

**操作系统实验报告**

**实验题目 扫描 FAT12 文件系统管理的软盘**

**学生姓名**  付炎平

**学 号**  2019217819

**专业班级**  物联网工程19-2班

**指导教师**  田卫东

**完成日期**  2021.11.27

**合肥工业大学 计算机与信息学院**

**实验 13 扫描 FAT12 文件系统管理的软盘**

# 实验目的和任务要求

1. 通过查看 FAT12 文件系统的扫描数据，并调试扫描的过程，理解 FAT12

文件系统管理软盘的方式。

1. 通过改进 FAT12 文件系统的扫描功能，加深对 FAT12 文件系统的理解。

# 实验原理

FAT12 文件系统技术细节。

# 实验内容

## 准备实验

按照下面的步骤准备实验：

* + 1. 启动 OS Lab。
    2. 新建一个 EOS Kernel 项目。

## 阅读控制台命令“ sd ”相关的源代码 ，并查看其执行的结果

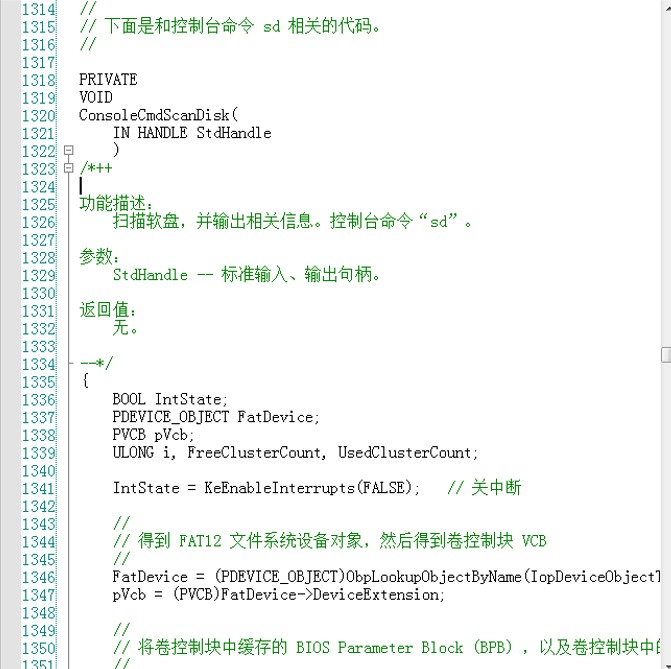
阅读 ke/sysproc.c 文件中第 1320 行的 ConsoleCmdScanDisk 函数，学习“sd”命令是如何扫描软盘上的 FAT12 文件系统的。在阅读的过程中需要注意下面几点：

1. 在开始扫描软盘之前要关闭中断，之后要打开中断，这样可以防止在命令执行的过程中有其它线程修改软盘上的数据。
2. 以软盘的盘符“A:”做为 ObpLookupObjectByName 函数的参数，就可以获得 FAT12 文件系统设备对象的指针。
3. FAT12 文件系统设备对象的扩展块（FatDevice->DeviceExtension）是一个卷控制块（VCB，在文件 io/driver/fat12.h 的第 115 行定义），从其中可以获得文件系统的重要参数，并可以扫描 FAT 表。
4. FatGetFatEntryValue 函数可以根据第二个参数所指定的簇号，返回簇在

FAT 表中对应项的值，在扫描 FAT 表时通过调用此函数来统计空闲簇的数量

（FreeClusterCount）。

阅读 ConsoleCmdScanDisk 函数如下：





按照下面的步骤执行控制台命令“sd”，查看扫描的结果：

* 1. 按 F7 生成在本实验 3.1 中创建的 EOS Kernel 项目。
  2. 按 F5 启动调试。
  3. 待 EOS 启动完毕，在 EOS 控制台中输入命令“sd”后按回车。

观察命令执行的结果，如图 21-1 所示，可以了解 FAT12 文件系统的信息。



## 根据 BPB 中的信息计算出其他信息

* + 1. **要求**

修改“sd”命令函数 ConsoleCmdScanDisk 的源代码，在输出 BPB 中保存的信息后，不再通过 pVcb->FirstRootDirSector 等变量的值进行打印输出，而是通过 BPB 中保存的信息重新计算出下列信息，并打印输出：

1. 计算并打印输出根目录的起始扇区号，即 pVcb->FirstRootDirSector

的值。

1. 计算并打印输出根目录的大小，即 pVcb->RootDirSize 的值。
2. 计算并打印输出数据区的起始扇区号，即 pVcb->FirstDataSector 的值。
3. 计算并打印输出数据区中簇的数量，即 pVcb->NumberOfClusters 的值。

## 测试方法

1. ConsoleCmdScanDisk 函数的源代码修改完毕后，按 F7 生成项目。
2. 按 F5 启动调试。
3. 待 EOS 启动完毕，在 EOS 控制台中输入命令“sd”后按回车。输出的内容应该仍然与图 19-1 所示的内容相同。

## 提示

在 ConsoleCmdScanDisk 函数中，使用了下面的语句打印输出根目录的起始扇区号：fprintf(StdHandle, "First Sector of Root Directroy: d\n", pVcb->FirstRootDirSector);

根目录的起始扇区号可以使用保留扇区的数量加上 FAT 表占用扇区的数量来计算获得，而这些信息都可以从 BPB 中获得，所以上面的语句可以修改为： fprintf(StdHandle, "First Sector of Root Directroy: d\n", pVcb->Bpb.ReservedSectors +pVcb->Bpb.Fats \* pVcb->Bpb.SectorsPerFat);对于根目录的大小、数据区的起始扇区号、数据区中簇的数量这些信息也可

以采用类似的方式计算获得。修改代码：

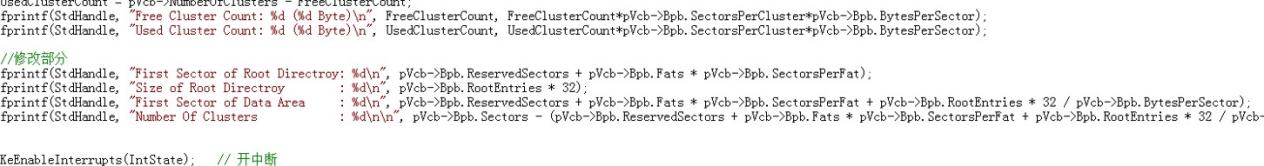
fprintf(StdHandle, "First Sector of Root Directroy: d\n", pVcb->Bpb.ReservedSectors + pVcb->Bpb.Fats \* pVcb->Bpb.SectorsPerFat);

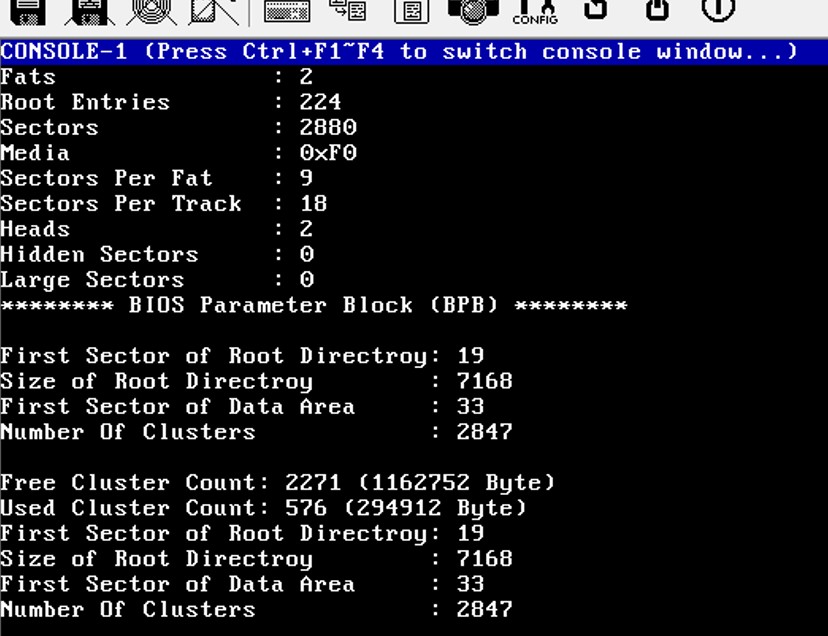
fprintf(StdHandle, "Size of Root Directroy : d\n", pVcb->Bpb.RootEntries \* 32);

fprintf(StdHandle, "First Sector of Data Area : d\n", pVcb->Bpb.ReservedSectors + pVcb->Bpb.Fats \* pVcb->Bpb.SectorsPerFat

+ pVcb->Bpb.RootEntries \* 32 / pVcb->Bpb.BytesPerSector);

fprintf(StdHandle, "Number Of Clusters : d\n\n", pVcb->Bpb.Sectors - (pVcb->Bpb.ReservedSectors + pVcb->Bpb.Fats \* pVcb->Bpb.SectorsPerFat + pVcb->Bpb.RootEntries \* 32 / pVcb->Bpb.BytesPerSector) / pVcb->Bpb.SectorsPerCluster);

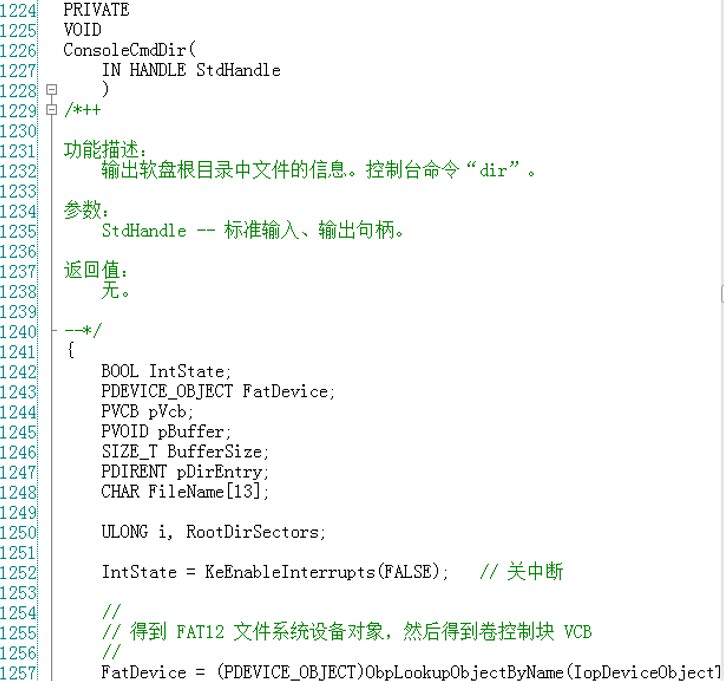
如下图所示：

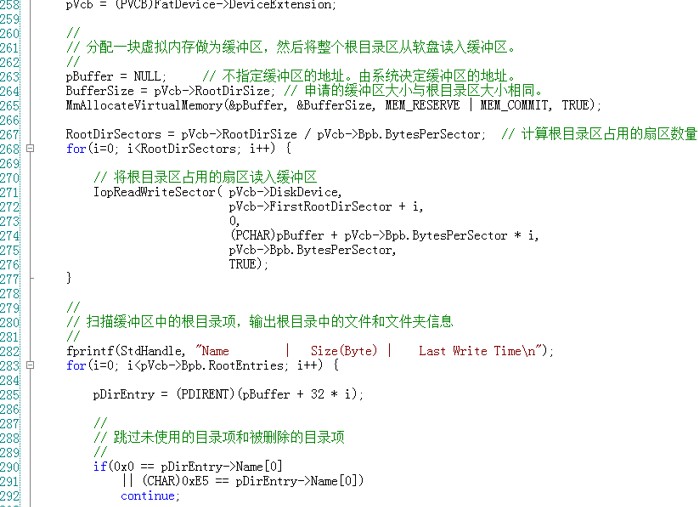
测试结果如下：

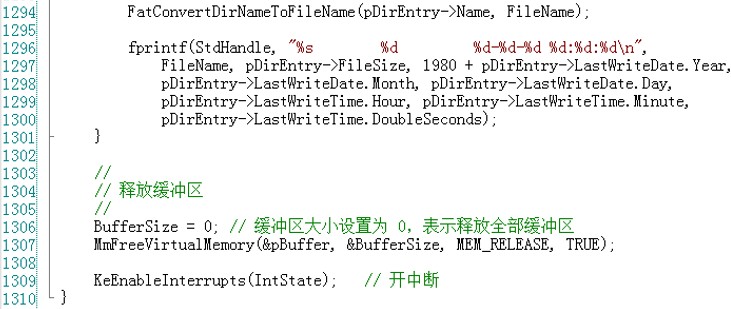
## 阅读控制台命令“ dir ”相关的源代码，并查看其执行的结果

阅读 ke/sysproc.c 文件中第 1226 行的 ConsoleCmdDir 函数，学习“dir” 命令是如何扫描软盘的根目录并输出根目录中的文件信息的。在阅读的过程中需要注意下面几点：

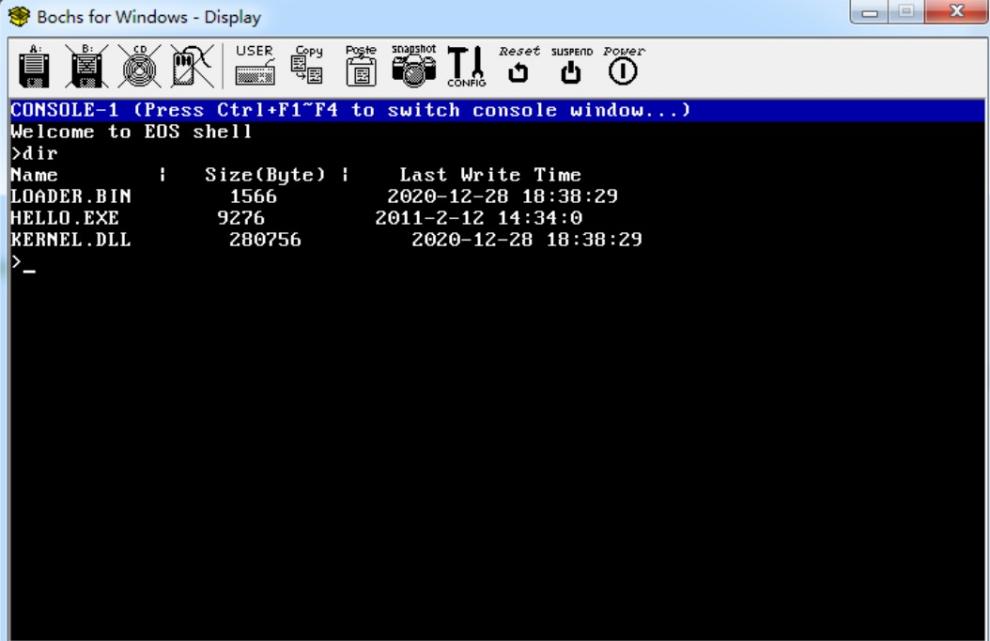
1. 在开始扫描根目录之前要关闭中断，之后要打开中断，这样可以防止在命令执行的过程中有其它线程修改软盘上的数据。
2. 以软盘的盘符“A:”做为 ObpLookupObjectByName 函数的参数，就可以获得 FAT12 文件系统设备对象的指针。
3. FAT12 文件系统设备对象的扩展块（FatDevice->DeviceExtension） 是一个卷控制块（VCB，在文件 io/driver/fat12.h 的第 115 行定义），从其中可以获得文件系统的重要参数，可用于扫描根目录。
4. 由于根目录的数据在软盘上，所以调用 MmAllocateVirtualMemory 函数分配了一块与根目录大小相同的缓冲区，然后调用 IopReadWriteSector 函数将根目录占用的扇区依次读入了缓冲区。注意在命令执行的最后需要调用 MmFreeVirtualMemory 函数释放缓冲区。
5. 在扫描缓冲区中的目录项时，跳过了未使用的目录项和已经被删除的目录项，而只输出当前使用的目录项（文件）信息，包括文件名、文件大小和最后改写时间。

研究 ConsoleCmdDir 函数如下：





测试结果如下：



## 输出每个文件所占用的磁盘空间的大小

* + 1. **要求**

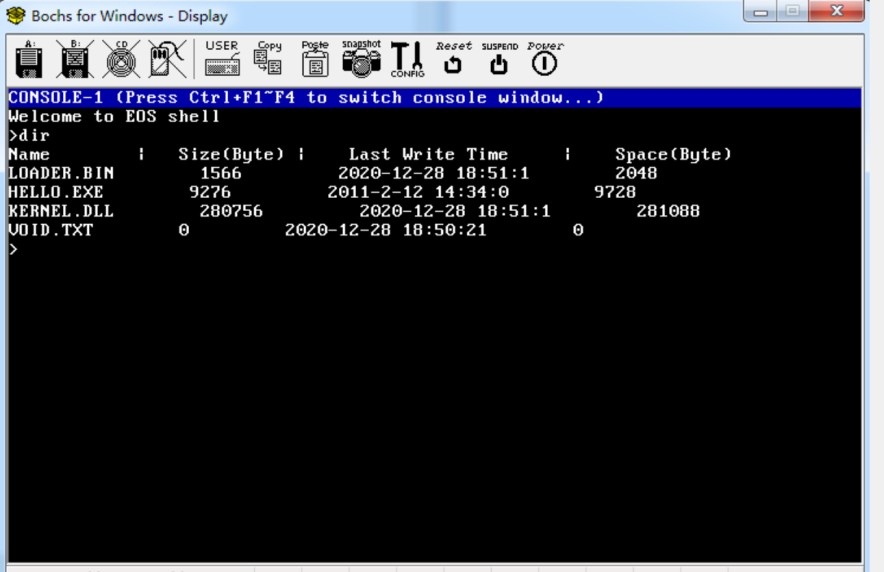
修改“dir”命令函数 ConsoleCmdDir 的源代码，要求在输出每个文件的名称、大小、最后改写时间后，再输出每个文件所占用的磁盘空间（以字节为单位）。

## 测试方法

1. ConsoleCmdDir 函数的源代码修改完毕后，按 F7 生成项目。
2. 在 “ 项 目 管 理 器 ” 窗 口 中 双 击 Floppy.img 文 件 ， 使 用

FloppyImageEditor 工具打开此软盘镜像。

1. 将“学生包”本实验文件夹中的 void.txt 文件（大小为 0）添加到软盘镜像的根目录中（将 void.txt 文件拖动到 FloppyImageEditor 窗口中释放即可）。
2. 点击 FloppyImageEditor 工具栏上的保存按钮，关闭该工具。
3. 按 F5 启动调试。
4. 待 EOS 启动完毕，在 EOS 控制台中输入命令“dir”后按回车。

输出的内容应该与图 21-3 所示的内容相同，或者可以在“项目管理器”窗口中双击 Floppy.img 文件，使用 FloppyImageEditor 查看文件的相关信息， 检验输出的结果是否正确。

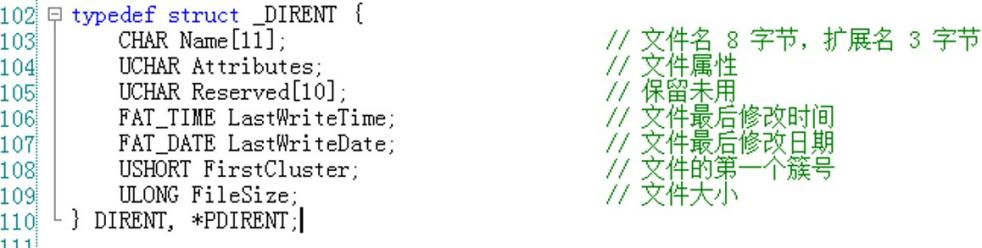
## 提示

文件的大小与文件所占用的磁盘空间是两个不同的概念，文件所占用的磁盘空间是簇的整数倍，所以文件所占用的磁盘空间总是大于或等于文件的大小。例如，如果一个簇只包含一个扇区，大小是 1 字节的文件，其占用的磁盘空间至

少是一个簇的大小即 512 字节，大小是 600 字节的文件，其占用的磁盘空间至

少是两个簇的大小即 1024 字节。但是，不能简单的认为大小是 1 字节的文件就一定只占用一个簇，该文件完全可以占用多个簇。所以，统计文件所占用的磁盘空间的方法应该是，根据文件在 FAT 表中的簇链来进行统计，而不能简单的将文件的大小对齐到簇的大小。其他需要提示的内容有：

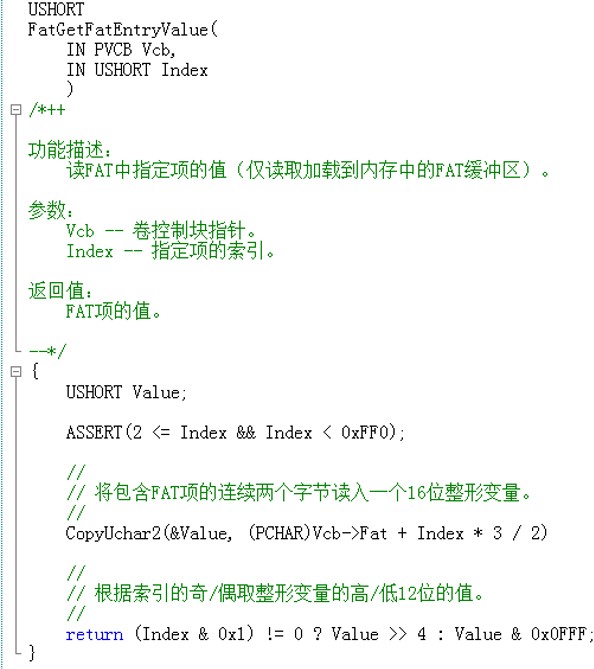
（1）目录项结构体的定义可以参见文件 io/driver/fat12.h 的第 102 行，

其中的 FirstCluster 域记录了文件的起始簇号。

（2）获得文件的起始簇号后，可以循环调用 FatGetFatEntryValue 函数

（在文件 io/driver/fat12.c 的第 307 行定义），遍历文件占用的簇所组成的簇链，当 FAT 表项的值为 0xFF8 时，表示簇链结束。

（3）对于大小为 0 的文件，并不代表文件不会占用簇，只有当文件的起始

簇号为 0 时，才代表文件没有占用簇。

# 实验的思考与问题分析

* 1. 在 ConsoleCmdScanDisk 函数中扫描 FAT 表时，为什么不使用 FAT 表项的数量进行计数，而是使用簇的数量进行计数呢？而且为什么簇的数量要从 2 开始计数呢？

答：因为一个 FAT 占一个扇区，该表的所有指针数加起来有可能大于总簇的数量，所以在统计时不用 FAT 的表项的数量，而是使用簇的数量。簇的数量从 2

开始计数是由于前面两个簇是固定的簇，表示固定的大小，不可使用。

* 1. 在 ConsoleCmdScanDisk 函数中扫描 FAT 表时，统计了空闲簇的数量， 然后使用簇的总数减去空闲簇的数量做为占用簇的数量，这种做法正确吗？是否还有其他类型的簇没有考虑到呢？

答：不准确，可能还有坏簇，保留簇等簇的类型存在，没有被考虑到。

# 总结和感想体会

1. 通过使用 dir，sd 命令，我对于文件目录的查询方式、FAT12 文件系统在软盘上的扫描方式都有了深刻的体会。在扫描前需要关中断，扫描完成后再进行开中断操作。
2. 在修改代码的过程中，我对于 BPB 的了解更加深刻，并且学会了对于磁盘数据的计算。
3. 在对于 dir 命令的使用过程中，我练习了按照簇进行大小计算的方法，对于文件夹大小有了进一步的了解。
4. 在调试 FAT12 文件系统的扫描功能的过程中，我对于 FAT12 文件系统管理软盘的方式了解更加深刻，也得到了实际操作的练习，明白了 FAT12 文件系统是如何具体实现的。