_RAB : REF $RAB_DECL,      ! ptr to RAB for ascii file
;  291     0365  3         OPEN_STATUS,                    ! status retd from $OPEN
;  292     0366  3         STATUS;                         ! status retd from called routines
;  293     0367  3
;  294     0368  4     IF NOT (STATUS = LIB$GET_VM (%REF (FAB$C_BLN), PARAM_BLOCK [PARAM_A_TXT_FAB]))
;  295     0369  3     THEN
;  296     0370  3         RETURN (.STATUS);
```

```
;  297          0371  3
;  298          0372  4      IF NOT (STATUS = LIB$GET_VM (%REF (RAB$C_BLN), PARAM_BLOCK [PARAM_A_TXT_RAB]))
;  299          0373  3      THEN
;  300          0374  3          RETURN (.STATUS);
;  301          0375  3
;  302          0376  3      ASCII_FAB = .PARAM_BLOCK [PARAM_A_TXT_FAB];
;  303          0377  3      ASCII_RAB = .PARAM_BLOCK [PARAM_A_TXT_RAB];
;  304          0378  3
;  305          0379  3  !+
;  306          0380  3  ! First attempt to open the ascii TERMTABLE.TXT.  If this fails,
;  307          0381  3  ! processing ends.
;  308          0382  3  !-
;  309          0383  3
;  310        P 0384  3      $FAB_INIT (FAB = .ASCII_FAB, FNM = 'TERMTABLE', DNM = '.TXT',
;  311          0385  3                 FAC = GET, SHR = NIL, ORG = SEQ);
;  312          0386  3                                          ! set up FAB fields
;  313          0387  4      IF NOT (OPEN_STATUS = $OPEN (FAB = .ASCII_FAB))
;  314          0388  3      THEN
;  315          0389  3          RETURN (.OPEN_STATUS);
;  316          0390  3
;  317          0391  3  !+
;  318          0392  3  ! If we get here, we successfully opened TERMTABLE.TXT.  Connect a
;  319          0393  3  ! RAB to the FAB so we can access it.
;  320          0394  3  !-
;  321          0395  3
;  322          0396  3      $RAB_INIT (RAB = .ASCII_RAB, FAB = .ASCII_FAB); ! init RAB fields
;  323          0397  3
;  324          0398  4      IF NOT (STATUS = $CONNECT (RAB = .ASCII_RAB))
;  325          0399  3      THEN
;  326          0400  3          RETURN (.STATUS);                ! connect to ASCII_FAB
;  327          0401  3
;  328          0402  2      END;
;  329          0403  2
;  330          0404  2  !+
;  331          0405  2  ! Open the binary TERMTABLE.EXE.  We will be using block I/O to write to this
;  332          0406  2  ! file.
;  333          0407  2  !
;  334          0408  2  ! We always want to create a new file.
;  335          0409  2  !-
;  336          0410  3      BEGIN
;  337          0411  3      LOCAL
;  338          0412  3          STATUS,                          ! status retd from calls
;  339          0413  3          BINARY_FAB : REF $FAB_DECL,      ! ptr to FAB for binary file
;  340          0414  3          BINARY_RAB : REF $RAB_DECL,      ! ptr to RAB for binary file
;  341          0415  3          RESULT_NAME : VECTOR [NAM$C_MAXRSS,BYTE],! storage for name block
;  342          0416  3          EXPAND_NAME : VECTOR [NAM$C_MAXRSS,BYTE];! storage for name block
;  343          0417  3
;  344          0418  4      IF NOT (STATUS = LIB$GET_VM (%REF (FAB$C_BLN),
;  345          0419  4                                  PARAM_BLOCK [PARAM_A_BINARY_FAB]))
;  346          0420  3      THEN
;  347          0421  3          RETURN (.STATUS);
;  348          0422  3
;  349          0423  4      IF NOT (STATUS = LIB$GET_VM (%REF (RAB$C_BLN),
;  350          0424  4                                  PARAM_BLOCK [PARAM_A_BINARY_RAB]))
;  351          0425  3      THEN
;  352          0426  3          RETURN (.STATUS);
;  353          0427  3
```

```
;  354          0428  4     IF NOT (STATUS = LIB$GET_VM (%REF (NAM$C_BLN),
;  355          0429  4                                 BINARY_NAM))
;  356          0430  3     THEN
;  357          0431  3         RETURN (.STATUS);
;  358          0432  3
;  359          0433  3 !+
;  360          0434  3 ! Establish an exit handler which will delete TERMTABLE.EXE if the file is
;  361          0435  3 ! still open when it is called.  (Normally we should close the file ourselves
;  362          0436  3 ! without the DLT bit set.)  This prevents partial .exe's from being
;  363          0437  3 ! left in the user's directory.
;  364          0438  3 !-
;  365          0439  3
;  366          0440  3     $DECLARE_EXIT_HANDLER;
;  367          0441  3
;  368          0442  3     BINARY_FAB = .PARAM_BLOCK [PARAM_A_BINARY_FAB];
;  369          0443  3     BINARY_RAB = .PARAM_BLOCK [PARAM_A_BINARY_RAB];
;  370          0444  3
;  371          0445  3 !+
;  372          0446  3 ! Need a NAM block in order to use the DLT bit (delete on close).  DLT
;  373          0447  3 ! prevents an incomplete file from being left in the user's directory
;  374          0448  3 ! if the compiler is interupted.
;  375          0449  3 !-
;  376          0450  3
;  377        P 0451  3     $NAM_INIT (NAM = .BINARY_NAM, RSA = RESULT_NAME, RSS = NAM$C_MAXRSS,
;  378          0452  3                ESA = EXPAND_NAME, ESS = NAM$C_MAXRSS);
;  379          0453  3
;  380        P 0454  3     $FAB_INIT (FAB = .BINARY_FAB, ALQ = 20, FAC=<BIO,GET,PUT,DEL>,
;  381        P 0455  3                NAM = .BINARY_NAM, FOP = CBT, SHR=NIL, ORG = SEQ,
;  382          0456  3                FNM = 'TERMTABLE', DNM = '.EXE');
;  383          0457  3
;  384          0458  4     IF NOT (STATUS = $CREATE (FAB = .BINARY_FAB))
;  385          0459  3     THEN
;  386          0460  3         RETURN (.STATUS);
;  387          0461  3
;  388          0462  3 !+
;  389          0463  3 ! Connect a RAB to the binary FAB so we can access the records.
;  390          0464  3 !-
;  391          0465  3
;  392          0466  3     $RAB_INIT (RAB = .BINARY_RAB, FAB = .BINARY_FAB);
;  393          0467  3
;  394          0468  4     IF NOT (STATUS = $CONNECT (RAB = .BINARY_RAB))
;  395          0469  3     THEN
;  396          0470  3         RETURN (.STATUS);
;  397          0471  3
;  398          0472  2     END;
;  399          0473  2
;  400          0474  2     RETURN (SS$_NORMAL);
;  401          0475  1     END;                                    ! End of routine OPEN_FILES


                 45  4C  42  41  54  4D  52  45  54  00089 P.AAA:  .ASCII  \TERMTABLE\                ;
                                     54  58  54  2E  00092 P.AAB:  .ASCII  \.TXT\                    ;
                 45  4C  42  41  54  4D  52  45  54  00096 P.AAC:  .ASCII  \TERMTABLE\                ;
                                     45  58  45  2E  0009F P.AAD:  .ASCII  \.EXE\                    ;

                                                                   .EXTRN  SYS$OPEN, SYS$CONNECT
```

```
                                                .EXTRN  SYS$DCLEXH, SYS$CREATE

                                        0FFC 00000 OPEN_FILES:
                                                .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11    ; 0315
                                5B 00000000G    00  9E 00002         MOVAB   LIB$GET_VM, R11
                                5A 00000000'    EF  9E 00009         MOVAB   BINARY_NAM, R10
                                5E        FDFC  CE  9E 00010         MOVAB   -516(SP), SP
                                57          04  AC  D0 00015         MOVL    PARAM_BLOCK, R7         ; 0368
                                              24  A7  9F 00019         PUSHAB  36(R7)
                                    04    AE    50  8F  9A 0001C      MOVZBL  #80, 4(SP)
                                              04  AE  9F 00021         PUSHAB  4(SP)
                                6B                02  FB 00024         CALLS   #2, LIB$GET_VM
                                59                50  D0 00027         MOVL    R0, STATUS
                                70                59  E9 0002A         BLBC    STATUS, 2$
                                              28  A7  9F 0002D         PUSHAB  40(R7)                  ; 0372
                                    04    AE    44  8F  9A 00030      MOVZBL  #68, 4(SP)
                                              04  AE  9F 00035         PUSHAB  4(SP)
                                6B                02  FB 00038         CALLS   #2, LIB$GET_VM
                                59                50  D0 0003B         MOVL    R0, STATUS
                                70                59  E9 0003E         BLBC    STATUS, 3$
                                56          24  A7  D0 00041         MOVL    36(R7), ASCII_FAB       ; 0376
                                58          28  A7  D0 00045         MOVL    40(R7), ASCII_RAB       ; 0377
0050    8F              00      6E                00  2C 00049         MOVC5   #0, (SP), #0, #80, (ASCII_FAB)  ; 0385
                                66                    00050
                                66        5003  8F  B0 00051         MOVW    #20483, (ASCII_FAB)
                                    16    A6  2002  8F  B0 00056      MOVW    #8194, 22(ASCII_FAB)
                                              1D  A6  94 0005C         CLRB    29(ASCII_FAB)
                                    1F    A6        02  90 0005F      MOVB    #2, 31(ASCII_FAB)
                                    2C    A6    80  AF  9E 00063      MOVAB   P.AAA, 44(ASCII_FAB)
                                    30    A6    84  AF  9E 00068      MOVAB   P.AAB, 48(ASCII_FAB)
                                    34    A6  0409  8F  B0 0006D      MOVW    #1033, 52(ASCII_FAB)
                                              56  DD 00073         PUSHL   ASCII_FAB               ; 0387
                            00000000G       00      01  FB 00075     CALLS   #1, SYS$OPEN
                                            01      50  E8 0007C     BLBS    OPEN_STATUS, 1$
                                                    04 0007F         RET
0044    8F              00      6E                00  2C 00080 1$:    MOVC5   #0, (SP), #0, #68, (ASCII_RAB)  ; 0396
                                68                    00087
                                68        4401  8F  B0 00088         MOVW    #17409, (ASCII_RAB)
                                    3C    A8        56  D0 0008D     MOVL    ASCII_FAB, 60(ASCII_RAB)
                                              58  DD 00091         PUSHL   ASCII_RAB               ; 0398
                            00000000G       00      01  FB 00093     CALLS   #1, SYS$CONNECT
                                            59      50  D0 0009A     MOVL    R0, STATUS
                                            38      59  E9 0009D 2$: BLBC    STATUS, 4$
                                          2C  A7  9F 000A0         PUSHAB  44(R7)                  ; 0419
                                    04    AE    50  8F  9A 000A3      MOVZBL  #80, 4(SP)              ; 0418
                                              04  AE  9F 000A8         PUSHAB  4(SP)
                                6B                02  FB 000AB         CALLS   #2, LIB$GET_VM          ; 0419
                                59                50  D0 000AE         MOVL    R0, STATUS
                                24                59  E9 000B1 3$: BLBC    STATUS, 4$
                                          30  A7  9F 000B4         PUSHAB  48(R7)                  ; 0424
                                    04    AE    44  8F  9A 000B7      MOVZBL  #68, 4(SP)              ; 0423
                                              04  AE  9F 000BC         PUSHAB  4(SP)
                                6B                02  FB 000BF         CALLS   #2, LIB$GET_VM          ; 0424
                                59                50  D0 000C2         MOVL    R0, STATUS
                                10                59  E9 000C5         BLBC    STATUS, 4$
                                                  5A  DD 000C8         PUSHL   R10                     ; 0428
                                    04    AE    60  8F  9A 000CA      MOVZBL  #96, 4(SP)
```

```
                                                 04   AE  9F 000CF          PUSHAB  4(SP)
                                              6B      02  FB 000D2          CALLS   #2, LIB$GET_VM
                                              59      50  D0 000D5          MOVL    R0, STATUS
                                              03      59  E8 000D8 4$       BLBS    STATUS, 5$
                                                  00B3  31 000DB          BRW     6$
                                        0C    AA  0000V CF  9E 000DE 5$:    MOVAB   CLOSE_WITH_DLT, EXIT_BLOCK+4                    0431
                                        10    AA        02  D0 000E4        MOVL    #2, EXIT_BLOCK+8
                                        14    AA    04  AA  9E 000E8        MOVAB   EXIT_REASON, EXIT_BLOCK+12
                                        18    AA        57  D0 000ED        MOVL    R7, EXIT_BLOCK+16
                                                08  AA  9F 000F1          PUSHAB  EXIT_BLOCK
                                  00000000G     00        01  FB 000F4        CALLS   #1, SYS$DCLEXH
                                              56    2C  A7  D0 000FB        MOVL    44(R7), BINARY_FAB                              0442
                                              58    30  A7  D0 000FF        MOVL    48(R7), BINARY_RAB                              0443
                                              57        6A  D0 00103        MOVL    BINARY_NAM, R7                                  0452
    0060  8F          00                    6E            00  2C 00106        MOVC5   #0, (SP), #0, #96, (R7)
                                                        67    0010D
                                              67  6002  8F  B0 0010E        MOVW    #24578, (R7)
                                        02    A7        01  8E 00113        MNEGB   #1, 2(R7)
                                        04    A7  FF00  CD  9E 00117        MOVAB   RESULT_NAME, 4(R7)
                                        0A    A7        01  8E 0011D        MNEGB   #1, 10(R7)
                                        0C    A7    04  AE  9E 00121        MOVAB   EXPAND_NAME, 12(R7)
    0050  8F          00                    6E            00  2C 00126        MOVC5   #0, (SP), #0, #80, (BINARY_FAB)                 0456
                                                        66    0012D
                                              66  5003  8F  B0 0012E        MOVW    #20483, (BINARY_FAB)
                                        04    A6 00200000 8F  D0 00133        MOVL    #2097152, 4(BINARY_FAB)
                                        10    A6        14  D0 0013B        MOVL    #20, 16(BINARY_FAB)
                                        16    A6  2027  8F  B0 0013F        MOVW    #8231, 22(BINARY_FAB)
                                                  1D  A6  94 00145          CLRB    29(BINARY_FAB)
                                        1F    A6        02  90 00148        MOVB    #2, 31(BINARY_FAB)
                                        28    A6        57  D0 0014C        MOVL    R7, 40(BINARY_FAB)
                                        2C    A6  FE9F  CF  9E 00150        MOVAB   P.AAC, 44(BINARY_FAB)
                                        30    A6  FEA2  CF  9E 00156        MOVAB   P.AAD, 48(BINARY_FAB)
                                        34    A6  0409  8F  B0 0015C        MOVW    #1033, 52(BINARY_FAB)
                                                        56  DD 00162        PUSHL   BINARY_FAB                                      0458
                                  00000000G     00        01  FB 00164        CALLS   #1, SYS$CREATE
                                              59        50  D0 0016B        MOVL    R0, STATUS
                                              20        59  E9 0016E        BLBC    STATUS, 6$
    0044  8F          00                    6E            00  2C 00171        MOVC5   #0, (SP), #0, #68, (BINARY_RAB)                 0466
                                                        68    00178
                                              68  4401  8F  B0 00179        MOVW    #17409, (BINARY_RAB)
                                        3C    A8        56  D0 0017E        MOVL    BINARY_FAB, 60(BINARY_RAB)
                                                        58  DD 00182        PUSHL   BINARY_RAB                                      0468
                                  00000000G     00        01  FB 00184        CALLS   #1, SYS$CONNECT
                                              59        50  D0 0018B        MOVL    R0, STATUS
                                              04        59  E8 0018E        BLBS    STATUS, 7$
                                              50        59  D0 00191 6$:    MOVL    STATUS, R0                                      0470
                                                            04 00194        RET
                                              50        01  D0 00195 7$:    MOVL    #1, R0                                          0474
                                                            04 00198        RET                                                     0475

; Routine Size:  409 bytes,    Routine Base:  _SMG$CODE + 00A3
```

```
;  403          0476  1 %SBTTL 'PARSE_TERM_DEFS'
;  404          0477  1 ROUTINE PARSE_TERM_DEFS ( TPARSE_BLOCK : REF BLOCK [,BYTE]
;  405          0478  1     ) =
;  406          0479  1
;  407          0480  1 !++
;  408          0481  1 ! FUNCTIONAL DESCRIPTION:
;  409          0482  1 !
;  410          0483  1 !       This routine moves data from TERMTABLE.TXT to TERMTABLE.EXE.
;  411          0484  1 !       The data is compacted and stored in a manner that allows easy
;  412          0485  1 !       retrieval.
;  413          0486  1 !
;  414          0487  1 !       PARSE_TERM_DEFS initializes some uniq • fields in TERMTABLE.EXE
;  415          0488  1 !       (for instance the ident), and inserts t e terminal definition
;  416          0489  1 !       index at the end of the file.  Terminal definitions are actually
;  417          0490  1 !       parsed and moved by LIB$TPARSE.
;  418          0491  1 !
;  419          0492  1 !
;  420          0493  1 ! CALLING SEQUENCE:
;  421          0494  1 !
;  422          0495  1 !       ret_status = PARSE_TERM_DEFS (PARAM_BLOCK.rz.r)
;  423          0496  1 !
;  424          0497  1 ! FORMAL PARAMETERS:
;  425          0498  1 !
;  426          0499  1 !       PARAM_BLOCK.rz.r        Address of control block with info needed
;  427          0500  1 !                               to access files
;  428          0501  1 !
;  429          0502  1 ! IMPLICIT INPUTS:
;  430          0503  1 !
;  431          0504  1 !       NONE
;  432          0505  1 !
;  433          0506  1 ! IMPLICIT OUTPUTS:
;  434          0507  1 !
;  435          0508  1 !       NONE
;  436          0509  1 !
;  437          0510  1 ! COMPLETION STATUS:
;  438          0511  1 !
;  439          0512  1 !
;  440          0513  1 !
;  441          0514  1 ! SIDE EFFECTS:
;  442          0515  1 !
;  443          0516  1 !       NONE
;  444          0517  1 !
;  445          0518  1 !--
;  446          0519  1
;  447          0520  2     BEGIN
;  448          0521  2     LOCAL
;  449          0522  2         TXT_RAB : REF $RAB_DECL,        ! ptr to RAB for TERMTABLE.TXT
;  450          0523  2         BUFFER : VECTOR [255,BYTE],     ! buffer to hold .TXT records
;  451          0524  2         PARALLEL_BUFFER : VECTOR [255,BYTE],
;  452          0525  2                                         ! another buffer to hold .TXT records
;  453          0526  2         VM_STATUS,                      ! status from LIB$GET/FREE_VM
;  454          0527  2         TERM_INDEX;                     ! addr for local terminal index
;  455          0528  2
;  456          0529  2 !+
;  457          0530  2 ! Allocate the buffers used to construct terminal definitions.  We write
;  458          0531  2 ! to these buffers until we start a new terminal definition or until they
;  459          0532  2 ! are full, and then we do a block I/O write to flush them.
```

```
;  460          0533  2 !-
;  461          0534  2
;  462          0535  3     IF NOT (VM_STATUS = LIB$GET_VM (%REF (SMG$K_HEADER_SIZE),
;  463          0536  3                                     TPARSE_BLOCK [PARAM_A_HEADER]))
;  464          0537  2     THEN
;  465          0538  2         RETURN (.VM_STATUS);
;  466          0539  2
;  467          0540  2     CH$FILL (0, SMG$K_HEADER_SIZE, .TPARSE_BLOCK [PARAM_A_HEADER]);
;  468          0541  2
;  469          0542  3     IF NOT (VM_STATUS = LIB$GET_VM (%REF (SMG$K_TERM_DEF_SIZE),
;  470          0543  3                                     TPARSE_BLOCK [PARAM_A_CAP_PTRS]))
;  471          0544  2     THEN
;  472          0545  2         RETURN (.VM_STATUS);
;  473          0546  2
;  474          0547  2     CH$FILL (0, SMG$K_TERM_DEF_SIZE, .TPARSE_BLOCK [PARAM_A_CAP_PTRS]);
;  475          0548  2
;  476          0549  2     TPARSE_BLOCK [PARAM_A_CAP_DATA] = .TPARSE_BLOCK [PARAM_A_CAP_PTRS] +
;  477          0550  2                                       SMG$K_CAP_PTRS_SIZE;
;  478          0551  2
;  479          0552  2     TPARSE_BLOCK [PARAM_L_CUR_DATA_BYTE] = .TPARSE_BLOCK [PARAM_A_CAP_DATA];
;  480          0553  2                                         ! start with 1st byte of data buffer
;  481          0554  2
;  482          0555  2 !+
;  483          0556  2 ! Point to the first terminal definition.  A terminal definition consists
;  484          0557  2 ! of capability pointers followed by capability data.  Each terminal
;  485          0558  2 ! definition will begin on a block boundary.
;  486          0559  2 !
;  487          0560  2 ! See SMGTABDEF.REQ for more info on the structure of TERMTABLE.EXE.
;  488          0561  2 !-
;  489          0562  2
;  490          0563  3     BEGIN                                   ! new block for BIND
;  491          0564  3
;  492          0565  3     BIND
;  493          0566  3         TERM_TAB = .TPARSE_BLOCK [PARAM_A_HEADER] : BLOCK [,BYTE];
;  494          0567  3
;  495          0568  3 !+
;  496          0569  3 ! Set up some fields at the beginning of TERMTABLE.EXE.
;  497          0570  3 !-
;  498          0571  3
;  499          0572  3     TERM_TAB [TTB_W_IDENT] = 1;             ! 1st version of terminal table
;  500          0573  3
;  501          0574  3 !+
;  502          0575  3 ! We call LIB$TPARSE to parse and move all the terminal definitions into
;  503          0576  3 ! TERMTABLE.EXE.
;  504          0577  3 !
;  505          0578  3 ! Since the TPARSE action routines have access to the RAB, we can set up
;  506          0579  3 ! a buffer to hold TERMTABLE.TXT records here.
;  507          0580  3 !
;  508          0581  3 ! We set up 2 buffers, one for the RAB and a second 'parallel' buffer.
;  509          0582  3 ! The text in the RAB buffer will be upcased for parsing purposes, and
;  510          0583  3 ! the text in the parallel buffer will remain the way the user specified
;  511          0584  3 ! in TERMTABLE.TXT for copying into TERMTABLE.EXE.  (Note that case is
;  512          0585  3 ! significant in escape and control sequences, but we parse only uppercase
;  513          0586  3 ! to avoid having upper and lower case keywords.)
;  514          0587  3 !-
;  515          0588  3
;  516          0589  3     TXT_RAB = .TPARSE_BLOCK [PARAM_A_TXT_RAB];
```

```
;  517          0590  3     TXT_RAB [RAB$W_USZ] = 255;
;  518          0591  3     TXT_RAB [RAB$L_UBF] = BUFFER [0];
;  519          0592  3
;  520          0593  3     TPARSE_BLOCK [PARAM_L_ORIG_TXT] = PARALLEL_BUFFER [0];
;  521          0594  3
;  522          0595  3 !+
;  523          0596  3 ! Initialize fields needed by LIB$TPARSE.
;  524          0597  3 !-
;  525          0598  3
;  526          0599  3     TPARSE_BLOCK [TPA$L_COUNT] = TPA$K_COUNTO;
;  527          0600  3     TPARSE_BLOCK [TPA$V_BLANKS] = 0;
;  528          0601  3                                   ! start by skipping over blanks
;  529          0602  3
;  530          0603  3 !+
;  531          0604  3 ! Allocate temporary virtual memory to hold the terminal index.  This
;  532          0605  3 ! index belongs at the end of the terminal table but we don't know how
;  533          0606  3 ! much space the terminal definitions will take.  So we will move it
;  534          0607  3 ! there after all the terminal definitions have been processed.
;  535          0608  3 !-
;  536          0609  3
;  537          0610  4     IF NOT (VM_STATUS = LIB$GET_VM (%REF (SMG$K_TERM_INDEX_SIZE), TERM_INDEX))
;  538          0611  3     THEN
;  539          0612  3         RETURN (.VM_STATUS);
;  540          0613  3
;  541          0614  3     CH$FILL (0, SMG$K_TERM_INDEX_SIZE, .TERM_INDEX);
;  542          0615  3
;  543          0616  3     TPARSE_BLOCK [PARAM_A_TERM_INDEX] = .TERM_INDEX;
;  544          0617  3
;  545          0618  3 !+
;  546          0619  3 ! Ready to do the real work.
;  547          0620  3 !-
;  548          0621  3
;  549          0622  3     IF NOT LIB$TPARSE (.TPARSE_BLOCK, SMG$$A_STMT_STATES, SMG$$A_STMT_KEYWDS)
;  550          0623  3     THEN
;  551          0624  3         SIGNAL_STOP (SMG$_SYNERR);              ! syntax error
;  552          0625  3
;  553          0626  3 !+
;  554          0627  3 ! Now that all terminal definitions are in place, append the terminal index.
;  555          0628  3 !-
;  556          0629  3
;  557          0630  4     BEGIN
;  558          0631  4     LOCAL
;  559          0632  4         BINARY_RAB : REF $RAB_DECL;
;  560          0633  4     MAP
;  561          0634  4         TERM_INDEX : REF VECTOR [,BYTE];
;  562          0635  4
;  563          0636  4     BINARY_RAB = .TPARSE_BLOCK [PARAM_A_BINARY_RAB];
;  564          0637  4
;  565          0638  4     TERM_INDEX [.TPARSE_BLOCK [PARAM_L_TERM_INDEX_SIZE]] = 0;
;  566          0639  4                                       ! mark end of index with a zero
;  567          0640  4
;  568          0641  4     TERM_TAB [TTB_L_INDEX_OFFSET] = (.SMG$$CURRENT_DEF_BLOCK - 1) * 512;
;  569          0642  4                                       ! offset in bytes
;  570          0643  4
;  571          0644  4     BINARY_RAB [RAB$W_RSZ] = .TPARSE_BLOCK [PARAM_L_TERM_INDEX_SIZE] + 1;
;  572          0645  4                                     ! add one for zero
;  573          0646  4     BINARY_RAB [RAB$L_RBF] = .TERM_INDEX;
```

```
;  574          0647  4
;  575          0648  4     BINARY_RAB [RAB$L_BKT] = .SMG$$CURRENT_DEF_BLOCK;
;  576          0649  4
;  577          0650  4     $WRITE (RAB = .BINARY_RAB);
;  578          0651  4
;  579          0652  4 !+
;  580          0653  4 ! Write out the header block.
;  581          0654  4 !-
;  582          0655  4
;  583          0656  4     BINARY_RAB [RAB$W_RSZ] = SMG$K_HEADER_SIZE;
;  584          0657  4
;  585          0658  4     BINARY_RAB [RAB$L_RBF] = .TPARSE_BLOCK [PARAM_A_HEADER];
;  586          0659  4
;  587          0660  4     BINARY_RAB [RAB$L_BKT] = 1;          ! 1st block of file
;  588          0661  4
;  589          0662  4     $WRITE (RAB = .BINARY_RAB);
;  590          0663  4
;  591          0664  3     END;
;  592          0665  3
;  593          0666  3 !+
;  594          0667  3 ! Deallocate temporary storage.
;  595          0668  3 !-
;  596          0669  3
;  597          0670  4     IF NOT (VM_STATUS = LIB$FREE_VM (%REF (SMG$K_HEADER_SIZE),
;  598          0671  4                                      TPARSE_BLOCK [PARAM_A_HEADER]))
;  599          0672  3     THEN
;  600          0673  3         RETURN (.VM_STATUS);
;  601          0674  3
;  602          0675  4     IF NOT (VM_STATUS = LIB$FREE_VM (%REF (SMG$K_TERM_DEF_SIZE),
;  603          0676  4                                      TPARSE_BLOCK [PARAM_A_CAP_PTRS]))
;  604          0677  3     THEN
;  605          0678  3         RETURN (.VM_STATUS);
;  606          0679  3
;  607          0680  4     IF NOT (VM_STATUS = LIB$FREE_VM (%REF (SMG$K_TERM_INDEX_SIZE), TERM_INDEX))
;  608          0681  3     THEN
;  609          0682  3         RETURN (.VM_STATUS);
;  610          0683  3
;  611          0684  3     RETURN (SS$_NORMAL);
;  612          0685  3
;  613          0686  2     END;                                    ! end of BIND
;  614          0687  1     END;                                    ! End of routine PARSE_TERM_DEFS


                                                    .EXTRN  SYS$WRITE

                                        0FFC 00000 PARSE_TERM_DEFS:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0477
                                5B 00000000G   00   9E 00002    MOVAB   LIB$GET_VM, R11
                                5A 00000000G   00   9E 00009    MOVAB   LIB$FREE_VM, R10
                                5E      FDF8   CE   9E 00010    MOVAB   -520(SP), SP
                                56        04   AC   D0 00015    MOVL    TPARSE_BLOCK, R6                    ; 0536
                                58        3C   A6   9E 00019    MOVAB   60(R6), R8
                                               58   DD 0001D    PUSHL   R8
                             04 AE        0200   8F   3C 0001F    MOVZWL  #512, 4(SP)                     ; 0535
                                           0-   AE   9F 00025    PUSHAB  4(SP)
                                6B              02   FB 00028    CALLS   #2, LIB$GET_VM                  ; 0536
```

```
                              59            50  D0 0002B           MOVL    R0, VM_STATUS                              ;
                              69            59  E9 0002E           BLBC    VM_STATUS, 1$                              ;
  0200  8F          00        6E            00  2C 00031           MOVC5   #0, (SP), #0, #512, @0(R8)                 ; 0540
                                     00     B8     00038
                                     34     A6  9F 0003A           PUSHAB  52(R6)                                     ; 0543
                        04    AE   1A00     8F  3C 0003D           MOVZWL  #6656, 4(SP)                               ; 0542
                                     04     AE  9F 00043           PUSHAB  4(SP)                                      ;
                              6B            02  FB 00046           CALLS   #2, LIB$GET_VM                             ; 0543
                              59            50  D0 00049           MOVL    R0, VM_STATUS                              ;
                              4B            59  E9 0004C           BLBC    VM_STATUS, 1$                              ;
  1A00  8F          00        6E            00  2C 0004F           MOVC5   #0, (SP), #0, #6656, @52(R6)               ; 0547
                                     34     B6     00056
              38    A6  34    A6 00000600   8F  C1 00058           ADDL3   #1536, 52(R6), 56(R6)                      ; 0549
                        4C    A6     38     A6  D0 00062           MOVL    56(R6), 76(R6)                             ; 0552
                              57            68  D0 00067           MOVL    (R8), R7                                   ; 0566
                              67            01  B0 0006A           MOVW    #1, (R7)                                   ; 0572
                              50     28     A6  D0 0006D           MOVL    40(R6), TXT_RAB                            ; 0589
                        20    A0     FF     8F  9B 00071           MOVZBW  #255, 32(TXT_RAB)                          ; 0590
                        24    A0   FF00     CD  9E 00076           MOVAB   BUFFER, 36(TXT_RAB)                        ; 0591
                        5C    A6     08     AE  9E 0007C           MOVAB   PARALLEL_BUFFER, 92(R6)                    ; 0593
                              66            08  D0 00081           MOVL    #8, (R6)                                   ; 0599
                        04    A6            01  8A 00084           BICB2   #1, 4(R6)                                  ; 0600
                                     04     AE  9F 00088           PUSHAB  TERM_INDEX                                 ; 0610
                        04    AE   1388     8F  3C 0008B           MOVZWL  #5000, 4(SP)                               ;
                                     04     AE  9F 00091           PUSHAB  4(SP)                                      ;
                              6B            02  FB 00094           CALLS   #2, LIB$GET_VM                             ;
                              59            50  D0 00097           MOVL    R0, VM_STATUS                              ;
                              03            59  E8 0009A 1$:       BLBS    VM_STATUS, 2$                              ;
                                   00BC     31     0009D           BRW     4$                                         ;
  1388  8F          00        6E            00  2C 000A0 2$:       MOVC5   #0, (SP), #0, #5000, @TERM_INDEX           ; 0614
                                     04     BE     000A7
                        44    A6     04     AE  D0 000A9           MOVL    TERM_INDEX, 68(R6)                         ; 0616
                                00000000G   00  9F 000AE           PUSHAB  SMG$$A_STMT_KEYWDS                         ; 0622
                                00000000G   00  9F 000B4           PUSHAB  SMG$$A_STMT_STATES                         ;
                                            56  DD 000BA           PUSHL   R6                                         ;
                 00000000G    00            03  FB 000BC           CALLS   #3, LIB$TPARSE                             ;
                              0D            50  E8 000C3           BLBS    R0, 3$                                     ;
                                00000000G   8F  DD 000C6           PUSHL   #SMG$_SYNERR                               ; 0624
                 00000000G    00            01  FB 000CC           CALLS   #1, LIB$STOP                               ;
                              52     30     A6  D0 000D3 3$:       MOVL    48(R6), BINARY_RAB                         ; 0636
                  50    04    AE     40     A6  C1 000D7           ADDL3   64(R6), TERM_INDEX, R0                     ; 0638
                              60            94     000DD           CLRB    (R0)                                       ;
                              51 00000000G  00  D0 000DF           MOVL    SMG$$CURRENT_DEF_BLOCK, R1                 ; 0641
                              50     FF     A1  9E 000E6           MOVAB   -1(R1), R0                                 ;
              04    A7        50            09  78 000EA           ASHL    #9, R0, 4(R7)                              ;
              22    A2  40    A6            01  A1 000EF           ADDW3   #1, 64(R6), 34(BINARY_RAB)                 ; 0644
                        28    A2     04     AE  D0 000F5           MOVL    TERM_INDEX, 40(BINARY_RAB)                 ; 0646
                        38    A2            51  D0 000FA           MOVL    R1, 56(BINARY_RAB)                         ; 0648
                                            52  DD 000FE           PUSHL   BINARY_RAB                                 ; 0650
                 00000000G    00            01  FB 00100           CALLS   #1, SYS$WRITE                              ;
                        22    A2   0200     8F  B0 00107           MOVW    #512, 34(BINARY_RAB)                       ; 0656
                        28    A2            68  D0 0010D           MOVL    (R8), 40(BINARY_RAB)                       ; 0658
                        38    A2            01  D0 00111           MOVL    #1, 56(BINARY_RAB)                         ; 0660
                                            52  DD 00115           PUSHL   BINARY_RAB                                 ; 0662
                 00000000G    00            01  FB 00117           CALLS   #1, SYS$WRITE                              ;
                                            58  DD 0011E           PUSHL   R8                                         ; 0671
                        04    AE   0200     8F  3C 00120           MOVZWL  #512, 4(SP)                                ; 0670
```

```
                                    04   AE   9F 00126          PUSHAB  4(SP)
                                6A       02   FB 00129          CALLS   #2, LIB$FREE_VM                 ; 0671
                                59       50   D0 0012C          MOVL    R0, VM_STATUS
                                2A       59   E9 0012F          BLBC    VM_STATUS, 4$
                                     34  A6   9F 00132          PUSHAB  52(R6)                          ; 0676
                            04   AE  1A00  8F   3C 00135          MOVZWL  #6656, 4(SP)                    ; 0675
                                     04  AE   9F 0013B          PUSHAB  4(SP)
                                6A       02   FB 0013E          CALLS   #2, LIB$FREE_VM                 ; 0676
                                59       50   D0 00141          MOVL    R0, VM_STATUS
                                15       59   E9 00144          BLBC    VM_STATUS, 4$
                                     04  AE   9F 00147          PUSHAB  TERM_INDEX                      ; 0680
                            04   AE  1388  8F   3C 0014A          MOVZWL  #5000, 4(SP)
                                     04  AE   9F 00150          PUSHAB  4(SP)
                                6A       02   FB 00153          CALLS   #2, LIB$FREE_VM
                                59       50   D0 00156          MOVL    R0, VM_STATUS
                                04       59   E8 00159          BLBS    VM_STATUS, 5$
                                50       59   D0 0015C 4$:      MOVL    VM_STATUS, R0                   ; 0682
                                              04 0015F          RET
                                50       01   D0 00160 5$:      MOVL    #1, R0                          ; 0684
                                              04 00163          RET                                     ; 0687

; Routine Size:  356 bytes,    Routine Base:  _SMG$CODE + 023C
```

```
;  616          0688  1 %SBTTL 'CLOSE_FILES'
;  617          0689  1 ROUTINE CLOSE_FILES (PARAM_BLOCK
;  618          0690  1     ) =
;  619          0691  1
;  620          0692  1 !++
;  621          0693  1 ! FUNCTIONAL DESCRIPTION:
;  622          0694  1 !
;  623          0695  1 !       This routine cleans up after TERMTABLE.TXT has been processed.  It
;  624          0696  1 !       closes TERMTABLE.TXT and TERMTABLE.EXE.
;  625          0697  1 !
;  626          0698  1 ! CALLING SEQUENCE:
;  627          0699  1 !
;  628          0700  1 !       ret_status = CLOSE_FILES (PARAM_BLOCK.rz.r)
;  629          0701  1 !
;  630          0702  1 ! FORMAL PARAMETERS:
;  631          0703  1 !
;  632          0704  1 !       PARAM_BLOCK.rz.r        Address of control block with info needed
;  633          0705  1 !                               to access files
;  634          0706  1 !
;  635          0707  1 ! IMPLICIT INPUTS:
;  636          0708  1 !
;  637          0709  1 !       NONE
;  638          0710  1 !
;  639          0711  1 ! IMPLICIT OUTPUTS:
;  640          0712  1 !
;  641          0713  1 !       NONE
;  642          0714  1 !
;  643          0715  1 ! COMPLETION STATUS:
;  644          0716  1 !
;  645          0717  1 !       SS$_NORMAL
;  646          0718  1 !       Errors from $CLOSE
;  647          0719  1 !
;  648          0720  1 ! SIDE EFFECTS:
;  649          0721  1 !
;  650          0722  1 !       NONE
;  651          0723  1 !
;  652          0724  1 !--
;  653          0725  1
;  654          0726  2     BEGIN
;  655          0727  2
;  656          0728  2     LOCAL
;  657          0729  2         STATUS;
;  658          0730  2
;  659          0731  2     MAP
;  660          0732  2         PARAM_BLOCK : REF BLOCK [,BYTE];
;  661          0733  2
;  662          0734  2 !+
;  663          0735  2 ! Close the ascii TERMTABLE.
;  664          0736  2 !-
;  665          0737  2
;  666          0738  3     IF NOT (STATUS = $CLOSE (FAB = .PARAM_BLOCK [PARAM_A_TXT_FAB]))
;  667          0739  2     THEN
;  668          0740  2         RETURN (.STATUS);
;  669          0741  2
;  670          0742  2 !+
;  671          0743  2 ! Close the binary TERMTABLE.
;  672          0744  2 !-
```

```
;  673          0745  2
;  674          0746  3      IF NOT (STATUS = $CLOSE (FAB = .PARAM_BLOCK [PARAM_A_BINARY_FAB]))
;  675          0747  2      THEN
;  676          0748  2          RETURN (.STATUS);
;  677          0749  2
;  678          0750  2      !+
;  679          0751  2      ! Now that we have closed the file properly, there's no need to worry about
;  680          0752  2      ! an incomplete .exe (due to a ctrl Y or ctrl C).
;  681          0753  2      !-
;  682          0754  2
;  683          0755  2      $CANEXH (DESBLK = EXIT_BLOCK);
;  684          0756  2
;  685          0757  2      !+
;  686          0758  2      ! Deallocate virtual memory used for FABs and RABs.
;  687          0759  2      !-
;  688          0760  2
;  689          0761  3      IF NOT (STATUS = LIB$FREE_VM (%REF (FAB$C_BLN), PARAM_BLOCK [PARAM_A_TXT_FAB]))
;  690          0762  2      THEN
;  691          0763  2          RETURN (.STATUS);
;  692          0764  2
;  693          0765  3      IF NOT (STATUS = LIB$FREE_VM (%REF (RAB$C_BLN), PARAM_BLOCK [PARAM_A_TXT_RAB]))
;  694          0766  2      THEN
;  695          0767  2          RETURN (.STATUS);
;  696          0768  2
;  697          0769  3      IF NOT (STATUS = LIB$FREE_VM (%REF (FAB$C_BLN), PARAM_BLOCK [PARAM_A_BINARY_FAB]))
;  698          0770  2      THEN
;  699          0771  2          RETURN (.STATUS);
;  700          0772  2
;  701          0773  3      IF NOT (STATUS = LIB$FREE_VM (%REF (RAB$C_BLN), PARAM_BLOCK [PARAM_A_BINARY_RAB]))
;  702          0774  2      THEN
;  703          0775  2          RETURN (.STATUS);
;  704          0776  2
;  705          0777  3      IF NOT (STATUS = LIB$FREE_VM (%REF (NAM$C_BLN), BINARY_NAM))
;  706          0778  2      THEN
;  707          0779  2          RETURN (.STATUS);
;  708          0780  2
;  709          0781  2      RETURN (SS$_NORMAL);
;  710          0782  1      END;                                    ! End of routine CLOSE_FILES


                                                .EXTRN  SYS$CLOSE, SYS$CANEXH

                                    003C 00000 CLOSE_FILES:
                                                .WORD   Save R2,R3,R4,R5                ; 0689
                    55 00000000G    00   9E 00002          MOVAB   SYS$CLOSE, R5
                    54 00000000G    00   9E 00009          MOVAB   LIB$FREE_VM, R4
                    5E              04   C2 00010          SUBL2   #4, SP
                    52        04    AC   D0 00013          MOVL    PARAM_BLOCK, R2         ; 0738
                              24    A2   DD 00017          PUSHL   36(R2)
                    65              01   FB 0001A          CALLS   #1, SYS$CLOSE
                    53              50   D0 0001D          MOVL    R0, STATUS
                    66              53   E9 00020          BLBC    STATUS, 1$
                              2C    A2   DD 00023          PUSHL   44(R2)                  ; 0746
                    65              01   FB 00026          CALLS   #1, SYS$CLOSE
                    53              50   D0 00029          MOVL    R0, STATUS
                    74              53   E9 0002C          BLBC    STATUS, 2$
```

```
                    00000000'  EF  9F 0002F          PUSHAB  EXIT_BLOCK                      ; 0755
          00000000G 00         01  FB 00035          CALLS   #1, SYS$CANEXH
                     24        A2  9F 0003C          PUSHAB  36(R2)                          ; 0761
                 04  AE        50  8F 9A 0003F      MOVZBL  #80, 4(SP)
                     04        AE  9F 00044          PUSHAB  4(SP)
                     64        02  FB 00047          CALLS   #2, LIB$FREE_VM
                     53        50  D0 0004A          MOVL    R0, STATUS
                     53        53  E9 0004D          BLBC    STATUS, 2$
                     28        A2  9F 00050          PUSHAB  40(R2)                          ; 0765
                 04  AE        44  8F 9A 00053      MOVZBL  #68, 4(SP)
                     04        AE  9F 00058          PUSHAB  4(SP)
                     64        02  FB 0005B          CALLS   #2, LIB$FREE_VM
                     53        50  D0 0005E          MOVL    R0, STATUS
                     3F        53  E9 00061          BLBC    STATUS, 2$
                     2C        A2  9F 00064          PUSHAB  44(R2)                          ; 0769
                 04  AE        50  8F 9A 00067      MOVZBL  #80, 4(SP)
                     04        AE  9F 0006C          PUSHAB  4(SP)
                     64        02  FB 0006F          CALLS   #2, LIB$FREE_VM
                     53        50  D0 00072          MOVL    R0, STATUS
                     2B        53  E9 00075          BLBC    STATUS, 2$
                     30        A2  9F 00078          PUSHAB  48(R2)                          ; 0773
                 04  AE        44  8F 9A 0007B      MOVZBL  #68, 4(SP)
                     04        AE  9F 00080          PUSHAB  4(SP)
                     64        02  FB 00083          CALLS   #2, LIB$FREE_VM
                     53        50  D0 00086          MOVL    R0, STATUS
                     17        53  E9 00089  1$:     BLBC    STATUS, 2$
                    00000000'  EF  9F 0008C          PUSHAB  BINARY_NAM                      ; 0777
                 04  AE        60  8F 9A 00092      MOVZBL  #96, 4(SP)
                     04        AE  9F 00097          PUSHAB  4(SP)
                     64        02  FB 0009A          CALLS   #2, LIB$FREE_VM
                     53        50  D0 0009D          MOVL    R0, STATUS
                     04        53  E8 000A0          BLBS    STATUS, 3$
                     50        53  D0 000A3  2$:     MOVL    STATUS, R0                      ; 0779
                               04 000A6             RET
                     50        01  D0 000A7  3$:     MOVL    #1, R0                          ; 0781
                               04 000AA             RET                                     ; 0782

; Routine Size:  171 bytes,    Routine Base:  _SMG$CODE + 03A0
```

```
;  712          0783  1 %SBTTL 'CLOSE_WITH_DLT'
;  713          0784  1 ROUTINE CLOSE_WITH_DLT (EXIT_REASON,
;  714          0785  1                         PARAM_BLOCK : REF BLOCK [,BYTE]
;  715          0786  1     ) =
;  716          0787  1
;  717          0788  1 !++
;  718          0789  1 ! FUNCTIONAL DESCRIPTION:
;  719          0790  1 !
;  720          0791  1 !       This routine is an exit handler.  It is called only if
;  721          0792  1 !       we are interupted (it is de-established as an exit handler
;  722          0793  1 !       somewhere before the end of execution).  This routine will
;  723          0794  1 !       close TERMTABLE.EXE, if it is still open, with the DLT bit
;  724          0795  1 !       set.  We ignore errors since there is a possibility that we
;  725          0796  1 !       were interupted before TERMTABLE.EXE was opened, or just
;  726          0797  1 !       after we did a normal close.
;  727          0798  1 !
;  728          0799  1 ! CALLING SEQUENCE:
;  729          0800  1 !
;  730          0801  1 !       ret_status = CLOSE_FILES (EXIT_REASON.rl.r, PARAM_BLOCK.rz.r)
;  731          0802  1 !
;  732          0803  1 ! FORMAL PARAMETERS:                                                    53
;  733          0804  1 !
;  734          0805  1 !       EXIT_REASON.rl.r        Address of exit reason.  $DCLEXH passes
;  735          0806  1 !                               this although we don't need it here.
;  736          0807  1 !
;  737          0808  1 !       PARAM_BLOCK.rz.r        Address of control block with info needed
;  738          0809  1 !                               to access files
;  739          0810  1 !
;  740          0811  1 ! IMPLICIT INPUTS:
;  741          0812  1 !
;  742          0813  1 !       NONE
;  743          0814  1 !
;  744          0815  1 ! IMPLICIT OUTPUTS:
;  745          0816  1 !
;  746          0817  1 !       NONE
;  747          0818  1 !
;  748          0819  1 ! COMPLETION STATUS:
;  749          0820  1 !
;  750          0821  1 !       SS$_NORMAL
;  751          0822  1 !
;  752          0823  1 ! SIDE EFFECTS:
;  753          0824  1 !
;  754          0825  1 !       NONE
;  755          082   1 !
;  756          0827  1 !--
;  757          0828  2     BEGIN
;  758          0829  2     BIND
;  759          0830  2         BINARY_FAB = .PARAM_BLOCK [PARAM_A_BINARY_FAB] : $FAB_DECL;
;  760          0831  2
;  761          0832  2     BINARY_FAB [FAB$V_DLT] = 1;          ! mark the file for delete
;  762          0833  2
;  763          0834  2     $CLOSE (FAB = BINARY_FAB);
;  764          0835  2
;  765          0836  2     RETURN (SS$_NORMAL);
;  766          0837  1     END;                                 ! end of routine CLOSE_WITH_DLT
```

```
                                    0000 00000 CLOSE_WITH_DLT:
                                                       .WORD   Save nothing                                    ; 0784
                          50    08    AC  D0 00002             MOVL    PARAM_BLOCK, R0                         ; 0830
                          50    2C    A0  D0 00006             MOVL    44(R0), R0                              ;
                       05 A0    80    8F  88 0000A             BISB2   #128, 5(R0)                             ; 0832
                                      50  DD 0000F             PUSHL   R0                                      ; 0834
                00000000G  00         01  FB 00011             CALLS   #1, SYS$CLOSE                           ;
                          50          01  D0 00018             MOVL    #1, R0                                  ; 0836
                                          04 0001B             RET                                             ; 0837

; Routine Size:  28 bytes,    Routine Base:  _SMG$CODE + 044B

;  767          0838  1 !<BLF/PAGE>
```

```
;  769          0839  1 END                          ! End of module SMG$BUILD_TERM_TABLE
;  770          0840  1
;  771          0841  0 ELUDOM


                                                      .EXTRN  LIB$STOP

;                               PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;   _SMG$DATA                     28  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,  PIC,ALIGN(2)
;   _SMG$CODE                   1127  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        97         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         0         0          38       00:00.4
;   _$255$DUA28:[SMGRTL.OBJ]SMGTPALIB.L32;1
;                                           41        18        43          10       00:00.1
;   _$255$DUA28:[SYSLIB]TPAMAC.L32;1        42         0         0          14       00:00.1



;                                    COMMAND QUALIFIERS

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

; Size:          1101 code + 54 data bytes
; Run Time:        00:23.8
; Elapsed Time:    01:25.1
; Lines/CPU Min:    2119
; Lexemes/CPU-Min: 30869
; Memory Used:  181 pages
; Compilation Complete
```

SMGDATSTR
REQ

SMGLNK
REQ

SMGKQB
SDL

SMGBLDTRM
LIS

SMGDISCHA
LIS

SMGKTH
SDL

SMGMACROS
REQ

SMGPROLOG
REQ

SMGTERM
REQ

SMGTRMSTR
R32

SMGTRMPTR
SDL

SMGSCRMAC
REQ

SMGALLESC
LIS

SMGTPACTL
REQ

SMGSCRTCB
REQ

SMGLIB
REQ

SMGTPALIB
REQ

SMGDEFKEY
LIS

SMGTABDEF
REQ

SMGTRMMAC
REQ

SMGBOOTAB
LIS