E 9

```
LLL              BBBBBBBBBBBB    RRRRRRRRRRRR
LLL              BBBBBBBBBBBB    RRRRRRRRRRRR
LLL              BBBBBBBBBBBB    RRRRRRRRRRRR
LLL              BBB      BBB    RRR         RRR
LLL              BBB      BBB    RRR         RRR
LLL              BBB      BBB    RRR         RRR
LLL              BBB      BBB    RRR         RRR
LLL              BBB      BBB    RRR         RRR
LLL              BBB      BBB    RRR         RRR
LLL              BBBBBBBBBBBB    RRRRRRRRRRRR
LLL              BBBBBBBBBBBB    RRRRRRRRRRRR
LLL              BBBBBBBBBBBB    RRRRRRRRRRRR
LLL              BBB      BBB    RRR   RRR
LLL              BBB      BBB    RRR   RRR
LLL              BBB      BBB    RRR   RRR
LLL              BBB      BBB    RRR      RRR
LLL              BBB      BBB    RRR      RRR
LLL              BBB      BBB    RRR      RRR
LLLLLLLLLLLLLLL  BBBBBBBBBBBB    RRR         RRR
LLLLLLLLLLLLLLL  BBBBBBBBBBBB    RRR         RRR
LLLLLLLLLLLLLLL  BBBBBBBBBBBB    RRR         RRR
```

FILEID**DUMP


```
DDDDDDDD    UU      UU  MM      MM  PPPPPPPP
DDDDDDDD    UU      UU  MM      MM  PPPPPPPP
DD      DD  UU      UU  MMMM  MMMM  PP      PP
DD      DD  UU      UU  MMMM  MMMM  PP      PP
DD      DD  UU      UU  MM  MM  MM  PP      PP
DD      DD  UU      UU  MM  MM  MM  PP      PP
DD      DD  UU      UU  MM      MM  PPPPPPPP
DD      DD  UU      UU  MM      MM  PPPPPPPP
DD      DD  UU      UU  MM      MM  PP
DD      DD  UU      UU  MM      MM  PP
DD      DD  UU      UU  MM      MM  PP              ....
DD      DD  UU      UU  MM      MM  PP              ....
DDDDDDDD    UUUUUUUUUU  MM      MM  PP              ....
DDDDDDDD    UUUUUUUUUU  MM      MM  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
```

```
0 MODULE lbr_dump (                                       ! Library examination routines
0                       LANGUAGE (BLISS32),
0                       IDENT = 'V04-000') =
1 BEGIN
1
1 !
1 !****************************************************************************
1 !*                                                                          *
1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
1 !*  ALL RIGHTS RESERVED.                                                    *
1 !*                                                                          *
1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
1 !*  TRANSFERRED.                                                            *
1 !*                                                                          *
1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
1 !*  CORPORATION.                                                            *
1 !*                                                                          *
1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
1
1 !++
1 !
1 ! FACILITY:  Library access procedures
1 !
1 ! ABSTRACT:
1 !
1 !       The VAX/VMS librarian procedures implement a standard access method
1 !       to libraries through a shared, common procedure set.
1 !
1 ! ENVIRONMENT:
1 !
1 !       VAX native, user mode.
1 !
1 !--
1 !
1 !
1 ! AUTHOR: Tim Halvorsen, Benn Schreiber, Bob Grosso      16-Mar-1981
1 !
1 ! MODIFIED BY:
1 !
1 !       V02-003         RPG0003         Bob Grosso      15-Dec-1981
1 !               Enhance command interpretation and help.
1 !               Add next function.
1 !
1 !       V02-002         RPG0002         Bob Grosso      30-Jul-1981
1 !               Remove non-pic descriptors and add some error checking.
1 !
1 !--
```

```
1 %SBTTL 'Declarations';
1
1 LIBRARY 'SYS$LIBRARY:STARLET.L32';       ! VAX/VMS common definitions
1
1 REQUIRE 'PREFIX';                        ! Librarian general definitions
1
1 REQUIRE 'LBRDEF';                        ! Librarian structure definitions
1
1 FORWARD ROUTINE
1       lbr$dump_indexblocks,              ! Dump the entire index in hex and ASCII
1       lbr$dump_index;                    ! Dump the index
1
1 EXTERNAL ROUTINE
1       find_block : JSB_3,
1       find_index : JSB_2,                                    ! Find index block in memory
1       lookup_cache : JSB_2,
1       scr$set_cursor : ADDRESSING_MODE (GENERAL),
1       scr$get_screen : ADDRESSING_MODE (GENERAL),
1       scr$erase_line : ADDRESSING_MODE (GENERAL),
1       scr$erase_page : ADDRESSING_MODE (GENERAL),
1       scr$put_buffer : ADDRESSING_MODE (GENERAL),
1       scr$put_screen : ADDRESSING_MODE (GENERAL),
1       scr$set_buffer : ADDRESSING_MODE (GENERAL),
1       ots$cvt_tz_l   : ADDRESSING_MODE (GENERAL), ! convert text to hex
1       lib$put_output : ADDRESSING_MODE (GENERAL); ! Write to SYS$OUTPUT
1
1 EXTERNAL
1       lbr$gl_control: REF BBLOCK; ! Address of control block
1
1 OWN
1       cur_vbn,
1       faostring1 : countedstring ('!/Index block !XL at address !XL'),
1       faostring2 : countedstring ('(parent = !XL, used = !XW)!/'),
1       faostring3 : countedstring ('!XW   !XL !XW   !AC'),
1       faostring4 : countedstring ('!XW   !XL !XW   !XL'),
1       faodmpblk  : countedstring ('!XL !XL !XL !XL !AF'),
1       faohash   : countedstring ('          !XL'),   ! print the contents of a hash bucket
1       faohashentry : countedstring ('!XL !XL'),    ! print the address of the hash bucket followed by it's cont
1       faoblock : countedstring ('!XL !XL !XL !XL !XL !XL !XL !XL  !AF'), ! dump a line of the block in hex and
1       faoentry : countedstring ('!XL ');            ! print one of the four longwords of a cache entry
1
1 BIND
1       help_desc_0 = $DESCRIPTOR ('From SYS$INPUT, enter a single letter command with an <ESC>.'),
1       help_desc_1 = $DESCRIPTOR ('?  Help'),
1       help_desc_2 = $DESCRIPTOR ('N  (North) move up the hash table'),
1       help_desc_3 = $DESCRIPTOR ('S  (South) move down a bucket'),
1       help_desc_4 = $DESCRIPTOR ('E  (East) move over to the next cache entry in current hash bucket'),
1       help_desc_5 = $DESCRIPTOR ('W  (West) move back in the linked list of cache entries'),
1       help_desc_6 = $DESCRIPTOR ('B  jump Back to the first cache entry in current bucket'),
1       help_desc_7 = $DESCRIPTOR ('O  return to Origin; first entry of first bucket '),
1       help_desc_8 = $DESCRIPTOR ('J  Jump down 10 buckets in hash table'),
1       help_desc_9 = $DESCRIPTOR ('D  Display block of current cache entry'),
1       help_desc_10 = $DESCRIPTOR ('H  Display library header block'),
1       help_desc_11 = $DESCRIPTOR ('L  query for a VBN to locate in cache'),
1       help_desc_12 = $DESCRIPTOR ('G  query for a VBN to get from library and cache'),
1       help_desc_13 = $DESCRIPTOR ('X  dump neXt vbn'),
1       help_desc_14 = $DESCRIPTOR ('R  Reprint the screen'),
```

```
    help_desc_15 = $DESCRIPTOR ('Q  Quit and clear the screen');

LITERAL
    first_bucket = 0,                           ! offset to first bucket in hash
    last_bucket = lbr$c_hashsize/4 - 1;         ! last bucket in hash table

MACRO
    write_screen (line, col, string) =          ! write to specified position on screen
        scr$put_screen ( fao_buffer (string     ! string must contain fao format line followed by fao arguem
        %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI),
        line, col)%;
```

```
    1 %SBTTL 'LBR$DUMP_INDEX';
    1
    1 GLOBAL ROUTINE lbr$dump_index (index) =
    1
    1 !---
    1 !
    1 !       Dump the contents of the entire index structure
    1 !       to SYS$OUTPUT.
    1 !
    1 ! Inputs:
    1 !
    1 !       index = Primary index number
    1 !
    1 ! Outputs:
    1 !
    1 !       None
    1 !
    1 !---
    1
    2 BEGIN
    2 MACRO
  M 2     write (string) =
  M 2         lib$put_output (fao_buffer (string
    2         %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI))%;
    2
    2 EXTERNAL ROUTINE
    2     lib$put_output : ADDRESSING_MODE (GENERAL);
    2
    2 ROUTINE fao_buffer (ctrstr,args) =
    3 BEGIN
    3 OWN
    3     desc :      BBLOCK [dsc$c_s_bln],   ! Result descriptor
    3     buf :       VECTOR [80, BYTE];      ! Output buffer
    3 MAP
    3     ctrstr :    REF VECTOR [,BYTE],
    3     args :      VECTOR [4];
    3
    3 LOCAL
    3     faodesc : BBLOCK [dsc$c_s_bln];
    3
    3 faodesc [dsc$w_length] = .ctrstr [0];
    3 faodesc [dsc$a_pointer] = ctrstr [1];
    3 desc [dsc$w_length] = 80;                        ! Set up result descriptor
    3 desc [dsc$a_pointer] = buf;
    3 $faol (ctrstr=faodesc, outlen=desc, outbuf=desc, prmlst=args);
    3 RETURN desc;
    2 END;

                                                   .TITLE  LBR_DUMP
                                                   .IDENT  \V04-000\

                                                   .PSECT  $SPLIT$,NOWRT,NOEXE,2

2C 54 55 50 4E 49 24 53 59 53 20 6D 6F 72 46 00000 P.AAB: .ASCII \From SYS$INPUT, enter a single letter co\ ;
65 6C 67 6E 69 73 20 61 20 72 65 74 6E 65 20 0000F                                                           ;
                  6F 63 20 72 65 74 74 65 6C 20 0001E                                                           ;
```

```
3C 20 6E 61 20 68 74 69 77 20 64 6E 61 6D 6D  00028           .ASCII  \mmand with an <ESC>.\
                              2E 3E 43 53 45  00037
                                   0000003C.  0003C P.AAA:    .LONG   60
                                   00000000'  00040           .ADDRESS P.AAB
                        70 6C 65 48 20 20 3F  00044 P.AAD:    .ASCII  \?  Help\
                                              0004B           .BLKB   1
                                   00000007.  0004C P.AAC:    .LONG   7
                                   00000000'  00050           .ADDRESS P.AAD
65 76 6F 6D 20 29 68 74 72 6F 4E 28 20 20 4E  00054 P.AAF:    .ASCII  \N  (North) move up the hash table\
61 74 20 68 73 61 68 20 65 68 74 20 70 75 20  00063
                                    65 6C 62  00072
                                              00075           .BLKB   3
                                   00000021.  00078 P.AAE:    .LONG   33
                                   00000000'  0007C           .ADDRESS P.AAF
65 76 6F 6D 20 29 68 74 75 6F 53 28 20 20 53  00080 P.AAH:    .ASCII  \S  (South) move down a bucket\
   74 65 6B 63 75 62 20 61 20 6E 77 6F 64 20  0008F
                                              0009D           .BLKB   3
                                   0000001D.  000A0 P.AAG:    .LONG   29
                                   00000000'  000A4           .ADDRESS P.AAH
20 65 76 6F 6D 20 29 74 73 61 45 28 20 20 45  000A8 P.AAJ:    .ASCII  \E  (East) move over to the next cache en\
78 65 6E 20 65 68 74 20 6F 74 20 72 65 76 6F  000B7
               6E 65 20 65 68 63 61 63 20 74  000C6
20 74 6E 65 72 72 75 63 20 6E 69 20 79 72 74  000D0           .ASCII  \try in current hash bucket\
            74 65 6B 63 75 62 20 68 73 61 68  000DF
                                              000EA           .BLKB   2
                                   00000042.  000EC P.AAI:    .LONG   66
                                   00000000'  000F0           .ADDRESS P.AAJ
20 65 76 6F 6D 20 29 74 73 65 57 28 20 20 57  000F4 P.AAL:    .ASCII  \W  (West) move back in the linked list o\
6E 69 6C 20 65 68 74 20 6E 69 20 6B 63 61 62  00103
               6F 20 74 73 69 6C 20 64 65 6B  00112
73 65 69 72 74 6E 65 20 65 68 63 61 63 20 66  0011C           .ASCII  \f cache entries\
                                              0012B           .BLKB   1
                                   00000037.  0012C P.AAK:    .LONG   55
                                   00000000'  00130           .ADDRESS P.AAL
6F 74 20 6B 63 61 42 20 70 6D 75 6A 20 20 42  00134 P.AAN:    .ASCII  \B  jump Back to the first cache entry in\
68 63 61 63 20 74 73 72 69 66 20 65 68 74 20  00143
               6E 69 20 79 72 74 6E 65 20 65  00152
74 65 6B 63 75 62 20 74 6E 65 72 72 75 63 20  0015C           .ASCII  \ current bucket\
                                              0016B           .BLKB   1
                                   00000037.  0016C P.AAM:    .LONG   55
                                   00000000'  00170           .ADDRESS P.AAN
72 4F 20 6F 74 20 6E 72 75 74 65 72 20 20 4F  00174 P.AAP:    .ASCII  \O  return to Origin; first entry of firs\
74 6E 65 20 74 73 72 69 66 20 3B 6E 69 67 69  00183
               73 72 69 66 20 66 6F 20 79 72  00192
                  20 74 65 6B 63 75 62 20 74  0019C           .ASCII  \t bucket \
                                              001A5           .BLKB   3
                                   00000031.  001A8 P.AAO:    .LONG   49
                                   00000000'  001AC           .ADDRESS P.AAP
30 31 20 6E 77 6F 64 20 70 6D 75 4A 20 20 4A  001B0 P.AAR:    .ASCII  \J  Jump down 10 buckets in hash table\
73 61 68 20 6E 69 20 73 74 65 6B 63 75 62 20  001BF
                        65 6C 62 61 74 20 68  001CE
                                              001D5           .BLKB   3
                                   00000025.  001D8 P.AAQ:    .LONG   37
                                   00000000'  001DC           .ADDRESS P.AAR
63 6F 6C 62 20 79 61 6C 70 73 69 44 20 20 44  001E0 P.AAT:    .ASCII  \D  Display block of current cache entry\
61 63 20 74 6E 65 72 72 75 63 20 66 6F 20 6B  001EF
                  79 72 74 6E 65 20 65 68 63  001FE
```

```
                                                       00207              .BLKB    1
                                          00000027     00208 P.AAS:       .LONG    39                                          ;
                                          00000000'    0020C              .ADDRESS P.AAT                                       ;
72 62 69 6C 20 79 61 6C 70 73 69 44 20 20 48           00210 P.AAV:       .ASCII   \H  Display library header block\          ;
63 6F 6C 62 20 72 65 64 61 65 68 20 79 72 61           0021F                                                               ;
                                          6B           0022E                                                               ;
                                                       0022F              .BLKB    1
                                          0000001F     00230 P.AAU:       .LONG    31                                          ;
                                          00000000'    00234              .ADDRESS P.AAV                                       ;
20 61 20 72 6F 66 20 79 72 65 75 71 20 20 4C           00238 P.AAX:       .ASCII   \L  query for a VBN to locate in cache\    ;
69 20 65 74 61 63 6F 6C 20 6F 74 20 4E 42 56           00247                                                               ;
               65 68 63 61 63 20 6E                    00256                                                               ;
                                                       0025D              .BLKB    3
                                          00000025     00260 P.AAW:       .LONG    37                                          ;
                                          00000000'    00264              .ADDRESS P.AAX                                       ;
20 61 20 72 6F 66 20 79 72 65 75 71 20 20 47           00268 P.AAZ:       .ASCII   \G  query for a VBN to get from library a\ ;
6D 6F 72 66 20 74 65 67 20 6F 74 20 4E 42 56           00277                                                               ;
            61 20 79 72 61 72 62 69 6C 20              00286                                                               ;
               65 68 63 61 63 20 64 6E                 00290              .ASCII   \nd cache\                                  ;
                                          00000030     00298 P.AAY:       .LONG    48                                          ;
                                          00000000'    0029C              .ADDRESS P.AAZ                                       ;
62 76 20 74 58 65 6E 20 70 6D 75 64 20 20 58           002A0 P.ABB:       .ASCII   \X  dump neXt vbn\                          ;
                                          6E           002AF                                                               ;
                                          00000010     002B0 P.ABA:       .LONG    16                                          ;
                                          00000000'    002B4              .ADDRESS P.ABB                                       ;
20 65 68 74 20 74 6E 69 72 70 65 52 20 20 52           002B8 P.ABD:       .ASCII   \R  Reprint the screen\                     ;
                           6E 65 65 72 63 73           002C7                                                               ;
                                                       002CD              .BLKB    3
                                          00000015     002D0 P.ABC:       .LONG    21                                          ;
                                          00000000'    002D4              .ADDRESS P.ABD                                       ;
65 6C 63 20 64 6E 61 20 74 69 75 51 20 20 51           002D8 P.ABF:       .ASCII   \Q  Quit and clear the screen\             ;
      6E 65 65 72 63 73 20 65 68 74 20 72 61           002E7                                                               ;
                                          0000001C     002F4 P.ABE:       .LONG    28                                          ;
                                          00000000'    002F8              .ADDRESS P.ABF                                       ;

                                                                          .PSECT   $OWN$,NOEXE,2

                                                       00000 CUR_VBN:.BLKB    4
                                          20           00004 FAOSTRING1:
                                                                          .BYTE    32                                          ;
21 20 6B 63 6F 6C 62 20 78 65 64 6E 49 2F 21           00005              .ASCII   \!/Index block !XL at address !XL\          ;
21 20 73 73 65 72 64 64 61 20 74 61 20 4C 58           00014                                                               ;
                                       4C 58           00023                                                               ;
                                                       00025              .BLKB    3
                                          1C           00028 FAOSTRING2:
                                                                          .BYTE    28                                          ;
20 2C 4C 58 21 20 3D 20 74 6E 65 72 61 70 28           00029              .ASCII   \(parent = !XL, used = !XW)!/\              ;
   2F 21 29 57 58 21 20 3D 20 64 65 73 75              00038                                                               ;
                                                       00045              .BLKB    3
                                          13           00048 FAOSTRING3:
                                                                          .BYTE    19                                          ;
20 20 57 58 21 20 4C 58 21 20 20 20 57 58 21           00049              .ASCII   \!XW   !XL !XW   !AC\                      ;
                              43 41 21 20              00058                                                               ;
                                          13           0005C FAOSTRING4:
                                                                          .BYTE    19                                          ;
20 20 57 58 21 20 4C 58 21 20 20 20 57 58 21           0005D              .ASCII   \!XW   !XL !XW   !XL\                      ;
                              4C 58 21 20              0006C                                                               ;
```

```
                                                         13  00070 FAODMPBLK:
                                                                                  .BYTE   19
4C 58 21 20 4C 58 21 20 4C 58 21 20 4C 58 21  00071             .ASCII  \!XL !XL !XL !XL !AF\                       ;
                                     46 41 21 20  00080                                                            ;
                                              0C  00084 FAOHASH:.BYTE   12                                          ;
         4C 58 21 20 20 20 20 20 20 20 20 20  00085             .ASCII  \         !XL\                              ;
                                                  00091             .BLKB   3
                                              07  00094 FAOHASHENTRY:
                                                                    .BYTE   7                                       ;
                           4C 58 21 20 4C 58 21  00095             .ASCII  \!XL !XL\                                ;
                                              24  0009C FAOBLOCK:
                                                                    .BYTE   36                                      ;
4C 58 21 20 4C 58 21 20 4C 58 21 20 4C 58 21  0009D             .ASCII  \!XL !XL !XL !XL !XL !XL !XL !XL  !AF\      ;
58 21 20 4C 58 21 20 4C 58 21 20 4C 58 21 20  000AC                                                                ;
                           46 41 21 20 20 4C  000BB                                                                ;
                                                  000C1             .BLKB   3
                                              04  000C4 FAOENTRY:
                                                                    .BYTE   4                                       ;
                                  20 4C 58 21  000C5             .ASCII  \!XL \                                     ;
                                                  000C9             .BLKB   3
                                                  000CC DESC:       .BLKB   8
                                                  000D4 BUF:        .BLKB   80

                                                        HELP_DESC_0=            P.AAA
                                                        HELP_DESC_1=            P.AAC
                                                        HELP_DESC_2=            P.AAE
                                                        HELP_DESC_3=            P.AAG
                                                        HELP_DESC_4=            P.AAI
                                                        HELP_DESC_5=            P.AAK
                                                        HELP_DESC_6=            P.AAM
                                                        HELP_DESC_7=            P.AAO
                                                        HELP_DESC_8=            P.AAQ
                                                        HELP_DESC_9=            P.AAS
                                                        HELP_DESC_10=           P.AAU
                                                        HELP_DESC_11=           P.AAW
                                                        HELP_DESC_12=           P.AAY
                                                        HELP_DESC_13=           P.ABA
                                                        HELP_DESC_14=           P.ABC
                                                        HELP_DESC_15=           P.ABE
                                                                    .EXTRN  FIND_BLOCK, FIND_INDEX
                                                                    .EXTRN  LOOKUP_CACHE, SCR$SET_CURSOR
                                                                    .EXTRN  SCR$GET_SCREEN, SCR$ERASE_LINE
                                                                    .EXTRN  SCR$ERASE_PAGE, SCR$PUT_BUFFER
                                                                    .EXTRN  SCR$PUT_SCREEN, SCR$SET_BUFFER
                                                                    .EXTRN  OTS$CVT_TZ_L, LIB$PUT_OUTPUT
                                                                    .EXTRN  LBR$GL_CONTROL, SYS$FAOL

                                                                    .PSECT  $CODE$,NOWRT,2

                                                      0004 00000 FAO_BUFFER:
                                                                    .WORD   Save R2                                 ; 0882
                       52       0000'   CF  9E  00002             MOVAB   DESC, R2                                 ;
                       5E          08   C2  00007             SUBL2   #8, SP                                   ;
                       6E          04   BC  9B  0000A             MOVZBW  @CTRSTR, FAODESC                         ; 0894
      04    AE     04  AC          01   C1  0000E             ADDL3   #1, CTRSTR, FAODESC+4                    ; 0895
                       62          50   8F  9B  00014             MOVZBW  #80, DESC                                ; 0896
                 04    A2          08   A2  9E  00018             MOVAB   BUF, DESC+4                              ; 0897
```

```
                        08   AC  9F 0001D       PUSHAB  ARGS                    ; 0898
                             52  DD 00020       PUSHL   R2                      ;
                             52  DD 00022       PUSHL   R2                      ;
                        0C   AE  9F 00024       PUSHAB  FAODESC                 ;
        00000000G  00        04  FB 00027       CALLS   #4, SYS$FAOL            ;
                   50        62  9E 0002E       MOVAB   DESC, R0                ; 0899
                                 04 00031       RET                             ; 0900

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 0000
```

```
  2 %SBTTL 'dump_block2';
  2
  2 ROUTINE dump_block2 (index, vbn) =
  3 BEGIN
  3 LOCAL
  3      entry,
  3      index_desc: REF BBLOCK,       ! Index descriptor
  3      index_block: REF BBLOCK;      ! Index block address
  3
  3 index_desc = .lbr$gl_control [lbr$l_hdrptr] + lhd$c_idxdesc
  3              + (.index-1)*idd$c_length;
  3
  3 perform (find_index (.vbn, index_block));
  3
  3 write(faostring1,.vbn,.index_block);
P 3 write(faostring2,
P 3         .index_block [index$l_parent],
  3         .index_block [index$w_used]);
  3
  3 entry = .index_block+index$c_entries;
  3 WHILE (.entry LSS .index_block+index$c_entries+.index_block[index$w_used]-1) DO
  4      BEGIN
  4      MAP entry: REF BBLOCK;
  4      IF .index_desc [idd$v_ascii]          ! If ASCII keys,
  4      THEN
P 4         write(faostring3,
P 4                 .entry - .index_block,
P 4                 .entry [idx$l_vbn], .entry [idx$w_offset],
  4                 entry [idx$b_keylen])
  4      ELSE
P 4         write(faostring4,
P 4                 .entry - .index_block,
P 4                 .entry [idx$l_vbn], .entry [idx$w_offset],
  4                 .entry [idx$l_keyid]);
  4      entry = .entry + idx$c_rfaplsbyt + .entry [idx$b_keylen];
  3      END;
  3
  3 entry = .index_block+index$c_entries;
  3 WHILE (.entry LSS .index_block+index$c_entries+.index_block[index$w_used]-1) DO
  4      BEGIN
  4      MAP entry: REF BBLOCK;
  4      IF .entry [idx$w_offset] EQL rfa$c_index    ! If subindex,
  4      THEN
  4         dump_block2(.index, .entry [idx$l_vbn]);
  4      entry = .entry + idx$c_rfaplsbyt + .entry [idx$b_keylen];
  3      END;
  3
  3 RETURN true;
  2 END;


                              0FFC 00000 DUMP_BLOCK2:
                                               .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11     ; 0903
              59      C9   AF   9E 00002       MOVAB    FAO_BUFFER, R9                           ;
```

```
                   58 00000000G  00  9E 00006          MOVAB   LIB$PUT_OUTPUT, R8                ;
                   5E            04  C2 0000D          SUBL2   #4, SP                            ;
                   51      0000G CF  D0 00010          MOVL    LBR$GL_CONTROL, R1                ; 0910
                   50         04 AC  D0 00015          MOVL    INDEX, R0                         ; 0911
                   57    0A B140 7E  00019             MOVAQ   @10(R1)[R0], INDEX_DESC           ;
                   57       00BC C7  9E 0001E          MOVAB   188(R7), INDEX_DESC               ;
                   51            6E  9E 00023          MOVAB   INDEX_BLOCK, R1                   ; 0913
                   50         08 AC  D0 00026          MOVL    VBN, R0                           ;
                               0000G 30 0002A          BSBW    FIND_INDEX                        ;
                   01            50  E8 0002D          BLBS    STATUS, 1$                        ;
                                     04 00030          RET                                       ;
                   53            6E  D0 00031  1$:     MOVL    INDEX_BLOCK, R3                   ; 0915
                                 53  DD 00034          PUSHL   R3                                ;
                              08 AC  DD 00036          PUSHL   VBN                               ;
                           0000' CF  9F 00039          PUSHAB  FAOSTRING1                        ;
                   69            03  FB 0003D          CALLS   #3, FAO_BUFFER                    ;
                                 50  DD 00040          PUSHL   R0                                ;
                   68            01  FB 00042          CALLS   #1, LIB$PUT_OUTPUT                ;
                   54            63  3C 00045          MOVZWL  (R3), R4                          ; 0918
                                 54  DD 00048          PUSHL   R4                                ;
                              02 A3  DD 0004A          PUSHL   2(R3)                             ;
                           0000' CF  9F 0004D          PUSHAB  FAOSTRING2                        ;
                   69            03  FB 00051          CALLS   #3, FAO_BUFFER                    ;
                                 50  DD 00054          PUSHL   R0                                ;
                   68            01  FB 00056          CALLS   #1, LIB$PUT_OUTPUT                ;
                   56         0C A3  9E 00059          MOVAB   12(R3), R6                        ; 0920
                   52            56  D0 0005D          MOVL    R6, ENTRY                         ;
                   55    0B A443 9E  00060     2$:     MOVAB   11(R4)[R3], R5                    ; 0921
                   55            52  D1 00065          CMPL    ENTRY, R5                         ;
                                 3A  18 00068          BGEQ    5$                                ;
         50        52            53  C3 0006A          SUBL3   R3, ENTRY, R0                     ; 0929
                   11            67  E9 0006E          BLBC    (INDEX_DESC), 3$                  ; 0924
                              06 A2  9F 00071          PUSHAB  6(ENTRY)                          ; 0929
                   7E         04 A2  3C 00074          MOVZWL  4(ENTRY), -(SP)                   ;
                                 62  DD 00078          PUSHL   (ENTRY)                           ;
                                 50  DD 0007A          PUSHL   R0                                ;
                           0000' CF  9F 0007C          PUSHAB  FAOSTRING3                        ;
                                 0F  11 00080          BRB     4$                                ;
                              06 A2  DD 00082  3$:     PUSHL   6(ENTRY)                          ; 0934
                   7E         04 A2  3C 00085          MOVZWL  4(ENTRY), -(SP)                   ;
                                 62  DD 00089          PUSHL   (ENTRY)                           ;
                                 50  DD 0008B          PUSHL   R0                                ;
                           0000' CF  9F 0008D          PUSHAB  FAOSTRING4                        ;
                   69            05  FB 00091  4$:     CALLS   #5, FAO_BUFFER                    ;
                                 50  DD 00094          PUSHL   R0                                ;
                   68            01  FB 00096          CALLS   #1, LIB$PUT_OUTPUT                ;
                   50         06 A2  9A 00099          MOVZBL  6(ENTRY), R0                      ; 0935
                   52    07 A042 9E  0009D             MOVAB   7(R0)[ENTRY], ENTRY               ;
                                 BC  11 000A2          BRB     2$                                ; 0921
                   52            56  D0 000A4  5$:     MOVL    R6, ENTRY                         ; 0938
                   55            52  D1 000A7  6$:     CMPL    ENTRY, R5                         ; 0939
                                 1C  18 000AA          BGEQ    8$                                ;
            FFFF   8F         04 A2  B1 000AC          CMPW    4(ENTRY), #65535                  ; 0942
                                 09  12 000B2          BNEQ    7$                                ;
                                 62  DD 000B4          PUSHL   (ENTRY)                           ; 0944
                              04 AC  DD 000B6          PUSHL   INDEX                             ;
              32   A9            02  FB 000B9          CALLS   #2, DUMP_BLOCK2                   ;
```

```
      50        06   A2  9A 000BD 7$:     MOVZBL  6(ENTRY), R0                              ; 0945
      52        07 A042  9E 000C1         MOVAB   7(R0)[ENTRY], ENTRY                       ;
                     DF  11 000C6         BRB     6$                                        ; 0939
      50             01  D0 000C8 8$:     MOVL    #1, R0                                    ; 0948
                         04 000CB         RET                                               ; 0949

; Routine Size:  204 bytes,    Routine Base:  $CODE$ + 0032
```

```
%SBTTL 'dump_block';

ROUTINE dump_block (index, vbn) =
BEGIN
LOCAL
     index_desc: REF BBLOCK,        ! Index descriptor
     index_block: REF BBLOCK;       ! Index block address

index_desc = .lbr$gl_control [lbr$l_hdrptr] + lhd$c_idxdesc
                  + (.index-1)*idd$c_length;

perform (find_index (.vbn, index_block));

write(faostring1,.vbn,.index_block);
write(faostring2,
     .index_block [index$l_parent],
     .index_block [index$w_used]);

INCRU entry FROM .index_block+index$c_entries
     TO .index_block+index$c_entries+.index_block [index$w_used]-1
     BY idx$c_length + .index_desc [idd$w_keylen]
DO
     BEGIN
     MAP entry: REF BBLOCK;
     IF .index_desc [idd$v_ascii]          ! If ASCII keys,
     THEN
          write(faostring3,
                  .entry - .index_block,
                  .entry [idx$l_vbn], .entry [idx$w_offset],
                  entry [idx$b_keylen])
     ELSE
          write(faostring4,
                  .entry - .index_block,
                  .entry [idx$l_vbn], .entry [idx$w_offset],
                  .entry [idx$l_keyid]);
     END;

INCRU entry FROM .index_block+index$c_entries
     TO .index_block+index$c_entries+.index_block [index$w_used]-1
     BY idx$c_length + .index_desc [idd$w_keylen]
DO
     BEGIN
     MAP entry: REF BBLOCK;
     IF .entry [idx$w_offset] EQL rfa$c_index    ! If subindex,
     THEN
          dump_block(.index, .entry [idx$l_vbn]);
     END;

RETURN true;
END;
```

```
                    OFFC 00000 DUMP_BLOCK:
                                 .WORD    Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 0952
```

```
                 58      FEFC   CF  9E 00002        MOVAB   FAO_BUFFER, R8                          ;
                 57 00000000G   00  9E 00007        MOVAB   LIB$PUT_OUTPUT, R7                      ;
                 5E             04  C2 0000E        SUBL2   #4, SP                                  ;
                 51     0000G   CF  D0 00011        MOVL    LBR$GL_CONTROL, R1                      ; 0958
                 50        04   AC  D0 00016        MOVL    INDEX, R0                               ; 0959
                 54     0A B140  7E 0001A        MOVAQ   @10(R1)[R0], INDEX_DESC                 ;
                 54      00BC   C4  9E 0001F        MOVAB   188(R4), INDEX_DESC                     ;
                 51             6E  9E 00024        MOVAB   INDEX_BLOCK, R1                         ; 0961
                 50        08   AC  D0 00027        MOVL    VBN, R0                                 ;
                           0000G  30 0002B        BSBW    FIND_INDEX                              ;
                 01             50  E8 0002E        BLBS    STATUS, 1$                              ;
                                    04 00031        RET                                             ;
                 53             6E  D0 00032 1$:    MOVL    INDEX_BLOCK, R3                         ; 0963
                                53  DD 00035        PUSHL   R3                                      ;
                           08   AC  DD 00037        PUSHL   VBN                                     ;
                        0000'   CF  9F 0003A        PUSHAB  FAOSTRING1                              ;
                 68             03  FB 0003E        CALLS   #3, FAO_BUFFER                          ;
                                50  DD 00041        PUSHL   R0                                      ;
                 67             01  FB 00043        CALLS   #1, LIB$PUT_OUTPUT                      ;
                 52             63  3C 00046        MOVZWL  (R3), R2                                ; 0966
                                52  DD 00049        PUSHL   R2                                      ;
                           02   A3  DD 0004B        PUSHL   2(R3)                                   ;
                        0000'   CF  9F 0004E        PUSHAB  FAOSTRING2                              ;
                 68             03  FB 00052        CALLS   #3, FAO_BUFFER                          ;
                                50  DD 00055        PUSHL   R0                                      ;
                 67             01  FB 00057        CALLS   #1, LIB$PUT_OUTPUT                      ;
                 56      0B A243    9E 0005A        MOVAB   11(R2)[R3], R6                          ; 0969
                 55        02   A4  3C 0005F        MOVZWL  2(INDEX_DESC), R5                       ; 0970
                 55             06  C0 00063        ADDL2   #6, R5                                  ;
                 52        0C   A3  9E 00066        MOVAB   12(R3), ENTRY                           ;
                                32  11 0006A        BRB     5$                                      ;
      50         52             53  C3 0006C 2$:    SUBL3   R3, ENTRY, R0                           ; 0979
                 11             64  E9 00070        BLBC    (INDEX_DESC), 3$                        ; 0974
                           06   A2  9F 00073        PUSHAB  6(ENTRY)                                ; 0979
                 7E        04   A2  3C 00076        MOVZWL  4(ENTRY), -(SP)                         ;
                                62  DD 0007A        PUSHL   (ENTRY)                                 ;
                                50  DD 0007C        PUSHL   R0                                      ;
                        0000'   CF  9F 0007E        PUSHAB  FAOSTRING3                              ;
                                0F  11 00082        BRB     4$                                      ;
                           06   A2  DD 00084 3$:    PUSHL   6(ENTRY)                                ; 0984
                 7E        04   A2  3C 00087        MOVZWL  4(ENTRY), -(SP)                         ;
                                62  DD 0008B        PUSHL   (ENTRY)                                 ;
                                50  DD 0008D        PUSHL   R0                                      ;
                        0000'   CF  9F 0008F        PUSHAB  FAOSTRING4                              ;
                 68             05  FB 00093 4$:    CALLS   #5, FAO_BUFFER                          ;
                                50  DD 00096        PUSHL   R0                                      ;
                 67             01  FB 00098        CALLS   #1, LIB$PUT_OUTPUT                      ;
                 52             55  C0 0009B        ADDL2   R5, ENTRY                               ; 0968
                 56             52  D1 0009E 5$:    CMPL    ENTRY, R6                               ;
                                C9  1B 000A1        BLEQU   2$                                      ;
                 52        0C   A3  9E 000A3        MOVAB   12(R3), ENTRY                           ; 0989
                                15  11 000A7        BRB     8$                                      ;
        FFFF     8F        04   A2  B1 000A9 6$:    CMPW    4(ENTRY), #65535                        ; 0993
                                0A  12 000AF        BNEQ    7$                                      ;
                                62  DD 000B1        PUSHL   (ENTRY)                                 ; 0995
                           04   AC  DD 000B3        PUSHL   INDEX                                   ;
        00FE     C8             02  FB 000B6        CALLS   #2, DUMP_BLOCK                          ;
```

```
              52          55  C0 000BB 7$:      ADDL2   R5, ENTRY                              ; 0987
              56          52  D1 000BE 8$:      CMPL    ENTRY, R6                              ;
                          E6  1B 000C1          BLEQU   6$                                     ;
              50          01  D0 000C3          MOVL    #1, R0                                 ; 0998
                              04 000C6          RET                                            ; 0999

; Routine Size:  199 bytes,    Routine Base:  $CODE$ + 00FE
```

```
;   2 %SBTTL 'main body of LBR$DUMP_INDEX';
;   2
;   2 !
;   2 !       Main body of lbr$dump_index procedure
;   2 !
;   2 LOCAL
;   2     index_desc: REF BBLOCK;               ! Index descriptor
;   2
;   2 index_desc = .lbr$gl_control [lbr$l_hdrptr] + lhd$c_idxdesc
;   2            + (.index-1)*idd$c_length;
;   2 IF .index_desc [idd$l_vbn] EQL 0        ! If empty index,
;   2 THEN
;   2     RETURN true;                          ! return immediately
;   2
;   2 IF .index_desc [idd$v_varlenidx]
;   2 THEN
;   3     perform(dump_block2 (.index, .index_desc [idd$l_vbn]))
;   2 ELSE
;   2     perform(dump_block (.index, .index_desc [idd$l_vbn]));
;   2
;   2 RETURN true;
;   2
;   1 END;    ! lbr$dump_index ()


                                     0000 00000        .ENTRY  LBR$DUMP_INDEX, Save nothing          ; 0856
              51     0000G   CF   D0 00002             MOVL    LBR$GL_CONTROL, R1                    ; 1008
              50        04   AC   D0 00007             MOVL    INDEX, R0                             ; 1009
              51     0A B140   7E 0000B                MOVAQ   @10(R1)[R0], INDEX_DESC               ;
              51      00BC   C1   9E 00010             MOVAB   188(R1), INDEX_DESC                   ;
              50        04   A1   D0 00015             MOVL    4(INDEX_DESC), R0                     ; 1010
                             1D   13 00019             BEQL    3$                                    ;
      0C      61             02   E1 0001B             BBC     #2, (INDEX_DESC), 1$                  ; 1014
                             50   DD 0001F             PUSHL   R0                                    ; 1016
                        04   AC   DD 00021             PUSHL   INDEX                                 ;
       FE44   CF             02   FB 00024             CALLS   #2, DUMP_BLOCK2                       ;
                             0A   11 00029             BRB     2$                                    ;
                             50   DD 0002B 1$:         PUSHL   R0                                    ; 1018
                        04   AC   DD 0002D             PUSHL   INDEX                                 ;
       FF04   CF             02   FB 00030             CALLS   #2, DUMP_BLOCK                        ;
              03             50   E9 00035 2$:         BLBC    STATUS, 4$                            ;
              50             01   D0 00038 3$:         MOVL    #1, R0                                ; 1020
                                  04 0003B 4$:         RET                                           ; 1022

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 01C5
```

```
  1 %SBTTL 'LBR$DUMP_INDEXBLOCKS';
  1
  1 GLOBAL ROUTINE lbr$dump_indexblocks (index) =
  1
  1 !---
  1 !
  1 !       Dump the entire index structure in both hex and ASCII
  1 !       to SYS$OUTPUT.
  1 !
  1 ! Inputs:
  1 !
  1 !       index = Primary index number
  1 !
  1 ! Outputs:
  1 !
  1 !       None
  1 !
  1 !---
  1
  2 BEGIN
  2 MACRO
M 2     write (string) =
M 2         lib$put_output (fao_buffer (string
  2         %IF %LENGTH GTR 1 %THEN ,%REMAINING %FI))%;
  2
  2 EXTERNAL ROUTINE
  2     lib$put_output : ADDRESSING_MODE (GENERAL);
  2
  2 ROUTINE fao_buffer (ctrstr,args) =
  3 BEGIN
  3 OWN
  3     desc :      BBLOCK [dsc$c_s_bln],   ! Result descriptor
  3     buf :       VECTOR [80, BYTE];      ! Output buffer
  3 MAP
  3     ctrstr :    REF VECTOR [,BYTE],
  3     args :      VECTOR [4];
  3
  3 LOCAL
  3     faodesc : BBLOCK [dsc$c_s_bln];
  3
  3 faodesc [dsc$w_length] = .ctrstr [0];
  3 faodesc [dsc$a_pointer] = ctrstr [1];
  3 desc [dsc$w_length] = 80;                       ! Set up result descriptor
  3 desc [dsc$a_pointer] = buf;
  3 $faol (ctrstr=faodesc, outlen=desc, outbuf=desc, prmlst=args);
  3 RETURN desc;
  2 END;

                                        .PSECT  $OWN$,NOEXE,2

                        00124 DESC:     .BLKB   8
                        0012C BUF:      .BLKB   80
```

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

                                  0004 00000 FAO_BUFFER:
                                                       .WORD   Save R2                          ; 1051
                     52     0000'  CF   9E 00002       MOVAB   DESC, R2                         ;
                     5E            08   C2 00007       SUBL2   #8, SP                           ;
                     6E       04   BC   9B 0000A       MOVZBW  @CTRSTR, FAODESC                 ; 1063
      04   AE    04  AC            01   C1 0000E       ADDL3   #1, CTRSTR, FAODESC+4            ; 1064
                     62       50   8F   9B 00014       MOVZBW  #80, DESC                        ; 1065
                 04  A2       08   A2   9E 00018       MOVAB   BUF, DESC+4                      ; 1066
                              08   AC   9F 0001D       PUSHAB  ARGS                             ; 1067
                                   52   DD 00020       PUSHL   R2                               ;
                                   52   DD 00022       PUSHL   R2                               ;
                              0C   AE   9F 00024       PUSHAB  FAODESC                          ;
          00000000G  00            04   FB 00027       CALLS   #4, SYS$FAOL                     ;
                     50            62   9E 0002E       MOVAB   DESC, R0                         ; 1068
                                   04 00031            RET                                      ; 1069

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 0201
```

```
%SBTTL 'dump_wholeblk2';

ROUTINE dump_wholeblk2 (index, vbn) =
BEGIN
LOCAL
    entry,
    index_desc: REF BBLOCK,        ! Index descriptor
    index_block: REF BBLOCK;       ! Index block address

index_desc = .lbr$gl_control [lbr$l_hdrptr] + lhd$c_idxdesc
            + (.index-1)*idd$c_length;

perform (find_index (.vbn, index_block));

write(faostring1,.vbn,.index_block);
INCRU i FROM 0 TO 511 BY 16 DO
    write(faodmpblk,
        .index_block [.i+12,0,32,0], .index_block [.i+8,0,32,0],
        .index_block [.i+4,0,32,0], .index_block [.i,0,32,0],
        16, index_block [.i,0,0,0] );

entry = .index_block+index$c_entries;
WHILE (.entry LSS .index_block+index$c_entries+.index_block[index$w_used]-1) DO
    BEGIN
    MAP entry: REF BBLOCK;
    IF .index_desc [idd$v_ascii]           ! If ASCII keys,
    THEN
        write(faostring3,
            .entry - .index_block,
            .entry [idx$l_vbn], .entry [idx$w_offset],
            entry [idx$b_keylen])
    ELSE
        write(faostring4,
            .entry - .index_block,
            .entry [idx$l_vbn], .entry [idx$w_offset],
            .entry [idx$l_keyid]);
    entry = .entry + idx$c_rfaplsbyt + .entry [idx$b_keylen];
    END;

entry = .index_block+index$c_entries;
WHILE (.entry LSS .index_block+index$c_entries+.index_block[index$w_used]-1) DO
    BEGIN
    MAP entry: REF BBLOCK;
    IF .entry [idx$w_offset] EQL rfa$c_index    ! If subindex,
    THEN
        dump_wholeblk2(.index, .entry [idx$l_vbn]);
    entry = .entry + idx$c_rfaplsbyt + .entry [idx$b_keylen];
    END;

RETURN true;
END;
```

OFFC 00000 DUMP_WHOLEBLK2:

```
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1072
                58        C9    AF  9E 00002        MOVAB   FAO_BUFFER, R8                              ;
                57 00000000G    00  9E 00006        MOVAB   LIB$PUT_OUTPUT, R7                          ;
                5E              04  C2 0000D        SUBL2   #4, SP                                      ;
                51     0000G    CF  D0 00010        MOVL    LBR$GL_CONTROL, R1                          ; 1079
                50        04    AC  D0 00015        MOVL    INDEX, R0                                   ; 1080
                56     0A B140  7E  00019           MOVAQ   @10(R1)[R0], INDEX_DESC                     ;
                56      00BC    C6  9E 0001E        MOVAB   188(R6), INDEX_DESC                         ;
                51              6E  9E 00023        MOVAB   INDEX_BLOCK, R1                             ; 1082
                50        08    AC  D0 00026        MOVL    VBN, R0                                     ;
                          0000G 30  0002A           BSBW    FIND_INDEX                                  ;
                01              50  E8 0002D        BLBS    STATUS, 1$                                  ;
                                    04 00030        RET                                                 ;
                53              6E  D0 00031 1$:    MOVL    INDEX_BLOCK, R3                             ; 1084
                                53  DD 00034        PUSHL   R3                                          ;
                          08    AC  DD 00036        PUSHL   VBN                                         ;
                        0000'   CF  9F 00039        PUSHAB  FAOSTRING1                                  ;
                68              03  FB 0003D        CALLS   #3, FAO_BUFFER                              ;
                                50  DD 00040        PUSHL   R0                                          ;
                67              01  FB 00042        CALLS   #1, LIB$PUT_OUTPUT                          ;
                                52  D4 00045        CLRL    I                                           ; 1085
                          6243      9F 00047 2$:    PUSHAB  (I)[R3]                                     ; 1089
                                10  DD 0004A        PUSHL   #16                                         ;
                          6243      9F 0004C        PUSHAB  (I)[R3]                                     ;
                                9E  DD 0004F        PUSHL   @(SP)+                                      ;
                       04 A243      9F 00051        PUSHAB  4(I)[R3]                                    ;
                                9E  DD 00055        PUSHL   @(SP)+                                      ;
                       08 A243      9F 00057        PUSHAB  8(I)[R3]                                    ;
                                9E  DD 0005B        PUSHL   @(SP)+                                      ;
                       0C A243      9F 0005D        PUSHAB  12(I)[R3]                                   ;
                                9E  DD 00061        PUSHL   @(SP)+                                      ;
                        0000'   CF  9F 00063        PUSHAB  FAODMPBLK                                   ;
                68              07  FB 00067        CALLS   #7, FAO_BUFFER                              ;
                                50  DD 0006A        PUSHL   R0                                          ;
                67              01  FB 0006C        CALLS   #1, LIB$PUT_OUTPUT                          ;
                52              10  C0 0006F        ADDL2   #16, I                                      ;
       000001FF 8F              52  D1 00072        CMPL    I, #511                                     ;
                                CC  1B 00079        BLEQU   2$                                          ;
                55        0C    A3  9E 0007B        MOVAB   12(R3), R5                                  ; 1091
                52              55  D0 0007F        MOVL    R5, ENTRY                                   ;
                50              63  3C 00082 3$:    MOVZWL  (R3), R0                                    ; 1092
                54     0B A043      9E 00085        MOVAB   11(R0)[R3], R4                              ;
                54              52  D1 0008A        CMPL    ENTRY, R4                                   ;
                                3A  18 0008D        BGEQ    6$                                          ;
     50         52              53  C3 0008F        SUBL3   R3, ENTRY, R0                               ; 1100
                11              66  E9 00093        BLBC    (INDEX_DESC), 4$                            ; 1095
                          06    A2  9F 00096        PUSHAB  6(ENTRY)                                    ; 1100
                7E        04    A2  3C 00099        MOVZWL  4(ENTRY), -(SP)                             ;
                                62  DD 0009D        PUSHL   (ENTRY)                                     ;
                                50  DD 0009F        PUSHL   R0                                          ;
                        0000'   CF  9F 000A1        PUSHAB  FAOSTRING3                                  ;
                                0F  11 000A5        BRB     5$                                          ;
                          06    A2  DD 000A7 4$:    PUSHL   6(ENTRY)                                    ; 1105
                7E        04    A2  3C 000AA        MOVZWL  4(ENTRY), -(SP)                             ;
                                62  DD 000AE        PUSHL   (ENTRY)                                     ;
                                50  DD 000B0        PUSHL   R0                                          ;
                        0000'   CF  9F 000B2        PUSHAB  FAOSTRING4                                  ;
```

```
                 68         05 FB 000B6 5$:    CALLS   #5, FAO_BUFFER                ;
                            50 DD 000B9        PUSHL   R0                            ;
                 67         01 FB 000BB        CALLS   #1, LIB$PUT_OUTPUT            ;
                 50     06  A2 9A 000BE        MOVZBL  6(ENTRY), R0                  ; 1106
                 52  07 A042 9E 000C2          MOVAB   7(R0)[ENTRY], ENTRY           ;
                            B9 11 000C7        BRB     3$                            ; 1092
                 52         55 D0 000C9 6$:    MOVL    R5, ENTRY                     ; 1109
                 54         52 D1 000CC 7$:    CMPL    ENTRY, R4                     ; 1110
                            1C 18 000CF        BGEQ    9$                            ;
        FFFF     8F     04  A2 B1 000D1        CMPW    4(ENTRY), #65535              ; 1113
                            09 12 000D7        BNEQ    8$                            ;
                            62 DD 000D9        PUSHL   (ENTRY)                       ; 1115
                        04  AC DD 000DB        PUSHL   INDEX                         ;
             32  A8         02 FB 000DE        CALLS   #2, DUMP_WHOLEBLK2            ;
                 50     06  A2 9A 000E2 8$:    MOVZBL  6(ENTRY), R0                  ; 1116
                 52  07 A042 9E 000E6          MOVAB   7(R0)[ENTRY], ENTRY           ;
                            DF 11 000EB        BRB     7$                            ; 1110
                 50         01 D0 000ED 9$:    MOVL    #1, R0                        ; 1119
                               04 000F0        RET                                   ; 1120

; Routine Size:  241 bytes,    Routine Base:  $CODE$ + 0233
```

```
  2 %SBTTL 'dump_wholeblk';
  2
  2 ROUTINE dump_wholeblk (index, vbn) =
  3 BEGIN
  3 LOCAL
  3     index_desc: REF BBLOCK,     ! Index descriptor
  3     index_block: REF BBLOCK;    ! Index block address
  3
  3 index_desc = .lbr$gl_control [lbr$l_hdrptr] + lhd$c_idxdesc
  3                 + (.index-1)*idd$c_length;
  3
  3 perform (find_index (.vbn, index_block));
  3
  3 write(faostring1,.vbn,.index_block);
  3 INCRU i FROM 0 TO 511 BY 16 DO
P 3     write(faodmpblk,
P 3         .index_block [.i+12,0,32,0], .index_block [.i+8,0,32,0],
P 3         .index_block [.i+4,0,32,0], .index_block [.i,0,32,0],
  3         16, index_block [.i,0,0,0] );
  3
  3 INCRU entry FROM .index_block+index$c_entries
  3         TO .index_block+index$c_entries+.index_block [index$w_used]-1
  3         BY idx$c_length + .index_desc [idd$w_keylen]
  3 DO
  4     BEGIN
  4     MAP entry: REF BBLOCK;
  4     IF .index_desc [idd$v_ascii]         ! If ASCII keys,
  4     THEN
P 4         write(faostring3,
P 4                 .entry - .index_block,
P 4                 .entry [idx$l_vbn], .entry [idx$w_offset],
  4                 entry [idx$b_keylen])
  4     ELSE
P 4         write(faostring4,
P 4                 .entry - .index_block,
P 4                 .entry [idx$l_vbn], .entry [idx$w_offset],
  4                 .entry [idx$l_keyid]);
  3     END;
  3
  3 INCRU entry FROM .index_block+index$c_entries
  3         TO .index_block+index$c_entries+.index_block [index$w_used]-1
  3         BY idx$c_length + .index_desc [idd$w_keylen]
  3 DO
  4     BEGIN
  4     MAP entry: REF BBLOCK;
  4     IF .entry [idx$w_offset] EQL rfa$c_index    ! If subindex,
  4     THEN
  4         dump_wholeblk(.index, .entry [idx$l_vbn]);
  3     END;
  3
  3 RETURN true;
  2 END;
```

```
                                OFFC 00000 DUMP_WHOLEBLK:
                                           .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11           ; 1123
              58      FED7   CF   9E 00002         MOVAB   FAO_BUFFER, R8                                 ;
              57 00000000G   00   9E 00007         MOVAB   LIB$PUT_OUTPUT, R7                             ;
              5E             04   C2 0000E         SUBL2   #4, SP                                         ;
              51      0000G  CF   D0 00011         MOVL    LBR$GL_CONTROL, R1                             ; 1129
              50        04   AC   D0 00016         MOVL    INDEX, R0                                      ; 1130
              54   0A B140   7E   0001A            MOVAQ   @10(R1)[R0], INDEX_DESC                        ;
              54      00BC   C4   9E 0001F         MOVAB   188(R4), INDEX_DESC                            ;
              51             6E   9E 00024         MOVAB   INDEX_BLOCK, R1                                ; 1132
              50        08   AC   D0 00027         MOVL    VBN, R0                                        ;
                         0000G    30 0002B         BSBW    FIND_INDEX                                     ;
              01             50   E8 0002E         BLBS    STATUS, 1$                                     ;
                                  04 00031         RET                                                    ;
              53             6E   D0 00032 1$:     MOVL    INDEX_BLOCK, R3                                ; 1134
                             53   DD 00035         PUSHL   R3                                             ;
                        08   AC   DD 00037         PUSHL   VBN                                            ;
                      0000'  CF   9F 0003A         PUSHAB  FAOSTRING1                                     ;
              68             03   FB 0003E         CALLS   #3, FAO_BUFFER                                 ;
                             50   DD 00041         PUSHL   R0                                             ;
              67             01   FB 00043         CALLS   #1, LIB$PUT_OUTPUT                             ;
                             52   D4 00046         CLRL    I                                              ; 1135
                           6243   9F 00048 2$:     PUSHAB  (I)[R3]                                        ; 1139
                             10   DD 0004B         PUSHL   #16                                            ;
                           6243   9F 0004D         PUSHAB  (I)[R3]                                        ;
                             9E   DD 00050         PUSHL   @(SP)+                                         ;
                        04 A243   9F 00052         PUSHAB  4(I)[R3]                                       ;
                             9E   DD 00056         PUSHL   @(SP)+                                         ;
                        08 A243   9F 00058         PUSHAB  8(I)[R3]                                       ;
                             9E   DD 0005C         PUSHL   @(SP)+                                         ;
                        0C A243   9F 0005E         PUSHAB  12(I)[R3]                                      ;
                             9E   DD 00062         PUSHL   @(SP)+                                         ;
                      0000'  CF   9F 00064         PUSHAB  FAODMPBLK                                      ;
              68             07   FB 00068         CALLS   #7, FAO_BUFFER                                 ;
                             50   DD 0006B         PUSHL   R0                                             ;
              67             01   FB 0006D         CALLS   #1, LIB$PUT_OUTPUT                             ;
              52             10   C0 00070         ADDL2   #16, I                                         ;
     000001FF 8F             52   D1 00073         CMPL    I, #511                                        ;
                             CC   1B 0007A         BLEQU   2$                                             ;
              50             63   3C 0007C         MOVZWL  (R3), R0                                       ; 1142
              56   0B A043   9E   0007F            MOVAB   11(R0)[R3], R6                                 ;
              55        02   A4   3C 00084         MOVZWL  2(INDEX_DESC), R5                              ; 1143
              55             06   C0 00088         ADDL2   #6, R5                                         ;
              52        0C   A3   9E 0008B         MOVAB   12(R3), ENTRY                                  ;
                             32   11 0008F         BRB     6$                                             ;
   50         52             53   C3 00091 3$:     SUBL3   R3, ENTRY, R0                                  ; 1152
              11             64   E9 00095         BLBC    (INDEX_DESC), 4$                               ; 1147
                        06   A2   9F 00098         PUSHAB  6(ENTRY)                                       ; 1152
              7E        04   A2   3C 0009B         MOVZWL  4(ENTRY), -(SP)                                ;
                             62   DD 0009F         PUSHL   (ENTRY)                                        ;
                             50   DD 000A1         PUSHL   R0                                             ;
                      0000'  CF   9F 000A3         PUSHAB  FAOSTRING3                                     ;
                             0F   11 000A7         BRB     5$                                             ;
                        06   A2   DD 000A9 4$:     PUSHL   6(ENTRY)                                       ; 1157
              7E        04   A2   3C 000AC         MOVZWL  4(ENTRY), -(SP)                                ;
                             62   DD 000B0         PUSHL   (ENTRY)                                        ;
                             50   DD 000B2         PUSHL   R0                                             ;
```

```
                           0000'  CF  9F 000B4      PUSHAB  FAOSTRING4                          ;
               68                 05  FB 000B8 5$:  CALLS   #5, FAO_BUFFER                      ;
                                  50  DD 000BB      PUSHL   R0                                  ;
               67                 01  FB 000BD      CALLS   #1, LIB$PUT_OUTPUT                  ;
               52                 55  C0 000C0      ADDL2   R5, ENTRY                           ; 1141
               56                 52  D1 000C3 6$:  CMPL    ENTRY, R6                           ;
                                  C9  1B 000C6      BLEQU   3$                                  ;
               52           0C    A3  9E 000C8      MOVAB   12(R3), ENTRY                       ; 1162
                                  15  11 000CC      BRB     9$                                  ;
        FFFF   8F           04    A2  B1 000CE 7$:  CMPW    4(ENTRY), #65535                    ; 1166
                                  0A  12 000D4      BNEQ    8$                                  ;
                                  62  DD 000D6      PUSHL   (ENTRY)                             ; 1168
                            04    AC  DD 000D8      PUSHL   INDEX                               ;
        0123   C8                 02  FB 000DB      CALLS   #2, DUMP_WHOLEBLK                   ;
               52                 55  C0 000E0 8$:  ADDL2   R5, ENTRY                           ; 1160
               56                 52  D1 000E3 9$:  CMPL    ENTRY, R6                           ;
                                  E6  1B 000E6      BLEQU   7$                                  ;
               50                 01  D0 000E8      MOVL    #1, R0                              ; 1171
                                      04 000EB      RET                                         ; 1172

; Routine Size:  236 bytes,    Routine Base:  $CODE$ + 0324
```

```
2 %SBTTL 'Main body of LBR$DUMP_INDEXBLOCKS';
2
2 !
2 !       Main body of lbr$dump_indexblocks procedure
2 !
2 LOCAL
2     index_desc: REF BBLOCK;               ! Index descriptor
2
2 index_desc = .lbr$gl_control [lbr$l_hdrptr] + lhd$c_idxdesc
2             + (.index-1)*idd$c_length;
2 IF .index_desc [idd$l_vbn] EQL 0        ! If empty index,
2 THEN
2     RETURN true;                         ! return immediately
2 IF .index_desc [idd$v_varlenidx]
2 THEN
3     perform( dump_wholeblk2 (.index, .index_desc [idd$l_vbn]))
2 ELSE
2     perform( dump_wholeblk (.index, .index_desc [idd$l_vbn]));
2
2 RETURN true;
2
1 END;    ! lbr$dump_indexblocks ()
```

```
                                  0000 00000        .ENTRY  LBR$DUMP_INDEXBLOCKS, Save nothing      ; 1025
                  51     0000G  CF  D0 00002        MOVL    LBR$GL_CONTROL, R1                     ; 1181
                  50        04  AC  D0 00007        MOVL    INDEX, R0                              ; 1182
                  51     0A B140  7E 0000B          MOVAQ   @10(R1)[R0], INDEX_DESC                ;
                  51      00BC  C1  9E 00010        MOVAB   188(R1), INDEX_DESC                    ;
                  50        04  A1  D0 00015        MOVL    4(INDEX_DESC), R0                      ; 1183
                                1D  13 00019        BEQL    3$                                     ;
        0C        61            02  E1 0001B        BBC     #2, (INDEX_DESC), 1$                   ; 1186
                                50  DD 0001F        PUSHL   R0                                     ; 1188
                            04  AC  DD 00021        PUSHL   INDEX                                  ;
             FDFA CF            02  FB 00024        CALLS   #2, DUMP_WHOLEBLK2                     ;
                                0A  11 00029        BRB     2$                                     ;
                                50  DD 0002B 1$:    PUSHL   R0                                     ; 1190
                            04  AC  DD 0002D        PUSHL   INDEX                                  ;
             FEDF CF            02  FB 00030        CALLS   #2, DUMP_WHOLEBLK                      ;
                  03            50  E9 00035 2$:    BLBC    STATUS, 4$                             ;
                  50            01  D0 00038 3$:    MOVL    #1, R0                                 ; 1192
                                  04 0003B 4$:      RET                                            ; 1194

; Routine Size:  60 bytes,    Routine Base:  $CODE$ + 0410
```

```
%SBTTL 'LBR$DUMP_CACHE';

GLOBAL ROUTINE lbr$dump_cache  =

!---
!
!       Interactively dump the cache to SYS$OUTPUT.
!
! Inputs:
!
!       From SYS$INPUT, enter any of the single letter commands with an <ESC>.
!           N    move (North) up the hash table
!           S    move (South) down to next bucket in hash table
!           E    move (East) over to the next cache entry in current hash bucket
!           W    move (West) back in the linked list of cache entries.
!           B    jump Back to the first cache entry in current bucket
!           J    Jump down 10 buckets in hash table
!           D    Display the VBN for the current cache entry
!           O    return to Origin; first entry of first bucket
!           L    query for a VBN to locate in cache
!           G    query for a VBN to get from library and cache
!           R    Reprint the screen
!           Q    Quit and clear the screen
!
! Outputs:
!
!       Displays requested contents of library cache on VT100
!
!---

BEGIN

FORWARD ROUTINE
    display_entries,     ! Print the cache entry list for current and next bucket
    display_entry_1,     ! Print the cache entry list for the current bucket
    display_entry_2,     ! Print the cache entry list for the next bucket
    display_block;       ! Locate memory address of current block and display it.

LITERAL
    first_entry = 0,     ! first entry in linked list of cache entries
    blk_col = 20,        ! column to print block dump in
    blk_lin = 9,         ! line to start block dump on
    level_1 = 1,         ! Cache entry list for current bucket
    input_lin = 24,      ! Place cursor for command input
    input_col = 0,       ! Place cursor for command input
    depth_offset = 1,    ! How many lines down to display cache entry list
    wid_offset = 20,     ! How many columns over to display cache entry list
    bcktoffset = 2,      ! How many lines down to display window of hash buckets
    entry_depth = 5,     ! Number of lines to display a level 1 cache entry
    entry_wid = 9,       ! Number of characters taken by display of one cache entry
    entries_wide = 12,   ! Max number of enties in cache entry linked list to fit on screen.
    half_hash_window = 5; ! Half size of window surrounding current bucket

ROUTINE fao_buffer (ctrstr,args) =
BEGIN
OWN
    desc :      BBLOCK [dsc$c_s_bln],   ! Result descriptor
```

```
;  534      1252  3      buf :        VECTOR [132, BYTE];      ! Output buffer
;  535      1253  3  MAP
;  536      1254  3      ctrstr :     REF VECTOR [,BYTE],
;  537      1255  3      args :       VECTOR [4];
;  538      1256  3
;  539      1257  3  LOCAL
;  540      1258  3      faodesc : BBLOCK [dsc$c_s_bln];
;  541      1259  3
;  542      1260  3  faodesc [dsc$w_length] = .ctrstr [0];
;  543      1261  3  faodesc [dsc$a_pointer] = ctrstr [1];
;  544      1262  3  desc [dsc$w_length] = 132;                          ! Set up result descriptor
;  545      1263  3  desc [dsc$a_pointer] = buf;
;  546      1264  3  $faol (ctrstr=faodesc, outlen=desc, outbuf=desc, prmlst=args);
;  547      1265  3  RETURN desc;
;  548      1266  2  END;


                                                              .PSECT  $OWN$,NOEXE,2

                                          0017C DESC:   .BLKB   8
                                          00184 BUF:    .BLKB   132


                                                              .PSECT  $CODE$,NOWRT,2

                                     0004 00000 FAO_BUFFER:
                                                        .WORD   Save R2                              ; 1248
                         52     0000'  CF  9E 00002     MOVAB   DESC, R2                             ;
                         5E            08  C2 00007     SUBL2   #8, SP                               ;
                         6E      04    BC  9B 0000A     MOVZBW  @CTRSTR, FAODESC                     ; 1260
         04    AE        04    AC      01  C1 0000E     ADDL3   #1, CTRSTR, FAODESC+4                ; 1261
                         62      84    8F  9B 00014     MOVZBW  #132, DESC                           ; 1262
                   04    A2      08    A2  9E 00018     MOVAB   BUF, DESC+4                          ; 1263
                                 08    AC  9F 0001D     PUSHAB  ARGS                                 ; 1264
                                       52  DD 00020     PUSHL   R2                                   ;
                                       52  DD 00022     PUSHL   R2                                   ;
                                 0C    AE  9F 00024     PUSHAB  FAODESC                              ;
                  00000000G  00        04  FB 00027     CALLS   #4, SYS$FAOL                         ;
                             50        62  9E 0002E     MOVAB   DESC, R0                             ; 1265
                                           04 00031     RET                                          ; 1266

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 044C


;  549          1267  2
```

```
;  551        1268  2 %SBTTL 'display_bucket';
;  552        1269  2
;  553        1270  2 ROUTINE display_bucket ( bucket, entry) =
;  554        1271  2 !+++
;  555        1272  2 !
;  556        1273  2 !       For the current bucket, display the linked list of cache entries
;  557        1274  2 !       starting at the offset ENTRY, and display the linked list for the
;  558        1275  2 !       the bucket following the current bucket.  Dump the block for the
;  559        1276  2 !       the current cache entry.
;  560        1277  2 !
;  561        1278  2 !---
;  562        1279  3 BEGIN
;  563        1280  3 perform ( display_entry_1 ( .bucket, .entry) );
;  564        1281  3 perform ( display_entry_2 ( .bucket+1, first_entry) );
;  565        1282  3 perform ( display_block (.bucket, .entry) );
;  566        1283  3 RETURN true;
;  567        1284  2 END;     ! routine display_bucket


                                      0000 00000 DISPLAY_BUCKET:
                                                      .WORD   Save nothing                          ; 1270
                       7E        04    AC  7D 00002   MOVQ    BUCKET, -(SP)                         ; 1280
               0000V   CF              02  FB 00006   CALLS   #2, DISPLAY_ENTRY_1                   ;
                       1E              50  E9 0000B   BLBC    STATUS, 1$                            ;
                                       7E  D4 0000E   CLRL    -(SP)                                 ; 1281
          7E      04   AC              01  C1 00010   ADDL3   #1, BUCKET, -(SP)                     ;
               0000V   CF              02  FB 00015   CALLS   #2, DISPLAY_ENTRY_2                   ;
                       0F              50  E9 0001A   BLBC    STATUS, 1$                            ;
                       7E        04    AC  7D 0001D   MOVQ    BUCKET, -(SP)                         ; 1282
               0000V   CF              02  FB 00021   CALLS   #2, DISPLAY_BLOCK                     ;
                       03              50  E9 00026   BLBC    STATUS, 1$                            ;
                       50              01  D0 00029   MOVL    #1, R0                                ; 1283
                                           04 0002C 1$: RET                                         ; 1284

; Routine Size:  45 bytes,    Routine Base:  $CODE$ + 047E


;  568        1285  2
```

```
;  570          1286  2 %SBTTL 'erase_block';
;  571          1287  2
;  572          1288  2 ROUTINE erase_block (lin,col,nlines) =
;  573          1289  2 !+++
;  574          1290  2 !
;  575          1291  2 !       Erase a block on the screen.
;  576          1292  2 !       Called by
;  577          1293  2 !               dump_block
;  578          1294  2 !               display_block
;  579          1295  2 !
;  580          1296  2 !---
;  581          1297  3 BEGIN
;  582          1298  3 INCR i FROM .lin TO (.lin + .nlines - 1) BY 1 DO
;  583          1299  3 scr$erase_line (.i,.col);
;  584          1300  3 RETURN true;
;  585          1301  2 END;



                                          000C 00000 ERASE_BLOCK:
                                                            .WORD    Save R2,R3                                  ; 1288
         53        04   AC        0C   AC   C1 00002        ADDL3    NLINES, LIN, R3                             ; 1298
         52        04   AC             01   C3 00008        SUBL3    #1, LIN, I                                  ; 1299
                                       0C   11 0000D        BRB      2$                                          ;
                                  08   AC   DD 0000F 1$:    PUSHL    COL                                         ;
                                       52   DD 00012        PUSHL    I                                           ;
             00000000G  00             02   FB 00014        CALLS    #2, SCR$ERASE_LINE                          ;
         F0             52             53   F2 0001B 2$:    AOBLSS   R3, I, 1$                                   ;
                        50             01   D0 0001F        MOVL     #1, R0                                      ; 1300
                                            04 00022        RET                                                  ; 1301

; Routine Size:  35 bytes,    Routine Base:  $CODE$ + 04AB
```

```
  2 %SBTTL 'display_entries';
  2
  2 ROUTINE display_entries ( bucket, entry, level) =
  2 !+++
  2 !
  2 !       Display the cache entry linked list for the given bucket
  2 !       On the screen at the given level.
  2 !       Called by:
  2 !               display_entry_1
  2 !
  2 !---
  3 BEGIN
  3
  3 BIND
  3     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,
  3     cache = .context [ctx$l_cache] : VECTOR;
  3
  3 LOCAL
  3     line,
  3     cache_entry : REF BBLOCK,
  3     entry_num,
  3     entries_out;
  3
  3 cache_entry = .cache[.bucket];
  3 line = entry_depth * (.level-1) + depth_offset;
  3 entries_out = 0;
  3 entry_num = 0;
  3
  3 IF NOT ( (.bucket LSS first_bucket) OR (.bucket GTR last_bucket) ) THEN
  3     WHILE (.cache_entry NEQ 0) AND (.entries_out LSS entries_wide) DO
  4         BEGIN
  4         IF ( .entry_num GEQ .entry ) THEN
  5             BEGIN
P 5             write_screen( .line, wid_offset + .entries_out*entry_wid,
  5                 faoentry, .cache_entry[ cache$l_link ] );
P 5             write_screen( .line+1, wid_offset + .entries_out*entry_wid,
  5                 faoentry, .cache_entry[ cache$l_vbn ] );
P 5             write_screen( .line+2, wid_offset + .entries_out*entry_wid,
  5                 faoentry, .cache_entry[ cache$l_address ] );
P 5             write_screen( .line+3, wid_offset + .entries_out*entry_wid,
  5                 faoentry, .cache_entry[ cache$w_flags ] );
  5             entries_out = .entries_out + 1;
  4             END;
  4         cache_entry = .cache_entry[ cache$l_link];
  4         entry_num = .entry_num + 1;
  3         END;    ! WHILE still entries in bucket, and not off edge of screen
  3
  3 IF (.entries_out LSS entries_wide ) THEN
  4     BEGIN
  4     perform ( scr$erase_line(.line , wid_offset + .entries_out * entry_wid) );
  4     perform ( scr$erase_line(.line+1 , wid_offset + .entries_out * entry_wid));
  4     perform ( scr$erase_line(.line+2 , wid_offset + .entries_out * entry_wid));
  4     perform ( scr$erase_line(.line+3 , wid_offset + .entries_out * entry_wid));
  3     END;
  3 RETURN true;
  2 END;    ! Routine display_entries
```

```
                                          07FC 00000 DISPLAY_ENTRIES:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10        ; 1304
                       5A      FF78   CF   9E 00002         MOVAB   FAO_BUFFER, R10                          ;
                       59     0000'   CF   9E 00007         MOVAB   FAOENTRY, R9                             ;
                       58 00000000G   00   9E 0000C         MOVAB   SCR$ERASE_LINE, R8                       ;
                       57 00000000G   00   9E 00013         MOVAB   SCR$PUT_SCREEN, R7                       ;
                       50     0000G   CF   D0 0001A         MOVL    LBR$GL_CONTROL, R0                       ; 1316
                       51        0E   A0   D0 0001F         MOVL    14(R0), R1                               ;
                       50        04   AC   D0 00023         MOVL    BUCKET, R0                               ; 1325
                       55   08 B140        D0 00027         MOVL    @8(R1)[R0], CACHE_ENTRY                  ;
       52         0C   AC             05   C5 0002C         MULL3   #5, LEVEL, R2                            ; 1326
                       52             04   C2 00031         SUBL2   #4, LINE                                 ;
                                      53   D4 00034         CLRL    ENTRIES_OUT                              ; 1327
                                      56   D4 00036         CLRL    ENTRY_NUM                                ; 1328
                                      50   D5 00038         TSTL    R0                                       ; 1330
                                      70   19 0003A         BLSS    3$                                       ;
         0000007F      8F             50   D1 0003C         CMPL    R0, #127                                 ;
                                      67   14 00043         BGTR    3$                                       ;
                                      55   D5 00045 1$:     TSTL    CACHE_ENTRY                              ; 1331
                                      63   13 00047         BEQL    3$                                       ;
                       0C             53   D1 00049         CMPL    ENTRIES_OUT, #12                         ;
                                      5E   18 0004C         BGEQ    3$                                       ;
                  08   AC             56   D1 0004E         CMPL    ENTRY_NUM, ENTRY                         ; 1333
                                      51   19 00052         BLSS    2$                                       ;
       54              53             09   C5 00054         MULL3   #9, ENTRIES_OUT, R4                      ; 1336
                                14    A4   9F 00058         PUSHAB  20(R4)                                   ;
                                      52   DD 0005B         PUSHL   LINE                                     ;
                                      65   DD 0005D         PUSHL   (CACHE_ENTRY)                            ;
                                      59   DD 0005F         PUSHL   R9                                       ;
                       6A             02   FB 00061         CALLS   #2, FAO_BUFFER                           ;
                                      50   DD 00064         PUSHL   R0                                       ;
                       67             03   FB 00066         CALLS   #3, SCR$PUT_SCREEN                       ;
                                14    A4   9F 00069         PUSHAB  20(R4)                                   ; 1338
                                01    A2   9F 0006C         PUSHAB  1(LINE)                                  ;
                                04    A5   DD 0006F         PUSHL   4(CACHE_ENTRY)                           ;
                                      59   DD 00072         PUSHL   R9                                       ;
                       6A             02   FB 00074         CALLS   #2, FAO_BUFFER                           ;
                                      50   DD 00077         PUSHL   R0                                       ;
                       67             03   FB 00079         CALLS   #3, SCR$PUT_SCREEN                       ;
                                14    A4   9F 0007C         PUSHAB  20(R4)                                   ; 1340
                                02    A2   9F 0007F         PUSHAB  2(LINE)                                  ;
                                08    A5   DD 00082         PUSHL   8(CACHE_ENTRY)                           ;
                                      59   DD 00085         PUSHL   R9                                       ;
                       6A             02   FB 00087         CALLS   #2, FAO_BUFFER                           ;
                                      50   DD 0008A         PUSHL   R0                                       ;
                       67             03   FB 0008C         CALLS   #3, SCR$PUT_SCREEN                       ;
                                14    A4   9F 0008F         PUSHAB  20(R4)                                   ; 1342
                                03    A2   9F 00092         PUSHAB  3(LINE)                                  ;
                       7E       0C    A5   3C 00095         MOVZWL  12(CACHE_ENTRY), -(SP)                   ;
                                      59   DD 00099         PUSHL   R9                                       ;
                       6A             02   FB 0009B         CALLS   #2, FAO_BUFFER                           ;
                                      50   DD 0009E         PUSHL   R0                                       ;
                       67             03   FB 000A0         CALLS   #3, SCR$PUT_SCREEN                       ;
```

```
                    53  D6 000A3         INCL    ENTRIES_OUT                            ; 1343
      55            65  D0 000A5 2$:     MOVL    (CACHE_ENTRY), CACHE_ENTRY             ; 1345
                    56  D6 000A8         INCL    ENTRY_NUM                              ; 1346
                    99  11 000AA         BRB     1$                                     ; 1331
      0C            53  D1 000AC 3$:     CMPL    ENTRIES_OUT, #12                       ; 1349
                    32  18 000AF         BGEQ    4$                                     ;
      53            09  C4 000B1         MULL2   #9, R3                                 ; 1351
              14    A3  9F 000B4         PUSHAB  20(R3)                                 ;
                    52  DD 000B7         PUSHL   LINE                                   ;
      68            02  FB 000B9         CALLS   #2, SCR$ERASE_LINE                     ;
      27            50  E9 000BC         BLBC    STATUS, 5$                             ;
              14    A3  9F 000BF         PUSHAB  20(R3)                                 ; 1352
              01    A2  9F 000C2         PUSHAB  1(LINE)                                ;
      68            02  FB 000C5         CALLS   #2, SCR$ERASE_LINE                     ;
      1B            50  E9 000C8         BLBC    STATUS, 5$                             ;
              14    A3  9F 000CB         PUSHAB  20(R3)                                 ; 1353
              02    A2  9F 000CE         PUSHAB  2(LINE)                                ;
      68            02  FB 000D1         CALLS   #2, SCR$ERASE_LINE                     ;
      0F            50  E9 000D4         BLBC    STATUS, 5$                             ;
              14    A3  9F 000D7         PUSHAB  20(R3)                                 ; 1354
              03    A2  9F 000DA         PUSHAB  3(LINE)                                ;
      68            02  FB 000DD         CALLS   #2, SCR$ERASE_LINE                     ;
      03            50  E9 000E0         BLBC    STATUS, 5$                             ;
      50            01  D0 000E3 4$:     MOVL    #1, R0                                 ; 1356
                        04 000E6 5$:     RET                                            ; 1357

; Routine Size:  231 bytes,    Routine Base:  $CODE$ + 04CE


;  643          1358  2
```

```
;  645      1359  2 %SBTTL 'display_entry_1';
;  646      1360  2
;  647      1361  2 ROUTINE display_entry_1 ( bucket, entry) =
;  648      1362  2 !+++
;  649      1363  2 !
;  650      1364  2 !       Display the linked list of cache entries at BUCKET,
;  651      1365  2 !       which is the current bucket, and display at level 1,
;  652      1366  2 !       beginning witht the ENTRYth entry in the list.
;  653      1367  3 BEGIN
;  654      1368  3 LOCAL status;
;  655      1369  3 status = display_entries (.bucket, .entry, level_1);
;  656      1370  3 RETURN .status;
;  657      1371  2 END;



                                        0000 00000 DISPLAY_ENTRY_1:
                                                        .WORD   Save nothing                            ; 1361
                                          01  DD 00002  PUSHL   #1                                      ; 1369
                    7E          04        AC  7D 00004  MOVQ    BUCKET, -(SP)                           ;
              FF0C  CF                    03  FB 00008  CALLS   #3, DISPLAY_ENTRIES                     ;
                                              04 0000D  RET                                             ; 1371

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 05B5


;  658      1372  2
```

```
  2 %SBTTL 'display_entry_2';
  2
  2 ROUTINE display_entry_2 ( bucket, entry) =
  2 !+++
  2 !
  2 !       Display the linked list of cache entries at bucket,
  2 !       where bucket is the next bucket after the current
  2 !       bucket.  Count in ENTRY entries before displaying.
  2 !       Display at level 2, and only display the VBN and flags
  2 !       field from the cache entry due to line limit on screen.
  2 !
  2 !---
  3 BEGIN
  3
  3 BIND
  3     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,
  3     cache = .context [ctx$l_cache] : VECTOR;
  3
  3 LOCAL
  3     status,
  3     line,
  3     entry_num,
  3     cache_entry : REF BBLOCK,
  3     entries_out;
  3
  3 cache_entry = .cache[.bucket];
  3 line = entry_depth + depth_offset;
  3 entries_out = 0;
  3 entry_num = 0;
  3
  3 IF NOT ( (.bucket LSS first_bucket) OR (.bucket GTR last_bucket) ) THEN
  3     WHILE (.cache_entry NEQ 0) AND (.entries_out LSS entries_wide) DO
  4         BEGIN
  4         IF ( .entry_num GEQ .entry ) THEN
  5             BEGIN
P 5             write_screen( .line, wid_offset + .entries_out*entry_wid,
  5                 faoentry, .cache_entry[ cache$l_vbn ] );
P 5             write_screen( .line+1, wid_offset + .entries_out*entry_wid,
  5                 faoentry, .cache_entry[ cache$w_flags ] );
  5             entries_out = .entries_out + 1;
  4             END;
  4         cache_entry = .cache_entry[ cache$l_link];
  4         entry_num = .entry_num + 1;
  3         END;    ! WHILE still entries in bucket, and not off edge of screen
  3
  3 IF (.entries_out LSS entries_wide ) THEN
  4     BEGIN
  4     perform (scr$erase_line(.line , wid_offset + .entries_out * entry_wid));
  4     perform (scr$erase_line(.line+1 , wid_offset + .entries_out * entry_wid));
  3     END;
  3 RETURN true;
  2 END;    ! Routine display_entry_2
```

```
                                       01FC 00000 DISPLAY_ENTRY_2:
                                                            .WORD   Save R2,R3,R4,R5,R6,R7,R8            ; 1375
                    58 00000000G   00   9E 00002          MOVAB   SCR$ERASE_LINE, R8                     ;
                    57 00000000G   00   9E 00009          MOVAB   SCR$PUT_SCREEN, R7                     ;
                    50      0000G   CF   D0 00010          MOVL    LBR$GL_CONTROL, R0                     ; 1388
                    51         0E   A0   D0 00015          MOVL    14(R0), R1                             ;
                    50         04   AC   D0 00019          MOVL    BUCKET, R0                             ; 1398
                    54       08 B140    D0 0001D          MOVL    @8(R1)[R0], CACHE_ENTRY                ;
                                    52   D4 00022          CLRL    ENTRIES_OUT                            ; 1400
                    55              06   7D 00024          MOVQ    #6, LINE                               ; 1399
                                    50   D5 00027          TSTL    R0                                     ; 1403
                                    53   19 00029          BLSS    3$                                     ;
          0000007F  8F              50   D1 0002B          CMPL    R0, #127                               ;
                                    4A   14 00032          BGTR    3$                                     ;
                                    54   D5 00034 1$:      TSTL    CACHE_ENTRY                            ; 1404
                                    46   13 00036          BEQL    3$                                     ;
                    0C              52   D1 00038          CMPL    ENTRIES_OUT, #12                       ;
                                    41   18 0003B          BGEQ    3$                                     ;
               08   AC              56   D1 0003D          CMPL    ENTRY_NUM, ENTRY                       ; 1406
                                    34   19 00041          BLSS    2$                                     ;
          53        52              09   C5 00043          MULL3   #9, ENTRIES_OUT, R3                    ; 1409
                               14   A3   9F 00047          PUSHAB  20(R3)                                 ;
                                    55   DD 0004A          PUSHL   LINE                                   ;
                               04   A4   DD 0004C          PUSHL   4(CACHE_ENTRY)                         ;
                            0000'   CF   9F 0004F          PUSHAB  FAOENTRY                               ;
               FE31 CF              02   FB 00053          CALLS   #2, FAO_BUFFER                         ;
                                    50   DD 00058          PUSHL   R0                                     ;
                    67              03   FB 0005A          CALLS   #3, SCR$PUT_SCREEN                     ;
                               14   A3   9F 0005D          PUSHAB  20(R3)                                 ; 1411
                               01   A5   9F 00060          PUSHAB  1(LINE)                                ;
                    7E         0C   A4   3C 00063          MOVZWL  12(CACHE_ENTRY), -(SP)                 ;
                            0000'   CF   9F 00067          PUSHAB  FAOENTRY                               ;
               FE19 CF              02   FB 0006B          CALLS   #2, FAO_BUFFER                         ;
                                    50   DD 00070          PUSHL   R0                                     ;
                    67              03   FB 00072          CALLS   #3, SCR$PUT_SCREEN                     ;
                                    52   D6 00075          INCL    ENTRIES_OUT                            ; 1412
                    54              64   D0 00077 2$:      MOVL    (CACHE_ENTRY), CACHE_ENTRY             ; 1414
                                    56   D6 0007A          INCL    ENTRY_NUM                              ; 1415
                                    B6   11 0007C          BRB     1$                                     ; 1404
                    0C              52   D1 0007E 3$:      CMPL    ENTRIES_OUT, #12                       ; 1418
                                    1A   18 00081          BGEQ    4$                                     ;
                    52              09   C4 00083          MULL2   #9, R2                                 ; 1420
                               14   A2   9F 00086          PUSHAB  20(R2)                                 ;
                                    55   DD 00089          PUSHL   LINE                                   ;
                    68              02   FB 0008B          CALLS   #2, SCR$ERASE_LINE                     ;
                    0F              50   E9 0008E          BLBC    STATUS, 5$                             ;
                               14   A2   9F 00091          PUSHAB  20(R2)                                 ; 1421
                               01   A5   9F 00094          PUSHAB  1(LINE)                                ;
                    68              02   FB 00097          CALLS   #2, SCR$ERASE_LINE                     ;
                    03              50   E9 0009A          BLBC    STATUS, 5$                             ;
                    50              01   D0 0009D 4$:      MOVL    #1, R0                                 ; 1423
                                    04  000A0 5$:          RET                                            ; 1424

; Routine Size:  161 bytes,    Routine Base:  $CODE$ + 05C3


;  712          1425  2
```

```
  2 %SBTTL 'dump_block';
  2
  2 ROUTINE dump_block ( blk ) =
  2 !+++
  2 !
  2 !         Dump the block at address BLK, in hex and ASCII.
  2 !
  2 !
  3 BEGIN
  3 MAP
  3      blk : REF BBLOCK;
  3 LOCAL
  3      status,
  3      line;
  3
  3 IF .blk EQL 0 THEN
  4      BEGIN
  4      status = erase_block ( blk_lin,blk_col,16);
  4      RETURN .status;
  3      END;
  3 line = blk_lin;
  3 INCR i FROM 0 TO 512-32 BY 32 DO
  4      BEGIN
  4      write_screen ( .line, blk_col, faoblock,
P 4        .blk [.i+28,0,32,0], .blk [.i+24,0,32,0], .blk [.i+20,0,32,0],
P 4        .blk [.i+16,0,32,0], .blk [.i+12,0,32,0], .blk [.i+8,0,32,0],
P 4        .blk [.i+4,0,32,0], .blk [.i,0,32,0],
  4       32, blk [.i,0,0,0] );
  4      line = .line + 1;
  3      END;
  3 RETURN true;
  2 END;     ! routine dump_block
```

```
                                  001C 00000 DUMP_BLOCK:
                                                    .WORD   Save R2,R3,R4          ; 1428
          52        04   AC   D0 00002              MOVL    BLK, R2                ; 1441
                         0C   12 00006              BNEQ    1$                     ;
                         10   DD 00008              PUSHL   #16                    ; 1443
                         14   DD 0000A              PUSHL   #20                    ;
                         09   DD 0000C              PUSHL   #9                     ;
        FE34    CF       03   FB 0000E              CALLS   #3, ERASE_BLOCK        ;
                              04 00013              RET                            ; 1444
          54             09   D0 00014 1$:          MOVL    #9, LINE               ; 1446
                         53   D4 00017              CLRL    I                      ; 1447
                         14   DD 00019 2$:          PUSHL   #20                    ; 1453
                         54   DD 0001B              PUSHL   LINE                   ;
                       6342   9F 0001D              PUSHAB  (I)[R2]                ;
                         20   DD 00020              PUSHL   #32                    ;
                       6342   9F 00022              PUSHAB  (I)[R2]                ;
                         9E   DD 00025              PUSHL   @(SP)+                 ;
                    04 A342   9F 00027              PUSHAB  4(I)[R2]               ;
                         9E   DD 0002B              PUSHL   @(SP)+                 ;
                    08 A342   9F 0002D              PUSHAB  8(I)[R2]               ;
```

```
                                    9E  DD 00031        PUSHL   @(SP)+                           ;
                             0C A342    9F 00033        PUSHAB  12(I)[R2]                        ;
                                    9E  DD 00037        PUSHL   @(SP)+                           ;
                             10 A342    9F 00039        PUSHAB  16(I)[R2]                        ;
                                    9E  DD 0003D        PUSHL   @(SP)+                           ;
                             14 A342    9F 0003F        PUSHAB  20(I)[R2]                        ;
                                    9E  DD 00043        PUSHL   @(SP)+                           ;
                             18 A342    9F 00045        PUSHAB  24(I)[R2]                        ;
                                    9E  DD 00049        PUSHL   @(SP)+                           ;
                             1C A342    9F 0004B        PUSHAB  28(I)[R2]                        ;
                                    9E  DD 0004F        PUSHL   @(SP)+                           ;
                            0000'   CF  9F 00051        PUSHAB  FAOBLOCK                         ;
                 FD8E   CF          0B  FB 00055        CALLS   #11, FAO_BUFFER                  ;
                                    50  DD 0005A        PUSHL   R0                               ;
            00000000G   00          03  FB 0005C        CALLS   #3, SCR$PUT_SCREEN               ;
                                    54  D6 00063        INCL    LINE                             ; 1454
     FFAA          53   20 000001E0 8F  F1 00065        ACBL    #480, #32, I, 2$                 ; 1447
                        50          01  D0 0006F        MOVL    #1, R0                           ; 1456
                                        04 00072        RET                                      ; 1457

; Routine Size:  115 bytes,    Routine Base:  $CODE$ + 0664


;  746          1458  2
```

```
;  748      1459  2 %SBTTL 'display_block';
;  749      1460  2
;  750      1461  2 ROUTINE display_block ( bucket, entry) =
;  751      1462  2 !+++
;  752      1463  2 !
;  753      1464  2 !        Locate the ENTRYth cache entry in BUCKET,  obtain the address
;  754      1465  2 !        of the cached block and call dump_block to print the hex and
;  755      1466  2 !        ASCII dump of the block.
;  756      1467  2 !
;  757      1468  2 !---
;  758      1469  3 BEGIN
;  759      1470  3
;  760      1471  3 BIND
;  761      1472  3     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,
;  762      1473  3     cache = .context [ctx$l_cache] : VECTOR;
;  763      1474  3
;  764      1475  3 LOCAL
;  765      1476  3     status,
;  766      1477  3     entry_num,
;  767      1478  3     vbn,
;  768      1479  3     cache_entry : REF BBLOCK;
;  769      1480  3
;  770      1481  3 IF (.bucket LSS first_bucket) OR (.bucket GTR last_bucket) THEN
;  771      1482  3     RETURN erase_block(blk_lin,blk_col,16);
;  772      1483  3 cache_entry = .cache[.bucket];
;  773      1484  3 IF .cache_entry EQL 0 THEN
;  774      1485  3     RETURN erase_block(blk_lin,blk_col,16);
;  775      1486  3 entry_num = 0;
;  776      1487  3 WHILE (.cache_entry NEQ 0) AND (.entry_num LSS .entry) DO
;  777      1488  4     BEGIN
;  778      1489  4     cache_entry = .cache_entry[cache$l_link];
;  779      1490  4     entry_num = .entry_num + 1;
;  780      1491  3     END;
;  781      1492  4 IF ( (.entry_num EQL .entry) AND (.cache_entry[cache$l_address] NEQ 0) )
;  782      1493  3 THEN
;  783      1494  4     BEGIN
;  784      1495  4     cur_vbn = .cache_entry [cache$l_vbn];
;  785      1496  4     status = dump_block ( .cache_entry[cache$l_address]);
;  786      1497  4     END
;  787      1498  3 ELSE
;  788      1499  3     status = erase_block (blk_lin,blk_col,16);
;  789      1500  3 RETURN .status;
;  790      1501  2 END;     ! routine display_block


                                 000C 00000 DISPLAY_BLOCK:
                                                  .WORD   Save R2,R3                                 : 1461
                 50      0000G   CF   D0 00002    MOVL    LBR$GL_CONTROL, R0                         : 1472
                 53         0E   A0   D0 00007    MOVL    14(R0), R3                                 :
                 52         04   AC   D0 0000B    MOVL    BUCKET, R2                                 : 1481
                                 3D   19 0000F    BLSS    3$                                         :
       0000007F  8F              52   D1 00011    CMPL    R2, #127                                   :
                                 34   14 00018    BGTR    3$                                         :
                 52       08 B342     D0 0001A    MOVL    @8(R3)[R2], CACHE_ENTRY                    : 1483
```

```
                          2D   13 0001F        BEQL    3$                              ; 1484
                          50   D4 00021        CLRL    ENTRY_NUM                       ; 1486
                          52   D5 00023 1$:    TSTL    CACHE_ENTRY                     ; 1487
                          0D   13 00025        BEQL    2$                              ;
        08    AC          50   D1 00027        CMPL    ENTRY_NUM, ENTRY                ;
                          07   18 0002B        BGEQ    2$                              ;
              52          62   D0 0002D        MOVL    (CACHE_ENTRY), CACHE_ENTRY      ; 1489
                          50   D6 00030        INCL    ENTRY_NUM                       ; 1490
                          EF   11 00032        BRB     1$                              ; 1487
        08    AC          50   D1 00034 2$:    CMPL    ENTRY_NUM, ENTRY                ; 1492
                          14   12 00038        BNEQ    3$                              ;
                    08    A2   D5 0003A        TSTL    8(CACHE_ENTRY)                  ;
                          0F   13 0003D        BEQL    3$                              ;
      0000'   CF    04    A2   D0 0003F        MOVL    4(CACHE_ENTRY), CUR_VBN         ; 1495
                    08    A2   DD 00045        PUSHL   8(CACHE_ENTRY)                  ; 1496
      FF40    CF          01   FB 00048        CALLS   #1, DUMP_BLOCK                  ;
                               04 0004D        RET                                     ; 1492
                          10   DD 0004E 3$:    PUSHL   #16                             ; 1499
                          14   DD 00050        PUSHL   #20                             ;
                          09   DD 00052        PUSHL   #9                              ;
      FD7B    CF          03   FB 00054        CALLS   #3, ERASE_BLOCK                 ;
                               04 00059        RET                                     ; 1501

; Routine Size:  90 bytes,    Routine Base:  $CODE$ + 06D7


;  791          1502  2
```

```
2 %SBTTL 'find_vbn';
2
2 ROUTINE find_vbn ( read, bucket, entry, vbn) =
2 !+++
2 !
2 !       Search the cache for a cache_entry for the requested VBN.
2 !       If found, reset the current bucket and entry to point to it.
2 !       If function is read then read it in and cache it.
2 !
2 !       bucket: address of global longword containing current bucket
2 !       entry : address of global current offset in cache entry list
2 !       vbn   : virtual block to search cache for
2 !
2 !---
3 BEGIN
3 BIND
3     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,
3     cache = .context [ctx$l_cache] : VECTOR;
3 LOCAL
3     bckt,
3     blkadr,
3     cache_addr,
3     cache_entry :        REF BBLOCK,
3     ntry,
3     status;
3
3 IF .read
4 THEN perform ( find_block ( .vbn, blkadr, cache_addr ) )
3 ELSE perform ( lookup_cache (.vbn, cache_addr) );
3
3 bckt = (.vbn - 1) MOD (lbr$c_hashsize/4);
3 cache_entry = .cache[.bckt];
3 ntry = 0;
3 UNTIL ( (.cache_entry EQL 0) OR (.cache_entry[cache$l_vbn] EQL .vbn)) DO
4     BEGIN
4     cache_entry = .cache_entry[cache$l_link];
4     ntry = .ntry + 1;
3     END;
4 IF (.cache_entry EQL 0) OR (.cache_entry NEQ .cache_addr)
3 THEN 0=0                  ! access violate to trap logic error
3 ELSE
4     BEGIN
4     .bucket = .bckt;
4     .entry = .ntry;
3     END;
3 RETURN true;
2 END;    ! ROUTINE find_vbn
```

```
                        0FFC 00000 FIND_VBN:
                                         .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11        ; 1505
         5E          08  C2 00002        SUBL2   #8, SP                                      ;
         50    0000G CF  D0 00005        MOVL    LBR$GL_CONTROL, R0                          ; 1519
         53       0E  A0  D0 0000A       MOVL    14(R0), R3                                  ;
```

```
                      10      04   AC  E9 0000E        BLBC    READ, 1$                                  ; 1529
                      52      04   AE  9E 00012        MOVAB   CACHE_ADDR, R2                            ; 1530
                      51           6E  9E 00016        MOVAB   BLKADR, R1                                ;
                      50      10   AC  D0 00019        MOVL    VBN, R0                                   ;
                               0000G   30 0001D        BSBW    FIND_BLOCK                                ;
                                   0B  11 00020        BRB     2$                                        ;
                      51      04   AE  9E 00022 1$:    MOVAB   CACHE_ADDR, R1                            ; 1531
                      50      10   AC  D0 00026        MOVL    VBN, R0                                   ;
                               0000G   30 0002A        BSBW    LOOKUP_CACHE                              ;
                      49           50  E9 0002D 2$:    BLBC    STATUS, 8$                                ;
   7E FFFFFFFF   8F   10   AC          01  7A 00030        EMUL    #1, VBN, #-1, -(SP)                       ; 1533
   51            51        8E 00000080 8F  7B 0003A        EDIV    #128, (SP)+, BCKT, BCKT                   ;
                      50      08 B341  D0 00043        MOVL    @8(R3)[BCKT], CACHE_ENTRY                 ; 1534
                                   52  D4 00048        CLRL    NTRY                                      ; 1535
                                   50  D5 0004A 3$:    TSTL    CACHE_ENTRY                               ; 1536
                                   0E  13 0004C        BEQL    4$                                        ;
                 10   AC      04   A0  D1 0004E        CMPL    4(CACHE_ENTRY), VBN                       ;
                                   07  13 00053        BEQL    4$                                        ;
                      50           60  D0 00055        MOVL    (CACHE_ENTRY), CACHE_ENTRY                ; 1538
                                   52  D6 00058        INCL    NTRY                                      ; 1539
                                   EE  11 0005A        BRB     3$                                        ; 1536
                                   50  D5 0005C 4$:    TSTL    CACHE_ENTRY                               ; 1541
                                   06  13 0005E        BEQL    5$                                        ;
                      04   AE      50  D1 00060        CMPL    CACHE_ENTRY, CACHE_ADDR                   ;
                                   08  13 00064        BEQL    6$                                        ;
                          00000000 9F  D4 00066 5$:    CLRL    @#^X00000000                              ; 1542
                                   08  11 0006C        BRB     7$                                        ;
                      08   BC      51  D0 0006E 6$:    MOVL    BCKT, @BUCKET                             ; 1545
                      0C   BC      52  D0 00072        MOVL    NTRY, @ENTRY                              ; 1546
                      50           01  D0 00076 7$:    MOVL    #1, R0                                    ; 1548
                                       04 00079 8$:    RET                                               ; 1549

; Routine Size:  122 bytes,    Routine Base:  $CODE$ + 0731


;  840          1550  2
```

```
2 %SBTTL 'locate_vbn';
2
2 ROUTINE locate_vbn ( read, bucket, entry ) =
2 !+++
2 !
2 !       Request from SYS$INPUT a VBN to find.  Locate the VBN in the cache,
2 !       and if found, reset the current bucket and entry.  If READ is true,
2 !       then read the block into the cache if not already there.
2 !
2 !       bucket: address of global longword containing current bucket
2 !       entry : address of global current offset in cache entry list
2 !
2 !---
3 BEGIN
3
3 OWN
3     desc :      BBLOCK [dsc$c_s_bln],    ! Result descriptor
3     buf :       VECTOR [32, BYTE];       ! Output buffer
3 LOCAL
3     status,
3     vbn;
3
3 desc [dsc$w_length] = 32;                              ! Set up result descriptor
3 desc [dsc$a_pointer] = buf;
3 perform ( scr$set_cursor(input_lin, input_col) );
3 perform( scr$get_screen( desc,0,desc[dsc$w_length] ) );
3 perform( ots$cvt_tz_l ( desc, vbn ) );                 ! convert text string to hex
3 status = find_vbn (.read, .bucket, .entry, .vbn);
3 RETURN .status;
2 END;    ! routine locate_vbn


                                                          .PSECT  $OWN$,NOEXE,2

                                       00208 DESC:        .BLKB   8
                                       00210 BUF:         .BLKB   32


                                                          .PSECT  $CODE$,NOWRT,2

                                  0004 00000 LOCATE_VBN:
                                                          .WORD   Save R2                          ; 1553
                 52       0000'  CF  9E 00002             MOVAB   DESC, R2                         ;
                 5E              04  C2 00007             SUBL2   #4, SP                           ;
                 62              20  B0 0000A             MOVW    #32, DESC                        ; 1573
          04     A2       08     A2  9E 0000D             MOVAB   BUF, DESC+4                      ; 1574
                 7E              18  7D 00012             MOVQ    #24, -(SP)                       ; 1575
    00000000G    00              02  FB 00015             CALLS   #2, SCR$SET_CURSOR               ;
                 2C              50  E9 0001C             BLBC    STATUS, 1$                       ;
                                 52  DD 0001F             PUSHL   R2                               ; 1576
                                 7E  D4 00021             CLRL    -(SP)                            ;
                                 52  DD 00023             PUSHL   R2                               ;
    00000000G    00              03  FB 00025             CALLS   #3, SCR$GET_SCREEN               ;
                 1C              50  E9 0002C             BLBC    STATUS, 1$                       ;
                          4004   8F  BB 0002F             PUSHR   #^M<R2,SP>                       ; 1577
```

```
                  00000000G  00          02  FB 00033        CALLS   #2, OTS$CVT_TZ_L                 ;
                             0E          50  E9 0003A        BLBC    STATUS, 1$                       ;
                                         6E  DD 0003D        PUSHL   VBN                              ; 1578
                             7E      08  AC  7D 0003F        MOVQ    BUCKET, -(SP)                    ;
                                     04  AC  DD 00043        PUSHL   READ                             ;
                       FF3B  CF          04  FB 00046        CALLS   #4, FIND_VBN                     ;
                                             04 0004B  1$:   RET                                      ; 1580

; Routine Size:  76 bytes,    Routine Base:  $CODE$ + 07AB


;  872          1581  2
```


; 1
; 1
; 1
; 1
; 1
; 1
; 1
; 1
; 1

```
; 2 %SBTTL 'display_hash';
; 2
; 2 ROUTINE display_hash ( bucket ) =
; 2 !+++
; 2 !
; 2 !       Display window of the hash surrounding the current bucket
; 2 !
; 2 !---
; 3 BEGIN
; 3
; 3 BIND
; 3     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,
; 3     cache = .context [ctx$l_cache] : VECTOR;
; 3
; 3 LOCAL
; 3     status,
; 3     bckt,
; 3     begwin,
; 3     endwin;
; 3
; 3 IF ( .bucket LEQ half_hash_window ) THEN
; 4     BEGIN
; 4     begwin = 0;
; 4     endwin = 2 * half_hash_window;
; 4     END
; 3 ELSE
; 4     BEGIN
; 4     begwin = .bucket - half_hash_window;
; 4     endwin = .bucket + half_hash_window;
; 3     END;
; 3 IF .endwin GTR last_bucket THEN endwin = last_bucket;
; 3 bckt = 0;
; 3 WHILE ( .bckt LEQ .endwin ) DO
; 4     BEGIN
; 4     IF ( .bckt EQL .bucket) THEN
;P 4         write_screen ( .bckt - .begwin + bcktoffset, 0, faohashentry,
; 4           cache + 4* .bckt, .cache[.bckt])
; 4     ELSE
; 4         IF .bckt GEQ .begwin THEN
; 4             write_screen ( .bckt - .begwin + bcktoffset, 0, faohash, .cache[.bckt]);
; 4     bckt = .bckt + 1;
; 3     END;        ! WHILE
; 3 RETURN true;
; 2 END;      ! routine display_hash ( bucket )


                              003C 00000 DISPLAY_HASH:
                                         .WORD   Save R2,R3,R4,R5          ; 1584
      50     0000G  CF  D0 00002         MOVL    LBR$GL_CONTROL, R0        ; 1593
      53        0E  A0  D0 00007         MOVL    14(R0), R3                ;
      05        04  AC  D1 0000B         CMPL    BUCKET, #5                ; 1602
                    05  14 0000F         BGTR    1$                        ;
      54            0A  7D 00011         MOVQ    #10, ENDWIN               ; 1605
                    0A  11 00014         BRB     2$                        ; 1602
```

```
       55      04   AC        05 C3 00016 1$:    SUBL3   #5, BUCKET, BEGWIN            ; 1609
       54      04   AC        05 C1 0001B        ADDL3   #5, BUCKET, ENDWIN            ; 1610
          0000007F  8F        54 D1 00020 2$:    CMPL    ENDWIN, #127                  ; 1612
                              04 15 00027        BLEQ    3$                            ;
                    54   7F   8F 9A 00029        MOVZBL  #127, ENDWIN                  ;
                              52 D4 0002D 3$:    CLRL    BCKT                          ; 1613
                    54        52 D1 0002F 4$:    CMPL    BCKT, ENDWIN                  ; 1614
                              4A 14 00032        BGTR    8$                            ;
               04   AC        52 D1 00034        CMPL    BCKT, BUCKET                  ; 1616
                              1C 12 00038        BNEQ    5$                            ;
                              7E D4 0003A        CLRL    -(SP)                         ; 1618
       50           52        55 C3 0003C        SUBL3   BEGWIN, BCKT, R0              ;
                         02   A0 9F 00040        PUSHAB  2(R0)                         ;
                      08 B342 DD 00043           PUSHL   @8(R3)[BCKT]                  ;
                      08 B342 DF 00047           PUSHAL  @8(R3)[BCKT]                  ;
                       0000'  CF 9F 0004B        PUSHAB  FAOHASHENTRY                  ;
             FC01   CF        03 FB 0004F        CALLS   #3, FAO_BUFFER                ;
                              1B 11 00054        BRB     6$                            ;
                    55        52 D1 00056 5$:    CMPL    BCKT, BEGWIN                  ; 1620
                              1F 19 00059        BLSS    7$                            ;
                              7E D4 0005B        CLRL    -(SP)                         ; 1621
       50           52        55 C3 0005D        SUBL3   BEGWIN, BCKT, R0              ;
                         02   A0 9F 00061        PUSHAB  2(R0)                         ;
                      08 B342 DD 00064           PUSHL   @8(R3)[BCKT]                  ;
                       0000'  CF 9F 00068        PUSHAB  FAOHASH                       ;
             FBE4   CF        02 FB 0006C        CALLS   #2, FAO_BUFFER                ;
                              50 DD 00071 6$:    PUSHL   R0                            ;
          00000000G 00        03 FB 00073        CALLS   #3, SCR$PUT_SCREEN            ;
                              52 D6 0007A 7$:    INCL    BCKT                          ; 1622
                              B1 11 0007C        BRB     4$                            ; 1614
                    50        01 D0 0007E 8$:    MOVL    #1, R0                        ; 1624
                                 04 00081        RET                                   ; 1625

; Routine Size:  130 bytes,    Routine Base:  $CODE$ + 07F7

;  918          1626  2
```

```
;  920            1627  2 %SBTTL 'display_header';
;  921            1628  2
;  922            1629  2 ROUTINE display_header =
;  923            1630  2 !+++
;  924            1631  2 !
;  925            1632  2 !         Dump VBN 1, the library header.
;  926            1633  2 !
;  927            1634  2 !---
;  928            1635  3 BEGIN
;  929            1636  3 LOCAL status;
;  930            1637  3 status = dump_block( .lbr$gl_control[lbr$l_hdrptr] );
;  931            1638  3 RETURN .status
;  932            1639  2 END;



                                          0000 00000 DISPLAY_HEADER:
                                                          .WORD   Save nothing                              ; 1629
                        50      0000G  CF  D0 00002       MOVL    LBR$GL_CONTROL, R0                        ; 1637
                                   0A  A0  DD 00007       PUSHL   10(R0)                                    ;
                 FDDC   CF                 01  FB 0000A       CALLS   #1, DUMP_BLOCK                            ;
                                           04 0000F       RET                                               ; 1639

; Routine Size:  16 bytes,    Routine Base:  $CODE$ + 0879


;  933            1640  2
```

```
;  935        1641  2 %SBTTL 'up';
;  936        1642  2
;  937        1643  2 ROUTINE up (bucket, entry) =
;  938        1644  2 !+++
;  939        1645  2 !
;  940        1646  2 !       Decrement the current bucket by one and re-display the cache
;  941        1647  2 !       entry lists for the current bucket and the bucket after the
;  942        1648  2 !       current bucket.  Update the display of the hash window around
;  943        1649  2 !       the current bucket.
;  944        1650  2 !
;  945        1651  2 !       bucket: address of global longword containing current bucket
;  946        1652  2 !       entry : address of global current offset in cache entry list
;  947        1653  2 !
;  948        1654  2 !---
;  949        1655  3 BEGIN
;  950        1656  3 IF ..bucket NEQ first_bucket THEN
;  951        1657  3     .bucket = ..bucket - 1;
;  952        1658  3 display_hash (..bucket);
;  953        1659  3 display_entry_1 (..bucket, ..entry);
;  954        1660  3 display_entry_2 (..bucket + 1, first_entry);
;  955        1661  3 dump_block(0);
;  956        1662  3 RETURN true;
;  957        1663  2 END;



                                 0004 00000 UP:      .WORD   Save R2                         ; 1643
              52        04    AC   D0 00002          MOVL    BUCKET, R2                      ; 1656
                              62   D5 00006          TSTL    (R2)                            ;
                              02   13 00008          BEQL    1$                              ;
                              62   D7 0000A          DECL    (R2)                            ; 1657
                              62   DD 0000C 1$:      PUSHL   (R2)                            ; 1658
            FF5B    CF        01   FB 0000E          CALLS   #1, DISPLAY_HASH                ;
                        08    BC   DD 00013          PUSHL   @ENTRY                          ; 1659
                              62   DD 00016          PUSHL   (R2)                            ;
            FD0F    CF        02   FB 00018          CALLS   #2, DISPLAY_ENTRY_1             ;
                              7E   D4 0001D          CLRL    -(SP)                           ; 1660
       7E           62        01   C1 0001F          ADDL3   #1, (R2), -(SP)                 ;
            FD12    CF        02   FB 00023          CALLS   #2, DISPLAY_ENTRY_2             ;
                              7E   D4 00028          CLRL    -(SP)                           ; 1661
            FDAC    CF        01   FB 0002A          CALLS   #1, DUMP_BLOCK                  ;
                    50        01   D0 0002F          MOVL    #1, R0                          ; 1662
                                   04 00032          RET                                     ; 1663

; Routine Size:  51 bytes,    Routine Base:  $CODE$ + 0889


;  958        1664  2
;  959        1665  2 ROUTINE down (bucket, entry) =
;  960        1666  2 !+++
;  961        1667  2 !
;  962        1668  2 !       Increment the current bucket by one and re-display the cache
;  963        1669  2 !       entry lists for the current bucket and the bucket after the
;  964        1670  2 !       current bucket.  Update the display of the hash window around
;  965        1671  2 !       the current bucket.
```

```
;   966            1672  2 !
;   967            1673  2 !        bucket: address of global longword containing current bucket
;   968            1674  2 !        entry : address of global current offset in cache entry list
;   969            1675  2 !
;   970            1676  2 !---
;   971            1677  3 BEGIN
;   972            1678  3 IF ..bucket LSS last_bucket THEN
;   973            1679  3     .bucket = ..bucket + 1
;   974            1680  3 ELSE
;   975            1681  3     .bucket = last_bucket;
;   976            1682  3 display_hash (..bucket);
;   977            1683  3 display_entry_1 (..bucket, ..entry);
;   978            1684  3 display_entry_2 (..bucket + 1, first_entry);
;   979            1685  3 dump_block(0);
;   980            1686  3 RETURN true;
;   981            1687  2 END;


                                                    0000 00000 DOWN:   .WORD   Save nothing                                    ; 1665
                    0000007F    8F       04    BC    D1 00002           CMPL    @BUCKET, #127                                   ; 1678
                                                05    18 0000A           BGEQ    1$
                                         04    BC    D6 0000C           INCL    @BUCKET                                         ; 1679
                                                05    11 0000F           BRB     2$
                          04    BC       7F    8F    9A 00011 1$:       MOVZBL  #127, @BUCKET                                   ; 1681
                                         04    BC    DD 00016 2$:       PUSHL   @BUCKET                                         ; 1682
                        FF1D    CF             01    FB 00019           CALLS   #1, DISPLAY_HASH                                ;
                                         08    BC    DD 0001E           PUSHL   @ENTRY                                          ; 1683
                                         04    BC    DD 00021           PUSHL   @BUCKET                                         ;
                        FCD0    CF             02    FB 00024           CALLS   #2, DISPLAY_ENTRY_1                             ;
                                               7E    D4 00029           CLRL    -(SP)                                           ; 1684
                  7E      04    BC             01    C1 0002B           ADDL3   #1, @BUCKET, -(SP)                              ;
                        FCD2    CF             02    FB 00030           CALLS   #2, DISPLAY_ENTRY_2                             ;
                                               7E    D4 00035           CLRL    -(SP)                                           ; 1685
                        FD6C    CF             01    FB 00037           CALLS   #1, DUMP_BLOCK                                  ;
                                50             01    D0 0003C           MOVL    #1, R0                                          ; 1686
                                                     04 0003F           RET                                                     ; 1687

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 08BC


;   982            1688  2
;   983            1689  2 ROUTINE right (bucket, entry) =
;   984            1690  2 !+++
;   985            1691  2 !
;   986            1692  2 !        Increment the current entry by one and re-display the cache
;   987            1693  2 !        entry list for the current bucket.
;   988            1694  2 !
;   989            1695  2 !        bucket: address of global longword containing current bucket
;   990            1696  2 !        entry : address of global current offset in cache entry list
;   991            1697  2 !
;   992            1698  2 !---
;   993            1699  3 BEGIN
;   994            1700  3 .entry = ..entry + 1;
;   995            1701  3 display_entry_1 (..bucket, ..entry);
```

```
;  996        1702  3 dump_block(0);
;  997        1703  3 RETURN true;
;  998        1704  2 END;



                                      0000 00000 RIGHT:  .WORD    Save nothing                                  ; 1689
                               08   BC   D6 00002        INCL     @ENTRY                                        ; 1700
                               08   BC   DD 00005        PUSHL    @ENTRY                                        ; 1701
                               04   BC   DD 00008        PUSHL    @BUCKET                                       ;
                  FCA9    CF        02   FB 0000B        CALLS    #2, DISPLAY_ENTRY_1                           ;
                                    7E   D4 00010        CLRL     -(SP)                                         ; 1702
                  FD51    CF        01   FB 00012        CALLS    #1, DUMP_BLOCK                                ;
                          50        01   D0 00017        MOVL     #1, R0                                        ; 1703
                                         04 0001A        RET                                                    ; 1704

; Routine Size:  27 bytes,    Routine Base:  $CODE$ + 08FC


;  999        1705  2
; 1000        1706  2 ROUTINE left (bucket, entry) =
; 1001        1707  2 !+++
; 1002        1708  2 !
; 1003        1709  2 !         Decrement the current entry by one and re-display the cache
; 1004        1710  2 !         entry list for the current bucket.
; 1005        1711  2 !
; 1006        1712  2 !         bucket: address of global longword containing current bucket
; 1007        1713  2 !         entry : address of global current offset in cache entry list
; 1008        1714  2 !
; 1009        1715  2 !---
; 1010        1716  3 BEGIN
; 1011        1717  3 IF ..entry NEQ 0 THEN
; 1012        1718  3     .entry = ..entry - 1;
; 1013        1719  3 display_entry_1 (..bucket, ..entry);
; 1014        1720  3 dump_block(0);
; 1015        1721  3 RETURN true;
; 1016        1722  2 END;



                                      0000 00000 LEFT:   .WORD    Save nothing                                  ; 1706
                          50   08   AC   D0 00002        MOVL     ENTRY, R0                                     ; 1717
                                    60   D5 00006        TSTL     (R0)                                          ;
                                    02   13 00008        BEQL     1$                                            ;
                                    60   D7 0000A        DECL     (R0)                                          ; 1718
                                    60   DD 0000C 1$:    PUSHL    (R0)                                          ; 1719
                               04   BC   DD 0000E        PUSHL    @BUCKET                                       ;
                  FC88    CF        02   FB 00011        CALLS    #2, DISPLAY_ENTRY_1                           ;
                                    7E   D4 00016        CLRL     -(SP)                                         ; 1720
                  FD30    CF        01   FB 00018        CALLS    #1, DUMP_BLOCK                                ;
                          50        01   D0 0001D        MOVL     #1, R0                                        ; 1721
                                         04 00020        RET                                                    ; 1722
```

```
; Routine Size:  33 bytes,    Routine Base:  $CODE$ + 0917

; 1017          1723  2
```

```
; 1019        1724  2 %SBTTL 'display_help';
; 1020        1725  2
; 1021        1726  2 ROUTINE display_help =
; 1022        1727  3 BEGIN
; 1023        1728  3 !+++
; 1024        1729  3 !
; 1025        1730  3 !       Help by printing the valid commands
; 1026        1731  3 !
; 1027        1732  3 !---
; 1028        1733  3 LOCAL
; 1029        1734  3      hlplin,
; 1030        1735  3      hlpcol;
; 1031        1736  3
; 1032        1737  3 hlplin = 5;
; 1033        1738  3 hlpcol = 5;
; 1034        1739  3
; 1035        1740  3 scr$erase_page (1,1);           ! Start out with a shiny blank screen
; 1036        1741  3 scr$put_screen ( help_desc_0, (hlplin = .hlplin + 1), .hlpcol);
; 1037        1742  3 scr$put_screen ( help_desc_1, (hlplin = .hlplin + 1), .hlpcol);
; 1038        1743  3 scr$put_screen ( help_desc_2, (hlplin = .hlplin + 1), .hlpcol);
; 1039        1744  3 scr$put_screen ( help_desc_3, (hlplin = .hlplin + 1), .hlpcol);
; 1040        1745  3 scr$put_screen ( help_desc_4, (hlplin = .hlplin + 1), .hlpcol);
; 1041        1746  3 scr$put_screen ( help_desc_5, (hlplin = .hlplin + 1), .hlpcol);
; 1042        1747  3 scr$put_screen ( help_desc_6, (hlplin = .hlplin + 1), .hlpcol);
; 1043        1748  3 scr$put_screen ( help_desc_7, (hlplin = .hlplin + 1), .hlpcol);
; 1044        1749  3 scr$put_screen ( help_desc_8, (hlplin = .hlplin + 1), .hlpcol);
; 1045        1750  3 scr$put_screen ( help_desc_9, (hlplin = .hlplin + 1), .hlpcol);
; 1046        1751  3 scr$put_screen ( help_desc_10, (hlplin = .hlplin + 1), .hlpcol);
; 1047        1752  3 scr$put_screen ( help_desc_11, (hlplin = .hlplin + 1), .hlpcol);
; 1048        1753  3 scr$put_screen ( help_desc_12, (hlplin = .hlplin + 1), .hlpcol);
; 1049        1754  3 scr$put_screen ( help_desc_13, (hlplin = .hlplin + 1), .hlpcol);
; 1050        1755  3 scr$put_screen ( help_desc_14, (hlplin = .hlplin + 1), .hlpcol);
; 1051        1756  3 scr$put_screen ( help_desc_15, (hlplin = .hlplin + 1), .hlpcol);
; 1052        1757  3 RETURN true;
; 1053        1758  2 END;    ! Routine display_help


                                   001C 00000 DISPLAY_HELP:
                                                        .WORD   Save R2,R3,R4                          ; 1726
                 54 00000000G  00    9E 00002          MOVAB   SCR$PUT_SCREEN, R4                     ;
                 52            05    D0 00009          MOVL    #5, HLPLIN                             ; 1737
                 53            05    D0 0000C          MOVL    #5, HLPCOL                             ; 1738
                               01    DD 0000F          PUSHL   #1                                     ; 1740
                               01    DD 00011          PUSHL   #1                                     ;
           00000000G  00       02    FB 00013          CALLS   #2, SCR$ERASE_PAGE                     ;
                               53    DD 0001A          PUSHL   HLPCOL                                 ; 1741
                               52    D6 0001C          INCL    HLPLIN                                 ;
                               52    DD 0001E          PUSHL   HLPLIN                                 ;
                      0000'    CF    9F 00020          PUSHAB  HELP_DESC_0                            ;
                 64            03    FB 00024          CALLS   #3, SCR$PUT_SCREEN                     ;
                               53    DD 00027          PUSHL   HLPCOL                                 ; 1742
                               52    D6 00029          INCL    HLPLIN                                 ;
                               52    DD 0002B          PUSHL   HLPLIN                                 ;
                      0000'    CF    9F 0002D          PUSHAB  HELP_DESC_1                            ;
```

```
          64        03 FB 00031         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 00034         PUSHL   HLPCOL                          ; 1743
                    52 D6 00036         INCL    HLPLIN                          ;
                    52 DD 00038         PUSHL   HLPLIN                          ;
             0000'  CF 9F 0003A         PUSHAB  HELP_DESC_2                     ;
          64        03 FB 0003E         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 00041         PUSHL   HLPCOL                          ; 1744
                    52 D6 00043         INCL    HLPLIN                          ;
                    52 DD 00045         PUSHL   HLPLIN                          ;
             0000'  CF 9F 00047         PUSHAB  HELP_DESC_3                     ;
          64        03 FB 0004B         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 0004E         PUSHL   HLPCOL                          ; 1745
                    52 D6 00050         INCL    HLPLIN                          ;
                    52 DD 00052         PUSHL   HLPLIN                          ;
             0000'  CF 9F 00054         PUSHAB  HELP_DESC_4                     ;
          64        03 FB 00058         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 0005B         PUSHL   HLPCOL                          ; 1746
                    52 D6 0005D         INCL    HLPLIN                          ;
                    52 DD 0005F         PUSHL   HLPLIN                          ;
             0000'  CF 9F 00061         PUSHAB  HELP_DESC_5                     ;
          64        03 FB 00065         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 00068         PUSHL   HLPCOL                          ; 1747
                    52 D6 0006A         INCL    HLPLIN                          ;
                    52 DD 0006C         PUSHL   HLPLIN                          ;
             0000'  CF 9F 0006E         PUSHAB  HELP_DESC_6                     ;
          64        03 FB 00072         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 00075         PUSHL   HLPCOL                          ; 1748
                    52 D6 00077         INCL    HLPLIN                          ;
                    52 DD 00079         PUSHL   HLPLIN                          ;
             0000'  CF 9F 0007B         PUSHAB  HELP_DESC_7                     ;
          64        03 FB 0007F         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 00082         PUSHL   HLPCOL                          ; 1749
                    52 D6 00084         INCL    HLPLIN                          ;
                    52 DD 00086         PUSHL   HLPLIN                          ;
             0000'  CF 9F 00088         PUSHAB  HELP_DESC_8                     ;
          64        03 FB 0008C         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 0008F         PUSHL   HLPCOL                          ; 1750
                    52 D6 00091         INCL    HLPLIN                          ;
                    52 DD 00093         PUSHL   HLPLIN                          ;
             0000'  CF 9F 00095         PUSHAB  HELP_DESC_9                     ;
          64        03 FB 00099         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 0009C         PUSHL   HLPCOL                          ; 1751
                    52 D6 0009E         INCL    HLPLIN                          ;
                    52 DD 000A0         PUSHL   HLPLIN                          ;
             0000'  CF 9F 000A2         PUSHAB  HELP_DESC_10                    ;
          64        03 FB 000A6         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 000A9         PUSHL   HLPCOL                          ; 1752
                    52 D6 000AB         INCL    HLPLIN                          ;
                    52 DD 000AD         PUSHL   HLPLIN                          ;
             0000'  CF 9F 000AF         PUSHAB  HELP_DESC_11                    ;
          64        03 FB 000B3         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 000B6         PUSHL   HLPCOL                          ; 1753
                    52 D6 000B8         INCL    HLPLIN                          ;
                    52 DD 000BA         PUSHL   HLPLIN                          ;
             0000'  CF 9F 000BC         PUSHAB  HELP_DESC_12                    ;
          64        03 FB 000C0         CALLS   #3, SCR$PUT_SCREEN              ;
                    53 DD 000C3         PUSHL   HLPCOL                          ; 1754
```

```
                        52  D6 000C5        INCL    HLPLIN                          ;
                        52  DD 000C7        PUSHL   HLPLIN                          ;
             0000'      CF  9F 000C9        PUSHAB  HELP_DESC_13                    ;
        64              03  FB 000CD        CALLS   #3, SCR$PUT_SCREEN              ;
                        53  DD 000D0        PUSHL   HLPCOL                          ; 1755
                        52  D6 000D2        INCL    HLPLIN                          ;
                        52  DD 000D4        PUSHL   HLPLIN                          ;
             0000'      CF  9F 000D6        PUSHAB  HELP_DESC_14                    ;
        64              03  FB 000DA        CALLS   #3, SCR$PUT_SCREEN              ;
                        53  DD 000DD        PUSHL   HLPCOL                          ; 1756
                        52  D6 000DF        INCL    HLPLIN                          ;
                        52  DD 000E1        PUSHL   HLPLIN                          ;
             0000'      CF  9F 000E3        PUSHAB  HELP_DESC_15                    ;
        64              03  FB 000E7        CALLS   #3, SCR$PUT_SCREEN              ;
        50              01  D0 000EA        MOVL    #1, R0                          ; 1757
                            04 000ED        RET                                     ; 1758

; Routine Size:  238 bytes,    Routine Base:  $CODE$ + 0938

; 1054          1759  2
```

LBR
V04

6D

6F
62

```
2 %SBTTL 'main body of LBR$DUMP_CACHE';
2
2 !+++
2 !       Begin body of GLOBAL ROUTINE lbr$dump_cache ( )
2 !---
2
2 OWN
2      scr80_strng : countedstring ('[?3l'),
2      scr132_strng : countedstring ('[?3h[23;24r');
2
2 LOCAL
2      continue,
2      scr80_desc : BBLOCK [dsc$c_s_bln],
2      scr132_desc : BBLOCK [dsc$c_s_bln],
2      desc :       BBLOCK [dsc$c_s_bln],
2      buf  :       VECTOR [20,BYTE],
2      bucket,
2      entry;
2 BIND
2      context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK,
2      cache = .context [ctx$l_cache] : VECTOR,
2      command = buf[0] : BYTE;
2
2 SWITCHES NOOPTIMIZE;     ! keep compiler from stashing away buf[0] before the
2                          ! the call to scr$get_screen(desc)
2
2 IF cache EQL 0 THEN RETURN true;          ! there is no cache
2
2 CH$FILL (0,dsc$c_s_bln,scr80_desc);
2 scr80_desc [dsc$w_length] = .scr80_strng [0];
2 scr80_desc [dsc$a_pointer] = scr80_strng [1];
2 CH$FILL (0,dsc$c_s_bln,scr132_desc);
2 scr132_desc [dsc$w_length] = .scr132_strng [0];
2 scr132_desc [dsc$a_pointer] = scr132_strng [1];
2
2 perform (lib$put_output (scr132_desc) );
2 bucket = 0;       ! Set global current bucket to first bucket in Hash table
2 entry = 0;        ! Set global entry offset in linked list of cache entries
2                   !   first entry in the list.
2 scr$erase_page (1,1);             ! Start out with a shiny blank screen
2 display_hash (.bucket); ! Display the window around the current hashtable bucket
2 display_bucket (.bucket,.entry); ! Display the linked list of cache entries for
2                                  !   for the current bucket and one after current.
2 cur_vbn = 1;                     ! set for first block
2 desc [dsc$w_length] = 20;        ! Initialize input command buffer.
2 desc [dsc$a_pointer] = buf;      ! Initialize input command buffer.
2
2 continue = true;         ! Continue obtaining and executing command requests
2 WHILE .continue DO       ! Until a Quit or invalid command is entered.
3      BEGIN
3      perform ( scr$set_cursor(input_lin, input_col) );
3      perform ( scr$get_screen(desc) );   ! Read screen to obtain command
4      IF (.command GEQ %X'41') AND (.command LEQ %X'5A')
3      THEN command = .command + %X'20';
3      SELECTONE .command OF              ! Take appropriate action
3          SET
3          [%X'3F'] :  display_help();             ! ? Help with the commands
```

```
        [%X'6E'] :  up (bucket, entry);            ! 'N'orth
        [%X'73'] :  down (bucket, entry);          ! 'S'outh
        [%X'65'] :  right (bucket, entry);         ! 'E'ast
        [%X'77'] :  left (bucket, entry);          ! 'W'est
        [%X'64'] :  display_block(.bucket,.entry);! 'D'isplay block
        [%X'68'] :  display_header();              ! display 'H'eader
        [%X'6C'] :  BEGIN                          ! 'L'ocate a vbn in the cache
                    IF locate_vbn ( false, bucket, entry)
                    THEN
                        BEGIN
                        display_hash (.bucket);
                        display_bucket (.bucket,.entry);
                        END;
                    END;
        [%X'67'] :  BEGIN                          ! 'G'et a vbn and cache it
                    IF locate_vbn ( true, bucket, entry)
                    THEN
                        BEGIN
                        display_hash (.bucket);
                        display_bucket (.bucket,.entry);
                        END;
                    END;
        [%X'6F'] :  BEGIN                          ! 'O'rigin
                    bucket = 0;
                    entry = 0;
                    scr$erase_page (1,1);
                    display_hash (.bucket);
                    display_bucket (.bucket,.entry);
                    END;
        [%X'62'] :  BEGIN                          ! 'B'ack
                    entry = 0;
                    scr$erase_page (1,1);
                    display_hash (.bucket);
                    display_bucket (.bucket,.entry);
                    END;
        [%X'72'] :  BEGIN                          ! 'R'efresh the screen
                    scr$erase_page (1,1);
                    display_hash (.bucket);
                    display_bucket (.bucket,.entry);
                    END;
        [%X'6A'] :  BEGIN                          ! 'J'ump down
                    bucket = .bucket + 10;
                    display_hash (.bucket);
                    display_bucket (.bucket,.entry);
                    END;
        [%X'78'] :  BEGIN                          ! ne'X't
                    IF find_vbn (true, bucket, entry, .cur_vbn + 1)
                    THEN
                        BEGIN
                        display_hash (.bucket);
                        display_bucket (.bucket,.entry);
                        END;
                    END;
        [%X'71'] :  continue = false;              ! 'Q'uit

        [OTHERWISE] :
                    BEGIN                          ! Print the help and quit
```

```
; 1170          1874 4                          display_help();
; 1171          1875 4                          continue = false;
; 1172          1876 3                          END;
; 1173          1877 3              TES;
; 1174          1878 2          END;
; 1175          1879 2
; 1176          1880 2 perform (lib$put_output (scr80_desc) );
; 1177          1881 2 RETURN true;
; 1178          1882 1 END;      ! lbr$dump_cache


                                                                   .PSECT  $OWN$,NOEXE,2

                                                     05 00230 SCR80_STRNG:
                                                                   .BYTE   5
                                    6C  33  3F  5B  1B  00231      .ASCII  <27>\[?3l\                                ;
                                                        00236      .BLKB   2
                                                     0D 00238 SCR132_STRNG:
                                                                   .BYTE   13
    72  34  32  3B  33  32  5B  1B  68  33  3F  5B  1B  00239      .ASCII  <27>\[?3h\<27>\[23;24r\                   ;


                                                                   .PSECT  $CODE$,NOWRT,2

                                                   00FC 00000      .ENTRY  LBR$DUMP_CACHE, Save R2,R3,R4,R5,R6,R7    ; 1197
                              57 00000000G     00  9E 00002        MOVAB   SCR$ERASE_PAGE, R7                        ;
                              56 00000000G     00  9E 00009        MOVAB   LIB$PUT_OUTPUT, R6                        ;
                              5E               34  C2 00010        SUBL2   #52, SP                                   ;
                              50        0000G  CF  D0 00013        MOVL    LBR$GL_CONTROL, R0                        ; 1779
                              50           0E  A0  D0 00018        MOVL    14(R0), R0                                ;
                              50           08  A0  D0 0001C        MOVL    8(R0), R0                                 ; 1780
                                               03  12 00020        BNEQ    1$                                        ; 1786
                                             01A1  31 00022        BRW     30$                                       ;
      08           00         6E               00  2C 00025 1$:    MOVC5   #0, (SP), #0, #8, SCR80_DESC              ; 1788
                                           2C  AE     0002A                                                          ;
                    2C   AE             0000'  CF  9B 0002C        MOVZBW  SCR80_STRNG, SCR80_DESC                   ; 1789
                    30   AE             0000'  CF  9E 00032        MOVAB   SCR80_STRNG+1, SCR80_DESC+4               ; 1790
      08           00         6E               00  2C 00038        MOVC5   #0, (SP), #0, #8, SCR132_DESC             ; 1791
                                           24  AE     0003D                                                          ;
                    24   AE             0000'  CF  9B 0003F        MOVZBW  SCR132_STRNG, SCR132_DESC                 ; 1792
                    28   AE             0000'  CF  9E 00045        MOVAB   SCR132_STRNG+1, SCR132_DESC+4             ; 1793
                                           24  AE  9F 0004B        PUSHAB  SCR132_DESC                               ; 1795
                              66               01  FB 0004E        CALLS   #1, LIB$PUT_OUTPUT                        ;
                              49               50  E9 00051        BLBC    STATUS, 4$                                ;
                                               6E  7C 00054        CLRQ    ENTRY                                     ; 1797
                                               01  DD 00056        PUSHL   #1                                        ; 1799
                                               01  DD 00058        PUSHL   #1                                        ;
                              67               02  FB 0005A        CALLS   #2, SCR$ERASE_PAGE                        ;
                                           04  AE  DD 0005D        PUSHL   BUCKET                                    ; 1800
                    FD6C CF                    01  FB 00060        CALLS   #1, DISPLAY_HASH                          ;
                                               6E  DD 00065        PUSHL   ENTRY                                     ; 1801
                                           08  AE  DD 00067        PUSHL   BUCKET                                    ;
                    F9E9 CF                    02  FB 0006A        CALLS   #2, DISPLAY_BUCKET                        ;
                    0000' CF                   01  D0 0006F        MOVL    #1, CUR_VBN                               ; 1803
                      1C  AE                   14  B0 00074        MOVW    #20, DESC                                 ; 1804
```

```
       20    AE        08   AE  9E 00078         MOVAB   BUF, DESC+4                  ; 1805
             53             01  D0 0007D         MOVL    #1, CONTINUE                 ; 1807
             03             53  E8 00080 2$:     BLBS    CONTINUE, 3$                 ; 1808
                          0137  31 00083         BRW     29$                          ;
             7E             18  7D 00086 3$:     MOVQ    #24, -(SP)                   ; 1810
  00000000G  00             02  FB 00089         CALLS   #2, SCR$SET_CURSOR           ;
             0A             50  E9 00090         BLBC    STATUS, 4$                   ;
                       1C   AE  9F 00093         PUSHAB  DESC                         ; 1811
  00000000G  00             01  FB 00096         CALLS   #1, SCR$GET_SCREEN           ;
             01             50  E8 0009D 4$:     BLBS    STATUS, 5$                   ;
                                04 000A0         RET                                  ;
       41    8F        08   AE  91 000A1 5$:     CMPB    COMMAND, #65                 ; 1812
                            0B  1F 000A6         BLSSU   6$                           ;
       5A    8F        08   AE  91 000A8         CMPB    COMMAND, #90                 ;
                            04  1A 000AD         BGTRU   6$                           ;
       08    AE             20  80 000AF         ADDB2   #32, COMMAND                 ; 1813
             52        08   AE  9A 000B3 6$:     MOVZBL  COMMAND, R2                  ; 1814
             3F             52  91 000B7         CMPB    R2, #63                      ; 1816
                            07  12 000BA         BNEQ    8$                           ;
     FE51    CF             00  FB 000BC         CALLS   #0, DISPLAY_HELP             ;
                            BD  11 000C1 7$:     BRB     2$                           ;
       6E    8F             52  91 000C3 8$:     CMPB    R2, #110                     ; 1817
                            0C  12 000C7         BNEQ    9$                           ;
                            5E  DD 000C9         PUSHL   SP                           ;
                       08   AE  9F 000CB         PUSHAB  BUCKET                       ;
     FD90    CF             02  FB 000CE         CALLS   #2, UP                       ;
                            AB  11 000D3         BRB     2$                           ;
       73    8F             52  91 000D5 9$:     CMPB    R2, #115                     ; 1818
                            0C  12 000D9         BNEQ    10$                          ;
                            5E  DD 000DB         PUSHL   SP                           ;
                       08   AE  9F 000DD         PUSHAB  BUCKET                       ;
     FDB1    CF             02  FB 000E0         CALLS   #2, DOWN                     ;
                            99  11 000E5         BRB     2$                           ;
       65    8F             52  91 000E7 10$:    CMPB    R2, #101                     ; 1819
                            0C  12 000EB         BNEQ    11$                          ;
                            5E  DD 000ED         PUSHL   SP                           ;
                       08   AE  9F 000EF         PUSHAB  BUCKET                       ;
     FDDF    CF             02  FB 000F2         CALLS   #2, RIGHT                    ;
                            87  11 000F7         BRB     2$                           ;
       77    8F             52  91 000F9 11$:    CMPB    R2, #119                     ; 1820
                            0C  12 000FD         BNEQ    12$                          ;
                            5E  DD 000FF         PUSHL   SP                           ;
                       08   AE  9F 00101         PUSHAB  BUCKET                       ;
     FDE8    CF             02  FB 00104         CALLS   #2, LEFT                     ;
                            B6  11 00109         BRB     7$                           ;
       64    8F             52  91 0010B 12$:    CMPB    R2, #100                     ; 1821
                            0C  12 0010F         BNEQ    13$                          ;
                            6E  DD 00111         PUSHL   ENTRY                        ;
                       08   AE  DD 00113         PUSHL   BUCKET                       ;
     FB96    CF             02  FB 00116         CALLS   #2, DISPLAY_BLOCK            ;
                            A4  11 0011B         BRB     7$                           ;
       68    8F             52  91 0011D 13$:    CMPB    R2, #104                     ; 1822
                            07  12 00121         BNEQ    14$                          ;
     FD2B    CF             00  FB 00123         CALLS   #0, DISPLAY_HEADER           ;
                            97  11 00128         BRB     7$                           ;
       6C    8F             52  91 0012A 14$:    CMPB    R2, #108                     ; 1823
                            09  12 0012E         BNEQ    15$                          ;
```

```
                                   5E  DD 00130          PUSHL   SP                                    ; 1824
                            08     AE  9F 00132          PUSHAB  BUCKET                                ;
                                   7E  D4 00135          CLRL    -(SP)                                 ;
                                   0D  11 00137          BRB     16$                                   ;
              67     8F            52  91 00139 15$:     CMPB    R2, #103                              ; 1831
                                   0E  12 0013D          BNEQ    17$                                   ;
                                   5E  DD 0013F          PUSHL   SP                                    ; 1832
                            08     AE  9F 00141          PUSHAB  BUCKET                                ;
                                   01  DD 00144          PUSHL   #1                                    ;
            FC3A     CF            03  FB 00146 16$:     CALLS   #3, LOCATE_VBN                        ;
                                   49  11 0014B          BRB     24$                                   ;
              6F     8F            52  91 0014D 17$:     CMPB    R2, #111                              ; 1839
                                   05  12 00151          BNEQ    18$                                   ;
                            04     AE  D4 00153          CLRL    BUCKET                                ; 1840
                                   06  11 00156          BRB     19$                                   ; 1841
              62     8F            52  91 00158 18$:     CMPB    R2, #98                               ; 1846
                                   04  12 0015C          BNEQ    20$                                   ;
                                   6E  D4 0015E 19$:     CLRL    ENTRY                                 ; 1847
                                   06  11 00160          BRB     21$                                   ; 1848
              72     8F            52  91 00162 20$:     CMPB    R2, #114                              ; 1852
                                   09  12 00166          BNEQ    22$                                   ;
                                   01  DD 00168 21$:     PUSHL   #1                                    ; 1853
                                   01  DD 0016A          PUSHL   #1                                    ;
                     67            02  FB 0016C          CALLS   #2, SCR$ERASE_PAGE                    ;
                                   28  11 0016F          BRB     25$                                   ; 1854
              6A     8F            52  91 00171 22$:     CMPB    R2, #106                              ; 1857
                                   06  12 00175          BNEQ    23$                                   ;
              04     AE            0A  C0 00177          ADDL2   #10, BUCKET                           ; 1858
                                   1C  11 0017B          BRB     25$                                   ; 1859
              78     8F            52  91 0017D 23$:     CMPB    R2, #120                              ; 1862
                                   2A  12 00181          BNEQ    26$                                   ;
       7E  0000'     CF            01  C1 00183          ADDL3   #1, CUR_VBN, -(SP)                    ; 1863
                            04     AE  9F 00189          PUSHAB  ENTRY                                 ;
                            0C     AE  9F 0018C          PUSHAB  BUCKET                                ;
                                   01  DD 0018F          PUSHL   #1                                    ;
            FB75     CF            04  FB 00191          CALLS   #4, FIND_VBN                          ;
                     21            50  E9 00196 24$:     BLBC    R0, 28$                               ;
                            04     AE  DD 00199 25$:     PUSHL   BUCKET                                ; 1866
            FC30     CF            01  FB 0019C          CALLS   #1, DISPLAY_HASH                      ;
                                   6E  DD 001A1          PUSHL   ENTRY                                 ; 1867
                            08     AE  DD 001A3          PUSHL   BUCKET                                ;
            F8AD     CF            02  FB 001A6          CALLS   #2, DISPLAY_BUCKET                    ;
                                   0D  11 001AB          BRB     28$                                   ; 1814
              71     8F            52  91 001AD 26$:     CMPB    R2, #113                              ; 1870
                                   05  13 001B1          BEQL    27$                                   ;
            FD5A     CF            00  FB 001B3          CALLS   #0, DISPLAY_HELP                      ; 1874
                                   53  D4 001B8 27$:     CLRL    CONTINUE                              ; 1875
                                 FEC3  31 001BA 28$:     BRW     2$                                    ; 1808
                            2C     AE  9F 001BD 29$:     PUSHAB  SCR80_DESC                            ; 1880
                     66            01  FB 001C0          CALLS   #1, LIB$PUT_OUTPUT                    ;
                     03            50  E9 001C3          BLBC    STATUS, 31$                           ;
                     50            01  D0 001C6 30$:     MOVL    #1, R0                                ; 1881
                                       04 001C9 31$:     RET                                           ; 1882

; Routine Size:  458 bytes,    Routine Base:  $CODE$ + 0A26
```

```
; 1179          1883  1
; 1180          1884  1 END
; 1181          1885  0 ELUDOM



;                                PSECT SUMMARY
;
;       Name                     Bytes                       Attributes
;
;   $OWN$                          582  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $PLIT$                         764  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;   $CODE$                        3056  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        14         0         581         00:01.0



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

; Size:           3056 code + 1346 data bytes
; Run Time:          00:55.9
; Elapsed Time:      01:52.2
; Lines/CPU Min:       2024
; Lexemes/CPU-Min: 23029
; Memory Used:  250 pages
; Compilation Complete
```

0197 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

CACHE
LIS

DATA
LIS

LBR

LBRSHR
MAP

OLDLIBFMT
MDL

DUMP
LIS

LBR
MDL

PREFIX
REQ

LBRUSR
SDL

0198 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

GETHELP
LIS

INDEX
LIS

GETPUT
LIS

GETMEM
LIS