**电子科技大学2013-2014学年第 2学期期 末 考试 A 卷**

**答案及评分细则**

课程名称：计算机操作系统 考试形式： 闭卷 考试日期： 2014年 月 日 考试时长：\_120\_分钟

课程成绩构成（中文班）：平时10 %， 期中 10 %， 实验 10 %， 期末60%，课程设计10 %

课程成绩构成（双语班）：平时20 %， 期中 10 %， 实验 10 %， 期末50%，课程设计10 %

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 题号 | 一 | 二 | 三 | 四 | 五 | 合计 |
| 得分 |  |  |  |  |  |  |

得 分

1. 选择题（共20分, 每题2分，共10题。）
2. 不一定会导致进程切换的是（C）。

A. 时间片结束；B. 从磁盘读数据；C. 子函数调用返回；D. 进程被换出至SWAP分区

1. 当执行系统调用时，进程会进入 (A) 态

A. 内核；B. 用户；C. 就绪；D. 阻塞

1. 下列选项中，降低进程优先级的合理时机是（ Ａ ）

A. 进程的时间片用完 B. 进程刚完成I/O，进入就绪队列

C. 进程长期处于就绪队列 D. 进程从就绪转为运行态

1. 采用SPOOLing技术的目的是（ D ）

A. 减轻用户编程的负担 B. 提高程序的运行速度

C. 提高外存空间的利用率 D. 提高I/O的速度和主机效率

1. 随机存取速度最快文件结构是（D）

A. 堆文件； B. 顺序文件； C. 索引文件； D.直接文件

1. 对目录和文件的描述正确的是（A）

A. 目录也是文件； B. 文件大小只受磁盘容量的限制；

C. 目录中可容纳文件的数量只受磁盘容量的限制； D. 多级目录结构形成一棵严格的多叉树。

1. 假设一个计算进程的生命周期为1小时，I/O设备写一个缓冲区需要10s，计算进程每隔6s读一个缓冲区（读缓冲的时间忽略不计）。如果采取预先写缓冲的方式，缓冲区管理采取循环缓冲，要求计算进程不能因为读缓冲区而被阻塞，那么循环缓冲中至少应该有（ B ）个缓冲区？
2. 180 B. 240 C. 360 D. 480
3. 为了保证一个程序在主存中改变了位置之后仍能正确执行，则对主存空间应采用( B )技术。

A.静态重定位 B.动态重定位

C.编译或汇编 D.存储扩充

1. 某磁盘每条磁道可存储10MB数据，转速为7200rpm，则读取3MB数据的传输时间为（A）

A. 2.5ms； B. 25ms; C. 3.6ms; D. 36ms

1. 通常不采用( D )方法来解除死锁。

A.终止一个死锁进程 B.终止所有死锁进程

C.从死锁进程处抢夺资源 D.从非死锁进程处抢夺资源

得 分

二、填空题（共10分, 每空2分，共5空。）

1. 在某设置有快表的页式存储系统中，快表的命中率为95%，若一次内存存取需100ns，快表访问为10ns，则平均访存时间为（**114.5**）ns。  
   计算方法：95% \* (10 + 100) + (1-95%) \* 2 \* 100 = 114.5ns
2. 操作系统中常见的三种I/O控制方式，按照CPU参与度的减少分别是：程序控制，（**中断**）和（**DMA**）。
3. 若信号量S的初值为2，当前值为-1，则说明有（**1**）个进程在等待此信号量。
4. 若当前目录为/home/knuth/projects/tex/src，则用相对路径表示/home/knuth/projects/metafont/build为（**../../metafont/build**）。

三、应用与计算题（共30分,每题10分，共3题）

得 分

1. 某文件系统采用混合索引的方式组织元数据，包含10个直接块，2个一次间接块和1个两次间接块，若每个物理块为1KB，每个地址项占4字节，请计算本文件系统中可支持的最大文件规模。

答：（1）直接块索引块数：10 (1分)

（2）一次间接块索引块数：1024/4 \* 2 = 512 (3分)

(3)二次间接块索引块数：(1024/4)2 \* 1 = 256 \* 256 = 65536 (5分)

总共容量：(65536 + 512 + 10) \* 1K = 66058K = 64.51MB (1分)

2. 何谓临界资源？若采用以下算法解决哲学家就餐问题，是否会导死锁？为什么？

semaphore fork[5] = {1, 1, 1, 1, 1};

void main()

{

cobegin {

philosopher(0);

philosopher(1);

philosopher(2);

philosopher(3);

philosopher(4);

} coend;

}

void philosopher(int i)

{

while(true) {

thinking();

if (i != 4) {

P(fork[i]);

P(fork[(i+1)%5]);

} else {

P(fork[(i+1)%5]);

P(fork[i]);

}

eating();

if (i != 4) {

V(fork[i]);

V(fork[(i+1)%5]);

} else {

V(fork[(i+1)%5]);

V(fork[i]);

}

}

}

答：所谓临界资源，是指必须互斥使用的资源。即在一段时间内，只允许一个进程使用的资源。（1分）

结论：该算法不会导致死锁。（3分）

原因：它破坏了产生死锁的四个充要条件之一——循环等待条件（3分）。上述算法实质上是对每个临界资源——餐刀进行编号，保证每个哲学家必须按照从小到大的次序依次申请资源，从而不会产生循环等待现象。（3分）

1. 请求分页管理系统中，某进程的页表内容如下：

|  |  |  |
| --- | --- | --- |
| 页号 | 页框号 | 有效位 |
| 0 | 121H | 1 |
| 1 | \_\_\_\_ | 0 |
| 2 | 242H | 1 |

页面大小为4KB，一次内存的访问时间是100ns，一次快表(TLB)的访问时间是10ns，处理一次缺页的平均时间为0.1s(已含更新TLB和页表的时间)，进程的驻留集大小固定为2，采用最近最少使用置换算法(LRU)和局部淘汰策略。假设：(1) TLB初始为空；(2) 地址转换时先访问TLB，若TLB未命中，再访问页表(忽略访问页表之后的TLB更新时间)；(3) 有效位为0表示页面不在内存，产生缺页中断，缺页中断处理后，返回到产生缺页中断的指令处重新执行。请计算依次访问虚地址序列23A3H、15B5H、25C5H所需要的时间，以及虚地址15B5H的物理地址，要求给出计算过程。

答：页面大小L=4K=1000H个字节

虚地址23A3H的页号为:(int)(23A3H/1000H)=2

虚地址15B5H的页号为:(int)(15B5H/1000H)=1

虚地址25C5H的页号为:(int)(25C5H/1000H)=2

以上（2分）

假设访问虚地址序列23A3H、15B5H、25C5H所需要的时间T1、T2、T3，分别计算为：

T1 = 访问TLB的时间＋访问页表时间＋访问内存单元的时间 ＝ 10ns+100ns+100ns=210ns （2分）

T2 = 访问TLB的时间＋访问页表时间＋调页时间+访问TLB的时间＋访问内存单元的时间＝10ns+100ns+100000000+10ns+100ns=100000220ns （2分）

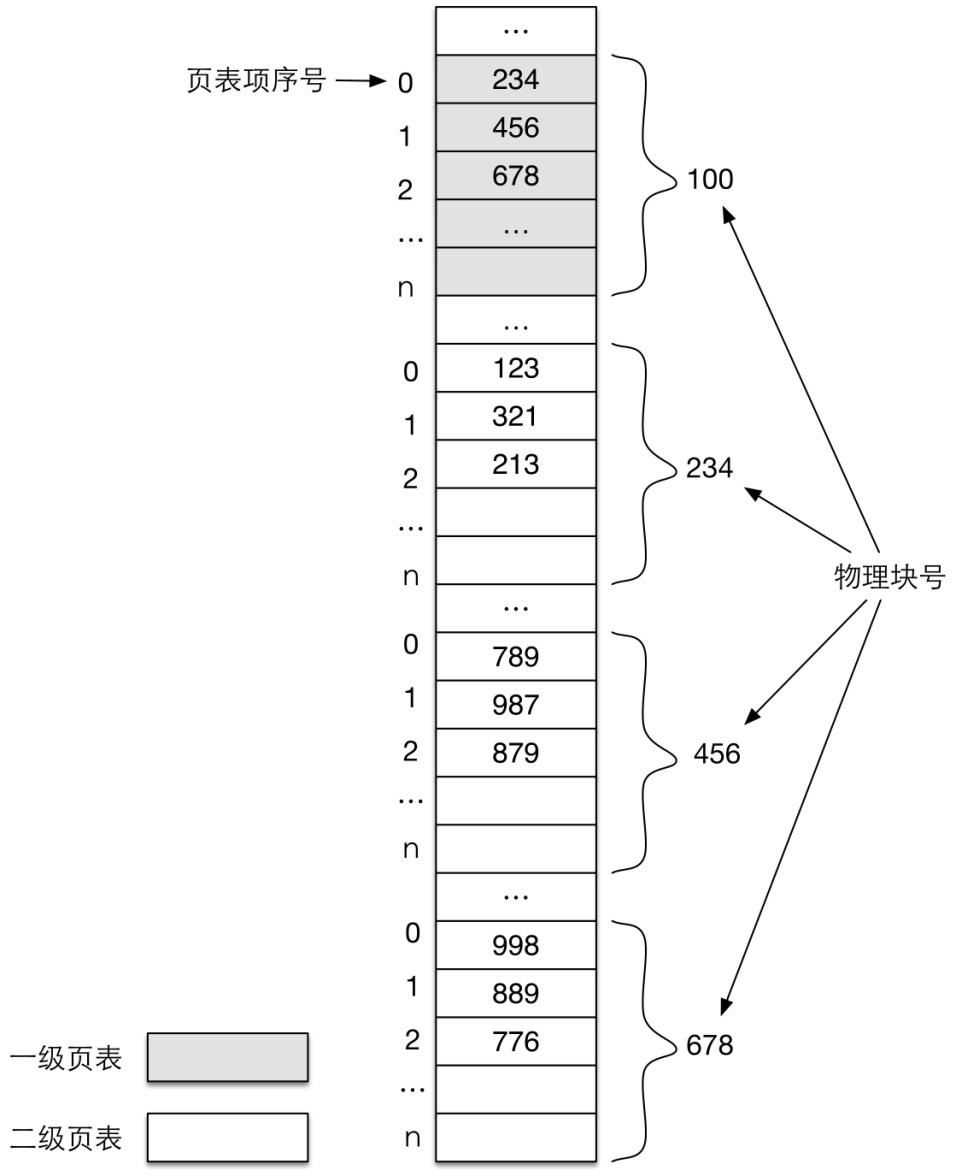
T2 = 访问TLB的时间＋访问内存单元的时间＝10ns+100ns=110ns（2分）

15B5H的页号为1，按照LRU算法，其分配的页框号为121H。因其页内偏移量为5B5H，故物理地址为1215B5H. （2分）

得 分

四、综合分析题（ 共40分，每题20分，共2题）

1. 某32位存储系统采用页式管理，页表分为两级，存储在主存中。每个页面1KB，每个页表项占4字节，某进程的页表内容如下图所示（图中数字全部为10进制）。请计算逻辑地址526245（十进制）对应的物理地址。



答：每个进程的总页面数为232B/1KB=222个，每个页面可存储页表项1KB/4B=256=28个，存储二级页表所需要的页面数为：222/28=214个。

所以32位逻辑地址的划分为：

0->9位：页内偏移地址

10->17位：二级页表中页表项序号

18->31位：一级页表中二级页表序号

（以上4分）

526245对应的二进制表示（斜线表示上述逻辑切分）：10/00 0000 01/11 1010 0101

按照上述划分方法得到：二级页表在一级页表中的序号为2，二级页表中页表项序号1，页内地址0x3A5H。 （7分）

一级页表第2项查找到二级页表的物理块号：678， （3分）

在678物理块的第1项找到物理块号889（0x379H），形成物理地址块号， （4分）

与页内地址0x3A5H拼接，形成完整的地址：1101 1110 01/11 1010 0101🡪0xDE7A5H（十进制：911269）（2分）

2.机场中有些道路，航班和摆渡车都可以通行。当航班通行时，摆渡车必须等待，直到所有航班通过后才能通行。当摆渡车通行时，航班必须等待，直到所有摆渡车通过后才能通行。请用信号量描述摆渡车和航班通行的控制过程。

答：

（定义6分）

semaphore wait=1; // 同步航班和摆渡车

semaphore mutex=1; // 互斥道路

int count\_shuttle=0, count\_flight=0; // 连续通行车辆和航班计数

semaphore flight=1; // 航班数计数互斥访问

semaphore shuttle=1; // 摆渡车计数互斥访问

航班：（7分）

P(flight)

count\_flight = count\_flight + 1;

if count\_flight = 1 then P(wait);

V(flight)

P(mutex);

通行；

V(mutex);

P(flight)

count\_flight = count\_flight + 1;

if count\_flight = 1 then V(wait);

V(flight)

摆渡车：（7分）

P(shuttle);

count\_shuttle = count\_shuttle + 1;

if count\_shuttle = 1 then P(wait);

V(shuttle);

P(mutex);

通行；

V(mutex);

P(shuttle);

count\_shuttle = count\_shuttle - 1;

if count\_shuttle = 0 then V(wait);

V(shuttle);