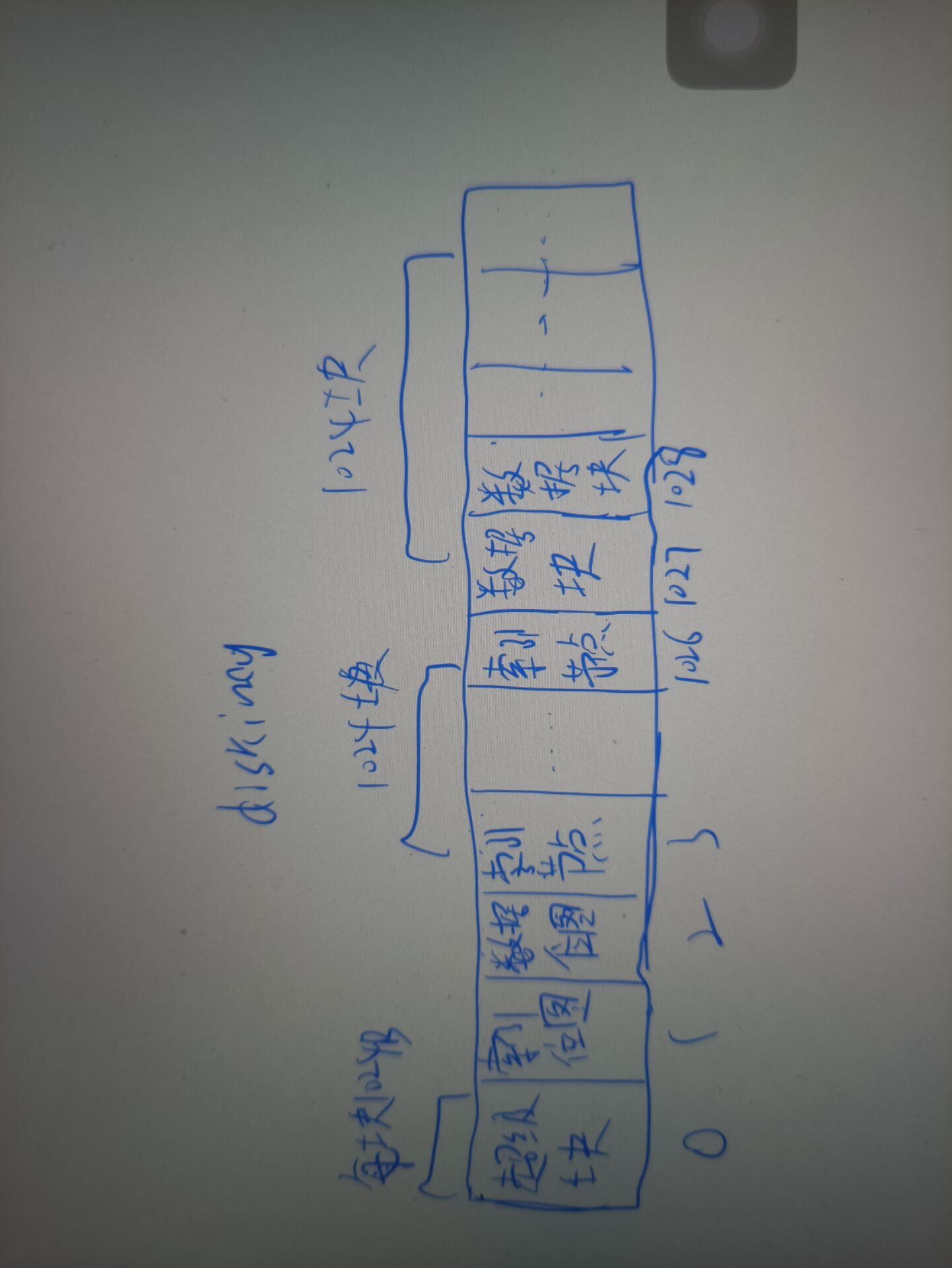
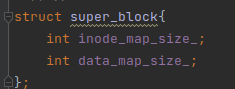
**用户态文件系统**

**1程序主要设计思路、实现方式**

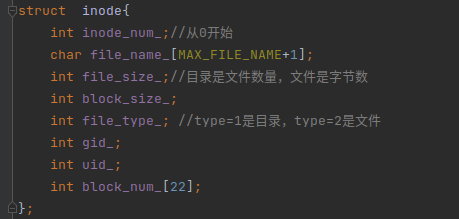
基于磁盘的用户态文件系统，如的磁盘镜像类似于ext2的设计。如上图。



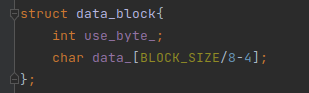
第1块是超级块，它记了inode位图的比特数和data位图的比特数。

第2块是索引位图，在我的设计里面他是占1024bit，也就是刚好一块的大小，可以在宏自定义。

第3块是数据位图，在我的实际里面它是占1024bit也就是刚好一块的大小，可以在宏自定义。

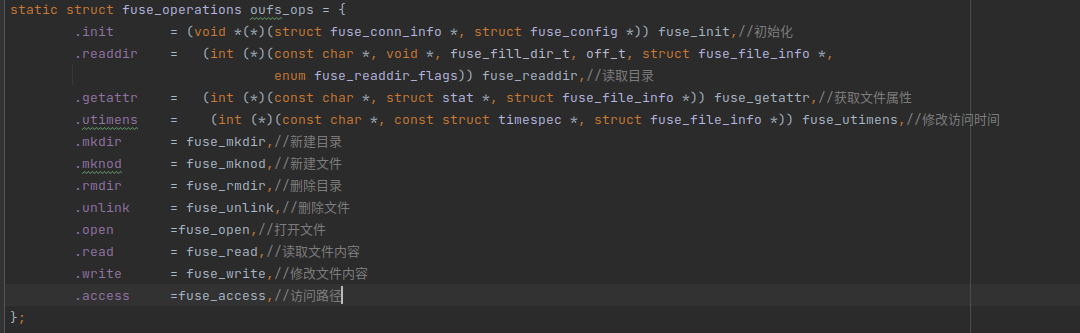


第4块是到1026块是索引节点，索引节点的块数等于索引位图的字节数。如上图所示，每个索引节点刚好是1024bit，索引节点依次记录了inode号，文件名字，文件大小，实际文件占的块数，文件类型，组id，用户id，数据节点的块号数组。



第1028-最后是数据块，数据块的快熟等于数据位图的字节数。如上图所示，每个数据块的大小刚好收1024bit，如果这个数据块是属于目录文件的，那么在data\_中记录的就是目录下的文件的inode号。如果这个数据块是属于文本文件的，那么在data\_中记录的就是文件的实际内容。Use\_byte\_是记录在data块中用了多少个字节。

**2程序的模块划分，及对每个模块的说明**



大概实现了以上的操作

1. 初始化模块

初始化，如果没有磁盘镜像文件那么就创建磁盘镜像并且初始化，如果有磁盘镜像文件，就把超级块读取出来。

1. 读取目录

首先读取此路径下的inode，如果不存在这个路径，那么返回。然后判断这个inode节点的文件是不是一个目录文件，如果不是目录文件那么返回。如果这个文件是一个目录文件那么就把这个文件名保存在buf中。

1. 获取文件信息

首先读取此路径下的inode，如果不存在这个路径，那么返回。如果这个inode节点是一个目录文件，那么就把文件的mode设置为目录，并且通过inode中保存的信息设置文件的信息。如果这个inode是一个文本文件，那么就把文件的mode设置为文件，并且通过inode中保存的信息设置文件的信息。

1. 创建目录

首先读取此路径下的父目录的inode，如果不存在父目录，那么返回。然后查找这个父目录下是否存在要创建的目录，如果不存在那么创建此目录。如果父目录的数据块满了，需要重新申请一块数据库来存放文件的inode号。然后对于新的文件，需要申请一个inode节点和一块数据块，然后把inode号保存在父目录的数据块中。

1. 新建文件

首先读取此路径下的父目录的inode，如果不存在父目录，那么返回。然后查找这个父目录下是否存在要创建的文本文件，如果不存在那么创建此w恩本文将。如果父目录的数据块满了，需要重新申请一块数据库来存放文件的inode号。然后对于新的文件，需要申请一个inode节点和一块数据块，然后把inode号保存在父目录的数据块中。

1. 删除目录

首先查找父目录的inode，如果不存在父目录，那么返回。然后查找这个目录下是否存在改文件夹，如果存在这个文件夹并且文件夹下没有其他文件，那么删除这个目录。删除目录需要找到这个目录的inode号，然后查找到改目录占有的数据块号，重新设置数据位图。并且重新设置目录的inode位图。最后在父目录的inode中把要删除的inode号给去掉。

1. 删除文件

首先查找父目录的inode，如果不存在父目录，那么返回。然后查找这个目录下是否存在该文件，如果存在这个文件夹那么删除这个文件。删除文件需要找到这个文件的inode号，然后查找到该文件占有的数据块号，重新设置数据位图。并且重新设置文件的inode位图。最后在父目录的inode中把要删除的inode号给去掉。

1. 读取文件

首先找到文件的inode节点，如果没有这个节点那么返回。如果offset大于文件的大小那么直接返回。首先通过offset找到其实数据在第几块的数据块。然后获取第一块数据块，如果获取的size小于这个数据块要读的数据，那么获取下一个数据块，然后依次读取。返回最终读取到的数据大小，因为size可能大于文件的大小。

1. 写入文件

首先找到文件的inode节点，如果没有这个节点那么返回。如果offset大于文件的大小那么直接返回。首先通过offset找到其实数据在第几块的数据块。然后获取第一块数据块，然后开始写入。如果这个数据块不能容纳这么大的数据，那么获取下一块数据块，然后继续写入。最后返回写入的数据大小。

**3所遇到的问题及解决的方法**

不知道怎么利用libfuse实现文件系统，在这里参考了一些文章和代码链接如下:

<https://github.com/hewei2001/HITsz-fuse-ext2-filesystem>

<https://github.com/libfuse/libfuse>

<https://github.com/linyueQ/Operating-System-Course-Design-Simple-File-System-Based-on-FUSE>

**4.程序运行结果及使用说明**

使用说明:首先要安装libfuse-dev，然后新建一个build文件，进入build文件运行cmke .. 最后make即可，会生成一个Filesystem\_in\_Userspace可执行文件,然后创建一个fuse文件用来挂载，./ Filesystem\_in\_Userspace -f fuse

所有命令如下:

mkdir build

cd build

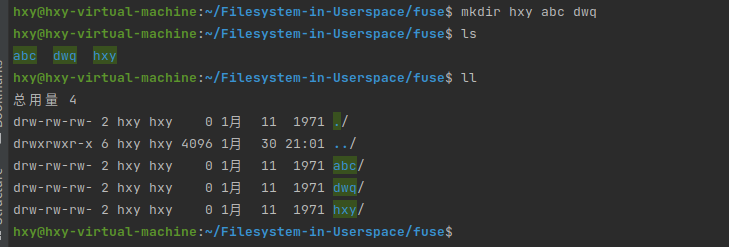
cmake ..

make

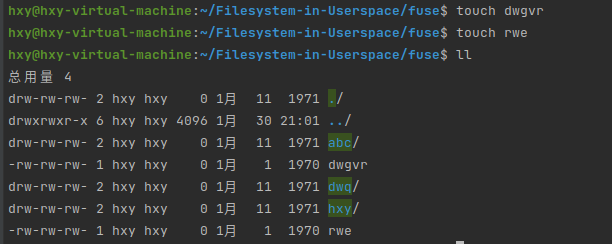
mkdir fuse

./Filesystem\_in\_Userspace -f fuse

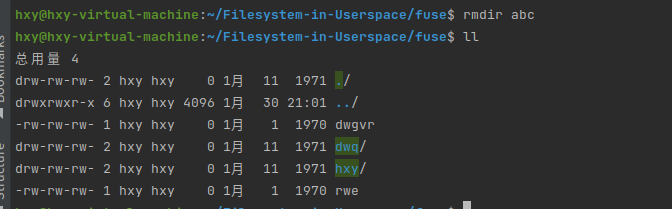
**创建目录**



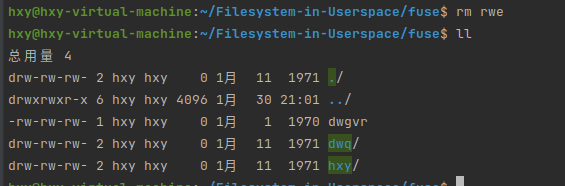
**创建文件**



**删除目录**



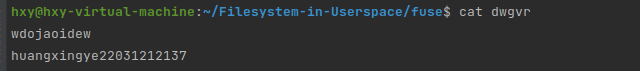
**删除文件**



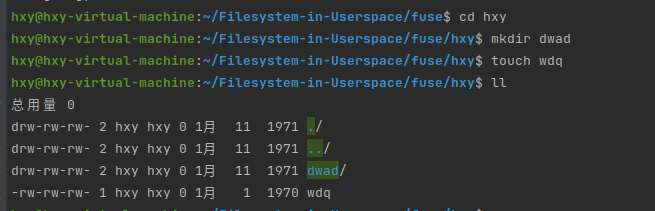
**写入文件**



**读取文件**



**进入二级目录并且创建文件**



**进入三级目录并且创建文件**

