

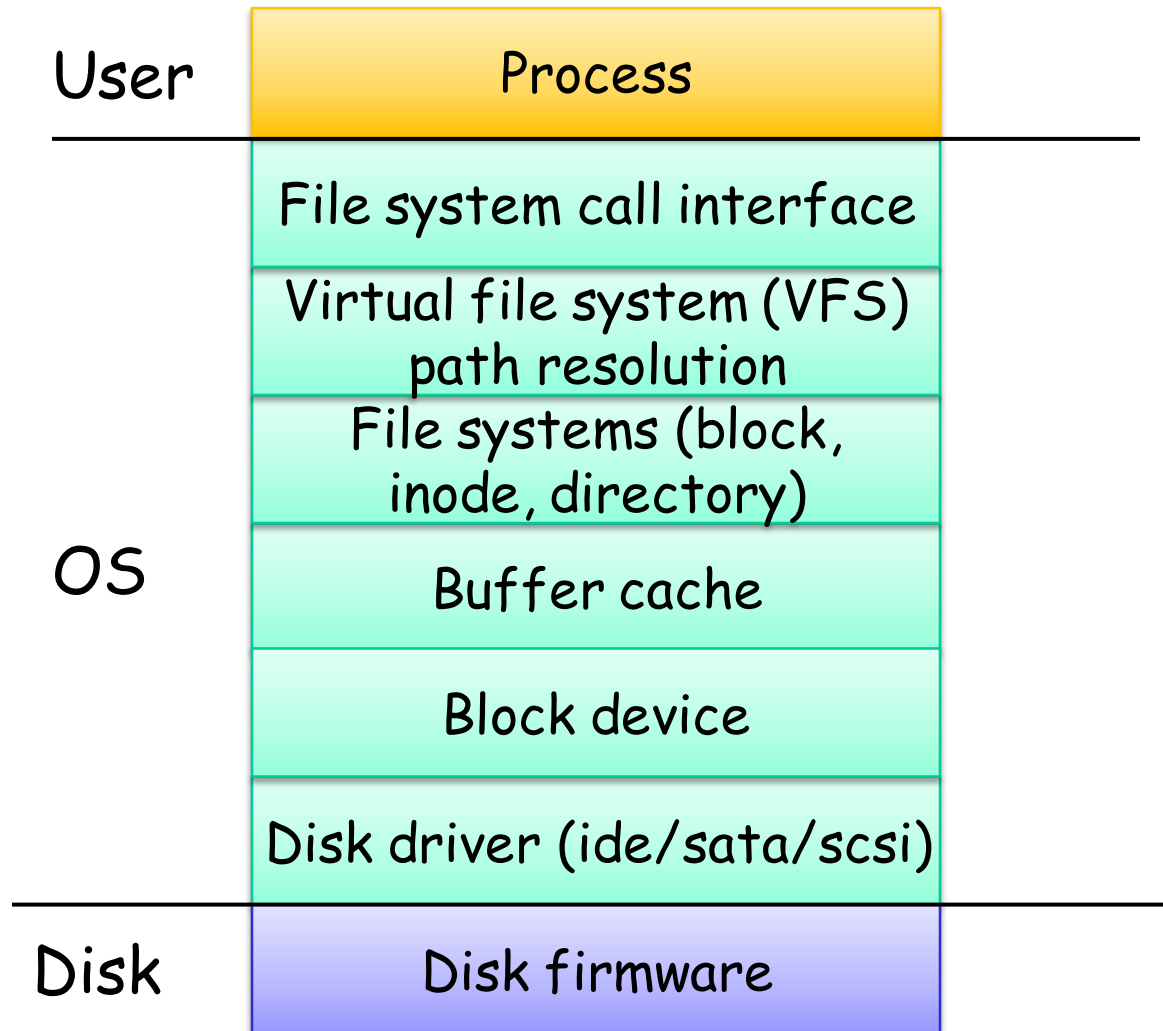
W4118: xv6 file and disk systems



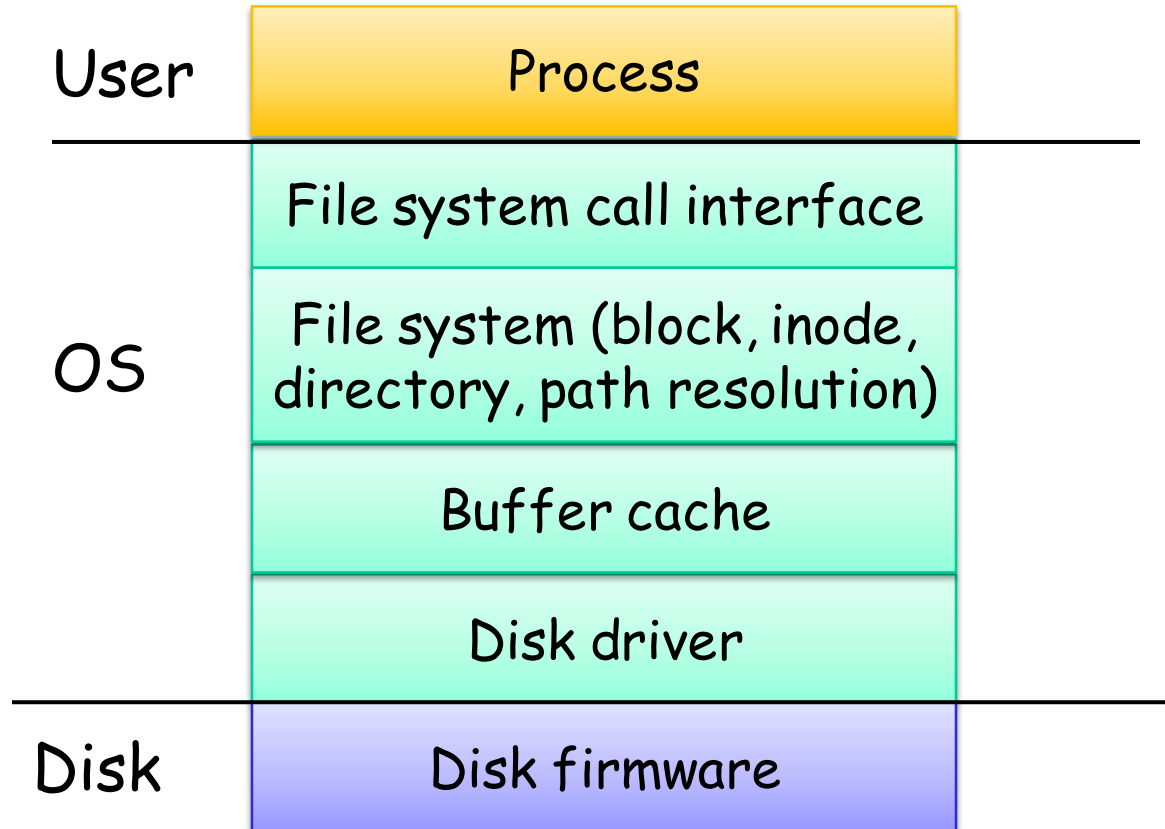
Instructor: Junfeng Yang

References: Modern Operating Systems (3rd edition), Operating Systems Concepts (8th edition), previous W4118, and OS at MIT, Stanford, and UWisc

Layered approach to storage systems



xv6 storage layers



xv6 disk driver

- ❑ `ide.c`
- ❑ `iderw(struct buf *b)`: read or write disk sector
- ❑ `idestart(struct buf *b)`: start request for b
- ❑ `ideintr()`: ide interrupt handler
- ❑ `ideinit()`: ide initializer

xv6 buffer cache

- ❑ `bio.c`
- ❑ *struct buf*
 - flags: `B_BUSY`, `B_VALID`, `B_DIRTY`
- ❑ *struct bcache*
 - head: LRU list of cached blocks
- ❑ `bread()`: read disk sector and return buffer
- ❑ `bwrite()`: write buffer to disk sector
- ❑ `bget()`: look up buffer cache for sector and set busy flag
- ❑ `brelease()`: clear busy flag and move buffer to head
- ❑ `binit()`: initialize buffer cache

xv6 buffer cache locking

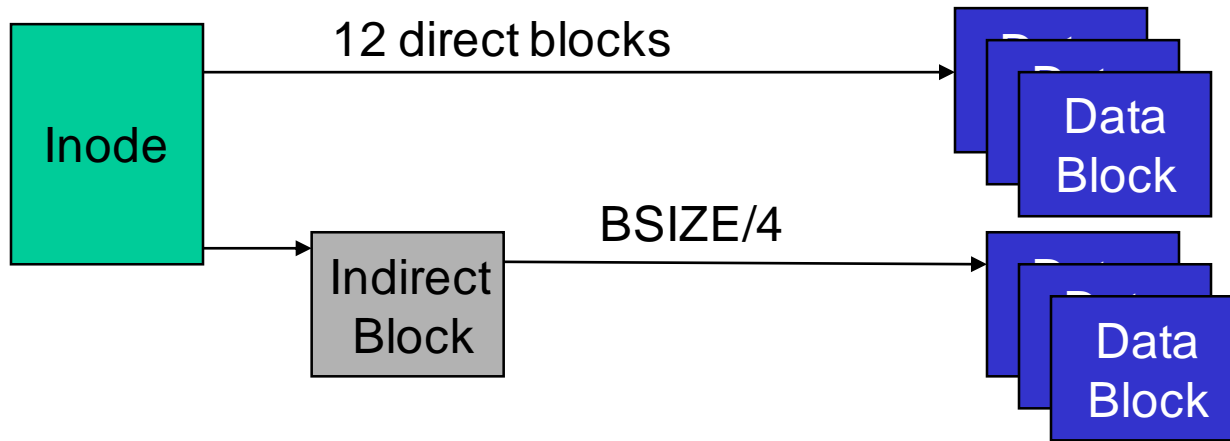
- `bcache.lock`: lock for entire buffer cache
- `b->flags & B_BUSY`: busy bit for each buffer
 - Why flag? Hold no spinlock for disk access
- Ensures that only one process can be touching a *struct buf* at any time

xv6 file system layout



- ❑ `fs.h, fs.c, mkfs.c`
- ❑ *struct superblock*

xv6 file and directory layout



- ❑ $\text{NDIRECT} = 12$
- ❑ $\text{NINDIRECT} = \text{BSIZE}/4 = 128$
- ❑ *struct dinode* in *fs.h*, *struct inode* in *file.h*
- ❑ *struct dirent* in *fs.h*

xv6 block operations

- ❑ `readsb()`: read on-disk super block into in-mem super block
- ❑ `bzero()`: zero a block
- ❑ `balloc()`: allocate a block, set bitmap
- ❑ `bfree()`: free a block, clear bitmap

xv6 inode operations

- ❑ `bmap()`: map data block number to disk block number
- ❑ `itrunc()`
- ❑ `ialloc()`: allocate a new inode
- ❑ `iupdate()`

xv6 inode synchronization operations

- ❑ `iget()`: find in-memory inode from inode cache and bump reference count
- ❑ `idup()`: bump reference count
- ❑ `iput()`: decrement reference count and truncate inode if necessary
- ❑ `ilock()`: lock inode for read and write by setting `I_BUSY` flag
- ❑ `iunlock()`: unlock inode by clearing `I_BUSY` flag; must call `iunlock()` before `iput()`

xv6 file system calls

- `file.c, sysfile.c`
- Examples file system calls
 - `sys_open()`
 - `sys_mkdir()`
- Path resolution
 - `namei()`
 - `nameiparent()`