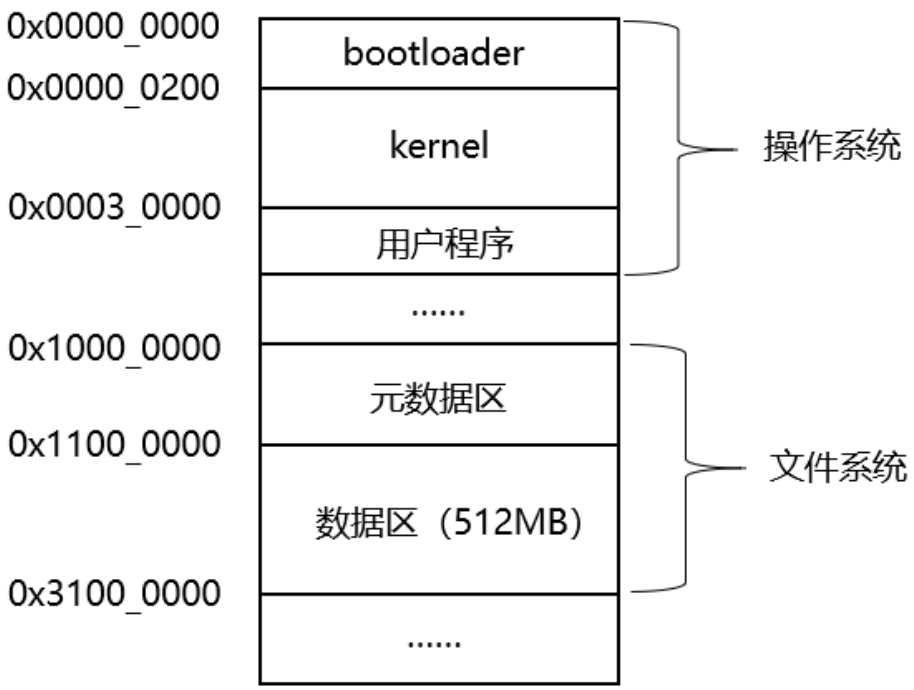
Project 6 File System 设计文档

中国科学院大学
贾志杰
2021 年 1 月 11 日

1. 文件系统初始化设计

（1）请阐述你设计的文件系统对磁盘的布局（可以使用图例表示），包括从磁盘哪个位置开始，superblock、inode map、block 或 sector map、inode table 以及数据区各自占用的磁盘空间大小

磁盘的布局如下图所示：



文件系统从 0x10000000 处开始，元数据区占 16MB，其中依次是占 1 个 sector 的 superblock、占 1 个 sector 的 inode map、占 32 个 sector 的 block map、占 512 个 sector 的 inode table。数据区大小 512MB，从 0x11000000 处开始。

（2）请列出你设计的 superblock 和 inode 数据结构，并阐明各项含义。请说明你设计的文件系统能支持的最大文件大小，最多文件数目，以及单个目录下能支持的最多文件/子目录数目。

superblock 数据结构如下：

```
typedef struct super_block
{
    uint32_t magic;
    // seg0: offset is sector offset
```

```
    uint32_t start_sector;
    uint32_t total_sector;
    uint32_t sector_used;

    uint32_t inode_map_offset;
    uint32_t inode_map_size;
    uint32_t inode_used;

    uint32_t block_map_offset;
    uint32_t block_map_size;
    uint32_t block_used;

    uint32_t inode_offset;
    uint32_t inode_array_size;
    // seg1: offset is block offset
    uint32_t data_start_block;
    uint32_t data_total_block;

    uint32_t inode_entry_size;
    uint32_t dir_entry_size;

} super_block_t;
```

首先是 magic，标识文件系统。然后第一部分是对元数据区的描述，元数据区在磁盘开始的 sector 号，总共的 sector 数量，已经使用的 sector 数量。然后是 inode map 在元数据区内的偏移（以 sector 计数），inode map 大小，inode 已经使用的数量。之后是 block map 的偏移、大小和已经使用的数量。最后是 inode array 的偏移和大小。第二部分是对数据区的描述，包括数据区在磁盘的开始块号和总共块数量。最后是 inode entry 和 dir entry 这两个数据结构的大小。

inode 数据结构如下：

```
typedef struct inode_entry
{
    uint32_t size;
    uint32_t type;
    uint32_t mode;
    uint32_t leaf;
    uint32_t direct_block[8];
    uint32_t indirect_block[4];

} inode_entry_t;
```

inode 包括 4 个描述该 inode 所指的文件或目录属性的变量，以及 8 个直接指针和 4 个间接指针。size 表示 inode 所指文件或目录的大小，对于目录他表示一个目录项的大小。type 表示该 inode 所指的是一个文件、目录或者链接。mode 表示文件的操作权限，当 type 为 FILE 时有效。leaf 该 inode 所指的文件或目录是否是一个叶子节点。

inode 中有 8 个直接指针、4 个 1 级间接指针。因此最大文件大小=

8*4KB+4*4MB=16MB32KB。文件数量取决于 inode 数，inode 数为 4096，因此最多支持 4096 个文件。单个目录下可支持 12 个文件/子目录，但由于“.”和“..”目录占用 2 个位置，实际最多容纳 10 个文件/子目录。

2. 文件操作设计

（1）请说明创建一个文件所涉及的元数据新增和修改操作，例如需要新增哪些元数据，需要修改哪些元数据

创建一个文件需要新增一个 inode、一个数据块（初始为 0）。需要修改父目录的 inode、目录项，还需要在 imap、block map 里修改新增 inode、数据块对应的位，最后还需修改超级块中 inode 使用数、数据块使用数等信息。

（2）设计或实现过程中遇到的问题和得到的经验

对于文件的大小，一开始是以文件占用的数据块数量计数，这样就存在一个问题，文件实际写入范围可能不足一个完整块，在读出时可能越界但判定不出，因此改为以字节计数。

3. 目录操作设计

（1）请说明文件系统执行 ls 命令查看一个绝对路径时的操作流程

①解析路径，以 / 开头是绝对路径，从 root_dir_entry 开始寻找；否则是相对路径，从 current_dir_entry 开始寻找。划分各级目录名，并检查路径是否合法（但不检查各目录是否存在）。

②从第一级目录开始，检查目录是否存在，如果某一级不存在返回错误；如果存在读出该目录的 ino，查找对应 inode，根据 inode 中第一个直接指针，找到目录项所在磁盘块。读出该块，在该目录项中寻找下一级目录。重复这一过程直到找到最后一级目录的目录块。如果最后一级是文件，那么就不用找目录块了，直接打印该文件的 inode 的信息，然后返回。

③根据最后一级目录的目录块，打印出其子目录名字。在得到子目录名字的同时，我们也知道了子目录的 ino，可以继续在 inode 中找到并打印子目录的更详细的信息。

参考文献

[1] MIPS® Architecture For Programmers Volume II-A: The MIPS64® Instruction Set Reference Manual, Revision 6.05, 2016