


```

MM          MM      DDDDDDDDD      LL
MM          MM      DDDDDDDDD      LL
MMM        MMM      DD              DD  LL
MMM        MMM      DD              DD  LL
MM  MM     MM       DD              DD  LL
MM  MM     MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DD              DD  LL
MM          MM       DDDDDDDDD      LLLLLLLLLLLL
MM          MM       DDDDDDDDD      LLLLLLLLLLLL

```


LBR

Version 'V04-000'

* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY *
* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS. *
* ALL RIGHTS RESERVED. *

* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED *
* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE *
* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER *
* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY *
* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY *
* TRANSFERRED. *

* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE *
* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT *
* CORPORATION. *

* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS *
* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL. *

MODIFIED BY:

V03-003 JWT0086 Jim Teague 11-Jan-1983
Add new sanity id for DCX data-reduced libraries.

V03-002 JWT0062 Jim Teague 25-Oct-1982
Add expand/compress descriptor to context block.

V03-001 JWT0056 Jim Teague 20-Sep-1982
Add structures for implementation of DCX data
reduction and expansion.

V02-017 PCG0005 Peter George 07-Jan-1982
Add flag bit CTX\$V_OUTPUTHLP.
Update major and minor ids.

V02-016 PCG0004 Peter George 10-Dec-1981
Correct length calculations.

V02-015 PCG0003 Peter George 10-Dec-1981
Add HPD\$L_LENARRAY.

V02-014 RPG0014 Bob Grosso 02-Dec-1981
Support adjacent allocation of cache header blocks.

V02-013 RPG0013 Bob Grosso 14-Aug-1981
Support lower case keywords

V02-012 PCG0002 Peter George 10-Aug-1981

Correct the value of the HLP\$C_FACILITY constant.

```

V02-011      RPG0041      Bob Grosso      30-Jul-1981
Add sanity code for V3 libs.

V02-010      RPG0038      Bob Grosso      17-Jul-1981
Add padding in library header.

V02-009      RPG0037      Bob Grosso      18-Jun-1981
Change lhd$l_*luhrec to lhd$w_*luhrec.

V02-008      RPG0036      Bob Grosso      12-Jun-1981
Bump minorid for /HIST changes

V02-007      PCG0001      Peter George    08-May-1981
Add LBR$OUTPUT_HELP definitions.

V02-006      RPG0035      Bob Grosso      20-Apr-1981
Replace lhd$l_updhis with lhd$w_closererror and lhd$w_spareword.
Remove luh$c_op_*.

V02-005      RPG0032      Bob Grosso      10-Apr-1981
More definitions for LUH.

V02-004      RPG0027      Bob Grosso      26-Mar-1981
Add library update history definition.

V02-003      RPG0017      Bob Grosso      26-Feb-1981
Add idx$c_rfaplsbyt, the length in an index entry of the
rfa plus the keyname length byte.
Add idd$v_varlenidx to flag variable length keyname storage.

V02-002      RPG0011      Bob Grosso      19-Jan-1981
Add hivbn to context block to record the highest VBN in
the cache.

V02.01      HJ0003      Herb Jacobs      19-Aug-1980
Fix syntax of V(field) definition of help flags

```

Librarian internal context control block

\$STRUCT CTX

```

F      ISI,W      : ISI for library when open
F      IFI,W      : IFI for library when open
F      CTLFLG,L   : Control flags
V      <M
LIBOPN      : Library is open
LKPDON      : Lookup was done
REPROG      : Replace in progress
HDRDIRTY    : Header has been modified
MHDOUT      : Module header needs to be written
OLDLIB      : Library is old (VMS R1) format
FOUND1      : Found match in LBR$GET_INDEX
RONLY       : Library is read only

```



```

OUTPUTHLP      : LBR$GET HELP treats "HELP" keyword as normal keyword
                : Only LBR$OUTPUT_HELP sets and clears this flag
>
F  CACHE,L      : Listhead of index cache list
F  RECRAB,L     : RAB address for record I/O
F  RPHASH,L     : Pointer to hash table for deleted symbols
F  RPLDESC,L    : String descriptor for keyname
F  ,L          : of key being replaced
F  RPNEWTXT,B,6 : RFA of new text for replaced key
F  DLTXRFA,B,6  : RFA of text to delete (replace)
F  EOMODRFA,B,6 : RFA of end of module (VMS R1 library)
F  READRFA,B,6  : RFA of next LBR$GET_RECORD
F  READBUF,L    : Address of internal record buffer
F  RDBUFR,L     : Address of block buffer
F  RDVBN1,L     : VBN of first block in buffer
F  RDBLKS,L     : Number of blocks in the buffer
F  NXTPUTRFA,B,6 : RFA for next sequential PUT
F  ,W          : Spare
F  HIVBN,L      : Highest VBN in cache
F  CHDALLSIZ,L  : size in bytes left in block for cache header entry allocation
F  CHDALLADR,L  : address of block for cache header entry allocation
C  <
  CHDALLBLK,7   : Number of blocks for initial allocation
>
F  DCXCTX,L     : DCX context longword
F  DCXMAPDSC,L  : Pointer to DCX map descriptor
F  DCXRECDSC,L  : Descriptor for expanding/compressing
F  ,L          : Second lword of rec desc
F  ,L,9         : Spares
L  LENGTH
E

```

```

: Library header (VBN 1 of the library)
:

```

\$STRUCT LHD

```

F  TYPE,B       : Type of library
F  NINDEX,B     : Number of indices
F  ,W,1         : Reserved word
F  SANITY,L     : ID for sanity check
C  <
  SANEID,123454321 : ID for sanity check
  SANEID3,233579905 : ID for sanity check for V3 libraries
  SANEIDC,319232342 : ID for sanity check for DCX libraries
>
F  MAJORID,W    : Library format level major id
F  MINORID,W    : Library format level minor id
C  <
  MAJORID,3      : Major id level
  MINORID,0      : Minor id level
                : Changed to 2 for Update History Changes
>
F  LBRVER,T,32  : ASCII version of librarian
                : that created library

```

```

F      CREDAT,L      ; Creation date/time
F      .L
F      UPDTIM,L      ; Date/time of last update
F      .L
F      MHDUSZ,B,1    ; Size in bytes of additional
                        ; module header data
F      IDXBLKF,W      ; Number of disk blocks in index segment
F      .B,1
F      CLOSEROR,W     ; Spares
C      <
      CORRUPTED,57005 ; Toggle during library close to trap an interrupted write.
      >
      ; "XDEAD"
F      SPAREWORD,W    ; Spare
F      FREEVBN,L      ; VBN of 1st deleted block
F      FREEBLK,L      ; Number of deleted blocks
F      NEXTRFA,B,6    ; Next free spot for data puts
F      NEXTVBN,L      ; Next free VBN for alloc_block
F      FREIDXBLK,L    ; Number of free pre-allocated index blocks
F      FREEIDX,L      ; Listhead for pre-allocated index blocks
F      HIPREAL,L      ; VBN of highest pre-allocated index block
F      HIPRUSD,L      ; VBN of highest pre-allocated block in use
F      IDXBLKS,L      ; Number of index blocks in use
F      IDXCNT,L       ; Number of index entries (total)
F      MODCNT,L       ; Number of entries in index 1 (module names)
F      .W
F      MODHDSR,L      ; Spare
F      IDXOVH,L        ; Number of module headers in library
F      MAXLUHREC,W     ; Number of overhead index pointers
F      ; Max number of update history records.
F      ; If zero then lib won't have history
F      ; Count of history records.
F      ; RFA of beginning of history
F      ; RFA of end of library update history
F      ; VBN of DCX map (if in reduced format)
F      .L,13
F      IDXDSC          ; Spares
E      ; Start of index descriptors

```

```

:
: Index descriptors in header
:

```

\$STRUCT IDD

```

F      FLAGS,W        ; Flags longword
V      <M
      ASCII           ; Keys are ASCII if 1
      LOCKED          ; Index is locked from modification
      VARLENIDX        ; Index entries have variable length
      NOCASECMP        ; Do not upcase match keyword on search compare
      NOCASENTR        ; Don't upcase keyword on entry
      UPCASNTRY        ; Upcase the index entry when comparing against a match keyword
      >
F      KEYLEN,W        ; Total length of key
F      VBN,L           ; VBN of first block of index
E      LENGTH

```



```

;
; Index block structure
;

```

\$STRUCT INDEX

```

F      USED,W      ; Total bytes in use
F      PARENT,L    ; VBN of parent index
F      ,B,6        ; Reserved
L      ENTRIES     ; Start of index entries
F      ,B,500      ; (Index entry space)
C      <
          BLKSIZ,500 ; Maximum useable index space
          >
L      LENGTH      ; Length of entire block (512)
E

```

```

;
; Entry in an index (binary)
;

```

\$STRUCT IDX

```

F      RFA,B,6      ; RFA of text (or index)
S      VBN,0,L      ; VBN of RFA
S      OFFSET,4,W   ; Offset to byte of RFA
L      LENGTH
M      1
F      KEYID,L      ; Binary keyID

```

```

;
; Entry in an index (ASCII)
;

```

```

P      1
F      KEYLEN,B     ; Length of ASCII keyname
L      RFAPLSBYT    ; Length of rfa plus the length byte
F      KEYNAME,T,0  ; ASCII keyname
E

```

```

;
; Format of RFA disk pointer
;

```

\$STRUCT RFA

```

F      VBN,L        ; Virtual block number in file
F      OFFSET,W     ; Byte offset within block
C      <
          INDEX,65535 ; Offset = FFFF indicate index
          >
L      LENGTH      ; Length of RFA pointer
E

```

```

;
; Data block structure
;

```

end

end

modu

```

/*
/*
/*

```

```

cons
cons
cons
cons
cons
cons
cons

```

```

cons
cons
/*
/*
/*

```

```

;
$STRUCT DATA
F      RECS,B      ; Number of records in this block
F      ,B          ; Spare
F      LINK,L      ; Link to next block
L      DATA       ; Start of data area
F      ,B,506      ; Data area
L      LENGTH      ; Length of data block
E

; Disk block cache list entry
;
$STRUCT CACHE
F      LINK,L      ; Link to next entry or 0
F      VBN,L       ; VBN of index block
F      ADDRESS,L   ; Address of block in memory
F      FLAGS,W     ; Flags
V      <           ; True if this block modified
      DIRTY        ; Data block as opposed to index block
      DATA
      >
L      LENGTH      ; Length of one entry
E

; Cache entry for replace key list
;
$STRUCT RKB
F      LINK,L      ; Link in hash bucket
F      INDEX,L     ; Index number symbol is in
F      KEYID,L     ; ID of key (binary keys)
F      KEYLEN,W    ; Length of keyname (ASCII keys)
F      ,W          ; Upper word of first lw of descriptor
F      KEYADDR,L   ; Pointer to KEYNAME
F      KEYNAME,T,0 ; Start of key name
L      LENGTH      ; Length of fixed part of block
E

; Update history list
;
$STRUCT LUH
M      1           ; VBN of next block or 0 if last
F      NXTLUHBLK,L ; spare word
F      SPARE,W     ; Begin data area for LUH records
L      DATA       ; Room for LUH records
F      ,B,506      ; length of a block for LUH storage
L      LENGTH
P      1

```

cons
cons/*
/* L
/*cons
cons
cons
cons
cons
cons
conscons
cons
cons
cons
cons
cons
cons
cons
cons

cons

/*
/* L
/*

aggr

/*
/* L
/*

end

aggr


```

F      RECHDR,W      ; Mark beginning of new record
F      RECLN,W       ; Length of update history record
L      RECHDRLEN     ; Size of header field
C      <
      RECHDRMRK,43962 ; %X'ABBA'
      DATFLDLEN,506   ; (512 - 6) bytes are used for data
      >              ; see LBRUSR.MDL for LBR$C_MAXLUHLEN,1024
E

```

```

; Data structures for help processing
;

```

```

$STRUCT HLP

```

```

C      <
      MAXRECSIZ,80    ; Maximum length of help record line
      >

F      HLPFLAGS,L     ; Flags for help processing
S      USERFLAGS,,W   ; Flags user will see when called to print line
V      <M
      UNOHLP          ; No help text was found
      UKEYLIN         ; Line contains keyname
      UOTHINFO        ; Line is part of other info
      >

S      LBRFLAGS,,W    ; Flags used internally by help processor
V      <M
      ,8
      ANYHELP,        ; Some help text was found (%X'1000000')
      HELPHLP,        ; Help is for 'HELP' (%X'2000000')
      KEYLINE,        ; Line contains a key (%X'4000000')
      QUALINE,        ; Line contains a qualifier (%X'8000000')
      QUALHELP,       ; Processing 'HELP KEY /QUALIFIER' (%X'10000000')
      HLPFOUND,       ; Help text was found (%X'20000000')
      ALLHELP,        ; "... was seen (%X'400000000')
      >

F      BUFDESC,L      ; Buffer descriptor
F      ,L
F      CURPTR,L       ; Pointer into buffer
F      NCHARS,L       ; Number of characters in buffer
F      CURLEVEL,L     ; Current help level
F      LASTLEVEL,L    ; Help level last looked at
F      TABINDEX,L     ; Current tab index
F      WIDTH,L        ; Width of output line
F      KEYLIST,L      ; Address of found keys descriptors
F      REALKEYS,L     ; Number of "real" keys passed
F      PMATCH,L       ; Number of key1 partial matches found
F      PMTDESC,B,8    ; String descriptor for 1st partial match
F      PMTRFA,B,6     ; RFA of 1st partial match module
F      KEY2RFA,B,6    ; RFA of 1st level 2 key within module
F      WILDFLAGS,T,8  ; Bitvector of wild flags (64 bits)
F      READSTS,L      ; Status from last read operation
F      READRFA,B,6    ; Address of current reading rfa
F      LSTKEYRFA,B,6  ; RFA of last key found

```

```

/*
/*
/*

```

end

agg

```

/*
/*
/*

```

end

agg

```

/*
/*
/*

```

end

agg

```

/*
/*
/*

```

end

agg

```

/*
/*
/*

```

end

agg

```

L      SIZE                      ; Length of block
C      <
OUTROUT,      1                  ; First parameter
OUTWIDTH,     2                  ; Second parameter
LINEDESC,     3                  ; Third parameter
LIBNAME,      4                  ; Fourth parameter
FLAGS,        5                  ; Fifth parameter
INROUT,       6                  ; Sixth parameter
PARAMS,       6                  ; Number of lbr$output_help parameters
INDENT,       2                  ; No. of characters to indent per level
MAXKEYS,     10                  ; Maximum number of keys
PROMPT,      25                  ; Number of bytes in topic prompts
MAXWIDTH,    132                 ; Maximum line width
PAGESIZE,    512                 ; Size of a page
FACILITY,    7733248            ; Facility code 118 * 65536
LISWIDTH,    80                 ; Default listing width
MAXLISWID,   256                ; Maximum listing width
KEYLOGTAB,   2                  ; Width of logical tab for listing keys as found
LOGTAB,      11                 ; Width of logical tab
>

K      <
INFO,         1                  ; Offset to info block pointer
LINEWIDTH,    2                  ; Offset to line width
USEROUT,      3                  ; Offset to user routine address
USERDATA,     4                  ; Offset to user data address
KEY1DESC,     5                  ; Offset to key1 descriptor
>

```

E

```

;
; Help control flags
;

```

\$STRUCT HCF

```

M      1
F      USERLIB,L                ; User library flags
F      PROMPT,L                 ; Prompting flags
L      LENGTH
P      1
V      <M
CONT                     ; Continue with prompting
STAY                     ; Stay at same prompting level
MORE                     ; More information is available
INFO                     ; Some information was printed
BACKUP                   ; Backing up a prompt level
>
C      <HCF,$M
NOPROMPT,-1              ; Prompting turned off
>
E

```

;

/*
/*
/*/*
/*
/*
/*
/*/*
/*
/*
end

agg

/*
/*
/*

end

agg

/*
/*
/*

end

agg

/*
/*
/*

end

agg


```

; Current library indices
;

```

\$STRUCT HLI

```

F      MAININDEX,L      ; Main library (/LIB) index
F      LASTINDEX,L      ; Last library opened index
F      LASTNUMB,L       ; Last library opened number, relative to
                        ; user specified default libraries
L      LENGTH
E

```

```

; Output driver print data
;

```

\$STRUCT HPD

```

M      1
F      SUBPMTPTR,L      ; Ptr used for filling subprompt buffer
F      SUBPMTLEV,L      ; Current prompt level
F      OUTPUTROU,L      ; User specified output routine
F      LENARRAY,L       ; Address of the keyword length array
F      TRUEKEYS,B       ; Number of help keys
F      HELPLEVEL,B      ; Current key depth
F      PRINTFLAG,B      ; Flags
L      LENGTH
P      1
V      <M
      INIT              ; Printing initiated
      ALL              ; Print all information found
      FOUND            ; Keys successfully found
      >
E

```

```

; Old format library information structure
;

```

\$STRUCT OFL

```

F      MNTVBN,L         ; VBN of start of module name table
F      MNTESIZ,L        ; size of an entry (in bytes) in MNT
F      NUMODS,L         ; Number of modules in MNT
F      MNTBLKS,L        ; Number of blocks in MNT
F      MNTPEPBLK,L      ; Number of entries per block
F      MNTADR,L         ; Address of MNT window block
F      MNTBLSZ,L        ; Size of MNT window block
F      GSTVBN,L         ; VBN of start of global symbol table
F      GSTESIZ,L        ; Size of an entry in GST
F      NUMSYMS,L        ; Number of symbols in GST
F      GSTBLKS,L        ; Number of blocks in GST
F      GSTPEPBLK,L      ; Number of entries per block
F      GSTADR,L         ; Address of GST window block
F      GSTBLSZ,L        ; Size of GST window block

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

      F      WINBVBVBN,L      ; VBN of base of current window
      F      WINTVBVBN,L      ; VBN of top of window
      F      WINBLKS,L        ; Number of blocks in window
;
; Do not change the order of 'ENT', 'RBN', 'ADR'
;
      F      TRILENT,L         ; Trial table entry's number
      F      TRILRBN,L         ; within block and relative block
                                ; within window
      F      TRILADR,L         ; Pointer to trial entry in table
      F      LOWENT,L          ; Lowest possible name entry
      F      LOWRBN,L          ; Relative block within window
      F      LOWADR,L          ; and its address within table
      F      HIENT,L           ; Highest possible name entry
      F      HIRBN,L           ; Relative block number within window
      F      HIADR,L           ; and its address within table
      L      LENGTH
E

```

```

;
; Data structure to extract information for either the GST or MNT
; of an old format library
;

```

\$STRUCT OIB

```

      F      VBN,L             ; VBN of start of index
      F      ESIZ,L            ; Size of an entry in bytes
      F      NENTS,L           ; Number of entries in index
      F      NBLKS,L           ; Number of blocks in index
      F      ENTPBLK,L         ; Number of entries in a block
      F      TBLADR,L          ; Address of window table in memory
      F      TBLSIZ,L          ; Size of window table
      L      LENGTH
E

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

/*
/*
/*

```

```

end
agg

```

```

end
end

```


