E 9

```
LLL              BBBBBBBBBBBB     RRRRRRRRRRRR
LLL              BBBBBBBBBBBB     RRRRRRRRRRRR
LLL              BBBBBBBBBBBB     RRRRRRRRRRRR
LLL              BBB         BBB  RR          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
```

```
_$2

Val
---
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
002
00F
00F
00F
00F
00F
00F
00F
7FF
7FF
7FF
7FF
7FF
7FF
7FF
7FF
```

```
**FILE**ID**OPENCLOSE                                  G  8                                                   LB
                                                                                                             VO
```

```
  000000    PPPPPPPP    EEEEEEEEEE  NN      NN    CCCCCCCC  LL            000000      SSSSSSSS  EEEEEEEEEE
  000000    PPPPPPPP    EEEEEEEEEE  NN      NN    CCCCCCCC  LL            000000      SSSSSSSS  EEEEEEEEEE
00      00  PP      PP  EE          NN      NN  CC          LL          00      00  SS          EE
00      00  PP      PP  EE          NN      NN  CC          LL          00      00  SS          EE
00      00  PP      PP  EE          NNNN    NN  CC          LL          00      00  SS          EE
00      00  PP      PP  EE          NNNN    NN  CC          LL          00      00  SS          EE
00      00  PPPPPPPP    EEEEEEEE    NN  NN  NN  CC          LL          00      00    SSSSSS    EEEEEEEE
00      00  PPPPPPPP    EEEEEEEE    NN  NN  NN  CC          LL          00      00    SSSSSS    EEEEEEEE
00      00  PP          EE          NN    NNNN  CC          LL          00      00          SS  EE
00      00  PP          EE          NN    NNNN  CC          LL          00      00          SS  EE
00      00  PP          EE          NN      NN  CC          LL          00      00          SS  EE              ....
00      00  PP          EE          NN      NN  CC          LL          00      00          SS  EE              ....
  000000    PP          EEEEEEEEEE  NN      NN    CCCCCCCC  LLLLLLLLLL    000000    SSSSSSSS    EEEEEEEEEE      ....
  000000    PP          EEEEEEEEEE  NN      NN    CCCCCCCC  LLLLLLLLLL    000000    SSSSSSSS    EEEEEEEEEE      ....

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_OPENCLOSE (                       . Open/close routines for LIBRARIAN
0                 LANGUAGE (BLISS32),
0                 IDENT = 'V04-000'
0                 ) =
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:  Benn Schreiber,       CREATION DATE:  7-Jun-1979
1 !
1 ! MODIFIED BY:
1 !
1 !       V03-009 GJA0098         Greg Awdziewicz         13-Aug-1984
1 !               - Allow larger buffers for reading DCX encoded libraries.
1 !               - Replace references to obj$c_maxrecsiz with lbr$_maxrecsiz
1 !               to be consistent.
1 !
1 !       V03-008 JWT0186         Jim Teague              6-Jul-1984
1 !               Read up to 10 map blocks at once in order to speed
```

```
:   58          0058  1 !                up processing of DCX data-reduced libraries.
:   59          0059  1 !
:   60          0060  1 !        V03-007 GJA0082         Greg Awdziewicz         10-Apr-1984
:   61          0061  1 !                - Turn the OFP (output file parsing) bit in all cases if
:   62          0062  1 !                we are creating a library.  (It had been made conditional,
:   63          0063  1 !                ie, set equal to NULLPARAMETER(5), as part of v3-003)
:   64          0064  1 !
:   65          0065  1 !        V03-006 JWT0114         Jim Teague      20-Apr-1983
:   66          0066  1 !                Activate DCXSHR dynamically when needed.
:   67          0067  1 !
:   68          0068  1 !        V03-005 JWT0101         Jim Teague      08-Mar-1983
:   69          0069  1 !                Fix error in the checking of maximum control
:   70          0070  1 !                index number.
:   71          0071  1 !
:   72          0072  1 !        V03-004 JWT0085         Jim Teague      11-Jan-1982
:   73          0073  1 !                Use lib$get_ef to set a timer wait for locked
:   74          0074  1 !                libraries.  Free with lib$free_ef.  Also add
:   75          0075  1 !                new sanity id for compressed libraries.
:   76          0076  1 !
:   77          0077  1 !        V03-003 JWT0067         Jim Teague      11-Nov-1982
:   78          0078  1 !                Enlarged the space allocated for DCX records;
:   79          0079  1 !                made LBR clean up after itself more thoroughly;
:   80          0080  1 !                for library creation, if a related name block is
:   81          0081  1 !                passed, then USE IT.
:   82          0082  1 !
:   83          0083  1 !        V03-002 JWT0062         Jim Teague      26-Oct-1982
:   84          0084  1 !                Made DCX descriptors static as part of CTX block.
:   85          0085  1 !
:   86          0086  1 !        V03-001 JWT0056         Jim Teague      16-Sep-1982
:   87          0087  1 !                Implemented interface to DCX data compression/expansion
:   88          0088  1 !                facility.
:   89          0089  1 !
:   90          0090  1 !--
:   91          0091  1
:   92          0092  1
```

```
    1 %SBTTL  'Declarations';
    1 LIBRARY
    1         'SYS$LIBRARY:STARLET.L32';      !System macros
    1 REQUIRE
    1         'PREFIX';                       !Librarian general definitions
    1 REQUIRE
    1         'LBRDEF';                       !Librarian structure definitions
    1 REQUIRE
    1         'OLDFMTDEF';                    !Old format (VMS R1) library structure
    1
    1
    1 ! Replacing uses of obj$c_maxrecsiz with lbr$c_maxrecsiz requires that
    1 ! they have the same value.  Also, provide a larger value for DCX
    1 ! encoded records since they may in fact grow when they are "reduced" --
    1 ! e.g., adding a message pointer object module to an object library.
    1 !
  U 1 %IF lbr$c_maxrecsiz NEQ obj$c_maxrecsiz %THEN
  U 1 %ERROR ('lbr$c_maxrecsiz is not equivalent to obj$c_maxrecsiz')
    1 %FI
    1
    1 LITERAL
    1     lbr_dcx$c_maxrecsiz= 2 * lbr$c_maxrecsiz,   ! Allow DCX maximum record size
    1                                                 ! to be larger than normal.
    1     num_dcx_routines = 10,
    1     top_index = 1;
    1
    1 EXTERNAL LITERAL
    1     !
    1     !   success codes
    1     !
    1     dcx$_normal,
    1     dcx$_again,
    1     lbr$_normal,        ! success
    1     lbr$_oldlibrary,    ! old format library opened
    1
    1     !
    1     !   Warning codes
    1     !
    1     lbr$_oldmismch,     ! old format library type mismatch
    1     lbr$_typmismch,     ! library type mismatch
    1     lbr$_errclose,      ! Error occurred in closing library
    1
    1     !
    1     !   Error codes
    1     !
    1     lbr$_illctl,        ! illegal control index
    1     lbr$_illcreopt,     ! illegal create options
    1     lbr$_illfmt,        ! illegal library format
    1     lbr$_illfunc,       ! illegal library function
    1     lbr$_illtyp,        ! illegal library type
    1     lbr$_libnotopn,     ! library not open
    1     lbr$_libopn,        ! library already open
    1     lbr$_nofilnam,      ! no file specification found
    1     lbr$_toomnylib;     ! too many libraries open
    1
    1 EXTERNAL
    1     dcx_analyze_init,
        dcx_analyze_data,
        dcx_analyze_done,
        dcx_expand_init ,
        dcx_expand_done ,
        dcx_compress_init,
        dcx_compress_done,
        dcx_make_map    ,
        dcxshr_address,                          !Base address of dcxshr if mapped
        lbr$gl_control : REF BLOCK [,BYTE],      !Pointer to current control table
        lbr$al_ctltab : VECTOR [lbr$c_maxctl],   !Table of pointers to all known control tables
        lbr$gl_hictl,                            !Highest control number allocated
        mem$l_maxblk,                            !Max size of expand region request to get memory
        mem$l_memexp,                            !Number of pages in expand region request
        lbr$gl_maxread,                          !Max. number blocks to read at once
        lbr$gl_rmsstv,                           !Return RMS STV code here on errors
        lbr$gt_lbrver : VECTOR [32, BYTE];       !ASCIC string of librarian ID

    EXTERNAL ROUTINE
        lib$adr_image,
        lib$get_ef  :     addressing_mode (general),
        lib$free_ef :     addressing_mode (general),
        alloc_block:  jsb_2,
        lbr$get_index,
        lbr$find,
        lbr$get_record,
        lbr_old_lib_dat,                         !Extract info for old library
        read_block : JSB_2,                      !Read a disk block
        write_block : JSB_2,                     !Write a disk block
        add_cache : JSB_2,                       !Add entry to cache list
        dealloc_cache,                           ! Empty disk block cache
        validate_ctl : JSB_1,                    !Validate control blocks
        get_mem : JSB_2,                         ! Allocate dynamic memory
        get_zmem : JSB_2,                        !Allocate and zero virtual memory
        dealloc_mem : JSB_2;                     ! Deallocate dynamic memory

    FORWARD ROUTINE
        lbr$load_dcx,
        lbr$dcx_map,
        dcx_it,
        prealloc_index,                          !Preallocate index blocks
        all_control_idx,                         !Allocate a control table index number
        dea_control_idx : NOVALUE,               !Deallocate a control table index number
        lbr$close,                               !Close open library file, delete
                                                 ! all allocated memory
        lbr_deal_mem : NOVALUE,                  !Deallocate all allocated memory
        read_n_map_blocks;                       ! Read up to 10 DCX map blocks

    OWN
        dcxshr_string : countedstring('DCXSHR'),
        default_string : countedstring('SYS$SHARE:.EXE'),
        lib_control_index,
        local_dcx_context;
```

```
1 %SBTTL  'LBR$INI_CONTROL';
1 GLOBAL ROUTINE lbr$ini_control (control_index, func, type, namblk) =
2 BEGIN
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine initializes a control table for use by the library
2 !       access procedures.
2 !
2 ! CALLING SEQUENCE:
2 !
2 !       STATUS = LBR$INI_CONTROL(control_index,func,type,namblk)
2 !
2 ! INPUT PARAMETERS:
2 !       func                    Address of a longword containing the desired
2 !                                function - LBR$C_CREATE, LBR$C_READ, or
2 !                               LBR$C_UPDATE.
2 !       type                    The type of library expected to open.  If not
2 !                                supplied, or 0, no type checking is done.
2 !       namblk                  The (optional) address of a NAM block.
2 !                                If it has been previously filled in,
2 !                                the file will be opened by NAM block,
2 !                                otherwise the NAM block will be filled
2 !                                in for later use.
2 !
2 ! IMPLICIT INPUTS:
2 !       NONE
2 !
2 ! OUTPUT PARAMETERS:
2 !
2 !       control_index           Receives the control_table index to use
2 !                                on all subsequent calls to the librarian
2 !                                for this library.
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       The control_table is initialized.
2 !
2 ! ROUTINE VALUE:
2 !
2 !       lbr$_normal             Control table initialized
2 !       lbr$_illtyp             Illegal library type specified
2 !       lbr$_illfunc            Illegal function requested
2 !       lbr$_toomnylib          Too many libraries
2 !
2 !--
2
2 MAP
2     namblk : REF BBLOCK;
2 BUILTIN
2     NULLPARAMETER;
2
2 IF NOT NULLPARAMETER (3)                ! If type specified,
2 THEN
2     IF ..type GTRU lbr$c_typ_decmx      ! If expected type illegal,
;  261          1082  2         AND ..type LSSU lbr$c_typ_rdec
;  262          1083  2     THEN
;  263          1084  2         RETURN lbr$_illtyp;            ! Return with error
;  264          1085  2
;  265          1086  2 IF ..func GTRU lbr$c_maxfunc             ! If function is illegal
;  266          1087  2 THEN RETURN lbr$_illfunc;               ! then return with error
;  267          1088  2
;  268          1089  2 perform (all_control_idx (.control_index, lbr$gl_control));!Get an index number
;  269          1090  2                                          !(also allocate control table)
;  270          1091  2
;  271          1092  2 lbr$gl_control [lbr$b_id] = lbr$c_ctltblid;
;  272          1093  2 lbr$gl_control [.br$b_tblsiz] = lbr$c_length;
;  273          1094  2 lbr$gl_control [lbr$b_func] = ..func;   ! Set function code
;  274          1095  2
;  275          1096  2 IF NOT NULLPARAMETER (3)                 ! If type specified,
;  276          1097  2 THEN
;  277          1098  2     lbr$gl_control [lbr$b_type] = ..type; ! Set type of library expected
;  278          1099  2
;  279          1100  2 IF NOT NULLPARAMETER (4)
;  280          1101  2 THEN lbr$gl_control [lbr$l_usrnam] = .namblk;
;  281          1102  2 RETURN lbr$_normal
;  282          1103  1 END;                                     ! Of LBR$INI_CONTROL


                                                               .TITLE  LBR_OPENCLOSE
                                                               .IDENT  \V04-000\

                                                               .PSECT  $OWN$,NOEXE,2

                                          06  00000 DCXSHR_STRING:
                                                               .BYTE   6
                         52  48  53  58  43  44  00001         .ASCII  \DCXSHR\                        ;
                                                  00007        .BLKB   1
                                          0E  00008 DEFAULT_STRING:
                                                               .BYTE   14                              ;
 45  58  45  2E  3A  45  52  41  48  53  24  53  59  53  00009  .ASCII  \SYS$SHARE:.EXE\                ;
                                                  00017        .BLKB   1
                                              00018 LIB_CONTROL_INDEX:
                                                               .BLKB   4
                                              0001C LOCAL_DCX_CONTEXT:
                                                               .BLKB   4

                                                               .EXTRN  DCX$_NORMAL, DCX$_AGAIN
                                                               .EXTRN  LBR$_NORMAL, LBR$_OLDLIBRARY
                                                               .EXTRN  LBR$_OLDMISMCH, LBR$_TYPMISMCH
                                                               .EXTRN  LBR$_ERRCLOSE, LBR$_ILLCTL
                                                               .EXTRN  LBR$_ILLCREOPT, LBR$_ILLFMT
                                                               .EXTRN  LBR$_ILLFUNC, LBR$_ILLTYP
                                                               .EXTRN  LBR$_LIBNOTOPN, LBR$_LIBOPN
                                                               .EXTRN  LBR$_NOFILNAM, LBR$_TOOMNYLIB
                                                               .EXTRN  DCX_ANALYZE_INIT
                                                               .EXTRN  DCX_ANALYZE_DATA
                                                               .EXTRN  DCX_ANALYZE_DONE
                                                               .EXTRN  DCX_EXPAND_INIT
                                                               .EXTRN  DCX_EXPAND_DONE
                                                               .EXTRN  DCX_COMPRESS_INIT
                                                               .EXTRN  DCX_COMPRESS_DONE
```

```
                                                                        .EXTRN  DCX_MAKE_MAP, DCXSHR_ADDRESS
                                                                        .EXTRN  LBR$GL_CONTROL, LBR$AL_CTLTAB
                                                                        .EXTRN  LBR$GL_HICTL, MEM$L_MAXBLK
                                                                        .EXTRN  MEM$L_MEMEXP, LBR$GL_MAXREAD
                                                                        .EXTRN  LBR$GL_RMSSTV, LBR$GT_LBRVER
                                                                        .EXTRN  LIB$ADR_IMAGE, LIB$GET_EF
                                                                        .EXTRN  LIB$FREE_EF, ALLOC_BLOCK
                                                                        .EXTRN  LBR$GET_INDEX, LBR$FIND
                                                                        .EXTRN  LBR$GET_RECORD, LBR_OLD_LIB_DAT
                                                                        .EXTRN  READ_BLOCK, WRITE_BLOCK
                                                                        .EXTRN  ADD_CACHE, DEALLOC_CACHE
                                                                        .EXTRN  VALIDATE_CTL, GET_MEM
                                                                        .EXTRN  GET_ZMEM, DEALLOC_MEM

                                                                        .PSECT  $CODE$,NOWRT,2

                                                  0000 00000            .ENTRY  LBR$INI_CONTROL, Save nothing                         ; 1026
                           03               6C      91 00002            CMPB    (AP), #3                                              ; 1079
                                            1D      1F 00005            BLSSU   1$                                                    ;
                                    0C      AC      D5 00007            TSTL    12(AP)                                                ;
                                            18      13 0000A            BEQL    1$                                                    ;
                           05       0C      BC      D1 0000C            CMPL    @TYPE, #5                                             ; 1081
                                            12      1B 00010            BLEQU   1$                                                    ;
                 0000007F  8F       0C      BC      D1 00012            CMPL    @TYPE, #127                                           ; 1082
                                            08      1E 0001A            BGEQU   1$                                                    ;
                           50 00000000G     8F      D0 0001C            MOVL    #LBR$_ILLTYP, R0                                      ; 1084
                                                    04 00023            RET                                                           ;
                           02       08      BC      D1 00024 1$:        CMPL    @FUNC, #2                                             ; 1086
                                            08      1B 00028            BLEQU   2$                                                    ;
                           50 00000000G     8F      D0 0002A            MOVL    #LBR$_ILLFUNC, R0                                     ; 1087
                                                    04 00031            RET                                                           ;
                                 0000G      CF      9F 00032 2$:        PUSHAB  LBR$GL_CONTROL                                        ; 1089
                                    04      AC      DD 00036            PUSHL   CONTROL_INDEX                                         ;
                     0000V  CF              02      FB 00039            CALLS   #2, ALL_CONTROL_IDX                                   ;
                            34              50      E9 0003E            BLBC    STATUS, 5$                                            ;
                            50   0000G      CF      D0 00041            MOVL    LBR$GL_CONTROL, R0                                    ; 1092
                            60    1ECB      8F      B0 00046            MOVW    #7883, (R0)                                           ;
                      03    A0      08      BC      90 0004B            MOVB    @FUNC, 3(R0)                                          ; 1094
                            03              6C      91 00050            CMPB    (AP), #3                                              ; 1096
                                            0A      1F 00053            BLSSU   3$                                                    ;
                                    0C      AC      D5 00055            TSTL    12(AP)                                                ;
                                            05      13 00058            BEQL    3$                                                    ;
                      02    A0      0C      BC      90 0005A            MOVB    @TYPE, 2(R0)                                          ; 1098
                            04              6C      91 0005F 3$:        CMPB    (AP), #4                                              ; 1100
                                            0A      1F 00062            BLSSU   4$                                                    ;
                                    10      AC      D5 00064            TSTL    16(AP)                                                ;
                                            05      13 00067            BEQL    4$                                                    ;
                      16    A0      10      AC      D0 00069            MOVL    NAMBLK, 22(R0)                                        ; 1101
                            50 00000000G    8F      D0 0006E 4$:        MOVL    #LBR$_NORMAL, R0                                      ; 1102
                                                    04 00075 5$:        RET                                                           ; 1103

; Routine Size:  118 bytes,    Routine Base:  $CODE$ + 0000
```

```
1 %SBTTL  'LBR$DCX_MAP';
1 GLOBAL ROUTINE lbr$dcx_map (ctl_index,  dcx_map_desc) =
2 BEGIN
2
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine provides a DCX map to the calling routine.
2 !       If a data-reduced library is being opened, the existing
2 !       DCX map is used.  Otherwise a new map is produced by
2 !       analyzing the contents of all modules in the library.
2 !
2 ! INPUT PARAMETERS:
2 !       ctl_index       The control index for the library.
2 !
2 ! IMPLICIT INPUTS:
2 !       NONE
2 !
2 ! OUTPUT PARAMETERS:
2 !       dcx_map_dsc     The address of a two-longword descriptor
2 !                       into which the DCX map length and address
2 !                       are stored
2 !
2 ! IMPLICIT OUTPUTS:
2 !       NONE
2 !
2 !
2 !--
2 BUILTIN
2     NULLPARAMETER;
2 MAP
2     dcx_map_desc  : REF VECTOR;
2
2 LOCAL
2     index,
2     status,
2     ok;
2
2 if .dcxshr_address eql 0
2 then
2     perform(lbr$load_dcx());
2
2 !
2 ! Input library is in expanded format
2 !
2 IF NOT NULLPARAMETER(1)
2 THEN
3     BEGIN
4     IF NOT (status = validate_ctl(..ctl_index))
3     THEN
3         RETURN .status;
3
3     index = top_index;
3     lib_control_index = ..ctl_index;
3     perform ( (.dcx_analyze_init) ( local_dcx_context ));
3
```

```
    DO
        rms_perform ( lbr$get_index (lib_control_index, index, dcx_it))
    WHILE
        (ok = (.dcx_make_map) (local_dcx_context, dcx_map_desc [1], dcx_map_desc [0] )) EQL dcx$_again ;

    IF .ok
    THEN
        perform ( (.dcx_analyze_done) ( local_dcx_context ) );  ! free vm in dcx

    END
!
! Input library is in compressed format
!
ELSE
    BEGIN
    LOCAL
        header : REF BBLOCK,
        mapvbn,
        block_addr,
        map_begin,
        map_moved,
        map_left,
        map_pointer,
        map_blocks,
        blocks_left;

    header = .lbr$gl_control[lbr$l_hdrptr];
    mapvbn = .header[lhd$l_dcxmapvbn];
    perform(read_block(.mapvbn, block_addr));
    perform(get_mem(dcx_map_desc[0] = ..block_addr, map_pointer));
    map_begin = .map_pointer;
    map_left = .dcx_map_desc[0];
    CH$MOVE(map_moved = MIN(.dcx_map_desc[0],lbr$c_pagesize-4), .block_addr+4, dcx_map_desc[1] = .map_pointe
    IF (blocks_left = (map_blocks = .dcx_map_desc[0] / lbr$c_pagesize +
            (IF (.dcx_map_desc[0] MOD lbr$c_pagesize) GTR 0 THEN 1 ELSE 0)) - 1) GTR 0
    THEN
        BEGIN
        mapvbn = .mapvbn + 1;
        INCR i FROM 2 TO .map_blocks BY 10 DO
            BEGIN
            LOCAL
                blocks_read;
            perform(read_n_map_blocks(.mapvbn , block_addr,
                        blocks_read = MIN (.blocks_left, 10) ));
            mapvbn = .mapvbn + .blocks_read;
            map_pointer = .map_pointer + .map_moved;
            map_left = .map_left - .map_moved;
            blocks_left = .blocks_left - .blocks_read;
            CH$MOVE(map_moved = MIN(.map_left,.blocks_read * lbr$c_pagesize),.block_addr, .map_pointer);
            END;
        END;

    ok = dcx$_normal;
    END;

RETURN .ok;
END;
```

```
                                0FFC 00000      .ENTRY  LBR$DCX_MAP, Save R2,R3,R4,R5,R6,R7,R8,R9,-  ; 1105
                                                        R10,R11                                      ;
            5E                10  C2 00002      SUBL2   #16, SP                                      ;
                  0000G       CF  D5 00005      TSTL    DCXSHR_ADDRESS                               ; 1143
                              08  12 00009      BNEQ    1$                                           ;
    0000V   CF                00  FB 0000B      CALLS   #0, LBR$LOAD_DCX                             ; 1145
            3D                50  E9 00010      BLBC    STATUS, 3$                                   ;
            56       08       AC  D0 00013 1$:  MOVL    DCX_MAP_DESC, R6                             ; 1164
                              6C  95 00017      TSTB    (AP)                                         ; 1150
                              65  13 00019      BEQL    6$                                           ;
                     04       AC  D5 0001B      TSTL    4(AP)                                        ;
                              60  13 0001E      BEQL    6$                                           ;
            50       04       BC  D0 00020      MOVL    @CTL_INDEX, R0                               ; 1153
                           0000G  30 00024      BSBW    VALIDATE_CTL                                 ;
            6E                50  E9 00027      BLBC    STATUS, 7$                                   ;
       04   AE                01  D0 0002A      MOVL    #1, INDEX                                    ; 1157
    0000'   CF       04       BC  D0 0002E      MOVL    @CTL_INDEX, LIB_CONTROL_INDEX                ; 1158
                  0000'       CF  9F 00034      PUSHAB  LOCAL_DCX_CONTEXT                            ; 1159
    0000G   DF                01  FB 00038      CALLS   #1, @DCX_ANALYZE_INIT                        ;
            6C                50  E9 0003D      BLBC    STATUS, 8$                                   ;
                  0000V       CF  9F 00040 2$:  PUSHAB  DCX_IT                                       ; 1162
                     08       AE  9F 00044      PUSHAB  INDEX                                        ;
                  0000'       CF  9F 00047      PUSHAB  LIB_CONTROL_INDEX                            ;
    0000G   CF                03  FB 0004B      CALLS   #3, LBR$GET_INDEX                            ;
            59                50  E9 00050 3$:  BLBC    STATUS, 8$                                   ;
                              56  DD 00053      PUSHL   R6                                           ; 1164
                     04       A6  9F 00055      PUSHAB  4(R6)                                        ;
                  0000'       CF  9F 00058      PUSHAB  LOCAL_DCX_CONTEXT                            ;
    0000G   DF                03  FB 0005C      CALLS   #3, @DCX_MAKE_MAP                            ;
            6E                50  D0 00061      MOVL    R0, OK                                       ;
00000000G   8F                6E  D1 00064      CMPL    OK, #DCX$_AGAIN                              ;
                              D3  13 0006B      BEQL    2$                                           ;
            03                6E  E8 0006D      BLBS    OK, 5$                                       ; 1166
                            00E5  31 00070 4$:  BRW     18$                                          ;
                  0000'       CF  9F 00073 5$:  PUSHAB  LOCAL_DCX_CONTEXT                            ; 1168
    0000G   DF                01  FB 00077      CALLS   #1, @DCX_ANALYZE_DONE                        ;
            F1                50  E8 0007C      BLBS    STATUS, 4$                                   ;
                                  04 0007F      RET                                                  ;
            50    0000G       CF  D0 00080 6$:  MOVL    LBR$GL_CONTROL, R0                           ; 1187
            50       0A       A0  D0 00085      MOVL    10(R0), HEADER                               ;
            5B     008C       C0  D0 00089      MOVL    140(HEADER), MAPVBN                          ; 1188
            51       0C       AE  9E 0008E      MOVAB   BLOCK_ADDR, R1                               ; 1189
            50                5B  D0 00092      MOVL    MAPVBN, R0                                   ;
                           0000G  30 00095      BSBW    READ_BLOCK                                   ;
            11                50  E9 00098 7$:  BLBC    STATUS, 8$                                   ;
            51       08       AE  9E 0009B      MOVAB   MAP_POINTER, R1                              ; 1190
            52       0C       AE  D0 0009F      MOVL    BLOCK_ADDR, R2                               ;
            66                62  D0 000A3      MOVL    (R2), (R6)                                   ;
            50                66  D0 000A6      MOVL    (R6), R0                                     ;
                           0000G  30 000A9      BSBW    GET_MEM                                      ;
            74                50  E9 000AC 8$:  BLBC    STATUS, 14$                                  ;
            50       08       AE  D0 000AF      MOVL    MAP_POINTER, MAP_BEGIN                       ; 1191
```

```
                                        58             66 D0 000B3          MOVL     (R6), MAP_LEFT                               ; 1192
                                        50             66 D0 000B6          MOVL     (R6), R0                                     ; 1193
                             000001FC   8F             50 D1 000B9          CMPL     R0, #508                                     ;
                                        05             15 15 000C0          BLEQ     9$                                           ;
                                        50      01FC   8F 3C 000C2          MOVZWL   #508, R0                                     ;
                                        59             50 D0 000C7 9$:      MOVL     R0, MAP_MOVED                                ;
                                        51         08  AE D0 000CA          MOVL     MAP_POINTER, R1                              ;
                                   04   A6             51 D0 000CE          MOVL     R1, 4(R6)                                    ;
                          61       04   A2             50 28 000D2          MOVC3    R0, 4(R2), (R1)                              ;
                          50            66 00000200    8F C7 000D7          DIVL3    #512, (R6), R0                               ; 1194
            7E            00            66             01 7A 000DF          EMUL     #1, (R6), #0, -(SP)                          ; 1195
            51            51            8E 00000200    8F 7B 000E4          EDIV     #512, (SP)+, R1, R1                          ;
                                                       51 D5 000ED          TSTL     R1                                           ;
                                                       05 15 000EF          BLEQ     10$                                          ;
                                        51             01 D0 000F1          MOVL     #1, R1                                       ;
                                                       02 11 000F4          BRB      11$                                          ;
                                                       51 D4 000F6 10$:     CLRL     R1                                           ;
                          57            50             51 C1 000F8 11$:     ADDL3    R1, R0, MAP_BLOCKS                           ;
                                        56         FF  A7 9E 000FC          MOVAB    -1(R7), BLOCKS_LEFT                          ;
                                                       4F 15 00100          BLEQ     17$                                          ;
                                                       5B D6 00102          INCL     MAPVBN                                       ; 1198
                                        5A             08 CE 00104          MNEGL    #8, I                                        ; 1199
                                                       42 11 00107          BRB      16$                                          ;
                                        50             56 D0 00109 12$:     MOVL     BLOCKS_LEFT, R0                              ; 1204
                                        0A             50 D1 0010C          CMPL     R0, #10                                      ;
                                                       03 15 0010F          BLEQ     13$                                          ;
                                        50             0A D0 00111          MOVL     #10, R0                                      ;
                                        52             50 D0 00114 13$:     MOVL     R0, BLOCKS_READ                              ;
                                                       50 DD 00117          PUSHL    R0                                           ;
                                                   10  AE 9F 00119          PUSHAB   BLOCK_ADDR                                   ;
                                                       5B DD 0011C          PUSHL    MAPVBN                                       ;
                                0000V   CF             03 FB 0011E          CALLS    #3, READ_N_MAP_BLOCKS                        ;
                                        35             50 E9 00123 14$:     BLBC     STATUS, 19$                                  ;
                                        5B             52 C0 00126          ADDL2    BLOCKS_READ, MAPVBN                          ; 1205
                                   08   AE             59 C0 00129          ADDL2    MAP_MOVED, MAP_POINTER                       ; 1206
                                        58             59 C2 0012D          SUBL2    MAP_MOVED, MAP_LEFT                          ; 1207
                                        56             52 C2 00130          SUBL2    BLOCKS_READ, BLOCKS_LEFT                     ; 1208
                          52            52             09 78 00133          ASHL     #9, R2, R2                                   ; 1209
                                        50             58 D0 00137          MOVL     MAP_LEFT, R0                                 ;
                                        52             50 D1 0013A          CMPL     R0, R2                                       ;
                                                       03 15 0013D          BLEQ     15$                                          ;
                                        50             52 D0 0013F          MOVL     R2, R0                                       ;
                                        59             50 D0 00142 15$:     MOVL     R0, MAP_MOVED                                ;
                     08   BE       0C   BE             50 28 00145          MOVC3    R0, @BLOCK_ADDR, @MAP_POINTER                ;
         FFB8             5A            0A             57 F1 0014B 16$:     ACBL     MAP_BLOCKS, #10, I, 12$                      ; 1199
                                        6E 00000000G   8F D0 00151 17$:     MOVL     #DCX$_NORMAL, OK                             ; 1213
                                        50             6E D0 00158 18$:     MOVL     OK, R0                                       ; 1216
                                                          04 0015B 19$:     RET                                                   ; 1217

; Routine Size:  348 bytes,    Routine Base:  $CODE$ + 0076
;  399      1218  1 GLOBAL ROUTINE dcx_it (keydesc, modrfa) =
;  400      1219  2 BEGIN
;  401      1220  2 !++
;  402      1221  2 !
;  403      1222  2 !       This routine is called for every module when a DCX map
;  404      1223  2 !       is generated a new library.  Every record of the module
;  405      1224  2 !       is read, and analyzed by DCX.
;  406      1225  2 !
;  407      1226  2 !--
;  408      1227  2 MAP
;  409      1228  2     keydesc : REF BBLOCK [dsc$c_s_bln];
;  410      1229  2
;  411      1230  2 LOCAL
;  412      1231  2    rms_status,
;  413      1232  2    header : BBLOCK [lbr$c_pagesize],
;  414      1233  2    bufdesc: BBLOCK [dsc$c_s_bln];
;  415      1234  2
;  416      1235  2 rms_perform( lbr$find ( lib_control_index, .modrfa));
;  417      1236  2
;  418      1237  2 bufdesc [dsc$a_pointer] = header;
;  419      1238  2 bufdesc [dsc$b_class] = dsc$k_class_d;
;  420      1239  3 WHILE (bufdesc [dsc$w_length] = lbr$c_pagesize;
;  421      1240  3         rms_status = lbr$get_record ( lib_control_index, bufdesc, bufdesc );
;  422      1241  4         IF NOT .rms_status AND (.rms_status NEQ rms$_eof)
;  423      1242  3         THEN
;  424      1243  4             BEGIN
;  425      1244  4             SIGNAL (.rms_status);
;  426      1245  4             EXITLOOP;
;  427      1246  3             END;
;  428      1247  3
;  429      1248  3         .rms_status NEQ rms$_eof )
;  430      1249  3
;  431      1250  2 DO
;  432      1251  2         (.dcx_analyze_data) (local_dcx_context, bufdesc);
;  433      1252  2
;  434      1253  2 RETURN true
;  435      1254  1 END;



                                        0004 00000          .ENTRY  DCX_IT, Save R2                          ; 1218
                  5E     FDF8    CE  9E 00002          MOVAB   -520(SP), SP                             ;
                           08    AC  DD 00007          PUSHL   MODRFA                                   ; 1235
                         0000'   CF  9F 0000A          PUSHAB  LIB_CONTROL_INDEX                        ;
           0000G  CF             02  FB 0000E          CALLS   #2, LBR$FIND                             ;
                  4F             50  E9 00013          BLBC    STATUS, 4$                               ;
             04   AE       08    AE  9E 00016          MOVAB   HEADER, BUFDESC+4                        ; 1237
             03   AE             02  90 0001B          MOVB    #2, BUFDESC+3                            ; 1238
                  6E     0200    8F  B0 0001F 1$:      MOVW    #512, BUFDESC                            ; 1239
                                 5E  DD 00024          PUSHL   SP                                       ; 1240
                           04    AE  9F 00026          PUSHAB  BUFDESC                                  ;
                         0000'   CF  9F 00029          PUSHAB  LIB_CONTROL_INDEX                        ;
           0000G  CF             03  FB 0002D          CALLS   #3, LBR$GET_RECORD                       ;
                  52             50  D0 00032          MOVL    R0, RMS_STATUS                           ;
                  14             52  E8 00035          BLBS    RMS_STATUS, 2$                           ; 1241
                  0001827A   8F            52  D1 00038        CMPL    RMS_STATUS, #98938                 ;
                                           0B  13 0003F        BEQL    2$                                 ;
                                           52  DD 00041        PUSHL   RMS_STATUS                         ; 1244
                  00000000G  00            01  FB 00043        CALLS   #1, LIB$SIGNAL                     ;
                                           16  11 0004A        BRB     3$                                 ; 1243
                  0001827A   8F            52  D1 0004C 2$:    CMPL    RMS_STATUS, #98938                 ; 1248
                                           0D  13 00053        BEQL    3$                                 ;
                                           5E  DD 00055        PUSHL   SP                                 ; 1251
                                   0000'   CF  9F 00057        PUSHAB  LOCAL_DCX_CONTEXT                  ;
                      0000G  DF            02  FB 0005B        CALLS   #2, @DCX_ANALYZE_DATA              ;
                                           BD  11 00060        BRB     1$                                 ;
                             50            01  D0 00062 3$:    MOVL    #1, R0                             ; 1253
                                               04 00065 4$:    RET                                        ; 1254

; Routine Size:  102 bytes,    Routine Base:  $CODE$ + 01D2
```

LB
V0

```
;   437           1255  1 %SBTTL 'LBR$LOAD_DCX';
;   438           1256  1 GLOBAL ROUTINE lbr$load_dcx =
;   439           1257  2 BEGIN
;   440           1258  2 !++
;   441           1259  2 !  FUNCTIONAL DESCRIPTION:
;   442           1260  2 !
;   443           1261  2 !       Load DCXSHR and relocate entry points by the base address.
;   444           1262  2 !
;   445           1263  2 !--
;   446           1264  2 bind
;   447           1265  2     dcx_address_table = dcx_analyze_init;
;   448           1266  2
;   449           1267  2 local
;   450           1268  2     dcxshr_desc: block [dsc$c_s_bln,byte],
;   451           1269  2     default_desc: block [dsc$c_s_bln,byte];
;   452           1270  2
;   453           1271  2 dcxshr_desc[dsc$w_length] = .dcxshr_string<0,8>;                ! set up filename  descriptor
;   454           1272  2 dcxshr_desc[dsc$a_pointer] = dcxshr_string+1;
;   455           1273  2 default_desc[dsc$w_length] = .default_string<0,8>;              !   and default filename descriptor
;   456           1274  2 default_desc[dsc$a_pointer] = default_string+1;
;   457           1275  2
;   458           1276  2 perform ( lib$adr_image(dcxshr_desc,default_desc, dcxshr_address));     ! map image and return base address
;   459           1277  2
;   460           1278  2 !
;   461           1279  2 ! Loop through the address table of dcx routines called by lbrshr and relocate them
;   462           1280  2 !       by the base address of DCXSHR
;   463           1281  2 !
;   464           1282  2 incr i to (num_dcx_routines-1) do
;   465           1283  2     dcx_address_table + (4 * .i) = .(dcx_address_table + (4 * .i)) + .dcxshr_address;
;   466           1284  2
;   467           1285  2 return true;
;   468           1286  1 end;



                                            0000 00000          .ENTRY  LBR$LOAD_DCX, Save nothing                       ; 1256
                         5E              10  C2 00002           SUBL2   #16, SP                                          ;
                    08   AE       0000'  CF  9B 00005           MOVZBW  DCXSHR_STRING, DCXSHR_DESC                       ; 1271
                    0C   AE       0000'  CF  9E 0000B           MOVAB   DCXSHR_STRING+1, DCXSHR_DESC+4                   ; 1272
                         6E       0000'  CF  9B 00011           MOVZBW  DEFAULT_STRING, DEFAULT_DESC                     ; 1273
                    04   AE       0000'  CF  9E 00016           MOVAB   DEFAULT_STRING+1, DEFAULT_DESC+4                 ; 1274
                                  0000G  CF  9F 0001C           PUSHAB  DCXSHR_ADDRESS                                   ; 1276
                                     04  AE  9F 00020           PUSHAB  DEFAULT_DESC                                     ;
                                     10  AE  9F 00023           PUSHAB  DCXSHR_DESC                                      ;
                  0000G  CF              03  FB 00026           CALLS   #3, LIB$ADR_IMAGE                                ;
                         11              50  E9 0002B           BLBC    STATUS, 2$                                       ;
                                         50  D4 0002E           CLRL    I                                                ; 1283
                  0000GCF40       0000G  CF  C0 00030 1$:       ADDL2   DCXSHR_ADDRESS, DCX_ADDRESS_TABLE[I]             ;
            F4           50              09  F3 00038           AOBLEQ  #9, I, 1$                                        ;
                         50              01  D0 0003C           MOVL    #1, R0                                           ; 1285
                                             04 0003F 2$:       RET                                                      ; 1286

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 0238
```

```
%SBTTL  'LBR$OPEN';
GLOBAL ROUTINE lbr$open (control_index, fns, create_options, dns, rlfna,
                         rns, rnslen, dcx_map_desc) =
BEGIN
!++
!
! FUNCTIONAL DESCRIPTION:
!
!       This routine opens an existing library for reading or updating,
!       or creates a new library.  This routine must be called before
!       any other library access procedures except LBR$INI_CONTROL.
!
! CALLING SEQUENCE:
!
!       status = LBR$OPEN (control_index[, fns, create_options, dns,
!                       rlfna, rns, rnslen])
!
! INPUT PARAMETERS:
!
!       control_index   is the address of a longword containing the
!                        index returned from LBR$INI_CONTROL
!       dns             is the address of a string descriptor for the
!                        default filename string.
!       fns             is the address of a string descriptor for the
!                        filename string.
!       rlfna           is the address of a NAM block for the related
!                        file.
!       rns             is the address of a string descriptor for the
!                        resultant name string.
!       create_options  is the address of an array of create options.
!                        This argument is needed only if the function
!                        is LBR$C_CREATE.
!
! OUTPUT PARAMETERS:
!
!       rnslen          is the address of a longword to return the
!                        length of the resultant name string.
!
!       The specified library is opened.  The library header is
!       read into memory (or constructed if creating the library).
!       The default index is set to index 0.
!
!       If there is an error while opening the library, the expanded
!       name string will be returned, rather than the resultant name.
!
! ROUTINE VALUE:
!
!       lbr$_illfmt     illegal format in library
!       lbr$_illfunc    illegal function
!       lbr$_illctl     illegal control table
!       lbr$_illcreopt  illegal create options
!       lbr$_libopn     library already open
!       lbr$_typmismch  library type does not match requested type
!       lib$_insvirmem  insufficient virtual memory
!       lib$_badblosiz  bad block size
!--
```

```
;  2 BUILTIN
;  2         NULLPARAMETER;                          ! True if parameter omitted
;  2 MAP
;  2         dcx_map_desc : REF VECTOR,
;  2         dns : REF BBLOCK [dsc$c_s_bln], ! Pointer to string descriptor
;  2         fns : REF BBLOCK [dsc$c_s_bln], ! Pointer to string descriptor
;  2         rlfna : REF BBLOCK,             ! Pointer to NAM block
;  2         rns : REF BBLOCK [dsc$c_s_bln], ! Pointer to string descriptor
;  2         create_options : REF BBLOCK;    ! and the create options
;  2 LOCAL
;  2         event_flag,
;  2         lbrfab : BBLOCK [fab$c_bln],    ! Allocate a FAB to open library
;  2         recrab : REF BBLOCK [rab$c_bln], ! Pointer to record I/O RAB
;  2         lbrnam : REF BBLOCK [nam$c_bln], ! Pointer to NAM block
;  2         status,
;  2         return_status,
;  2         blksiz,
;  2         retries,
;  2         one_second : VECTOR [2],
;  2         hdradr,
;  2         context : REF BBLOCK,           ! Pointer to context block
;  2         header : REF BBLOCK;            ! Pointer to header block
;  2
;  2 lbr$gl_rmsstv = 0;
;  2 status = validate_ctl (..control_index); !Validate the control block
;  2 IF NOT .status AND .status NEQ lbr$_libnotopn !If failed and not becuase library
;  2 THEN RETURN .status;                     !then its really bad, so return error
;  2
;  2 IF .lbr$gl_control [lbr$v_open]          !If library already open
;  2 THEN RETURN lbr$_libopn;                 ! then return an error
;  2
;  2 IF .lbr$gl_control [lbr$b_func] EQL lbr$c_create
;  2 THEN
;  3     BEGIN
;  3     IF NULLPARAMETER (3)                ! Options required on create
;  3     THEN RETURN lbr$_illcreopt;         ! return error if not
;  3     IF (.create_options [cre$l_keylen] GTR lbr$c_maxkeylen) OR
;  3        (.create_options [cre$l_luhmax] GTR lbr$c_maxluhrec) OR
;  3        (.create_options [cre$l_vertyp] LSS 0) OR
;  4        (.create_options [cre$l_vertyp] GTR cre$c_vmsv3)
;  3     THEN RETURN lbr$_illcreopt;         ! return error if not
;  2     END;
;  2 !       Allocate and initialize the internal context area
;  2 !
;  2 perform (get_zmem (ctx$c_length, lbr$gl_control [lbr$l_ctxptr]));
;  2 !
;  2 !       Allocate a RAB and NAM block to open the file.
;  2 !
;  2 status = get_zmem (rab$c_bln+nam$c_bln, recrab);
;  2 IF NOT .status                          ! If not enough memory,
;  2 THEN
;  3     BEGIN
;  3     lbr_deal_mem (..control_index);     ! Deallocate everything
;  3     RETURN .status;                     ! and return with error
;  2     END;
;  2 !
;  2 ! Initialize the FAB, RAB, and NAM blocks
```

```
2 !
2 context = .lbr$gl_control [lbr$l_ctxptr];        ! Point to the context block
2 context [ctx$l_recrab] = .recrab;                ! Save record I/O RAB address
2 IF (lbrnam = .lbr$gl_control [lbr$l_usrnam]) EQL 0 !If no user-supplied NAM block
3 THEN BEGIN
3     lbrnam = .recrab + rab$c_bln;       !then use ours
3     lbr$gl_control [lbr$l_usrnam] = .lbrnam;
3     lbrnam [nam$b_bln] = nam$c_bln;              !Identify the NAM block
3     lbrnam [nam$b_bid] = nam$c_bid;              !As a NAM block
2     END;
2
2 CH$FILL (0, fab$c_bln, lbrfab);          !Zero the FAB
2 lbrfab [fab$b_bln] = fab$c_bln;          !Identify it as a fab with the length
2 lbrfab [fab$b_bid] = fab$c_bid;          !And ID
2 lbrfab [fab$v_bio] = true;               !Set for block I/O
2 lbrfab [fab$v_get] = true;               !Set to allow $READs
2 IF .lbr$gl_control [lbr$b_func] EQL lbr$c_create !If creating
2 OR .lbr$gl_control [lbr$b_func] EQL lbr$c_update !or updating
2 THEN
2     lbrfab [fab$v_put] = true            !then we will do $WRITEs also
2 ELSE
2     context [ctx$v_ronly] = true;        ! otherwise flag read only file
2
2 lbrfab [fab$w_mrs] = lbr$c_pagesize;     !Set the maximum record size
2 lbrfab [fab$b_rfm] = fab$c_fix;          !Set fixed record format
2
2 IF .lbrnam [nam$w_fid_num] NEQ 0         ! Was filled-in NAM block passed
2 THEN
2     lbrfab [fab$v_nam] = true            !  yes--use it
2 ELSE
3     BEGIN
3     IF NOT NULLPARAMETER (2)             ! Otherwise, did we get an FNS?
3     THEN
4         BEGIN
4         lbrfab [fab$b_fns] = .fns [dsc$w_length]; ! Set file name string
4         IF .lbrfab [fab$b_fns] NEQ 0
4         THEN lbrfab [fab$l_fna] = .fns [dsc$a_pointer];
4         END
3     ELSE                                 !  Has there been a previous open over the net
4         BEGIN                            !  i.e.  no fid_num but nam filled in.
4         IF .lbrnam [nam$b_rsl] NEQ 0
4         THEN
5             BEGIN
5             lbrfab [fab$b_fns] = .lbrnam [nam$b_rsl];
5             lbrfab [fab$l_fna] = .lbrnam [nam$l_rsa];
5             END
4         ELSE
5             BEGIN                        !  No, that is an error
5             lbr_deal_mem (..control_index);    ! Deallocate all memory
5             RETURN lbr$_nofilnam;
4             END;
3         END;
2     END;
2
2 IF NOT NULLPARAMETER (4)                 ! Set default name string
3 THEN BEGIN
3     lbrfab [fab$b_dns] = .dns [dsc$w_length];
```

```
;  641          1458  3      IF .lbrfab [fab$b_dns] NEQ 0
;  642          1459  3      THEN lbrfab [fab$l_dna] = .dns [dsc$a_pointer];
;  643          1460  2      END;
;  644          1461  2
;  645          1462  2  IF NOT NULLPARAMETER (5)                 ! Related filename block arg present?
;  646          1463  2  THEN
;  647          1464  3          begin
;  648          1465  3          lbrnam [nam$l_rlf] = .rlfna;
;  649          1466  3          lbrfab[fab$v_nam] = true;
;  650          1467  2          end;
;  651          1468  2
;  652          1469  2  lbrfab [fab$l_nam] = .lbrnam;            !Point to the NAM block
;  653          1470  2
;  654          1471  2  IF .lbr$gl_control [lbr$b_func] EQL lbr$c_create !If creating the file
;  655          1472  3  THEN BEGIN
;  656          1473  3      lbrfab [fab$v_cbt] = true;                   !Set contiguous best try
;  657          1474  3      lbrfab [fab$v_ofp] = true;                   ! Set output file parse bit.
;  658          1475  3      lbrfab [fab$l_alq] = .create_options [cre$l_alloc]; ! Set initial allocation
;  659          1476  2      END;
;  660          1477  2
;  661          1478  2  recrab [rab$b_bln] = rab$c_bln;          !Identify the RAB
;  662          1479  2  recrab [rab$b_bid] = rab$c_bid;          !with length and ID
;  663          1480  2  recrab [rab$l_fab] = lbrfab;             !Set pointer to FAB
;  664          1481  2  recrab [rab$v_loc] = true;               !Set locate mode
;  665          1482  2  recrab [rab$v_bio] = true;               !Set for block I/O only
;  666          1483  2
;  667          1484  2  IF NOT NULLPARAMETER (6)                 !If result string arg present
;  668          1485  2  THEN IF (lbrnam [nam$b_rss] = .rns [dsc$w_length]) NEQ 0
;  669          1486  3  THEN BEGIN
;  670          1487  3      lbrnam [nam$b_ess] = .rns [dsc$w_length]; ! Copy to expanded name area
;  671          1488  3      lbrnam [nam$l_rsa] = .rns [dsc$a_pointer];! so that error messages will
;  672          1489  3      lbrnam [nam$l_esa] = .rns [dsc$a_pointer];! have the right information
;  673          1490  2      END;
;  674          1491  2  !
;  675          1492  2  !       Open the library, and connect the record stream.
;  676          1493  2  !
;  677          1494  3  IF ( status = lib$get_ef( event_flag ))          ! if we can find an unused event flag
;  678          1495  2  THEN
;  679          1496  3      status = (IF .lbr$gl_control [lbr$b_func] EQL lbr$c_create
;  680          1497  4                THEN $CREATE (FAB = lbrfab)
;  681          1498  4                ELSE BEGIN
;  682          1499  4                    retries = lbr$c_retryopen;  ! Set max. number of retries
;  683          1500  4                    one_second [0] = -(10*1000*1000*lbr$c_retrywait); ! Set one second wait
;  684          1501  4                    one_second [1] = -1;
;  685          1502  4                    WHILE (status = $OPEN (FAB = lbrfab)) EQL rms$_flk ! while file is locked
;  686          1503  4                            AND .retries GTR 0              ! and we can still retry
;  687          1504  4                    DO
;  688          1505  5                        BEGIN
;  689          1506  5                        retries = .retries - 1;                          ! count a retry
;  690          1507  5                        $SETIMR (EFN = .event_flag, DAYTIM = one_second);        ! Set timer for a second
;  691          1508  5                        $WAITFR (EFN = .event_flag );                    ! And wait for it
;  692          1509  4                        END;
;  693          1510  4                    lib$free_ef( event_flag );                  ! release our event flag
;  694          1511  4                    .status                                     ! Return the status
;  695          1512  2                    END);
;  696          1513  2  !
;  697          1514  2  !       Return result name string if an error occurred.
```

```
2 !
2 IF NOT NULLPARAMETER (7)                         !Returning length of resultant name string?
2 THEN IF (.rnslen = .lbrnam [nam$b_rsl]) EQL 0
2     THEN IF (.rnslen = .lbrnam [nam$b_esl]) EQL 0
3         THEN BEGIN
3             .rnslen = .lbrfab [fab$b_fns];
3             CH$MOVE (MIN (.rns [dsc$w_length], .lbrfab [fab$b_fns]),     !Bad error, so copy file name into r
3                      .lbrfab [fab$l_fna], .rns [dsc$a_pointer]);
2             END;
2 !
2 !       If error occurred, then give up
2 !
2 IF NOT .status                              ! If the open or create failed
3 THEN BEGIN
3     lbr$gl_rmsstv = .lbrfab [fab$l_stv];         ! Return STV on error
3     lbr_deal_mem (..control_index);        !Deallocate memory
3     RETURN .status;                        !Return the OPEN status
2     END;
2 !
2 !       Connect the record stream.
2 !
2 context [ctx$w_ifi] = .lbrfab [fab$w_ifi];       !Save IFI for close
3 IF NOT (status = $CONNECT (RAB = .recrab))       !Connect the record stream
3 THEN BEGIN                                 !and if that fails
3     lbr$gl_rmsstv = .recrab [rab$l_stv];         !then return stv
3     lbr$close (.control_index);            !then close the file (which
3                                            ! deallocates all memory)
3     RETURN .status                         ! return with error
2     END;
2 context [ctx$w_isi] = .recrab [rab$w_isi];       !Save ISI
2 lbr$gl_control [lbr$l_curidx] = 1;         !Set current index to 1
2 !
2 !       Allocate a cache hash table
2 !
2 perform (get_zmem (lbr$c_hashsize, context [ctx$l_cache]));
2 return_status = lbr$_normal;               !Set to return normal status
2 !
2 !       If create, initialize memory resident header block.
2 !
2 IF .lbr$gl_control [lbr$b_func] EQL lbr$c_create
2 THEN
3     BEGIN
3         LOCAL
3             hdrnxtrfa : REF BBLOCK;
3 !
3 !       Allocate library header block
3 !
3         status = get_zmem (lbr$c_pagesize, lbr$gl_control [lbr$l_hdrptr]);
3         IF NOT .status                             ! If error occurred,
3         THEN
4             BEGIN
4                 lbr$close (.control_index);     ! close the library
4                 RETURN .status;                 ! and return if error
3             END;
3
3     header = .lbr$gl_control [lbr$l_hdrptr];    ! Point at the header
3     hdrnxtrfa = header [lhd$b_nextrfa];        ! End of library RFA
```

```
3       header [lhd$b_type] = .create_options [cre$l_type];
3       header [lhd$b_nindex] = .create_options [cre$l_idxmax];
3       header [lhd$w_majorid] = lhd$c_majorid;                 ! Set library format level
3       header [lhd$w_minorid] = lhd$c_minorid;
3       header [lhd$b_mhdusz] = .create_options [cre$l_uhdmax];
4       header [lhd$l_sanity] = (IF .create_options [cre$l_vertyp] EQL cre$c_vmsv2
4                               THEN lhd$c_saneid
3                               ELSE lhd$c_saneid3);
3       header [lhd$w_maxluhrec] = .create_options [cre$l_luhmax]; ! set maximum number of library update histor
3   !
3   ! Preallocate index blocks and, if /COMPRESS=REDUCE, cache map blocks
3   !
4       IF NOT (status = prealloc_index (.header, .create_options))
4       THEN BEGIN
4           lbr$close (.control_index);
4           RETURN .status;
3           END;
3       hdrnxtrfa [rfa$l_vbn] = .header [lhd$l_nextvbn]; ! Set next available VBN
3       hdrnxtrfa [rfa$w_offset] = 0;                    !   and offset
3       CH$MOVE (.lbr$gt_lbrver [0]+1, lbr$gt_lbrver, ! Set librarian version
3                   header [lhd$t_lbrver]);
3       $GETTIM (TIMADR = header [lhd$l_credat]);   ! Get creation date/time
3       !
3       !   Initialize all index descriptors
3       !
3       INCR i FROM 1 TO .header [lhd$b_nindex]     ! Do all descriptors
3       DO
4           BEGIN
4           BIND
4               index_desc = header [lhd$c_idxdesc-idd$c_length,0,0,0]:
4                   BLOCKVECTOR [,idd$c_length,BYTE];
4
4           index_desc [.i, idd$w_flags] = 0; ! Preset flags to 0
4           IF .create_options [cre$l_keylen] NEQ 0 ! If ASCII keys,
4           THEN
5               BEGIN
5               index_desc [.i, idd$v_ascii] = true; ! Set to ASCII keys
5               IF (.create_options [cre$l_vertyp] EQL 0) OR
6                  (.create_options [cre$l_vertyp] GEQ cre$c_vmsv3)
5               THEN
6                   BEGIN
6                   index_desc [.i, idd$v_varlenidx] = true; ! Set to variable length ASCII keys
6                   index_desc [.i, idd$v_nocasecmp] =      ! should match keyword
6                       .create_options [cre$v_nocasecmp];  ! be upcased.
6                   index_desc [.i, idd$v_nocasentr] =      ! should index entry be upcased
6                       .create_options [cre$v_nocasentr];  ! when compared with match keyword.
6                   index_desc [.i, idd$v_upcasntry] =      ! should the index entry be
6                       .create_options [cre$v_upcasntry];  ! upcased when entered.
5                   END;
5               index_desc [.i, idd$w_keylen] = .create_options [cre$l_keylen] + 1; ! (+1 for count byte)
5               END
4           ELSE
4               index_desc [.i, idd$w_keylen] = 4;  ! Set to binary keys
4           index_desc [.i, idd$l_vbn] = 0;         ! Set no index yet
3           END;
3
3       header[lhd$l_dcxmapvbn] = 0;
```

```
3
3   IF NOT NULLPARAMETER(8)
3   THEN
4       BEGIN
4       BIND
4           dcx_rec_desc = context[ctx$l_dcxrecdsc] : BBLOCK[dsc$c_s_bln];
4       LOCAL
4           map_begin,
4           map_offset,
4           map_len,
4           map_blocks,
4           newvbn,
4           newvbnadr,
4
4           cache_entry : REF BBLOCK;
4
4       if .dcxshr_address eql 0
4       then
4           perform ( lbr$load_dcx());
4
4       map_begin = .dcx_map_desc[1];
4       map_offset = 0;
4       map_len = .dcx_map_desc[0] + 4;
4
4       map_blocks = .map_len / lbr$c_pagesize +
5                          (IF (.map_len MOD lbr$c_pagesize) GTR 0
4                          THEN 1 ELSE 0);
4
4       INCR j FROM 1 TO .map_blocks DO
5           BEGIN
5           IF .map_len GTR lbr$c_pagesize
5           THEN
5               map_len = lbr$c_pagesize;
5           perform(alloc_block(newvbn,newvbnadr));
5           add_cache(.newvbn,cache_entry);
5           cache_entry[cache$l_address] = .newvbnadr;
5           cache_entry[cache$v_data] = true;
5           cache_entry[cache$v_dirty] = true;
5           IF .header[lhd$l_dcxmapvbn] EQL 0
5           THEN
6               BEGIN
6               .newvbnadr = .dcx_map_desc[0];
6               IF .map_len + 4  GTR  lbr$c_pagesize
6               THEN
6                   map_len = .map_len - 4;
6               header[lhd$l_dcxmapvbn] = .newvbn;
6               newvbnadr = .newvbnadr + 4;
5               END;
5
5           CH$MOVE (.map_len, .map_begin + .map_offset, .newvbnadr);
5           map_offset = .map_offset + .map_len ;
5           map_len = .dcx_map_desc[0]  - .map_offset;
4           END;
4       perform((.dcx_compress_init) (context[ctx$l_dcxctx], dcx_map_desc[1]));
4       context[ctx$l_dcxmapdsc] = .dcx_map_desc;
4       dcx_rec_desc[dsc$b_dtype] = dsc$k_dtype_t;
4       dcx_rec_desc[dsc$b_class] = dsc$k_class_s;
```

```
4         perform(get_mem(lbr_dcx$c_maxrecsiz, dcx_rec_desc[dsc$a_pointer]));
3         END;
3     END
3 !
3 !       If open, read the library header from disk.
3 !
2 ELSE
3     BEGIN
3     status = read_block (1, lbr$gl_control [lbr$l_hdrptr]); ! Read block 1 of file
3     IF NOT .status                          ! If error reading block,
4     THEN BEGIN
4         lbr$close (.control_index);     ! Close the file
4         RETURN .status;                 ! and return with error
3         END;
3
3     header = .lbr$gl_control [lbr$l_hdrptr];
3
3     IF (.header [lhd$l_sanity] NEQ lhd$c_saneid) AND ! If not valid header,
3         (.header [lhd$l_sanity] NEQ lhd$c_saneid3) AND
4         (.header [lhd$l_sanity] NEQ lhd$c_saneidc)
4     THEN BEGIN
4         IF .header [ohd$b_fmtlvl] EQL ofl$c_fmtlvl ! Is it an old format library?
5         THEN BEGIN
5             header [ohd$b_type] = .header [ohd$b_type] + 1; ! Adjust
5                                                 ! library type to map
5                                                 ! into new format
5             lbr_old_lib_dat (.header);  ! Old format--extract information
5             context [ctx$v_oldlib] = true; ! Flag old format library
5             lbr$gl_control [lbr$b_func] = lbr$c_read; ! Only read access allowed
5             return_status = lbr$_oldlibrary; ! Set return status
5             END
5         ELSE BEGIN
5             lbr$close (.control_index); ! Close the file
5             RETURN lbr$_illfmt;         ! return illegal format file
4             END;
3         END;
3
3     IF .lbr$gl_control [lbr$b_type] NEQ 0 ! If user specified type,
3         AND .header [lhd$b_type] NEQ .lbr$gl_control [lbr$b_type]
4     THEN BEGIN
4         IF .return_status EQL lbr$_normal
4         THEN return_status = lbr$_typmismch     ! return type mismatch
4         ELSE IF .return_status EQL lbr$_oldlibrary
4         THEN return_status = lbr$_oldmismch;
4         lbr$gl_control [lbr$b_type] = .header [lhd$b_type];
3         END;
3     IF .header [lhd$w_closerror] THEN return_status = lbr$_errclose;
3
3     IF .header[lhd$l_dcxmapvbn] NEQ 0  AND  NULLPARAMETER(8)
3     THEN
4         BEGIN
4         BIND
4             dcx_rec_desc = context[ctx$l_dcxrecdsc] : BBLOCK [dsc$c_s_bln];
4         if .dcxshr_address eql 0
4         then
4             perform( lbr$load_dcx());
4         perform(get_mem(dsc$c_s_bln, context[ctx$l_dcxmapdsc]));
```

```
   4          perform(lbr$dcx_map(0, .context[ctx$l_dcxmapdsc]));
   4
   4          IF .lbr$gl_control[lbr$b_func] EQL lbr$c_read
   4          THEN
   4              ! If we are reading the DCX-encoded library tell DCX that we
   4              ! will be "expanding" and allocate a buffer for the eventually
   4              ! returned records:
   4              !
   5              BEGIN
P  5              perform((.dcx_expand_init)
   5                      (context[ctx$l_dcxctx], .context[ctx$l_dcxmapdsc]+4));
   5              perform(get_mem(lbr$c_maxrecsiz, dcx_rec_desc[dsc$a_pointer]));
   5              END
   5
   4          ELSE
   4              ! If we are writing into the DCX-encoded library tell DCX that we
   4              ! will be "compressing" and allocate a buffer for the "reduced"
   4              ! records:
   4              !
   5              BEGIN
P  5              perform((.dcx_compress_init)
   5                      (context[ctx$l_dcxctx], .context[ctx$l_dcxmapdsc]+4));
   5              perform(get_mem(lbr_dcx$c_maxrecsiz, dcx_rec_desc[dsc$a_pointer]));
   4              END;
   4
   4          dcx_rec_desc[dsc$b_dtype] = dsc$k_dtype_t;
   4          dcx_rec_desc[dsc$b_class] = dsc$k_class_s;
   3          END;
   2      END;
   2  IF .header[lhd$l_dcxmapvbn] NEQ 0              ! If this is a DCX data-reduced
   2  THEN                                            !  lib, assign new sanity id
   2      header[lhd$l_sanity] = lhd$c_saneidc;
   2  lbr$gl_rmsstv = .header [lhd$b_type];          !Return type of library opened
   2  !
   2  !       Mark file open successfully.
   2  !
   2  IF .lbr$gl_maxread EQL 0                        ! Max read length known?
   3  THEN BEGIN
   3      $ADJWSL (WSETLM = blksiz);                  ! Get working set limit
   3      lbr$gl_maxread = MIN (.blksiz - lbr$c_maxread, lbr$c_maxread);      ! Determine max number blocks to rea
   3      IF .lbr$gl_maxread LSS lbr$c_minread        ! but if too small
   3          THEN lbr$gl_maxread = lbr$c_minread;    !  then use the minimum
   3      IF .lbrnam [nam$v_node]                     ! If opening across network
   3          THEN lbr$gl_maxread = lbr$c_minread;    !  then reduce to minimum
   3      mem$l_memexp = .lbr$gl_maxread + lbr$c_memxtra; ! Allow extra pages on expand region
   3      mem$l_maxblk = .mem$l_memexp * lbr$c_pagesize; ! Set size of largest request
   2      END;
   2  IF .lbr$gl_control [lbr$b_func] EQL lbr$c_update !If function is update
   2  OR .lbr$gl_control [lbr$b_func] EQL lbr$c_create ! or create
   2  THEN
   3      BEGIN
   3      LOCAL
   3          oldheader : REF BBLOCK;
   3
   3      $GETTIM (TIMADR = header [lhd$l_updtim]); !Then get update time
   3      context [ctx$v_hdrdirty] = true;       !and mark header as dirty
   3      !
```

```
;   983           1800  3       !   Store unmodified header block in core with the diddle bit set.
;   984           1801  3       !   Before initating a write to the library, original header will
;   985           1802  3       !   be written out.  If the update is unsuccessful, the header will
;   986           1803  3       !   record the failure.
;   987           1804  3       !
;   988           1805  3       status = get_zmem (lbr$c_pagesize, lbr$gl_control [lbr$l_oldhdrptr]);
;   989           1806  3       IF NOT .status                                  ! If error occurred,
;   990           1807  3       THEN
;   991           1808  4           BEGIN
;   992           1809  4           lbr$close (.control_index);     ! close the library
;   993           1810  4           RETURN .status;                 ! and return if error
;   994           1811  3           END;
;   995           1812  3       CH$MOVE ( lbr$c_pagesize, .lbr$gl_control [lbr$l_hdrptr], .lbr$gl_control [lbr$l_oldhdrptr] );
;   996           1813  3       oldheader = .lbr$gl_control [lbr$l_oldhdrptr];
;   997           1814  3       oldheader [lhd$w_closerror] = lhd$c_corrupted;
;   998           1815  2       END;
;   999           1816  2 lbr$gl_control [lbr$v_open] = true;      ! In control block also
;  1000           1817  2 context [ctx$v_libopn] = true;           ! Flag library open
;  1001           1818  2
;  1002           1819  2 RETURN .return_status;                    ! Return with status
;  1003           1820  1 END;



                                                                   .EXTRN  SYS$CREATE, SYS$OPEN
                                                                   .EXTRN  SYS$SETIMR, SYS$WAITFR
                                                                   .EXTRN  SYS$CONNECT, SYS$GETTIM
                                                                   .EXTRN  SYS$ADJWSL

                                            0FFC 00000             .ENTRY  LBR$OPEN, Save R2,R3 R4,R5,R6,R7,R8,R9,R10,-; 1288
                                                                           R11                                         ;
                              5E      FF70  CE  9E 00002           MOVAB   -144(SP), SP                                ;
                                     0000G  CF  D4 00007           CLRL    LBR$GL_RMSSTV                               ; 1367
                              50       04   BC  D0 0000B           MOVL    @CONTROL_INDEX, R0                          ; 1368
                                         0000G  30 0000F           BSBW    VALIDATE_CTL                                ;
                              6E            50  D0 00012           MOVL    R0, STATUS                                  ;
                              0C            6E  E8 00015           BLBS    STATUS, 1$                                  ; 1369
                   00000000G  8F            6E  D1 00018           CMPL    STATUS, #LBR$_LIBNOTOPN                     ;
                                            03  13 0001F           BEQL    1$                                          ;
                                          066F  31 00021           BRW     77$                                         ;
                              51     0000G  CF  D0 00024 1$:       MOVL    LBR$GL_CONTROL, R1                          ; 1372
              08          06  A1            01  E1 00029           BBC     #1, 6(R1), 2$                               ;
                              50 00000000G  8F  D0 0002E           MOVL    #LBR$_LIBOPN, R0                            ; 1373
                                                04 00035           RET                                                 ;
                                       03   A1  95 00036 2$:       TSTB    3(R1)                                       ; 1375
                                            35  12 00039           BNEQ    4$                                          ;
                              03            6C  91 0003B           CMPB    (AP), #3                                    ; 1378
                                            28  1F 0003E           BLSSU   3$                                          ;
                                       0C   AC  D5 00040           TSTL    12(AP)                                      ;
                                            23  13 00043           BEQL    3$                                          ;
                              50       0C   AC  D0 00045           MOVL    CREATE_OPTIONS, R0                          ; 1380
                   00000080   8F       04   A0  D1 00049           CMPL    4(R0), #128                                 ;
                                            15  14 00051           BGTR    3$                                          ;
                   00008000   8F       18   A0  D1 00053           CMPL    24(R0), #32768                              ; 1381
                                            0B  14 0005B           BGTR    3$                                          ;
                                       1C   A0  D5 0005D           TSTL    28(R0)                                      ; 1382
                                            06  19 00060           BLSS    3$                                          ;
```

```
                                03      1C  A0  D1 00062          CMPL    28(R0), #3                              ; 1383
                                            08  15 00066          BLEQ    4$                                      ;
                                50 00000000G  8F  D0 00068 3$:    MOVL    #LBR$_ILLCREOPT, R0                     ; 1384
                                                04 0006F          RET                                             ;
                                51          0E  C0 00070 4$:      ADDL2   #14, R1                                 ; 1388
                                50      86  8F  9A 00073          MOVZBL  #134, R0                                ;
                                        0000G   30 00077          BSBW    GET_ZMEM                                ;
                                01          50  E8 0007A          BLBS    STATUS, 5$                              ;
                                                04 0007D          RET                                             ;
                                51      20  AE  9E 0007E 5$:      MOVAB   RECRAB, R1                              ; 1392
                                50      A4  8F  9A 00082          MOVZBL  #164, R0                                ;
                                        0000G   30 00086          BSBW    GET_ZMEM                                ;
                                6E          50  D0 00089          MOVL    R0, STATUS                              ;
                                03          6E  E8 0008C          BLBS    STATUS, 6$                              ; 1393
                                        01D7    31 0008F          BRW     25$                                     ;
                                56      0000G  CF  D0 00092 6$:   MOVL    LBR$GL_CONTROL, R6                      ; 1402
                                57          0E  A6  D0 00097      MOVL    14(R6), CONTEXT                         ;
                                58          20  AE  D0 0009B      MOVL    RECRAB, R8                              ; 1403
                           0C   A7              58  D0 0009F      MOVL    R8, 12(CONTEXT)                         ;
                                59          16  A6  D0 000A3      MOVL    22(R6), LBRNAM                          ; 1404
                                                0D  12 000A7      BNEQ    7$                                      ;
                                59          44  A8  9E 000A9      MOVAB   68(R8), LBRNAM                          ; 1406
                           16   A6              59  D0 000AD      MOVL    LBRNAM, 22(R6)                          ; 1407
                                69        6002  8F  B0 000B1      MOVW    #24578, (LBRNAM)                        ; 1409
      0050   8F           00    6E              00  2C 000B6 7$:  MOVC5   #0, (SP), #0, #80, LBRFAB               ; 1412
                                            40  AE     000BD                                                      ;
                           40   AE        5003  8F  B0 000BF      MOVW    #20483, LBRFAB                          ; 1414
                           56   AE              22  88 000C5      BISB2   #34, LBRFAB+22                          ; 1416
                                            03  A6  95 000C9      TSTB    3(R6)                                   ; 1417
                                                06  13 000CC      BEQL    8$                                      ;
                                02          03  A6  91 000CE      CMPB    3(R6), #2                               ; 1418
                                                06  12 000D2      BNEQ    9$                                      ;
                           56   AE              01  88 000D4 8$:  BISB2   #1, LBRFAB+22                           ; 1420
                                                05  11 000D8      BRB     10$                                     ;
                           04   A7          80  8F  88 000DA 9$:  BISB2   #128, 4(CONTEXT)                        ; 1422
                           76   AE        0200  8F  B0 000DF 10$: MOVW    #512, LBRFAB+54                         ; 1424
                           5F   AE              01  90 000E5      MOVB    #1, LBRFAB+31                           ; 1425
                                            24  A9  B5 000E9      TSTW    36(LBRNAM)                              ; 1427
                                                06  13 000EC      BEQL    11$                                     ;
                           47   AE              01  88 000EE      BISB2   #1, LBRFAB+7                            ; 1429
                                                3C  11 000F2      BRB     14$                                     ;
                                02              6C  91 000F4 11$: CMPB    (AP), #2                                ; 1432
                                                16  1F 000F7      BLSSU   12$                                     ;
                                            08  AC  D5 000F9      TSTL    8(AP)                                   ;
                                                11  13 000FC      BEQL    12$                                     ;
                                50          08  AC  D0 000FE      MOVL    FNS, R0                                 ; 1435
                           74   AE              60  90 00102      MOVB    (R0), LBRFAB+52                         ;
                                                28  13 00106      BEQL    14$                                     ; 1436
                           6C   AE          04  A0  D0 00108      MOVL    4(R0), LBRFAB+44                        ; 1437
                                                21  11 0010D      BRB     14$                                     ; 1432
                                            03  A9  95 0010F 12$: TSTB    3(LBRNAM)                               ; 1441
                                                0C  13 00112      BEQL    13$                                     ;
                           74   AE          03  A9  90 00114      MOVB    3(LBRNAM), LBRFAB+52                    ; 1444
                           6C   AE          04  A9  D0 00119      MOVL    4(LBRNAM), LBRFAB+44                    ; 1445
                                                10  11 0011E      BRB     14$                                     ; 1441
                                            04  BC  DD 00120 13$: PUSHL   @CONTROL_INDEX                          ; 1449
                        0000V   CF              01  FB 00123      CALLS   #1, LBR_DEAL_MEM                        ;
```

```
                50 00000000G  8F  D0 00128        MOVL    #LBR$_NOFILNAM, R0                        ; 1450
                                  04 0012F        RET                                               ;
                04            6C  91 00130 14$:   CMPB    (AP), #4                                  ; 1455
                              14  1F 00133        BLSSU   15$                                       ;
                         10   AC  D5 00135        TSTL    16(AP)                                    ;
                              0F  13 00138        BEQL    15$                                       ;
           50            10   AC  D0 0013A        MOVL    DNS, R0                                   ; 1457
      75   AE                 60  90 0013E        MOVB    (R0), LBRFAB+53                           ;
                              05  13 00142        BEQL    15$                                       ; 1458
      70   AE            04   A0  D0 00144        MOVL    4(R0), LBRFAB+48                          ; 1459
           05                 6C  91 00149 15$:   CMPB    (AP), #5                                  ; 1462
                              0E  1F 0014C        BLSSU   16$                                       ;
                         14   AC  D5 0014E        TSTL    20(AP)                                    ;
                              09  13 00151        BEQL    16$                                       ;
      10   A9            14   AC  D0 00153        MOVL    RLFNA, 16(LBRNAM)                         ; 1465
      47   AE                 01  88 00158        BISB2   #1, LBRFAB+7                              ; 1466
      68   AE                 59  D0 0015C 16$:   MOVL    LBRNAM, LBRFAB+40                         ; 1469
           50          0000G  CF  D0 00160        MOVL    LBR$GL_CONTROL, R0                        ; 1471
                         03   A0  95 00165        TSTB    3(R0)                                     ;
                              0F  12 00168        BNEQ    17$                                       ;
      46   AE          2020   8F  A8 0016A        BISW2   #8224, LBRFAB+7                           ; 1474
           50            0C   AC  D0 00170        MOVL    CREATE_OPTIONS, R0                        ; 1475
      50   AE            08   A0  D0 00174        MOVL    8(R0), LBRFAB+16                          ;
           68          4401   8F  B0 00179 17$:   MOVW    #17409, (R8)                              ; 1479
      3C   A8            40   AE  9E 0017E        MOVAB   LBRFAB, 60(R8)                            ; 1480
      05   A8          0108   8F  A8 00183        BISW2   #264, 5(R8)                               ; 1481
           06                 6C  91 00189        CMPB    (AP), #6                                  ; 1484
                              22  1F 0018C        BLSSU   18$                                       ;
                         18   AC  D5 0018E        TSTL    24(AP)                                    ;
                              1D  13 00191        BEQL    18$                                       ;
           50            18   AC  D0 00193        MOVL    RNS, R0                                   ; 1485
           51                 60  3C 00197        MOVZWL  (R0), R1                                  ;
      02   A9                 51  90 0019A        MOVB    R1, 2(LBRNAM)                             ;
                              51  D5 0019E        TSTL    R1                                        ;
                              0E  13 001A0        BEQL    18$                                       ;
      0A   A9                 60  90 001A2        MOVB    (R0), 10(LBRNAM)                          ; 1487
      04   A9            04   A0  D0 001A6        MOVL    4(R0), 4(LBRNAM)                          ; 1488
      0C   A9            04   A0  D0 001AB        MOVL    4(R0), 12(LBRNAM)                         ; 1489
                         24   AE  9F 001B0 18$:   PUSHAB  EVENT_FLAG                                ; 1494
00000000G  00                 01  FB 001B3        CALLS   #1, LIB$GET_EF                            ;
           6E                 50  D0 001BA        MOVL    R0, STATUS                                ;
           69                 6E  E9 001BD        BLBC    STATUS, 22$                               ;
           50          0000G  CF  D0 001C0        MOVL    LBR$GL_CONTROL, R0                        ; 1496
                         03   A0  95 001C5        TSTB    3(R0)                                     ;
                              0F  12 001C8        BNEQ    19$                                       ;
                         40   AE  9F 001CA        PUSHAB  LBRFAB                                    ; 1497
00000000G  00                 01  FB 001CD        CALLS   #1, SYS$CREATE                            ;
           6E                 50  D0 001D4        MOVL    R0, STATUS                                ;
                              50  11 001D7        BRB     22$                                       ;
           52                 1E  D0 001D9 19$:   MOVL    #30, RETRIES                              ; 1499
      38   AE FF676980        8F  D0 001DC        MOVL    #-10000000, ONE_SECOND                    ; 1500
      3C   AE                 01  CE 001E4        MNEGL   #1, ONE_SECOND+4                          ; 1501
                         40   AE  9F 001E8 20$:   PUSHAB  LBRFAB                                    ; 1502
00000000G  00                 01  FB 001EB        CALLS   #1, SYS$OPEN                              ;
           6E                 50  D0 001F2        MOVL    R0, STATUS                                ;
0001828A   8F                 6E  D1 001F5        CMPL    STATUS, #98954                            ;
                              21  12 001FC        BNEQ    21$                                       ;
```

```
                                          52  D5 001FE          TSTL    RETRIES                                  ; 1503
                                          1D  15 00200          BLEQ    21$                                      ;
                                          52  D7 00202          DECL    RETRIES                                  ; 1506
                                          7E  7C 00204          CLRQ    -(SP)                                    ; 1507
                                    40    AE  9F 00206          PUSHAB  ONE_SECOND                               ;
                                    30    AE  DD 00209          PUSHL   EVENT_FLAG                               ;
                   00000000G  00          04  FB 0020C          CALLS   #4, SYS$SETIMR                           ;
                                    24    AE  DD 00213          PUSHL   EVENT_FLAG                               ; 1508
                   00000000G  00          01  FB 00216          CALLS   #1, SYS$WAITFR                           ;
                                          C9  11 0021D          BRB     20$                                      ; 1502
                                    24    AE  9F 0021F 21$:     PUSHAB  EVENT_FLAG                               ; 1510
                   00000000G  00          01  FB 00222          CALLS   #1, LIB$FREE_EF                          ;
                              07          6C  91 00229 22$:     CMPB    (AP), #7                                 ; 1516
                                          32  1F 0022C          BLSSU   24$                                      ;
                                    1C    AC  D5 0022E          TSTL    28(AP)                                   ;
                                          2D  13 00231          BEQL    24$                                      ;
                              50    1C    AC  D0 00233          MOVL    RNSLEN, R0                               ; 1517
                              60    03    A9  9A 00237          MOVZBL  3(LBRNAM), (R0)                          ;
                                          23  12 0023B          BNEQ    24$                                      ;
                              60    0B    A9  9A 0023D          MOVZBL  11(LBRNAM), (R0)                         ; 1518
                                          1D  12 00241          BNEQ    24$                                      ;
                              60    74    AE  9A 00243          MOVZBL  LBRFAB+52, (R0)                          ; 1520
                              50    18    AC  D0 00247          MOVL    RNS, R0                                  ; 1521
                              51          60  3C 0024B          MOVZWL  (R0), R1                                 ;
     51        74   AE        08          00  ED 0024E          CMPZV   #0, #8, LBRFAB+52, R1                    ;
                              04          18  0024E          BGEQ    23$                                      ;
                              51    74    AE  9A 00256          MOVZBL  LBRFAB+52, R1                            ;
               04   B0      6C  BE        51  28 0025A 23$:     MOVC3   R1, @LBRFAB+44, @4(R0)                   ; 1522
                              11          6E  E8 00260 24$:     BLBS    STATUS, 26$                              ; 1527
                      0000G   CF    4C    AE  D0 00263          MOVL    LBRFAB+12, LBR$GL_RMSSTV                 ; 1529
                                    04    BC  DD 00269 25$:     PUSHL   @CONTROL_INDEX                           ; 1530
                      0000V   CF          01  FB 0026C          CALLS   #1, LBR_DEAL_MEM                         ;
                                        041F  31 00271          BRW     77$                                      ; 1531
                         02   A7    42    AE  B0 00274 26$:     MOVW    LBRFAB+2, 2(CONTEXT)                     ; 1536
                                          58  DD 00279          PUSHL   R8                                       ; 1537
                   00000000G  00          01  FB 0027B          CALLS   #1, SYS$CONNECT                          ;
                              6E          50  D0 00282          MOVL    R0, STATUS                               ;
                              09          6E  E8 00285          BLBS    STATUS, 28$                              ;
                      0000G   CF    0C    A8  D0 00288          MOVL    12(R8), LBR$GL_RMSSTV                    ; 1539
                                        03FA  31 0028E 27$:     BRW     76$                                      ; 1540
                              67    02    A8  B0 00291 28$:     MOVW    2(R8), (CONTEXT)                         ; 1544
                              50  0000G   CF  D0 00295          MOVL    LBR$GL_CONTROL, R0                       ; 1545
                         12   A0          01  D0 0029A          MOVL    #1, 18(R0)                               ;
                              51    08    A7  9E 0029E          MOVAB   8(CONTEXT), R1                           ; 1549
                              50  0200    8F  3C 002A2          MOVZWL  #512, R0                                 ;
                                        0000G 30 002A7          BSBW    GET_ZMEM                                 ;
                              01          50  E8 002AA          BLBS    STATUS, 29$                              ;
                                              04 002AD          RET                                              ;
                         10   AE 00000000G 8F D0 002AE 29$:     MOVL    #LBR$_NORMAL, RETURN_STATUS              ; 1550
                51    0000G   CF          0A  C1 002B6          ADDL3   #10, LBR$GL_CONTROL, R1                  ; 1562
                              50  0000G   CF  D0 002BC          MOVL    LBR$GL_CONTROL, R0                       ; 1554
                                    03    A0  95 002C1          TSTB    3(R0)                                    ;
                                          03  13 002C4          BEQL    30$                                      ;
                                        01EA  31 002C6          BRW     54$                                      ;
                              50  0200    8F  3C 002C9 30$:     MOVZWL  #512, R0                                 ; 1562
                                        0000G 30 002CE          BSBW    GET_ZMEM                                 ;
                              6E          50  D0 002D1          MOVL    R0, STATUS                               ;
```

```
                               B7              6E  E9 002D4          BLBC    STATUS, 27$                             ; 1563
                               50      0000G   CF  D0 002D7          MOVL    LBR$GL_CONTROL, R0                      ; 1570
                               56         0A   A0  D0 002DC          MOVL    10(R0), HEADER                          ;
                               52         4C   A6  9E 002E0          MOVAB   76(R6), HDRNXTRFA                       ; 1571
                               58         0C   AC  D0 002E4          MOVL    CREATE_OPTIONS, R8                      ; 1572
                               66              68  90 002E8          MOVB    (R8), (HEADER)                          ;
                        01     A6         0C   A8  90 002EB          MOVB    12(R8), 1(HEADER)                       ; 1573
                        08     A6              03  D0 002F0          MOVL    #3, 8(HEADER)                           ; 1574
                        3C     A6         10   A8  90 002F4          MOVB    16(R8), 60(HEADER)                      ; 1576
                        18     AE         04   A6  9E 002F9          MOVAB   4(R6), 24(SP)                           ; 1577
                               02         1C   A8  D1 002FE          CMPL    28(R8), #2                              ;
                                               09  12 00302          BNEQ    31$                                     ;
                               50 075BC371     8F  D0 00304          MOVL    #123454321, R0                          ;
                                               07  11 0030B          BRB     32$                                     ;
                               50 0DEC2581     8F  D0 0030D 31$:     MOVL    #233579905, R0                          ;
                        18     BE              50  D0 00314 32$:     MOVL    R0, @24(SP)                             ;
                        7C     A6         18   A8  B0 00318          MOVW    24(R8), 124(HEADER)                     ; 1580
                                      0140     8F  BB 0031D          PUSHR   #^M<R6,R8>                              ; 1584
                     0000V     CF              02  FB 00321          CALLS   #2, PREALLOC_INDEX                      ;
                               6E              50  D0 00326          MOVL    R0, STATUS                              ;
                               03              6E  E8 00329          BLBS    STATUS, 33$                             ;
                                             035C  31 0032C          BRW     76$                                     ;
                               62         52   A6  D0 0032F 33$:     MOVL    82(HEADER), (HDRNXTRFA)                 ; 1589
                                          04   A2  B4 00333          CLRW    4(HDRNXTRFA)                            ; 1590
                               50      0000G   CF  9A 00336          MOVZBL  LBR$GT_LBRVER, R0                       ; 1591
                                               50  D6 0033B          INCL    R0                                      ;
            0C     A6      0000G   CF              50  28 0033D          MOVC3   R0, LBR$GT_LBRVER, 12(HEADER)           ; 1592
                                          2C   A6  9F 00344          PUSHAB  44(HEADER)                              ; 1593
                  00000000G    00              01  FB 00347          CALLS   #1, SYS$GETTIM                          ;
                               54         01   A6  9A 0034E          MOVZBL  1(HEADER), R4                           ; 1597
                               52       00BC   C6  9E 00352          MOVAB   188(R6), R2                             ; 1601
                                               50  D4 00357          CLRL    I                                       ; 1605
                                               55  11 00359          BRB     39$                                     ;
                               51            6240  7E 0035B 34$:     MOVAQ   (R2)[I], R1                             ; 1604
                                               61  B4 0035F          CLRW    (R1)                                    ;
                   53          50              03  78 00361          ASHL    #3, I, R3                               ; 1621
                               53              02  C0 00365          ADDL2   #2, R3                                  ;
                                          04   A8  D5 00368          TSTL    4(R8)                                   ; 1605
                                               37  13 0036B          BEQL    37$                                     ;
                               61              01  88 0036D          BISB2   #1, (R1)                                ; 1608
                                          1C   A8  D5 00370          TSTL    28(R8)                                  ; 1609
                                               06  13 00373          BEQL    35$                                     ;
                               03         1C   A8  D1 00375          CMPL    28(R8), #3                              ; 1610
                                               1F  19 00379          BLSS    36$                                     ;
                               61              04  88 0037B 35$:     BISB2   #4, (R1)                                ; 1613
      61           01          03         20   A8  F0 0037E          INSV    32(R8), #3, #1, (R1)                    ; 1615
      55    20     A8          01              01  EF 00384          EXTZV   #1, #1, 32(R8), R5                      ; 1617
      61           01          04              55  F0 0038A          INSV    R5, #4, #1, (R1)                        ;
      55    20     A8          01              02  EF 0038F          EXTZV   #2, #1, 32(R8), R5                      ; 1619
      61           01          05              55  F0 00395          INSV    R5, #5, #1, (R1)                        ;
                                             6342  9F 0039A 36$:     PUSHAB  (R3)[R2]                                ; 1621
                   9E          04  A8          01  A1 0039D          ADDW3   #1, 4(R8), @(SP)+                       ;
                                               06  11 003A2          BRB     38$                                     ; 1605
                                             6342  9F 003A4 37$:     PUSHAB  (R3)[R2]                                ; 1624
                               9E              04  B0 003A7          MOVW    #4, @(SP)+                              ;
                                       04    A240  7F 003AA 38$:     PUSHAQ  4(R2)[I]                                ; 1625
                                               9E  D4 003AE          CLRL    @(SP)+                                  ;
```

```
                          A7            50            54 F3 003B0 39$:   AOBLEQ  R4, I, 34$                                   ; 1597
                                   08   AE     008C   C6 9E 003B4        MOVAB   140(R6), 8(SP)                               ; 1628
                                                 08   BE D4 003BA        CLRL    @8(SP)                                       ;
                                        08            6C 91 003BD        CMPB    (AP), #8                                     ; 1630
                                                      03 1E 003C0        BGEQU   41$                                          ;
                                                    0238 31 003C2 40$:   BRW     69$                                          ;
                                                 20   AC D5 003C5 41$:   TSTL    32(AP)                                       ;
                                                      F8 13 003C8        BEQL    40$                                          ;
                                        5B       5A   A7 9E 003CA        MOVAB   90(CONTEXT), R11                             ; 1634
                                              0000G   CF D5 003CE        TSTL    DCX$SHR_ADDRESS                              ; 1645
                                                      08 12 003D2        BNEQ    42$                                          ;
                                 FBE7   CF            00 FB 003D4        CALLS   #0, LBR$LOAD_DCX                             ; 1647
                                        54            50 E9 003D9        BLBC    STATUS, 47$                                  ;
                                        5A       20   AC D0 003DC 42$:   MOVL    DCX_MAP_DESC, R10                            ; 1649
                                   04   AE       04   AA D0 003E0        MOVL    4(R10), MAP_BEGIN                            ;
                                                 0C   AE D4 003E5        CLRL    MAP_OFFSET                                   ; 1650
                          58            6A            04 C1 003E8        ADDL3   #4, (R10), MAP_LEN                           ; 1651
                          50            58 00000200   8F C7 003EC        DIVL3   #512, MAP_LEN, R0                            ; 1653
            7E            00            58            01 7A 003F4        EMUL    #1, MAP_LEN, #0, -(SP)                       ; 1654
            51            51            8E 00000200   8F 7B 003F9        EDIV    #512, (SP)+, R1, R1                          ;
                                                      51 D5 00402        TSTL    R1                                           ;
                                                      05 15 00404        BLEQ    43$                                          ;
                                        51            01 D0 00406        MOVL    #1, R1                                       ;
                                                      02 11 00409        BRB     44$                                          ;
                                                      51 D4 0040B 43$:   CLRL    R1                                           ;
                    1C    AE            50            51 C1 0040D 44$:   ADDL3   R1, R0, MAP_BLOCKS                           ;
                                                 14   AE D4 00412        CLRL    J                                            ; 1678
                                                      6A 11 00415        BRB     50$                                          ;
                             00000200   8F            58 D1 00417 45$:   CMPL    MAP_LEN, #512                                ; 1659
                                                      05 15 0041E        BLEQ    46$                                          ;
                                        58     0200   8F 3C 00420        MOVZWL  #512, MAP_LEN                                ; 1661
                                        51       28   AE 9E 00425 46$:   MOVAB   NEWVBNADR, R1                                ; 1662
                                        50       2C   AE 9E 00429        MOVAB   NEWVBN, R0                                   ;
                                                   0000G 30 0042D        BSBW    ALLOC_BLOCK                                  ;
                                        5F            50 E9 00430 47$:   BLBC    STATUS, 51$                                  ;
                                        51       30   AE 9E 00433        MOVAB   CACHE_ENTRY, R1                              ; 1663
                                        50       2C   AE D0 00437        MOVL    NEWVBN, R0                                   ;
                                                   0000G 30 0043B        BSBW    ADD_CACHE                                    ;
                                        50       30   AE D0 0043E        MOVL    CACHE_ENTRY, R0                              ; 1664
                                   08   A0       28   AE D0 00442        MOVL    NEWVBNADR, 8(R0)                             ;
                                   0C   A0            03 88 00447        BISB2   #3, 12(R0)                                   ; 1666
                                                 08   BE D5 0044B        TSTL    @8(SP)                                       ; 1667
                                                      1D 12 0044E        BNEQ    49$                                          ;
                                   28   BE            6A D0 00450        MOVL    (R10), @NEWVBNADR                            ; 1670
                                        50       04   A8 9E 00454        MOVAB   4(R8), R0                                    ; 1671
                             00000200   8F            50 D1 00458        CMPL    R0, #512                                     ;
                                                      03 15 0045F        BLEQ    48$                                          ;
                                        58            04 C2 00461        SUBL2   #4, MAP_LEN                                  ; 1673
                                   08   BE       2C   AE D0 00464 48$:   MOVL    NEWVBN, @8(SP)                               ; 1674
                                   28   AE            04 C0 00469        ADDL2   #4, NEWVBNADR                                ; 1675
                          7E       0C   AE       04   AE C1 0046D 49$:   ADDL3   MAP_BEGIN, MAP_OFFSET, -(SP)                 ; 1678
                    28    BE            9E            58 28 00473        MOVC3   MAP_LEN, @(SP)+, @NEWVBNADR                  ;
                                   0C   AE            58 C0 00478        ADDL2   MAP_LEN, MAP_OFFSET                          ; 1679
                          58            6A       0C   AE C3 0047C        SUBL3   MAP_OFFSET, (R10), MAP_LEN                   ; 1680
                          90       14   AE       1C   AE F3 00481 50$:   AOBLEQ  MAP_BLOCKS, J, 45$                           ; 1657
                                                 04   AA 9F 00487        PUSHAB  4(R10)                                       ; 1682
                                                 52   A7 9F 0048A        PUSHAB  82(CONTEXT)                                  ;
```

```
     0000G   DF              02  FB 0048D         CALLS   #2, @DCX_COMPRESS_INIT                  ;
             01              50  E8 00492  51$:   BLBS    STATUS, 52$                             ;
                                 04 00495         RET                                             ;
        56   A7              5A  D0 00496  52$:   MOVL    R10, 86(CONTEXT)                        ; 1683
        02   AB      010E    8F  B0 0049A         MOVW    #270, 2(R11)                            ; 1684
             51        04    AB  9E 004A0         MOVAB   4(R11), R1                              ; 1686
             50      1000    8F  3C 004A4         MOVZWL  #4096, R0                               ;
                          0000G  30 004A9         BSBW    GET_MEM                                 ;
             03              50  E9 004AC         BLBC    STATUS, 53$                             ;
                           014B  31 004AF         BRW     69$                                     ;
                                 04 004B2  53$:   RET                                             ;
             50              01  D0 004B3  54$:   MOVL    #1, R0                                  ; 1694
                          0000G  30 004B6         BSBW    READ_BLOCK                              ;
             6E              50  D0 004B9         MOVL    R0, STATUS                              ;
             03              6E  E8 004BC         BLBS    STATUS, 55$                             ; 1695
                           01C9  31 004BF         BRW     76$                                     ;
             50     0000G    CF  D0 004C2  55$:   MOVL    LBR$GL_CONT[illegible], R0              ; 1701
             56        0A    A0  D0 004C7         MOVL    10(R0), H[illegible]                    ;
        18   AE        04    A6  9E 004CB         MOVAB   4(R6), 24([illegible]P)                 ; 1703
  075BC371   8F        18    BE  D1 004D0         CMPL    @24(SP), #123454321                     ;
                             4B  13 004D8         BEQL    57$                                     ;
  0DEC2581   8F        18    BE  D1 004DA         CMPL    @24(SP), #233579905                     ; 1704
                             41  13 004E2         BEQL    57$                                     ;
  13071956   8F        18    BE  D1 004E4         CMPL    @24(SP), #319232342                     ; 1705
                             37  13 004EC         BEQL    57$                                     ;
        81   8F        01    A6  91 004EE         CMPB    1(HEADER), #129                         ; 1707
                             20  12 004F3         BNEQ    56$                                     ;
                             66  96 004F5         INCB    (HEADER)                                ; 1709
                             56  DD 004F7         PUSHL   HEADER                                  ; 1712
     0000G   CF              01  FB 004F9         CALLS   #1, LBR_OLD_LIB_DAT                     ;
        04   A7              20  88 004FE         BISB2   #32, 4(CONTEXT)                         ; 1713
             50     0000G    CF  D0 00502         MOVL    LBR$GL_CONTROL, R0                      ; 1714
        03   A0              01  90 00507         MOVB    #1, 3(R0)                               ;
        10   AE 00000000G    8F  D0 0050B         MOVL    #LBR$_OLDLIBRARY, RETURN_STATUS         ; 1715
                             10  11 00513         BRB     57$                                     ; 1707
                       04    AC  DD 00515  56$:   PUSHL   CONTROL_INDEX                           ; 1718
     0000V   CF              01  FB 00518         CALLS   #1, LBR$CLOSE                           ;
             50 00000000G    8F  D0 0051D         MOVL    #LBR$_ILLFMT, R0                        ; 1719
                                 04 00524         RET                                             ;
             50     0000G    CF  D0 00525  57$:   MOVL    LBR$GL_CONTROL, R0                      ; 1723
                       02    A0  95 0052A         TSTB    2(R0)                                   ;
                             30  13 0052D         BEQL    60$                                     ;
        02   A0              66  91 0052F         CMPB    (HEADER), 2(R0)                         ; 1724
                             2A  13 00533         BEQL    60$                                     ;
 00000000G   8F        10    AE  D1 00535         CMPL    RETURN_STATUS, #LBR$_NORMAL             ; 1726
                             0A  12 0053D         BNEQ    58$                                     ;
        10   AE 00000000G    8F  D0 0053F         MOVL    #LBR$_TYPMISMCH, RETURN_STATUS          ; 1727
                             12  11 00547         BRB     59$                                     ;
 00000000G   8F        10    AE  D1 00549  58$:   CMPL    RETURN_STATUS, #LBR$_OLDLIBRARY         ; 1728
                             08  12 00551         BNEQ    59$                                     ;
        10   AE 00000000G    8F  D0 00553         MOVL    #LBR$_OLDMISMCH, RETURN_STATUS          ; 1729
        02   A0              66  90 0055B  59$:   MOVB    (HEADER), 2(R0)                         ; 1730
        08   40              A6  E9 0055F  60$:   BLBC    64(HEADER), 61$                         ; 1732
        10   AE 00000000G    8F  D0 00563         MOVL    #LBR$_ERRCLOSE, RETURN_STATUS           ;
        08   AE      008C    C6  9E 0056B  61$:   MOVAB   140(R6), 8(SP)                          ; 1734
                       08    BE  D5 00571         TSTL    @8(SP)                                  ;
                             03  12 00574         BNEQ    62$                                     ;
```

```
                                                          L 10
LBR_OPENCLOSE                                           16-Sep-1984 02:01:23    VAX-11 Bliss-32 V4.0-742          Page  31
V04-000          LBR$OPEN                               14-Sep-1984 12:37:45    [LBR.SRC]OPENCLOSE.B32;1                (7)

                                  0084  31 00576         BRW     69$                                  ;
            08                      6C  91 00579 62$:    CMPB    (AP), #8                             ;
                                    05  1F 0057C         BLSSU   63$                                  ;
                          20        AC  D5 0057E         TSTL    32(AP)                               ;
                                    7A  12 00581         BNEQ    69$                                  ;
            52            5A        A7  9E 00583 63$:    MOVAB   90(CONTEXT), R2                      ; 1738
                       0000G        CF  D5 00587         TSTL    DCXSHR_ADDRESS                       ; 1739
                                    08  12 0058B         BNEQ    64$                                  ;
       FA2E CF                      00  FB 0058D         CALLS   #0, LBR$LOAD_DCX                     ; 1741
            5E                      50  E9 00592         BLBC    STATUS, 67$                          ;
            51            56        A7  9E 00595 64$:    MOVAB   86(CONTEXT), R1                      ; 1742
            50                      08  D0 00599         MOVL    #8, R0                               ;
                                 0000G  30 0059C         BSBW    GET_MEM                              ;
            51                      50  E9 0059F         BLBC    STATUS, 67$                          ;
                          56        A7  DD 005A2         PUSHL   86(CONTEXT)                          ; 1743
                                    7E  D4 005A5         CLRL    -(SP)                                ;
       F852 CF                      02  FB 005A7         CALLS   #2, LBR$DCX_MAP                      ;
            44                      50  E9 005AC         BLBC    STATUS, 67$                          ;
            51            52        A7  9E 005AF         MOVAB   82(R7), R1                           ; 1753
            50         0000G        CF  D0 005B3         MOVL    LBR$GL_CONTROL, R0                   ; 1745
            01            03        A0  91 005B8         CMPB    3(R0), #1                            ;
                                    1A  12 005BC         BNEQ    65$                                  ;
 7E         56 A7                   04  C1 005BE         ADDL3   #4, 86(CONTEXT), -(SP)               ; 1753
                                    51  DD 005C3         PUSHL   R1                                   ;
      0000G DF                      02  FB 005C5         CALLS   #2, @DCX_EXPAND_INIT                 ;
            26                      50  E9 005CA         BLBC    STATUS, 67$                          ;
            51            04        A2  9E 005CD         MOVAB   4(R2), R1                            ; 1754
            50          0800        8F  3C 005D1         MOVZWL  #2048, R0                            ;
                                    18  11 005D6         BRB     66$                                  ;
 7E         56 A7                   04  C1 005D8 65$:    ADDL3   #4, 86(CONTEXT), -(SP)               ; 1764
                                    51  DD 005DD         PUSHL   R1                                   ;
      0000G DF                      02  FB 005DF         CALLS   #2, @DCX_COMPRESS_INIT               ;
            0C                      50  E9 005E4         BLBC    STATUS, 67$                          ;
            51            04        A2  9E 005E7         MOVAB   4(R2), R1                            ; 1765
            50          1000        8F  3C 005EB         MOVZWL  #4096, R0                            ;
                                 0000G  30 005F0 66$:    BSBW    GET_MEM                              ;
            01                      50  E8 005F3 67$:    BLBS    STATUS, 68$                          ;
                                        04 005F6         RET                                          ;
         02 A2          010E        8F  B0 005F7 68$:    MOVW    #270, 2(R2)                          ; 1768
                          08        BE  D5 005FD 69$:    TSTL    @8(SP)                               ; 1772
                                    08  13 00600         BEQL    70$                                  ;
         18 BE      13071956        8F  D0 00602         MOVL    #319232342, @24(SP)                  ; 1774
      0000G CF                      66  9A 0060A 70$:    MOVZBL  (HEADER), LBR$GL_RMSSTV              ; 1775
                       0000G        CF  D5 0060F         TSTL    LBR$GL_MAXREAD                       ; 1779
                                    44  12 00613         BNEQ    74$                                  ;
                          34        AE  9F 00615         PUSHAB  BLKSIZ                               ; 1781
                                    7E  D4 00618         CLRL    -(SP)                                ;
  000000000G 00                     02  FB 0061A         CALLS   #2, SYS$ADJWSL                       ;
 50         34 AE                   32  C3 00621         SUBL3   #50, BLKSIZ, R0                      ; 1782
               32                   50  D1 00626         CMPL    R0, #50                              ;
               03                   15  00629            BLEQ    71$                                  ;
               50                   32  D0 0062B         MOVL    #50, R0                              ;
      0000G CF                      50  D0 0062E 71$:    MOVL    R0, LBR$GL_MAXREAD                   ;
            02         0000G        CF  D1 00633         CMPL    LBR$GL_MAXREAD, #2                   ; 1783
                                    05  18 00638         BGEQ    72$                                  ;
      0000G CF                      02  D0 0063A         MOVL    #2, LBR$GL_MAXREAD                   ; 1784
 05         36 A9                   01  E1 0063F 72$:    BBC     #1, 54(LBRNAM), 73$                  ; 1785
```

```
                         0000G  CF             02  D0 00644        MOVL    #2, LBR$GL_MAXREAD                         ; 1786
          0000G  CF      0000G  CF             32  C1 00649 73$:   ADDL3   #50, LBR$GL_MAXREAD, MEM$L_MEMEXP          ; 1787
          0000G  CF      0000G  CF             09  78 00651        ASHL    #9, MEM$L_MEMEXP, MEM$L_MAXBLK             ; 1788
                                50   0000G  CF  D0 00659 74$:       MOVL    LBR$GL_CONTROL, R0                         ; 1790
                                02     03   A0  91 0065E           CMPB    3(R0), #2                                  ;
                                            05  13 00662           BEQL    75$                                        ;
                                       03   A0  95 00664           TSTB    3(R0)                                      ; 1791
                                            45  12 00667           BNEQ    79$                                        ;
                                       34   A6  9F 00669 75$:       PUSHAB  52(HEADER)                                 ; 1797
                    00000000G   00          01  FB 0066C           CALLS   #1, SYS$GETTIM                             ;
                           04   A7          08  88 00673           BISB2   #8, 4(CONTEXT)                             ; 1798
                 51     0000G   CF          1A  C1 00677           ADDL3   #26, LBR$GL_CONTROL, R1                    ; 1805
                                50   0200   8F  3C 0067D           MOVZWL  #512, R0                                   ;
                                          0000G 30 00682           BSBW    GET_ZMEM                                   ;
                                6E          50  D0 00685           MOVL    R0, STATUS                                 ;
                                0C          6E  E8 00688           BLBS    STATUS, 78$                                ; 1806
                                       04   AC  DD 0068B 76$:       PUSHL   CONTROL_INDEX                              ; 1809
                         0000V  CF          01  FB 0068E           CALLS   #1, LBR$CLOSE                              ;
                                50          6E  D0 00693 77$:       MOVL    STATUS, R0                                 ; 1810
                                                04 00696           RET                                                ;
                                56   0000G  CF  D0 00697 78$:       MOVL    LBR$GL_CONTROL, R6                         ; 1812
             1A   B6       0A   B6   0200   8F  28 0069C           MOVC3   #512, @10(R6), @26(R6)                     ;
                                50     1A   A6  D0 006A4           MOVL    26(R6), OLDHEADER                          ; 1813
                           40   A0   DEAD   8F  B0 006A8           MOVW    #-8531, 64(OLDHEADER)                      ; 1814
                                50   0000G  CF  D0 006AE 79$:       MOVL    LBR$GL_CONTROL, R0                         ; 1816
                           06   A0          02  88 006B3           BISB2   #2, 6(R0)                                  ;
                           04   A7          01  88 006B7           BISB2   #1, 4(CONTEXT)                             ; 1817
                                50     10   AE  D0 006BB           MOVL    RETURN_STATUS, R0                          ; 1819
                                                04 006BF           RET                                                ; 1820

; Routine Size:  1728 bytes,    Routine Base:  $CODE$ + 0278
```

```
1 %SBTTL  'LBR$CLOSE';
1 GLOBAL ROUTINE lbr$close (control_index) =
2 BEGIN
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !      This routine closes an open library file and release all virtual
2 !      memory allocated while the file was open.
2 !
2 ! CALLING SEQUENCE:
2 !
2 !      status = LBR$CLOSE (control_index)
2 !
2 ! INPUT PARAMETERS:
2 !
2 !      control_index          is the address of a longword containing the
2 !                              index returned from LBR$INI_CONTROL
2 !
2 ! IMPLICIT INPUTS:
2 !
2 !      NONE
2 !
2 ! OUTPUT PARAMETERS:
2 !      NONE
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !      The library file is closed.  All virtual memory allocated for the
2 !      processing of the library is deallocated.
2 !
2 ! ROUTINE VALUE:
2 !
2 !      lbr$_libnotopn  library was not open
2 !      lbr$_illctl     illegal control block
2 !
2 ! SIDE EFFECTS:
2 !      NONE
2 !
2 !--
2 LOCAL
2        header_status,
2        cache_status,
2        disc_rec_sts,
2        close_status;
2
2 IF ..control_index EQL 0                ! 0 gets a return immediately
2     THEN RETURN true;
2 IF NOT validate_ctl (..control_index)   !Validate the control table
2 THEN RETURN lbr$_illctl
2 ELSE IF NOT .lbr$gl_control [lbr$v_open]        !library must be open also
2 THEN RETURN lbr$_libnotopn
3 ELSE BEGIN
3 !
3 ! Write back to file if necessary, close library, and deallocate
3 ! dynamic virtual memory.
3 !
```

```
  3 LOCAL
  3     lbrfab : BBLOCK [fab$c_bln];                    !FAB for closing library
  3 BIND
  3     context = .lbr$gl_control [lbr$l_ctxptr] : BBLOCK, !Context block pointer
  3     header = .lbr$gl_control [lbr$l_hdrptr] : BBLOCK,   ! library header
  3     recrab = .context [ctx$l_recrab] : BBLOCK;  !Record RAB
  3
  3 header_status = true;
  3 disc_rec_sts = true;
  3 CH$FILL (0,fab$c_bln,lbrfab);               !Zero the FAB
  3 lbrfab [fab$b_bln] = fab$c_bln;             !Identify it as a FAB
  3 lbrfab [fab$b_bid] = fab$c_bid;
  3 lbrfab [fab$w_ifi] = .context [ctx$w_ifi];          !Set IFI for close
  3 !
  3 !       Write all modified blocks to the library file.
  3 !
  3 cache_status = dealloc_cache (); ! Write cached disk blocks if necessary
  3 IF .context  [ctx$v_hdrdirty]               ! If header modified,
  3 AND NOT .context  [ctx$v_oldlib]                    ! and not old format library
  4 THEN BEGIN
  4     INCR I FROM 1 TO .header [lhd$b_nindex]     !Clear the lock bit in index descriptors
  5     DO BEGIN
  5         BIND
  5             index_desc = header [lhd$c_idxdesc - idd$c_length, 0, 0, 0] :
  5                         BLOCKVECTOR [,idd$c_length, BYTE];
  5
  5         index_desc [.i, idd$v_locked] = false;
  4         END;
  4
  4     IF .cache_status    ! If no error in writing out cache then write header back
  4     THEN
  5         BEGIN
  5         header_status = write_block (.lbr$gl_control  [lbr$l_hdrptr], 1);
  5         perform(dealloc_mem(lbr$c_pagesize, .lbr$gl_control[lbr$l_oldhdrptr]));
  4         END;
  3     END;
  3
  3 IF .header[lhd$l_dcxmapvbn] NEQ 0
  3 THEN
  4     BEGIN
  4     BIND
  4         dcx_rec_desc = context[ctx$l_dcxrecdsc] : BBLOCK [dsc$c_s_bln];
  4     IF .lbr$gl_control[lbr$b_func] EQL lbr$c_read
  4     THEN
  5         BEGIN
  5         perform((.dcx_expand_done) (context[ctx$l_dcxctx]));
  5         perform(dealloc_mem(lbr$c_maxrecsiz, .dcx_rec_desc[dsc$a_pointer]));
  5         END
  4     ELSE
  5         BEGIN
  5         perform((.dcx_compress_done) (context[ctx$l_dcxctx]));
  5         perform(dealloc_mem(lbr_dcx$c_maxrecsiz, .dcx_rec_desc[dsc$a_pointer]));
  4         END;
P 4     perform(dealloc_mem(..context[ctx$l_dcxmapdsc],
  4                         .(.context[ctx$l_dcxmapdsc]+4)));
  3     END;
  3
```

```
; 1119          1935  3 !
; 1120          1936  3 !       Close the file.
; 1121          1937  3 !
; 1122          1938  3 IF .recrab [rab$w_isi] NEQ 0          !If stream is connected
; 1123          1939  3 THEN disc_rec_sts = $DISCONNECT (RAB = recrab); !Disconnect record stream
; 1124          1940  3 close_status  = $CLOSE (FAB = lbrfab);  !Close the file
; 1125          1941  3 lbr_deal_mem (..control_index);         !Deallocate memory
; 1126          1942  3 IF NOT .header_status
; 1127          1943  3     THEN RETURN .header_status
; 1128          1944  3     ELSE IF NOT .cache_status
; 1129          1945  3         THEN RETURN .cache_status
; 1130          1946  3         ELSE IF NOT .disc_rec_sts
; 1131          1947  3             THEN RETURN .disc_rec_sts
; 1132          1948  3             ELSE IF NOT .close_status
; 1133          1949  3                 THEN RETURN .close_status
; 1134          1950  3                 ELSE RETURN lbr$_normal
; 1135          1951  3 END
; 1136          1952  1 END;                                     ! Of LBR$CLOSE


                                                                          .EXTRN  SYS$DISCONNECT, SYS$CLOSE

                                                      0FFC 00000          .ENTRY  LBR$CLOSE, Save R2,R3,R4,R5,R6,R7,R8,R9,-     ; 1822
                                                                                  R10,R11                                         ;
                                   5E        B0   AE   9E 00002           MOVAB   -80(SP), SP                                     ;
                                   59        04   BC   D0 00006           MOVL    @CONTROL_INDEX, R9                              ; 1867
                                                  04   12 0000A           BNEQ    1$                                              ;
                                   50             01   D0 0000C           MOVL    #1, R0                                          ; 1868
                                                       04 0000F           RET                                                     ;
                                   50             59   D0 00010 1$:       MOVL    R9, R0                                          ; 1869
                                               0000G   30 00013           BSBW    VALIDATE_CTL                                    ;
                                   08             50   E8 00016           BLBS    R0, 2$                                          ;
                                   50 00000000G   8F   D0 00019           MOVL    #LBR$_ILLCTL, R0                                ; 1871
                                                       04 00020           RET                                                     ;
                                   50      0000G  CF   D0 00021 2$:       MOVL    LBR$GL_CONTROL, R0                              ;
                  08         06    A0             01   E0 00026           BBS     #1, 6(R0), 3$                                   ;
                                   50 00000000G   8F   D0 0002B           MOVL    #LBR$_LIBNOTOPN, R0                             ; 1872
                                                       04 00032           RET                                                     ;
                                   56        0E   A0   D0 00033 3$:       MOVL    14(R0), R6                                      ; 1881
                                   57        0A   A0   D0 00037           MOVL    10(R0), R7                                      ; 1882
                                   58        0C   A6   D0 0003B           MOVL    12(R6), R8                                      ; 1883
                                   5B             01   D0 0003F           MOVL    #1, HEADER_STATUS                               ; 1885
                                   5A             01   D0 00042           MOVL    #1, DISC_REC_STS                                ; 1886
    0050    8F    00               6E             00   2C 00045           MOVC5   #0, (SP), #0, #80, LBRFAB                       ; 1887
                                   6E                     0004C                                                                   ;
                                   6E      5003   8F   B0 0004D           MOVW    #20483, LBRFAB                                  ; 1889
                             02    AE        02   A6   B0 00052           MOVW    2(R6), LBRFAB+2                                 ; 1890
                          0000G    CF             00   FB 00057           CALLS   #0, DEALLOC_CACHE                               ; 1894
                                   53             50   D0 0005C           MOVL    R0, CACHE_STATUS                                ;
                  42         04    A6             03   E1 0005F           BBC     #3, 4(R6), 6$                                   ; 1895
                  3D         04    A6             05   E0 00064           BBS     #5, 4(R6), 6$                                   ; 1896
                                   51        01   A7   9A 00069           MOVZBL  1(R7), R1                                       ; 1898
                                                  50   D4 0006D           CLRL    I                                               ;
                                                  08   11 0006F           BRB     5$                                              ;
                                          00BC C740   7F 00071 4$:       PUSHAQ  188(R7)[I]                                      ; 1904
                                   9E             02   8A 00076           BICB2   #2, @(SP)+                                      ;
```

```
         F4          50           51  F3 00079 5$:      AOBLEQ  R1, I, 4$                                 ; 1898
                     26           53  E9 0007D          BLBC    CACHE_STATUS, 6$                          ; 1907
                     52      0000G CF  D0 00080          MOVL    LBR$GL_CONTROL, R2                        ; 1910
                     51           01  D0 00085          MOVL    #1, R1                                    ;
                     50        0A  A2  D0 00088          MOVL    10(R2), R0                                ;
                              0000G 30 0008C          BSBW    WRITE_BLOCK                               ;
                     5B           50  D0 0008F          MOVL    R0, HEADER_STATUS                         ;
                     52      0000G CF  D0 00092          MOVL    LBR$GL_CONTROL, R2                        ; 1911
                     51        1A  A2  D0 00097          MOVL    26(R2), R1                                ;
                     50      0200  8F  3C 0009B          MOVZWL  #512, R0                                  ;
                              0000G 30 000A0          BSBW    DEALLOC_MEM                               ;
                     4F           50  E9 000A3          BLBC    STATUS, 9$                                ;
                             008C  C7  D5 000A6 6$:      TSTL    140(R7)                                   ; 1915
                                  4C  13 000AA          BEQL    10$                                       ;
                     52        5A  A6  9E 000AC          MOVAB   90(R6), R2                                ; 1919
                     50      0000G CF  D0 000B0          MOVL    LBR$GL_CONTROL, R0                        ; 1920
                     01        03  A0  91 000B5          CMPB    3(R0), #1                                 ;
                                  16  12 000B9          BNEQ    7$                                        ;
                               52  A6  9F 000BB          PUSHAB  82(R6)                                    ; 1923
               0000G DF           01  FB 000BE          CALLS   #1, @DCX_EXPAND_DONE                      ;
                     79           50  E9 000C3          BLBC    STATUS, 16$                               ;
                     51        04  A2  D0 000C6          MOVL    4(R2), R1                                 ; 1924
                     50      0800  8F  3C 000CA          MOVZWL  #2048, R0                                 ;
                                  14  11 000CF          BRB     8$                                        ;
                               52  A6  9F 000D1 7$:      PUSHAB  82(R6)                                    ; 1928
               0000G DF           01  FB 000D4          CALLS   #1, @DCX_COMPRESS_DONE                    ;
                     63           50  E9 000D9          BLBC    STATUS, 16$                               ;
                     51        04  A2  D0 000DC          MOVL    4(R2), R1                                 ; 1929
                     50      1000  8F  3C 000E0          MOVZWL  #4096, R0                                 ;
                              0000G 30 000E5 8$:      BSBW    DEALLOC_MEM                               ;
                     54           50  E9 000E8          BLBC    STATUS, 16$                               ;
                     52        56  A6  D0 000EB          MOVL    86(R6), R2                                ; 1932
                     50           62  7D 000EF          MOVQ    (R2), R0                                  ;
                              0000G 30 000F2          BSBW    DEALLOC_MEM                               ;
                     47           50  E9 000F5 9$:      BLBC    STATUS, 16$                               ;
                               02  A8  B5 000F8 10$:     TSTW    2(R8)                                     ; 1938
                                  0C  13 000FB          BEQL    11$                                       ;
                                  58  DD 000FD          PUSHL   R8                                        ; 1939
           00000000G 00           01  FB 000FF          CALLS   #1, SYS$DISCONNECT                        ;
                     5A           50  D0 00106          MOVL    R0, DISC_REC_STS                          ;
                                  5E  DD 00109 11$:     PUSHL   SP                                        ; 1940
           00000000G 00           01  FB 0010B          CALLS   #1, SYS$CLOSE                             ;
                     52           50  D0 00112          MOVL    R0, CLOSE_STATUS                          ;
                                  59  DD 00115          PUSHL   R9                                        ; 1941
               0000V CF           01  FB 00117          CALLS   #1, LBR_DEAL_MEM                          ;
                     04           5B  E8 0011C          BLBS    HEADER_STATUS, 12$                        ; 1942
                     50           5B  D0 0011F          MOVL    HEADER_STATUS, R0                         ; 1944
                                  04 00122          RET                                               ;
                     04           53  E8 00123 12$:     BLBS    CACHE_STATUS, 13$                         ;
                     50           53  D0 00126          MOVL    CACHE_STATUS, R0                          ; 1945
                                  04 00129          RET                                               ;
                     04           5A  E8 0012A 13$:     BLBS    DISC_REC_STS, 14$                         ; 1946
                     50           5A  D0 0012D          MOVL    DISC_REC_STS, R0                          ; 1947
                                  04 00130          RET                                               ;
                     04           52  E8 00131 14$:     BLBS    CLOSE_STATUS, 15$                         ; 1948
                     50           52  D0 00134          MOVL    CLOSE_STATUS, R0                          ; 1949
                                  04 00137          RET                                               ;
                              50 00000000G  8F  D0 00138 15$:    MOVL    #LBR$_NORMAL, R0                  ; 1950
                                                04 0013F 16$:    RET                                       ; 1952

; Routine Size:  320 bytes,    Routine Base:  $CODE$ + 0938
; 1138          1953  1 %SBTTL  'all_control_idx';
; 1139          1954  1 ROUTINE all_control_idx (control_index, control_table) =
; 1140          1955  2 BEGIN
; 1141          1956  2 !
; 1142          1957  2 ! This routine allocates an index number and returns it in control_index.
; 1143          1958  2 ! If no index number is found then lbr$_toomnylib is returned.  the control
; 1144          1959  2 ! table is then allocated and the address returned in control_table.  The
; 1145          1960  2 ! control table address is also store in lbr$al_ctltab.
; 1146          1961  2 !
; 1147          1962  2 INCR i FROM 0 TO (lbr$c_maxctl - 1) DO
; 1148          1963  2     IF .lbr$al_ctltab [.i] EQL 0                !If we found one
; 1149          1964  2     THEN
; 1150          1965  3         BEGIN
; 1151          1966  3         .control_index = .i + 1;               !Return index to caller
; 1152          1967  3         perform (get_zmem (lbr$c_length, .control_table)); !Allocate the control table
; 1153          1968  3         lbr$al_ctltab [.i] = ..control_table; !Set address into table
; 1154          1969  3         IF .i GTRU .lbr$gl_hictl THEN lbr$gl_hictl = .i; !Update hictl if needed
; 1155          1970  3         RETURN true;
; 1156          1971  2         END;
; 1157          1972  2 RETURN lbr$_toomnylib
; 1158          1973  1 END;                                        !Of all_control_idx



                                               0FFC 00000 ALL_CONTROL_IDX:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1954
                                          52   D4 00002           CLRL    I                                             ; 1962
                                   0000GCF42   D5 00004 1$:       TSTL    LBR$AL_CTLTAB[I]                              ; 1963
                                          29   12 00009           BNEQ    3$                                            ;
                   04    BC         01    A2   9E 0000B           MOVAB   1(R2), @CONTROL_INDEX                         ; 1966
                         51         08    AC   D0 00010           MOVL    CONTROL_TABLE, R1                             ; 1967
                         50               1E   D0 00014           MOVL    #30, R0                                       ;
                                       0000G   30 00017           BSBW    GET_ZMEM                                      ;
                         22               50   E9 0001A           BLBC    STATUS, 4$                                    ;
                  0000GCF42         08    BC   D0 0001D           MOVL    @CONTROL_TABLE, LBR$AL_CTLTAB[I]              ; 1968
                  0000G  CF               52   D1 00024           CMPL    I, LBR$GL_HICTL                               ; 1969
                                          05   1B 00029           BLEQU   2$                                            ;
                  0000G  CF               52   D0 0002B           MOVL    I, LBR$GL_HICTL                               ;
                         50               01   D0 00030 2$:       MOVL    #1, R0                                        ; 1970
                                               04 00033           RET                                                   ;
            CC           52               0F   F3 00034 3$:       AOBLEQ  #15, I, 1$                                    ; 1963
                         50 00000000G     8F   D0 00038           MOVL    #LBR$_TOOMNYLIB, R0                           ; 1972
                                               04 0003F 4$:       RET                                                   ; 1973

; Routine Size:  64 bytes,    Routine Base:  $CODE$ + 0A78
; 1160          1974  1 %SBTTL  'dea_control_idx';
; 1161          1975  1 ROUTINE dea_control_idx (control_index) : NOVALUE =
; 1162          1976  2 BEGIN
; 1163          1977  2 !
; 1164          1978  2 ! This routine deallocates an index number and updates lbr$gl_hictl if necessary.
; 1165          1979  2 !
; 1166          1980  2 LOCAL
; 1167          1981  2     index;
; 1168          1982  2
; 1169          1983  2 index = .control_index - 1;
; 1170          1984  2 dealloc_mem (lbr$c_length, .lbr$al_ctltab [.index]);    !Deallocate control table
; 1171          1985  2 lbr$al_ctltab [.index] = 0;                       !Zero the table entry
; 1172          1986  2 INCR i FROM .index TO .lbr$gl_hictl               !See if any higher indices allocated
; 1173          1987  2 DO IF .lbr$al_ctltab [.i] NEQ 0                   !If there are
; 1174          1988  2         THEN return                                !then done
; 1175          1989  2         ELSE IF .i EQL .lbr$gl_hictl               !If we go all the way to end
; 1176          1990  3             THEN BEGIN
; 1177          1991  3                 lbr$gl_hictl = .index;            !Then new low is just below us
; 1178          1992  3                 RETURN;
; 1179          1993  2                 END;
; 1180          1994  1 END;                                               !Of dea_control_idx



                                              0FFC 00000 DEA_CONTROL_IDX:
                                                              .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 1975
                        54      0000G  CF   9E 00002          MOVAB   LBR$AL_CTLTAB, R4                             ;
                        53      0000G  CF   9E 00007          MOVAB   LBR$GL_HICTL, R3                              ;
        52      04      AC             01   C3 0000C          SUBL3   #1, CONTROL_INDEX, INDEX                      ; 1983
                        51           6442   D0 00011          MOVL    LBR$AL_CTLTAB[INDEX], R1                      ; 1984
                        50             1E   D0 00015          MOVL    #30, R0                                       ;
                                    0000G   30 00018          BSBW    DEALLOC_MEM                                   ;
                                     6442   D4 0001B          CLRL    LBR$AL_CTLTAB[INDEX]                          ; 1985
                        51             63   D0 0001E          MOVL    LBR$GL_HICTL, R1                              ; 1986
                        50        FF   A2   9E 00021          MOVAB   -1(R2), I                                     ;
                                       0E   11 00025          BRB     2$                                            ;
                                     6440   D5 00027 1$:      TSTL    LBR$AL_CTLTAB[I]                              ; 1987
                                       0D   12 0002A          BNEQ    3$                                            ;
                        63             50   D1 0002C          CMPL    I, LBR$GL_HICTL                               ; 1989
                                       04   12 0002F          BNEQ    2$                                            ;
                        63             52   D0 00031          MOVL    INDEX, LBR$GL_HICTL                           ; 1991
                                            04 00034          RET                                                   ; 1990
        EE              50             51   F3 00035 2$:      AOBLEQ  R1, I, 1$                                     ; 1987
                                            04 00039 3$:      RET                                                   ; 1994

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 0AB8
; 1182          1995  1 %SBTTL  'prealloc_index';
; 1183          1996  1 ROUTINE prealloc_index (header, create_options) =
; 1184          1997  2 BEGIN
; 1185          1998  2 !
; 1186          1999  2 ! This routine pre-allocates index blocks in a library being created.
; 1187          2000  2 !
; 1188          2001  2 MAP
; 1189          2002  2     header : REF BBLOCK,
; 1190          2003  2     create_options : REF BBLOCK;
; 1191          2004  2
; 1192          2005  2 LOCAL
; 1193          2006  2     indexblocks,
; 1194          2007  2     entall,
; 1195          2008  2     entsperblk,
; 1196          2009  2     bufblks,
; 1197          2010  2     cachentry : REF BBLOCK,
; 1198          2011  2     bufadr : REF VECTOR [,LONG];
; 1199          2012  2
; 1200          2013  2 entall = .create_options [cre$l_entall];                    !Pick up user request
; 1201          2014  2 IF .entall EQL 0 THEN entall = lbr$c_defentall; ! and default null request
; 1202          2015  3 entsperblk = (IF .create_options [cre$l_keylen] + 1 NEQ 0 !Determine if ASCII or binary
; 1203          2016  3                 THEN .create_options [cre$l_keylen]   ! and determine length of keys
; 1204          2017  2                 ELSE 4);
; 1205          2018  2 entsperblk = (index$c_length - index$c_entries) / (.entsperblk + rfa$c_length);
; 1206          2019  2 indexblocks = .entall/.entsperblk;                  !compute # blocks to allocate
; 1207          2020  2 IF .indexblocks EQL 0                               !Always allocate at least 1 block
; 1208          2021  2     THEN indexblocks = 1;
; 1209          2022  2 IF .create_options [cre$l_vertyp] EQL cre$c_vmsv2        ! If index is not variable key storage
; 1210          2023  2 THEN indexblocks = (.indexblocks*4)/3;                     ! add in the fudge factor
; 1211          2024  2
; 1212          2025  2 !
; 1213          2026  2 ! Write the pre-allocated index
; 1214          2027  2 !
; 1215          2028  2 INCRU i FROM 1 TO .indexblocks                      !Create the index
; 1216          2029  3 DO BEGIN
; 1217          2030  3     perform (get_mem (lbr$c_pagesize, bufadr)); !Allocate a page
; 1218          2031  3     perform (add_cache (.i+1, cachentry));       !Add to cache
; 1219          2032  3     cachentry [cache$l_address] = .bufadr;
; 1220          2033  3     cachentry [cache$v_dirty] = true;            !Mark block as modified
; 1221          2034  4     bufadr [0] = (IF .i NEQ .indexblocks                !Set link to next block
; 1222          2035  4                     THEN .i+2                     !
; 1223          2036  3                     ELSE 0);                      ! or 0 if on last block
; 1224          2037  2     END;
; 1225          2038  2 header [lhd$l_hipreal] = .indexblocks + 1;         !Set vbn of highest preallocated index block
; 1226          2039  2 header [lhd$l_nextvbn] = .indexblocks + 2;         !Set next available vbn
; 1227          2040  2 header [lhd$l_freeidx] = 2;                        ! and pointer to first free index block
; 1228          2041  2 header [lhd$l_freidxblk] = .indexblocks;                  !Set count of available blocks
; 1229          2042  2 RETURN true
; 1230          2043  1 END;                                               !OF prealloc_index


                                         0FFC 00000 PREALLOC_INDEX:
                                                    .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11         ; 1996
                           5E            08  C2 00002       SUBL2   #8, SP                                       ;
```

```
                                51      08  AC  D0 00005          MOVL    CREATE_OPTIONS, R1                          ; 2013
                                53      14  A1  D0 00009          MOVL    20(R1), ENTALL                              ;
                                            05  12 0000D          BNEQ    1$                                          ; 2014
                                53    012C  8F  3C 0000F          MOVZWL  #300, ENTALL                                ;
                 50        04   A1          01  C1 00014 1$:      ADDL3   #1, 4(R1), R0                               ; 2015
                                            06  13 00019          BEQL    2$                                          ;
                                50      04  A1  D0 0001B          MOVL    4(R1), ENTSPERBLK                           ; 2016
                                            03  11 0001F          BRB     3$                                          ;
                                50          04  D0 00021 2$:      MOVL    #4, ENTSPERBLK                              ; 2015
                                52      06  A0  9E 00024 3$:      MOVAB   6(R0), R2                                   ; 2018
                 50 000001F4    8F          52  C7 00028          DIVL3   R2, #500, ENTSPERBLK                        ;
                                53          50  C6 00030          DIVL2   ENTSPERBLK, INDEXBLOCKS                     ; 2019
                                            03  12 00033          BNEQ    4$                                          ; 2020
                                53          01  D0 00035          MOVL    #1, INDEXBLOCKS                             ; 2021
                                02      1C  A1  D1 00038 4$:      CMPL    28(R1), #2                                  ; 2022
                                            08  12 0003C          BNEQ    5$                                          ;
                 50             53          02  78 0003E          ASHL    #2, INDEXBLOCKS, R0                         ; 2023
                 53             50          03  C7 00042          DIVL3   #3, R0, INDEXBLOCKS                         ;
                                52          01  D0 00046 5$:      MOVL    #1, I                                       ; 2028
                                            3B  11 00049          BRB     9$                                          ;
                                51          6E  9E 0004B 6$:      MOVAB   BUFADR, R1                                  ; 2030
                                50    0200  8F  3C 0004E          MOVZWL  #512, R0                                    ;
                                         0000G  30 00053          BSBW    GET_MEM                                     ;
                                4B          50  E9 00056          BLBC    STATUS, 10$                                 ;
                                51      04  AE  9E 00059          MOVAB   CACHENTRY, R1                               ; 2031
                                50      01  A2  9E 0005D          MOVAB   1(I), R0                                    ;
                                         0000G  30 00061          BSBW    ADD_CACHE                                   ;
                                3D          50  E9 00064          BLBC    STATUS, 10$                                 ;
                                50      04  AE  D0 00067          MOVL    CACHENTRY, R0                               ; 2032
                           08   A0          6E  D0 0006B          MOVL    BUFADR, 8(R0)                               ;
                           0C   A0          01  88 0006F          BISB2   #1, 12(R0)                                  ; 2033
                                53          52  D1 00073          CMPL    I, INDEXBLOCKS                              ; 2034
                                            06  13 00076          BEQL    7$                                          ;
                                50      02  A2  9E 00078          MOVAB   2(R2), R0                                   ; 2035
                                            02  11 0007C          BRB     8$                                          ;
                                            50  D4 0007E 7$:      CLRL    R0                                          ; 2034
                           00   BE          50  D0 00080 8$:      MOVL    R0, @BUFADR                                 ;
                                            52  D6 00084          INCL    I                                           ; 2028
                                53          52  D1 00086 9$:      CMPL    I, INDEXBLOCKS                              ;
                                            C0  1B 00089          BLEQU   6$                                          ;
                                50      04  AC  D0 0008B          MOVL    HEADER, R0                                  ; 2038
                           5E   A0      01  A3  9E 0008F          MOVAB   1(R3), 94(R0)                               ;
                           52   A0      02  A3  9E 00094          MOVAB   2(R3), 82(R0)                               ; 2039
                           5A   A0          02  D0 00099          MOVL    #2, 90(R0)                                  ; 2040
                           56   A0          53  D0 0009D          MOVL    INDEXBLOCKS, 86(R0)                         ; 2041
                                50          01  D0 000A1          MOVL    #1, R0                                      ; 2042
                                                04 000A4 10$:     RET                                                 ; 2043

; Routine Size:  165 bytes,    Routine Base:  $CODE$ + 0AF2
```

```
1 %SBTTL  'lbr_deal_mem';
1 GLOBAL ROUTINE lbr_deal_mem (control_index) : NOVALUE =
2 BEGIN
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine deallocates all memory allocated during the processing
2 !       of a library.  This includes the librarian context block, the header
2 !       block, the RAB/NAM block, the block buffer, and any indices left.
2 !
2 ! CALLING SEQUENCE:
2 !
2 !       LBR_DEAL_MEM()
2 !
2 ! INPUT PARAMETERS:
2 !
2 !
2 ! OUTPUT PARAMETERS:
2 !       NONE
2 !
2 ! IMPLICIT OUTPUTS:
2 !
2 !       The librarian context block, the RAB/NAM block, and the block
2 !       buffer are all deallocated.
2 !
2 ! SIDE EFFECTS:
2 !       NONE
2 !
2 !--
2
2 LOCAL
2     context : REF BLOCK [,BYTE];                ! Pointer to context block
2
2 BIND
2     header= .lbr$gl_control[lbr$l_hdrptr]: BLOCK [, Byte];
2 IF header NEQ 0                                 !If there is a header
2 THEN
3     BEGIN
3     LOCAL
3         maxrecsiz;                              ! Maximum record size.
3     IF .header[lhd$l_dcxmapvbn] EQL 0 THEN      ! If not a DCX library
3         maxrecsiz = lbr$c_maxrecsiz             !   use normal maxrecsize,
3     ELSE                                        ! if DCX
3         maxrecsiz = lbr_dcx$c_maxrecsiz;        !   use larger value.
3     dealloc_mem (lbr$c_pagesize, header);       ! Then deallocate the header.
3     IF (context = .lbr$gl_control [lbr$l_ctxptr]) NEQ 0 !If there is a context block
3     THEN
4         BEGIN
4         IF .context [ctx$l_recrab] NEQ 0        !If there is a RAB allocated
4         THEN                                    !then deallocate it
4             dealloc_mem (rab$c_bln+nam$c_bln, .context [ctx$l_recrab]);
4         IF .context [ctx$l_readbuf] NEQ 0       !If read buffer allocated
4         THEN
4             dealloc_mem (.maxrecsiz, .context [ctx$l_readbuf]);
4         IF .context [ctx$l_rdbufr] NEQ 0        !Read buffer allocated?
4         THEN
```

```
; 1289          2101  4             dealloc_mem (.lbr$gl_maxread * lbr$c_pagesize,
; 1290          2102  4                         .context [ctx$l_rdbufr]);
; 1291          2103  4 !       IF .context [ctx$l_rphasht] NEQ 0           !Replace hash table allocated?
; 1292          2104  4 !       THEN dealloc_mem (lbr$c_pagesize, .context [ctx$l_rphasht]);
; 1293          2105  4         IF .context [ctx$l_cache] NEQ 0         !Disk block cache hash table allocated?
; 1294          2106  4         THEN
; 1295          2107  4             dealloc_mem (lbr$c_hashsize, .context [ctx$l_cache]);
; 1296          2108  4         dealloc_mem (ctx$c_length, .context);   !Deallocate the context block
; 1297          2109  3         END;
; 1298          2110  2     END;
; 1299          2111  2 dea_control_idx (.control_index);                       !Deassign control index
; 1300          2112  2 RETURN true
; 1301          2113  1 END;                                    ! Of lbr_deal_mem





                                           0FFC 00000          .ENTRY  LBR_DEAL_MEM, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 2045
                                                                       R10,R11                                        ;
                          54     0000G  CF   9E 00002          MOVAB   DEALLOC_MEM, R4                                ;
                          50     0000G  CF   D0 00007          MOVL    LBR$GL_CONTROL, R0                             ; 2079
                          51        0A  A0   D0 0000C          MOVL    10(R0), R1                                     ;
                                        6B   13 00010          BEQL    7$                                             ; 2080
                                 008C   C1   D5 00012          TSTL    140(R1)                                        ; 2085
                                        07   12 00016          BNEQ    1$                                             ;
                          53     0800   8F   3C 00018          MOVZWL  #2048, MAXRECSIZ                               ; 2086
                                        05   11 0001D          BRB     2$                                             ;
                          53     1000   8F   3C 0001F 1$:      MOVZWL  #4096, MAXRECSIZ                               ; 2088
                          50     0200   8F   3C 00024 2$:      MOVZWL  #512, R0                                       ; 2089
                                        64   16 00029          JSB     DEALLOC_MEM                                    ;
                          50     0000G  CF   D0 0002B          MOVL    LBR$GL_CONTROL, R0                             ; 2090
                          52        0E  A0   D0 00030          MOVL    14(R0), CONTEXT                                ;
                                        47   13 00034          BEQL    7$                                             ;
                                    0C  A2   D5 00036          TSTL    12(CONTEXT)                                    ; 2093
                                        0A   13 00039          BEQL    3$                                             ;
                          51        0C  A2   D0 0003B          MOVL    12(CONTEXT), R1                                ; 2095
                          50        A4  8F   9A 0003F          MOVZBL  #164, R0                                       ;
                                        64   16 00043          JSB     DEALLOC_MEM                                    ;
                                    2E  A2   D5 00045 3$:      TSTL    46(CONTEXT)                                    ; 2096
                                        09   13 00048          BEQL    4$                                             ;
                          51        2E  A2   D0 0004A          MOVL    46(CONTEXT), R1                                ; 2098
                          50            53   D0 0004E          MOVL    MAXRECSIZ, R0                                  ;
                                        64   16 00051          JSB     DEALLOC_MEM                                    ;
                                    32  A2   D5 00053 4$:      TSTL    50(CONTEXT)                                    ; 2099
                                        0C   13 00056          BEQL    5$                                             ;
             50     0000G   CF          09   78 00058          ASHL    #9, LBR$GL_MAXREAD, R0                         ; 2101
                          51        32  A2   D0 0005E          MOVL    50(CONTEXT), R1                                ;
                                        64   16 00062          JSB     DEALLOC_MEM                                    ;
                                    08  A2   D5 00064 5$:      TSTL    8(CONTEXT)                                     ; 2105
                                        0B   13 00067          BEQL    6$                                             ;
                          51        08  A2   D0 00069          MOVL    8(CONTEXT), R1                                 ; 2107
                          50     0200   8F   3C 0006D          MOVZWL  #512, R0                                       ;
                                        64   16 00072          JSB     DEALLOC_MEM                                    ;
                          51            52   D0 00074 6$:      MOVL    CONTEXT, R1                                    ; 2108
                          50        86  8F   9A 00077          MOVZBL  #134, R0                                       ;
                                        64   16 0007B          JSB     DEALLOC_MEM                                    ;
```

```
                                          04    AC  DD 0007D 7$:      PUSHL    CONTROL_INDEX                                     ; 2111
                           FE9C    CF               01  FB 00080      CALLS    #1, DEA_CONTROL_IDX                               ;
                                                        04 00085      RET                                                        ; 2113

; Routine Size:  134 bytes,    Routine Base:  $CODE$ + 0B97
```

```
1 %SBTTL  'LBR$GET_HEADER';
1 GLOBAL ROUTINE lbr$get_header (control_index, retary) =
2 BEGIN
2 !++
2 !
2 ! FUNCTIONAL DESCRIPTION:
2 !
2 !       This routine retrieves the information from the library header and stores
2 !       it into an array for the caller.
2 !
2 ! CALLING SEQUENCE:
2 !
2 !       status = LBR$GET_HEADER (control_index, retary)
2 !
2 ! INPUT PARAMETERS:
2 !
2 !       control_index           is the address of a longword containing the
2 !                                index returned by LBR$INI_CONTROL.
2 !
2 ! IMPLICIT INPUTS:
2 !       NONE
2 !
2 ! OUTPUT PARAMETERS:
2 !
2 !       The 128-longword array retary is filled in with the information from
2 !       the library header.
2 !
2 ! IMPLICIT OUTPUTS:
2 !       NONE
2 !
2 ! ROUTINE VALUE:
2 !
2 !       lbr$_libnotopn  library was not open
2 !       lbr$_illctl     illegal control block
2 !
2 ! SIDE EFFECTS:
2 !       NONE
2 !
2 !--
2
2 MAP
2       retary : REF BLOCK [,BYTE];
2 LOCAL
2       header : REF BLOCK [,BYTE];             !Pointer to header
2
2 perform (validate_ctl (..control_index));       !validate the control table
2 IF NOT .lbr$gl_control [lbr$v_open]            !If library not open
2 THEN RETURN lbr$_libnotopn;                    ! thats an error too
2 IF (header = .lbr$gl_control [lbr$l_hdrptr]) EQL 0!If no header in memory
4 OR ((.header [lhd$l_sanity] NEQ lhd$c_saneid)   !  or header appears bogus
4 AND (.header [lhd$l_sanity] NEQ lhd$c_saneid3)
3 AND (.header [lhd$l_sanity] NEQ lhd$c_saneidc))
2 THEN RETURN lbr$_illctl                    !  then error
3 ELSE BEGIN
3 BIND
3       hdrnxtrfa = header [lhd$b_nextrfa] : BLOCK [,BYTE],
3       retnxtrfa = retary [lhi$b_nextrfa] : BLOCK [,BYTE];
```

```
; 1360          2171  3 !
; 1361          2172  3 ! Copy info from the header into the array
; 1362          2173  3 !
; 1363          2174  3 retary [lhi$l_type] = .header [lhd$b_type];      !Library type
; 1364          2175  3 retary [lhi$l_nindex] = .header [lhd$b_nindex]; !Number of indices
; 1365          2176  3 retary [lhi$l_majorid] = .header [lhd$w_majorid]; !Copy format level major/minor id
; 1366          2177  3 retary [lhi$l_minorid] = .header [lhd$w_minorid];
; 1367          2178  3 CH$MOVE (32,header [lhd$t_lbrver],retary [lhi$t_lbrver]); !Creating librarian version
; 1368          2179  3 retary [lhi$l_credat] = .header [lhd$l_credat]; !Creation date/time
; 1369          2180  3 retary [lhi$l_credat]+4 = .(header [lhd$l_credat]+4);
; 1370          2181  3 retary [lhi$l_updtim] = .header [lhd$l_updtim]; !Date/time of last update
; 1371          2182  3 retary [lhi$l_updtim]+4 = .(header [lhd$l_updtim]+4);
; 1372          2183  3 retary [lhi$l_updhis] = 0;       ! Update history VBN is now obsolete
; 1373          2184  3 retary [lhi$l_freevbn] = .header [lhd$l_freevbn];        ! 1st deleted block
; 1374          2185  3 retary [lhi$l_freeblk] = .header [lhd$l_freeblk];        ! Number of deleted blocks
; 1375          2186  3 retnxtrfa [rfa$l_vbn] = .hdrnxtrfa [rfa$l_vbn];
; 1376          2187  3 retnxtrfa [rfa$w_offset] = .hdrnxtrfa [rfa$w_offset];
; 1377          2188  3 retary [lhi$w_rfaxtr] = 0;
; 1378          2189  3 retary [lhi$l_nextvbn] = .header [lhd$l_nextvbn];        !Next VBN to allocate
; 1379          2190  3 retary [lhi$l_freidxblk] = .header [lhd$l_freidxblk];
; 1380          2191  3 retary [lhi$l_freeidx] = .header [lhd$l_freeidx];
; 1381          2192  3 retary [lhi$l_hipreal] = .header [lhd$l_hipreal];
; 1382          2193  3 retary [lhi$l_idxblks] = .header [lhd$l_idxblks];
; 1383          2194  3 retary [lhi$l_idxcnt] = .header [lhd$l_idxcnt];
; 1384          2195  3 retary [lhi$l_modcnt] = .header [lhd$l_modcnt];
; 1385          2196  3 retary [lhi$l_mhdusz] = .header [lhd$b_mhdusz];
; 1386          2197  3 retary [lhi$l_maxluhrec] = .header [lhd$w_maxluhrec];
; 1387          2198  3 retary [lhi$l_numluhrec] = .header [lhd$w_numluhrec];
; 1388          2199  3 IF .header [lhd$w_closerror] EQL lhd$c_corrupted
; 1389          2200  3 THEN retary [lhi$l_libstatus] = false
; 1390          2201  3 ELSE retary [lhi$l_libstatus] = true;
; 1391          2202  3 RETURN lbr$_normal
; 1392          2203  3 END
; 1393          2204  1 END;                                                  !Of LBR$GET_HEADER



                                          0FFC 00000        .ENTRY  LBR$GET_HEADER, Save R2,R3,R4,R5,R6,R7,R8,- ; 2115
                                                                    R9,R10,R11                                  ;
                         50        04   BC  D0 00002        MOVL    @CONTROL_INDEX, R0                          ; 2159
                                     0000G  30 00006        BSBW    VALIDATE_CTL                                ;
                         01          50   E8 00009          BLBS    STATUS, 1$                                  ;
                                          04 0000C          RET                                                 ;
                         50      0000G   CF  D0 0000D 1$:   MOVL    LBR$GL_CONTROL, R0                          ; 2160
             08     06   A0              01  E0 00012       BBS     #1, 6(R0), 2$                               ;
                         50 00000000G   8F  D0 00017        MOVL    #LBR$_LIBNOTOPN, R0                         ; 2161
                                          04 0001E          RET                                                 ;
                         57        0A   A0  D0 0001F 2$:    MOVL    10(R0), HEADER                              ; 2162
                                    1E  13 00023            BEQL    3$                                          ;
                075BC371   8F        04   A7  D1 00025      CMPL    4(HEADER), #123454321                       ; 2163
                                    1C  13 0002D            BEQL    4$                                          ;
                0DEC2581   8F        04   A7  D1 0002F      CMPL    4(HEADER), #233579905                       ; 2164
                                    12  13 00037            BEQL    4$                                          ;
                13071956   8F        04   A7  D1 00039      CMPL    4(HEADER), #319232342                       ; 2165
                                    08  13 00041            BEQL    4$                                          ;
```

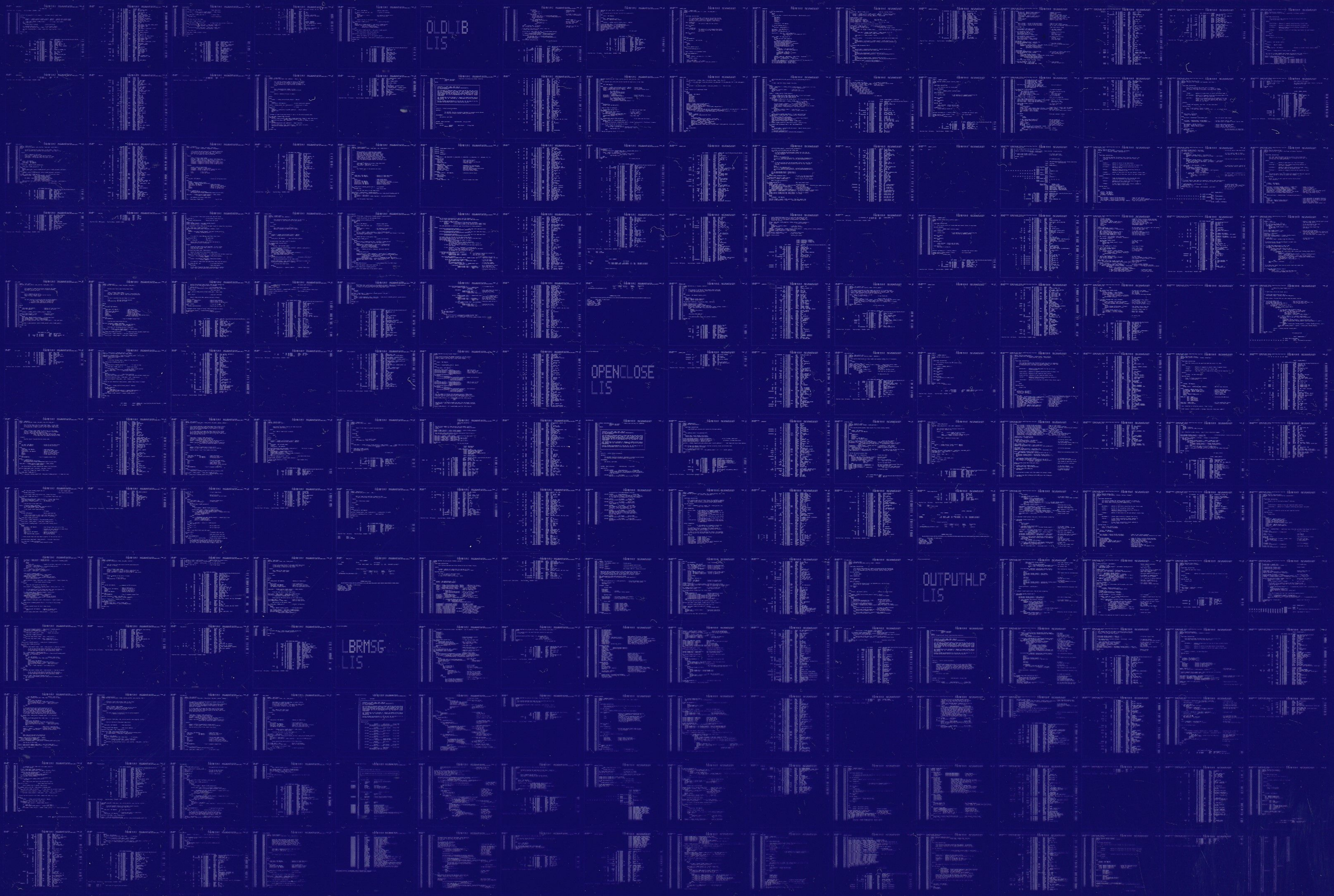
```
                                     50 00000000G  8F   D0 00043 3$:      MOVL    #LBR$_ILLCTL, R0                           ; 2170
                                                        04 0004A          RET
                                     59        4C   A7  9E 0004B 4$:      MOVAB   76(HEADER), R9                             ; 2169
                                     56        08   AC  D0 0004F          MOVL    RETARY, R6                                 ; 2170
                                     58        4C   A6  9E 00053          MOVAB   76(R6), R8
                                     66             67  9A 00057          MOVZBL  (HEADER), (R6)                             ; 2174
                                04   A6        01   A7  9A 0005A          MOVZBL  1(HEADER), 4(R6)                           ; 2175
                                08   A6        08   A7  3C 0005F          MOVZWL  8(HEADER), 8(R6)                           ; 2176
                                0C   A6        0A   A7  3C 00064          MOVZWL  10(HEADER), 12(R6)                         ; 2177
                10    A6        0C   A7             20  28 00069          MOVC3   #32, 12(HEADER), 16(R6)                    ; 2178
                                30   A6        2C   A7  7D 0006F          MOVQ    44(HEADER), 48(R6)                         ; 2179
                                38   A6        34   A7  7D 00074          MOVQ    52(HEADER), 56(R6)                         ; 2181
                                               40   A6  D4 00079          CLRL    64(R6)                                     ; 2183
                                44   A6        44   A7  7D 0007C          MOVQ    68(HEADER), 68(R6)                         ; 2184
                                     68             69  D0 00081          MOVL    (R9), (R8)                                 ; 2186
                                04   A8        04   A9  B0 00084          MOVW    4(R9), 4(R8)                               ; 2187
                                               52   A6  B4 00089          CLRW    82(R6)                                     ; 2188
                                54   A6        52   A7  7D 0008C          MOVQ    82(HEADER), 84(R6)                         ; 2189
                                5C   A6        5A   A7  7D 00091          MOVQ    90(HEADER), 92(R6)                         ; 2191
                                64   A6        66   A7  7D 00096          MOVQ    102(HEADER), 100(R6)                       ; 2193
                                6C   A6        6E   A7  D0 0009B          MOVL    110(HEADER), 108(R6)                       ; 2195
                                70   A6        3C   A7  9A 000A0          MOVZBL  60(HEADER), 112(R6)                        ; 2196
                                74   A6        7C   A7  3C 000A5          MOVZWL  124(HEADER), 116(R6)                       ; 2197
                                78   A6        7E   A7  3C 000AA          MOVZWL  126(HEADER), 120(R6)                       ; 2198
                              DEAD   8F        40   A7  B1 000AF          CMPW    64(HEADER), #57005                         ; 2199
                                                    05  12 000B5          BNEQ    5$
                                               7C   A6  D4 000B7          CLRL    124(R6)                                    ; 2200
                                                    04  11 000BA          BRB     6$
                                7C   A6             01  D0 000BC 5$:      MOVL    #1, 124(R6)                                ; 2201
                                     50 00000000G  8F   D0 000C0 6$:      MOVL    #LBR$_NORMAL, R0                           ; 2202
                                                        04 000C7          RET                                                ; 2204

; Routine Size:  200 bytes,    Routine Base:  $CODE$ + 0C1D


; 1394          2205  1
```

```
: 1396          2206  1 %SBTTL  'LBR$SET_LOCATE';
: 1397          2207  1
: 1398          2208  1 GLOBAL ROUTINE lbr$set_locate (control_index) =
: 1399          2209  2 BEGIN
: 1400          2210  2 !++
: 1401          2211  2 !
: 1402          2212  2 ! FUNCTIONAL DESCRIPTION:
: 1403          2213  2 !
: 1404          2214  2 !       This routine turns on locate mode.
: 1405          2215  2 !
: 1406          2216  2 ! CALLING SEQUENCE:
: 1407          2217  2 !
: 1408          2218  2 !       status = LBR$SET_LOCATE (control_index)
: 1409          2219  2 !
: 1410          2220  2 ! INPUT PARAMETERS:
: 1411          2221  2 !
: 1412          2222  2 !       control_index           is the address of a longword containing the
: 1413          2223  2 !                                index returned by LBR$INI_CONTROL.
: 1414          2224  2 !
: 1415          2225  2 ! IMPLICIT INPUTS:
: 1416          2226  2 !       NONE
: 1417          2227  2 !
: 1418          2228  2 ! IMPLICIT OUTPUTS:
: 1419          2229  2 !       NONE
: 1420          2230  2 !
: 1421          2231  2 ! ROUTINE VALUE:
: 1422          2232  2 !
: 1423          2233  2 !       lbr$_libnotopn  library was not open
: 1424          2234  2 !       lbr$_illctl     illegal control block
: 1425          2235  2 !
: 1426          2236  2 ! SIDE EFFECTS:
: 1427          2237  2 !       Locate mode is turned on.
: 1428          2238  2 !
: 1429          2239  2 !--
: 1430          2240  2
: 1431          2241  2 perform (validate_ctl (..control_index));       ! Validate the control table
: 1432          2242  2
: 1433          2243  2 lbr$gl_control [lbr$v_locate] = true;          ! set locate mode
: 1434          2244  2 RETURN lbr$_normal;
: 1435          2245  1 END;               ! lbr$set_locate



                                         0FFC 00000        .ENTRY  LBR$SET_LOCATE, Save R2,R3,R4,R5,R6,R7,R8,- ; 2208
                                                                   R9,R10,R11                                  ;
                          50        04   BC  D0 00002       MOVL    @CONTROL_INDEX, R0                          ; 2241
                                     0000G 30 00006         BSBW    VALIDATE_CTL                                ;
                          10          50  E9 00009          BLBC    STATUS, 1$                                  ;
                          50     0000G  CF  D0 0000C        MOVL    LBR$GL_CONTROL, R0                          ; 2243
                     06   A0            01  88 00011        BISB2   #1, 6(R0)                                   ;
                          50 00000000G  8F  D0 00015        MOVL    #LBR$_NORMAL, R0                            ; 2244
                                            04 0001C 1$:    RET                                                 ; 2245

; Routine Size:  29 bytes,    Routine Base:  $CODE$ + 0CE5
; 1436          2246  1
; 1438          2247  1 %SBTTL  'LBR$SET_MOVE';
; 1439          2248  1
; 1440          2249  1 GLOBAL ROUTINE lbr$set_move (control_index) =
; 1441          2250  2 BEGIN
; 1442          2251  2 !++
; 1443          2252  2 !
; 1444          2253  2 ! FUNCTIONAL DESCRIPTION:
; 1445          2254  2 !
; 1446          2255  2 !       This routine turns off locate mode and leaves library in move mode.
; 1447          2256  2 !
; 1448          2257  2 ! CALLING SEQUENCE:
; 1449          2258  2 !
; 1450          2259  2 !       status = LBR$SET_MOVE (control_index)
; 1451          2260  2 !
; 1452          2261  2 ! INPUT PARAMETERS:
; 1453          2262  2 !
; 1454          2263  2 !       control_index           is the address of a longword containing the
; 1455          2264  2 !                                index returned by LBR$INI_CONTROL.
; 1456          2265  2 !
; 1457          2266  2 ! IMPLICIT INPUTS:
; 1458          2267  2 !       NONE
; 1459          2268  2 !
; 1460          2269  2 ! IMPLICIT OUTPUTS:
; 1461          2270  2 !       NONE
; 1462          2271  2 !
; 1463          2272  2 ! ROUTINE VALUE:
; 1464          2273  2 !
; 1465          2274  2 !       lbr$_libnotopn  library was not open
; 1466          2275  2 !       lbr$_illctl     illegal control block
; 1467          2276  2 !
; 1468          2277  2 ! SIDE EFFECTS:
; 1469          2278  2 !       Move mode is turned on.
; 1470          2279  2 !
; 1471          2280  2 !--
; 1472          2281  2
; 1473          2282  2 perform (validate_ctl (..control_index));       ! Validate the control table
; 1474          2283  2
; 1475          2284  2 lbr$gl_control [lbr$v_locate] = false;         ! set locate mode
; 1476          2285  2 RETURN lbr$_normal;
; 1477          2286  1 END;               ! lbr$set_move



                                        0FFC 00000          .ENTRY  LBR$SET_MOVE, Save R2,R3,R4,R5,R6,R7,R8,R9,-; 2249
                                                                    R10,R11                                         ;
                          50       04   BC  D0 00002          MOVL    @CONTROL_INDEX, R0                              ; 2282
                                     0000G  30 00006          BSBW    VALIDATE_CTL                                    ;
                          10             50  E9 00009          BLBC    STATUS, 1$                                      ;
                          50    0000G    CF  D0 0000C          MOVL    LBR$GL_CONTROL, R0                              ; 2284
                     06   A0             01  8A 00011          BICB2   #1, 6(R0)                                       ;
                          50 00000000G   8F  D0 00015          MOVL    #LBR$_NORMAL, R0                                ; 2285
                                             04 0001C 1$:      RET                                                     ; 2286

; Routine Size:  29 bytes,    Routine Base:  $CODE$ + 0002
; 1478          2287  1
; 1480          2288  1 %SBTTL  'LBR$RET_RMSSTV';
; 1481          2289  1
; 1482          2290  1 GLOBAL ROUTINE lbr$ret_rmsstv =
; 1483          2291  2 BEGIN
; 1484          2292  2 !++
; 1485          2293  2 !
; 1486          2294  2 ! FUNCTIONAL DESCRIPTION:
; 1487          2295  2 !
; 1488          2296  2 !       This routine returns the RMS status value.
; 1489          2297  2 !
; 1490          2298  2 ! CALLING SEQUENCE:
; 1491          2299  2 !
; 1492          2300  2 !       status = LBR$RET_RMSSTV ()
; 1493          2301  2 !
; 1494          2302  2 ! INPUT PARAMETERS:
; 1495          2303  2 !       NONE
; 1496          2304  2 !
; 1497          2305  2 ! IMPLICIT INPUTS:
; 1498          2306  2 !       NONE
; 1499          2307  2 !
; 1500          2308  2 ! IMPLICIT OUTPUTS:
; 1501          2309  2 !       NONE
; 1502          2310  2 !
; 1503          2311  2 ! ROUTINE VALUE:
; 1504          2312  2 !       the contents of lbr$gl_rmsstv
; 1505          2313  2 !
; 1506          2314  2 ! SIDE EFFECTS:
; 1507          2315  2 !       NONE
; 1508          2316  2 !
; 1509          2317  2 !--
; 1510          2318  2 RETURN .lbr$gl_rmsstv;
; 1511          2319  1 END;              ! lbr$ret_rmsstv



                                                0000 00000         .ENTRY  LBR$RET_RMSSTV, Save nothing                 ; 2290
                            50     0000G  CF  D0 00002          MOVL    LBR$GL_RMSSTV, R0                            ; 2318
                                                  04 00007          RET                                                  ; 2319

; Routine Size:  8 bytes,    Routine Base:  $CODE$ + 0D1F


; 1512          2320  1
; 1514          2321  1 %SBTTL  'read_n_map_blocks'
; 1515          2322  1
; 1516          2323  1 ROUTINE read_n_map_blocks ( vbn, addr, numblocks ) =
; 1517          2324  1 !++
; 1518          2325  1 !
; 1519          2326  1 !       This routine reads up to 10 blocks from the DCX
; 1520          2327  1 !       map for this library file.
; 1521          2328  1 !
; 1522          2329  1 !       Inputs:
; 1523          2330  1 !               vbn     = block number at which to begin
; 1524          2331  1 !                               reading from file
; 1525          2332  1 !               addr    = longword to receive address of blocks
; 1526          2333  1 !               numblocks = number of blocks to read
; 1527          2334  1 !
; 1528          2335  1 !       Outputs:
; 1529          2336  1 !               addr = address of blocks read
; 1530          2337  1 !
; 1531          2338  1 !--
; 1532          2339  1
; 1533          2340  2 BEGIN
; 1534          2341  2
; 1535          2342  2 BIND
; 1536          2343  2     context = .lbr$gl_control [lbr$l_ctxptr]: BBLOCK,
; 1537          2344  2     rab     = .context [ctx$l_recrab]       : BBLOCK;
; 1538          2345  2
; 1539          2346  2 LOCAL
; 1540          2347  2     status;
; 1541          2348  2
; 1542          2349  2 perform ( get_mem ( lbr$c_pagesize * .numblocks, rab[rab$l_ubf]));
; 1543          2350  2 rab[rab$w_usz] = lbr$c_pagesize * .numblocks;
; 1544          2351  2 rab[rab$l_bkt] = .vbn;
; 1545          2352  2
; 1546          2353  2 status = $READ ( RAB = rab );
; 1547          2354  2
; 1548          2355  2 IF NOT .status
; 1549          2356  2 THEN
; 1550          2357  2     lbr$gl_rmsstv = .rab [rab$l_stv];
; 1551          2358  2
; 1552          2359  2 .addr = .rab [rab$l_rbf];
; 1553          2360  2
; 1554          2361  2 RETURN .status;
; 1555          2362  2
; 1556          2363  1 END;


                                                                  .EXTRN  SYS$READ

                                             OFFC 00000 READ_N_MAP_BLOCKS:
                                                                  .WORD   Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11          ; 2323
                          50      0000G  CF  D0 000C2             MOVL    LBR$GL_CONTROL, R0                            ; 2343
                          50         0E  A0  D0 00007             MOVL    14(R0), R0                                    ;
                          52         0C  A0  D0 0000B             MOVL    12(R0), R2                                    ; 2344
                          51         24  A2  9E 0000F             MOVAB   36(R2), R1                                    ; 2349
       53         0C      AC             09  78 00013             ASHL    #9, NUMBLOCKS, R3                             ;
                          50             53  D0 00018             MOVL    R3, R0                                        ;
                                     0000G  30 0001B              BSBW    GET_MEM                                       ;
```

```
                                   20              50  E9 0001E         BLBC    STATUS, 2$                              ;
                          20       A2              53  B0 00021         MOVW    R3, 32(R2)                              ; 2350
                          38       A2       04     AC  D0 00025         MOVL    VBN, 56(R2)                             ; 2351
                                                   52  DD 0002A         PUSHL   R2                                      ; 2353
                   00000000G       00              01  FB 0002C         CALLS   #1, SYS$READ
                                   06              50  E8 00033         BLBS    STATUS, 1$                              ; 2355
                       0000G       CF       0C     A2  D0 00036         MOVL    12(R2), LBR$GL_RMSSTV                   ; 2357
                          08       BC       28     A2  D0 0003C 1$:     MOVL    40(R2), @ADDR                           ; 2359
                                                   04  00041 2$:        RET                                             ; 2363

; Routine Size:  66 bytes,    Routine Base:  $CODE$ + 0D27


; 1557          2364  1
; 1558          2365  0 END ELUDOM




                                                                        .EXTRN  LIB$SIGNAL

;                                   PSECT SUMMARY
;
;       Name                       Bytes                       Attributes
;
;  $OWN$                              32  NOVEC,  WRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $CODE$                           3433  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       142         1        581          00:01.0





;                                   COMMAND QUALIFIERS

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

; Size:          3433 code + 32 data bytes
; Run Time:         01:13.6
; Elapsed Time:     02:27.0
; Lines/CPU Min:      1927
; Lexemes/CPU-Min: 21986
; Memory Used:  598 pages
; Compilation Complete
```

0199 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



OLDLIB
LIS

OPENCLOSE
LIS

OUTPUTHLP
LIS

LBRMSG
LIS