文件系统设计详细说明：

数据结构

|  |  |  |
| --- | --- | --- |
| 成员名 | 类型 | 说明 |
| **系统打开文件表file[Nfile]** | | |
| f\_flag | int | 可读可写标志，0可读，1可读可写 |
| f\_state | int | 是否可用标志,0表示可用，1表示被占用 |
| f\_fcb | int | 在/..目录下打开文件则对应的文件控制块表fcb表下标，主目录或子目录下打开文件则对应要打开文件控制块数据在磁盘中的地址 |
| f\_woffset | long | 写指针，已写个数 |
| f\_roffset | long | 读指针，已读个数 |
| **文件控制块表fcb[Nfcb]（根目录区）** | | |
| filename | char数组 | 文件名或目录名 |
| state | char | 是否可用标志,0表示可用，1表示被占用 |
| type | char | 属性字段，0表示目录文件，1表示数据文件 |
| size | int | 文件长度 |
| firstnode | int | 首个物理块号，也即Fat文件分配表下标 |
| **文件分配表Fatfiletable[1000]** | | |
| state | char | 是否可用标志,0表示可用，1表示被占用 |
| nextbl | int | 0表示当前块为最后块，不为零指示下一物理块号 |
| **当前目录的文件控制块Curfcb** | | |
| filename | char数组 | 文件名或目录名 |
| state | char | 是否可用标志,0表示可用，1表示被占用 |
| type | char | 属性字段，0表示目录文件，1表示数据文件 |
| size | int | 文件长度 |
| firstnode | int | 首个物理块号，也即Fat文件分配表下标 |
| index | int | 当前目录为根目录区时，指向文件控制块表中某一项，为-1表示当前目录是在子目录下，此时无意义 |
| address | int | 当前目录是在子目录下时，保存子目录文件fcb信息在磁盘中的地址，留以后读取当前目录fcb用 |

用户文件描述符表：在文件系统中的作用为进程打开文件表。其记录的是当前进程已经打开文件的索引。

FCB块：文件名：设置为8个字节。

State：代表当前文件块是否已被使用

Size：文件大小/目录大小

Firstnode：该文件/目录所分配物理块的头结点。

Fat:state代表当前块是否已被使用

Nextfat代表下一个物理块的地址，当其为0代表已经没有块了。

### 文件卷设计



文件卷共分为**四个部分**：引导区、根目录区、FAT表1、FAT表2、数据区

**引导区**：开机运行时通过主引导区读取自举程序完成操作系统初始化，因为我们用模拟磁盘文件txt进行设计文件卷，所以直接对文件进行操作，不需要引导区。可作为空的内容，即不考虑。

**根目录区：**存放的是root根目录的信息，如根目录名、文件类型（目录）、目录文件的大小、以及目录数据存放的第一块物理块的位置。注：根目录里面存放的是子目录及文件FCB（文件控制块）的信息。

**FAT表1：**记录的是物理文件的数据存放信息。假设物理块（数据区）总块数为N，则FAT表的大小为FAT[N]。FAT表的数据结构设计为：state（当前物理块的是否被使用）、nextblock（写一个物理块的位置）。其中FAT表的下标即对应着物理块的位置。

**FAT表2：**为FAT表1的备份，可以完全将FAT表1的内容进行复制。

**数据区：**为目录、文件存放的数据区域。当为目录时存放的为其子目录/文件的FCB信息；当为文件时，存放的文件的数据信息。

采用的物理存储结构：物理结构的存储有连续结构、连接结构、索引结构。因为每种结构的方式都有它们的优缺点。这里我们采用连接结构：因为它只需要记录文件/目录的第一个物理块的位置，而物理块的位置和FAT表的下标一一对应，从而可以连续知道物理块的下一个内容。并且结合了连续结构、索引结构的优点：即可以进行连续存取文件数据、又可以动态的增加删除数据。

对于FCB块，总共设置了32，代表文件系统中最多能存32个文件

对于系统打开文件表和进程打开文件表设置的为10，代表当前进程所打开的文件数为10，并且对于进程打开文件表来说，只存放了索引，所以最终是根据索引去找到系统打开文件表。

磁盘结构设计

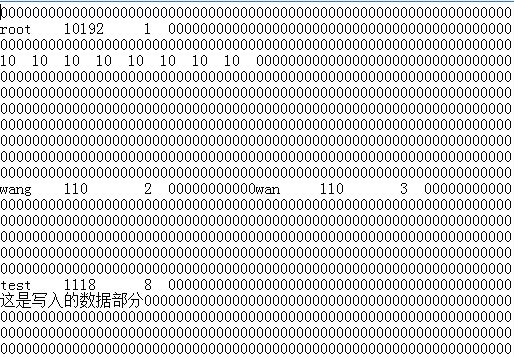
对于磁盘结构，总共采用了512\*1000，每个块的大小设置为512B，即一个扇区的大小，而在FAT32位管理系统中，是对簇进行的管理，簇是扇区的2的整数次倍，这里取0，所以就是每个簇就是一个扇区。

FCB在扇区中的存储为32B，格式如下：

8个是名字，接下来1个是状态，接着类型，接着8个数为文件大小，3个数为第一个物理块号。这里设计的原因是本次磁盘只设置了1000个块，0-999三位数刚好能够表示（注：因为是模拟磁盘文件，所以磁盘文件的数都是用十进制表示）FAB都占32字节，后面的11个0为保留部分无含义。

从12行开始是root目录文件的内容，即root目录下的子目录或文件的fcb，可知

www子目录 的fcb为”www 1032 7 00000000000”，进入www子目录并创建文件test，test的fcb保存在第18行（即11+7），为”test 1118 8 00000000000”，可知test文件的文件长度为18，首个块号为8，对应第19行（即11+8），看出可知，文件的数据为”这是写入的数据部分”，截图如下：



0

11

17

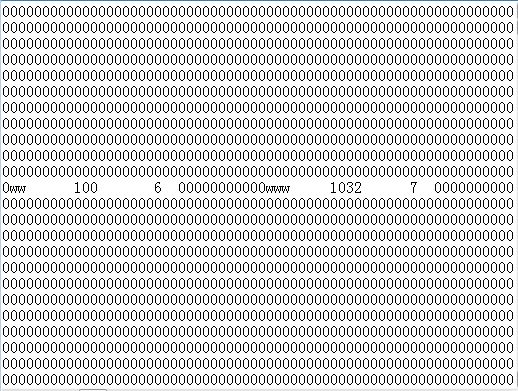
18

0

11

17

18



程序流程：

1：进入文件系统，先进行初始化先将进程打开文件表全部置为-1代表未使用，当模拟磁盘文件不存在时，则会创建一个模拟磁盘文件，并初始化为0，此时对于根目录也需要重新设计，则需要对磁盘文件进行写入初始化数据。同时对于FAT表也需要置1，代表根目录已被使用，同时将已打开文件表初始化。

2：开启文件系统，从内存中读取fat表的信息，对于该步，先读一个字节，作为状态信息，再读三个字节，作为下一个标识块。然后再读root的fcb信息，并将当前目录设置为根目录。

3：对于空闲空间的管理：采用FAT表进行管理，依次遍历FAT表，若此时FAT存在状态为0的块，则将其进行分配，mode为参数，若其为0代表首次申请块，

不为0的话，则表示需要对当前文件管理的块进行增加，而增加的块放在mode这个参数后面，若没有多余的块了则返回-1进行标识。

4：创建目录：先检查目录名是否重复，然后看是否已有根目录，若没有则创建，否则是在当前根目录或子目录下创建。

在根目录或者是当前子目录下创建：

参数：创建的目录名或文件名，标识文件或目录的标记

分为两大步：创建目录项和分配空间（修改磁盘空间）。

先判断有无空闲的空间（空闲盘块)，若有则开始创建。为了之后判断重名，所有的名字都是补全8个字节。接下来则是物理空间的分配，判断当前目录所占的块是否已满，若满则给当前目录分配新块，方法是先找到P最后一个物理块

然后%上512即可。接下来修改磁盘空间，若为根目录则先修改根目录的盘块，否则直接修改子目录的盘块（FCB)。之后则找到该目录最后一项，也就是新写的目录项，找到其在磁盘上的空间，对其进行写入基本数据。

打开文件：在根目录或子目录下打开文件：

输入参数（名字，读写状态）

1：先找到当前文件控制块的FCB信息

2：将其放置系统打开文件表和进程打开文件表中。

先遍历当前目录所有的FCB，与当前需要打开的文件名字是否相同，然后读取文件的大小，再找到打开文件的FCB信息在磁盘文件中的位置。然后将其信息复制到系统打开文件表中，再在进程打开文件表中记录。

读取文件：输入参数（读取文件名，读取的字节个数）

1：通过系统打开文件表找到FCB的地址并找到该文件大小。

2：根据读指针找到在磁盘文件中的地址。将该文件大小减去已读的即为可读部分。

3：每次读写都是按块读取，读了一次块就--

4：最后读完就修改读指针。

写入文件：（写入文件名，写入字节）

1：先判断打开文件是否有权限，再找到文件FCB的信息，再根据FCB的信息找到文件的最后一个块。

1. 根据写指针，判断当前写的位置，判断能否将要写的数据写入进去，若不行则将块的剩余部分写完，再去申请新的块。

3；根据写入的数据修改写指针，并修改FCB的大小。

关闭文件：

将该项的进程打开文件表项置于-1，将系统打开文件表的状态修改至0

列举目录：

遍历一遍当前目录块即可。