Typedef struct filestatus\_t {

char name[MAX\_FILENAME\_BYTE];

int inode;

int mode;

int size;

int index\_in\_block;

} filestatus\_t; //128 bytes

Struct for a file in root directory,

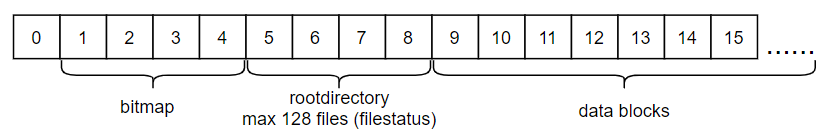
Name: file name

Inode: Point to fcb table

Mode: appending/reading/nothing

Sise: size of file

Index\_in\_block: filestatus size is 128 bytes, each block has 32 filestatus, index\_in\_block from 0-31



Each block in data blocks has a sec\_len field to mark current block length.

Algorithm:

Use block 1,2,3,4 to store filestatus struct(128bytes)

Each filestatus has Inode field to point to block that has 4kB/4bytes = 1kB indexes, these indexes point to blocks that store its data (mean 1kB\*4kB = 4MB per file).

Sys\_create: set name in filestatus and find an empty block where inode field need to point to.

Sys\_open: find a created file by inode, then write mode into mode field in filestatus.

Sys\_close: clear mode in filestatus.

Sys\_read: open inode block and read all data in blocks that inode point to.

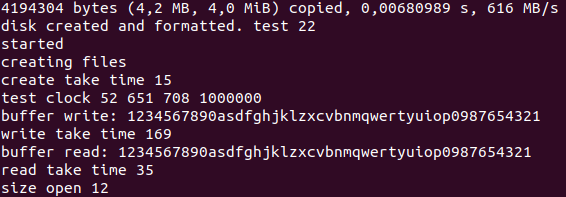
Sys\_append: open inode block and write more data in free blocks, then set inode field point to new blocks.

Sys\_delete: delete all data that file store and its information.

Result:

Maximum size can be created: 2^31

size 2^22 = 4,2 MB



Create maybe not much different when I increase size. But read/write time takes more little time when increasing size.