题目六　模拟操作系统实现

一、课程设计目的

通过模拟操作系统的实现，加深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，并可练习合作完成系统的团队精神和提高程序设计能力。

二、小组人数

建议 4～6 人一组共同完成模拟系统的实现。

选择题目“模拟操作系统实现”的小组在最终提交时须公开演示及讲解。由于这个题目最复杂，难度和工作量最大，故小组成员最后得分也酌情高于选择其它题目的同学的分数（高 5~15 分）。

三、编程语言

建议使用一些 Windows 环境下的程序设计语言如 VC、Java，以借助这些语言的多线程来模拟并行发生的行为。要求图形界面。

四、课程设计内容

模拟一个采用多道程序设计方法的单用户操作系统，该操作系统包括进程管理、存储管理、设备管理、文件管理和用户接口四部分。

五、课程设计具体内容和要求

1、文件管理和用户接口

文件管理和用户接口部分实现的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、文件的保护和用户接口的实现。这一部分可参考题目五的说明。

（1） 文件的逻辑结构

文件的逻辑结构采用流式结构；文件均采用文本文件。

假设系统中只有两种文件，一种是存放任意字符的普通文本文件，一种是可执行文件。**可执行文件的程序内容手工输入，事先创建约 10 个可执行文件，将来用这些可执行文件进行后续的进程创建、内存分配、进程执行/调度和设备分配。**

这里，“可执行文件”中的“指令”只有 5 种，包括：

x=?　给 x 赋值（数值不用太大，一位数、两位数即可。不要求存储或任意指定变量名，只是实现“赋值”即可）。

x++　x 加 1 （设 x 值总是小于等于 255、大于等于 0）。

x--　　x 减 1。

!? ?　　!是“特殊命令（I/O）的前缀”，第一个?为 A,B,C 中的某个设备，第二个?为一位整数，表示使用设备的时间（例如假定一个数，这个数随着系统时间增加而递减（时间单位自定，例如：秒）。减到 0 时，认为设备工作完成）。

end　表示“可执行文件”结束。

每个可执行文件中可以包含多条同一类指令，假设每条指令在文件中占 1 字节（自己思考如何把前述 5 种指令用一个字节表示并存储在后述的 disk 磁盘块里，其实这是一个“汇编/编译”的模拟过程）。如果可执行文件中的指令超过 64 条，就再多分配一个磁盘块，以此类推。

假设每种指令都是原子性的（即不考虑各指令的汇编实现细节，每种指令都是原子执行的），且执行时间都是一个时间单位，后面进程调度的时间片是 n 个时间单位，例如 n=6，就表示本进程执行 6 条指令之后将发生进程调度。

（2） 磁盘模拟

用一个文件 disk 模拟磁盘，设磁盘的每个盘块 64 字节，模拟磁盘共有 256 块。第 0、1 块存放文件分配表，第 2 块存放根目录，其余存放子目录和文件。（所以你创建的目录和流式文件不能太大太多，但至少要包含 5 个目录和 15 个文件。注意：文件对磁盘块是独占的，即每个文件至少占据一个磁盘块，不会让两个文件共栖于同一磁盘块。）

（3） 目录结构

目录结构采用树型目录结构。

(a) 目录项内容：

每个目录项 8 个字节，其中：

目录名或文件名：3 个字节；

扩展名：1 个字节（可执行文件扩展名为 e，目录没有扩展名）；

目录属性、文件属性：1 字节；

起始盘号：1 字节；

文件长度：2 字节。

(b) 根目录

根目录位置固定，为磁盘第 2 块，大小固定，共可包含 8 个目录项，占用模拟磁盘第 2 块；

(c) 子目录

位置不固定，大小不固定。

（4） 磁盘分配

磁盘的分配采用链接结构（显式链接）的分配方式。系统采用文件分配表方式记录磁盘空间的使用情况和链接结构的指针。（参考 MSDOS 的 FAT，题目五中也有 FAT 的提示说明）

文件分配表中一项需要 1 字节，而磁盘有 256 块，因而有 256 项，模拟磁盘空间中的第 0、1 块用来存放文件分配表。

（5） 用户接口

用户接口提供用户命令接口，接收用户从键盘键入的命令。如果不使用键盘输入命令的方式，那么就模拟 windows 操作方式，采用“右击快捷菜单”方式提供“创建删除文件/目录，修改属性”等命令，拖动来移动/复制文件等。

要求实现以下命令：

(下面例子中的 $ 只是命令提示符而已，和 UNIX 无关。除 aa 是目录名外，其余均为文件名。你可以根据自己实现方便或喜欢而自定义命令参数。)

需要实现的命令包括：

创建文件：create　　例如 $ create \aa\bb.e

删除文件：delete　　例如 $ delete \aa\yy

显示文件：type　　例如 $ type \zz

拷贝文件：copy　　例如 $ copy \xx \aa\yy

建立目录：mkdir　　例如 $ mkdir \dd

删除空目录：rmdir　　目录非空时，要报错。

可选实现的命令包括：

改变目录路径：chdir

删除目录：deldir（既可删除空目录又可删除非空目录）

移动文件：move

改变文件属性：change

磁盘格式化：format

磁盘分区命令：fdisk

上述命令在实际系统中都是需要建立进程才可以运行的，这里为简单起见，这些命令执行时不必在模拟系统中建立进程，可直接让 Windows 执行你编写的相应函数。

（6） 屏幕显示

如图 1，屏幕显示要求包括：

用户命令接口:用于系统运行时用户用键盘输入或模拟 win 命令；

磁盘目录显示：要求显示磁盘的目录结构；

磁盘使用情况：显示磁盘每一个磁盘块的空间是占用还是空闲。

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。

用链表模拟内存空间分配表。存储管理采用动态分区存储管理方式，采用首次适配、下次适配、最佳适配均可。

（1）模拟系统中，主存部分分为两部分，一部分是系统区，这里只存放进程控制块和内存分配表，一部分是用户区，存放可执行文件。

系统区包括 PCB 区域（最多容纳 10 个 PCB）、内存空间分配表；（长度自己定）

用户区用数组模拟，大小为 512 字节。例如要执行一个 36 字节大的 bb.e，则分配 36 字节给这个 bb 进程。如果 cc.e 需要 110 字节，

太大而无法分配，则 cc.e 无法载入“内存”而等待。

（2）屏幕显示

如图 1，屏幕显示要求包括：内存使用情况示意图，以不同的颜色表示哪些区域未分配或已分配（已分配给哪个进程）。

3、设备管理

设备管理主要包括设备的分配和回收。（可设一张“设备分配表”和设备等待队列）

（1）模拟系统中有 A、B、C 三种独占型设备，A 设备 2 个，B 设备 3 个，C 设备 3 个。（同一种设备的不同实体被认为是相同的）

（2）设备的申请是由于前述可执行文件中的!? ?指令引起，有空闲设备时分配设备，然后进程阻塞，同时设备使用倒计时至 0 后释放设备（不考虑设备具体的 I/O 操作）并唤醒进程继续运行；无空闲设备时阻塞进程，直至其它进程释放设备时才分配设备并使用，设备使用完后唤醒进程。

注意：设备使用倒计时期间，本进程阻塞，需要调度另外一个进程去占用 CPU 执行；假设设备使用完后立即释放该设备，后续指令需再次使用该设备时重新分配。

（3）不考虑死锁。

（4）屏幕显示如图 1 所示，屏幕显示要求包括：每个设备是否被使用，哪个进程在使用该设备，哪些进程在等待使用该设备。

4、进程管理

进程管理主要包括进程调度，进程的创建和撤销、进程的阻塞和唤醒，中断作用的实现。

（1）硬件工作的模拟

（a） 中央处理器的模拟

用函数 CPU()（该函数没有参数）模拟单个中央处理器。

该函数主要负责解释“可执行文件”中的指令。（为简单计，用

户命令接口部分实现的命令不必用 CPU()解释执行）

设一个“程序计数器”，跟踪现在执行到哪条指令。

如果是赋值或加减指令，那么就赋值或加减，并在屏幕上显示中间结果；

如果是!?设备申请指令，那么要和设备管理联系起来，看设备是否可分配，然后采取不同动作。

如果是 end 指令，则释放内存，结束本进程。

考虑到 CPU()函数执行、系统时钟递增、相对时钟（时间片）递减、设备倒计时递减、随机选择新进程诞生等多种任务的同时存在，可能需要多线程编程，才能保证这几种任务的并发。

（b） 主要寄存器的模拟

用全局变量或数组模拟重要寄存器，如数据寄存器（这里可以只设一个 AX，放 x 的值），程序状态寄存器 PSW，指令寄存器 IR，程序计数器 PC 等。

（c） 中断的模拟

I、 中断的发现应该是硬件的工作，但在这里，用在函数 CPU()中检测 PSW 的方式来模拟。

在 CPU()函数中，每执行一条指令之前，先检查 PSW，判断有无中断，若有则先进行中断处理，然后再解释运行指令。

CPU 函数应该不断循环执行。

II、 模拟中断的种类和中断处理方式

发生下述 3 种模拟中断时，分别将 PSW 中的 3 个 bit 设置为 1，处理完中断后将相应 bit 设置为 0。

① 程序结束（执行指令 end 形成的软中断）：在屏幕上输出 x 的值，调用“进程撤销原语”撤销进程，然后进行进程调度；

② 时间片结束（当相对时钟减到 0 时）：将正在运行进程的 CPU 现场（寄存器值即可，暂不考虑“栈”）保存在进程

控制块中，然后进行进程调度；

③ I/O 中断发生（设备使用时间变量倒计时至 0，完成输入输出）：将输入输出完成的进程唤醒，同时将等待该设备的另一个进程唤醒。

（d） 时钟的模拟

系统时钟和相对时钟用全局变量模拟。系统时钟用来记录开机以后的单位时间，相对时钟用来存放进程可运行的时间片（如 6 个单位时间），在进程调度时设置初值，随系统时间的增值 1 而减值 1，减到 0 时，发出时钟中断。

这里的系统时钟并不是计算机的真正时钟，而是单位时间，时间单位长度自定（例如：秒，0.5 秒等），可以使用程序语言里面的一些有关时间的函数来实现。

（2）进程控制块 PCB

进程控制块内容包括进程标识符、主要寄存器、进程状态、阻塞原因等。本模拟系统最多容纳 10 个进程块。PCB 区域用数组模拟。

进程控制块根据内容的不同组成三个队列：空白 PCB 队列，就绪队列和阻塞队列。正在运行的进程只有一个，系统初始时只有空白 PCB 队列。

（3）进程调度

(a) 首先随机选择前面创建的 10 个可执行文件之一，创建进程 PCB，分配内存，然后逐条执行其中的指令；然后经过随机时间后，再选择一个可执行文件，创建进程……，如此往复，模拟操作系统中进程随机到达的过程。

采用时间片轮转调度算法，时间片长度为 6。

(b) 进程调度函数的主要工作是：

将正在运行进程的现场（寄存器组）保存在该进程的 PCB 中；（保存现场）

从就绪队列中选择一个进程；

将这个进程的 PCB 中记录的各寄存器内容恢复到 CPU 各个寄存器内（恢复现场），根据程序计数器 PC 继续执行后续指令。

(c) 闲逛进程

建立一个闲逛进程，当就绪队列为空时，系统调用该进程运行。当有进程就绪时，就调用就绪进程运行。闲逛进程什么有用的事也不做，只是起到系统能正常运转的作用。

（4）进程控制

建立 4 个函数模拟进程创建、撤销、阻塞和唤醒四个原语

(a) 进程创建 create()，主要工作是：

第一步，申请空白进程控制块；

第二步，申请主存空间，申请成功，装入主存；

第三步，初始化进程控制块；

第四步，在屏幕上显示进程执行结果，进程撤销。

(b) 进程撤销 destroy()，主要工作是：

第一步，回收进程所占内存；

第二步，回收进程控制块；

第三步，在屏幕上显示进程执行结果，进程撤销。

(c) 进程阻塞 block()，主要工作是：

第一步，保存运行进程的 CPU 现场；

第二步，修改进程状态；

第三步，将进程链入对应的阻塞队列，然后转向进程调度。

(d) 进程唤醒 awake()

进程唤醒的主要工作是将进程由阻塞队列中摘下，修改进程状态为就绪，然后链入就绪队列。

（5）屏幕显示

如图 1，屏幕显示要求包括：

显示系统时钟；

显示正在运行进程的进程 ID、运行的指令、中间结果（x 的当前

值）、相对时钟（时间片剩余值）；

显示就绪队列中进程 ID；

显示阻塞队列中进程 ID。

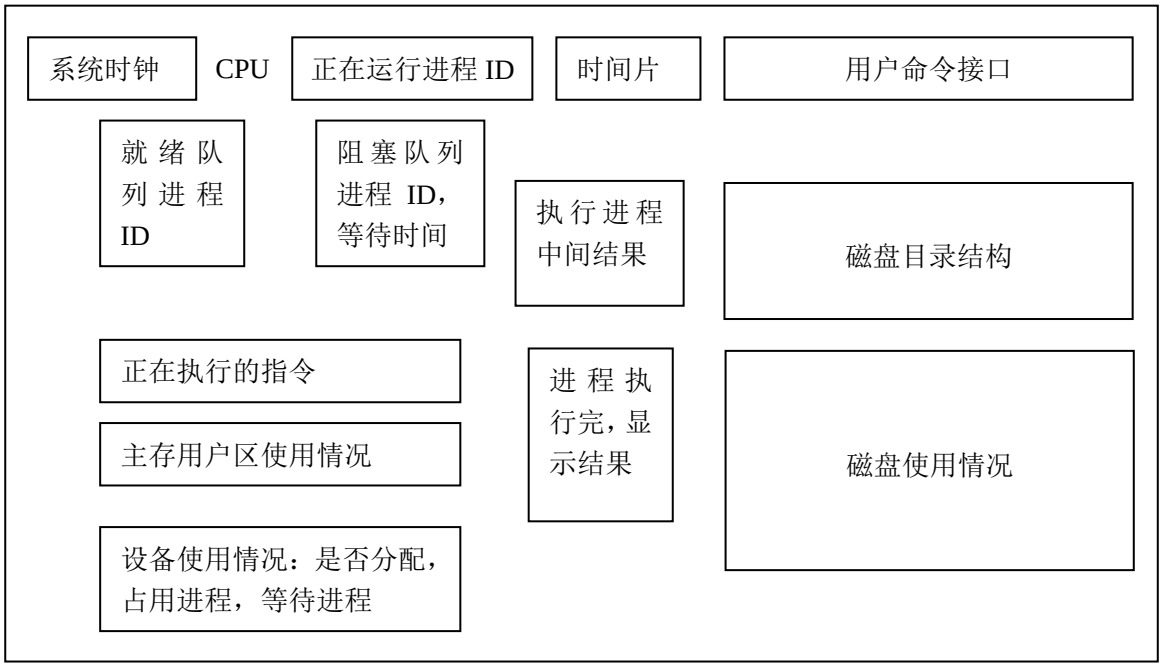

图 1 模拟操作系统的屏幕显示布局和内容