**华 南 农 业 大 学 数 学 与 信 息（软 件） 学 院**

《操作系统分析与设计实习》成绩单

开设时间：2017 学年第一学期

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 小组成员、组内分工、工作量比例、各成员个人成绩 | | | | | | | | | | | |
| 学号 | 201515080107 | 姓名 | 胡燕妮 | 分工 | 界面设计 | | 工作量比例 | | 10% | 成绩 |  |
| 学号 | 201525050305 | 姓名 | 曽志鹏 | 分工 | 指令系统设计，用户命令接口交互 | | 工作量比例 | | 20% | 成绩 |  |
| 学号 | 201525050315 | 姓名 | 林丹鹏 | 分工 | 文件系统、设备管理、模拟CPU、存储管理的实现，各模块的整合 | | 工作量比例 | | 30% | 成绩 |  |
| 学号 | 201525050323 | 姓名 | 张红英 | 分工 | 界面设计 | | 工作量比例 | | 10% | 成绩 |  |
| 学号 | 201525050320 | 姓名 | 许佳钿 | 分工 | 界面与内部数据的交互逻辑代码 | | 工作量比例 | | 15% | 成绩 |  |
| 学号 | 201525050327 | 姓名 | 庄展济 | 分工 | 进程管理，PPT设计 | | 工作量比例 | | 15% | 成绩 |  |
| 实验题目 | 模拟操作系统实现 | | | | | | | | | | |
| 自我评价 | 胡燕妮：在这次实验中，通过模拟操作系统实现，加深了我对操作系统工作原理理解，进一步了解操作系统的实现方法，在合作完成系统的过程中认识到团队精神的可贵，并提高了程序设计能力。同时我也认识到自己的不足，由于水平有限，没有参与核心代码的设计，学无止境，路漫漫其修远兮，希望以后能通过学习提升自己。  曾志鹏：本次实验我主要参与了指令的设计和用户命令接口的实现，这部分的设计不算难，不过我对其他模块的设计也是很感兴趣，和其他组员交流之后，自己也学习了很多知识，知道了应该如何设计各个模块，经过这次实验，自己的程序设计能力有所提高，有一点是这次合作我们使用了git，这在协作方面给我们带来了很多的便利，而且这次实验也让我对操作系统有了更多的认识。  林丹鹏：本次实验最大的收获是在设计实验的过程，我反复翻阅了操作系统的课本，虽然最终实现真正用到的地方很少，但这却让我发现了自己的知识盲区，特别是进程这一方面，这对我学习底层编程有很大的帮助；其次是对于内存映射文件这方面的理解又有所加深，收获颇多，不好的一点就是本学期事有点多，做这个实验做得有点赶，哈哈。  张红英：在做实验前一定要将知识吃透，因为这是做实验的基础，否则在做实验时的难度就会加大，浪费做实验的宝贵时间，使你事倍功半，做实验时一定要亲力亲为，务必要将每个步骤，每个细节弄清楚，弄明白，这次实验拓宽我们的眼界,使我们认识到操作系统的应用是那么的广泛，通过这次实验，使我学到了不少实用的知识，更重要的是，做实验的过程，思考问题的方法，这与做其他的实验是通用的，真正使我们受益匪浅。  许佳钿：这次的课程综合性设计，我负责界面与内部数据的交互逻辑代码，向我们组的林丹鹏同学请教了问题，学习到了很多知识，很感激组员对我的耐心帮助，也感谢张猜老师对我们小组的指导，这次课程设计学习了很多知识。  庄展济：“千里之行，始于足下”。这个课程设计也是如此，从一开始准备到后来的实现，都是一点点的积累起来。前期准备过程中，通过重复看说明文档以及上网查找相关资料，对整个课程设计的内容有一个大概的理解。实现过程中不断摸索，通过思维导图的方式，使得自己对操作系统的架构、运行有一个系统的整体的知识体系。总的来说，这次课程设计给我带来的提升是巨大的，尤其体现在实验技巧、知识水平以及沟通能力。 | | | | | | | | | | |
| 教师评语 | 评价指标：  题目内容和要求完成情况 优□ 良□ 中□ 差□  对算法原理的理解程度 优□ 良□ 中□ 差□  程序设计水平 优□ 良□ 中□ 差□  程序运行效果及正确性 优□ 良□ 中□ 差□  课程设计报告结构清晰 优□ 良□ 中□ 差□  报告中总结和分析详尽 优□ 良□ 中□ 差□ | | | | | | | | | | |
|  |  | | | | | **教师签名** | |  | | | |
|  |  | | | | |  | |  | | | |

**一、需求分析**

本次实验要制作一个具有模拟CPU、模拟内存、模拟硬盘、模拟CPU管理、模拟内存管理、模拟硬盘管理、模拟进程管理、模拟指令系统、模拟中断处理的一个能运行在WINDOWS平台上的一个模拟的单用户多任务操作系统。

**1、实验要求**

（1） 文件管理

文件管理和主要是单用户的磁盘文件管理部分，包括文件的逻辑结构、物理结构、目录、磁盘分配回收文件。

（2）存储管理

存储管理部分主要实现内存空间的分配和回收、存储保护 。

（3）设备管理

设备管理 主要包括 设备的分配和回收 。

（4） 进程管理

进程管理要包括进程调度，进程的创建和撤销进程的阻塞和唤醒中断。

（5）命令接口

命令接口用于用户输入命令，调用相应的方法。

1. **输入形式与范围**
2. 输入入口  
    用户可以在命令接口模块输入命令。
3. 输入形式  
    命令+{文件路径，}.
4. 输入范围

可选的命令有：create/delete/type/run/mkdir/rmdir/chang/copy/format/open

1. **输出的形式**
2. 提示输出  
    当在命令接口输入命令后，接口会对命令的操作结果进行提示，方便用户使用。
3. 界面输出

当执行命令成功后，界面会响应用户命令的执行结果，比如使用create命令建立文件后，磁盘分配表视图和目录树视图会产生相应的变化。

1. **达到的功能**

本程序可以模拟简单操作系统的运行。

1. 测试数据

测试数据与测试结果参考下文。

1. **概要设计**

系统顶层设计：



各模块设计：

**1、模拟CPU的实现**

（1）CPU组成

四个数据寄存器（AX,BX,CX,DX）：用于存储操作数。

一个程序计数器（PC）：用于存储下一条指令的地址。

一个指令寄存器（IR）：用于存储当前执行的指令。

1. 运行流程

CPU实体为一个单独的线程不断运行，执行过程分为3个周期，分别为取值、译码和执行（包括写回）。

**2、模拟指令系统**

（1）规则表：

|  |  |  |
| --- | --- | --- |
| 指令 | 汇编代码 | 译码 |
| NOP | NOP | 0000 0000 |
| X++ | INC DR | 0001 XX00 (XX为寄存器编号) |
| X-- | DEC DR | 0010 XX00 (XX为寄存器编号) |
| !?? | !?? | 0011 XXYY (XX为设备编号，YY为使用时间) |
| END | END | 0100 0000 |
| X=? | MOV DR IMM | 0101 XX00 XXXX XXXX(双字指令) |

**3、 内存管理**

（1）内存组成

系统内存：系统内存是用来存放进程控制块，主要使用java的API进行管理，没有 具体的大小。

用户内存：用户内存占512字节，由一个数组来模拟，当创建时系统分配内存给进程， 撤销时释放内存。

1. 管理方案

本系统采用动态分区算法管理内存，使用一个内存分配表进行维护。内存分配表 使 用链式结构，其中每一个元素指向一个空闲分区或者使用中的分区，含有大小、起始 地址、进程ID等属性，初始时只有一个元素，即整块系统内存。在系统运行时按初始 给定大小在物理内存中开辟一块内存空间作为本系统的模拟内存 供该系统使用，主 要用于存放进程。

**4、文件管理**

（1）硬盘结构

系统硬盘由一个二进制文件模拟，总大小为8KB，被划分为128个磁盘块，每个磁盘 块64字节。前两个磁盘块用来存放磁盘分配表，其他磁盘块用来存放数据。

1. 目录结构

一个目录由目录项和内容组成，目录项占8个字节，存放目录的信息，目录项磁盘块 在分配表中指向同目录下的下一个目录；内容代表目录的数据，如果目录是普通文件， 则内容是文件的数据，如果目录是文件夹，则内容是文件夹的子目录。

1. 内部实现

系统将模拟磁盘载入为一个内存映射文件进行管理，对文件的操作都会及时反映到文件里，不会出现数据不一致的情况；对于文件的读写操作并没有采用模拟内存作为缓存，而是直接使用外部的缓冲数组。

1. **设备管理**
2. 设备组成

硬件设备由3个实体对象模拟，对象中名为count的属性代表设备的数量，当设备被 分配给进程时数量减1s，设备使用完毕时加1，数量为0说明已无可用设备。

1. 内部设计  
    设备请求队列：当进程申请使用设备时，会生成一个请求信号放入 该队列中，然后设 备管理器不断从队列中获取申请， 判断是否有可用设备，有则分配设备给请求的进程， 没有则将请求重新放入队列中。

设备使用队列：被进程使用的设备会生成一个信号放入该队列，当时间片一到信号自动 释放，然后发送设备中断给CPU，从而唤醒进程。

1. **进程管理**
2. 创建进程  
    当使用run命令运行文件时，系统首先检查内存是否足够，如果足够则分配进程控制 块和内存，此时再判断当前运行的进程是否为闲逛进程，是的话就申请进程调度，否则 将该进程放入就绪队列。
3. 阻塞进程  
    运行中的进程申请使用设备时，将会被调入阻塞队列，然后进行进程调度。
4. 唤醒进程  
    当设备使用结束后，会发生中断信号给CPU，CPU将阻塞中的进程放入就绪队列。
5. 撤销进程

执行end指令时发生软中断，系统释放进程占用的内存，并进行进程调度。

1. 进程调度  
    保存当前运行进程的现场，并从就绪队列获取一个进程，恢复该进程的现场，分配时间 片给其运行。
2. **详细设计**

**由于程序代码过多，我这里只挑选关键代码分析。**

**1、用户命令接口函数调用**

void executeCMD（String 用户命令）{

命令程序，文件路径名，...=解析用户命令（用户命令）；

switch（命令程序）{

case “create”:fileOperator.create(文件名，文件属性); break；

case “type”:fileOperator.type(文件路径名);

break;

....

default:print(“命令不存在”);break;

}

}

**2、文件管理类fileOperator**

/\*

\*创建目录

\*/

public void create(String 文件名，String 文件属性){

int 可用磁盘块=找第一个可用磁盘块（）；

if（不存在可用磁盘块）

throw new Exception(“磁盘空间不足”);

if（存在相同目录）

throw new Exception(“存在同名目录”);

//分配空间给该文件

修改磁盘分配表；

修改目录指针；

}

/\*

\*显示目录内容

\*/

public String type(String 文件名){

OpenedFile 打开文件项=打开文件（）；

byte[] content=read（打开文件项，读取长度）；

关闭文件（）；

return 数组转字符串(content)；

}

/\*

\*读取文件内容

\*/

public byte read(OpenedFile 打开文件项，int 读取长度){

if（当前文件不可读）

throw new Exception（“文件不能读取”）;

Pointer 读取指针=打开文件项.get读取指针（）；

byte[] content;

while（指针未到结尾或者未达到读取长度）{

content[i]=磁盘文件读取下一个字节；

}

return content；

}

**3、模拟CPU**

public void run（）{

while（系统未关闭）{

加锁（）；

取值（）；

译码（）；

执行写回（）；

释放锁（）；

}

}

**4、设备管理操作**

//设备请求队列处理逻辑

while（系统未关闭）{

从请求队列中获取请求信号；

switch(请求的设备){

case ‘A’:

if(A设备数量>0)

分配A设备，将设备使用信号放入设备使用队列；

else

重新将信号放入请求队列；

break；

case ‘B’:

if(B设备数量>0)

分配B设备，将设备使用信号放入设备使用队列；

else

重新将信号放入请求队列；

break；

case ‘C’:

if(C设备数量>0)

分配C设备，将设备使用信号放入设备使用队列；

else

重新将信号放入请求队列；

}

}

//设备使用队列处理逻辑

while(系统未关闭){

从设备使用队列中获取使用完设备的进程ID；

释放设备；

唤醒进程；

}

**5、进程管理操作**

/\*

\*创建新进程

\*/

public void createProcess(){

if(内存不足){

throw new Exception(“内存不足”);

}

分配进程控制块；

分配内存块；

将文件数据复制到用户区；

if（当前运行进程不是闲逛进程）{

加锁（）；

申请进程调度；

释放锁（）；

}

else{

将进程放入就绪队列；

}

}

**6、模拟时钟**

public void run(){

while(系统未关闭){

系统时间++；

if（时间片到了）{

加锁（）；

申请进程调度；

释放锁（）；

}

}

}

**四、调试分析**

**1、实验过程的主要问题及解决方法**

实验一开始遇到的问题是对于文件系统理解有点困难，不知道如何去实现它，后来去办公室问了老师，然后自己再认真研读一下指导书的提示，才开始有了眉目。理解了原理后，编码的过程还是比较顺利的，当然中间还是有遇到一些技术上的问题，比如说磁盘数据如何保证一致性，如何对模块进行等等；解决了文件系统后，其他的管理模块就相对比较简单了，主要是解决并发的问题，并发的难点主要在于出错了难以调试，总是要重复执行好多遍才能发现问题，后来去认真学习了一下多线程的调试技术，才顺利地把bug解决；还有就是界面的问题，写一个可视化界面不难，问题是如何让界面高效及时地响应数据的变化，还要与核心代码解耦，这点是需要一定经验与技巧的，我们暂时也找不到较好的办法。

1. **算法时间复杂度及其改进**

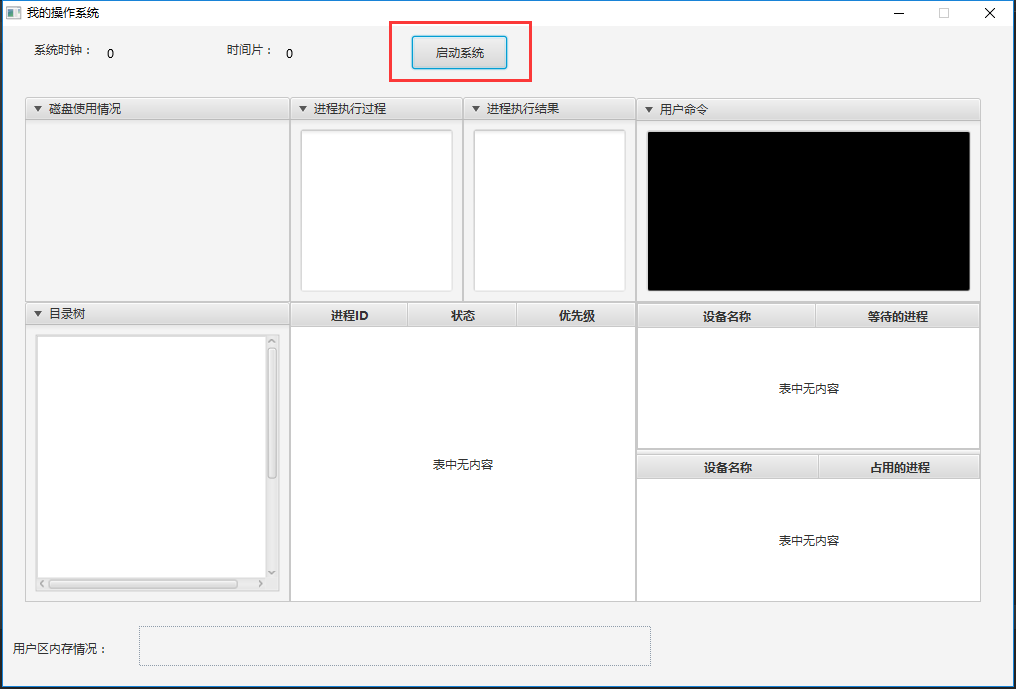
本程序大部分代码的复杂度只是O(n)而已，但是文件系统部分就用到比较多的O(n!)的算法，比如查找目录时用到了深度遍历，这点是需要提升的，我感觉这个主要是设计上的问题，而不是算法上的问题，所以打算用linux系统的文件系统结构来重构这个项目。

**3、设计过程的经验与体会**

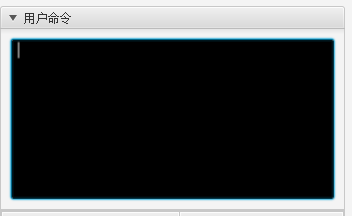
本次实验遇到了很多的困难，也走了不少弯路，比如一开始审题不细心，还以为是要从头到尾设计一个真实的操作系统，所以就买了一本《30天自制操作系统》，后来才发现是模拟而已，再从头开始用java设计。总的来说，这次实验让我们明白了最实验也好，做项目也好，首先要理解好需求再下手，这是至关重要的。

**五、用户使用说明**

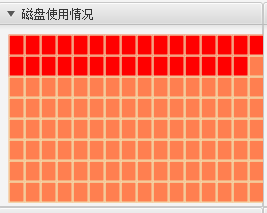
开始：点击启动系统



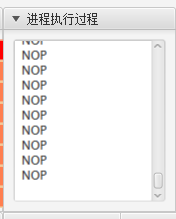
启动系统后，在命令框中输入你所需实现功能的命令，命令格式及可选的命令参照上文



查看磁盘分区使用情况：



查看进程执行中间结果：



查看放入内存的进程队列：



查看内存用户区使用情况：



查看设备使用情况：

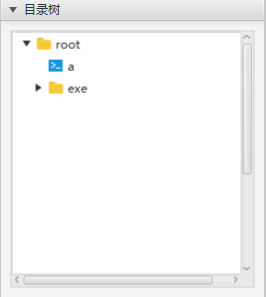
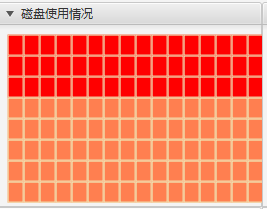


**六、测试与运行结果**

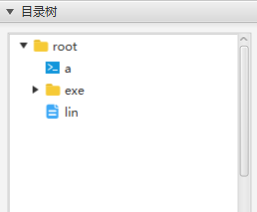
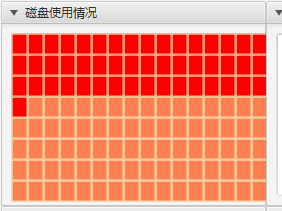
1、测试文件系统

（1）create命令

假设要在磁盘root目录下新建一个lin文件，测试前目录情况如下：



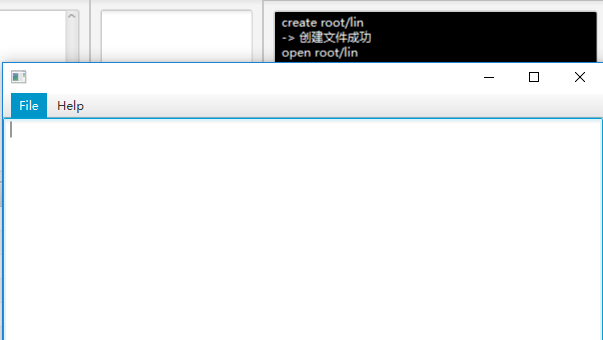
输入命令后：



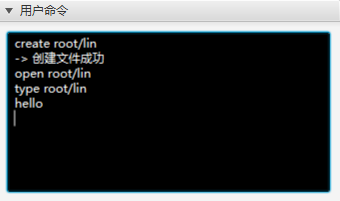
可以看到，磁盘分配表和目录树都有了相应的变化，测试成功。

1. open和type命令

输入open root/lin，可以看到文件内容是空的：



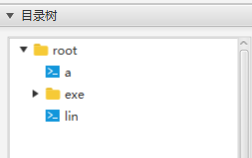
输入“hello”然后点击菜单里的save条目，保存文件，然后使用type命令查看：



可以看到文件内容已经改变。

1. change 命令

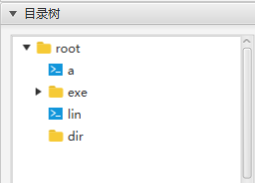
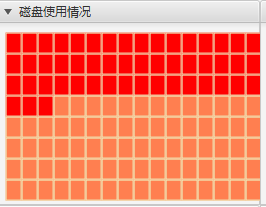
输入change root/lin 16命令，将普通文件改为可执行文件：



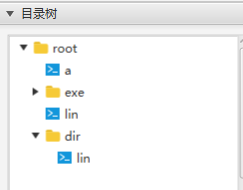
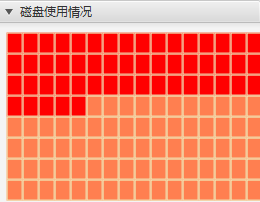
图标已经改变，说明属性有了变化。

1. mkdir和copy 命令

先创建一个文件夹root/dir，输入mkdir root/dir：



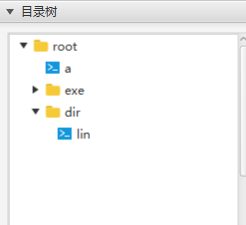
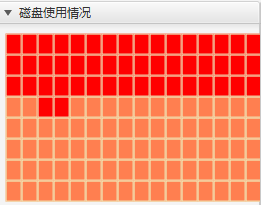
目录创建成功，输入copy root/lin root/dir/lin:



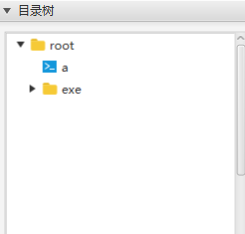
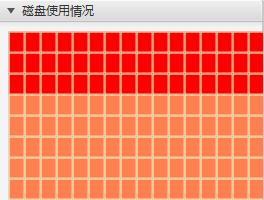
可以看到文件已经复制成功。

1. delete和rmdir命令

输入delete root/lin命令测试：



可以看到文件删除成功，再输入rmdir root/dir命令：



文件夹已经被删除。

1. run命令

输入 run root/exe/0e，运行我们事先准备好的可执行文件：



可以看到出现了新的进程，运行成功，接下来测试其他模块。

2、测试进程模块

系统一开始只有一条闲逛进程：



当运行文件时，闲逛进程被放入就绪队列，刚创建的进程被执行，如图：



当进程等待设备或使用设备时，进程被阻塞:



进程模块测试无误。

3、测试内存模块

当运行进程时，内存划分一块区域给改进程（红色部分）：



当程序运行结束后，内存释放：



4、测试设备模块

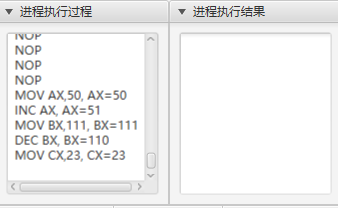
当运行的进程请求使用设备，并且当前有可用设备时，系统分配设备给该进程，并将进 程调入阻塞队列，如图：



当运行的进程请求使用设备，但当前无可用设备时，该请求会继续放在请求队列，直到有可用设备，如图：



5、查看CPU执行情况



可以看到文件指令已经被成功解析了。