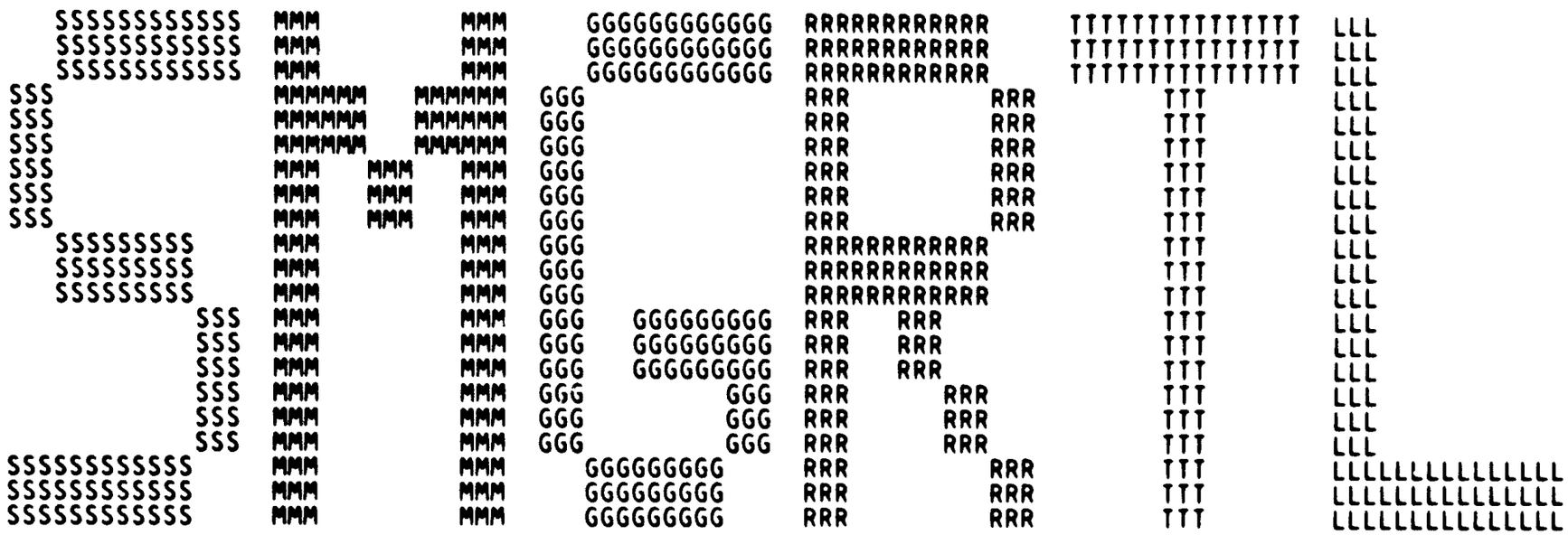

_$2

Val

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

FILEID**SMGKEYUTI


```
  SSSSSSSS  MM      MM    GGGGGGGG  KK      KK  EEEEEEEEEE  YY      YY  UU      UU  TTTTTTTTTT    IIIIII
  SSSSSSSS  MM      MM    GGGGGGGG  KK      KK  EEEEEEEEEE  YY      YY  UU      UU  TTTTTTTTTT    IIIIII
SS          MMMM  MMMM  GG          KK      KK  EE          YY      YY  UU      UU      TT        II
SS          MMMM  MMMM  GG          KK      KK  EE          YY      YY  UU      UU      TT        II
SS          MM  MM  MM  GG          KK    KK    EE            YY  YY    UU      UU      TT        II
SS          MM  MM  MM  GG          KK    KK    EE            YY  YY    UU      UU      TT        II
  SSSSSS    MM      MM  GG          KKKKKK      EEEEEEEE        YY      UU      UU      TT        II
  SSSSSS    MM      MM  GG          KKKKKK      EEEEEEEE        YY      UU      UU      TT        II
        SS  MM      MM  GG  GGGGGG  KK    KK    EE              YY      UU      UU      TT        II
        SS  MM      MM  GG  GGGGGG  KK    KK    EE              YY      UU      UU      TT        II
        SS  MM      MM  GG      GG  KK      KK  EE              YY      UU      UU      TT        II          ....
        SS  MM      MM  GG      GG  KK      KK  EE              YY      UU      UU      TT        II          ....
SSSSSSSS    MM      MM    GGGGGG    KK      KK  EEEEEEEEEE      YY      UUUUUUUUUU      TT      IIIIII        ....
SSSSSSSS    MM      MM    GGGGGG    KK      KK  EEEEEEEEEE      YY      UUUUUUUUUU      TT      IIIIII        ....

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
```

```
0000     1          .TITLE  SMG$$KEY_UTIL - Key translation utility procedures
0000     2          .IDENT  /1-004/         ; File: SMGKEYUTI.MAR Edit: STAN1004
0000     3
0000     4 ;
0000     5 ;*****************************************************************************
0000     6 ;*                                                                           *
0000     7 ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
0000     8 ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
0000     9 ;*  ALL RIGHTS RESERVED.                                                     *
0000    10 ;*                                                                           *
0000    11 ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
0000    12 ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    13 ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
0000    14 ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
0000    15 ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
0000    16 ;*  TRANSFERRED.                                                             *
0000    17 ;*                                                                           *
0000    18 ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
0000    19 ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    20 ;*  CORPORATION.                                                             *
0000    21 ;*                                                                           *
0000    22 ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
0000    23 ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
0000    24 ;*                                                                           *
0000    25 ;*                                                                           *
0000    26 ;*****************************************************************************
0000    27 ;
0000    28
0000    29 ;++
0000    30 ; FACILITY: Run-Time Library Screen Management
0000    31 ;
0000    32 ; ABSTRACT:
0000    33 ;
0000    34 ;       This module contains routines which convert:
0000    35 ;               Terminator sequences to key codes
0000    36 ;               Key names to key codes
0000    37 ;               Key codes to key names
0000    38 ;
0000    39 ; ENVIRONMENT: Runs at any access mode, AST Reentrant
0000    40 ;
0000    41 ; AUTHOR: Steven B. Lionel, CREATION DATE: 24-Feb-1983
0000    42 ;
0000    43 ; MODIFIED BY:
0000    44 ;
0000    45 ; 1-001 - Original. SBL 24-Feb-1983
0000    46 ; 1-002 - Add E1 - E6 as synonyms for editing keys.  SBL 29-Jul-1983
0000    47 ; 1-003 - Add routine to convert codes to names.  SBL 17-Aug-1983
0000    48 ; 1-004 - Make En the main names. STAN 8-Jul-1984.
0000    49 ;--
```

```
                     0000     51          .SBTTL  DECLARATIONS
                     0000     52 ;
                     0000     53 ; LIBRARY MACRO CALLS:
                     0000     54 ;
                     0000     55          $SMGDEF                     ; Get terminator codes
                     0000     56 ;
                     0000     57 ; EXTERNAL DECLARATIONS:
                     0000     58 ;
                     0000     59          .DSABL  GBL                 ; Force all external symbols to be declared
                     0000     60          .EXTRN  SMG$_INVKEYNAM      ; Invalid key name
                     0000     61 ;
                     0000     62 ; MACROS:
                     0000     63 ;
                     0000     64 ;        NONE
                     0000     65 ;
                     0000     66 ; EQUATED SYMBOLS:
                     0000     67 ;
                     0000     68
0000001A             0000     69          K_CTRLZ = 26
0000001B             0000     70          K_ESC = 27
0000008F             0000     71          K_SS3 = 143
0000009B             0000     72          K_CSI = 155
                     0000     73
                     0000     74 ;
                     0000     75 ; OWN STORAGE:
                     0000     76 ;
                     0000     77 ;        NONE
                     0000     78 ;
                     0000     79 ; PSECT DECLARATIONS:
                     0000     80 ;
                     0000     81
                 00000000     82          .PSECT _SMG$CODE PIC, USR, CON, REL, LCL, SHR, -
                     0000     83                      EXE, RD, NOWRT, LONG
                     0000     84
```

```
                     86          .SBTTL  Terminator definitions
                     87
                     88 ;+
                     89 ; Define macros for table generation
                     90 ;-
                     91
                     92          .MACRO  TERM1 SEQ, NAME
                     93          .ASCII  SEQ
                     94          .WORD   SMG$K_TRM_'NAME'
                     95          .ENDM
                     96
                     97          .MACRO  TERM2 SEQ, NAME
                     98          .ASCII  SEQ
                     99          .ASCII  "~"
                    100          .WORD   SMG$K_TRM_'NAME'
                    101          .ENDM
                    102
                    103          .MACRO  TERM3 SEQ, NAME
                    104          .ASCII  SEQ
                    105          .ASCII  "~"
                    106          .BYTE   0
                    107          .WORD   SMG$K_TRM_'NAME'
                    108          .ENDM
                    109
                    110 ;+
                    111 ; Terminators of the form <ESC>x
                    112 ;-
                    113
                    114 ESC_1BYTE:
                    115          TERM1   "A", UP
                    116          TERM1   "B", DOWN
                    117          TERM1   "C", RIGHT
                    118          TERM1   "D", LEFT
                    119          TERM1   "P", PF1
                    120          TERM1   "Q", PF2
                    121          TERM1   "R", PF3
                    122          TERM1   "S", PF4
00                  123          .BYTE   0
                    124
                    125 ;+
                    126 ; Terminators of the form <CSI>x or <SS3>x
                    127 ;-
                    128
                    129 CSI_1BYTE:
                    130          TERM1   "A", UP
                    131          TERM1   "B", DOWN
                    132          TERM1   "C", RIGHT
                    133          TERM1   "D", LEFT
                    134          TERM1   "M", ENTER
                    135          TERM1   "P", PF1
                    136          TERM1   "Q", PF2
                    137          TERM1   "R", PF3
                    138          TERM1   "S", PF4
                    139          TERM1   "l", COMMA
                    140          TERM1   "m", MINUS
                    141          TERM1   "n", PERIOD
                    142          TERM1   "p", KP0
```

```
                0040   143          TERM1   "q", KP1
                0043   144          TERM1   "r", KP2
                0046   145          TERM1   "s", KP3
                0049   146          TERM1   "t", KP4
                004C   147          TERM1   "u", KP5
                004F   148          TERM1   "v", KP6
                0052   149          TERM1   "w", KP7
                0055   150          TERM1   "x", KP8
                0058   151          TERM1   "y", KP9
          00    005B   152          .BYTE   0
                005C   153
                005C   154 ;+
                005C   155 ; Terminators of the form <CSI>n~
                005C   156 ;-
                005C   157
                005C   158 CSI_2BYTE:
                005C   159          TERM2   "1", FIND
                0060   160          TERM2   "2", INSERT_HERE
                0064   161          TERM2   "3", REMOVE
                0068   162          TERM2   "4", SELECT
                006C   163          TERM2   "5", PREV_SCREEN
                0070   164          TERM2   "6", NEXT_SCREEN
        0000    0074   165          .WORD   0
                0076   166
                0076   167 ;+
                0076   168 ; Terminators of the form <CSI>nn~
                0076   169 ;-
                0076   170
                0076   171 CSI_3BYTE:
                0076   172          TERM3   "17", F6
                007C   173          TERM3   "18", F7
                0082   174          TERM3   "19", F8
                0088   175          TERM3   "20", F9
                008E   176          TERM3   "21", F10
                0094   177          TERM3   "23", F11
                009A   178          TERM3   "24", F12
                00A0   179          TERM3   "25", F13
                00A6   180          TERM3   "26", F14
                00AC   181          TERM3   "28", HELP
                00B2   182          TERM3   "29", DO
                00B8   183          TERM3   "31", F17
                00BE   184          TERM3   "32", F18
                00C4   185          TERM3   "33", F19
                00CA   186          TERM3   "34", F20
  00000000      00D0   187          .LONG   0               ; End of list
                00D4   188
```

```
                     00D4   190          .SBTTL  SMG$$TERM_TO_KEYCODE - Translate terminator to key code
                     00D4   191 ;++
                     00D4   192 ; FUNCTIONAL DESCRIPTION:
                     00D4   193 ;
                     00D4   194 ;       SMG$$TERM_TO_KEYCODE translates a terminator character sequence
                     00D4   195 ;       to a key code.
                     00D4   196 ;
                     00D4   197 ; CALLING SEQUENCE:
                     00D4   198 ;
                     00D4   199 ;       key_code.wl.v = SMG$$TERM_TO_KEYCODE (terminator.rt.r, term_length.rl.v)
                     00D4   200 ;
                     00D4   201 ;
                     00D4   202 ; FORMAL PARAMETERS:
                     00D4   203 ;
00000004             00D4   204 ;       terminator = 4          ; The terminator string, passed by reference.
                     00D4   205 ;
00000008             00D4   206 ;       term_length = 8         ; The length of the terminator string, passed
                     00D4   207 ;                               ; by immediate value.
                     00D4   208 ;
                     00D4   209 ;
                     00D4   210 ; IMPLICIT INPUTS:
                     00D4   211 ;
                     00D4   212 ;       NONE
                     00D4   213 ;
                     00D4   214 ; IMPLICIT OUTPUTS:
                     00D4   215 ;
                     00D4   216 ;       NONE
                     00D4   217 ;
                     00D4   218 ; COMPLETION STATUS:
                     00D4   219 ;
                     00D4   220 ;       The code number of the terminator key, if any.
                     00D4   221 ;       SMG$K_TRM_BUFFER_FULL if terminator length is zero
                     00D4   222 ;       SMG$K_TRM_UNKNOWN if terminator is unrecognized
                     00D4   223 ;
                     00D4   224 ; SIDE EFFECTS:
                     00D4   225 ;
                     00D4   226 ;       NONE
                     00D4   227 ;
                     00D4   228 ;--
                     00D4   229
                 000C 00D4   230          .ENTRY  SMG$$TERM_TO_KEYCODE, ^M<R2,R3>
                     00D6   231
  52   04 AC   7D   00D6   232          MOVQ    terminator(AP), R2      ; Get terminator address and length
                     00DA   233                                          ; into R2 and R3
50   53   02   C3   00DA   234          SUBL3   #2, R3, R0              ; Is length within bounds?
                43   19   00DE   235          BLSS    CHECK_SINGLE            ; May be a single character
     03   50   D1   00E0   236          CMPL    R0, #3                  ; Must be between 2 and 5 chars
                38   1A   00E3   237          BGTRU   NOTRANS                 ; No translation
                     00E5   238
     51   82   9A   00E5   239          MOVZBL  (R2)+, R1               ; Get first character
     1B   51   91   00E8   240          CMPB    R1, #K_ESC              ; Is the first character <ESC>?
                24   12   00EB   241          BNEQ    NOT_ESC                 ; Skip if not
                53   D7   00ED   242          DECL    R3                      ; Look at next character
     51   82   9A   00EF   243          MOVZBL  (R2)+, R1               ;
  5B 8F   51   91   00F2   244          CMPB    R1, #^A/[/              ; <ESC>[ is the same as <CSI>
                48   13   00F6   245          BEQL    USE_CSI                 ; Skip if it is
  4F 8F   51   91   00F8   246          CMPB    R1, #^A/O/              ; <ESC>O is the same as <SS3>
```

```
                     33   13   00FC    247          BEQL    USE_SS3                         ; Skip if it is
                  3F 51   91   00FE    248          CMPB    R1, #^A/?/                      ; <ESC>? is <SS3> on VT52s
                     2E   13   0101    249          BEQL    USE_SS3                         ; Skip if it is
                                0103    250
                                0103    251 ;+
                                0103    252 ; We get here if the character following <ESC> does not make it <CSI>
                                0103    253 ; or <SS3>.  It must then be a single character to look up in ESC_1BYTE
                                0103    254 ;-
                                0103    255
                     53   D7    0103    256          DECL    R3                              ; Look at next character
                     16   12    0105    257          BNEQ    NOTRANS                         ; If not now zero, no translation
                50 FEF5 CF   9E   0107    258          MOVAB   W^ESC_1BYTE, R0                 ; Get table address
                     52   51   D0   010C    259          MOVL    R1, R2                          ; Load R2 with character
                     3E   11   010F    260          BRB     SEARCH_1BYTE                    ; Skip to table search code
                                0111    261
                                0111    262 ;+
                                0111    263 ; We get here if the first character was not <ESC>.  Check for <CSI> or <SS3>.
                                0111    264 ;-
                                0111    265
                                0111    266 NOT_ESC:
                  9B 8F   51   91   0111    267          CMPB    R1, #K_CSI                      ; Is it <CSI>?
                     29   13   0115    268          BEQL    USE_CSI                         ; Skip if it is
                  8F 8F   51   91   0117    269          CMPB    R1, #K_SS3                      ; Is it <SS3>?
                     14   13   011B    270          BEQL    USE_SS3                         ; Skip  f it is
                                011D    271
                                011D    272 ;+
                                011D    273 ; We get here if the terminator has no translation
                                011D    274 ;-
                                011D    275
                                011D    276 NOTRANS:
                50 01FF 8F   3C   011D    277          MOVZWL  #SMG$K_TRM_UNKNOWN, R0          ; Unknown terminator
                     04   0122    278          RET                                     ; Return terminator code
                                0123    279
                                0123    280 ;+
                                0123    281 ; We get here if the terminator is 0 or 1 characters.
                                0123    282 CHECK_SINGLE:
                     50   D6   0123    283          INCL    R0                              ; Check for length of 1
                     04   12   0125    284          BNEQ    BUFFER_FULL                     ; If original count was zero, full buffer
                50   62   9A   0127    285          MOVZBL  (R2), R0                        ; Get terminator byte
                     04   012A    286          RET                                     ; Return with terminator code
                                012B    287 BUFFER_FULL:
                50 01FE 8F   3C   012B    288          MOVZWL  #SMG$K_TRM_BUFFER_FULL, R0 ; Return buffer full code
                     04   0130    289          RET
                                0131    290
                                0131    291 ;+
                                0131    292 ; The terminator starts with <SS3>.
                                0131    293 ;-
                                0131    294
                                0131    295 USE_SS3:
                02   53   D1   0131    296          CMPL    R3, #2                          ; There must be only 1 character
                                0134    297                                          ; following <SS3>.
                     E7   12   0134    298          BNEQ    NOTRANS                         ; No translation if not 1 character
                50 FEDF CF   9E   0136    299          MOVAB   W^CSI_1BYTE, R0                 ; Use same table as <CSI>.
                     52   62   9A   013B    300          MOVZBL  (R2), R2                        ; Load R2 with character
                     0F   11   013E    301          BRB     SEARCH_1BYTE                    ; Use next byte only
                                0140    302
                                0140    303 ;+
```

```
                     0140   304 ; The terminator starts with <CSI>.
                     0140   305 ;-
                     0140   306
                     0140   307 USE_CSI:
        03   53   D1  0140   308         CMPL    R3, #3                  ; Are there 1, 2 or >=3 characters
                     0143   309                                         ; after <CSI>?
             30   14  0143   310         BGTR    SEARCH_3BYTE            ; >=3 more bytes
             17   13  0145   311         BEQL    SEARCH_2BYTE            ; 2 more bytes
   50   FECE CF   9E  0147   312         MOVAB   W^CSI_1BYTE, R0         ; 1 more byte
        52   62   9A  014C   313         MOVZBL  (R2), R2                ; Load R2 with character
                     014F   314
                     014F   315 ;+
                     014F   316 ; Use only the next byte to determine the key.  R0 has been loaded with the
                     014F   317 ; table address and R2 is the byte to look at.
                     014F   318 ;-
                     014F   319
                     014F   320 SEARCH_1BYTE:
        51   80   9A  014F   321         MOVZBL  (R0)+, R1               ; Get byte to compare against
             C9   13  0152   322         BEQL    NOTRANS                 ; End of table?
        52   51   91  0154   323         CMPB    R1, R2                  ; Compare character
             3D   13  0157   324         BEQL    FOUND                   ; End if found
        50   02   C0  0159   325         ADDL2   #2, R0                  ; Skip over key code
             F1   11  015C   326         BRB     SEARCH_1BYTE            ; Repeat until found or table end
                     015E   327
                     015E   328 ;+
                     015E   329 ; Use the next two bytes to determine the key.  (R2) is the word to look at.
                     015E   330 ;-
                     015E   331
                     015E   332 SEARCH_2BYTE:
   50   FEFA CF   9E  015E   333         MOVAB   W^CSI_2BYTE, R0         ; Get table address
        52   62   3C  0163   334         MOVZWL  (R2), R2                ; Get two bytes from terminator
        51   80   3C  0166   335 10$:     MOVZWL  (R0)+, R1               ; Get two bytes to compare against
             B2   13  0169   336         BEQL    NOTRANS                 ; End of table?
        52   51   B1  016B   337         CMPW    R1, R2                  ; Compare characters
             26   13  016E   338         BEQL    FOUND                   ; End if found
        50   02   C0  0170   339         ADDL2   #2, R0                  ; Skip over key code
             F1   11  0173   340         BRB     10$                     ; Repeat until found or table end
                     0175   341
                     0175   342 ;+
                     0175   343 ; Use the next two bytes to determine the key.  (R2) starts the bytes
                     0175   344 ; to look at.  Note that table CSI_3BYTE uses longword entries with the
                     0175   345 ; high byte zero.
                     0175   346 ;-
                     0175   347
                     0175   348 SEARCH_3BYTE:
        04   53   D1  0175   349         CMPL    R3, #4                  ; More than 3 bytes after <CSI>?
             A3   14  0178   350         BGTR    NOTRANS                 ; No translation if so
   50   FEF8 CF   9E  017A   351         MOVAB   W^CSI_3BYTE, R0         ; Get table address
52 62   18   00   EF  017F   352         EXTZV   #0, #24, (R2), R2       ; Get next three bytes
        51   80   D0  0184   353 10$:     MOVL    (R0)+, R1               ; Get four bytes to compare against
             0A   13  0187   354         BEQL    15$                     ; End of table?
        52   51   D1  0189   355         CMPL    R1, R2                  ; Compare characters
             08   13  018C   356         BEQL    FOUND                   ; End if found
        50   02   C0  018E   357         ADDL2   #2, R0                  ; Skip over key code
             F1   11  0191   358         BRB     10$                     ; Repeat until found or table end
                     0193   359
           FF87   31  0193   360 15$:     BRW     NOTRANS
```

```
                     0196   361
                     0196   362 ;+
                     0196   363 ; Translation found.  Store the key code which is in the word addressed by R0
                     0196   364 ;-
                     0196   365
                     0196   366 FOUND:
50   60   3C         0196   367          MOVZWL  (R0), R0                ; Get key code
     04   0199   368          RET
```

```
                        .SBTTL  List of key names

                ;+
                ; KEY_NAME_LIST is a list of all possible key names whose codes are in
                ; the range 256-383.  This excludes control keys.
                ; The format of this list is:
                ;
                ;               key_code-128    - 1 byte
                ;               ASCIC key name  - n bytes
                ;
                ; This format depends on knowing that no key names with codes higher than
                ; 383 are defined, that no character between 128 and 255 can
                ; appear in key names, and that the maximum length of a key name is <32.
                ;
                ; This list is used in two ways:
                ;   1.  To look up a key name (either to see if it is valid or to
                ;       get the corresponding code), do a MATCHC of the ASCIC key name
                ;       into KEY_NAME_LIST.  The byte preceding the found entry is the
                ;       key code minus 128.
                ;
                ;   2.  To convert a key code into a name, do a LOCC of the key code into
                ;       KEY_NAME_LIST.  The ASCIC key name follows the found entry.
                ;-

                ;+
                ; Create macro to add an entry to the list.
                ;-

                        .MACRO  KEY_ENTRY NAME
                        .BYTE   <SMG$K_TRM_'NAME' - 128>
                        .ASCIC  /NAME/
                        .ENDM

                KEY_NAME_LIST:
                        KEY_ENTRY PF1
                        KEY_ENTRY PF2
                        KEY_ENTRY PF3
                        KEY_ENTRY PF4
                        KEY_ENTRY KP0
                        KEY_ENTRY KP1
                        KEY_ENTRY KP2
                        KEY_ENTRY KP3
                        KEY_ENTRY KP4
                        KEY_ENTRY KP5
                        KEY_ENTRY KP6
                        KEY_ENTRY KP7
                        KEY_ENTRY KP8
                        KEY_ENTRY KP9
                        KEY_ENTRY ENTER
                        KEY_ENTRY MINUS
                        KEY_ENTRY COMMA
                        KEY_ENTRY PERIOD
                        KEY_ENTRY UP
                        KEY_ENTRY DOWN
                        KEY_ENTRY LEFT
                        KEY_ENTRY RIGHT
                        KEY_ENTRY F6
```

```
                     0218   427          KEY_ENTRY F7
                     021C   428          KEY_ENTRY F8
                     0220   429          KEY_ENTRY F9
                     0224   430          KEY_ENTRY F10
                     0229   431          KEY_ENTRY F11
                     022E   432          KEY_ENTRY F12
                     0233   433          KEY_ENTRY F13
                     0238   434          KEY_ENTRY F14
                     023D   435          KEY_ENTRY HELP
                     0243   436          KEY_ENTRY DO
                     0247   437          KEY_ENTRY F17
                     024C   438          KEY_ENTRY F18
                     0251   439          KEY_ENTRY F19
                     0256   440          KEY_ENTRY F20
                     025B   441          KEY_ENTRY E1
                     025F   442          KEY_ENTRY E2
                     0263   443          KEY_ENTRY E3
                     0267   444          KEY_ENTRY E4
                     026B   445          KEY_ENTRY E5
                     026F   446          KEY_ENTRY E6
                     0273   447          KEY_ENTRY FIND
                     0279   448          KEY_ENTRY INSERT_HERE
                     0286   449          KEY_ENTRY REMOVE
                     028E   450          KEY_ENTRY SELECT
                     0296   451          KEY_ENTRY PREV_SCREEN
                     02A3   452          KEY_ENTRY NEXT_SCREEN
                     02B0   453
00000116             02B0   454 KEY_NAME_LIST_LEN = .-KEY_NAME_LIST
                     02B0   455
```

```
                     02B0    457          .SBTTL  SMG$$NAME_TO_KEYCODE - Translate key name to key code
                     02B0    458 ;++
                     02B0    459 ; FUNCTIONAL DESCRIPTION:
                     02B0    460 ;
                     02B0    461 ;       SMG$$NAME_TO_KEYCODE translates a key name
                     02B0    462 ;       to a key code.
                     02B0    463 ;
                     02B0    464 ; CALLING SEQUENCE:
                     02B0    465 ;
                     02B0    466 ;       ret_status.wlc.v = SMG$$NAME_TO_KEYCODE (key_name.rt.r, key_code.wl.r)
                     02B0    467 ;
                     02B0    468 ;
                     02B0    469 ; FORMAL PARAMETERS:
                     02B0    470 ;
00000004             02B0    471          key_name = 4    ; The address of a counted string (byte count)
                     02B0    472                          ; containing the name of the key to look up.  It
                     02B0    473                          ; is assumed that the string has been upcased and
                     02B0    474                          ; stripped of trailing blanks.
                     02B0    475
00000008             02B0    476          key_code = 8    ; A longword into which is stored the key code.
                     02B0    477 ;
                     02B0    478 ; IMPLICIT INPUTS:
                     02B0    479 ;
                     02B0    480 ;       NONE
                     02B0    481 ;
                     02B0    482 ; IMPLICIT OUTPUTS:
                     02B0    483 ;
                     02B0    484 ;       NONE
                     02B0    485 ;
                     02B0    486 ; COMPLETION STATUS:
                     02B0    487 ;
                     02B0    488 ;       1 - Key found
                     02B0    489 ;       0 - Key not found
                     02B0    490 ;
                     02B0    491 ; SIDE EFFECTS:
                     02B0    492 ;
                     02B0    493 ;       NONE
                     02B0    494 ;
                     02B0    495 ;--
                     02B0    496
                 003C 02B0    497          .ENTRY  SMG$$NAME_TO_KEYCODE, ^M<R2,R3,R4,R5>
          54  04 AC   D0 02B2    498          MOVL    key_name(AP), R4        ; Get pointer to ASCIC key name
              55  64   9A 02B6    499          MOVZBL  (R4), R5                ; Get length in R5, data in (R4)
                  55   D6 02B9    500          INCL    R5                      ; Add one for the count
                  1C   13 02BB    501          BEQL    10$                     ; Exit if zero length
                06 55   D1 02BD    502          CMPL    R5, #6                  ; Could it be CTRLx?
                  1F   12 02C0    503          BNEQ    20$                     ; Skip if not
     4C525443 8F  01 A4   D1 02C2    504          CMPL    1(R4), #^A/CTRL/        ; Does it start with CTRL?
                  15   12 02CA    505          BNEQ    20$                     ; Skip if not
      50  05 A4  40 8F   83 02CC    506          SUBB3   #^X40, 5(R4), R0        ; Convert 5th character to code?
                  05   15 02D2    507          BLEQ    10$                     ; Skip if LSS "A"
                1A 50   D1 02D4    508          CMPL    R0, #K_CTRLZ            ; LEQ CTRLZ?
                  21   1B 02D7    509          BLEQU   30$                     ; Skip if not
        50  00000000'8F   D0 02D9    510 10$:     MOVL    #SMG$_INVKEYNAM, R0     ; Indicate invalid key name
                  04   02E0    511          RET
                       02E1    512
FEB0 CF  0116 8F  64 55   39 02E1    513 20$:     MATCHC  R5, (R4), #KEY_NAME_LIST_LEN, W^KEY_NAME_LIST
```

```
                                02EA    514
                                02EA    515             BNEQ    10$                             ; Look up key name
                             ED 12      02EA    516             SUBL2   R5, R3                  ; Skip if not 'ound
                          53 55 C2      02EC    517             MOVZBL  -1(R3), R0              ; Get back to first name byte
                       50 FF A3 9A      02EF    518             ADDL2   #128, R0                ; Get code (minus 128)
                50 00000080 8F C0       02F3    519 30$:        MOVL    R0, @key_code(AP)       ; Re-bias code
                   08 BC 50 D0          02FA    520             MOVL    #1, R0                  ; Store key code
                      50 01 D0          02FE    521             RET                             ; Return success
                            04          0301    522
                                        0302
```

```
                     0302   524          .SBTTL  SMG$$KEYCODE_TO_NAME - Translate key code to key name
                     0302   525 ;++
                     0302   526 ; FUNCTIONAL DESCRIPTION:
                     0302   527 ;
                     0302   528 ;       SMG$$KEYCODE_TO_NAME translates a key code
                     0302   529 ;       to a key name.
                     0302   530 ;
                     0302   531 ; CALLING SEQUENCE:
                     0302   532 ;
                     0302   533 ;       ret_status.wlc.v = SMG$$KEYCODE_TO_NAME (key_code.rl.v,
                     0302   534 ;                                               key_name_ptr.wa.r)
                     0302   535 ;
                     0302   536 ;
                     0302   537 ; FORMAL PARAMETERS:
                     0302   538 ;
00000004             0302   539 ;       key_code = 4    ; A longword containing the key code to be converted.
                     0302   540
00000008             0302   541 ;       key_name_ptr = 8; Address of an area to store an ASCIC key name.
                     0302   542
                     0302   543 ;
                     0302   544 ; IMPLICIT INPUTS:
                     0302   545 ;
                     0302   546 ;       NONE
                     0302   547 ;
                     0302   548 ; IMPLICIT OUTPUTS:
                     0302   549 ;
                     0302   550 ;       NONE
                     0302   551 ;
                     0302   552 ; COMPLETION STATUS:
                     0302   553 ;
                     0302   554 ;       1 - Key found
                     0302   555 ;       0 - Key not found
                     0302   556 ;
                     0302   557 ; SIDE EFFECTS:
                     0302   558 ;
                     0302   559 ;       NONE
                     0302   560 ;
                     0302   561 ;--
                     0302   562
                 003C 0302   563          .ENTRY  SMG$$KEYCODE_TO_NAME, ^M<R2,R3,R4,R5>
          54   04 AC   D0 0304   564          MOVL    key_code(AP), R4        ; Get key code
                   21   13 0308   565          BEQL    15$                     ; Not a known code
              1A   54   D1 030A   566          CMPL    R4, #26                 ; Is it CTRLA through CTRLZ?
                   13   1A 030D   567          BGTRU   10$                     ; Skip if not
        7E    54   40 8F   81 030F   568          ADDB3   #^X40, R4, -(SP)        ; Build CTRLx on stack (x=A-Z)
          4C525443 8F   DD 0314   569          PUSHL   #^A/CTRL/
              7E   05   90 031A   570          MOVB    #5, -(SP)               ; Push count
              51   5E   D0 031D   571          MOVL    SP, R1                  ; Put pointer to name in R1
                   28   11 0320   572          BRB     CODE_FOUND              ; Join common code
       00000100 8F   54   D1 0322   573 10$:    CMPL    R4, #256                ; Is it a keypad code?
                   03   1E 0329   574          BGEQU   20$                     ; Skip if so
                   50   D4 032B   575 15$:    CLRL    R0                      ; Key code not found
                   04   D0 032D   576          RET
       54   00000080 8F   C2 032E   577 20$:    SUBL2   #128, R4                ; Compensate for bias in table
       00000100 8F   54   D1 0335   578          CMPL    R4, #256                ; Not a valid code?
                   ED   1E 033C   579          BGEQU   15$                     ; Skip if not
FE54 CF    0116 8F   54   3A 033E   580          LOCC    R4, #KEY_NAME_LIST_LEN, W^KEY_NAME_LIST ; Look for code
```

```
                     E3   13   0346   581          BEQL    15$
                     51   D6   0348   582          INCL    R1                          ; R1 now points to key name
                                034A   583 CODE_FOUND:
                50   61   9A   034A   584          MOVZBL  (R1), R0                    ; Get length of string
                     50   D6   034D   585          INCL    R0                          ; Add one for count
          08 BC      61   50   28   034F   586          MOVC3   R0, (R1), @key_name_ptr(AP) ; Move key name
                     50   01   D0   0354   587          MOVL    #1, R0                      ; Success
                               04   0357   588          RET
                                0358   589
```

```
0358    591          .END                                ; End of module SMG$$KEY_UTIL
```

```
BUFFER_FULL                     0000012B R      02
CHECK_SINGLE                    00000123 R      02
CODE_FOUND                      0000034A R      02
CSI_1BYTE                       00000019 R      02
CSI_2BYTE                       0000005C R      02
CSI_3BYTE                       0000007b R      02
ESC_1BYTE                       00000000 R      02
FOUND                           00000196 R      02
KEY_CODE                      = 00000004
KEY_NAME                      = 00000004
KEY_NAME_LIST                   0000019A R      02
KEY_NAME_LIST_LEN             = 00000116
KEY_NAME_PTR                  = 00000008
K_CSI                         = 0000009B
K_CTRLZ                       = 0000001A
K_ESC                         = 0000001B
K_SS3                         = 0000008F
NOTRANS                         0000011D R      02
NOT_ESC                         00000111 R      02
SEARCH_1BYTE                    0000014F R      02
SEARCH_2BYTE                    0000015E R      02
SEARCH_3BYTE                    00000175 R      02
SMG$$KEYCODE_TO_NAME            00000302 RG     02
SMG$$NAME_TO_KEYCODE            000002B0 RG     02
SMG$$TERM_TO_KEYCODE            000000D4 RG     02
SMG$K_TRM_BUFFER_FULL         = 000001FE
SMG$K_TRM_COMMA               = 00000110
SMG$K_TRM_DO                  = 00000128
SMG$K_TRM_DOWN                = 00000113
SMG$K_TRM_E1                  = 00000137
SMG$K_TRM_E2                  = 00000138
SMG$K_TRM_E3                  = 00000139
SMG$K_TRM_E4                  = 0000013A
SMG$K_TRM_E5                  = 0000013B
SMG$K_TRM_E6                  = 0000013C
SMG$K_TRM_ENTER               = 0000010E
SMG$K_TRM_F10                 = 00000122
SMG$K_TRM_F11                 = 00000123
SMG$K_TRM_F12                 = 00000124
SMG$K_TRM_F13                 = 00000125
SMG$K_TRM_F14                 = 00000126
SMG$K_TRM_F17                 = 00000129
SMG$K_TRM_F18                 = 0000012A
SMG$K_TRM_F19                 = 0000012B
SMG$K_TRM_F20                 = 0000012C
SMG$K_TRM_F6                  = 0000011E
SMG$K_TRM_F7                  = 0000011F
SMG$K_TRM_F8                  = 00000120
SMG$K_TRM_F9                  = 00000121
SMG$K_TRM_FIND                = 00000137
SMG$K_TRM_HELP                = 00000127
SMG$K_TRM_INSERT_HERE         = 00000138
SMG$K_TRM_KP0                 = 00000104
SMG$K_TRM_KP1                 = 00000105
SMG$K_TRM_KP2                 = 00000106
SMG$K_TRM_KP3                 = 00000107
SMG$K_TRM_KP4                 = 00000108
SMG$K_TRM_KP5                 = 00000109
SMG$K_TRM_KP6                 = 0000010A
SMG$K_TRM_KP7                 = 0000010B
SMG$K_TRM_KP8                 = 0000010C
SMG$K_TRM_KP9                 = 0000010D
SMG$K_TRM_LEFT                = 00000114
SMG$K_TRM_MINUS               = 0000010F
SMG$K_TRM_NEXT_SCREEN         = 0000013C
SMG$K_TRM_PERIOD              = 00000111
SMG$K_TRM_PF1                 = 00000100
SMG$K_TRM_PF2                 = 00000101
SMG$K_TRM_PF3                 = 00000102
SMG$K_TRM_PF4                 = 00000103
SMG$K_TRM_PREV_SCREEN         = 0000013B
SMG$K_TRM_REMOVE              = 00000139
SMG$K_TRM_RIGHT               = 00000115
SMG$K_TRM_SELECT              = 0000013A
SMG$K_TRM_UNKNOWN             = 000001FF
SMG$K_TRM_UP                  = 00000112
SMG$_INVKEYNAM                  ********    X   00
TERMINATOR                    = 00000004
USE_CSI                         00000140 R      02
USE_SS3                         00000131 R      02
```

Psect synopsis

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000 (0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
_SMG$CODE	00000358 (856.)	02 (2.)	PIC USR CON REL LCL SHR EXE RD NOWRT NOVEC LONG

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	13	00:00:00.03	00:00:01.33
Command processing	88	00:00:00.55	00:00:07.18
Pass 1	147	00:00:04.09	00:00:20.06
Symbol table sort	0	00:00:00.28	00:00:00.80
Pass 2	98	00:00:01.43	00:00:07.42
Symbol table output	9	00:00:00.07	00:00:00.07
Psect synopsis output	3	00:00:00.04	00:00:00.08
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	358	00:00:06.49	00:00:36.95

The working set limit was 1050 pages.
22486 bytes (44 pages) of virtual memory were used to buffer the intermediate code.
There were 20 pages of symbol table space allocated to hold 233 non-local and 9 local symbols.
591 source lines were read in Pass 1, producing 21 object records in Pass 2.
12 pages of virtual memory were used to define 11 macros.

Macro library statistics

Macro library name	Macros defined
_255DUA28:[SMGRTL.OBJ]SMGRTL.MLB;1	1
_255DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	4

262 GETS were required to define 4 macros.

There were no errors, warnings or information messages.

```
MACRO/ENABLE=SUPPRESSION/DISABLE=(GLOBAL,TRACEBACK)/LIS=LIS$:SMGKEYUTI/OBJ=OBJ$:SMGKEYUTI MSRC$:SMGKEYUTI/UPDATE=(ENH$:SMGKEYUTI)+LI
```

SMGKEYUTI
LIS

SMGKEYPAD
LIS

SMGINPUT
LIS

SMGLIB
LIS