G 6

```
LLL                  IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR        AAAAAAAAA     RRRRRRRRRRRR
LLL                  IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR        AAAAAAAAA     RRRRRRRRRRRR
LLL                  IIIIIIIII     BBBBBBBBBBBB     RRRRRRRRRRRR        AAAAAAAAA     RRRRRRRRRRRR
LLL                     III        BBB         BBB  RRR         RRR  AAA         AAA  RRR         RRR
LLL                     III        BBB         BBB  RRR         RRR  AAA         AAA  RRR         RRR
LLL                     III        BBB         BBB  RRR         RRR  AAA         AAA  RRR         RRR
LLL                     III        BBB         BBB  RRR         RRR  AAA         AAA  RRR         RRR
LLL                     III        BBB         BBB  RRR         RRR  AAA         AAA  RRR         RRR
LLL                     III        BBB         BBB  RRR         RRR  AAA         AAA  RRR         RRR
LLL                     III        BBBBBBBBBBBB     RRRRRRRRRRRR     AAA         AAA  RRRRRRRRRRRR
LLL                     III        BBBBBBBBBBBB     RRRRRRRRRRRR     AAA         AAA  RRRRRRRRRRRR
LLL                     III        BBBBBBBBBBBB     RRRRRRRRRRRR     AAA         AAA  RRRRRRRRRRRR
LLL                     III        BBB         BBB  RRR   RRR        AAAAAAAAAAAAAAA  RRR   RRR
LLL                     III        BBB         BBB  RRR   RRR        AAAAAAAAAAAAAAA  RRR   RRR
LLL                     III        BBB         BBB  RRR   RRR        AAAAAAAAAAAAAAA  RRR   RRR
LLL                     III        BBB         BBB  RRR      RRR     AAA         AAA  RRR      RRR
LLL                     III        BBB         BBB  RRR      RRR     AAA         AAA  RRR      RRR
LLL                     III        BBB         BBB  RRR      RRR     AAA         AAA  RRR      RRR
LLLLLLLLLLLLLLL      IIIIIIIII     BBBBBBBBBBBB     RRR         RRR  AAA         AAA  RRR         RRR
LLLLLLLLLLLLLLL      IIIIIIIII     BBBBBBBBBBBB     RRR         RRR  AAA         AAA  RRR         RRR
LLLLLLLLLLLLLLL      IIIIIIIII     BBBBBBBBBBBB     RRR         RRR  AAA         AAA  RRR         RRR
```

```
IIIIII    NN       NN  PPPPPPPP    UU       UU  TTTTTTTTTT  HH       HH  LL          PPPPPPPP
IIIIII    NN       NN  PPPPPPPP    UU       UU  TTTTTTTTTT  HH       HH  LL          PPPPPPPP
  II      NN       NN  PP      PP  UU       UU      TT      HH       HH  LL          PP      PP
  II      NN       NN  PP      PP  UU       UU      TT      HH       HH  LL          PP      PP
  II      NNNN     NN  PP      PP  UU       UU      TT      HH       HH  LL          PP      PP
  II      NNNN     NN  PP      PP  UU       UU      TT      HH       HH  LL          PP      PP
  II      NN NN    NN  PPPPPPPP    UU       UU      TT      HHHHHHHHHHH  LL          PPPPPPPP
  II      NN NN    NN  PPPPPPPP    UU       UU      TT      HHHHHHHHHHH  LL          PPPPPPPP
  II      NN  NNNN PP  PP          UU       UU      TT      HH       HH  LL          PP
  II      NN  NNNN PP  PP          UU       UU      TT      HH       HH  LL          PP
  II      NN    NN PP  PP          UU       UU      TT      HH       HH  LL          PP          ....
  II      NN    NN PP  PP          UU       UU      TT      HH       HH  LL          PP          ....
IIIIII    NN    NN PP  PP          UUUUUUUUUUU      TT      HH       HH  LLLLLLLLLL  PP          ....
IIIIII    NN    NN PP  PP          UUUUUUUUUUU      TT      HH       HH  LLLLLLLLLL  PP          ....

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

```
;     1          0001  0 MODULE lib_inputhlp (                       ! Get next line of Help text
;     2          0002  0                 LANGUAGE (BLISS32),
;     3          0003  0                 IDENT = 'V04-000'
;     4          0004  0                 ) =
;     5          0005  1 BEGIN
;     6          0006  1
;     7          0007  1 !
;     8          0008  1 !**************************************************************************
;     9          0009  1 !*                                                                        *
;    10          0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                               *
;    11          0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
;    12          0012  1 !*  ALL RIGHTS RESERVED.                                                   *
;    13          0013  1 !*                                                                        *
;    14          0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    15          0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
;    16          0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17          0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18          0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19          0019  1 !*  TRANSFERRED.                                                           *
;    20          0020  1 !*                                                                        *
;    21          0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22          0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23          0023  1 !*  CORPORATION.                                                           *
;    24          0024  1 !*                                                                        *
;    25          0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26          0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
;    27          0027  1 !*                                                                        *
;    28          0028  1 !*                                                                        *
;    29          0029  1 !**************************************************************************
;    30          0030  1
;    31          0031  1 !++
;    32          0032  1 !
;    33          0033  1 ! FACILITY:  Library command processor
;    34          0034  1 !
;    35          0035  1 ! ABSTRACT:
;    36          0036  1 !
;    37          0037  1 !       The VAX/VMS librarian is invoked by DCL to process the LIBRARY
;    38          0038  1 !       command.  It utilizes the librarian procedure set to perform
;    39          0039  1 !       the actual modifications to the library.
;    40          0040  1 !
;    41          0041  1 ! ENVIRONMENT:
;    42          0042  1 !
;    43          0043  1 !       VAX native, user mode.
;    44          0044  1 !
;    45          0045  1 !--
;    46          0046  1 !
;    47          0047  1 !
;    48          0048  1 ! AUTHOR:  Benn Schreiber,       CREATION DATE:  20-Aug-1979
;    49          0049  1 !
;    50          0050  1 ! MODIFIED BY:
;    51          0051  1 !
;    52          0052  1 !       V03-005 GJA0087         Greg Awdziewicz         18-May-1984
;    53          0053  1 !               - (See v03-003) Allow the quotation marks (x22).
;    54          0054  1 !               Prohibiting them had broken the PL/I help library.
;    55          0055  1 !
;    56          0056  1 !       V03-004 GJA0080         Greg Awdziewicz         7-Apr-1984
;    57          0057  1 !               - (See v03-003) Allow the period (x2E).
```

```
;    58          0058  1 !
;    59          0059  1 !       V03-003 GJA0076         Greg Awdziewicz         11-Mar-1984
;    60          0060  1 !               - (See v03-002) Also exclude the quotation marks (x22),
;    61          0061  1 !               the comma (x2C), and the period (x2E) and their associated
;    62          0062  1 !               8 bit siblings: xA2, xAC, and xAE.
;    63          0063  1 !
;    64          0064  1 !       V03-002 GJA0067         Greg Awdziewicz         22-Feb-1984
;    65          0065  1 !               - Include the first character of a key in the check for
;    66          0066  1 !               valid symbols.
;    67          0067  1 !               - Expand the valid symbol set for help keys to be all
;    68          0068  1 !               printing characters (including the DEC Supplemental
;    69          0069  1 !               Graphic Set of the DEC Multinational Character Set),
;    70          0070  1 !               excluding the space (x20), the exclamation mark (x21),
;    71          0071  1 !               the del (x7F), and their 8 bit  siblings: xA0, xA1, and xFF.
;    72          0072  1 !               All control characters continue to be invalid,
;    73          0073  1 !               although the horizontal tab is considered to be a
;    74          0074  1 !               delimiter, just like a space.
;    75          0075  1 !               - Remove the unused routine Make_upper_case.
;    76          0076  1 !
;    77          0077  1 !       V03-001 JWT0089         Jim Teague              13-Jan-1983
;    78          0078  1 !               Clear up 9th level HELP problem.
;    79          0079  1 !
;    80          0080  1 !--
;    81          0081  1
;    82          0082  1
```

```
;    84          0083  1 LIBRARY
;    85          0084  1         'SYS$LIBRARY:STARLET.L32';
;    86          0085  1 REQUIRE
;    87          0086  1         'PREFIX';
;    88          0270  1 REQUIRE
;    89          0271  1         'LIBDEF';
;    90          0559  1 REQUIRE
;    91          0560  1         'LBRDEF';
;    92          1151  1
;    93          1152  1 EXTERNAL ROUTINE
;    94          1153  1     lib_log_op,                                  !log operation on console
;    95          1154  1     lib_log_upd,                                 !Record module names for LUH
;    96          1155  1     get_record,                                  !Read next input record
;    97          1156  1     lib$cvt_dtb : ADDRESSING_MODE (GENERAL),     !Convert decimal to binary
;    98          1157  1     lbr$put_record : ADDRESSING_MODE (GENERAL),  !Write record to library
;    99          1158  1     lbr$set_module : ADDRESSING_MODE (GENERAL),  !Read/update module header
;   100          1159  1     lbr$put_end : ADDRESSING_MODE (GENERAL),     !Terminate PUT
;   101          1160  1     lbr$lookup_key : ADDRESSING_MODE (GENERAL),  !Lookup key in library
;   102          1161  1     lbr$insert_key : ADDRESSING_MODE (GENERAL),  !Insert key into library
;   103          1162  1     lbr$delete_key : ADDRESSING_MODE (GENERAL),  !Delete key
;   104          1163  1     lbr$delete_data : ADDRESSING_MODE (GENERAL), !Delete data
;   105          1164  1     lbr$replace_key : ADDRESSING_MODE (GENERAL);!Replace or insert key in index
;   106          1165  1
;   107          1166  1 EXTERNAL
;   108          1167  1     lbr$gl_rmsstv : ADDRESSING_MODE (GENERAL),   !RMS STV from library procedures
;   109          1168  1     lib$gl_keysize,                              !Max length of keys
;   110          1169  1     lib$gl_libctl : BLOCK [2],                   !Library control index
;   111          1170  1     lib$gl_ctlmsk : BLOCK [1],                   !control flags
;   112          1171  1     lib$gl_inpfdb : REF BBLOCK,                  !Input file FDB
;   113          1172  1     lib$gl_libfdb : REF BBLOCK;                  !Library file FDB
;   114          1173  1
;   115          1174  1 EXTERNAL LITERAL
;   116          1175  1     lbr$_dupkey,                                 !Duplicate module
;   117          1176  1     lib$_invkeychar,                             ! Invalid character in key.
;   118          1177  1     lib$_inserterr,                              !Error inserting
;   119          1178  1     lib$_deldaterr,                              !Delete data error
;   120          1179  1     lib$_inserted,                               !Successfully inserted
;   121          1180  1     lib$_replaced,                               !Successfully replaced
;   122          1181  1     lib$_illkeylvl,                              !Illegal key level
;   123          1182  1     lib$_keynamlng,                              !Illegal key length
;   124          1183  1     lib$_dupmod,                                 !Duplicate module
;   125          1184  1     lib$_dupmodule,                              !Duplicate module warning
;   126          1185  1     lib$_nohlptxt;                               !No help text found
;   127          1186  1
;   128          1187  1 LITERAL
;   129          1188  1     CR  = 13,                                    ! Carriage return
;   130          1189  1     LF  = 10;                                    ! Line feed
;   131          1190  1
;   132          1191  1 FORWARD ROUTINE
;   133          1192  1     is_key_on_line,                              !determine if key on line
;   134          1193  1     symbol_char,                                 !Determine if character is a symbol
;   135          1194  1     scan_word,                                   !Scan a word
;   136          1195  1     skip_blanks;                                 !Skip blanks
;   137          1196  1 OWN
;   138          1197  1     lineptr,                                     !Pointer into line
;   139          1198  1     endptr,                                      !Pointer to end of line
;   140          1199  1     curchar,                                     !Current character
```

```
;  141          1200  1      linedesc : BBLOCK [dsc$c_s_bln];            !String descriptor for input line
;  142          1201  1
;  143          1202  1 BIND
;  144          1203  1      linelen = linedesc [dsc$w_length] : WORD,    .Length of line read
;  145          1204  1      lineaddr = linedesc [dsc$a_pointer] : REF VECTOR [,BYTE]; .And address of line
```

```
;  147          1205  1 GLOBAL ROUTINE lib_input_hlp  =
;  148          1206  2 BEGIN
;  149          1207  2
;  150          1208  2 !++
;  151          1209  2 ! FUNCTIONAL DESCRIPTION:
;  152          1210  2 !
;  153          1211  2 !       This routine inserts help text modules into a help library.
;  154          1212  2 !
;  155          1213  2 !
;  156          1214  2 !--
;  157          1215  2 ROUTINE put_record (linedesc, txtrfa) =
;  158          1216  3 BEGIN
;  159          1217  3
;  160          1218  3 !++
;  161          1219  3 ! Local routine to write record to library
;  162          1220  3 !
;  163          1221  3 !--
;  164          1222  3
;  165          1223  3
;  166        P 1224  3 rms_perform (lbr$put_record (lib$gl_libctl, .linedesc, .txtrfa),
;  167          1225  3                     lib$_writeerr, .lbr$gl_rmsstv, 1, lib$gl_libfdb [fdb$l_namdesc]);
;  168          1226  3 RETURN true
;  169          1227  2 END;


                                                        .TITLE  LIB_INPUTHLP
                                                        .IDENT  \V04-000\

                                                        .PSECT  $OWN$,NOEXE,2

                                                00000 LINEPTR:.BLKB   4
                                                00004 ENDPTR: .BLKB   4
                                                00008 CURCHAR:.BLKB   4
                                                0000C LINEDESC:
                                                        .BLKB   8

                                                LINELEN=            LINEDESC
                                                LINEADDR=           LINEDESC+4
                                                        .EXTRN  LIB_LOG_OP, LIB_LOG_UPD
                                                        .EXTRN  GET_RECORD, LIB$CVT_DTB
                                                        .EXTRN  LBR$PUT_RECORD, LBR$SET_MODULE
                                                        .EXTRN  LBR$PUT_END, LBR$LOOKUP_KEY
                                                        .EXTRN  LBR$INSERT_KEY, LBR$DELETE_KEY
                                                        .EXTRN  LBR$DELETE_DATA
                                                        .EXTRN  LBR$REPLACE_KEY
                                                        .EXTRN  LBR$GL_RMSSTV, LIB$GL_KEYSIZE
                                                        .EXTRN  LIB$GL_LIBCTL, LIB$GL_CTLMSK
                                                        .EXTRN  LIB$GL_INPFDB, LIB$GL_LIBFDB
                                                        .EXTRN  LBR$_DUPKEY, LIB$_INVKEYCHAR
                                                        .EXTRN  LIB$_INSERTERR, LIB$_DELDATERR
                                                        .EXTRN  LIB$_INSERTED, LIB$_REPLACED
                                                        .EXTRN  LIB$_ILLKEYLVL, LIB$_KEYNAMLNG
                                                        .EXTRN  LIB$_DUPMOD, LIB$_DUPMODULE
                                                        .EXTRN  LIB$_NOHLPTXT

                                                        .PSECT  $CODE$,NOWRT,2
```

```
                                0000 00000 PUT_RECORD:
                                                .WORD   Save nothing                                    ; 1216
                        7E      04      AC  7D 00002            MOVQ    LINEDESC, -(SP)                 ; 1225
                              0000G     CF  9F 00006            PUSHAB  LIB$GL_LIBCTL
                00000000G     00        03  FB 0000A            CALLS   #3, LBR$PUT_RECORD
                              1D        50  E8 00011            BLBS    STATUS, 1$
                                00000000G   00  DD 00014        PUSHL   LBR$GL_RMSSTV
                                            50  DD 0001A        PUSHL   STATUS
                7E      0000G   CF          10  C1 0001C        ADDL3   #16, LIB$GL_LIBFDB, -(SP)
                                            01  DD 00022        PUSHL   #1
                                00861002    8F  DD 00024        PUSHL   #8786130
                00000000G       00          05  FB 0002A        CALLS   #5, LIB$SIGNAL
                                50          01  D0 00031 1$:    MOVL    #1, R0                          ; 1226
                                                04 00034        RET                                     ; 1227

; Routine Size:  53 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  171          1228  2 ROUTINE cleanup (txtrfa) =
;  172          1229  3 BEGIN
;  173          1230  3
;  174          1231  3 !++
;  175          1232  3 !
;  176          1233  3 !       Clean up written text, module is no good.
;  177          1234  3 !
;  178          1235  3 !--
;  179          1236  3
;  180          1237  3 MAP
;  181          1238  3     txtrfa : REF BBLOCK;
;  182          1239  3
;  183          1240  3 IF .txtrfa [rfa$l_vbn] NEQ 0
;  184          1241  4 THEN BEGIN
;  185          1242  4     lbr$put_end (lib$gl_libctl);                    !Write end of module record
;  186          1243  4     lbr$delete_data (lib$gl_libctl, .txtrfa);       ! and delete the works
;  187          1244  3     END;
;  188          1245  3
;  189          1246  3 RETURN true
;  190          1247  2 END;


                                    0000 00000 CLEANUP:.WORD   Save nothing                    ; 1228
                                04   BC   D5 00002          TSTL    @TXTRFA                    ; 1240
                                     19   13 00005          BEQL    1$                         ;
                                0000G CF  9F 00007          PUSHAB  LIB$GL_LIBCTL              ; 1242
                          00000000G  00     01 FB 0000B     CALLS   #1, LBR$PUT_END            ;
                                04   AC   DD 00012          PUSHL   TXTRFA                     ; 1243
                                0000G CF  9F 00015          PUSHAB  LIB$GL_LIBCTL              ;
                          00000000G  00     02 FB 00019     CALLS   #2, LBR$DELETE_DATA        ;
                                50          01 D0 00020 1$: MOVL    #1, R0                     ; 1246
                                            04 00023        RET                                ; 1247

; Routine Size:  36 bytes,    Routine Base:  $CODE$ + 0035
```

```
;  192          1248  2 ROUTINE insertkey1 (keydesc, txtrfa, replacing, deltxtrfa) =
;  193          1249  3 BEGIN
;  194          1250  3
;  195          1251  3 !++
;  196          1252  3 ! Local routine to insert or replace a key1 in the index.
;  197          1253  3 !
;  198          1254  3 !--
;  199          1255  3
;  200          1256  3 MAP
;  201          1257  3     keydesc : REF BBLOCK,
;  202          1258  3     txtrfa : REF BBLOCK,
;  203          1259  3     deltxtrfa : REF BBLOCK;
;  204          1260  3
;  205          1261  3 !
;  206          1262  3 ! Determine if replacing or inserting this module
;  207          1263  3 !
;  208          1264  3
;  209          1265  3 IF .lib$gl_ctlmsk [lib$v_replace]
;  210          1266  4     THEN BEGIN
;  211        P 1267  4         rms_perform (lbr$replace_key (lib$gl_libctl, .keydesc, .deltxtrfa, .txtrfa),
;  212          1268  4                      lib$_inserterr, .lbr$gl_rmsstv, 2, keydesc, lib$gl_libfdb [fdb$l_namdesc]);
;  213          1269  4         IF .replacing
;  214        P 1270  4             THEN rms_perform (lbr$delete_data (lib$gl_libctl, .deltxtrfa),
;  215          1271  4                               lib$_deldaterr, .lbr$gl_rmsstv, 1, lib$gl_libfdb [fdb$l_namdesc]);
;  216          1272  4         END
;  217          1273  4
;  218          1274  3     ELSE
;  219          1275  4         BEGIN
;  220          1276  4         LOCAL
;  221          1277  4             status;
;  222          1278  4         status = lbr$insert_key (lib$gl_libctl, .keydesc, .txtrfa);
;  223          1279  4         IF NOT .status
;  224          1280  4         THEN
;  225          1281  5             BEGIN
;  226          1282  5             IF .status NEQ lbr$_dupkey
;  227          1283  5             THEN SIGNAL ( lib$_inserterr, 2, keydesc, lib$gl_libfdb [fdb$l_namdesc], .status, .lbr$gl_rmsstv
;  228          1284  5             ELSE
;  229          1285  6                 BEGIN   ! duplicate module, cleanup and continue
;  230          1286  6                 RETURN cleanup (txtrfa);
;  231          1287  5                 END;
;  232          1288  4             END;
;  233          1289  3         END;
;  234          1290  3
;  235          1291  4 lib_log_upd ((IF .replacing THEN lhe$c_replaced
;  236          1292  3                               ELSE lhe$c_inserted), .keydesc );
;  237          1293  4 lib_log_op ((IF .replacing THEN lib$_replaced
;  238          1294  3                               ELSE lib$_inserted), .keydesc, .lib$gl_libfdb);
;  239          1295  3
;  240          1296  3 CH$FILL (0, rfa$c_length, txtrfa [rfa$l_vbn]);
;  241          1297  3
;  242          1298  3 RETURN true
;  243          1299  2 END;
```

```
                                  07FC 00000 INSERTKEY1:
                                                     .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10     ; 1248
              5A      0000G   CF  9E 00002           MOVAB   LIB$GL_LIBCTL, R10                   ;
              59  00000000G   8F  D0 00007           MOVL    #LIB$_INSERTERR, R9                  ;
              58      0000G   CF  9E 0000E           MOVAB   LIB$GL_LIBFDB, R8                    ;
              57  00000000G   00  9E 00013           MOVAB   LIB$SIGNAL, R7                       ;
              56  00000000G   00  9E 0001A           MOVAB   LBR$GL_RMSSTV, R6                    ;
   4F      0000G  CF          05  E1 00021           BBC     #5, LIB$GL_CTLMSK+1, 2$              ; 1265
                         08   AC  DD 00027           PUSHL   TXTRFA                               ; 1268
                         10   AC  DD 0002A           PUSHL   DELTXTRFA                            ;
                         04   AC  DD 0002D           PUSHL   KEYDESC                              ;
                              5A  DD 00030           PUSHL   R10                                  ;
       00000000G  00          04  FB 00032           CALLS   #4, LBR$REPLACE_KEY                  ;
                  12          50  E8 00039           BLBS    STATUS, 1$                           ;
                              66  DD 0003C           PUSHL   LBR$GL_RMSSTV                        ;
                              50  DD 0003E           PUSHL   STATUS                               ;
   7E             68          10  C1 00040           ADDL3   #16, LIB$GL_LIBFDB, -(SP)            ;
                         04   AC  9F 00044           PUSHAB  KEYDESC                              ;
                              02  DD 00047           PUSHL   #2                                   ;
                              59  DD 00049           PUSHL   R9                                   ;
                  67          06  FB 0004B           CALLS   #6, LIB$SIGNAL                       ;
                  5A     0C   AC  E9 0004E 1$:       BLBC    REPLACING, 4$                        ; 1269
                         10   AC  DD 00052           PUSHL   DELTXTRFA                            ; 1271
                              5A  DD 00055           PUSHL   R10                                  ;
       00000000G  00          02  FB 00057           CALLS   #2, LBR$DELETE_DATA                  ;
                  4B          50  E8 0005E           BLBS    STATUS, 4$                           ;
                              66  DD 00061           PUSHL   LBR$GL_RMSSTV                        ;
                              50  DD 00063           PUSHL   STATUS                               ;
   7E             68          10  C1 00065           ADDL3   #16, LIB$GL_LIBFDB, -(SP)            ;
                              01  DD 00069           PUSHL   #1                                   ;
                  00000000G   8F  DD 0006B           PUSHL   #LIB$_DELDATERR                      ;
                  67          05  FB 00071           CALLS   #5, LIB$SIGNAL                       ;
                              36  11 00074           BRB     4$                                   ; 1265
                  7E     04   AC  7D 00076 2$:       MOVQ    KEYDESC, -(SP)                       ; 1278
                              5A  DD 0007A           PUSHL   R10                                  ;
       00000000G  00          03  FB 0007C           CALLS   #3, LBR$INSERT_KEY                   ;
                  26          50  E8 00083           BLBS    STATUS, 4$                           ; 1279
       00000000G  8F          50  D1 00086           CMPL    STATUS, #LBR$_DUPKEY                 ; 1282
                              14  13 0008D           BEQL    3$                                   ;
                              66  DD 0008F           PUSHL   LBR$GL_RMSSTV                        ; 1283
                              50  DD 00091           PUSHL   STATUS                               ;
   7E             68          10  C1 00093           ADDL3   #16, LIB$GL_LIBFDB, -(SP)            ;
                         04   AC  9F 00097           PUSHAB  KEYDESC                              ;
                              02  DD 0009A           PUSHL   #2                                   ;
                              59  DD 0009C           PUSHL   R9                                   ;
                  67          06  FB 0009E           CALLS   #6, LIB$SIGNAL                       ;
                              09  11 000A1           BRB     4$                                   ;
                         08   AC  9F 000A3 3$:       PUSHAB  TXTRFA                               ; 1286
           FF31   CF          01  FB 000A6           CALLS   #1, CLEANUP                          ;
                                  04 000AB           RET                                          ;
                         04   AC  DD 000AC 4$:       PUSHL   KEYDESC                              ; 1292
                  04     0C   AC  E9 000AF           BLBC    REPLACING, 5$                        ; 1291
                              03  DD 000B3           PUSHL   #3                                   ;
                              02  11 000B5           BRB     6$                                   ;
                              02  DD 000B7 5$:       PUSHL   #2                                   ;
           0000G  CF          02  FB 000B9 6$:       CALLS   #2, LIB_LOG_UPD                      ;
                              68  DD 000BE           PUSHL   LIB$GL_LIBFDB                        ; 1294
```

```
                                         04   AC  DD 000C0        PUSHL   KEYDESC                          ;
                              08         0C   AC  E9 000C3        BLBC    REPLACING, 7$                    ;
                                  00000000G   8F  DD 000C7        PUSHL   #LIB$_REPLACED                   ; 1293
                                         06   11     000CD        BRB     8$                               ;
                                  00000000G   8F  DD 000CF 7$:    PUSHL   #LIB$_INSERTED                   ;
                        0000G   CF       03   FB     000D5 8$:    CALLS   #3, LIB_LOG_OP                   ;
      06          00            6E       00   2C     000DA        MOVC5   #0, (SP), #0, #6, @TXTRFA        ; 1296
                                         08   BC     000DF                                                 ;
                              50         01   D0     000E1        MOVL    #1, R0                           ; 1298
                                              04     000E4        RET                                      ; 1299

; Routine Size:  229 bytes,    Routine Base:  $CODE$ + 0059
```

```
;  245        1300 2 ROUTINE put_end =
;  246        1301 3 BEGIN
;  247        1302 3
;  248        1303 3 !++
;  249        1304 3 !
;  250        1305 3 ! Local routine to call lbr$put_end
;  251        1306 3 !
;  252        1307 3 !--
;  253        1308 3
;  254      P 1309 3 rms_perform (lbr$put_end (lib$gl_libctl),
;  255        1310 3         lib$_writeerr, .lbr$gl_rmsstv, 1, lib$gl_libfdb [fdb$l_namdesc]);
;  256        1311 3
;  257        1312 3 RETURN true
;  258        1313 2 END;
```

```
                                         0000 00000 PUT_END:.WORD   Save nothing                       ; 1300
                              0000G  CF    9F 00002         PUSHAB  LIB$GL_LIBCTL                      ; 1310
              00000000G  00          01    FB 00006         CALLS   #1, LBR$PUT_END                    ;
                     1D              50    E8 0000D         BLBS    STATUS, 1$                         ;
                        00000000G    00    DD 00010         PUSHL   LBR$GL_RMSSTV                      ;
                                     50    DD 00016         PUSHL   STATUS                             ;
         7E     0000G  CF            10    C1 00018         ADDL3   #16, LIB$GL_LIBFDB, -(SP)          ;
                                     01    DD 0001E         PUSHL   #1                                 ;
                        008610D2     8F    DD 00020         PUSHL   #8786130                           ;
              00000000G  00          05    FB 00026         CALLS   #5, LIB$SIGNAL                     ;
                     50              01    D0 0002D 1$:     MOVL    #1, R0                             ; 1312
                                           04 00030         RET                                        ; 1313

; Routine Size:  49 bytes,    Routine Base:  $CODE$ + 013E
```

```
;  260     1314  2 !
;  261     1315  2 ! Main body of lib_input_hlp
;  262     1316  2 !
;  263     1317  2
;  264     1318  2 LOCAL
;  265     1319  2     keyname : VECTOR [lbr$c_pagesize, BYTE],!key name when up cased
;  266     1320  2     found1,                                 !True when key1 found
;  267     1321  2     current_level,                          !Current level of key
;  268     1322  2     get_status,                             !status from reading input file
;  269     1323  2     ret_status,                             !status from last library operation
;  270     1324  2     level,                                  !new level
;  271     1325  2     replacing,
;  272     1326  2     deltxtrfa : BBLOCK [rfa$c_length],  !RFA of text to delete
;  273     1327  2     txtrfa : BBLOCK [rfa$c_length],     !RFA returned from PUT_RECORD
;  274     1328  2     key1desc : BBLOCK [dsc$c_s_bln],    !String descriptor for saved key1
;  275     1329  2     keydesc : BBLOCK [dsc$c_s_bln];     !String descriptor for key
;  276     1330  2
;  277     1331  2  current_level = 0;
;  278     1332  2 found1 = false;
;  279     1333  2 CH$FILL (0, rfa$c_length, txtrfa);
;  280     1334  2 !
;  281     1335  2 ! Read records until end of file
;  282     1336  2 !
;  283     1337  2 WHILE (get_status = get_record (linedesc)) NEQ rms$_eof DO
;  284     1338  3     BEGIN
;  285     1339  3     !
;  286     1340  3     !   Check for RUNOFF lines and remove <CR><LF>
;  287     1341  3     !
;  288     1342  3     IF .linedesc [dsc$w_length] GEQU 2  AND                         ! If line has two or more characters
;  289     1343  3       (.linedesc [dsc$a_pointer] + .linedesc [dsc$w_length] -2)<0,16>
;  290     1344  4       EQLU (CR OR LF^8)                                           ! and last two are <CR><LF>
;  291     1345  3     THEN
;  292     1346  3         linedesc [dsc$w_length] = .linedesc [dsc$w_length] - 2;     ! Strip off last two characters
;  293     1347  3
;  294     1348  3     IF is_key_on_line (level, keydesc)
;  295     1349  4     THEN BEGIN
;  296     1350  4         IF NOT .found1                                          !If havent found key1 yet
;  297     1351  5         THEN BEGIN
;  298     1352  5             IF .level NEQ 1                                     ! and this is not level 1
;  299     1353  6                 THEN BEGIN
;  300     1354  6                     SIGNAL (lib$_illkeylvl, 4, .level, keydesc, lib$gl_inpfdb [fdb$l_namdesc], 1);
;  301     1355  6                     cleanup (txtrfa);
;  302     1356  6                     RETURN lib$_illkeylvl;
;  303     1357  5                     END;
;  304     1358  5             END
;  305     1359  5
;  306     1360  5         ELSE BEGIN                                              !This is not first key1
;  307     1361  5             IF .level GTR .current_level                        !If greater than current level
;  308     1362  5                 AND .level NEQ .current_level+1
;  309     1363  5                 OR .level EQL 0
;  310     1364  6             THEN BEGIN
;  311     1365  6                 SIGNAL (lib$_illkeylvl, 4, .level, keydesc, lib$gl_inpfdb [fdb$l_namdesc], .current_level);
;  312     1366  6                 cleanup (txtrfa);
;  313     1367  6                 RETURN lib$_illkeylvl;
;  314     1368  5                 END;
;  315     1369  5
;  316     1370  5             current_level = .level;                             !Set new current level
```

```
;  317     1371  4              END;
;  318     1372  4
;  319     1373  4          IF .level NEQ 1                                          !If not first level
;  320     1374  4              THEN put_record (linedesc, txtrfa)                   ! then write the record
;  321     1375  4
;  322     1376  4 !
;  323     1377  4 ! This is level 1 key.  If we have seen level 1 before, then finish writing previous module
;  324     1378  4 ! and insert the key.  Then check new key length and save descriptor.
;  325     1379  4 !
;  326     1380  5              ELSE BEGIN
;  327     1381  5
;  328     1382  5                  IF .found1                                       !If writing one already
;  329     1383  6                  THEN BEGIN
;  330     1384  6                      put_end ();                                  !Write end of module record
;  331     1385  6                      insertkey1 (key1desc, txtrfa, .replacing, deltxtrfa);
;  332     1386  5                      END;
;  333     1387  5
;  334     1388  5                  IF .keydesc [dsc$w_length] GTR .lib$gl_keysize   !Check key length
;  335     1389  6                  THEN BEGIN
;  336     1390  6                      SIGNAL (lib$_keynamlng, 2, keydesc, lib$gl_inpfdb [fdb$l_namdesc]);
;  337     1391  6                      RETURN lib$_keynamlng;
;  338     1392  5                      END;
;  339     1393  5
;  340     1394  5                  CH$MOVE (.keydesc [dsc$w_length], .keydesc [dsc$a_pointer], keyname [1]);       ! Store key1
;  341     1395  5                  keydesc [dsc$a_pointer] = keyname [1];
;  342     1396  5                  keyname [0] = .keydesc [dsc$w_length];           ! set length
;  343     1397  5                  CH$MOVE (dsc$c_s_bln, keydesc, key1desc);        ! and copy it to a safe place
;  344     1398  5                  replacing = lbr$lookup_key (lib$gl_libctl, key1desc, deltxtrfa);
;  345     1399  5
;  346     1400  5                  IF NOT .lib$gl_ctlmsk [lib$v_replace]            !If not replacing
;  347     1401  5                      AND .replacing                               ! but module already in library
;  348     1402  6                      THEN BEGIN
;  349     1403  6                           !
;  350     1404  6                           !   Signal duplicate now and then process it.  It will be caught again later
;  351     1405  6                           !   and deleted.
;  352     1406  6                           !
;  353     1407  6                           SIGNAL (lib$_dupmodule, 3, keyname, lib$gl_inpfdb [fdb$l_namdesc],
;  354     1408  6                                   lib$gl_libfdb [fdb$l_namdesc]);
;  355     1409  5                           END;
;  356     1410  5
;  357     1411  5                  put_record (linedesc, txtrfa);                   !Write the first record
;  358     1412  5                  found1 = true;                                   !Flag key1 found
;  359     1413  5                  current_level = .level;                          !Set current level
;  360     1414  4                  END;
;  361     1415  4          END
;  362     1416  4
;  363     1417  3          ELSE IF .found1                                          !Not a key line, if we have seen a k
;  364     1418  3              THEN put_record (linedesc, txtrfa);                  ! then write this line to module
;  365     1419  3
;  366     1420  2 END;                                                             !Of WHILE loop
;  367     1421  2
;  368     1422  2 !
;  369     1423  2 ! Done reading input file
;  370     1424  2 !
;  371     1425  2
;  372     1426  2 IF NOT .found1
;  373     1427  2     THEN SIGNAL (lib$_nohlptxt, 1, lib$gl_inpfdb [fdb$l_namdesc])
```

```
;  374          1428  2
;  375          1429  2 !
;  376          1430  2 ! The module ended with end of module. finish the module now.
;  377          1431  2 !
;  378          1432  2
;  379          1433  3     ELSE BEGIN
;  380          1434  3         put_end ();                                  !Write end of module record
;  381          1435  3         insertkey1 (keydesc, txtrfa, .replacing, deltxtrfa);   ! and then insert the key
;  382          1436  2         END;
;  383          1437  2
;  384          1438  2 RETURN true
;  385          1439  1 END;                                                 ! Of lib_input_hlp
; INFO#250                 L1:1385
; Referenced LOCAL symbol REPLACING is probably not initialized
```

```
                                        0FFC 00000          .ENTRY  LIB_INPUT_HLP, Save R2,R3,R4,R5,R6,R7,R8,-      ; 1205
                                                                    R9,R10,R11
                                5B 00000000G    00  9E 00002        MOVAB   LIB$SIGNAL, R11                         ;
                                5E      FDDC    CE  9E 00009        MOVAB   -548(SP), SP                            ;
                                                56  D4 0000E        CLRL    CURRENT_LEVEL                           ; 1331
                                                59  D4 00010        CLRL    FOUND1                                  ; 1332
        06              00              6E      00  2C 00012        MOVC5   #0, (SP), #0, #6, TXTRFA                ; 1333
                                                14  AE   00017                                                       ;
                                            0000'   CF  9F 00019 1$:    PUSHAB  LINEDESC                            ; 1337
                                    0000G   CF          01  FB 0001D    CALLS   #1, GET_RECORD                      ;
                                            5A          50  D0 00022    MOVL    R0, GET_STATUS                      ;
                                0001827A    8F          5A  D1 00025    CMPL    GET_STATUS, #98938                  ;
                                                        03  12 0002C    BNEQ    2$                                  ;
                                                    0135    31 0002E    BRW     18$                                 ;
                                    02      0000'   CF  B1 00031 2$:    CMPW    LINEDESC, #2                        ; 1342
                                                        17  1F 00036    BLSSU   3$                                  ;
                                    50      0000'   CF  3C 00038        MOVZWL  LINEDESC, R0                        ; 1343
                                    50      0000'   CF  C0 0003D        ADDL2   LINEDESC+4, R0                      ;
                        0A0D        8F        FE    A0  B1 00042        CMPW    -2(R0), #2573                       ; 1344
                                                    05  12 00048        BNEQ    3$                                  ;
                        0000'       CF              02  A2 0004A        SUBW2   #2, LINEDESC                        ; 1346
                                            04      AE  9F 0004F 3$:    PUSHAB  KEYDESC                             ; 1348
                                            04      AE  9F 00052        PUSHAB  LEVEL                               ;
                        0000V       CF              02  FB 00055        CALLS   #2, IS_KEY_ON_LINE                  ;
                                    03              50  E8 0005A        BLBS    R0, 4$                              ;
                                                00F4    31 0005D        BRW     15$                                 ;
                                    57              6E  D0 00060 4$:    MOVL    LEVEL, R7                           ; 1352
                                    09              59  E8 00063        BLBS    FOUND1, 5$                          ;
                                    01              57  D1 00066        CMPL    R7, #1                              ;
                                                    41  13 00069        BEQL    10$                                 ;
                                                    01  DD 0006B        PUSHL   #1                                  ; 1354
                                                    14  11 0006D        BRB     8$                                  ;
                                    56              57  D1 0006F 5$:    CMPL    R7, CURRENT_LEVEL                   ; 1361
                                                    09  15 00072        BLEQ    6$                                  ;
                                    50      01      A6  9E 00074        MOVAB   1(R6), R0                           ; 1362
                                    50              57  D1 00078        CMPL    R7, R0                              ;
                                                    04  12 0007B        BNEQ    7$                                  ;
                                                    57  D5 0007D 6$:    TSTL    R7                                  ; 1363
```

```
                                          28 12 0007F          BNEQ    9$                                       ;
                                          56 DD 00081 7$:      PUSHL   CURRENT_LEVEL                            ; 1365
                7E      0000G  CF         10 C1 00083 8$:      ADDL3   #16, LIB$GL_INPFDB, -(SP)                ;
                                   0C     AE 9F 00089          PUSHAB  KEYDESC                                  ;
                                          57 DD 0008C          PUSHL   R7                                       ;
                                          04 DD 0008E          PUSHL   #4                                       ;
                              00000000G   8F DD 00090          PUSHL   #LIB$_ILLKEYLVL                          ;
                               6B         06 FB 00096          CALLS   #6, LIB$SIGNAL                           ;
                                   14     AE 9F 00099          PUSHAB  TXTRFA                                   ; 1366
                        FE25   CF         01 FB 0009C          CALLS   #1, CLEANUP                              ;
                               50 00000000G 8F D0 000A1        MOVL    #LIB$_ILLKEYLVL, R0                      ; 1367
                                             04 000A8          RET                                              ;
                               56         57 D0 000A9 9$:      MOVL    R7, CURRENT_LEVEL                        ; 1370
                               01         57 D1 000AC 10$:     CMPL    R7, #1                                   ; 1373
                                          03 13 000AF          BEQL    11$                                      ;
                                        00A3 31 000B1          BRW     16$                                      ;
                               15         59 E9 000B4 11$:     BLBC    FOUND1, 12$                              ; 1382
                        FF13   CF         00 FB 000B7          CALLS   #0, PUT_END                              ; 1384
                                   1C     AE 9F 000BC          PUSHAB  DELTXTRFA                                ; 1385
                                          58 DD 000BF          PUSHL   REPLACING                                ;
                                   1C     AE 9F 000C1          PUSHAB  TXTRFA                                   ;
                                   18     AE 9F 000C4          PUSHAB  KEY1DESC                                 ;
                        FE1E   CF         04 FB 000C7          CALLS   #4, INSERTKEY1                           ;
  0000G  CF      04    AE                 10 00 ED 000CC 12$:  CMPZV   #0, #16, KEYDESC, LIB$GL_KEYSIZE         ; 1388
                                          1C 15 000D4          BLEQ    13$                                      ;
                7E      0000G  CF         10 C1 000D6          ADDL3   #16, LIB$GL_INPFDB, -(SP)                ; 1390
                                   08     AE 9F 000DC          PUSHAB  KEYDESC                                  ;
                                          02 DD 000DF          PUSHL   #2                                       ;
                              00000000G   8F DD 000E1          PUSHL   #LIB$_KEYNAMLNG                          ;
                               6B         04 FB 000E7          CALLS   #4, LIB$SIGNAL                           ;
                               50 00000000G 8F D0 000EA        MOVL    #LIB$_KEYNAMLNG, R0                      ; 1391
                                             04 000F1          RET                                              ;
         25    AE       08     BE     04  AE 28 000F2 13$:     MOVC3   KEYDESC, @KEYDESC+4, KEYNAME+1           ; 1394
                        08     AE     25  AE 9E 000F9          MOVAB   KEYNAME+1, KEYDESC+4                     ; 1395
                        24     AE     04  AE 90 000FE          MOVB    KEYDESC, KEYNAME                         ; 1396
         0C    AE       04     AE         08 28 00103          MOVC3   #8, KEYDESC, KEY1DESC                    ; 1397
                                   1C     AE 9F 00109          PUSHAB  DELTXTRFA                                ; 1398
                                   10     AE 9F 0010C          PUSHAB  KEY1DESC                                 ;
                                 0000G    CF 9F 0010F          PUSHAB  LIB$GL_LIBCTL                            ;
                   00000000G   00         03 FB 00113          CALLS   #3, LBR$LOOKUP_KEY                       ;
                               58         50 D0 0011A          MOVL    R0, REPLACING                            ;
                1D      0000G  CF         05 E0 0011D          BBS     #5, LIB$GL_CTLMSK+1, 14$                 ; 1400
                               1A         58 E9 00123          BLBC    REPLACING, 14$                           ; 1401
                7E      0000G  CF         10 C1 00126          ADDL3   #16, LIB$GL_LIBFDB, -(SP)                ; 1408
                7E      0000G  CF         10 C1 0012C          ADDL3   #16, LIB$GL_INPFDB, -(SP)                ; 1407
                                   2C     AE 9F 00132          PUSHAB  KEYNAME                                  ;
                                          03 DD 00135          PUSHL   #3                                       ; 1408
                              00000000G   8F DD 00137          PUSHL   #LIB$_DUPMODULE                          ;
                               6B         05 FB 0013D          CALLS   #5, LIB$SIGNAL                           ;
                                   14     AE 9F 00140 14$:     PUSHAB  TXTRFA                                   ; 1411
                                 0000'    CF 9F 00143          PUSHAB  LINEDESC                                 ;
                        FD45   CF         02 FB 00147          CALLS   #2, PUT_RECORD                           ;
                               59         01 D0 0014C          MOVL    #1, FOUND1                               ; 1412
                               56         57 D0 0014F          MOVL    R7, CURRENT_LEVEL                        ; 1413
                                          0F 11 00152          BRB     17$                                      ; 1348
                               0C         59 E9 00154 15$:     BLBC    FOUND1, 17$                              ; 1417
                                   14     AE 9F 00157 16$:     PUSHAB  TXTRFA                                   ; 1418
```

```
                         0000'  CF   9F 0015A          PUSHAB  LINEDESC                          ;
            FD2E   CF              02   FB 0015E          CALLS   #2, PUT_RECORD                    ;
                            FEB3   31 00163 17$:       BRW     1$                                ; 1337
                   13              59   E8 00166 18$:       BLBS    FOUND1, 19$                       ; 1426
      7E    0000G  CF              10   C1 00169          ADDL3   #16, LIB$GL_INPFDB, -(SP)         ; 1427
                                   01   DD 0016F          PUSHL   #1                                ;
                     00000000G     8F   DD 00171          PUSHL   #LIB$_NOHLPTXT                    ;
                   6B              03   FB 00177          CALLS   #3, LIB$SIGNAL                    ;
                                   15   11 0017A          BRB     20$                               ;
            FE4E   CF              00   FB 0017C 19$:       CALLS   #0, PUT_END                       ; 1434
                          1C       AE   9F 00181          PUSHAB  DELTXTRFA                         ; 1435
                                   58   DD 00184          PUSHL   REPLACING                         ;
                          1C       AE   9F 00186          PUSHAB  TXTRFA                            ;
                          18       AE   9F 00189          PUSHAB  KEY1DESC                          ;
            FD59   CF              04   FB 0018C          CALLS   #4, INSERTKEY1                    ;
                   50              01   D0 00191 20$:       MOVL    #1, R0                            ; 1438
                                        04 00194          RET                                       ; 1439

; Routine Size:  405 bytes,    Routine Base:  $CODE$ + 016F
```

```
1 ROUTINE is_key_on_line (level, keydesc) =
2 BEGIN
2 !
2 ! This routine returns true if there is a <number><key>
2 ! on a line and false if not.
2 !
2 MAP
2     keydesc : REF BBLOCK;
2
2 LOCAL
2     tokenptr,
2     tokenlen;
2
2 IF .linelen EQL 0 THEN RETURN false;
2
2 lineptr = .lineaddr - 1;
2 endptr = .lineaddr + .linelen;
2 curchar = CH$RCHAR (.lineptr + 1);
2 IF .curchar LEQU %ASCII'0'
2 OR .curchar GTRU %ASCII '9'
2 THEN RETURN false
3 ELSE BEGIN
3     skip_blanks ();
3     tokenlen = scan_word ();
3     tokenptr = .lineaddr;
3     IF NOT lib$cvt_dtb (.tokenlen, .tokenptr, .level)
3         THEN RETURN false;
3     IF NOT skip_blanks ()
3         THEN RETURN false
4         ELSE BEGIN
4             tokenptr = .lineptr;
4             tokenlen = scan_word ();
4             keydesc [dsc$w_length] = .tokenlen;
4             keydesc [dsc$a_pointer] = .tokenptr;
4             RETURN true;
3             END;
2     END;
1 END;                                          !Of is_key_on_line


                                 001C 00000 IS_KEY_ON_LINE:
                                                .WORD   Save R2,R3,R4                  ; 1440
                  54      0000'  CF  9E 00002   MOVAB   LINEADDR, R4
                  50        FC   A4  3C 00007   MOVZWL  LINELEN, R0                    ; 1453
                             63  13 0000B       BEQL    1$                             ;
 F0  A4           64         01  C3 0000D       SUBL3   #1, LINEADDR, LINEPTR          ; 1455
 F4  A4           64         50  C1 00012       ADDL3   R0, LINEADDR, ENDPTR           ; 1456
                  50   F0    A4  D0 00017       MOVL    LINEPTR, R0                    ; 1457
            F8    A4   01    A0  9A 0001B       MOVZBL  1(R0), CURCHAR                 ;
                  30   F8    A4  D1 00020       CMPL    CURCHAR, #48                   ; 1458
                             4A  1B 00024       BLEQU   1$                             ;
                  39   F8    A4  D1 00026       CMPL    CURCHAR, #57                   ; 1459
                             44  1A 0002A       BGTRU   1$                             ;
        0000V     CF         00  FB 0002C       CALLS   #0, SKIP_BLANKS                ; 1462
```

```
      0000V  CF          00  FB 00031        CALLS   #0, SCAN_WORD          ; 1463
             53          50  D0 00036        MOVL    R0, TOKENLEN           ;
             52          64  D0 00039        MOVL    LINEADDR, TOKENPTR     ; 1464
                     04  AC  DD 0003C        PUSHL   LEVEL                  ; 1465
                         52  DD 0003F        PUSHL   TOKENPTR               ;
                         53  DD 00041        PUSHL   TOKENLEN               ;
  00000000G  00          03  FB 00043        CALLS   #3, LIB$CVT_DTB        ;
             23          50  E9 0004A        BLBC    R0, 1$                 ;
      0000V  CF          00  FB 0004D        CALLS   #0, SKIP_BLANKS        ; 1467
             1B          50  E9 00052        BLBC    R0, 1$                 ;
             52      F0  A4  D0 00055        MOVL    LINEPTR, TOKENPTR      ; 1470
      0000V  CF          00  FB 00059        CALLS   #0, SCAN_WORD          ; 1471
             53          50  D0 0005E        MOVL    R0, TOKENLEN           ;
             50      08  AC  D0 00061        MOVL    KEYDESC, R0            ; 1472
             60          53  B0 00065        MOVW    TOKENLEN, (R0)         ;
         04  A0          52  D0 00068        MOVL    TOKENPTR, 4(R0)        ; 1473
             50          01  D0 0006C        MOVL    #1, R0                 ; 1474
                             04 0006F        RET                            ; 1461
                         50  D4 00070 1$:    CLRL    R0                     ; 1477
                             04 00072        RET                            ;

; Routine Size:  115 bytes,    Routine Base:  $CODE$ + 0304
```

```
1 ROUTINE scan_word =
2 BEGIN
2 !
2 ! This routine returns the length of the word which is pointed
2 ! to currently by lineptr and advances lineptr to the character past
2 ! the end of the word.
2 !
2 LOCAL
2     startptr;
2
2 startptr = .lineptr;
2 WHILE CH$DIFF (.endptr, .lineptr + 1) GTR 0
3 DO BEGIN
3     curchar = CH$A_RCHAR (lineptr);
3     IF NOT symbol_char () THEN RETURN .lineptr - .startptr;
2     END;
2 RETURN .lineptr + 1 - .startptr;
1 END;                                        !Of scan_word



                                    000C 00000 SCAN_WORD:
                                               .WORD   Save R2,R3                     ; 1478
          53     0000'   CF   9E 00002         MOVAB   LINEPTR, R3                    ;
          52             63   D0 00007         MOVL    LINEPTR, STARTPTR              ; 1488
  50      63             01   C1 0000A 1$:     ADDL3   #1, LINEPTR, R0                ; 1489
          50     04      A3   D1 0000E         CMPL    ENDPTR, R0                     ;
                         14   15 00012         BLEQ    2$                             ;
                         63   D6 00014         INCL    LINEPTR                        ; 1491
     08   A3     00      B3   9A 00016         MOVZBL  @LINEPTR, CURCHAR              ;
  0000V   CF             00   FB 0001B         CALLS   #0, SYMBOL_CHAR                ; 1492
          E7             50   E8 00020         BLBS    R0, 1$                         ;
  50      63             52   C3 00023         SUBL3   STARTPTR, LINEPTR, R0          ;
                              04 00027         RET                                    ;
  50      63             52   C3 00028 2$:     SUBL3   STARTPTR, LINEPTR, R0          ; 1494
                         50   D6 0002C         INCL    R0                             ;
                              04 0002E         RET                                    ; 1495

; Routine Size:  47 bytes,    Routine Base:  $CODE$ + 0377
```

```
1 ROUTINE skip_blanks =
2 BEGIN
2 !
2 ! This routine skips blanks and tabs in the input line.
2 ! Returns true if skipped to non-blank, non-tab character.
2 ! Returns false if skipped to exclamation point or end of line.
2 !
2 WHILE CH$DIFF (.endptr, .lineptr + 1) GTR 0
3 DO BEGIN
3     curchar = CH$A_RCHAR (lineptr);
3     IF .curchar EQL %ASCII'!' THEN
3         RETURN false
3     ELSE
3         IF .curchar NEQ %ASCII' ' AND .curchar NEQ %ASCII'      ' THEN
3             RETURN symbol_char ();
2     END;
2 RETURN false;                                   !Return false for end of line
1 END;                                            !OF skip_blanks


                                 0004 00000 SKIP_BLANKS:
                                                   .WORD   Save R2                  ; 1496
              52    0000'  CF  9E 00002            MOVAB   LINEPTR, R2
      50      62           01  C1 00007 1$:        ADDL3   #1, LINEPTR, R0          ; 1503
              50      04   A2  D1 0000B            CMPL    ENDPTR, R0
                           20  15 0000F            BLEQ    2$
                           62  D6 00011            INCL    LINEPTR                  ; 1505
         08   A2      00   B2  9A 00013            MOVZBL  @LINEPTR, CURCHAR
              50      08   A2  D0 00018            MOVL    CURCHAR, R0              ; 1506
              21           50  D1 0001C            CMPL    R0, #33
                           10  13 0001F            BEQL    2$
              20           50  D1 00021            CMPL    R0, #32                  ; 1509
                           E1  13 00024            BEQL    1$
              09           50  D1 00026            CMPL    R0, #9
                           DC  13 00029            BEQL    1$
       0000V  CF           00  FB 0002B            CALLS   #0, SYMBOL_CHAR          ; 1510
                               04 00030            RET
                           50  D4 00031 2$:        CLRL    R0                       ; 1513
                               04 00033            RET

; Routine Size:  52 bytes,    Routine Base:  $CODE$ + 03A6
```

```
;  464          1514  1 ROUTINE symbol_char =
;  465          1515  2 BEGIN
;  466          1516  2 !
;  467          1517  2 ! This routine returns true if curchar is a character that may be
;  468          1518  2 ! in a help key, and false if not.
;  469          1519  2 !
;  470          1520  2 OWN
;  471          1521  2     delimiters: VECTOR [4, BYTE] INITIAL ('     , !'), ! Tab, comma, space, & exclamation point.
;  472          1522  2     symbolics : VECTOR [96, BYTE] INITIAL
;  473          1523  2 ('!''#$%&''()*+-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~');
;  474          1524  2 !123456789_123456789_123456789_123456789_123456789_123456789_123456789_123456789_123456789_123456789_
;  475          1525  2 ! (Note that the vector size has to be a multiple of 4 large enough to
;  476          1526  2 ! hold the string inclusive of the delimiting quotes and counting the
;  477          1527  2 ! double quotes (which actually represents a single quote) as two.
;  478          1528  2 ! However, the value used in the Ch$find_ch lexical which follows,
;  479          1529  2 ! is the actual number of the allowed charcters.)
;  480          1530  2
;  481          1531  2 IF CH$FAIL (CH$FIND_CH (4, delimiters, .curchar)) THEN
;  482          1532  2     IF CH$FAIL (CH$FIND_CH (93, symbolics, (%X'7F' AND .curchar))) THEN
;  483          1533  2         SIGNAL (lib$_invkeychar, 4, 1, curchar, .curchar, linedesc)
;  484          1534  2     ELSE
;  485          1535  2         RETURN true;
;  486          1536  2
;  487          1537  2 RETURN false;
;  488          1538  1 END;                                    !Of symbol_char


                                                        .PSECT  $OWN$,NOEXE,2

                                                 21  20  2C  09  00014 DELIMITERS:
                                                                         .ASCII  <9>\, !\                        ;
30  2F  2E  2D  2B  2A  29  28  27  26  25  24  23  22  21  00018 SYMBOLICS:
                                                                         .ASCII  \!''#$%&'()*+-./0123456789:;<=>?@ABCDEFGHI\     ;
3F  3E  3D  3C  3B  3A  39  38  37  36  35  34  33  32  31  00027
                    49  48  47  46  45  44  43  42  41  40  00036
58  57  56  55  54  53  52  51  50  4F  4E  4D  4C  4B  4A  00040        .ASCII  \JKLMNOPQRSTUVWXYZ[\<92>\]^_`abcdefghijk\       ;
67  66  65  64  63  62  61  60  5F  5E  5D  5C  5B  5A  59  0004F
                                    6B  6A  69  68  0005E
7A  79  78  77  76  75  74  73  72  71  70  6F  6E  6D  6C  00062        .ASCII  \lmnopqrstuvwxyz{|}~\<0><0><0>                   ;
                            00  00  00  7E  7D  7C  7B  00071


                                                        .PSECT  $CODE$,NOWRT,2

                                                     000C 00000 SYMBOL_CHAR:
                                                                         .WORD   Save R2,R3                      ; 1514
                                        53      0000'    CF  9E 00002        MOVAB   CURCHAR, R3
                                        52               63  D0 00007        MOVL    CURCHAR, R2                    1531
                                0C  A3          04       52  3A 0000A        LOCC    R2, #4, DELIMITERS              ;
                                                         02  12 0000F        BNEQ    1$                              ;
                                                         51  D4 00011        CLRL    R1                              ;
                                                         51  D5 00013 1$:     TSTL    R1                              ;
                                                         32  12 00015        BNEQ    4$                              ;
                    50              52          07       00  EF 00017        EXTZV   #0, #7, R2, R0                  ; 1532
                                10  A3      005D  8F       50  3A 0001C        LOCC    R0, #93, SYMBOLICS              ;
```

```
                              02  12 00023          BNEQ    2$                    ;
                              51  D4 00025          CLRL    R1                    ;
                              51  D5 00027 2$:      TSTL    R1                    ;
                              1A  12 00029          BNEQ    3$                    ;
                        04    A3  9F 0002B          PUSHAB  LINEDESC              ; 1533
                              52  DD 0002E          PUSHL   R2                    ;
                              53  DD 00030          PUSHL   R3                    ;
                              01  DD 00032          PUSHL   #1                    ;
                              04  DD 00034          PUSHL   #4                    ;
                 00000000G    8F  DD 00036          PUSHL   #LIB$_INVKEYCHAR      ;
     00000000G  00            06  FB 0003C          CALLS   #6, LIB$SIGNAL        ;
                              04  11 00043          BRB     4$                    ;
             50               01  D0 00045 3$:      MOVL    #1, R0                ; 1535
                                  04 00048          RET                           ;
                              50  D4 00049 4$:      CLRL    R0                    ; 1537
                                  04 0004B          RET                           ; 1538

; Routine Size:  76 bytes,    Routine Base:  $CODE$ + 03DA
```

```
;  490          1539  0 END ELUDOM


                                                        .EXTRN  LIB$SIGNAL

;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $OWN$                          120  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                        1062  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                              Library Statistics
;
;                                      -------- Symbols --------      Pages       Processing
;       File                           Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1     9776        25         0       581          00:01.1


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


;                                    COMMAND QUALIFIERS

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

; Size:         1062 code + 120 data bytes
; Run Time:        00:26.8
; Elapsed Time:    00:52.0
; Lines/CPU Min:    3451
; Lexemes/CPU-Min: 35679
; Memory Used:  224 pages
; Compilation Complete
```

GETCMD
LIS

INPUTOBJ
LIS

INPUTTXT
LIS

LIBRARIAN
LIS

INPUTMAC
LIS

INPUTHLP
LIS