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

```
_$2
Val
---
001
001
001
001
001
001
001
001
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
```

```
                                   D  8
**FILE**ID**SMGKEYPAD











  SSSSSSSS  MM      MM    GGGGGGGG  KK      KK  EEEEEEEEEE  YY      YY  PPPPPPPP      AAAAAA    DDDDDDDD
  SSSSSSSS  MM      MM    GGGGGGGG  KK      KK  EEEEEEEEEE  YY      YY  PPPPPPPP      AAAAAA    DDDDDDDD
SS          MMMM  MMMM  GG          KK      KK  EE          YY      YY  PP      PP  AA      AA  DD      DD
SS          MMMM  MMMM  GG          KK      KK  EE          YY      YY  PP      PP  AA      AA  DD      DD
SS          MM  MM  MM  GG          KK    KK    EE            YY  YY    PP      PP  AA      AA  DD      DD
SS          MM  MM  MM  GG          KK    KK    EE            YY  YY    PP      PP  AA      AA  DD      DD
  SSSSSS    MM      MM  GG          KKKKKK      EEEEEEEE        YY      PPPPPPPP    AA      AA  DD      DD
  SSSSSS    MM      MM  GG          KKKKKK      EEEEEEEE        YY      PPPPPPPP    AA      AA  DD      DD
        SS  MM      MM  GG  GGGGGG  KK    KK    EE              YY      PP          AAAAAAAAAA  DD      DD
        SS  MM      MM  GG  GGGGGG  KK    KK    EE              YY      PP          AAAAAAAAAA  DD      DD
        SS  MM      MM  GG      GG  KK      KK  EE              YY      PP          AA      AA  DD      DD      ....
        SS  MM      MM  GG      GG  KK      KK  EE              YY      PP          AA      AA  DD      DD      ....
SSSSSSSS    MM      MM    GGGGGG    KK      KK  EEEEEEEEEE      YY      PP          AA      AA  DDDDDDDD        ....
SSSSSSSS    MM      MM    GGGGGG    KK      KK  EEEEEEEEEE      YY      PP          AA      AA  DDDDDDDD        ....


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
```

```
0 MODULE SMG$KEYPAD ( %TITLE 'Screen Management Facility Keypad Procedures'
0               IDENT = '1-006'           ! File: SMGKEYPAD.B32 Edit: SBL1006
0               ) =
1 BEGIN
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
1 !*  ALL RIGHTS RESERVED.                                                     *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                             *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                             *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1 !
1
1 !++
1 ! FACILITY:     Screen Management Facility
1 !
1 ! ABSTRACT:
1 !
1 !       This module contains procedures to manage keypad key definitions.
1 !
1 ! ENVIRONMENT:  User mode - AST reentrant
1 !
1 ! AUTHOR: Steven B. Lionel, CREATION DATE: 10-Feb-1983
1 !
1 ! MODIFIED BY:
1 !
1 ! 1-001 - Original.  SBL 10-Feb-1983
1 ! 1-002 - Use 'DEFAULT' as the default state name.  SBL 18-May-1983
1 ! 1-003 - Keep the default state name in the KTH, so it can be sticky
1 !         across operations.  SBL 23-Jun-1983
1 ! 1-004 - Add SMG$LIST_KEY_DEFS and SMG$SET_DEFAULT_STATE.  When creating
1 !         a "total key", swap the bytes of the key code so that control
1 !         keys sort in the proper order.  SBL 10-Aug-1983
1 ! 1-005 - Set KTH_A_DEF_KEYCODE in SMG$SET_DEFAULT_STATE.  SBL 14-Sep-1983
1 ! 1-006 - Insert required NUL into TKEY in SMG$SET_DEFAULT_STATE.  SBL 15-Sep-1983
1 !--
1
```

```
%SBTTL 'Declarations'
!
! PROLOGUE FILE:
!

REQUIRE 'RTLIN:SMGPROLOG';                              ! Screen management definitions

!
! LINKAGE
!

LINKAGE
    CONSTRUCT_TKEY$LNK =
        CALL (REGISTER=6, REGISTER=7; REGISTER=7);

!
! TABLE OF CONTENTS:
!

FORWARD ROUTINE
    SMG$CREATE_KEY_TABLE,                               ! Create a key table
    SMG$ADD_KEY_DEF,                                    ! Add a key definition
    SMG$GET_KEY_DEF,                                    ! Get a key definition
    SMG$DELETE_KEY_DEF,                                 ! Delete a key definition
    SMG$SET_DEFAULT_STATE,                              ! Set or get a default state
    SMG$$LOOKUP_KEY: SMG$$LOOKUP_KEY$LNK,               ! Lookup a key
    CONSTRUCT_TKEY: CONSTRUCT_TKEY$LNK,                 ! Construct the total key
    COMPARE_ROUTINE,                                    ! Compare two entries
    ALLOC_ROUTINE,                                      ! Allocate an entry
    SMG$DEFINE_KEY,                                     ! Process DEFINE/KEY
    DEFINE_KEY_HANDLER,                                 ! Handler for SMG$DEFINE_KEY
    SMG$LOAD_KEY_DEFS,                                  ! Load a file of key defs
    SMG$LIST_KEY_DEFS,                                  ! List key definitions
    TRAVERSE_ROUTINE;                                   ! Traverse key table

!
! MACROS:
!
!       NONE
!
! EQUATED SYMBOLS:
!

!+
! Define a mask of those control keys used by advanced line editing.
!-

LITERAL
    K_CTRLA = 1,
    K_CTRLB = 2,
    K_CTRLD = 4,
    K_CTRLE = 5,
    K_CTRLF = 6,
    K_BS = 8,
    K_HT = 9,
    K_LF = 10,
    M_EDIT_KEYS = (1^K_CTRLA) + (1^K_CTRLB) + (1^K_CTRLD) + (1^K_CTRLE) +
```

```
        (1^K_CTRLF) + (1^K_BS) + (1^K_LF);

!
! FIELDS:
!
!       NONE
!
! OWN STORAGE:
!
!       NONE
!
! EXTERNAL DECLARATIONS:
!

EXTERNAL ROUTINE
    CLI$DCL_PARSE,                      ! Parse command line
    CLI$GET_VALUE,                      ! Get value of item from command line
    CLI$PRESENT,                        ! Indicate whether item is present
    LIB$GET_VM,                         ! Allocate virtual memory
    LIB$ANALYZE_SDESC_R2: LIB$ANALYZE_SDESC_R2$LINKAGE, ! Analyze string desc
    LIB$SCOPY_R_DX6: LIB$SCOPY_R_DX6$LINKAGE,    ! Copy string
    LIB$SCOPY_DXDX6: LIB$SCOPY_DXDX6$LINKAGE,    ! Copy string
    LIB$SFREE1_DD,                               ! Free string
    LIB$SFREE1_DD6: LIB$SFREE1_DD6$LINKAGE,      ! Free string
    LIB$SFREEN_DD,                      ! Free strings
    LIB$SIGNAL,                         ! Signal continuable error
    LIB$INSERT_TREE,                    ! Insert item in balanced tree
    LIB$LOOKUP_TREE,                    ! Find item in tree
    LIB$TRAVERSE_TREE,                  ! Traverse balanced tree
    SMG$CREATE_VIRTUAL_KEYBOARD,        ! Create a virtual keyboard
    SMG$DELETE_VIRTUAL_KEYBOARD,        ! Delete a virtual keyboard
    SMG$READ_STRING,                    ! Read a string
    SMG$$KEYCODE_TO_NAME,               ! Convert key code to name
    SMG$$NAME_TO_KEYCODE;               ! Convert key name to key code

EXTERNAL LITERAL
    CLI$_FACILITY: UNSIGNED(6),         ! CLI facility code
    CLI$_NOCOMD,                        ! No command on line
    SMG$_FILTOOLON,                     ! Filespec too long
    SMG$_INVDEFATT,                     ! Invalid key-definition attributes
    SMG$_INVKEYNAM,                     ! Invalid key name
    SMG$_INVSTANAM,                     ! Invalid state name
    SMG$_KEYDEFPRO,                     ! Key definition is protected
    SMG$_KEYNOTDEF,                     ! Key is not defined
    SMG$_NOMOREKEYS,                    ! No more keys
    SMG$_PREDEFREP;                     ! Previous key-definition replaced

EXTERNAL
    LIB$AB_UPCASE,                      ! Upcase translate table
    SMG$$AB_DEFKEY_CLD;                 ! Command line definition for DEFINE/KEY
```

```
1 %SBTTL 'SMG$CREATE_KEY_TABLE'
1 GLOBAL ROUTINE SMG$CREATE_KEY_TABLE (
1     KEY_TABLE_ID: REF VECTOR [, LONG]
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure creates a key-definition table, which can then have
1 !       key definitions added to it by calls to SMG$ADD_KEY_DEF,
1 !       SMG$DEFINE_KEY and SMG$LOAD_KEY_DEFS.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       RET_STATUS.wlc.v = SMG$CREATE_KEY_TABLE (
1 !                               KEY_TABLE_ID.wl.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       KEY_TABLE_ID    A longword into which is written the identification
1 !                       of the newly-created key-definition table.  This
1 !                       identification can then be passed to other Screen
1 !                       Management procedures to uniquely identify this set
1 !                       of key definitions.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       LIB$_INSVIRMEM  Insufficient virtual memory
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         KTH: REF KTH_R_KTH_STRUCT,            ! Key table header
2         VM_STATUS;
2
2     !+
2     ! Allocate and initialize the key table header.  For a default terminator
2     ! mask, use all control characters except those used by advanced line
2     ! editing and <HT>.
2     !-
2
2     VM_STATUS = LIB$GET_VM (%REF(KTH_S_KTH_STRUCT), KEY_TABLE_ID [0]);
```

```
2        IF NOT .VM_STATUS
2        THEN
2            RETURN .VM_STATUS;
2        KTH = .KEY_TABLE_ID [0];
2        KTH [KTH_A_TREEHEAD] = 0;
2        KTH [KTH_L_FLAGS] = 0;
2        KTH [KTH_L_MODIFIERS] = 0;
2        KTH [KTH_L_TERM_MASK] = %X'FFFFFFFF' - M_EDIT_KEYS - (1^K_HT);
2
2        !+
2        ! Establish 'DEFAULT' as the default state.  The string contains
2        ! the default state name, a byte of zero, and two bytes for the
2        ! keycode.
2        !-
2
2        KTH [KTH_W_DEF_STATE_LEN] = %CHARCOUNT('DEFAULT') + 3;
2        KTH [KTH_B_DEF_STATE_CLASS] = DSC$K_CLASS_S;
2        KTH [KTH_B_DEF_STATE_DTYPE] = DSC$K_DTYPE_T;
2        KTH [KTH_A_DEF_STATE_POINTER] = KTH [KTH_T_DEF_STATE_STRING];
2        KTH [KTH_A_DEF_KEYCODE] = CH$MOVE (
2            %CHARCOUNT('DEFAULT') + 1,
2            UPLIT BYTE ('DEFAULT',0),
2            KTH [KTH_T_DEF_STATE_STRING]);
2
2        KTH [KTH_L_CHECK] = .KTH;
2
2        RETURN SS$_NORMAL;
2
1        END;                                  ! End of routine SMG$CREATE_KEY_TABLE

                                                      .TITLE  SMG$KEYPAD Screen Management Facility Keypad Pr
;                                                                        ocedures
                                                      .IDENT  \1-006\

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

        54  4C  55  41  46  45  44  00000 P.AAA:  .ASCII  \DEFAULT\                      ;
                                00  00007         .BYTE   0                              ;

                                                  .EXTRN  CLI$DCL_PARSE, CLI$GET_VALUE
                                                  .EXTRN  CLI$PRESENT, LIB$GET_VM
                                                  .EXTRN  LIB$ANALYZE_SDESC_R2
                                                  .EXTRN  LIB$SCOPY_R_DX6
                                                  .EXTRN  LIB$SCOPY_DXDX6
                                                  .EXTRN  LIB$SFREE1_DD, LIB$SFREE1_DD6
                                                  .EXTRN  LIB$SFREEN_DD, LIB$SIGNAL
                                                  .EXTRN  LIB$INSERT_TREE
                                                  .EXTRN  LIB$LOOKUP_TREE
                                                  .EXTRN  LIB$TRAVERSE_TREE
                                                  .EXTRN  SMG$CREATE_VIRTUAL_KEYBOARD
                                                  .EXTRN  SMG$DELETE_VIRTUAL_KEYBOARD
                                                  .EXTRN  SMG$READ_STRING
                                                  .EXTRN  SMG$$KEYCODE_TO_NAME
                                                  .EXTRN  SMG$$NAME_TO_KEYCODE
                                                  .EXTRN  CLI$_FACILITY, CLI$_NOCOMD
                                                  .EXTRN  SMG$_FILTOOLON, SMG$_INVDEFATT
```

```
                                                        .EXTRN   SMG$_INVKEYNAM, SMG$_INVSTANAM
                                                        .EXTRN   SMG$_KEYDEFPRO, SMG$_KEYNOTDEF
                                                        .EXTRN   SMG$_NOMOREKEYS
                                                        .EXTRN   SMG$_PREDEFREP, LIB$AB_UPCASE
                                                        .EXTRN   SMG$$AB_DEFKEY_CLD

                                         007C 00000     .ENTRY   SMG$CREATE_KEY_TABLE, Save R2,R3,R4,R5,R6   ; 0239
                      5E             04    C2 00002     SUBL2    #4, SP                                       ;
                                  04    AC DD 00005     PUSHL    KEY_TABLE_ID                                 ; 0294
                  04  AE          42    8F 9A 00008     MOVZBL   #66, 4(SP)                                   ;
                                  04    AE 9F 0000D     PUSHAB   4(SP)                                        ;
          00000000G   00                02 FB 00010     CALLS    #2, LIB$GET_VM                               ;
                      2D                50 E9 00017     BLBC     VM_STATUS, 1$                                ; 0295
                      56          04    BC D0 0001A     MOVL     @KEY_TABLE_ID, KTH                           ; 0298
                                        66 7C 0001E     CLRQ     (KTH)                                        ; 0299
                                  08    A6 D4 00020     CLRL     8(KTH)                                       ; 0301
                  0C  A6        F889    8F 32 00023     CVTWL    #-1911, 12(KTH)                              ; 0302
                  14  A6    010E000A    8F D0 00029     MOVL     #17694730, 20(KTH)                           ; 0310
                  18  A6          20    A6 9E 00031     MOVAB    32(KTH), 24(KTH)                             ; 0313
     20     A6    BE  AF                08 28 00036     MOVC3    #8, P.AAA, 32(KTH)                           ; 0317
                  1C  A6                53 D0 0003C     MOVL     R3, 28(KTH)                                  ;
                  10  A6                56 D0 00040     MOVL     KTH, 16(KTH)                                 ; 0319
                      50                01 D0 00044     MOVL     #1, R0                                       ; 0321
                                           04 00047 1$: RET                                                   ; 0323

; Routine Size:  72 bytes,    Routine Base:  _SMG$CODE + 0008
```

```
%SBTTL 'SMG$ADD_KEY_DEF'
GLOBAL ROUTINE SMG$ADD_KEY_DEF (
    KEY_TABLE_ID: REF VECTOR [, LONG],
    KEY_NAME: REF BLOCK [, BYTE],
    IF_STATE: REF BLOCK [, BYTE],
    ATTRIBUTES: REF VECTOR [, LONG],
    EQUIV_STRING: REF BLOCK [, BYTE],
    STATE_STRING: REF BLOCK [, BYTE]
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure adds a key definition to a table of key definitions.
!       The table must have been created by a call to SMG$CREATE_KEY_TABLE.
!
! CALLING SEQUENCE:
!
!       RET_STATUS.wlc.v = SMG$ADD_KEY_DEF (
!                               KEY_TABLE_ID.rl.r,
!                               KEY_NAME.rt.dx
!                               [, [IF_STATE.rt.dx]
!                               [, [ATTRIBUTES.rl.r]
!                               [, [EQUIV_STRING.rt.dx]
!                               [, [STATE_STRING.rt.dx] ]]]])
!
! FORMAL PARAMETERS:
!
!       KEY_TABLE_ID    A longword containing the identification of the
!                       key-definition table to which this definition is
!                       to be added.
!
!       KEY_NAME        A string containing the name of the key to be defined.
!                       The key name is stripped of trailing blanks and is
!                       converted to upper-case before use.  The key name must
!                       be one of the key names listed in table _\_\ TBS _\_\.
!
!       IF_STATE        A string containing a state name which further qualifies
!                       KEY_NAME.  If specified, this definition of KEY_NAME is
!                       used only if the current state is IF_STATE.  See the
!                       description of key states in the Screen Management
!                       Functional Specification.  If omitted, 'DEFAULT'
!                       is used.
!
!       ATTRIBUTES      A longword bit mask specifying additional attributes of
!                       this key definition.  If omitted, a mask of zero is
!                       used.  The attributes defined are:
!
!                         SMG$V_KEY_NOECHO - Bit 0
!                           If set, specifies that EQUIV_STRING is not to
!                           be echoed when this key is pressed.  If clear,
!                           EQUIV_STRING is echoed.  If SMG$V_KEY_TERMINATE
!                           is not set, SMG$V_KEY_NOECHO is ignored.
!
!                         SMG$V_KEY_TERMINATE - Bit 1
!                           If set, specifies that when this key is pressed
!                           (qualified by IF_STATE), that the input line is
```

```
1 !                                 complete and that more characters should not be
1 !                                 accepted.  If clear, more more characters may be
1 !                                 accepted.
1 !
1 !                               SMG$V_KEY_LOCKSTATE - Bit 2
1 !                                 If set, and if STATE_STRING is specified, the
1 !                                 state name specified by STATE_STRING remains the
1 !                                 current state until explicitly changed by a
1 !                                 subsequent keystroke whose definition includes
1 !                                 a STATE_STRING.  If clear, the state name specified
1 !                                 by STATE_STRING remains in effect only for the next
1 !                                 defineable key.
1 !
1 !                               SMG$V_KEY_PROTECTED - Bit 3
1 !                                 If set, indicates that this key definition cannot
1 !                                 be modified or deleted.  If clear, indicates that
1 !                                 this key definition can be modified or deleted.
1 !
1 !                               SMG$V_KEY_SETSTATE - Bit 4
1 !                                 This bit is ignored by SMG$ADD_KEY_DEF.  It is
1 !                                 returned by SMG$GET_KEY_DEF and SMG$LIST_KEY_DEFS
1 !                                 to indicate that the key sets a state.
1 !
1 !                             The remaining bits are undefined and must be zero.
1 !
1 !       EQUIV_STRING          A string which is to be substituted for the keystroke
1 !                             in the returned line.  Unless SMG$V_NOECHO is specified,
1 !                             the equivalence string is echoed.  If omitted, no
1 !                             equivalence string is defined for this key.
1 !
1 !       STATE_STRING          A string containing a new state name which is to become
1 !                             the current state when this key is pressed.  If omitted,
1 !                             no new state is defined.  If the current state is
1 !                             temporary (SMG$V_KEY_LOCKSTATE was not specified for
1 !                             the previous key), the current state reverts to
1 !                             the default state.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL            Normal successful completion
1 !       SMG$_INVDEFATT        Invalid key-definition attributes
1 !       SMG$_INVKTB_ID        Invalid key-table id
1 !       SMG$_KEYDEFPRO        Key-definition is protected against change or deletion
1 !       SMG$_PREDEFREP        Previous key-definition replaced
1 !       LIB$_INSVIRMEM        Insufficient virtual memory
1 !       LIB$_xxx              Any error status from LIB$SCOPY_DXDX
1 !
1 ! SIDE EFFECTS:
1 !
```

```
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         KTH: REF KTH_R_KTH_STRUCT,                  ! Key table header
2         KDE: REF KDE_R_KDE_STRUCT,                  ! Key definition entry
2         NEW_KDE,                                    ! Pointer to new KDE
2         TKEY: VECTOR [34, BYTE],                    ! Total key (state-key)
2         TKEY_DESC: BLOCK [8, BYTE],                 ! Descriptor for TKEY
2         TKEY_DESC_PTR: REF BLOCK [8, BYTE],         ! Pointer to a TKEY desc
2         TKEY_STATUS,                                ! Status from CONSTRUCT_TKEY
2         INSERT_STATUS,                              ! Status from insert
2         COPY_STATUS1, COPY_STATUS2,                 ! Status from calls to
2         COPY_STATUS3, COPY_STATUS4,                 ! LIB$SCOPY_ routines
2         RET_STATUS;                                 ! Our return status
2
2     BUILTIN
2         AP,
2         NULLPARAMETER;
2
2     !+
2     ! Define positions of arguments
2     !-
2
2     LITERAL
2         K_KEY_TABLE_ID = 1,
2         K_KEY_NAME = 2,
2         K_IF_STATE = 3,
2         K_ATTRIBUTES = 4,
2         K_EQUIV_STRING = 5,
2         K_STATE_STRING = 6;
2
2     !+
2     ! Validate KEY_TABLE_ID and get the Key Table Header.
2     !-
2
2     $SMG$VALIDATE_KTH (KEY_TABLE_ID, KTH);
2
2     !+
2     ! Construct TKEY from IF_STATE and KEY_NAME.
2     !-
2
2     TKEY_DESC [DSC$A_POINTER] = TKEY;
2     TKEY_DESC_PTR = TKEY_DESC;
2     TKEY_STATUS = CONSTRUCT_TKEY (.AP, .TKEY_DESC_PTR; TKEY_DESC_PTR);
2     IF NOT .TKEY_STATUS
2     THEN
2         RETURN .TKEY_STATUS;
2
2     INSERT_STATUS = LIB$INSERT_TREE (
2         KTH [KTH_A_TREEHEAD],       ! Address of treehead,
2         TKEY_DESC_PTR [0,0,0,0],    ! Descriptor of TKEY,
2         %REF(0),                    ! Return address of existing entry
2         COMPARE_ROUTINE,            ! Address of compare routine
```

```
          ALLOC_ROUTINE,              ! Address of allocation routine
          NEW_KDE,                    ! Address of new KDE
          0);                         ! Context - not used

      IF NOT .INSERT_STATUS
      THEN
          RETURN .INSERT_STATUS;

      !+
      ! Check for previously defined key.
      !-

      KDE = .NEW_KDE;
      RET_STATUS = SS$_NORMAL;        ! Assume not previously defined
      IF .KDE [KDE_V_DEFINED]         ! Already defined?
      THEN
          BEGIN
          RET_STATUS = SMG$_PREDEFREP;    ! Success - Previous key definition replaced
          IF .KDE [KDE_V_PROTECTED]
          THEN
              RETURN SMG$_KEYDEFPRO;      ! Key definition protected
          END;

      !+
      ! Turn off DEFINED and user attribute bits in KDE.  Other bits might
      ! still be set.
      ! Store items in new KDE.
      !-

      KDE [KDE_V_DEFINED] = 0;
      KDE [KDE_V_USER_ATTR] = 0;
      IF NOT NULLPARAMETER (K_ATTRIBUTES) ! ATTRIBUTES present?
      THEN
          BEGIN
          KDE [KDE_V_USER_ATTR] = .ATTRIBUTES [0];
          KDE [KDE_V_SETSTATE] = 0;       ! Assume no state set.
          IF (.KDE [KDE_L_ATTR] AND KDE_M_RESERVED) NEQ 0
          THEN
              RETURN SMG$_INVDEFATT;      ! Invalid key-definition attributes
          IF NOT .KDE [KDE_V_TERMINATE]
          THEN
              KDE [KDE_V_NOECHO] = 0;     ! NOTERMINATE implies ECHO
          END;

      COPY_STATUS1 = LIB$SCOPY_R_DX6 (.TKEY_DESC_PTR [DSC$W_LENGTH],
          .TKEY_DESC_PTR [DSC$A_POINTER], KDE [KDE_R_TKEY_DESC]);
      IF NOT .COPY_STATUS1
      THEN
          RETURN .COPY_STATUS1;

      IF NOT NULLPARAMETER (K_EQUIV_STRING)
      THEN
          BEGIN
          COPY_STATUS2 = LIB$SCOPY_DXDX6 (EQUIV_STRING [0,0,0,0],
              KDE [KDE_R_EQUIV_DESC]);
          IF NOT .COPY_STATUS2
          THEN
```

```
3                   RETURN .COPY_STATUS2;
3               END
2           ELSE IF .KDE [KDE_A_EQUIV_POINTER] NEQA 0
2           THEN
2               LIB$SFREE1_DD6 (KDE [KDE_R_EQUIV_DESC]);
2
2           IF NOT NULLPARAMETER (K_STATE_STRING)          ! STATE present?
2           THEN
3               BEGIN
3               COPY_STATUS3 = LIB$SCOPY_DXDX6 (STATE_STRING [0,0,0,0],
3                   KDE [KDE_R_STATE_DESC]);
3               IF NOT .COPY_STATUS3
3               THEN
3                   RETURN .COPY_STATUS3;
3               KDE [KDE_V_SETSTATE] = 1;
3               END
2           ELSE IF .KDE [KDE_A_STATE_POINTER] NEQA 0
2           THEN
2               LIB$SFREE1_DD6 (KDE [KDE_R_STATE_DESC]);
2
2           !+
2           ! If this is a control key, add that key's
2           ! bit to the terminator mask.
2           !-
2
2           KTH [KTH_L_TERM_MASK] = .KTH [KTH_L_TERM_MASK] OR
2               .KDE [KDE_L_TERM_MASK];
2
2           !+
2           ! Indicate that definition is valid and return success.
2           !-
2
2           KDE [KDE_V_DEFINED] = 1;
2           RETURN .RET_STATUS;
2
1           END;                                    ! End of routine SMG$ADD_KEY_DEF


                                                    .EXTRN  SMG$_INVKTB_ID

                                 0FFC 00000         .ENTRY  SMG$ADD_KEY_DEF, Save R2,R3,R4,R5,R6,R7,R8,-; 0325
                                                            R9,R10,R11                              ;
                 5E           34  C2 00002          SUBL2   #52, SP                                 ;
                 5A      04   BC  D0 00005          MOVL    @KEY_TABLE_ID, KTH                      ; 0477
                              06  13 00009          BEQL    1$                                      ;
                 5A      10   AA  D1 0000B          CMPL    16(KTH), KTH                            ;
                              08  13 0000F          BEQL    2$                                      ;
                 50 00000000G 8F  D0 00011 1$:      MOVL    #SMG$_INVKTB_ID, R0                     ;
                                  04 00018          RET                                             ;
         0C      AE      10   AE  9E 00019 2$:      MOVAB   TKEY, TKEY_DESC+4                       ; 0483
                 57      08   AE  9E 0001E          MOVAB   TKEY_DESC, TKEY_DESC_PTR                ; 0484
                 56           5C  D0 00022          MOVL    AP, R6                                  ; 0485
       0000V     CF           00  FB 00025          CALLS   #0, CONSTRUCT_TKEY                      ;
                 1E           50  E9 0002A          BLBC    TKEY_STATUS, 3$                         ; 0486
                              7E  D4 0002D          CLRL    -(SP)                                   ; 0492
                         08   AE  9F 0002F          PUSHAB  NEW_KDE                                 ;
```

```
                           0000V  CF  9F  00032       PUSHAB  ALLOC_ROUTINE                        ;
                           0000V  CF  9F  00036       PUSHAB  COMPARE_ROUTINE                      ;
                              10  AE  D4  0003A       CLRL    16(SP)                               ; 0493
                              10  AE  9F  0003D       PUSHAB  16(SP)                               ;
                                  57  DD  00040       PUSHL   TKEY_DESC_PTR                        ; 0492
                                  5A  DD  00042       PUSHL   KTH                                  ;
           00000000G  00          07  FB  00044       CALLS   #7, LIB$INSERT_TREE                  ;
                      60          50  E9  0004B 3$:   BLBC    INSERT_STATUS, 7$                    ; 0499
                      58      04  AE  D0  0004E       MOVL    NEW_KDE, KDE                         ; 0507
                      5B          01  D0  00052       MOVL    #1, RET_STATUS                       ; 0508
                      59      30  A8  9E  00055       MOVAB   48(KDE), R9                          ; 0509
                                  69  D5  00059       TSTL    (R9)                                 ;
                                  13  18  0005B       BGEQ    4$                                   ;
                      5B  00000000G   8F  D0  0005D   MOVL    #SMG$_PREDEFREP, RET_STATUS          ; 0512
    08                69          03  E1  00064       BBC     #3, (R9), 4$                         ; 0513
                      50  00000000G   8F  D0  00068   MOVL    #SMG$_KEYDEFPRO, R0                  ; 0515
                                      04  0006F       RET                                          ;
                      69  800000FF    8F  CA  00070 4$:   BICL2   #-2147483393, (R9)               ; 0524
                      04          6C  91  00077       CMPB    (AP), #4                             ; 0526
                                  21  1F  0007A       BLSSU   6$                                   ;
                              10  AC  D5  0007C       TSTL    16(AP)                               ;
                                  1C  13  0007F       BEQL    6$                                   ;
                      69      10  BC  90  00081       MOVB    @ATTRIBUTES, (R9)                    ; 0529
                      69          10  8A  00085       BICB2   #16, (R9)                            ; 0530
              E0      8F          69  93  00088       BITB    (R9), #224                           ; 0531
                                  08  13  0008C       BEQL    5$                                   ;
                      50  00000000G   8F  D0  0008E   MOVL    #SMG$_INVDEFATT, R0                  ; 0533
                                      04  00095       RET                                          ;
    03                69          01  E0  00096 5$:   BBS     #1, (R9), 6$                         ; 0534
                      69          01  8A  0009A       BICB2   #1, (R9)                             ; 0536
                      52      0C  A8  9E  0009D 6$:   MOVAB   12(KDE), R2                          ; 0540
                      51      04  A7  D0  000A1       MOVL    4(TKEY_DESC_PTR), R1                 ;
                      50          67  3C  000A5       MOVZWL  (TKEY_DESC_PTR), R0                  ;
               00000000G          00  16  000A8       JSB     LIB$SCOPY_R_DX6                      ;
                      67          50  E9  000AE 7$:   BLBC    COPY_STATUST, 12$                    ; 0541
                      05          6C  91  000B1       CMPB    (AP), #5                             ; 0545
                                  17  1F  000B4       BLSSU   8$                                   ;
                              14  AC  D5  000B6       TSTL    20(AP)                               ;
                                  12  13  000B9       BEQL    8$                                   ;
                      51      14  A8  9E  000BB       MOVAB   20(KDE), R1                          ; 0549
                      50      14  AC  D0  000BF       MOVL    EQUIV_STRING, R0                     ;
               00000000G          00  16  000C3       JSB     LIB$SCOPY_DXDX6                      ;
                      10          50  E8  000C9       BLBS    COPY_STATUS2, 9$                     ; 0550
                                      04  000CC       RET                                          ; 0552
                              18  A8  D5  000CD 8$:   TSTL    24(KDE)                              ; 0554
                                  0A  13  000D0       BEQL    9$                                   ;
                      50      14  A8  9E  000D2       MOVAB   20(KDE), R0                          ; 0556
               00000000G          00  16  000D6       JSB     LIB$SFREE1_DD6                       ;
                      06          6C  91  000DC 9$:   CMPB    (AP), #6                             ; 0558
                                  1B  1F  000DF       BLSSU   10$                                  ;
                              18  AC  D5  000E1       TSTL    24(AP)                               ;
                                  16  13  000E4       BEQL    10$                                  ;
                      51      1C  A8  9E  000E6       MOVAB   28(KDE), R1                          ; 0562
                      50      18  AC  D0  000EA       MOVL    STATE_STRING, R0                     ;
               00000000G          00  16  000EE       JSB     LIB$SCOPY_DXDX6                      ;
                      21          50  E9  000F4       BLBC    COPY_STATUS3, 12$                    ; 0563
                      69          10  88  000F7       BISB2   #16, (R9)                            ; 0566
```

```
                              0F  11 000FA        BRB     11$                             ; 0558
                     20       A8  D5 000FC  10$:  TSTL    32(KDE)                         ; 0568
                              0A  13 000FF        BEQL    11$                             ;
           50        1C       A8  9E 00101        MOVAB   28(KDE), R0                     ; 0570
               00000000G      00  16 00105        JSB     LIB$SFREE1_DD6                  ;
      0C   AA        34       A8  C8 0010B  11$:  BISL2   52(KDE), 12(KTH)                ; 0578
      03   A9        80       8F  88 00110        BISB2   #128, 3(R9)                     ; 0584
           50                 5B  D0 00115        MOVL    RET_STATUS, R0                  ; 0585
                                  04 00118  12$:  RET                                     ; 0587

; Routine Size:  281 bytes,    Routine Base:  _SMG$CODE + 0050
```

```
%SBTTL 'SMG$GET_KEY_DEF'
GLOBAL ROUTINE SMG$GET_KEY_DEF (
    KEY_TABLE_ID: REF VECTOR [, LONG],
    KEY_NAME: REF BLOCK [, BYTE],
    IF_STATE: REF BLOCK [, BYTE],
    ATTRIBUTES: REF VECTOR [, LONG],
    EQUIV_STRING: REF BLOCK [, BYTE],
    STATE_STRING: REF BLOCK [, BYTE]
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure returns the key definition associated with a
!       specified key and state name.
!
! CALLING SEQUENCE:
!
!       RET_STATUS.wlc.v = SMG$GET_KEY_DEF (
!                               KEY_TABLE_ID.rl.r,
!                               KEY_NAME.rt.dx
!                               [, [IF_STATE.rt.dx]
!                               [, [ATTRIBUTES.wl.r]
!                               [, [EQUIV_STRING.wt.dx]
!                               [, [STATE_STRING.wt.dx] ]]]])
!
! FORMAL PARAMETERS:
!
!       KEY_TABLE_ID    A longword containing the identification of the
!                       key-definition table in which this key is
!                       to be looked up.
!
!       KEY_NAME        A string containing the name of the key whose
!                       definition is to be returned.
!                       The key name is stripped of trailing blanks and is
!                       converted to upper-case before use.  The key name must
!                       be one of the key names listed in table _\_\ TBS _\_\.
!
!       IF_STATE        A string containing a state name which further qualifies
!                       KEY_NAME. If omitted, the null state is used.
!
!       ATTRIBUTES      A longword into which is stored the attributes
!                       bitmask for this key definition.  See the description
!                       of SMG$ADD_KEY_DEF for more information.
!
!       EQUIV_STRING    A string into which is stored the equivalence-string
!                       for this key definition.
!
!       STATE_STRING    A string into which is stored the new state-name,
!                       if any, which is set by this key definition.  If this
!                       key definition sets a state, the ATTRIBUTES flag
!                       SMG$V_KEY_SETSTATE is set.
!
! IMPLICIT INPUTS:
!
!       NONE
!
```

```
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL        Normal successful completion
!       SMG$_INVKTB_ID    Invalid key-table id
!       SMG$_KEYNOTDEF    Key is not defined
!       LIB$_xxx          Any error status from LIB$SCOPY_DXDX
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        FOUND_KDE,                              ! KDE found
        KDE: REF KDE_R_KDE_STRUCT,              ! Key definition entry
        KTH: REF KTH_R_KTH_STRUCT,              ! Key table header
        TKEY: VECTOR [34, BYTE],                ! Total key (state-key)
        TKEY_DESC: BLOCK [8, BYTE],             ! Descriptor for TKEY
        TKEY_DESC_PTR: REF BLOCK [8, BYTE],     ! Pointer to TKEY_DESC
        TKEY_STATUS,                            ! Status from CONSTRUCT_TKEY
        LOOKUP_STATUS,                          ! Status from LIB$LOOKUP_TREE
        COPY_STATUS1, COPY_STATUS2,             ! Status from calls to
        COPY_STATUS3;                           ! LIB$SCOPY_ routines

    BUILTIN
        AP,
        NULLPARAMETER;

    !+
    ! Define positions of arguments
    !-

    LITERAL
        K_KEY_TABLE_ID = 1,
        K_KEY_NAME = 2,
        K_IF_STATE = 3,
        K_ATTRIBUTES = 4,
        K_EQUIV_STRING = 5,
        K_STATE_STRING = 6;

    !+
    ! Validate KEY_TABLE_ID and get the Key Table Header.
    !-

    $SMG$VALIDATE_KTH (KEY_TABLE_ID, KTH);

    !+
    ! Construct TKEY from IF_STATE and KEY_NAME.
    !-

```

```
TKEY_DESC [DSC$A_POINTER] = TKEY;
TKEY_DESC_PTR = TKEY_DESC;
TKEY_STATUS = CONSTRUCT_TKEY (.AP, .TKEY_DESC_PTR; TKEY_DESC_PTR);
IF NOT .TKEY_STATUS
THEN
    RETURN .TKEY_STATUS;

!+
! Look up key name.
!-

LOOKUP_STATUS = LIB$LOOKUP_TREE (
    KTH [KTH_A_TREEHEAD],              ! Address of treehead
    .TKEY_DESC_PTR,                    ! Total state-key string
    COMPARE_ROUTINE,                   ! Key comparison routine
    FOUND_KDE);                        ! Pointer to found KDE
IF NOT .LOOKUP_STATUS
THEN
    RETURN SMG$_KEYNOTDEF;             ! Key not defined

!+
! If KDE is not defined, return "not found".
!-

KDE = .FOUND_KDE;
IF NOT .KDE [KDE_V_DEFINED]
THEN
    RETURN SMG$_KEYNOTDEF;  ! Key not defined

!+
! Store values.
!-

IF NOT NULLPARAMETER (K_ATTRIBUTES)
THEN
    ATTRIBUTES [0] = .KDE [KDE_L_ATTR] AND KDE_M_USER_ATTR;

IF NOT NULLPARAMETER (K_EQUIV_STRING)
THEN
    BEGIN
    COPY_STATUS1 = LIB$SCOPY_DXDX6 (KDE [KDE_R_EQUIV_DESC],
        EQUIV_STRING [0,0,0,0]);
    IF NOT .COPY_STATUS1
    THEN
        RETURN .COPY_STATUS1;
    END;

IF NOT NULLPARAMETER (K_STATE_STRING)
THEN
    BEGIN
    COPY_STATUS2 = LIB$SCOPY_DXDX6 (KDE [KDE_R_STATE_DESC],
        STATE_STRING [0,0,0,0]);
    IF NOT .COPY_STATUS2
    THEN
        RETURN .COPY_STATUS2;
    END;
```

```
;  686        0759  2     RETURN SS$_NORMAL;
;  687        0760  2
;  688        0761  1     END;                                              ! End of routine SMG$GET_KEY_DEF

                                          01FC 00000        .ENTRY  SMG$GET_KEY_DEF, Save R2,R3,R4,R5,R6,R7,R8   ; 0589
                        58 00000000G   00   9E 00002        MOVAB   LIB$SCOPY_DXDX6, R8                         ;
                        5E             30   C2 00009        SUBL2   #48, SP                                     ;
                        52        04   BC   D0 0000C        MOVL    @KEY_TABLE_ID, KTH                          ; 0696
                                       06   13 00010        BEQL    1$                                          ;
                        52        10   A2   D1 00012        CMPL    16(KTH), KTH                                ;
                                       08   13 00016        BEQL    2$                                          ;
                        50 00000000G   8F   D0 00018 1$:    MOVL    #SMG$_INVKTB_ID, R0                         ;
                                            04 0001F        RET                                                 ;
                  08    AE        0C   AE   9E 00020 2$:    MOVAB   TKEY, TKEY_DESC+4                           ; 0702
                        57        04   AE   9E 00025        MOVAB   TKEY_DESC, TKEY_DESC_PTR                    ; 0703
                        56             5C   D0 00029        MOVL    AP, R6                                      ; 0704
                0000V   CF             00   FB 0002C        CALLS   #0, CONSTRUCT_TKEY                          ;
                        64             50   E9 00031        BLBC    TKEY_STATUS, 8$                             ; 0705
                                       5E   DD 00034        PUSHL   SP                                          ; 0714
                               0000V   CF   9F 00036        PUSHAB  COMPARE_ROUTINE                             ;
                               0084    8F   BB 0003A        PUSHR   #^M<R2,R7>                                  ;
            00000000G   00             04   FB 0003E        CALLS   #4, LIB$LOOKUP_TREE                         ;
                        08             50   E9 00045        BLBC    LOOKUP_STATUS, 3$                           ; 0718
                        57             6E   D0 00048        MOVL    FOUND_KDE, KDE                              ; 0726
                                  33   A7   95 0004B        TSTB    51(KDE)                                     ; 0727
                                       08   19 0004E        BLSS    4$                                          ;
                        50 00000000G   8F   D0 00050 3$:    MOVL    #SMG$_KEYNOTDEF, R0                         ; 0729
                                            04 00057        RET                                                 ;
                        04             6C   91 00058 4$:    CMPB    (AP), #4                                    ; 0735
                                       0A   1F 0005B        BLSSU   5$                                          ;
                                  10   AC   D5 0005D        TSTL    16(AP)                                      ;
                                       05   13 00060        BEQL    5$                                          ;
                  10    BC        30   A7   9A 00062        MOVZBL  48(KDE), @ATTRIBUTES                        ; 0737
                        05             6C   91 00067 5$:    CMPB    (AP), #5                                    ; 0739
                                       12   1F 0006A        BLSSU   6$                                          ;
                                  14   AC   D5 0006C        TSTL    20(AP)                                      ;
                                       0D   13 0006F        BEQL    6$                                          ;
                        50        14   A7   9E 00071        MOVAB   20(KDE), R0                                 ; 0742
                        51        14   AC   D0 00075        MOVL    EQUIV_STRING, R1                            ; 0743
                                       68   16 00079        JSB     LIB$SCOPY_DXDX6                             ;
                        1A             50   E9 0007B        BLBC    COPY_STATUS1, 8$                            ; 0744
                        06             6C   91 0007E 6$:    CMPB    (AP), #6                                    ; 0749
                                       12   1F 00081        BLSSU   7$                                          ;
                                  18   AC   D5 00083        TSTL    24(AP)                                      ;
                                       0D   13 00086        BEQL    7$                                          ;
                        50        1C   A7   9E 00088        MOVAB   28(KDE), R0                                 ; 0752
                        51        18   AC   D0 0008C        MOVL    STATE_STRING, R1                            ; 0753
                                       68   16 00090        JSB     LIB$SCOPY_DXDX6                             ;
                        03             50   E9 00092        BLBC    COPY_STATUS2, 8$                            ; 0754
                        50             01   D0 00095 7$:    MOVL    #1, R0                                      ; 0759
                                            04 00098 8$:    RET                                                 ; 0761

; Routine Size:  153 bytes,    Routine Base:  _SMG$CODE + 0169
```

```
1 %SBTTL 'SMG$DELETE_KEY_DEF'
1 GLOBAL ROUTINE SMG$DELETE_KEY_DEF (
1     KEY_TABLE_ID: REF VECTOR [, LONG],
1     KEY_NAME: REF BLOCK [, BYTE],
1     IF_STATE: REF BLOCK [, BYTE]
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure deletes a key definition from the specified table
1 !       of key definitions.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       RET_STATUS.wlc.v = SMG$DELETE_KEY_DEF (
1 !                               KEY_TABLE_ID.rl.r,
1 !                               KEY_NAME.rt.dx
1 !                               [, [IF_STATE.rt.dx] ])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       KEY_TABLE_ID    A longword containing the identification of the
1 !                       key-definition table from which this key is
1 !                       to be deleted.
1 !
1 !       KEY_NAME        A string containing the name of the key which is
1 !                       to be deleted.
1 !                       The key name is stripped of trailing blanks and is
1 !                       converted to upper-case before use.  The key name must
1 !                       be one of the key names listed in table _\_\ TBS _\_\.
1 !
1 !       IF_STATE        A string containing a state name which further qualifies
1 !                       KEY_NAME. If omitted, the null state is used.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       SMG$_INVKTB_ID  Invalid key-table id
1 !       SMG$_KEYNOTDEF  Key is not defined
1 !       SMG$_KEYDEFPRO  Key definition is protected
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
```

```
LOCAL
    FOUND_KDE,                                  ! KDE found
    KDE: REF KDE_R_KDE_STRUCT,                  ! Key definition entry
    KTH: REF KTH_R_KTH_STRUCT,                  ! Ket table header
    TKEY: VECTOR [34, BYTE],                    ! Total key (state-key)
    TKEY_DESC: BLOCK [8, BYTE],                 ! Descriptor for TKEY
    TKEY_DESC_PTR: REF BLOCK [8, BYTE],         ! Pointer to a TKEY desc
    TKEY_STATUS,                                ! Status from CONSTRUCT_TKEY
    LOOKUP_STATUS;                              ! Status from LIB$LOOKUP_TREE

BUILTIN
    AP;

!+
! Define positions of arguments
!-

LITERAL
    K_KEY_TABLE_ID = 1,
    K_KEY_NAME = 2,
    K_IF_STATE = 3;

!+
! Validate KEY_TABLE_ID and get the Key Table Header.
!-

$SMG$VALIDATE_KTH (KEY_TABLE_ID, KTH);

!+
! Construct TKEY from IF_STATE and KEY_NAME.
!-

TKEY_DESC [DSC$A_POINTER] = TKEY;
TKEY_DESC_PTR = TKEY_DESC;
TKEY_STATUS = CONSTRUCT_TKEY (.AP, .TKEY_DESC_PTR; TKEY_DESC_PTR);
IF NOT .TKEY_STATUS
THEN
    RETURN .TKEY_STATUS;

!+
! Look up key name.
!-

LOOKUP_STATUS = LIB$LOOKUP_TREE (
    KTH [KTH_A_TREEHEAD],               ! Address of treehead
    .TKEY_DESC_PTR,                     ! Total state-key string
    COMPARE_ROUTINE,                    ! Key comparison routine
    FOUND_KDE);                         ! Pointer to found KDE
IF NOT .LOOKUP_STATUS
THEN
    RETURN SMG$_KEYNOTDEF;  ! Key not defined

KDE = .FOUND_KDE;
IF NOT .KDE [KDE_V_DEFINED]
THEN
    RETURN SMG$_KEYNOTDEF;  ! Key not defined
```

```
!+
! Delete key definition by freeing the strings and clearing KDE_V_DEFINED.
! We can't actually remove this KDE from the tree, but it can be reused if
! the same key is redefined.
!-

KDE [KDE_V_DEFINED] = 0;        ! Invalidate KDE
IF .KDE [KDE_A_EQUIV_POINTER] NEQA 0
THEN
    LIB$SFREE1_DD6 (KDE [KDE_R_EQUIV_DESC]);
IF .KDE [KDE_A_STATE_POINTER] NEQA 0
THEN
    LIB$SFREE1_DD6 (KDE [KDE_R_STATE_DESC]);
IF .KDE [KDE_A_PREFIX_POINTER] NEQA 0
THEN
    LIB$SFREE1_DD6 (KDE [KDE_R_PREFIX_DESC]);

!+
! If this was a control key, reflect the deletion
! in the KTH.
!-

KTH [KTH_L_TERM_MASK] = .KTH [KTH_L_TERM_MASK] AND
    (NOT .KDE [KDE_L_TERM_MASK]);

!+
! Note that we can't free TKEY since it needs to be compared
! against in future searches.
!-

RETURN SS$_NORMAL;

END;                                    ! End of routine SMG$GET_KEY_DEF
```

```
                          03FC 00000       .ENTRY  SMG$DELETE_KEY_DEF, Save R2,R3,R4,R5,R6,R7,-; 0763
                                                   R8,R9                                       ;
       59 00000000G  00  9E 00002          MOVAB   LIB$SFREE1_DD6, R9                          ;
       5E            30  C2 00009          SUBL2   #48, SP                                     ;
       58        04  BC  D0 0000C          MOVL    @KEY_TABLE_ID, KTH                          ; 0846
                     06  13 00010          BEQL    1$                                          ;
       58        10  A8  D1 00012          CMPL    16(KTH), KTH                                ;
                     08  13 00016          BEQL    2$                                          ;
       50 00000000G  8F  D0 00018 1$:      MOVL    #SMG$_INVKTB_ID, R0                         ;
                         04 0001F          RET                                                 ;
  08   AE        0C  AE  9E 00020 2$:      MOVAB   TKEY, TKEY_DESC+4                           ; 0852
       57        04  AE  9E 00025          MOVAB   TKEY_DESC, TKEY_DESC_PTR                    ; 0853
       56            5C  D0 00029          MOVL    AP, R6                                      ; 0854
0000V  CF            00  FB 0002C          CALLS   #0, CONSTRUCT_TKEY                          ;
       52            50  E9 00031          BLBC    TKEY_STATUS, 8$                             ; 0855
                     5E  DD 00034          PUSHL   SP                                          ; 0864
              0000V  CF  9F 00036          PUSHAB  COMPARE_ROUTINE                             ;
                     57  DD 0003A          PUSHL   TKEY_DESC_PTR                               ; 0865
```

```
                               58  DD 0003C          PUSHL   KTH                              ; 0864
            00000000G  00      04  FB 0003E          CALLS   #4, LIB$LOOKUP_TREE              ;
                       08      50  E9 00045          BLBC    LOOKUP_STATUS, 3$                ; 0868
                       57      6E  D0 00048          MOVL    FOUND_KDE, KDE                   ; 0872
                          33   A7  95 0004B          TSTB    51(KDE)                          ; 0873
                               08  19 0004E          BLSS    4$                               ;
                 50 00000000G  8F  D0 00050 3$:      MOVL    #SMG$_KEYNOTDEF, R0              ; 0875
                                   04 00057          RET                                      ;
                 33    A7   80  8F  8A 00058 4$:     BICB2   #128, 51(KDE)                    ; 0883
                            18  A7  D5 0005D         TSTL    24(KDE)                          ; 0884
                                06  13 00060         BEQL    5$                               ;
                       50   14  A7  9E 00062         MOVAB   20(KDE), R0                      ; 0886
                                69  16 00066         JSB     LIB$SFREE1_DD6                   ;
                            20  A7  D5 00068 5$:     TSTL    32(KDE)                          ; 0887
                                06  13 0006B         BEQL    6$                               ;
                       50   1C  A7  9E 0006D         MOVAB   28(KDE), R0                      ; 0889
                                69  16 00071         JSB     LIB$SFREE1_DD6                   ;
                            28  A7  D5 00073 6$:     TSTL    40(KDE)                          ; 0890
                                06  13 00076         BEQL    7$                               ;
                       50   24  A7  9E 00078         MOVAB   36(KDE), R0                      ; 0892
                                69  16 0007C         JSB     LIB$SFREE1_DD6                   ;
                 0C    A8   34  A7  CA 0007E 7$:     BICL2   52(KDE), 12(KTH)                 ; 0900
                       50       01  D0 00083         MOVL    #1, R0                           ; 0907
                                    04 00086 8$:     RET                                      ; 0909

; Routine Size:  135 bytes,    Routine Base:  _SMG$CODE + 0202
```

```
%SBTTL 'SMG$SET_DEFAULT_STATE'
GLOBAL ROUTINE SMG$SET_DEFAULT_STATE (
    KEY_TABLE_ID: REF VECTOR [, LONG],
    NEW_STATE: REF BLOCK [, BYTE],
    OLD_STATE: REF BLOCK [, BYTE]
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure sets a new default state name and/or returns the
!       default state name for a key definition table.
!
! CALLING SEQUENCE:
!
!       RET_STATUS.wlc.v = SMG$SET_DEFAULT_STATE (
!                               KEY_TABLE_ID.rl.r,
!                               [NEW_STATE.rt.dx]
!                               [, [OLD_STATE.wt.dx]])
!
! FORMAL PARAMETERS:
!
!       KEY_TABLE_ID    A longword containing the identification of the
!                       key-definition table for which the default state
!                       name is to be set or returned.
!
!       NEW_STATE       A string containing the new default state name to be set
!                       for this key-definition table.  If omitted, the
!                       existing default state name is unchanged.
!                       The state name is converted to upper case and is
!                       stripped of trailing blanks before use.
!
!       OLD_STATE       A string into which is written the existing default
!                       state name of the key-definition table.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL      Normal successful completion
!       SMG$_INVKTB_ID  Invalid key-table id
!       SMG$_INVSTANAM  Invalid state name
!       LIB$_INVSTRDES  Invalid string descriptor
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN
```

```
LOCAL
    KTH: REF KTH_R_KTH_STRUCT;              ! Key table header

BUILTIN
    NULLPARAMETER;

!+
! Define positions of arguments
!-

LITERAL
    K_KEY_TABLE_ID = 1,
    K_NEW_STATE= 2,
    K_OLD_STATE = 3;

!+
! Validate KEY_TABLE_ID and get the Key Table Header.
!-

$SMG$VALIDATE_KTH (KEY_TABLE_ID, KTH);

!+
! Return the old default state name, if requested.
!-

IF NOT NULLPARAMETER (K_OLD_STATE)
THEN
    BEGIN
    LOCAL
        COPY_STATUS;
    COPY_STATUS = LIB$SCOPY_R_DX6 ((.KTH [KTH_W_DEF_STATE_LEN] - 3),
        KTH [KTH_T_DEF_STATE_STRING], OLD_STATE [0,0,0,0]);
    IF NOT .COPY_STATUS
    THEN
        RETURN .COPY_STATUS;
    END;

!+
! Set the new default state, if requested.
!-

IF NOT NULLPARAMETER (K_NEW_STATE)
THEN
    BEGIN
    LOCAL
        NEW_STATE_PTR,                      ! Pointer to string
        NEW_STATE_LEN: WORD,                ! Length of string
        CHAR_PTR,                           ! Pointer to current character
        KEY_PTR,                            ! Pointer to current pos in TKEY
        ANL_STATUS;                         ! Status from LIB$ANALYZE_SDESC

    !+
    ! Analyze NEW_STATE string.
    !-

    ANL_STATUS = LIB$ANALYZE_SDESC_R2 (NEW_STATE [0,0,0,0];
```

```
3             NEW_STATE_LEN, NEW_STATE_PTR);
3         IF NOT .ANL_STATUS
3         THEN
3             RETURN .ANL_STATUS;
3
3         IF .NEW_STATE_LEN GTRU 31
3         THEN
3             RETURN SMG$_INVSTANAM;      ! Invalid state name
3
3         !+
3         ! Find position of last non-blank character.
3         !-
3
3         CHAR_PTR = .NEW_STATE_PTR + .NEW_STATE_LEN;
3         WHILE ((CHAR_PTR=.CHAR_PTR-1;CH$RCHAR(.CHAR_PTR)) EQLU %C' ') DO
4             BEGIN
4             NEW_STATE_LEN = .NEW_STATE_LEN - 1;
4             IF .NEW_STATE_LEN EQL 0 THEN EXITLOOP;
3             END;
3
3         KEY_PTR = CH$TRANSLATE (LIB$AB_UPCASE, .NEW_STATE_LEN, .NEW_STATE_PTR,
3             0, .NEW_STATE_LEN, KTH [KTH_T_DEF_STATE_STRING]);
3         CH$WCHAR_A (0, KEY_PTR);
3         KTH [KTH_A_DEF_KEYCODE] = .KEY_PTR;
3         KTH [KTH_W_DEF_STATE_LEN] = .NEW_STATE_LEN + 3;
2         END;
2
2     RETURN SS$_NORMAL;
2
1     END;                                ! End of routine SMG$SET_DEFAULT_STATE
```

```
                         00FC 00000          .ENTRY  SMG$SET_DEFAULT_STATE, Save R2,R3,R4,R5,R6,-; 0911
                                                     R7
      57         04   BC   D0 00002          MOVL    @KEY_TABLE_ID, KTH                          ; 0987
                      06   13 00006          BEQL    1$                                          ;
      57         10   A7   D1 00008          CMPL    16(KTH), KTH                                ;
                      08   13 0000C          BEQL    2$                                          ;
      50  00000000G   8F   D0 0000E  1$:     MOVL    #SMG$_INVKTB_ID, R0                         ;
                           04 00015          RET                                                 ;
      03              6C   91 00016  2$:     CMPB    (AP), #3                                    ; 0993
                      1D   1F 00019          BLSSU   3$                                          ;
                 0C   AC   D5 0001B          TSTL    12(AP)                                      ;
                      18   13 0001E          BEQL    3$                                          ;
      51         20   A7   9E 00020          MOVAB   32(KTH), R1                                 ; 0999
      50         14   A7   3C 00024          MOVZWL  20(KTH), R0                                 ; 0998
      50              03   C2 00028          SUBL2   #3, R0                                      ;
      52         0C   AC   D0 0002B          MOVL    OLD_STATE, R2                               ; 0999
           00000000G  00   16 0002F          JSB     LIB$SCOPY_R_DX6                             ;
      50              50   E9 00035          BLBC    COPY_STATUS, 8$                             ; 1000
      02              6C   91 00038  3$:     CMPB    (AP), #2                                    ; 1009
                      48   1F 0003B          BLSSU   7$                                          ;
                 08   AC   D5 0003D          TSTL    8(AP)                                       ;
                      43   13 00040          BEQL    7$                                          ;
```

```
                              50       08   AC  D0 00042           MOVL    NEW_STATE, R0                              ; 1023
                               00000000G   00  16 00046           JSB     LIB$ANALYZE_SDESC_R2                       ;
                              56            51  D0 0004C           MOVL    R1, NEW_STATE_LEN                          ;
                              36            50  E9 0004F           BLBC    ANL_STATUS, 8$                             ; 1025
                              1F            56  B1 00052           CMPW    NEW_STATE_LEN, #31                         ; 1029
                                            08  1B 00055           BLEQU   4$                                         ;
                              50 00000000G  8F  D0 00057           MOVL    #SMG$_INVSTANAM, R0                        ; 1031
                                                04 0005E           RET                                                ;
                              50            56  3C 0005F 4$:       MOVZWL  NEW_STATE_LEN, CHAR_PTR                    ; 1037
                              50            52  C0 00062           ADDL2   NEW_STATE_PTR, CHAR_PTR                    ;
                              20            70  91 00065 5$:       CMPB    -(CHAR_PTR), #32                           ; 1038
                                            04  12 00068           BNEQ    6$                                         ;
                                            56  B7 0006A           DECW    NEW_STATE_LEN                              ; 1040
                                            F7  12 0006C           BNEQ    5$                                         ; 1041
00000000G  00           00    62            56  2E 0006E 6$:       MOVTC   NEW_STATE_LEN, (NEW_STATE_PTR), #0, -      ; 1045
                         20   A7            56     00077                   LIB$AB_UPCASE, NEW_STATE_LEN, 32(KTH)      ;
                                            85  94 0007A           CLRB    (KEY_PTR)+                                 ; 1046
                         1C   A7            55  D0 0007C           MOVL    KEY_PTR, 28(KTH)                           ; 1047
              14    A7        56            03  A1 00080           ADDW3   #3, NEW_STATE_LEN, 20(KTH)                 ; 1048
                              50            01  D0 00085 7$:       MOVL    #1, R0                                     ; 1051
                                                04 00088 8$:       RET                                                ; 1053

; Routine Size:  137 bytes,    Routine Base:  _SMG$CODE + 0289
```

```
1 %SBTTL 'SMG$$LOOKUP_KEY'
1 GLOBAL ROUTINE SMG$$LOOKUP_KEY (
1     KTH: REF KTH_R_KTH_STRUCT,
1     TKEY_DESC: REF BLOCK [, BYTE];
1     KDE: REF KDE_R_KDE_STRUCT
1     ): SMG$$LOOKUP_KEY$LNK =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure is called by SMG$READ_COMPOSED_LINE to return
1 !       the Key Definition Entry for a given "total key".
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       ret_status.wlc.v = SMG$$LOOKUP_KEY (
1 !                               KTH.rr.r, TKEY_DESC.rr.r, KDE.wr.r)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
2     LOCAL
2         KEYCODE: REF VECTOR [2, BYTE],          ! Key code
2         TEMP: BYTE,                             ! Temporary
2         FOUND_KDE,                              ! KDE found
2         LOOKUP_STATUS;                          ! Status from LIB$LOOKUP_TREE
2
2     !+
2     ! Swap the two bytes of the key code so that a string comparison works.
2     !-
2
2     KEYCODE = .TKEY_DESC [DSC$A_POINTER] + (.TKEY_DESC [DSC$W_LENGTH] - 2);
2     TEMP = .KEYCODE [0];
2     KEYCODE [0] = .KEYCODE [1];
2     KEYCODE [1] = .TEMP;
2
2     !+
```

```
; 1041          1111  2        ! Look up key name.
; 1042          1112  2        !-
; 1043          1113  2
; 1044          1114  2        LOOKUP_STATUS = LIB$LOOKUP_TREE (
; 1045          1115  2            KTH [KTH_A_TREEHEAD],                      ! Address of treehead
; 1046          1116  2            TKEY_DESC [0,0,0,0],             ! Total state-key string
; 1047          1117  2            COMPARE_ROUTINE,                 ! Key comparison routine
; 1048          1118  2            FOUND_KDE);                      ! Pointer to found KDE
; 1049          1119  2        IF NOT .LOOKUP_STATUS
; 1050          1120  2        THEN
; 1051          1121  2            RETURN SMG$_KEYNOTDEF;           ! Key not defined
; 1052          1122  2
; 1053          1123  2        !+
; 1054          1124  2        ! If KDE is not defined, return "not found".
; 1055          1125  2        !-
; 1056          1126  2
; 1057          1127  2        KDE = .FOUND_KDE;
; 1058          1128  2        IF NOT .KDE [KDE_V_DEFINED]
; 1059          1129  2        THEN
; 1060          1130  2            RETURN SMG$_KEYNOTDEF;  ! Key not defined
; 1061          1131  2
; 1062          1132  2        RETURN SS$_NORMAL;
; 1063          1133  2
; 1064          1134  1        END;                                 ! End of routine SMG$$LOOKUP_KEY


                       5E            08  C2 00000 SMG$$LOOKUP_KEY::
                                                              SUBL2   #8, SP                                 ; 1055
                       56            51  D0 00003             MOVL    R1, R6
                       51            66  3C 00006             MOVZWL  (TKEY_DESC), R1                        ; 1105
                       51       04   A6  C0 00009             ADDL2   4(TKEY_DESC), R1
                                     51  D7 0000D             DECL    KEYCODE
                       6E            71  90 0000F             MOVB    -(KEYCODE), TEMP                       ; 1106
                       61       01   A1  90 00012             MOVB    1(KEYCODE), (KEYCODE)                  ; 1107
               01      A1            6E  90 00016             MOVB    TEMP, 1(KEYCODE)                       ; 1108
                                04   AE  9F 0001A             PUSHAB  FOUND_KDE                              ; 1116
                             0000V   CF  9F 0001D             PUSHAB  COMPARE_ROUTINE
                              0041   8F  BB 00021             PUSHR   #^M<R0,R6>
           00000000G   00            04  FB 00025             CALLS   #4, LIB$LOOKUP_TREE
                       09            50  E9 0002C             BLBC    LOOKUP_STATUS, 1$                      ; 1119
                       56       04   AE  D0 0002F             MOVL    FOUND_KDE, KDE                         ; 1127
                                33   A6  95 00033             TSTB    51(KDE)                                ; 1128
                                     09  19 00036             BLSS    2$
                       50 00000000G  8F  D0 00038 1$:         MOVL    #SMG$_KEYNOTDEF, R0                    ; 1130
                                     03  11 0003F             BRB     3$
                       50            01  D0 00041 2$:         MOVL    #1, R0                                 ; 1132
                       5E            08  C0 00044 3$:         ADDL2   #8, SP                                 ; 1134
                                         05 00047             RSB

; Routine Size:  72 bytes,    Routine Base:  _SMG$CODE + 0312
```

```
%SBTTL 'CONSTRUCT_TKEY'
ROUTINE CONSTRUCT_TKEY (
    CALLERS_AP: REF VECTOR [, LONG],
    TKEY_DESC: REF BLOCK [, BYTE];
    OUT_TKEY_DESC: REF BLOCK [, BYTE]
    ) : CONSTRUCT_TKEY$LNK =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure builds a "total key" from the IF_STATE and the
!       KEY_NAME in the caller's argument list.
!
! CALLING SEQUENCE:
!
!       CONSTRUCT_TKEY (CALLERS_AP.rl.v, TKEY_DESC.rr.r;
!               OUT_TKEY_DESC.wa.v)
!
! FORMAL PARAMETERS:
!
!       CALLERS_AP      - The caller's argument pointer.
!
!       TKEY_DESC       - A descriptor for the TKEY.
!
!       OUT_TKEY_DESC   - A longword into which is written the address
!                         of the TKEY descriptor actually used.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL      Normal successful completion
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        KEY_CODE,                               ! Key code number
        CODE_STATUS,                            ! Status from conversion
        NAME_TMP: VECTOR [32, BYTE],            ! Upcased/stripped key name
        KTH: REF KTH_R_KTH_STRUCT,              ! Key table header
        TKEY_PTR: REF VECTOR [, BYTE];          ! Pointer to current pos in TKEY

    !+
    ! Define positions of arguments
    !-
```

```
    LITERAL
        K_KEY_TABLE_ID = 1,
        K_KEY_NAME = 2,
        K_IF_STATE = 3;

    !+
    ! Upcase key name and strip blanks.  If key name has invalid length,
    ! return an error.
    !-

        BEGIN
        LOCAL
            KEY_NAME_PTR,          ! Pointer to KEY_NAME string
            KEY_NAME_LEN: WORD,    ! Length of key name
            CHAR_PTR,              ! Pointer to current character in KEY_NAME
            ANL_STATUS;            ! Status from LIB$ANALYZE_SDESC_R2

        ANL_STATUS = LIB$ANALYZE_SDESC_R2 (.CALLERS_AP [K_KEY_NAME];
            KEY_NAME_LEN, KEY_NAME_PTR);
        IF NOT .ANL_STATUS
        THEN
            RETURN .ANL_STATUS;
        IF (.KEY_NAME_LEN - 1) GTRU 30  ! Between 1 and 31?
        THEN
            RETURN SMG$_INVKEYNAM;              ! Invalid key name

        !+
        ! Find the position of the last non-blank character.
        !-

        CHAR_PTR = .KEY_NAME_PTR + .KEY_NAME_LEN;
        WHILE ((CHAR_PTR=.CHAR_PTR-1;CH$RCHAR(.CHAR_PTR)) EQLU %C' ') DO
            BEGIN
            KEY_NAME_LEN = .KEY_NAME_LEN - 1;
            IF .KEY_NAME_LEN EQL 0 THEN EXITLOOP;
            END;

        !+
        ! Move an upcased counted string to NAME_TMP;
        !-

        NAME_TMP [0] = .KEY_NAME_LEN;
        IF .NAME_TMP [0] EQL 0
        THEN
            RETURN SMG$_INVKEYNAM;       ! Invalid key name
        CH$TRANSLATE (LIB$AB_UPCASE, .KEY_NAME_LEN, .KEY_NAME_PTR,
            0, .KEY_NAME_LEN, NAME_TMP [1]);
        END;

    !+
    ! Get the code number corresponding to this key name, if any.
    !-

    CODE_STATUS = SMG$$NAME_TO_KEYCODE (NAME_TMP, KEY_CODE);
    IF NOT .CODE_STATUS
    THEN
```

```
	RETURN .CODE_STATUS;

    !+
    ! Construct TKEY by appending IF_STATE, a <NUL>, and KEY_CODE.
    ! Swap the bytes of KEY_CODE so that they compare as strings.
    ! Upcase and blank trim IF_STATE while we're at it.
    !-

    KTH = ..CALLERS_AP [K_KEY_TABLE_ID];
    OUT_TKEY_DESC = KTH [KTH_R_DEF_STATE_DESCR]; ! Assume default state
    TKEY_PTR = .KTH [KTH_A_DEF_KEYCODE];
    IF (.CALLERS_AP[0])<0,8> GEQU K_IF_STATE
    THEN
	IF .CALLERS_AP [K_IF_STATE] NEQA 0
	THEN
	    BEGIN
	    LOCAL
		IF_STATE_PTR,		! Pointer to IF_STATE string
		IF_STATE_LEN: WORD,	! Length of IF_STATE string
		CHAR_PTR,		! Pointer to current character
		ANL_STATUS;		! Status from LIB$ANALYZE_SDESC_R2

	    ANL_STATUS = LIB$ANALYZE_SDESC_R2 (.CALLERS_AP [K_IF_STATE];
		IF_STATE_LEN, IF_STATE_PTR);
	    IF NOT .ANL_STATUS
	    THEN
		RETURN .ANL_STATUS;
	    IF (.IF_STATE_LEN - 1) GTRU 30	! Between 1 and 31?
	    THEN
		RETURN SMG$_INVSTANAM;		! Invalid state name

	    !+
	    ! Find the position of the last non-blank character.
	    !-

	    CHAR_PTR = .IF_STATE_PTR+ .IF_STATE_LEN;
	    WHILE ((CHAR_PTR=.CHAR_PTR-1;CH$RCHAR(.CHAR_PTR)) EQLU %C' ') DO
		BEGIN
		IF_STATE_LEN = .IF_STATE_LEN - 1;
		IF .IF_STATE_LEN EQL 0 THEN EXITLOOP;
		END;

	    TKEY_DESC [DSC$W_LENGTH] = .IF_STATE_LEN + 3;
	    TKEY_PTR = CH$TRANSLATE (LIB$AB_UPCASE, .IF_STATE_LEN, .IF_STATE_PTR,
		0, .IF_STATE_LEN, .TKEY_DESC [DSC$A_POINTER]);
	    CH$WCHAR_A (0, TKEY_PTR);

	    OUT_TKEY_DESC = .TKEY_DESC;
	    OUT_TKEY_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
	    OUT_TKEY_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
	    END;

    !+
    ! Move the key code, swapping the bytes.
    !-

    TKEY_PTR [0]  = .KEY_CODE<8,8>;
```

```
; 1237      1306 2     TKEY_PTR [1] = .KEY_CODE<0,8>;
; 1238      1307 2
; 1239      1308 2     RETURN SS$_NORMAL;
; 1240      1309 2
; 1241      1310 1     END;                                         ! End of routine CONSTRUCT_TKEY


                                        0F3C 00000 CONSTRUCT_TKEY:
                                                          .WORD   Save R2,R3,R4,R5,R8,R9,R10,R11                  ; 1136
                          5B 00000000G  00   9E 00002     MOVAB   LIB$ANALYZE_SDESC_R2, R11                       ;
                          5A 00000000G  00   9E 00009     MOVAB   LIB$AB_UPCASE, R10                              ;
                          5E            24   C2 00010     SUBL2   #36, SP                                         ;
                          59            57   D0 00013     MOVL    R7, R9                                          ;
                          50        08  A6   D0 00016     MOVL    8(CALLERS_AP), R0                               ; 1210
                                        6B   16 0001A     JSB     LIB$ANALYZE_SDESC_R2                            ;
                          6F            50   E9 0001C     BLBC    ANL_STATUS, 5$                                  ; 1212
                          50            51   3C 0001F     MOVZWL  KEY_NAME_LEN, R0                                ; 1215
                                        50   D7 00022     DECL    R0                                              ;
                          1E            50   D1 00024     CMPL    R0, #30                                         ;
                                        15   1A 00027     BGTRU   3$                                              ;
                          50            51   3C 00029     MOVZWL  KEY_NAME_LEN, CHAR_PTR                          ; 1223
                          50            52   C0 0002C     ADDL2   KEY_NAME_PTR, CHAR_PTR                          ;
                          20            70   91 0002F 1$: CMPB    -(CHAR_PTR), #32                                ; 1224
                                        04   12 00032     BNEQ    2$                                              ;
                                        51   B7 00034     DECW    KEY_NAME_LEN                                    ; 1226
                                        F7   12 00036     BNEQ    1$                                              ; 1227
                     04   AE            51   90 00038 2$: MOVB    KEY_NAME_LEN, NAME_TMP                          ; 1234
                                        09   12 0003C     BNEQ    4$                                              ; 1235
                          50 00000000G  8F   D0 0003E 3$: MOVL    #SMG$_INVKEYNAM, R0                             ; 1237
                                        7A   11 00045     BRB     10$                                             ;
    6A            00      62            51   2E 00047 4$: MOVTC   KEY_NAME_LEN, (KEY_NAME_PTR), #0, -             ; 1239
                     05   AE            51      0004C             LIB$AB_UPCASE, KEY_NAME_LEN, NAME_TMP+1         ;
                                        5E   DD 0004F     PUSHL   SP                                              ; 1246
                                08      AE   9F 00051     PUSHAB  NAME_TMP                                        ;
                 00000000G      00      02   FB 00054     CALLS   #2, SMG$$NAME_TO_KEYCODE                        ;
                          63            50   E9 0005B     BLBC    CODE_STATUS, 10$                                ; 1247
                          57        04  B6   D0 0005E     MOVL    @4(CALLERS_AP), KTH                             ; 1257
                          58        14  A7   9E 00062     MOVAB   20(R7), OUT_TKEY_DESC                           ; 1258
                          55        1C  A7   D0 00066     MOVL    28(KTH), TKEY_PTR                               ; 1259
                          03            66   D1 0006A     CMPL    (CALLERS_AP), #3                                ; 1260
                                        47   1F 0006D     BLSSU   9$                                              ;
                                    0C  A6   D5 0006F     TSTL    12(CALLERS_AP)                                  ; 1262
                                        42   13 00072     BEQL    9$                                              ;
                          50        0C  A6   D0 00074     MOVL    12(CALLERS_AP), R0                              ; 1271
                                        6B   16 00078     JSB     LIB$ANALYZE_SDESC_R2                            ;
                          44            50   E9 0007A     BLBC    ANL_STATUS, 10$                                 ; 1273
                          50            51   3C 0007D     MOVZWL  IF_STATE_LEN, R0                                ; 1276
                                        50   D7 00080     DECL    R0                                              ;
                          1E            50   D1 00082     CMPL    R0, #30                                         ;
                                        09   1B 00085     BLEQU   6$                                              ;
                          50 00000000G  8F   D0 00087     MOVL    #SMG$_INVSTANAM, R0                             ; 1278
                                        31   11 0008E 5$: BRB     10$                                             ;
                          50            51   3C 00090 6$: MOVZWL  IF_STATE_LEN, CHAR_PTR                          ; 1284
                          50            52   C0 00093     ADDL2   IF_STATE_PTR, CHAR_PTR                          ;
```

```
                     20        70 91 00096 7$:   CMPB   -(CHAR_PTR), #32                                 ; 1285
                               04 12 00099       BNEQ   8$                                               ;
                               51 B7 0009B       DECW   IF_STATE_LEN                                     ; 1287
                               F7 12 0009D       BNEQ   7$                                               ; 1288
              69         51    03 A1 0009F 8$:   ADDW3  #3, IF_STATE_LEN, (TKEY_DESC)                    ; 1291
     6A       00         62    51 2E 000A3       MOVTC  IF_STATE_LEN, (IF_STATE_PTR), #0, -              ; 1293
                    04   B9    51    000A8              LIB$AB_UPCASE, IF_STATE_LEN, @4(TKEY_DESC)       ;
                               85 94 000AB       CLRB   (TKEY_PTR)+                                      ; 1294
                         58    59 D0 000AD       MOVL   TKEY_DESC, OUT_TKEY_DESC                         ; 1296
                    02   A8  010E 8F B0 000B0    MOVW   #270, 2(OUT_TKEY_DESC)                           ; 1297
                         65    01 AE 90 000B6 9$: MOVB  KEY_CODE+1, (TKEY_PTR)                           ; 1305
                    01   A5    6E 90 000BA       MOVB   KEY_CODE, 1(TKEY_PTR)                            ; 1306
                         50    01 D0 000BE       MOVL   #1, R0                                           ; 1308
                         57    58 D0 000C1 10$:  MOVL   R8, R7                                           ; 1310
                               04 000C4          RET                                                     ;
```

; Routine Size: 197 bytes,   Routine Base: _SMG$CODE + 035A

```
; 1243    1311  1 %SBTTL 'COMPARE_ROUTINE'
; 1244    1312  1 ROUTINE COMPARE_ROUTINE (
; 1245    1313  1     KEY_DESC: REF BLOCK [, BYTE],              ! Key being compared
; 1246    1314  1     KDE: REF KDE_R_KDE_STRUCT,                 ! Key definition being compared
; 1247    1315  1     CONTEXT                                    ! Not used
; 1248    1316  1     ) =
; 1249    1317  1
; 1250    1318  1 !++
; 1251    1319  1 ! FUNCTIONAL DESCRIPTION:
; 1252    1320  1 !
; 1253    1321  1 !       Routine called by LIB$INSERT_TREE and LIB$LOOKUP_TREE to compare
; 1254    1322  1 !       keys.
; 1255    1323  1 !
; 1256    1324  1 ! CALLING SEQUENCE:
; 1257    1325  1 !
; 1258    1326  1 !       Not directly called.
; 1259    1327  1 !
; 1260    1328  1 ! FORMAL PARAMETERS:
; 1261    1329  1 !
; 1262    1330  1 !
; 1263    1331  1 ! IMPLICIT INPUTS:
; 1264    1332  1 !
; 1265    1333  1 !       NONE
; 1266    1334  1 !
; 1267    1335  1 ! IMPLICIT OUTPUTS:
; 1268    1336  1 !
; 1269    1337  1 !       NONE
; 1270    1338  1 !
; 1271    1339  1 ! COMPLETION STATUS:
; 1272    1340  1 !
; 1273    1341  1 !       SS$_NORMAL      Normal successful completion
; 1274    1342  1 !
; 1275    1343  1 ! SIDE EFFECTS:
; 1276    1344  1 !
; 1277    1345  1 !       NONE
; 1278    1346  1 !
; 1279    1347  1 !--
; 1280    1348  1
; 1281    1349  2     BEGIN
; 1282    1350  2
; 1283    1351  2     RETURN CH$COMPARE (
; 1284    1352  2         .KEY_DESC [DSC$W_LENGTH],
; 1285    1353  2         .KEY_DESC [DSC$A_POINTER],
; 1286    1354  2         .KDE [KDE_W_TKEY_LENGTH],
; 1287    1355  2         .KDE [KDE_A_TKEY_POINTER]);
; 1288    1356  2
; 1289    1357  1     END;                                       ! End of routine COMPARE_ROUTINE


                                001C 00000 COMPARE_ROUTINE:
                                                .WORD   Save R2,R3,R4          ; 1312
              51      04   AC  D0 00002         MOVL    KEY_DESC, R1           ; 1352
              50      08   AC  D0 00006         MOVL    KDE, R0                ; 1354
              54           01  D0 0000A         MOVL    #1, R4                 ; 1351
```

```
     0C   A0          00      04   B1          61  2D 0000D         CMPC5   (R1), @4(R1), #0, 12(R0), @16(R0)      ;
                                          10   B0       00014                                                        ;
                                               03  1A 00016         BGTRU   1$                                       ;
                                   54          01  D9 00018         SBWC    #1, R4                                   ;
                                   50          54  D0 0001B 1$:     MOVL    R4, R0                                   ;
                                                   04 0001E         RET                                              ; 1357

; Routine Size:  31 bytes,    Routine Base:  _SMG$CODE + 041F
```

```
%SBTTL 'ALLOC_ROUTINE'
ROUTINE ALLOC_ROUTINE (
    TKEY_DESC: REF BLOCK [, BYTE],              ! Key descriptor
    KDE_POINTER: REF VECTOR [, LONG],           ! Pointer to new KDE
    CONTEXT                                     ! Not used
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Routine called by LIB$INSERT_TREE to allocate a node.
!
! CALLING SEQUENCE:
!
!       Not directly called.
!
! FORMAL PARAMETERS:
!
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL      Normal successful completion
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        VM_STATUS,                      ! Status from LIB$GET_VM
        KDE: REF KDE_R_KDE_STRUCT,      ! Key Definition Entry
        KEY_CODE: WORD;

    !+
    ! Allocate the memory.
    !-

    VM_STATUS = LIB$GET_VM (%REF(KDE_S_KDE_STRUCT), KDE_POINTER [0]);
    IF NOT .VM_STATUS
    THEN
        RETURN .VM_STATUS;

    !+
    ! Initialize the KDE.
    !-

```

```
; 1348      1415  2      KDE = .KDE_POINTER [0];
; 1349      1416  2      CH$FILL (0, KDE_S_KDE_STRUCT, KDE [0,0,0,0]);
; 1350      1417  2      KDE [KDE_B_TKEY_CLASS] = DSC$K_CLASS_D;
; 1351      1418  2      KDE [KDE_B_TKEY_DTYPE] = DSC$K_DTYPE_T;
; 1352      1419  2      KDE [KDE_B_EQUIV_CLASS] = DSC$K_CLASS_D;
; 1353      1420  2      KDE [KDE_B_EQUIV_DTYPE] = DSC$K_DTYPE_T;
; 1354      1421  2      KDE [KDE_B_STATE_CLASS] = DSC$K_CLASS_D;
; 1355      1422  2      KDE [KDE_B_STATE_DTYPE] = DSC$K_DTYPE_T;
; 1356      1423  2      KDE [KDE_B_PREFIX_CLASS] = DSC$K_CLASS_D;
; 1357      1424  2      KDE [KDE_B_PREFIX_DTYPE] = DSC$K_DTYPE_T;
; 1358      1425  2
; 1359      1426  2      !+
; 1360      1427  2      ! If the key being defined is CTRLx, set the terminator mask to include
; 1361      1428  2      ! this control key.
; 1362      1429  2      !-
; 1363      1430  2
; 1364      1431  2
; 1365      1432  4      KEY_CODE = .((.TKEY_DESC [DSC$A_POINTER] +
; 1366      1433  2          .TKEY_DESC [DSC$W_LENGTH]) - 2)<0,16>;
; 1367      1434  2      IF .KEY_CODE NEQ 0 AND
; 1368      1435  2          .KEY_CODE LEQU SMG$K_TRM_CTRLZ
; 1369      1436  2      THEN
; 1370      1437  2          KDE [KDE_L_TERM_MASK] = 1^.KEY_CODE;
; 1371      1438  2
; 1372      1439  2      RETURN SS$_NORMAL;
; 1373      1440  2
; 1374      1441  1      END;                                        ! End of routine ALLOC_ROUTINE



                                          007C 00000 ALLOC_ROUTINE:
                                                         .WORD    Save R2,R3,R4,R5,R6                  ; 1359
                5E                  04 C2 00002          SUBL2    #4, SP
                          08        AC DD 00005          PUSHL    KDE_POINTER                          ; 1406
            04  AE                  38 D0 00008          MOVL     #56, 4(SP)
                          04        AE 9F 0000C          PUSHAB   4(SP)
       00000000G  00                02 FB 0000F          CALLS    #2, LIB$GET_VM                       ;
                  41                50 E9 00016          BLBC     VM_STATUS, 2$                        ; 1407
                  56        08      BC D0 00019          MOVL     @KDE_POINTER, KDE                    ; 1415
  38        00    6E                00 2C 0001D          MOVC5    #0, (SP), #0, #56, (KDE)             ; 1416
                                    66    00022
            0E    A6      020E      8F B0 00023          MOVW     #526, 14(KDE)                        ; 1418
            16    A6      020E      8F B0 00029          MOVW     #526, 22(KDE)                        ; 1420
            1E    A6      020E      8F B0 0002F          MOVW     #526, 30(KDE)                        ; 1422
            26    A6      020E      8F B0 00035          MOVW     #526, 38(KDE)                        ; 1424
                  50        04      AC D0 0003B          MOVL     TKEY_DESC, R0                        ; 1432
                  52                60 3C 0003F          MOVZWL   (R0), R2                             ; 1433
        50        52        04      A0 C1 00042          ADDL3    4(R0), R2, R0
                  51        FE      A0 B0 00047          MOVW     -2(R0), KEY_CODE                     ; 1432
                                    0A 13 0004B          BEQL     1$                                   ; 1434
                  1A                51 B1 0004D          CMPW     KEY_CODE, #26                        ; 1435
                                    05 1A 00050          BGTRU    1$                                   ;
      34    A6    01                51 78 00052          ASHL     KEY_CODE, #1, 52(KDE)                ; 1437
                  50                01 D0 00057 1$:      MOVL     #1, R0                               ; 1439
                                       04 0005A 2$:      RET                                           ; 1441
```

```
; Routine Size:  91 bytes,    Routine Base:  _SMG$CODE + 043E
```

```
1 %SBTTL 'SMG$DEFINE_KEY'
1 GLOBAL ROUTINE SMG$DEFINE_KEY (
1     KEY_TABLE_ID: REF VECTOR [, LONG],
1     COMMAND_LINE: REF BLOCK [, BYTE]
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure parses and performs a DEFINE/KEY command.  It can
1 !       be used by utilities which accept DEFINE/KEY commands but which
1 !       do not wish to parse the commands themselves.
1 !
1 !       SMG$DEFINE_KEY calls CLI$DCL_PARSE to parse the command line and
1 !       then makes the appropriate call to SMG$ADD_KEY_DEF.  Use of this
1 !       procedure requires that the image be run under the DCL command
1 !       language interpreter.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       RET_STATUS.wlc.v = SMG$DEFINE_KEY (
1 !                               KEY_TABLE_ID.rl.r,
1 !                               COMMAND_LINE.rt.dx)
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       KEY_TABLE_ID    A longword containing the identification of the
1 !                       key-definition table for which the DEFINE/KEY
1 !                       command is to be performed.
1 !
1 !       COMMAND_LINE    A string containing the entire DEFINE/KEY command
1 !                       to be performed.  Obtain this by calling
1 !                       CLI$GET_VALUE with the '$LINE' keyword.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL      Normal successful completion
1 !       LIB$_xxx        Any error status from LIB$SCOPY_DXDX
1 !       CLI$_xxx        Any error status returned from CLI$ routines
1 !       SMG$_xxx        Any error status returned by SMG$ADD_KEY_DEF
1 !
1 ! SIDE EFFECTS:
1 !
1 !       NONE
1 !
1 !--
1
2     BEGIN
2
```

```
LOCAL
    DESC_LIST: BLOCKVECTOR [5, 8, BYTE],
                                        ! List of descriptors for
                                        ! items returned from CLI
    ATTRIBUTES,                         ! Attributes bit mask
    SET_STATE_PTR,                      ! Pointer to SET_STATE desc, if any
    IF_STATE_PTR,                       ! Pointer to IF_STATE desc, if any
    DESC_LIST_PTR: VOLATILE,            ! Pointer to DESC_LIST
    IF_STATE_PRESENT;                   ! 1 if IF_STATE was specified

BUILTIN
    NULLPARAMETER;

!+
! Define codes that index the descriptors.
!-

LITERAL
    K_KEY_NAME = 0,             ! Key name (P1)
    K_EQUIV_STRING = 1,         ! Equivalence string (P2)
    K_IF_STATE = 2,             ! /IF_STATE
    K_SET_STATE = 3,            ! /SET_STATE
    K_ITEM_NAME = 4;            ! Name of command line item

!+
! Enable the error handler.
!-

ENABLE DEFINE_KEY_HANDLER (DESC_LIST_PTR);

!+
! Initialize the descriptors.
!-

    BEGIN
    LOCAL
        PTR1: REF VECTOR [, LONG],
        PTR2: REF VECTOR [, LONG];

    DESC_LIST [K_KEY_NAME, DSC$W_LENGTH] = 0;
    DESC_LIST [K_KEY_NAME, DSC$B_DTYPE] = DSC$K_DTYPE_T;
    DESC_LIST [K_KEY_NAME, DSC$B_CLASS] = DSC$K_CLASS_D;
    DESC_LIST [K_KEY_NAME, DSC$A_POINTER] = 0;

    PTR1 = DESC_LIST [K_KEY_NAME, 0,0,0,0];
    PTR2 = DESC_LIST [K_EQUIV_STRING, 0,0,0,0];
    $LIB$COPY_QUAD_A (PTR1, PTR2);    ! Initialize EQUIV_STRING desc
    $LIB$COPY_QUAD_A (PTR1, PTR2);    ! Initialize IF_STATE desc
    $LIB$COPY_QUAD_A (PTR1, PTR2);    ! Initialize SET_STATE desc
    $LIB$COPY_QUAD_A (PTR1, PTR2);    ! Initialize ITEM_NAME desc
    DESC_LIST [K_ITEM_NAME, DSC$B_CLASS] = DSC$K_CLASS_S;
    END;

!+
! Set the enable argument pointer.
!-
```

```
DESC_LIST_PTR = DESC_LIST;

!+
! Parse the command line
!-

    BEGIN
    LOCAL
        CLI_STATUS;
    CLI_STATUS = CLI$DCL_PARSE (COMMAND_LINE [0,0,0,0],
        SMG$$AB_DEFKEY_C[D);
    IF NOT .CLI_STATUS
    THEN
        RETURN LIB$SIGNAL (.CLI_STATUS);
    END;

!+
! Get the key name.
!-

    BEGIN
    LOCAL
        STATUS;
    DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('P1');
    DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('P1');

    STATUS = CLI$GET_VALUE (DESC_LIST [K_ITEM_NAME, 0,0,0,0],
        DESC_LIST [K_KEY_NAME, 0,0,0,0]);
    IF NOT .STATUS
    THEN
        RETURN LIB$SIGNAL (.STATUS);
    END;

!+
! Get the equivalence string.
!-

    BEGIN
    LOCAL
        STATUS;
    DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('P2');
    DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('P2');

    STATUS = CLI$GET_VALUE (DESC_LIST [K_ITEM_NAME, 0,0,0,0],
        DESC_LIST [K_EQUIV_STRING, 0,0,0,0]);
    IF NOT .STATUS
    THEN
        RETURN LIB$SIGNAL (.STATUS);
    END;

!+
! Get the new state name, if any.
!-

DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('SET_STATE');
DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('SET_STATE');
SET_STATE_PTR = 0;
```

```
	IF CLI$PRESENT (DESC_LIST [K_ITEM_NAME, 0,0,0,0])
	THEN
	    BEGIN
	    LOCAL
		STATUS;
	    STATUS = CLI$GET_VALUE (DESC_LIST [K_ITEM_NAME, 0,0,0,0],
		DESC_LIST [K_SET_STATE, 0,0,0,0]);
	    IF NOT .STATUS
	    THEN
		RETURN LIB$SIGNAL (.STATUS);
	    SET_STATE_PTR = DESC_LIST [K_SET_STATE, 0,0,0,0];
	    END;

	!+
	! Get the attributes.
	!-

	ATTRIBUTES = 0;
	DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('ECHO');
	DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('ECHO');
	IF NOT CLI$PRESENT (DESC_LIST [K_ITEM_NAME, 0,0,0,0])
	THEN
	    ATTRIBUTES = .ATTRIBUTES OR SMG$M_KEY_NOECHO;

	DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('TERMINATE');
	DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('TERMINATE');
	IF CLI$PRESENT (DESC_LIST [K_ITEM_NAME, 0,0,0,0])
	THEN
	    ATTRIBUTES = .ATTRIBUTES OR SMG$M_KEY_TERMINATE;

	DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('LOCK_STATE');
	DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('LOCK_STATE');
	IF CLI$PRESENT (DESC_LIST [K_ITEM_NAME, 0,0,0,0])
	THEN
	    ATTRIBUTES = .ATTRIBUTES OR SMG$M_KEY_LOCK;

	!+
	! See if IF_STATE was specified, and set IF_STATE_PRESENT accordingly.
	!-

	DESC_LIST [K_ITEM_NAME, DSC$W_LENGTH] = %CHARCOUNT ('IF_STATE');
	DESC_LIST [K_ITEM_NAME, DSC$A_POINTER] = UPLIT BYTE ('IF_STATE');

	IF_STATE_PTR = 0;
	IF_STATE_PRESENT = CLI$PRESENT (DESC_LIST [K_ITEM_NAME, 0,0,0,0]);
	IF .IF_STATE_PRESENT
	THEN
	    IF_STATE_PTR = DESC_LIST [K_IF_STATE, 0,0,0,0];

	!+
	! Loop adding key definitions until all IF_STATE values are used, or
	! if no IF_STATE, just once.
	!-

	WHILE 1 DO
	    BEGIN
```

```
        !+
        ! If IF_STATE was on the command line, get the next value.
        !-

        IF .IF_STATE_PRESENT
        THEN
            BEGIN
            LOCAL
                STATUS;
            STATUS = CLI$GET_VALUE (DESC_LIST [K_ITEM_NAME, 0,0,0,0],
                DESC_LIST [K_IF_STATE, 0,0,0,0]);
            IF NOT .STATUS          ! Exit if no more values
            THEN
                EXITLOOP;
            END;

        !+
        ! Add the key definition.
        !-

            BEGIN
            LOCAL
                ADD_STATUS;
            ADD_STATUS = SMG$ADD_KEY_DEF (
                KEY_TABLE_ID [0],
                DESC_LIST [K_KEY_NAME, 0,0,0,0],
                .IF_STATE_PTR,
                ATTRIBUTES,
                DESC_LIST [K_EQUIV_STRING, 0,0,0,0],
                .SET_STATE_PTR);
            IF NOT .ADD_STATUS
            THEN
                RETURN LIB$SIGNAL (.ADD_STATUS);
            END;

        !+
        ! If IF_STATE wasn't on the command line, exit.
        !-

        IF NOT .IF_STATE_PRESENT
        THEN
            EXITLOOP;
        END;

    !+
    ! Free our dynamic strings.
    !-

    RETURN LIB$SFREEN_DD (%REF(4), DESC_LIST);

    END;                                    ! End of routine SMG$DEFINE_KEY


                      31  50  00499 P.AAB:   .ASCII  \P1\                         ;
                      32  50  0049B P.AAC:   .ASCII  \P2\                         ;
```

```
         45 54 41 54 53 5F 54 45 53 0049D P.AAD:  .ASCII  \SET_STATE\                                     ;
                           4F 48 43 45 004A6 P.AAE:  .ASCII  \ECHO\                                          ;
         45 54 41 4E 49 4D 52 45 54 004AA P.AAF:  .ASCII  \TERMINATE\                                     ;
      45 54 41 54 53 5F 4B 43 4F 4C 004B3 P.AAG:  .ASCII  \LOCK_STATE\                                    ;
            45 54 41 54 53 5F 46 49 004BD P.AAH:  .ASCII  \IF_STATE\                                      ;


                                01FC 00000          .ENTRY  SMG$DEFINE_KEY, Save R2,R3,R4,R5,R6,R7,R8      ; 1443
                58 00000000G  00  9E 00002          MOVAB   CLI$GET_VALUE, R8                               ;
                57        C8  AF  9E 00009          MOVAB   P.AAB, R7                                       ;
                56 00000000G  00  9E 0000D          MOVAB   CLI$PRESENT, R6                                 ;
                5E            34  C2 00014          SUBL2   #52, SP                                         ;
                          08  AE  D4 00017          CLRL    DESC_LIST_PTR                                   ; 1497
                6D      0135  CF  DE 0001A          MOVAL   13$, (FP)                                       ;
         0C     AE  020E0000  8F  D0 0001F          MOVL    #34471936, DESC_LIST                            ; 1538
                          10  AE  D4 00027          CLRL    DESC_LIST+4                                     ; 1541
                51        0C  AE  9E 0002A          MOVAB   DESC_LIST, PTR1                                 ; 1543
                50        14  AE  9E 0002E          MOVAB   DESC_LIST+8, PTR2                               ; 1544
                80            81  7D 00032          MOVQ    (PTR1)+, (PTR2)+                                ; 1545
                80            81  7D 00035          MOVQ    (PTR1)+, (PTR2)+                                ; 1546
                80            81  7D 00038          MOVQ    (PTR1)+, (PTR2)+                                ; 1547
                80            81  7D 0003B          MOVQ    (PTR1)+, (PTR2)+                                ; 1548
         2F     AE            01  90 0003E          MOVB    #1, DESC_LIST+35                                ; 1549
         08     AE        0C  AE  9E 00042          MOVAB   DESC_LIST, DESC_LIST_PTR                        ; 1556
                   00000000G  00  9F 00047          PUSHAB  SMG$$AB_DEFKEY_CLD                              ; 1565
                          08  AC  DD 0004D          PUSHL   COMMAND_LINE                                    ;
   00000000G    00            02  FB 00050          CALLS   #2, CLI$DCL_PARSE                               ;
                46            50  E9 00057          BLBC    CLI_STATUS, 1$                                  ; 1567
         2C     AE            02  B0 0005A          MOVW    #2, DESC_LIST+32                                ; 1579
         30     AE            67  9E 0005E          MOVAB   P.AAB, DESC_LIST+36                             ; 1580
                          0C  AE  9F 00062          PUSHAB  DESC_LIST                                       ; 1583
                          30  AE  9F 00065          PUSHAB  DESC_LIST+32                                    ; 1582
                68            02  FB 00068          CALLS   #2, CLI$GET_VALUE                               ; 1583
                32            50  E9 0006B          BLBC    STATUS, 1$                                      ; 1584
         2C     AE            02  B0 0006E          MOVW    #2, DESC_LIST+32                                ; 1596
         30     AE        02  A7  9E 00072          MOVAB   P.AAC, DESC_LIST+36                             ; 1597
                          14  AE  9F 00077          PUSHAB  DESC_LIST+8                                     ; 1600
                          30  AE  9F 0007A          PUSHAB  DESC_LIST+32                                    ; 1599
                68            02  FB 0007D          CALLS   #2, CLI$GET_VALUE                               ; 1600
                1D            50  E9 00080          BLBC    STATUS, 1$                                      ; 1601
         2C     AE            09  B0 00083          MOVW    #9, DESC_LIST+32                                ; 1610
         30     AE        04  A7  9E 00087          MOVAB   P.AAD, DESC_LIST+36                             ; 1611
                              55  D4 0008C          CLRL    SET_STATE_PTR                                   ; 1612
                          2C  AE  9F 0008E          PUSHAB  DESC_LIST+32                                    ; 1613
                66            01  FB 00091          CALLS   #1, CLI$PRESENT                                 ;
                13            50  E9 00094          BLBC    R0, 3$                                          ;
                          24  AE  9F 00097          PUSHAB  DESC_LIST+24                                    ; 1619
                          30  AE  9F 0009A          PUSHAB  DESC_LIST+32                                    ; 1618
                68            02  FB 0009D          CALLS   #2, CLI$GET_VALUE                               ; 1619
                03            50  E8 000A0 1$:      BLBS    STATUS, 2$                                      ; 1620
                        008E  31     000A3          BRW     10$                                             ;
                55        24  AE  9E 000A6 2$:      MOVAB   DESC_LIST+24, SET_STATE_PTR                     ; 1623
                          04  AE  D4 000AA 3$:      CLRL    ATTRIBUTES                                      ; 1630
         2C     AE            04  B0 000AD          MOVW    #4, DESC_LIST+32                                ; 1631
         30     AE        0D  A7  9E 000B1          MOVAB   P.AAE, DESC_LIST+36                             ; 1632
                          2C  AE  9F 000B6          PUSHAB  DESC_LIST+32                                    ; 1633
```

```
                 66          01  FB 000B9          CALLS    #1, CLI$PRESENT                      ;
                 04          50  E8 000BC          BLBS     R0, 4$                               ;
          04     AE          01  88 000BF          BISB2    #1, ATTRIBUTES                       ; 1635
          2C     AE          09  B0 000C3 4$:      MOVW     #9, DESC_LIST+32                     ; 1637
          30     AE    11    A7  9E 000C7          MOVAB    P.AAF, DESC_LIST+36                  ; 1638
                       2C    AE  9F 000CC          PUSHAB   DESC_LIST+32                         ; 1639
                 66          01  FB 000CF          CALLS    #1, CLI$PRESENT                      ;
                 04          50  E9 000D2          BLBC     R0, 5$                               ;
          04     AE          02  88 000D5          BISB2    #2, ATTRIBUTES                       ; 1641
          2C     AE          0A  B0 000D9 5$:      MOVW     #10, DESC_LIST+32                    ; 1643
          30     AE    1A    A7  9E 000DD          MOVAB    P.AAG, DESC_LIST+36                  ; 1644
                       2C    AE  9F 000E2          PUSHAB   DESC_LIST+32                         ; 1645
                 66          01  FB 000E5          CALLS    #1, CLI$PRESENT                      ;
                 04          50  E9 000E8          BLBC     R0, 6$                               ;
          04     AE          04  88 000EB          BISB2    #4, ATTRIBUTES                       ; 1647
          2C     AE          08  B0 000EF 6$:      MOVW     #8, DESC_LIST+32                     ; 1653
          30     AE    24    A7  9E 000F3          MOVAB    P.AAH, DESC_LIST+36                  ; 1654
                             52  D4 000F8          CLRL     IF_STATE_PTR                         ; 1656
                       2C    AE  9F 000FA          PUSHAB   DESC_LIST+32                         ; 1657
                 66          01  FB 000FD          CALLS    #1, CLI$PRESENT                      ;
                 53          50  D0 00100          MOVL     R0, IF_STATE_PRESENT                 ;
                 04          53  E9 00103          BLBC     IF_STATE_PRESENT, 7$                 ; 1658
                 52    1C    AE  9E 00106          MOVAB    DESC_LIST+16, IF_STATE_PTR           ; 1660
                 54          53  D2 0010A 7$:      MCOML    IF_STATE_PRESENT, R4                 ; 1709
                 0C          53  E9 0010D 8$:      BLBC     IF_STATE_PRESENT, 9$                 ;
                       1C    AE  9F 00110          PUSHAB   DESC_LIST+16                         ; 1680
                       30    AE  9F 00113          PUSHAB   DESC_LIST+32                         ; 1679
                 68          02  FB 00116          CALLS    #2, CLI$GET_VALUE                    ; 1680
                 25          50  E9 00119          BLBC     STATUS, 12$                          ; 1681
                             55  DD 0011C 9$:      PUSHL    SET_STATE_PTR                        ; 1699
                       18    AE  9F 0011E          PUSHAB   DESC_LIST+8                          ; 1698
                       0C    AE  9F 00121          PUSHAB   ATTRIBUTES                           ; 1695
                             52  DD 00124          PUSHL    IF_STATE_PTR                         ; 1698
                       1C    AE  9F 00126          PUSHAB   DESC_LIST                            ; 1695
                       04    AC  DD 00129          PUSHL    KEY_TABLE_ID                         ; 1698
        FA5A     CF          06  FB 0012C          CALLS    #6, SMG$ADD_KEY_DEF                  ;
                 0A          50  E8 00131          BLBS     ADD_STATUS, 11$                      ; 1700
                             50  DD 00134 10$:     PUSHL    ADD_STATUS                           ; 1702
   00000000G     00          01  FB 00136          CALLS    #1, LIB$SIGNAL                       ;
                                 04 0013D          RET                                           ;
                 CC          54  E9 0013E 11$:     BLBC     R4, 8$                               ; 1709
                       0C    AE  9F 00141 12$:     PUSHAB   DESC_LIST                            ; 1718
          04     AE          04  D0 00144          MOVL     #4, 4(SP)                            ;
                       04    AE  9F 00148          PUSHAB   4(SP)                                ;
   00000000G     00          02  FB 0014B          CALLS    #2, LIB$SFREEN_DD                    ;
                                 04 00152          RET                                           ; 1721
                               0000 00153 13$:     .WORD    Save nothing                         ; 1497
                 50    08    AC  D0 00155          MOVL     8(AP), R0                            ;
                 50    04    A0  D0 00159          MOVL     4(R0), R0                            ;
                       D4    A0  9F 0015D          PUSHAB   DESC_LIST_PTR                        ;
                             01  DD 00160          PUSHL    #1                                   ;
                             5E  DD 00162          PUSHL    SP                                   ;
                 7E    04    AC  7D 00164          MOVQ     4(AP), -(SP)                         ;
       0000V     CF          03  FB 00168          CALLS    #3, DEFINE_KEY_HANDLER               ;
                                 04 0016D          RET                                           ;

; Routine Size:  366 bytes,    Routine Base:  _SMG$CODE + 04C5
```

```
%SBTTL 'DEFINE_KEY_HANDLER'
ROUTINE DEFINE_KEY_HANDLER (
    SIGARGS: REF BLOCK [, BYTE],            ! Signal arguments list
    MCHARGS: REF BLOCK [, BYTE],            ! Mechanism arguments list
    ENBARGS: REF VECTOR [, LONG]            ! Enable arguments list
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure is the condition handler for SMG$DEFINE_KEY.
!       It intercepts CLI$ errors signalled by CLI routines which are
!       called from SMG$DEFINE_KEY, and also any errors signalled
!       by SMG$DEFINE_KEY itself.  The handler frees the dynamic strings
!       which were allocated by SMG$DEFINE_KEY and causes a return to
!       the caller of SMG$DEFINE_KEY with a return status value of the
!       error which was signalled.
!
! CALLING SEQUENCE:
!
!       Called by VMS exception handling
!
! FORMAL PARAMETERS:
!
!       SIGARGS         - The signal arguments list
!
!       MCHARGS         - The mechanism arguments list
!
!       ENBARGS         - The enable arguments list.  The enable
!                         argument is:
!
!                               ..ENBARGS [1] - Pointer to the first of
!                                               four dynamic string
!                                               descriptors to be freed.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL      Normal successful completion
!       SS$_RESIGNAL    Resignal condition to next handler
!
! SIDE EFFECTS:
!
!       Causes an unwind.
!
!--

    BEGIN

    LOCAL
```

```
          COND_NAME: BLOCK [4, BYTE];

      COND_NAME = .SIGARGS [CHF$L_SIG_NAME];

      !+
      ! Return if this is SS$_UNWIND.
      !-

      IF .COND_NAME EQLU SS$_UNWIND
      THEN
          RETURN SS$_RESIGNAL;

      !+
      ! If this is not a zero-level error and is not from the CLI$ facility,
      ! resignal.
      !

      IF (.COND_NAME [STS$V_FAC_NO] NEQU CLI$_FACILITY) AND
         (.MCHARGS [CHF$L_MCH_DEPTH] NEQU 0)
      THEN
          RETURN SS$_RESIGNAL;

      !+
      ! If the error is "no command on line", just change the return status
      ! to SS$_NORMAL, otherwise use the signalled status.
      !-

      IF .COND_NAME EQLU CLI$_NOCOMD
      THEN
          MCHARGS [CHF$L_MCH_SAVR0] = SS$_NORMAL
      ELSE
          MCHARGS [CHF$L_MCH_SAVR0] = .COND_NAME;

      !+
      ! Free the dynamic strings.
      !-

      IF ..ENBARGS [1] NEQA 0
      THEN
          LIB$SFREEN_DD (%REF(4), ..ENBARGS [1]);

      !+
      ! Unwind back to the caller of SMG$DEFINE_KEY.
      !-

      RETURN $UNWIND ();

      END;                                  ! End of routine DEFINE_KEY_HANDLER


                                                   .EXTRN  SYS$UNWIND

                                 0000 00000 DEFINE_KEY_HANDLER:
                                                   .WORD   Save nothing          ; 1723
             5E              04   C2 00002         SUBL2   #4, SP                ;
             50        04    AC   D0 00005         MOVL    SIGARGS, R0           ; 1781
```

```
                                51      04  A0  D0 00009       MOVL    4(R0), COND_NAME                              ;
                     00000920   8F          51  D1 0000D       CMPL    COND_NAME, #2336                              ; 1787
                                            10  13 00014       BEQL    1$                                            ;
       00G          51          0C          10  ED 00016       CMPZV   #16, #12, COND_NAME, S^CLI$_FACILITY          ; 1796
                                            0F  13 0001B       BEQL    2$                                            ;
                                50      08  AC  D0 0001D       MOVL    MCHARGS, R0                                   ; 1797
                                        08  A0  D5 00021       TSTL    8(R0)                                         ;
                                            06  13 00024       BEQL    2$                                            ;
                                50    0918  8F  3C 00026 1$:   MOVZWL  #2328, R0                                     ; 1799
                                                04 0002B       RET                                                   ;
                                50      08  AC  D0 0002C 2$:   MOVL    MCHARGS, R0                                   ; 1808
                    00000000G   8F          51  D1 00030       CMPL    COND_NAME, #CLI$_NOCOMD                       ; 1806
                                            06  12 00037       BNEQ    3$                                            ;
                           0C   A0          01  D0 00039       MOVL    #1, 12(R0)                                    ; 1808
                                            04  11 0003D       BRB     4$                                            ;
                           0C   A0          51  D0 0003F 3$:   MOVL    COND_NAME, 12(R0)                             ; 1810
                                50      0C  AC  D0 00043 4$:   MOVL    ENBARGS, R0                                   ; 1816
                                        04  B0  D5 00047       TSTL    @4(R0)                                        ;
                                            11  13 0004A       BEQL    5$                                            ;
                                        04  B0  DD 0004C       PUSHL   @4(R0)                                        ; 1818
                           04   AE          04  D0 0004F       MOVL    #4, 4(SP)                                     ;
                                        04  AE  9F 00053       PUSHAB  4(SP)                                         ;
                    00000000G   00          02  FB 00056       CALLS   #2, LIB$SFREEN_DD                             ;
                                            7E  7C 0005D 5$:   CLRQ    -(SP)                                         ; 1824
                    00000000G   00          02  FB 0005F       CALLS   #2, SYS$UNWIND                                ;
                                                04 00066       RET                                                   ; 1826

; Routine Size:  103 bytes,    Routine Base:  _SMG$CODE + 0633
```

```
1 %SBTTL 'SMG$LOAD_KEY_DEFS'
1 GLOBAL ROUTINE SMG$LOAD_KEY_DEFS (
1     KEY_TABLE_ID: REF VECTOR [, LONG],
1     FILESPEC: REF BLOCK [, BYTE],
1     DEFAULT_FILESPEC: REF BLOCK [, BYTE],
1     LOGNAM_FLAG: REF VECTOR [, LONG]
1     ) =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !
1 !       This procedure opens and reads a file containing DEFINE/KEY
1 !       command lines and calls SMG$DEFINE_KEY for each line in the file.
1 !       This can be used by utilities to conveniently load a default set
1 !       of key definitions.
1 !
1 !       Use of SMG$LOAD_KEY_DEFS requires that the calling program be
1 !       run under the DCL command language interpreter.
1 !
1 ! CALLING SEQUENCE:
1 !
1 !       RET_STATUS.wlc.v = SMG$LOAD_KEY_DEFS (
1 !                                   KEY_TABLE_ID.rl.r,
1 !                                   FILESPEC.rt.dx
1 !                                   [, [DEFAULT_FILESPEC.rt.dx]
1 !                                   [, [DEFAULT_LOGNAM.rt.dx] ]])
1 !
1 ! FORMAL PARAMETERS:
1 !
1 !       KEY_TABLE_ID      A longword containing the identification of the
1 !                         key-definition table into which are to be loaded
1 !                         the key definitions.
1 !
1 !       FILESPEC          A string containing the file specification for the
1 !                         file of DEFINE/KEY commands.
1 !
1 !       DEFAULT_FILESPEC A string containing the default file specification
1 !                         for the file of DEFINE/KEY commands.  If omitted,
1 !                         the empty string is used.
1 !
1 ! IMPLICIT INPUTS:
1 !
1 !       NONE
1 !
1 ! IMPLICIT OUTPUTS:
1 !
1 !       NONE
1 !
1 ! COMPLETION STATUS:
1 !
1 !       SS$_NORMAL        Normal successful completion
1 !       SMG$_FILTOOLON    File specification is too long (over 255 characters)
1 !       SMG$_xxx          Any error status returned from SMG$DEFINE_KEY
1 !       RMS$_xxx          Any error status returned from $OPEN
1 !
1 ! SIDE EFFECTS:
1 !
```

```
1 !          Errors encountered while processing command lines are signalled.
1 !
1 !--
1
2       BEGIN
2
2       LOCAL
2           KEYBOARD_ID,                          ! Keyboard id
2           FILESPEC_PTR,                         ! Pointer to filespec desc
2           RSLBUF: VECTOR [LNM$C_NAMLENGTH, BYTE], ! Resultant buffer
2           RSLBUF_DESC: BLOCK [8, BYTE],         ! Resultant descriptor
2           LOGNAM_DESC: BLOCK [8, BYTE],         ! Logical name descriptor
2           LINE_DESC: BLOCK [8, BYTE],           ! Command line descriptor
2           OPEN_STATUS;
2
2       BUILTIN
2           ACTUALCOUNT,
2           NULLPARAMETER;
2
2       LITERAL
2           K_KEY_TABLE_ID = 1,
2           K_FILESPEC = 2,
2           K_DEFAULT_FILESPEC = 3,
2           K_LOGNAM_FLAG = 4;
2
2       !+
2       ! If LOGNAM_FLAG was specified, replace FNM with the null string if
2       ! it doesn't translate as a logical name.
2       !-
2
2       FILESPEC_PTR = FILESPEC [0,0,0,0];
2       IF NOT NULLPARAMETER (K_LOGNAM_FLAG)
2       THEN
3           BEGIN
3           IF .LOGNAM_FLAG [0]
3           THEN
4               BEGIN
4               !+
4               ! Set up the descriptors.
4               !-
4
4               LOCAL
4                   ANL_STATUS;
4               ANL_STATUS = LIB$ANALYZE_SDESC_R2 (FILESPEC [0,0,0,0],
4                   LOGNAM_DESC [DSC$W_LENGTH], LOGNAM_DESC [DSC$A_POINTER]);
4               IF NOT .ANL_STATUS
4               THEN
4                   RETURN .ANL_STATUS;
4               LOGNAM_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
4               LOGNAM_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
4               RSLBUF_DESC [DSC$W_LENGTH] = LNM$C_NAMLENGTH;
4               RSLBUF_DESC [DSC$A_POINTER] = RSLBUF;
4               RSLBUF_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
4               RSLBUF_DESC [DSC$B_CLASS] = DSC$K_CLASS_S;
4
4               !+
4               ! Try translating the logical name.  If it doesn't translate,
```

```
                | use a zero-length string as the filespec.
                |-

                IF ($TRNLOG (LOGNAM=LOGNAM_DESC, RSLBUF=RSLBUF_DESC) EQLU
                    SS$_NOTRAN)
                THEN
                    BEGIN
                    LOGNAM_DESC [DSC$W_LENGTH] = 0;
                    FILESPEC_PTR = LOGNAM_DESC;
                    END;
                END;
            END;


        !+
        ! Open the file.
        !-

        OPEN_STATUS = SMG$CREATE_VIRTUAL_KEYBOARD (KEYBOARD_ID, .FILESPEC_PTR,
            (IF ACTUALCOUNT() GEQU K_DEFAULT_FILESPEC
             THEN
                .DEFAULT_FILESPEC
             ELSE
                0));
        IF NOT .OPEN_STATUS
        THEN
            RETURN .OPEN_STATUS;

        !+
        ! Set up LINE_DESC.
        !-

        LINE_DESC [DSC$W_LENGTH] = 0;
        LINE_DESC [DSC$B_DTYPE] = DSC$K_DTYPE_T;
        LINE_DESC [DSC$B_CLASS] = DSC$K_CLASS_D;
        LINE_DESC [DSC$A_POINTER] = 0;

        !+
        ! While we're not at EOF, read records and call SMG$DEFINE_KEY to
        ! process the command line.
        !-

        WHILE (SMG$READ_STRING (KEYBOARD_ID,LINE_DESC)) DO
            BEGIN
            LOCAL
                DEFINE_STATUS;
            DEFINE_STATUS = SMG$DEFINE_KEY (KEY_TABLE_ID [0], LINE_DESC);
            IF NOT .DEFINE_STATUS
            THEN
                SIGNAL (.DEFINE_STATUS);
            END;

        LIB$SFREE1_DD (LINE_DESC);
        SMG$DELETE_VIRTUAL_KEYBOARD (KEYBOARD_ID);

        RETURN SS$_NORMAL;
```

```
; 1934          1998  1     END;                                            ! End of routine SMG$LOAD_KEY_DEFS


                                                               .EXTRN  SYS$TRNLOG

                                        000C 00000             .ENTRY  SMG$LOAD_KEY_DEFS, Save R2,R3                   ; 1828
                     5E        FEE4   CE   9E 00002            MOVAB   -284(SP), SP                                    ;
                     53          08   AC   D0 00007            MOVL    FILESPEC, FILESPEC_PTR                          ; 1914
                     04               6C   91 0000B            CMPB    (AP), #4                                        ; 1915
                                      54   1F 0000E            BLSSU   1$                                              ;
                                 10   AC   D5 00010            TSTL    16(AP)                                          ;
                                      4F   13 00013            BEQL    1$                                              ;
                     4B          10   BC   E9 00015            BLBC    @LOGNAM_FLAG, 1$                                ; 1918
                     50          08   AC   D0 00019            MOVL    FILESPEC, R0                                    ; 1928
                          00000000G   00   16 0001D            JSB     LIB$ANALYZE_SDESC_R2                            ;
               0C    AE               51   B0 00023            MOVW    R1, LOGNAM_DESC                                 ;
               10    AE               52   D0 00027            MOVL    R2, LOGNAM_DESC+4                               ;
                     4E               50   E9 0002B            BLBC    ANL_STATUS, 4$                                  ; 1929
               0E    AE        010E   8F   B0 0002E            MOVW    #270, LOGNAM_DESC+2                             ; 1932
               18    AE          1C   AE   9E 00034            MOVAB   RSLBUF, RSLBUF_DESC+4                           ; 1935
               14    AE    010E00FF   8F   D0 00039            MOVL    #17694975, RSLBUF_DESC                          ; 1934
                                      7E   7C 00041            CLRQ    -(SP)                                           ; 1944
                                      7E   D4 00043            CLRL    -(SP)                                           ;
                                 20   AE   9F 00045            PUSHAB  RSLBUF_DESC                                     ;
                                      7E   D4 00048            CLRL    -(SP)                                           ;
                                 20   AE   9F 0004A            PUSHAB  LOGNAM_DESC                                     ;
          00000000G  00               06   FB 0004D            CALLS   #6, SYS$TRNLOG                                  ;
          00000629   8F               50   D1 00054            CMPL    R0, #1577                                       ;
                                      07   12 0005B            BNEQ    1$                                              ;
                                 0C   AE   B4 0005D            CLRW    LOGNAM_DESC                                     ; 1948
                     53          0C   AE   9E 00060            MOVAB   LOGNAM_DESC, FILESPEC_PTR                       ; 1949
                     03               6C   91 00064 1$:        CMPB    (AP), #3                                        ; 1960
                                      05   1F 00067            BLSSU   2$                                              ;
                                 0C   AC   DD 00069            PUSHL   DEFAULT_FILESPEC                                ; 1962
                                      02   11 0006C            BRB     3$                                              ;
                                      7E   D4 0006E 2$:        CLRL    -(SP)                                           ; 1960
                                      53   DD 00070 3$:        PUSHL   FILESPEC_PTR                                    ; 1959
                                 08   AE   9F 00072            PUSHAB  KEYBOARD_ID                                     ;
          00000000G  00               03   FB 00075            CALLS   #3, SMG$CREATE_VIRTUAL_KEYBOARD                 ;
                     4A               50   E9 0007C 4$:        BLBC    OPEN_STATUS, 7$                                 ; 1965
               04    AE    020E0000   8F   D0 0007F            MOVL    #34471936, LINE_DESC                            ; 1973
                                 08   AE   D4 00087            CLRL    LINE_DESC+4                                     ; 1976
                                 04   AE   9F 0008A 5$:        PUSHAB  LINE_DESC                                       ; 1983
                                 04   AE   9F 0008D            PUSHAB  KEYBOARD_ID                                     ;
          00000000G  00               02   FB 00090            CALLS   #2, SMG$READ_STRING                             ;
                     19               50   E9 00097            BLBC    R0, 6$                                          ;
                                 04   AE   9F 0009A            PUSHAB  LINE_DESC                                       ; 1987
                                 04   AC   DD 0009D            PUSHL   KEY_TABLE_ID                                    ;
               FD86  CF               02   FB 000A0            CALLS   #2, SMG$DEFINE_KEY                              ;
                     E2               50   E8 000A5            BLBS    DEFINE_STATUS, 5$                               ; 1988
                                      50   DD 000A8            PUSHL   DEFINE_STATUS                                   ; 1990
          00000000G  00               01   FB 000AA            CALLS   #1, LIB$SIGNAL                                  ;
                                      D7   11 000B1            BRB     5$                                              ; 1983
                                 04   AE   9F 000B3 6$:        PUSHAB  LINE_DESC                                       ; 1993
          00000000G  00               01   FB 000B6            CALLS   #1, LIB$SFREE1_DD                               ;
                                      5E   DD 000BD            PUSHL   SP                                              ; 1994
```

```
         00000000G  00         01  FB 000BF          CALLS   #1, SMG$DELETE_VIRTUAL_KEYBOARD          ;
                    50         01  D0 000C6          MOVL    #1, R0                                    ; 1996
                                   04 000C9 7$:      RET                                               ; 1998

; Routine Size:  202 bytes,    Routine Base:  _SMG$CODE + 069A
```

```
%SBTTL 'SMG$LIST_KEY_DEFS'
GLOBAL ROUTINE SMG$LIST_KEY_DEFS (
    KEY_TABLE_ID: REF VECTOR [, LONG],
    CONTEXT: REF VECTOR [, LONG],
    KEY_NAME: REF BLOCK [, BYTE],
    IF_STATE: REF BLOCK [, BYTE],
    ATTRIBUTES: REF VECTOR [, LONG],
    EQUIV_STRING: REF BLOCK [, BYTE],
    STATE_STRING: REF BLOCK [, BYTE]
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       This procedure, when called repeatedly, returns all of the
!       key definitions in a key table.
!
! CALLING SEQUENCE:
!
!       RET_STATUS.wlc.v = SMG$LIST_KEY_DEFS (
!                               KEY_TABLE_ID.rl.r,
!                               CONTEXT.ml.r,
!                               [, [KEY_NAME.rt.dx]
!                               [, [IF_STATE.rt.dx]
!                               [, [ATTRIBUTES.wl.r]
!                               [, [EQUIV_STRING.wt.dx]
!                               [, [STATE_STRING.wt.dx] ]]]]])
!
! FORMAL PARAMETERS:
!
!       KEY_TABLE_ID    A longword containing the identification of the
!                       key-definition table which is to be listed.
!
!       CONTEXT         A longword which supplies context to the search.
!                       On the initial call, CONTEXT should be set to zero
!                       by the caller.  It will be updated by SMG$LIST_KEY_DEFS;
!                       the updated value should then be passed on subsequent
!                       calls to obtain the next key definition.
!
!       KEY_NAME        A string into which is stored the key name of the next
!                       definition in the key table.
!
!       IF_STATE        A string into which is stored the state name which
!                       qualifies the next definition in the key table
!
!       ATTRIBUTES      A longword into which is stored the attributes
!                       bitmask for the next key definition.  See the description
!                       of SMG$ADD_KEY_DEF for more information.
!
!       EQUIV_STRING    A string into which is stored the equivalence-string
!                       for the next key definition.
!
!       STATE_STRING    A string into which is stored the new state-name,
!                       if any, which is set by the next key definition.
!                       If this key definition sets a state, the ATTRIBUTES
!                       flag SMG$V_KEY_SETSTATE is set.
!
```

```
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL        Normal successful completion
!       SMG$_INVKTB_ID    Invalid key-table id
!       SMG$_NOMOREKEYS   No more keys in table
!       LIB$_xxx          Any error status from LIB$SCOPY_DXDX
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        FOUND_KDE,                                        ! KDE found
        KDE: REF KDE_R_KDE_STRUCT,                        ! Key definition entry
        KTH: REF KTH_R_KTH_STRUCT,                        ! Key table header
        DUMMY_KDE: KDE_R_KDE_STRUCT;                      ! Used to store first entry

    BUILTIN
        AP,
        NULLPARAMETER;

    !+
    ! Define positions of arguments
    !-

    LITERAL
        K_KEY_TABLE_ID = 1,
        K_CONTEXT = 2,
        K_KEY_NAME = 3,
        K_IF_STATE = 4,
        K_ATTRIBUTES = 5,
        K_EQUIV_STRING = 6,
        K_STATE_STRING = 7;

    !+
    ! Validate KEY_TABLE_ID and get the Key Table Header.
    !-

    $SMG$VALIDATE_KTH (KEY_TABLE_ID, KTH);

    !+
    ! If this is the first time, form a linear linked list of the tree.
    !-

    KDE = .CONTEXT [0];
```

```
IF .KDE EQLA 0
THEN
    BEGIN
    LOCAL
        LAST_KDE: REF KDE_R_KDE_STRUCT,
        TRAVERSE_STATUS;

    LAST_KDE = DUMMY_KDE;
    DUMMY_KDE [KDE_A_NEXT] = 0;
    TRAVERSE_STATUS = LIB$TRAVERSE_TREE (
        KTH [KTH_A_TREEHEAD],          ! Address of treehead
        TRAVERSE_ROUTINE,              ! Traverse routine
        LAST_KDE);                     ! Pointer to previous KDE
    IF NOT .TRAVERSE_STATUS
    THEN
        RETURN .TRAVERSE_STATUS;
    KDE = DUMMY_KDE;            ! Get first KDE
    END;

!+
! Get address of next KDE.  If no more keys, return status.
! Skip over deleted keys.
!-

DO
    BEGIN
    KDE = .KDE [KDE_A_NEXT];
    CONTEXT [0] = .KDE;
    IF .KDE EQLA 0
    THEN
        RETURN SMG$_NOMOREKEYS;
    END
UNTIL .KDE [KDE_V_DEFINED];

!+
! Store values.
!-

IF NOT NULLPARAMETER (K_ATTRIBUTES)
THEN
    ATTRIBUTES [0] = .KDE [KDE_L_ATTR] AND KDE_M_USER_ATTR;

IF NOT NULLPARAMETER (K_KEY_NAME)
THEN
    BEGIN
    LOCAL
        KEY_NAME_AREA: VECTOR [32, BYTE],
        KEY_CODE_PTR: REF VECTOR [, BYTE],
        KEY_CODE: WORD,
        STATUS;

     !+
     ! Convert key code to key name.
     !-

    KEY_CODE_PTR = (.KDE [KDE_A_TKEY_POINTER] + .KDE [KDE_W_TKEY_LENGTH])
        - 2;
```

```
        KEY_CODE<8,8> = .KEY_CODE_PTR [0]; ! Swap bytes
        KEY_CODE<0,8> = .KEY_CODE_PTR [1];
        IF NOT SMG$$KEYCODE_TO_NAME (.KEY_CODE, KEY_NAME_AREA)
        THEN
            KEY_NAME_AREA [0] = 0;
        STATUS = LIB$SCOPY_R_DX6 (.KEY_NAME_AREA [0], KEY_NAME_AREA [1],
            KEY_NAME [0,0,0,0]);
        IF NOT .STATUS
        THEN
            RETURN .STATUS;
        END;

    IF NOT NULLPARAMETER (K_IF_STATE)
    THEN
        BEGIN
        LOCAL
            STATUS;
        STATUS = LIB$SCOPY_R_DX6 (.KDE [KDE_W_TKEY_LENGTH] - 3,
            .KDE [KDE_A_TKEY_POINTER], IF_STATE [0,0,0,0]);
        IF NOT .STATUS
        THEN
            RETURN .STATUS;
        END;

    IF NOT NULLPARAMETER (K_EQUIV_STRING)
    THEN
        BEGIN
        LOCAL
            STATUS;
        STATUS = LIB$SCOPY_DXDX6 (KDE [KDE_R_EQUIV_DESC],
            EQUIV_STRING [0,0,0,0]);
        IF NOT .STATUS
        THEN
            RETURN .STATUS;
        END;

    IF NOT NULLPARAMETER (K_STATE_STRING)
    THEN
        BEGIN
        LOCAL
            STATUS;
        STATUS = LIB$SCOPY_DXDX6 (KDE [KDE_R_STATE_DESC],
            STATE_STRING [0,0,0,0]);
        IF NOT .STATUS
        THEN
            RETURN .STATUS;
        END;

    RETURN SS$_NORMAL;

    END;                                        ! End of routine SMG$LIST_KEY_DEFS


                        03FC 00000        .ENTRY  SMG$LIST_KEY_DEFS, Save R2,R3,R4,R5,R6,R7,- ; 2000
```

```
                                                        R8,R9                             ;
              59 00000000G  00  9E 00002        MOVAB   LIB$SCOPY_DXDX6, R9              ;
              58 00000000G  00  9E 00009        MOVAB   LIB$SCOPY_R_DX6, R8              ;
              5E        A4  AE  9E 00010        MOVAB   -92(SP), SP                      ;
              50        04  BC  D0 00014        MOVL    @KEY_TABLE_ID, KTH               ; 2106
                            06  13 00018        BEQL    1$                               ;
              50        10  A0  D1 0001A        CMPL    16(KTH), KTH                     ;
                            08  13 0001E        BEQL    2$                               ;
              50 00000000G  8F  D0 00020 1$:    MOVL    #SMG$_INVKTB_ID, R0              ;
                                04 00027        RET                                      ;
              57        08  BC  D0 00028 2$:    MOVL    @CONTEXT, KDE                    ; 2112
                            1D  12 0002C        BNEQ    3$                               ; 2113
              6E        24  AE  9E 0002E        MOVAB   DUMMY_KDE, LAST_KDE              ; 2120
                        50  AE  D4 00032        CLRL    DUMMY_KDE+44                     ; 2121
                            5E  DD 00035        PUSHL   SP                               ; 2123
                     0000V  CF  9F 00037        PUSHAB  TRAVERSE_ROUTINE                 ;
                            50  DD 0003B        PUSHL   KTH                              ;
       00000000G  00        03  FB 0003D        CALLS   #3, LIB$TRAVERSE_TREE            ;
                  69        50  E9 00044        BLBC    TRAVERSE_STATUS, 7$              ; 2126
                  57    24  AE  9E 00047        MOVAB   DUMMY_KDE, KDE                   ; 2129
                  57    2C  A7  D0 0004B 3$:    MOVL    44(KDE), KDE                     ; 2139
              08  BC        57  D0 0004F        MOVL    KDE, @CONTEXT                    ; 2140
                            08  12 00053        BNEQ    4$                               ; 2141
              50 00000000G  8F  D0 00055        MOVL    #SMG$_NOMOREKEYS, R0             ; 2143
                                04 0005C        RET                                      ;
                        33  A7  95 0005D 4$:    TSTB    51(KDE)                          ; 2145
                            E9  18 00060        BGEQ    3$                               ;
                  05        6C  91 00062        CMPB    (AP), #5                         ; 2151
                            0A  1F 00065        BLSSU   5$                               ;
                        14  AC  D5 00067        TSTL    20(AP)                           ;
                            05  13 0006A        BEQL    5$                               ;
              14  BC    30  A7  9A 0006C        MOVZBL  48(KDE), @ATTRIBUTES             ; 2153
                  03        6C  91 00071 5$:    CMPB    (AP), #3                         ; 2155
                            3D  1F 00074        BLSSU   8$                               ;
                        0C  AC  D5 00076        TSTL    12(AP)                           ;
                            38  13 00079        BEQL    8$                               ;
                  50    0C  A7  3C 0007B        MOVZWL  12(KDE), R0                      ; 2168
                  50    10  A7  C0 0007F        ADDL2   16(KDE), R0                      ;
                  50        02  C2 00083        SUBL2   #2, KEY_CODE_PTR                 ; 2169
  51          08  08        60  F0 00086        INSV    (KEY_CODE_PTR), #8, #8, KEY_CODE ; 2170
                  51    01  A0  90 0008B        MOVB    1(KEY_CODE_PTR), KEY_CODE        ; 2171
                        04  AE  9F 0008F        PUSHAB  KEY_NAME_AREA                    ; 2172
                  7E        51  3C 00092        MOVZWL  KEY_CODE, -(SP)                  ;
       00000000G  00        02  FB 00095        CALLS   #2, SMG$$KEYCODE_TO_NAME         ;
                  03        50  E8 0009C        BLBS    R0, 6$                           ;
                        04  AE  94 0009F        CLRB    KEY_NAME_AREA                    ; 2174
                  51    05  AE  9E 000A2 6$:    MOVAB   KEY_NAME_AREA+1, R1              ; 2175
                  52    0C  AC  D0 000A6        MOVL    KEY_NAME, R2                     ; 2176
                  50    04  AE  9A 000AA        MOVZBL  KEY_NAME_AREA, R0                ;
                            68  16 000AE        JSB     LIB$SCOPY_R_DX6                  ;
                  4F        50  E9 000B0 7$:    BLBC    STATUS, 12$                      ; 2177
                  04        6C  91 000B3 8$:    CMPB    (AP), #4                         ; 2182
                            19  1F 000B6        BLSSU   9$                               ;
                        10  AC  D5 000B8        TSTL    16(AP)                           ;
                            14  13 000BB        BEQL    9$                               ;
                  50    0C  A7  3C 000BD        MOVZWL  12(KDE), R0                      ; 2187
                  50        03  C2 000C1        SUBL2   #3, R0                           ;
```

```
           52      10    AC   D0 000C4          MOVL    IF_STATE, R2                         ; 2188
           51      10    A7   D0 000C8          MOVL    16(KDE), R1                          ;
                         68   16 000CC          JSB     LIB$SCOPY_R_DX6                      ;
           31            50   E9 000CE          BLBC    STATUS, 12$                          ; 2189
           06            6C   91 000D1  9$:     CMPB    (AP), #6                             ; 2194
                         12   1F 000D4          BLSSU   10$                                  ;
                   18    AC   D5 000D6          TSTL    24(AP)                               ;
                         0D   13 000D9          BEQL    10$                                  ;
           50      14    A7   9E 000DB          MOVAB   20(KDE), R0                          ; 2199
           51      18    AC   D0 000DF          MOVL    EQUIV_STRING, R1                     ; 2200
                         69   16 000E3          JSB     LIB$SCOPY_DXDX6                      ;
           1A            50   E9 000E5          BLBC    STATUS, 12$                          ; 2201
           07            6C   91 000E8  10$:    CMPB    (AP), #7                             ; 2206
                         12   1F 000EB          BLSSU   11$                                  ;
                   1C    AC   D5 000ED          TSTL    28(AP)                               ;
                         0D   13 000F0          BEQL    11$                                  ;
           50      1C    A7   9E 000F2          MOVAB   28(KDE), R0                          ; 2211
           51      1C    AC   D0 000F6          MOVL    STATE_STRING, R1                     ; 2212
                         69   16 000FA          JSB     LIB$SCOPY_DXDX6                      ;
           03            50   E9 000FC          BLBC    STATUS, 12$                          ; 2213
           50            01   D0 000FF  11$:    MOVL    #1, R0                               ; 2218
                              04 00102  12$:    RET                                          ; 2220

; Routine Size:  259 bytes,    Routine Base:  _SMG$CODE + 0764
```

```
%SBTTL 'TRAVERSE_ROUTINE'
ROUTINE TRAVERSE_ROUTINE (
    NEXT_KDE_PTR:,                                  ! Pointer to next KDE
    LAST_KDE_PTR: REF VECTOR [, LONG]               ! Contains pointer to last KDE
    ) =

!++
! FUNCTIONAL DESCRIPTION:
!
!       Routine called by LIB$TRAVERSE_TREE.  It is used to link in this
!       KDE to the list that is being built of KDEs in order.
!
! CALLING SEQUENCE:
!
!       Not directly called.
!
! FORMAL PARAMETERS:
!
!       NEXT_KDE_PTR    A longword that contains a pointer to the next
!                       KDE.
!
!       LAST_KDE_PTR    A longword that contains a pointer to the previous
!                       KDE.
!
! IMPLICIT INPUTS:
!
!       NONE
!
! IMPLICIT OUTPUTS:
!
!       NONE
!
! COMPLETION STATUS:
!
!       SS$_NORMAL      Normal successful completion
!
! SIDE EFFECTS:
!
!       NONE
!
!--

    BEGIN

    LOCAL
        THIS_KDE: REF KDE_R_KDE_STRUCT, ! This Key Definition Entry
        LAST_KDE: REF KDE_R_KDE_STRUCT; ! Previous Key Definition Entry

    !+
    ! Get pointers.
    !-

    THIS_KDE = .NEXT_KDE_PTR;
    LAST_KDE = .LAST_KDE_PTR [0];

    !+
    ! Link in this KDE and zero the next pointer.
```

```
; 2216          2278  2       !-
; 2217          2279  2
; 2218          2280  2       LAST_KDE [KDE_A_NEXT] = THIS_KDE [0,0,0,0];
; 2219          2281  2       THIS_KDE [KDE_A_NEXT] = 0;
; 2220          2282  2
; 2221          2283  2       !+
; 2222          2284  2       ! Update the "last KDE" pointer.
; 2223          2285  2       !-
; 2224          2286  2
; 2225          2287  2       LAST_KDE_PTR [0] = THIS_KDE [0,0,0,0];
; 2226          2288  2
; 2227          2289  2       RETURN SS$_NORMAL;
; 2228          2290  2
; 2229          2291  1       END;                                          ! End of routine TRAVERSE_ROUTINE


                                    0000 00000 TRAVERSE_ROUTINE:
                                                     .WORD   Save nothing                          ; 2222
                 51      04  AC  D0 00002            MOVL    NEXT_KDE_PTR, THIS_KDE                ; 2273
                 50      08  BC  D0 00006            MOVL    @LAST_KDE_PTR, LAST_KDE               ; 2274
          2C  A0             51  D0 0000A            MOVL    THIS_KDE, 44(LAST_KDE)                ; 2280
                         2C  A1  D4 0000E            CLRL    44(THIS_KDE)                          ; 2281
          08  BC             51  D0 00011            MOVL    THIS_KDE, @LAST_KDE_PTR               ; 2287
                 50          01  D0 00015            MOVL    #1, R0                                ; 2289
                                 04 00018            RET                                           ; 2291

; Routine Size:  25 bytes,    Routine Base:  _SMG$CODE + 0867
```

```
; 2231          2292  1 END                                      ! End of module SMG$KEYPAD
; 2232          2293  1
; 2233          2294  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                            PSECT SUMMARY
;
;       Name                     Bytes                      Attributes
;
;     _SMG$CODE                      2176  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        26         0          581        00:01.0
;     _$255$DUA28:[SMGRTL.OBJ]RTLLIB.L32;1    36         5        13            8        00:00.1
;     _$255$DUA28:[SMGRTL.OBJ]SMGLIB.L32;1   469        62        13           38        00:00.5


;                             COMMAND QUALIFIERS

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

; Size:           2124 code + 52 data bytes
; Run Time:          00:47.2
; Elapsed Time:      02:36.0
; Lines/CPU Min:       2919
; Lexemes/CPU-Min: 18300
; Memory Used:  188 pages
; Compilation Complete
```

0358 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

SMGKEYUTI
LIS

SMGKEYPAD
LIS

SMGINPUT
LIS

SMGLIB
LIS