# 第一章 引论

# 第二章 进程管理

**24. 在生产者-消费者问题中,如果将两个wait操作即wait(full)和wait(mutex)互换位置;或者是将**　　signal(mutex)与signal(full)互换位置结果会如何

　　a. wait(full)和wait(mutex)互换位置后,因为mutex在这儿是全局变量,执行完wait(mutex),则mutex赋值为0,倘若full也为0,则该生产者进程就会转入进程链表进行等待,而生产者进程会因全局变量mutex为0而进行等待,使full始终为0,这样就形成了死锁.

　　b. 而signal(mutex)与signal(full)互换位置后,从逻辑上来说应该是一样的.

**25 我们为某临界区设置一把锁W,当W=1时,表示关锁;W=0时,表示锁已打开.试写出开锁原语和关锁原语,并利用它们去实现互斥.**

　　开锁原语:

　　unlock(W):

　　W=0;

　　关锁原语:

　　lock(W);

　　if(W==1) do no\_op;

　　W=1;

　　利用开关锁原语实现互斥:

　　var W: semaphore:=0;

　　begin

　　parbegin

　　process :

　　begin

　　repeat

　　lock(W);

　　critical section

　　unlock(W);

　　remainder section

　　until false;

　　end

parend

**26. 试修改下面生产者－消费者问题解法中的错误:**

producer:

begin

repeat

.

.

producer an item in nextp;

wait(mutex);

wait(full); /\* 应为wait(empty),而且还应该在wait(mutex)的前面 \*/

buffer(in):=nextp;

/\* 缓冲池数组游标应前移: in:=(in+1) mod n; \*/

signal(mutex);

/\* signal(full); \*/

until false;

end

consumer:

begin

repeat

wait(mutex);

wait(empty); /\* 应为wait(full),而且还应该在wait(mutex)的前面 \*/

nextc:=buffer(out);

out:=out+1; /\* 考虑循环，应改为: out:=(out+1) mod n; \*/

signal(mutex);

/\* signal(empty); \*/

consumer item in nextc;

until false;

end

**27 试利用记录型信号量写出一个不会出现死锁的哲学家进餐问题的算法.**

设初始值为1的信号量c[I]表示I号筷子被拿(I=1,2,3,4,...,2n),其中n为自然数.

send(I):

Begin

if I mod 2==1 then

{

P(c[I]);

P(c[I-1 mod 5]);

Eat;

V(c[I-1 mod 5]);

V(c[I]);

}

else

{

P(c[I-1 mod 5]);

P(c[I]);

Eat;

V(c[I]);

V(c[I-1 mod 5]);

}

End

**31 什么是AND信号量?试利用AND信号量写出生产者－消费者问题的解法.**

为解决并行所带来的死锁问题，在wait操作中引入AND条件，其基本思想是将进程在整个运行过程中所

需要的所有临界资源，一次性地全部分配给进程，用完后一次性释放.

解决生产者－消费者问题可描述如下:

var mutex,empty,full: semaphore:=1,n,0;

buffer: array[0,...,n-1] of item;

in,out: integer:=0,0;

begin

parbegin

producer: begin

repeat

.

.

produce an item in nextp;

.

.

wait(empty);

wait(s1,s2,s3,...,sn); //s1,s2,...,sn为执行生产者进程除empty外其余的条件

wait(mutex);

buffer(in):=nextp;

in:=(in+1) mod n;

signal(mutex);

signal(full);

signal(s1,s2,s3,...,sn);

until false;

end

consumer: begin

repeat

wait(full);

wait(k1,k2,k3,...,kn); //k1,k2,...,kn为执行消费者进程除full外其余的条件

wait(mutex);

nextc:=buffer(out);

out:=(out+1) mod n;

signal(mutex);

signal(empty);

signal(k1,k2,k3,...,kn);

consume the item in nextc;

until false;

end

parend

end

**41 什么是用户级线程和内核级线程?并对它们进行比较.**

a. 内核级线程是依赖于内核的，它存在于用户进程和系统进程中，它们的创建，撤消和切换都由内核

实现；

---用户级线程仅存在于用户级中，它们的创建，撤消和切换不利用系统调用来实现，因而与内核无关，

内核并不知道用户级线程的存在.

b. 内核级线程的调度和切换与进程十分相似，调度方式采用抢占式和非抢占式，调度算法采用时间轮转

法和优先权算法等，当由线程调度选中一个线程后，再将处理器分配给它；而用户级线程通常发生在

一个应用程序的诸线程之间，无需终端进入OS内核，切换规则也较简单，因而，用户级线程的切换

速度较快.

---用户级线程调用系统调用和调度另一个进程执行时，内核把它们看作是整个进程的行为，内核级线程

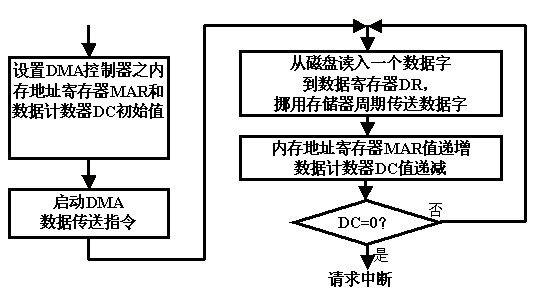
调用是以线程为单位，内核把系统调用看作是该线程的行为.

---对于用户级线程调用，进程的执行速度随着所含线程数目的增加而降低，对于内核级线程则相反.

# 第五章 文件管理

**8．试说明DMA的工作流程。**

以从磁盘读入数据为例来说明DMA方式的工作流程：当CPU要从磁盘读入一数据块时，便向磁盘控制器发送一条读命令，该命令被送入DMA控制器的命令寄存器CR中。同时，还需发送本次要将数据读入的内存起始目标地址，该地址被送入DMA控制器的内存地址寄存器MAR中；本次要读的字（节）数则送至DMA控制器的数据计数器DC中。另外，还需将磁盘中数据读取的源地址直接送到DMA控制器的I/O控制逻辑上。然后，启动DMA控制器进行数据传送。此后，CPU便可去处理其它任务，而整个的数据传送便由DMA控制器负责控制。当DMA控制器已从磁盘中读入一个字（节）的数据，并送入DMA控制器的数据寄存器DR后，再挪用一个存储器周期，将该字（节）传送到MAR所指示的内存单元中。接着，便对MAR内容加1和将DC内容减1。若DC内容减1后不为0，表示传送未完，便准备再传送下一个字（节），否则，由DMA控制器发出中断请求。参图所示：



**14 在具有快表的段页式存储管理方式中，如何实现地址变换?**

首先，必须配置一段表寄存器，在其中存放段表始址和段长TL. 进行地址变换时，先利用段号S，与段长TL进行比较，若S<TL，表示未越界，(若S>=TL,表示段号太大，访问越界，产生越界中断信号)于是利用段表始址和段号来求出该段对应的段表项在段表中的位置，从中求出该段的页表始址，并利用逻辑地址中的段内页号P来获得对应页的页表项位置，从中读出该页所在的物理块号b，再用块号b和页内地址构成物理地址.

**26 在一个请求分页系统中，采用LRU页面置换算法时，假如一个作业的页面走向**

---为4，3，2，1，4，3，5，4，3，2，1，5，当分配给该作业的物理块数M分别

---为3和4时，试计算访问过程中所发生的缺页次数和缺页率?比较所得结果?

a. 当分配给该作业的物理块数M为3时，所发生的缺页率为10，缺页率为: 10/12=0.583；

b. 当分配给该作业的物理块数M为4时，所发生的缺页率为8，缺页率为: 8/12=0.333.

# 第六章文件系统

**14. 在UNIX 中，如果一个盘块的大小为1KB，每个盘块号占4个字节，即每块可放256个地址。请转换下列文件的字节偏移量为物理地址：(1)9999；(2)18000；(3)420000。**

**答：** 步1 将逻辑文件的字节偏移量转换为文件的逻辑块号和块内偏移。方法是：将逻辑文件的字节偏移量/盘块大小，商为文件的逻辑块号，余数是块内偏移。

步2将文件的逻辑块号转换为物理块号。使用多重索引结构，在索引节点中根据逻辑块号通过直接索引或间接索引找到对应物理块号。

(1) 9999 L1=INT(9999，1024)=9 B1=MOD(9999，1024)=783

其逻辑块号为9，故直接索引addr[8]中可找到物理块号。

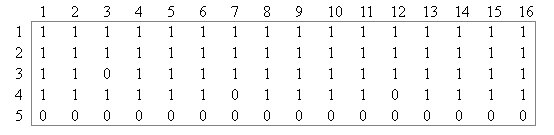
(2) 18000 L2=INT(18000，1024)=17 B1=MOD(18000，1024)=592

其逻辑块号为17，通过一次间接索引addr[10]中可找到物理块号。

(3) 420000 L1=INT(420000，1024)=410 B1=MOD(9000，1024)=160

其逻辑块号为410，通过二次间接索引addr[11]中可找到物理块号。

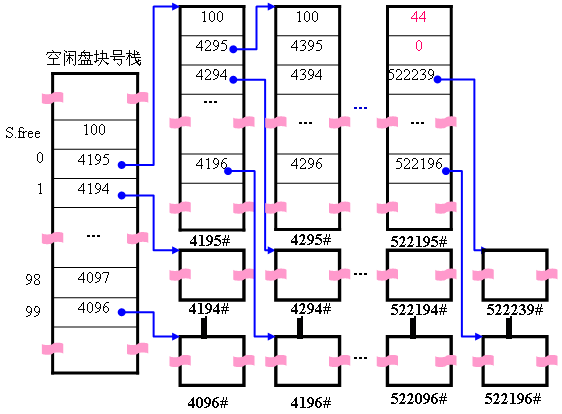
**23．计算机系统利用下面的位示图来管理空闲盘块，盘块大小为1KB，现要为某文件分配两个盘块，试说明盘块分配的具体过程。**



根据位示图为某文件分配两个盘块的具体过程如下：（1）顺序扫描位示图，找出两个其值均为空闲即0的二进制位Map[3, 3]，Map[4, 7]；（2）将二进制位Map[3, 3]和Map[4, 7]的行/列号转换为与之对应的盘块号35、55；（3）把盘块号为35和55的盘块分配给该文件，同时修改位示图中的二进制位Map[3, 3]=1，Map[4, 7]=1。

**补1．某计算机系统磁盘容量为520MB，盘块大小为1KB。其中前4MB用于存放索引结点等，后10MB用作对换区，采用成组链接法管理外存空间，每组100个盘块。试画出外存尚未使用的成组链接图。**

根据题意，该计算机系统尚未使用的外存空间为（520MB－4MB－10MB = 506MB），即506K（也就是518144）个盘块，其盘块号为4K#~510K#（即4096# ~ 522239#）。而每组100个盘块，故共有5182个盘块组，其中最后一个盘块组含44个盘块。因此，外存尚未使用的成组链接图如下所示：



**补2**. 一个UNIX/Linux文件，如果一个盘块的大小为1KB，每个盘块占4个字节，那么，若进程欲访问偏移为263168字节处的数据，需经过几次间接寻址？

**答：**UNIX/Linux文件系统中，直接寻址为10块，一次间接寻址为256块，二次间接寻址为2562块，三次间接寻址为2563块。

偏移为263168字节的逻辑块号是：263168/1024=257。块内偏移量=263168-257×1024=0。由于10<257<256+10，故263168字节在一次间接寻址内。