Section 8: Intro to Lab 5

CSE 451 19WI

Lab 5

Three Parts:

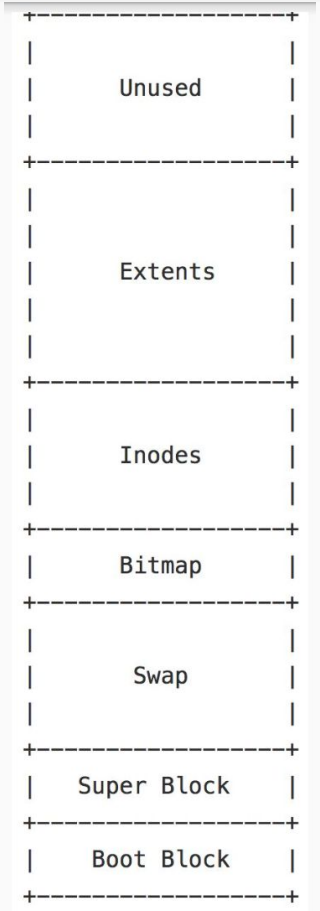
A. Enable file creation, writes, and appends

B. Enable concurrency for part A.

C. Make the file system crash-safe

Part A: Create, Write & Append

XK Disk Format



- Boot Block
  - Used by the boot loader
- Super Block
  - Describes how the disk is formatted
- Swap
  - Used for paging
- Bitmap
  - Keeps track of which blocks are free/used
- Inodes
  - Inode table holds an inode for each file (inode holds file metadata)
- Extents
  - Where file data is stored

See lab5.md for the disk diagram with block offsets included

struct dinode - inc/fs.h

```
25  // On-disk inode structure
26  struct dinode {
27    short type;           // File type
28    short devid;          // Device number (T_DEV only)
29    uint size;            // Size of file (bytes)
30    struct extent data; // Data blocks of file on disk
31    char pad[46];         // So disk inodes fit contiguosly in a block
32  };
```

struct extent - inc/extent.h

```
// represents a contiguous block on disk of data
struct extent {
  uint startblkno; // start block number
  uint nblocks;    // n blocks following the start block
};
```

struct dinode - inc/fs.h

```
25  // On-disk inode struc
26  struct dinode {
27    short type;
28    short devid;
29    uint size;
30    struct extent data;
31    char pad[46];
32  };
```

Why is there padding?

`struct dinode` - `inc/fs.h`

```
25  // On-disk inode struc
26  struct dinode {
27    short type;          2+
28    short devid;         2+
29    uint size;           4+
30    struct extent data;  8+
31    char pad[46];        46
32  };                     62
```

```
// represents a contiguous block on disk of data
struct extent {
  uint startblkno; // start block number
  uint nblocks;    // n blocks following the start block
};
```

Size should be a power of 2 to ensure no dinode is split across a page

Sizeof evaluates to **64 bytes**, due to padding (2 bytes at end)

`struct inode` - inc/file.h

```
6  // in-memory copy of an inode
7  struct inode {
8    uint dev;  // Device number
9    uint inum; // Inode number
10   int ref;   // Reference count
11   struct sleeplock lock;
12
13   short type; // copy of disk inode
14   short devid;
15   uint size;
16   struct extent data;
17 };
```

If you modify **`struct dinode`**, make sure to update **`struct inode`** as well!

Write

- Modify **`writei`** in kernel/fs.c so an inode can be used to write to disk
- Use **`bread, bwrite, brelse`**
- See **`readi`** for an example
- Also, change **`open`** to allow **`O_RDWR`**

Append

- Need to be able to extend the size of a file
- Allocate additional space using extra block pointers or extra extent pointers

Example: Need to be able to handle the case where the user tries to append to File 1 when the disk's extent region is laid out as follows.

Create

- Create a new file when **O_CREATE** is passed to **open**

“You need to create a empty inode on disk, change the root directory to add a link to the new file, and (depending on your disk layout) change bitmap on disk. The inode file length itself will change, so don't forget to update this as well.”

Note: File deletion is not required

Debugging Tips

- Use **make clean** && **make** if unsure about disk correctness
- Make sure to update in memory inodes as well as on disk data structures
- Use the fsck tool (coming soon). Or implement it yourself!
- Use **ilock** and **iunlock** and other wrappers for Part B.