第一章

1. 什么是操作系统的基本功能 ?

答：操作系统的职能是管理和控制计算机系统中的所有硬、 软件资源， 合理地组织计算机工作流程， 并为用户提供一个良好的工作环境和友好的接口。 操作系统的基本功能包括：处理机管理、存储管理、设备管理、信息管理 (文件系统管理) 和用户接口等。

2. 什么是批处理、分时和实时系统 ?各有什么特征 ?

答：批处理系统 (batch processingsystem) ：操作员把用户提交的作业分类，把一批作业编成一个作业执行序列，由专门编制的监督程序 (monitor) 自动依次处理。其主要特征是：用户脱机使用计算机、成批处理、多道程序运行。

分时系统 (time sharing operation system) ：把处理机的运行时间分成很短的时间片， 按时间片轮转的方式， 把处理机分配给各进程使用。 其主要特征是：交互性、多用户同时性、独立性。

实时系统 (real time system) ：在被控对象允许时间范围内作出响应。其主要特征是： 对实时信息分析处理速度要比进入系统快、 要求安全可靠、 资源利用率低。

3. 多道程序设计和多重处理有何区别 ?

答 ;多道程序 (multiprogramming) 是作业之间自动调度执行、 共享系统资源，并不是真正地同时执行多个作业；而多重处理 (multiprocessing) 系统配置多个 CPU 能真正同时执行多道程序。要有效使用多重处理，必须采用多道程序设计技术，而多道程序设计原则上不一定要求多重处理系统的支持。

4. 讨论操作系统可以从哪些角度出发，如何把它们统一起来？

答；讨论操作系统可以从以下角度出发：（ 1）操作系统是计算机资源的管理者；（2）操作系统为用户提供使用计算机及界面； （3）用进程管理观点研究操作系统，即围绕进程运行过程来讨论操作系统。

上述这些观点彼此并不矛盾， 分别代表了从不同角度对同一事物 （操作系统）的观点。每种观点都有助于理解、分析和设计操作系统。

6. 设计计算机操作系统与哪些硬件器件有关？

答；计算机操作系统的重要功能之一是对硬件资源的管理。 因此设计计算机操作系统时应考虑下述计算机硬件资源：（ 1）CPU与指令的长度与执行方式；（ 2）内存、缓存和高速缓存等存储装置；（ 3）各类寄存器，包括各种通用寄存器、控制寄存器和状态寄存器； （4）中断机构；（5）外部设备与 I/O 控制装置；（6）内部总线与外部总线；（ 7）对硬件进行操作的指令集。

第二章

1. 什么是作业 ?作业步？

答：把在一次应用业务处理过程中， 从输入开始到输出结束， 用户要求计算机所做的有关该次业务处理的全部工作称为一个作业。 作业由不同的顺序相连的作业步组成。作业步是在一个作业的处理过程中， 计算机所做的相对独立的工作。如，编辑输入是一个作业步，它产生源程序文件；编译也是一个作业步，它产生目标代码文件。

2. 作业由哪几部分组成 ？各有什么功能 ？

答：作业由三部分组成：程序、数据和作业说明书。程序和数据完成用户所要求的业务处理工作，作业说明书则体现用户的控制意图。

3. 作业的输入方式有哪几种 ?各有何特点

答：作业的输入方式有 5 种：联机输入方式、脱机输入方式、直接耦合方式、SPOOLING(Simultaneous Peripheral Operations Online) 系统和网络输入方式，各有如下特点：

(1) 联机输入方式：用户和系统通过交互式会话来输入作业。

(2) 脱机输入方式： 又称预输入方式， 利用低档个人计算机作为外围处理机进行输入处理， 存储在后备存储器上， 然后将此后援存储器连接到高速外围设备上和主机相连，从而在较短的时间内完成作业的输入工作。

(3) 直接耦合方式：把主机和外围低档机通过一个公用的大容量外存直接耦合起来，从而省去了在脱机输入中那种依靠人工干预来传递后援存储器的过程。

(4) SPOOLING 系统：可译为外围设备同时联机操作。在 SPOOLING 系统中，多台外围设备通过通道或 DMA 器件和主机与外存连接起来， 作业的输入输出过程由主机中的操作系统控制。

(5) 网络输入方式： 网络输入方式以上述几种输入方式为基础， 当用户需要把在计算机网络中某一台主机上输入的信息传送到同一网中另一台主机上进行操作或执行时，就构成了网络输入方式。

4. 试述 SPOOLING 系统的工作原理。

答：在 SPOOLING 系统中，多台外围设备通过通道或 DMA 器件和主机与外存连接起来， 作业的输入输出过程由主机中的操作系统控制。 操作系统中的输入程序包含两个独立的过程， 一个过程负责从外部设备把信息读入缓冲区， 另一个过程是写过程，负责把缓冲区中的信息送入到外存输入井中。

在系统输入模块收到作业输入请求后， 输入管理模块中的读过程负责将信息从输入装置读入缓冲区。 当缓冲区满时， 由写过程将信息从缓冲区写到外存输入井中。读过程和写过程反复循环， 直到一个作业输入完毕。 当读过程读到一个硬件结束标志后，系统再次驱动写过程把最后一批信息写入外存并调用中断处理程序结束该次输入。然后，系统为该作业建立作业控制块 JCB，从而使输入井中的作业进入作业等待队列，等待作业调度程序选中后进入内存。

5. 操作系统为用户提供哪些接口 ?它们的区别是什么 ？

答 :操作系统为用户提供两个接口， 一个是系统为用户提供的各种命令接口，用户利用这些操作命令来组织和控制作业的执行或管理计算机系统。 另一个接口是系统调用， 编程人员使用系统调用来请求操作系统提供服务， 例如申请和释放外设等类资源、控制程序的执行速度等。

6. 作业控制方式有哪几种 ?调查你周围的计算机的作业控制方式。

答：作业控制的主要方式有两种：脱机方式和联机方式。

脱机控制方式利用作业控制语言来编写表示用户控制意图的作业控制程序，也就是作业说明书。 作业控制语言的语句就是作业控制命令。 不同的批处理系统提供不同的作业控制语言。

联机控制方式不同于脱机控制方式，它不要求用户填写作业说明书，系统只为用户提供一组键盘或其他操作方式的命令。 用户使用操作系统提供的操作命令和系统会话，交互地控制程序执行和管理计算机系统。

7．什么是系统调用 ?系统调用与一般用户程序有什么区别 ?与库函数和实用程序又有什么区别 ?

答：系统调用是操作系统提供给编程人员的唯一接口。 编程人员利用系统调用，在源程序一级动态请求和释放系统资源， 调用系统中已有的系统功能来完成那些与机器硬件部分相关的工作以及控制程序的执行速度等。 因此，系统调用像一个黑箱子那样，对用户屏蔽了操作系统的具体动作而只提供有关的功能。

系统调用与一般用户程序、 库函数和实用程序的区别是： 系统调用程序是在核心态执行，调用它们需要一个类似于硬件中断处理的中断处理机制来提供系统服务。

8．简述系统调用的实现过程。

答：用户在程序中使用系统调用， 给出系统调用名和函数后， 即产生一条相应的陷入指令， 通过陷入处理机制调用服务， 引起处理机中断， 然后保护处理机现场，取系统调用功能号并寻找子程序入口， 通过入口地址表来调用系统子程序，然后返回用户程序继续执行。

9．为什么说分时系统没有作业的概念？

答：因为咋分时系统中， 每个用户得到的时间片有限， 用户的程序和数据信息直接输入到内存工作区中和其它程序一起抢占系统资源投入执行， 而不必进入外存输入井等待作业调度程序选择。 因此，分时系统没有作业控制表， 也没有作业调度程序。

第三章

1．有人说，一个进程是由伪处理机执行的一个程序，这话对吗？为什么？

答：对。因为伪处理机的概念只有在执行时才存在， 它表示多个进程在单处理机上并发执行的一个调度单位。 因此，尽管进程是动态概念， 是程序的执行过程，但是，在多个进程并发执行时，仍然只有一个进程占据处理机执行，而其它并发进程则处于就绪或等待状态。这些并发进程就相当于由伪处理机执行的程序。

2．试比较进程和程序的区别。

答：(1) 进程是一个动态概念， 而程序是一个静态概念， 程序是指令的有序集合，无执行含义，进程则强调执行的过程。

(2) 进程具有并行特征（独立性，异步性），程序则没有。

(3) 不同的进程可以包含同一个程序， 同一程序在执行中也可以产生多个进程。

4．试比较作业和进程的区别。

答：并非对所有的程序均成立。例如：

```
begin
    local x
    x:=10
    print(x)
end
```

上述程序中 x 是内部变量， 不可能被外部程序访问， 因此这段程序的运行不会受外界环境影响。

4. 试比较作业和进程的区别。

答 :一个进程是一个程序对某个数据集的执行过程， 是分配资源的基本单位。作业是用于需要计算机完成某项任务， 而要求计算机所做工作的集合。 一个作业的完成要经过作业提交， 作业收容、 作业执行和作业完成 4 个阶段。 而进程是已提交完毕的程序所执行过程的描述， 是资源分配的基本单位。 其主要区别关系如下：

(1) 作业是用户向计算机提交任务的任务实体。 在用户向计算机提交作业之后，系统将存储在外存中的作业等待队列中等待执行。 而进程则是完成用户任务的执行实体，是向系统申请分配资源的基本单位。任一进程，只要它被创建，总有相应的部分存在于内存中。

(2) 一个作业可由多个进程组成。 且必须至少由一个进程组成， 但反过来不成立。

(3) 作业的概念主要用在批处理系统中。像 Unix 这样的分时系统中，则没有作业概念。而进程的概念则用在几乎所有的多道程序系统中。

6. 什么是临界区 ?试举一临界区的例子。

答：临界区是指不允许多个并发进程交叉执行的一段程序。 它是由于不同并发进程的程序段共享公用数据或公用数据变量而引起的。 所以它又被称为访问公用数据的那段程序。

例如：

```
getspace        :
begin local g
            g=stact[top]
           top=top-1
End
release(ad)          :
Begin
  top=top+1
  stack[top]          ＝ad
End
```

7. 并发进程间的制约有哪两种 ?引起制约的原因是什么 ？

答：并发进程所受的制约有两种：直接制约和间接制约。

直接制约是由并发进程互相共享对方的私有资源所引起的。

间接制约是由竞争共有资源而引起的。

8. 什么是进程间的互斥 ?什么是进程间同步 ?

答：进程间的互斥是指： 一组并发进程中的一个或多个程序段， 因共享某一公有资源而导致它们必须以一个不许交叉执行的单位执行， 即不允许两个以上的共享该资源的并发进程同时进入临界区。

进程间的同步是指：异步环境下的一组并发进程因直接制约互相发送消息而进行互相合作、互相等待，使得各进程按一定的速度执行的过程。

9. 试比较 P, V 原语法和加锁法实现进程间互斥的区别。

答：互斥的加锁实现是这样的： 当某个进程进入临界区之后， 它将锁上临界区，直到它退出临界区时为止。 并发进程在申请进入临界区时， 首先测试该临界区是否是上锁的， 如果该临界区已被锁住， 则该进程要等到该临界区开锁之后才有可能获得临界区。

加锁法存在如下弊端：(1) 循环测试锁定位将损耗较多的 CPU计算时间；(2)产生不公平现象。

P, V 原语法采用信号量管理相应临界区的公有资源， 信号量的数值仅能由 P, V 原语操作改变，而 P, V 原语执行期间不允许中断发生。其过程是这样的：当某个进程正在临界区内执行时，其他进程如果执行了 P 原语，则该进程并不像 lock 时那样因进不了临界区而返回到 lock 的起点，等以后重新执行测试，而是在等待队列中等待由其他进程做 V 原语操作释放资源后，进入临界区，这时 P 原语才算真正结束。若有多个进程做 P原语操作而进入等待状态之后，一旦有 V 原语释放资源，则等待进程中的一个进入临界区，其余的继续等待。

总之，加锁法是采用反复测试 lock 而实现互斥的，存在 CPU浪费和不公平现像， P, V 原语使用了信号量，克服了加锁法的弊端。

10.

答：设第 I 块缓冲区的公有信号量为 metex[I] ，保证生产者进程和消费者进程对同一块缓冲区操作的互斥，初值为 1；

设信号量 avail 为生产者进程的私有信号量，初值为 m;

设信号量 full 为消费者进程的私有信号量，初值为 0。

用信号量和 P、V操作描述发送过程 deposit(data) 和接收过程 remove(data) 如下：

```
deposit(data) ：
        begin
        P(avail)
            选择一个空缓冲区 i
        P(mutex[i])
            送数据入缓冲区 i
        V(mutex[i])
        V(full)
End
```

remove(data) ：

begin

P(full)

选择一个满缓冲区 i

P(mutex[i])

取缓冲区 i 中的数据

V(mutex[i])

V(avail)

End

12.

答：定义数组 buf[0],buf[1] 。

设 bufempty[0],buffull[1] 是 Pa的公有信号量；

设 bufempty[1],buffull[0] 是 Pb的公有信号量；

初值为 :

bufempty[0]= bufempty[1]=n

buffull[0]= buffull[1]=0

用信号量和 P、V操作描述发送过程 send(i,m) 和接收过程 receive(i,m) 如下：

send (i,m) ：

begin

local x

P(bufempty[i])

按 FIFO 选择一个空缓冲区 buf[i](x)

buf[i](x)=m

buf[i](x) 置满标记

V(buffull[i])

End

receive(i,m) ：

begin

local x

P(buffull[i])

按 FIFO 选择一个满缓冲区 buf[i](x)

m=buf[i](x)

buf[i](x) 置空标记

V(bufempty[i])

End

Pa调用 send(0,m) 和 receive(1,m)

Pb调用 send(1,m) 和 receive(0,m)

14.

答：设信号量 c[i] ，初值为 1；i=0,1,2,3,4 。i 表示第 i 号筷子。

（1）第 i 个哲学家要吃饭：

eat(i) ：

begin

P(c[i])

P(c[i+1 mod 5])
吃饭
V(c[i+1 mod 5])
V(c[i])
End

该过程能保证两个邻座不同时吃饭，　但有可能出现每人只拿到一支筷子，　谁也吃不上饭的情况。

（1）为解决上述情况，让奇数号的哲学家先取右手边的筷子，偶数号的哲学家先取左手边的筷子。 这样只要有一个哲学家拿到了一支筷子，　就阻止了邻座的哲学家吃法的企图，从而不会死锁，除非某哲学家永远吃下去。

算法描述如下：

```
eat(i)　　：
                begin
        if i mod 2 == 0 then
                    {
                  P(c[i])
                        P(c[i+1 mod 5])
                                吃饭
                        V(c[i+1 mod 5])
                        V(c[i])
                    }
                    else
                    {
                        P(c[i+1 mod 5])
                  P(c[i])
                                吃饭
                        V(c[i])
                        V(c[i+1 mod 5])
                    }
                End
```

另解：最多只允许　4 个哲学家同时要求进餐，这样至少有一位哲学家能取到两只筷子并可以进餐，进餐后释放两只筷子，其他哲学家可以陆续进餐。

设哲学家进餐信号量　sm=4;筷子信号量　c[i]=1(i=0,1,2,3,4)

```
eat(i)　　：
                begin
            P(sm)
                P(c[i])
                P(c[i+1 mod 5])
                        吃饭
                V(c[i+1 mod 5])
                V(c[i])
            V(sm)
                End
```

15. 什么是线程 ？ 试述线程与进程的区别。

答；线程是在进程内用于调度和占有处理机的基本单位，它由线程控制表、存储线程上下文的用户栈以及核心栈组成。 线程可分为用户级线程、 核心级线程以及用户／核心混合型线程等类型。其中用户级线程在用户态下执行， CPU调度算法和各线程优先级都由用户设置， 与操作系统内核无关。 核心级线程的调度算法及线程优先级的控制权在操作系统内核。 混合型线程的控制权则在用户和操作系统内核二者。

线程与进程的主要区别有：

(1) 进程是资源管理的基本单位， 它拥有自己的地址空间和各种资源， 例如内存空间、 外部设备等； 线程只是处理机调度的基本单位， 它只和其他线程一起共享进程资源，但自己没有任何资源。

(2) 以进程为单位进行处理机切换和调度时， 由于涉及到资源转移以及现场保护等问题， 将导致处理机切换时间变长， 资源利用率降低。 以线程为单位进行处理机切换和调度时， 由于不发生资源变化， 特别是地址空间的变化， 处理机切换的时间较短，从而处理机效率也较高。

(3) 对用户来说， 多线程可减少用户的等待时间。 提高系统的响应速度。 例如，当一个进程需要对两个不同的服务器进行远程过程调用时， 对于无线程系统的操作系统来说需要顺序等待两个不同调用返回结果后才能继续执行， 且在等待中容易发生进程调度。 对于多线程系统而言， 则可以在同一进程中使用不同的线程同时进行远程过程调用，从而缩短进程的等待时间。

(4) 线程和进程一样，都有自己的状态．也有相应的同步机制，不过，由于线程没有单独的数据和程序空间， 因此， 线程不能像进程的数据与程序那样， 交换到外存存储空间。从而线程没有挂起状态。

(5) 进程的调度、同步等控制大多由操作系统内核完成， 而线程的控制既可以由操作系统内核进行，也可以由用户控制进行。

思考题：读者与写者关系问题（读者优先）。

答：设写者互斥信号量 wm=1;读者计数器 readcount=0; 互斥操作 readcount 的信号量 rm=1;

```
reader()
begin
  P(rm)
     Readcount:=readcount+1
     If readcount==1 then P(wm)
  V(rm)
   读数据
  P(rm)
     Readcount:=readcount-1
     If readcount==0 then V(wm)
  V(rm)
end
writer()
begin
  P(wm)
        写数据
  V(wm)
```

end

第四章

1. 什么是分级调度 ?分时系统中有作业调度的概念吗 ?如果没有, 为什么？ P86

答：处理机调度问题实际上也是处理机的分配问题。 显然只有那些参与竞争处理及所必需的资源都已得到满足的进程才能享有竞争处理机的资格。 这时它们处于内存就绪状态。这些必需的资源包括内存、外设及有关数据结构等。从而, 在进程有资格竞争处理机之前, 作业调度程序必须先调用存储管理、 外设管理程序，并按一定的选择顺序和策略从输入井中选择出几个处于后备状态的作业, 为它们分配资源和创建进程，使它们获得竞争处理机的资格。

另外，由于处于执行状态下的作业一般包括多个进程, 而在单机系统中, 每一时刻只能有一个进程占有处理机, 这样, 在外存中, 除了处于后备状态的作业外，还存在处于就绪状态而等待得到内存的作业。 我们需要有一定的方法和策略为这部分作业分配空间。因此处理机调度需要分级。

一般来说，处理机调度可分为 4 级：

(1) 作业调度： 又称宏观调度，或高级调度。

(2) 交换调度：又称中级调度。其主要任务是按照给定的原则和策略，将处于外存交换区中的就绪状态或等待状态的进程调入内存, 或处于内存就绪状态或等待状态的进程交换到外存交换区。交换调度主要涉及到内存管理与扩充。

(3) 进程调度：又称微观调度或低级调度。其主要任务是按照某种策略和方法选取一个处于就绪状态的进程占用处理机。在确立了占用处理机的进程之后，系统必须进行进程上下文切换以建立与占用处理机进程相适应的执行环境。

(4) 线程调度：进程中相关堆栈和控制表等的调度。

2. 试述作业调度的主要功能。 P88

答: 作业调度的主要功能是：

（1）记录系统中各作业的状况。

（2）按一定的原则对外存输入井上的大量后备作业进行选择。

（3）给选出的作业分配内存、输入输出设备等必要的资源，并建立相应进程，使该作业的相关进程获得竞争处理机的权利。

（4）当作业执行完毕时，还负责回收系统资源。

3. 作业调度的性能评价标准有哪些 ?这些性能评价标准在任何情况下都能反映调度策略的优劣吗 ？

答：调度的性能评价标准：

（1）对所有作业应该是公平合理的；

（2）应使设备有高的利用率；

（3）每天执行尽可能多的作业；

（4）有快的响应时间。

不能。对于批处理系统, 由于主要用于计算, 因而对于作业的周转时间要求较高。从而作业的平均周转时间或平均带权周转时间被用来衡量调度程序的优劣。但对于分时系统来说, 平均响应时间又被用来衡量调度策略的优劣。 对于分时系统, 除了要保证系统吞吐量大、 资源利用率高之外, 还应保证用户能够容忍

的响应时间。 因此， 在分时系统中， 仅仅用周转时间或带权周转时间来衡量调度性能是不够的。

对于实时系统，衡量调度算法优劣的主要标志则是满足用户要求的时限时间。

4. 进程调度的功能有哪些 ？ P91

答：进程调度的功能有：

(1) 记录和保存系统中所有进程的执行情况；

(2) 选择占有处理机的进程；

(3) 进行进程上下文切换。

5. 进程调度的时机有哪几种 ？ P92

答：进程调度的时机有：

(1) 正在执行的进程执行完毕。 这时如果不选择新的就绪进程执行， 将浪费处理机资源。

(2) 执行中进程自己调用阻塞原语将自己阻塞起来进入睡眠等待状态。

(3) 执行中进程调用了 P 原语操作，从而因资源不足而被阻塞：或调用了 V 原语操作激活了等待资源的进程队列。

(4) 执行中进程提出 I／O请求后被阻塞。

(5) 在分时系统中时间片已经用完。

(6) 在执行完系统调用等系统程序后返回用户程序时，可看做系统进程执行完毕，从而调度选择一新的用户进程执行。

(7) 在 CPU执行方式是可剥夺时， 还有：就绪队列中的某进程的优先级变得高于当前执行进程的优先级，从而也将引发进 程调度。

6. 假设有 4 道作业，它们的提交时间及执行时间由下表给出：

作业号	提交时刻 (时)	执行时间 (小时)
1	10:00	2
2	10:20	1
3	10:40	0.5
4	10:50	0.3

计算在单道程序环境下， 采用先来先服务调度算法和最短作业优先调度算法时的平均周转时间和平均带权周转时间，并指出它们的调度顺序。

答: (1) 先来先服务调度：

调度顺序： 1 、2、3、4

1 Ts1 ：10:00 Te1：12:00 Tr1 ：2 Tw1: 0

2 Ts2 ：10:20 Te2 ：13:00 Tr2 ：1 Tw2: 1.7

3 Ts3 ：10:40 Te3 ：13:30 Tr3: 0.5 Tw3 ：2.3

4 Ts4 ：10:50 Te4 ：13:50 Tr4: 0.3 Tw4 ：2.7

T=0.25*(2+2.7+2.8+3)=2.625(h)

W=0.25*(4+0+1.7／1+2.3／0.5+2.7 ／0.3)=4.825

(2) 最短作业优先调度 ：

调度顺序： 4、3、2、1

1 Ts4: 10:50 Te4 ：11:10 Tr4 ：0.3 Tw4 ：0

2 Ts3 ：10:40 Te3 ：11:40 Tr3: 0.5 Tw3 ：0.5

3 Ts2 ；10:20 Te2 ：12:40 Tr2: 1 Tw2: 1.3

4 Ts1 ：10:00 Te1: 14:40 Tr1 ：2 Tw1 ：2.7

T ：0.25*(0.3+1+2.3+4.7)= 2.075(h)

W: 0.25*(4+0+1+1.3+2.7 ／2)=1.9125

以上是 4 个作业提交完后的调度情况（提交过程中运行其它作业）。如果边提交边调度，又如何？

第五章

1. 存储管理的主要功能是什么 ？ P109 5.1 节

答：存储管理的主要功能包括以下几点：

(1) 在硬件的支持下完成统一管理内存和外存之间数据和程序段自动交换的虚拟存储。

(2) 将多个虚存的一维线性空间或多维线性空间变换到内存的唯一的一维物理线性地址。

(3) 控制内外存之间的数据传输。

(4) 实现内存的分配和回收。

(5) 实现内存信息的共享与保护。

2. 什么是虚拟存储器 ?其特点是什么 ？ P110

答：由进程中的目标代码、 数据等的虚拟地址组成的虚拟空间称为虚拟存储器。虚拟存储器不考虑物理存储器的大小和信息存放的实际位置， 只规定每个进程中相互关联信息的相对位置。 每个进程都拥有自己的虚拟存储器， 且虚拟存储器的容量是由计算机的地址结构和寻址方式来确定。

实现虚拟存储器要求有相应的地址变换机构， 以便把指令的虚拟地址变换为实际物理地址；另外，由于内存空间较小，进程只有部分内容存放于内存中，待执行时根据需要再调入内存。

3. 实现地址重定位的方法有哪几类 ？ P111

答：实现地址重定位的方法有两种：静态地址重定位和动态地址重定位。

(1) 静态地址重定位是在虚空间程序执行之前由装配程序完成地址映射工作。静态重定位的优点是不需要硬件支持， 但是用静态地址重定位方法进行地址变换无法实现虚拟存储器。 静 态重定位的另一个缺点是必须占用连续的内存空间和难以做到程序和数据的共享。

(2) 动态地址重定位是在程序执行过程中，在 CPU访问内存之前由硬件地址变换机构将要访问的程序或数据地址转换成内存地址。 动态地址重定位的主要优点有：

① 可以对内存进行非连续分配。

② 动态重定位提供了实现虚拟存储器的基础。

③ 动态重定位有利于程序段的共享。

4. 常用的内存信息保护方法有哪几种 ?它们各自的特点是什么 ？P113

答：常用的内存保护方法有硬件法、软件法和软硬件结合保护法三种。

上下界保护法是一种常用的硬件保护法。 上下界存储保护技术要求为每个进程设置对上下界寄存器。 上下界寄存器中装有被保护程序和数据段的起始地址和终止地址。在程序执行过程 中，在对内存进行访问操作时首先进行访问地址合法性检查， 即检查经过重定位之后的内存地址是否在上、 下界寄存器所规定的范围之内。若在规定的范围之内 ，则访问是合法的；否则是非法的，并产生访问越界中断。

保护键法也是一种常用的软件存储保护法。 保护键法为每一个被保护存储块分配一个单独的保护键。 在程序状态字中则设置相应的保护键开关字段， 对不同的进程赋予不同的开关代码 以和被保护的存储块中的保护键匹配。保护键可以设置成对读写同时保护的或只对读写进行单项保护的。 如果开关字段与保护键匹配或存储块未受到保护， 则访问该存储块是允许的， 否则将产生访问出错中断。

另外一种常用的硬软件内存保护方式是： 界限存储器与 CPU的用户态，核心态相结合的保护方式。 在这种保护方式下， 用户态进程只能访问那些在界限寄存器所规定范围内的内存部分，而核心态进程则可以访问整个内存地址空间。

6. 动态分区式管理的常用内存分配算法有哪几种 ?比较它们各自的优缺点。 P118

答 :动态分区式管理的常用内存分配算法有最先适应法 (FF) 、最佳适应法 (BF) 和最坏适应法 (WF)。

优缺点比较：

① 从搜索速度上看最先适应法最佳，最佳适应法和最坏适应法都要求把不同大小的空闲区按大小进行排队。

② 从回收过程来看，最先适应法也是最佳，因为最佳适应法和最坏适应法都必须重新调整空闲区的位置。

③ 最佳适应法找到的空闲区是最佳的，但是会造成内存碎片较多，影响了内存利用率， 而最坏适应法的内存碎片最少， 但是对内存的请求较多的进程有可能分配失败。

总之， 三种算法各有所长， 针对不同的请求队列， 它们的效率和功能是不一样的。

8. 简述什么是覆盖 ?什么是交换 ?覆盖和交换的区别是什么 ? P121

答：将程序划分为若干个功能上相对独立的程序段，按照程序的逻辑结构让那些不会同时执行的程序段共享同一块内存区的内存扩充技术就是覆盖。 交换是指先将内存某部分的程序或数据写入外存交换区， 再从外存交换区中调入指定的程序或数据到内存中来， 并让其执行的一种内存扩充技术。 与覆盖技术相比， 交换不要求程序员给出程序段之间的覆盖结构， 而且，交换主要是在进程或作业之间进行， 而覆盖则主要在同一个作业或同一个进程内进行。 另外， 覆盖只能覆盖那些与覆盖程序段无关的程序段。

9. 什么是页式管理 ?静态页式管理可以实现虚存吗 ? P123

答：页式管理就是把各进程的虚拟空间划分为若干长度相等的页，把指令按页面大小划分后存放在内存中执行或只在内存中存放那些经常被执行或即将被执行的页，而那些不被经常执行以及在近期内不可能被执行的页则存放于外存中，按一定规则调入的一种内存管理方式。

静态页式管理不能实现虚存，这是因为静态页式管理要求进程或作业在执行前全部被装入内存，作业或进程的大小仍受内存可用页面数的限制。

10. 什么是请求页式管理 ？ P127

答：请求页式管理是动态页式内存管理的一种，它在作业或进程开始执行之前，不把作业或进程的程序段和数据段一次性的全部装入内存，而只装入被认为是经常反复执行和调用的工作区部分。其他部分则在执行过程中动态装入。

请求页式管理的调入方式是，当需要执行某条指令而又发现它不在内存时，或当执行某条指令需要访问其他数据或指令时，而这些指令和数据又不在内存中，从而发生缺页中断，系统将外存中相应的页调入内存。

11. 请求页式管理中有哪几种常用的页面置换算法 ?试比较它们的优缺点。 P129

答：比较常用的页面置换算法有：

(1) 随机淘汰算法 (randomglongram) 。即随机地选择某个用户页面并将其换出。

(2) 轮转法 RR(roundrobin) 。轮转法循回换出内存可用区内一个可以被换出的页，无论该页是刚被换进或已经换进内存很长时间。

(3) 先进先出法 FIFO(firstinfirstout) 。FIFO 算法选择在内存驻留时间最长的一页将其淘汰。

(4) 最近最久未使用页面置换算法 LRU(1east recently unused) 。该算法的基本思想是：当需要淘汰某一页时，选择离当前时间最近的一段时间内最久没有使用过的页面先淘汰。

该算法很难实现，比较常用的近似算法：最不经常使用页面淘汰算法 LFU 最近没有使用页面淘汰算法 NUR。

(5) 理想型淘汰算法 OPT(optimalreplacementalgorithm) 。该算法淘汰在访问串中将来再也不出现的或是在离当前最远的位置上出现的页面。

12. 什么是 Belady 现象?找出一个 Belady 现象的例子。 P131

答：使用 FIFO 算法时，在未给进程或作业分配足它所要求的页面数时，有时会出现分配的页面数增多，缺页次数反而增加的奇怪现象。这种现象称为 Belady 现象。

假设进程 P 共有 5 个页，访问顺序是： 1,2,3,4,1,2,5,1,2,3,4,5 的缺页情况。

分配 3 个页面，缺页 9 次：缺页率 9/12=75%

1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	4	4	4	5	5	5	5	5	5
	2	2	2	1	1	1	1	1	3	3	3
		3	3	3	2	2	2	2	2	4	4
✓	✓	✓	✓	✓	✓	✓			✓	✓	

分配 4 个页面，缺页 10 次：缺页率 10/12=83.3%

1	2	3	4	1	2	5	1	2	3	4	5
1	1	1	1	1	1	5	5	5	5	4	4
	2	2	2	2	2	2	1	1	1	1	5
		3	3	3	3	3	3	2	2	2	2
			4	4	4	4	4	4	3	3	3
✓	✓	✓	✓			✓	✓	✓	✓	✓	✓

此例就出现分配的页面数增多，缺页次数反而增加的奇怪现象，即 Belady 现象。

14. 什么是段式管理 ?它与页式管理有何区别 ? P133

答：段式管理就是将程序按照内容或过程 (函数) 关系分成段，每段拥有自己的名字。一个用户作业或进程所包含的段对应于一个二维线性虚拟空间， 也就是一个二维虚拟存储器。 段式管理程序以段为单位分配内存， 然后通过地址映射机构把段式虚拟地址转换成实际的内存物理地址。 同页式管理时一样， 段式管理也采用只把那些经常访问的段驻留内存， 而把那些在将来一段时间内不被访问的段放入外存，待需要时自动调入相关段的方法实现二维虚拟存储器。

段式管理和页式管理的主要区别有：

(1) 页式管理中源程序进行编译链接时是将主程序、 子程序、数据区等按照线性空间的一维地址顺序排列起来。 段式管理则是将程序按照内容或过程 (函数) 关系分成段， 每段拥有自己的名字。 一个用户作业或进程所包含的段对应于一个二维线性虚拟空间，也就是一个二维虚拟存储器。

(2) 同动态页式管理一样，段式管理也提供了内外存统一管理的虚存实现。与页式管理不同的是： 段式虚存每次交换的是一段有意义的信息， 而不是像页式虚存管理那样只交换固定大小的页， 从而需要多次的缺页中断才能把所需信息完整地调入内存。

(3) 在段式管理中， 段长可根据需要动态增长。 这对那些需要不断增加或改变新数据或子程序的段来说，将是非常有好处的。

(4) 段式管理便于对具有完整逻辑功能的信息段进行共享。

(5) 段式管理便于进行动态链接，而页式管理进行动态链接的过程非常复杂。

15. 段式管理可以实现虚存吗 ?如果可以，简述实现方法。 P133

答：段式管理可以实现虚存。

段式管理把程序按照内容或过程 (函数) 关系分成段，每段拥有自己的名字。一个用户作业或进程所包含的段对应于一个二维线性虚拟空间 (段号 s 与段内相对地址 w)，也就是一个二维虚拟存储器。段式管理以段为单位分配内存，然后通过地址映射机构把段式虚拟地址转换成实际的内存物理地址。 只把那些经常访问的段驻留内存， 而把那些在将来一段时间内不被访问的段放入外存， 待需要时产生缺段中断，自动调入。

16. 为什么要提出段页式管理 ?它与段式管理及页式管理有何区别 ？ P138

答：因为段式管理和页式管理各有所长。 段式管理为用户提供了一个二维的虚拟地址空间， 反映了程序的逻辑结构， 有利于段的动态增长以及共享和内存保护等，这极大地方便了用户。 而分页系统则有效地克服了碎片， 提高了存储器的利用效率。从存储管理的目的来讲， 主要是方便用户的程序设计和提高内存的利用率。所以人们提出了将段式管理和页式管理结合起来让其互相取长补短的段页式管理。段页式管理与段式和页式管理相比，其访问时间较长。因此，执行效率低。

17. 为什么说段页式管理时的虚拟地址仍是二维的 ？

答：因为在段页式内存管理中， 对每一段内的地址空间进行分页式管理只是为了克服在内存分配过程中产生的大量碎片， 从而提高存储器的利用效率， 它并没有改变段内地址空间的一维结构， 所以段页式内存管理中的虚拟地址仍然和段式内存管理中的虚拟地址一样，是二维结构的。

18. 段页式管理的主要缺点是什么 ?有什么改进办法 ？

答：段页式管理的主要缺点是对内存中指令或数据进行存取时，至少需要对内存进行三次以上的访问。第一次是由段表地址寄存器取段表始址后访问段表，由此取出对应段的页表在内存中的地址。 第二次则是访问页表得到所要访问的指令或数据的物理地址。 只有在访问了段表和页表之后， 第三次才能访问真正需要访问的物理单元。显然。这将大大降低 CPU执行指令的速度。

改进办法是设置快速联想寄存器。在快速联想寄存器中．存放当前最常用的段号 s，页号 p 和对应的内存页面地址与其他控制项。当需要访问内存空间某一单元时， 可在通过段表、 页表进行内存地址查找的同时， 根据快速联想寄存器查找其段号和页号。 如果所要访问的段或页的地址在快速联想寄存器中， 则系统不再访问内存中的段表、 页表而直接把快速联想寄存器中的值与页内相对地址 d 拼接起来得到内存地址。

19. 什么是局部性原理 ?什么是抖动 ?你有什么办法减少系统的抖动现象 ？ P140

答：局部性原理是指在几乎所有程序的执行过程中，在一段时间内， CPU总是集中地访问程序中的某一个部分而不是对程序的所有部分具有平均的访问概率。

抖动是指当给进程分配的内存小于所要求的工作区时，由于内存外存之间交换频繁， 访问外存的时间和输入输出处理时间大大增加， 反而造成 CPU因等待

数据而空转，使得整个系统性能大大下降。

在物理系统中，为防止抖动的产生，在进行淘汰或替换时，一般总是把缺页进程锁住，不让其换出，从而防止抖动发生。防止抖动发生的另一个办法是设置较大的内存工作区。

第六章

1. 简述 Linux 系统进程的概念 ？

答：P148

2. Linux 进程上下文由哪几部分组成？为什么说核心程序不是进程上下文上午一部分？进程页表也在核心区，它们也不是进程上下文上午一部分吗？

答：进程上下文由 task_struct 结构、用户栈和核心栈的内容、用户地址空间的征文段、数据段、硬件寄存器的内容以及页表等组成。

核心页表被所有进程共享，所以不是进程上下文的一部分。而进程页表是进程上下文的一部分。

4. Linux 的调度策略是什么 ？ 调度时应该封锁中断吗？如果不封锁，会发生什么问题？

答：Linux 使用三种调度策略，动态优先数调度 SCHED_OTHER，先来先服务调度 SCHED_FIFO 和轮转法调度 SCHED_RR。其中动态优先级调度策略用于普通进程，后两种调度策略用于实时进程。

在调度时应封锁中断，否则在调度过程中由于中断会使进程上下文的切换出现错误。

6. Linux 在哪几种情况下发生调度 ？

答：两种情况： 一是进程自动放弃处理机时主动装入调度过程，二是在由核心态转入用户态时，系统设置了高优先级就绪进程的强迫调度标识 need_resched 时发生调度。

8. 什么是软中断 ？

答： P162

13. Linux 存储管理策略中交换和请求调页方式有何区别 ？

答： P171

第八章

1. 什么是文件、文件系统？文件系统有哪些功能？ P198

答：在计算机系统中，文件被解释为一组赋名的相关字符流的集合，或者是相关纪录的集合。

文件系统是操作系统中与管理文件有关的软件和数据。

文件系统的功能是为用户建立文件、撤销、读写、修改和复制文件，以及完成对文件的按名存取和进行存取控制。

2、文件一般按什么分类？可以分为哪几类？ P199

答：文件一般按性质、用途、组织形式、文件中的信息流向或文件的保护级别等分类。

按性质和用途可分为系统文件、库文件和用户文件。

按文件的组织形式可分为普通文件、目录文件和特殊文件。
按文件中的信息流向可分为输入文件、输出文件和输入 / 输出文件。
按文件的保护级别可分为只读文件、 读写文件、 可执行文件和不保护文件。

3、什么是文件的逻辑结构？什么是纪录？ P200

答：文件的逻辑结构就是用户可见的结构，可分为字符流式的无结构文件和记录式的有结构文件两大类。

记录是一个具有特定意义的信息单位， 他由该纪录在文件中的逻辑地址 （相对位置）与记录名所对应的一组关键字、属性及其属性值所组成。

7、文件的物理结构有哪几种？为什么说串联文件结构不适于随机存取？ P205

答：文件的物理结构是指文件在存储设备上的存取方法。常用的文件的物理结构有连续文件、串联文件和索引文件三种。

串联文件结构用非连续的物理块来存取文件信息。这些非连续的物理块之间没有顺序关系， 链接成一个串联队列。 搜索时只能按队列中的串联指针顺序搜索，存取方法应该是顺序存取的。 否则，为了读取某个信息块而造成的磁头大幅度移动将花去较多的时间。因此，串联文件结构不适于随机存取。

9、常用的文件存储设备的管理方法有哪些？试述主要优缺点。 P209

答：文件存储设备的管理实质上是一个空闲块的组织和管理问题。有 3 种不同的空闲块管理方法，即空闲文件目录、空闲块链和位示图。

空闲文件目录管理方法就是把文件存储设备中的空闲块的块号统一放在一个称为空闲文件目录的物理块中， 其中空闲文件目录的每个表项对应一个由多个空闲块构成的空闲区。 该方法实现简单， 适于连续文件结构的文件存储区的分配与回收。但是由于回收时不进行合并， 所以使用该方法容易产生大量的小块空闲区。

空闲块链法把文件存储设备上的所有空闲块链接在一起，从链头分配空闲块，把回收的空闲块插入到链尾。该方法不占用额外的空间，但实现复杂。

位示图法是从内存中划出若干字节，每个比特位对应一个物理块的使用情况。如果该位为 0 表示对应的块是空闲的，为 1 表示对应的物理块已分配出去。位示图法在查找空闲块时无需启动外设，但要占用内存空间。

11、什么是文件目录？文件目录中包含哪些信息？ P211

答：一个文件的文件名和对该文件实施控制管理的说明信息成为该文件的说明信息，又称为该文件的文件控制块 (FCB)。把所有的 FCB组织在一起，就构成了文件目录，即文件控制块的有序集合。

文件目录中包含文件名、与文件名相对应的文件内部标识以及文件信息在文件存储设备上的第一个物理块的地址等信息。 另外还可能包含关于文件的逻辑结构、物理结构、存取控制信息和管理等信息。

13、文件存取控制方式有哪几种？试比较它们各自的优缺点。 P216

答：文件存取控制方式一般有存取控制矩阵、存取控制表、口令和密码术 4 种方式。

存取控制矩阵方式以一个二维矩阵来进行存取控制。而且矩阵的一维是所有的用户。另一维是所有的文件。 对应的矩阵元素则是用户对文件的存取控制权。存取控制矩阵的方法在概念上比较简单， 但是当用户和文件较多时， 存取控制矩阵将变得非常庞大，从而时间和空间的开销都很大。

存取控制表以文件为单位，把用户按某种关系划分为若干组，同时规定每组的存取限制。这样所有用户组对文件权限的集合就形成了该文件的存取控制表。存取控制表方法占用空间较小，搜索效率也较高，但要对用户分组，引入了额外的开销。

口令方式有两种。一种是当用户进入系统时，为建立终端进程时获得系统使用权的口令。 另一种方式是， 每个用户在创建文件时， 为每个创建的文件设置一个口令， 且将其置于文件说明中。 当任一用户想使用该文件时， 都必须首先提供口令。口令方式比较简单，占用的内存单元以及验证口令所费时间都非常少。不过，相对来说，口令方式保密性能比较差。

密码术方式在用户创建源文件并写入存储设备时对文件进行编码加密，在读出文件时对文件进行译码解密。 加密方式具有保密性强的优点。 但是， 由于加密解密工作要耗费大量的处理时间， 因此, 加密技术是以牺牲系统开销为代价的。

第九章

1. 设备管理的目标和功能是什么 ？

答：设备管理的目标是：

(1) 选择和分配输入／输出设备以便进行数据传输操作；

(2) 控制输入／输出设备和 CPU(或内存) 之间交换数据 ；

(3) 为用户提供一个友好的透明接口 ；

(4) 提高设备和设备之间、 CPU和设备之间，以及进程和进程之间的并行操作，以使操作系统获得最佳效率。

设备管理的功能是：

(1) 提供和进程管理系统的接口；

(2) 进行设备分配；

(3) 实现设备和设备、设备和 CPU等之间的并行操作；

(4) 进行缓冲区管理。

2. 数据传送控制方式有哪几种 ?试比较它们各自的优缺点。

答：数据传送控制方式有程序直接控制方式、中断控制方式、 DMA方式和通道方式 4 种。

程序直接控制方式就是由用户进程来直接控制内存或 CPU和外围设备之间的数据传送。它的优点是控制简单，也不需要多少硬件支持。它的缺点是 CPU和外围设备只能串行工作 ;设备之间只能串行工作 ,无法发现和处理由于设备或其他硬件所产生的错误。

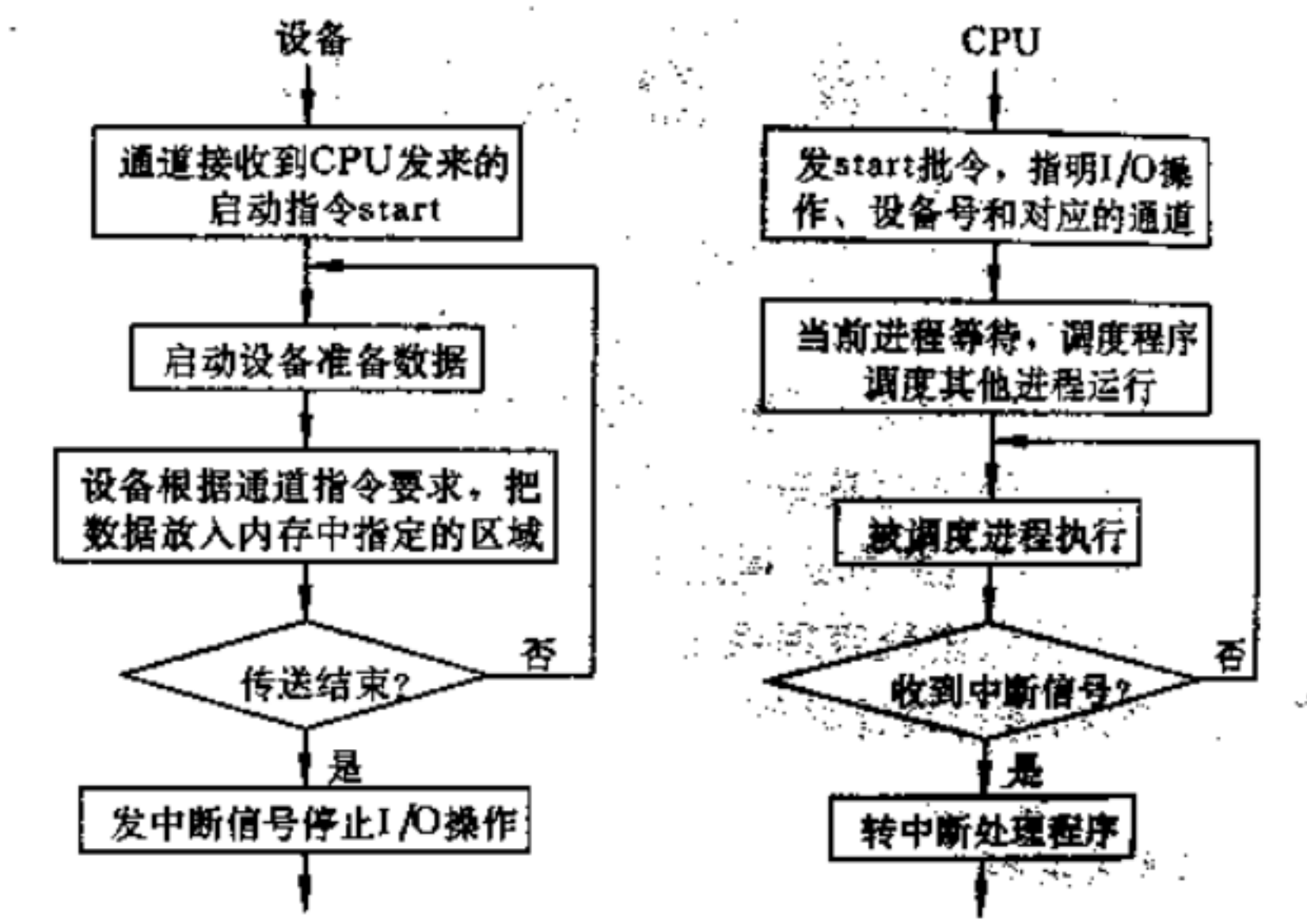
中断控制方式是利用向 CPU发送中断的方式控制外围设备和 CPU之间的数据传送。它的优点是大大提高了 CPU的利用率且能支持多道程序和设备的并行操作。它的缺点是由于数据缓冲寄存器比较小， 如果中断次数较多， 仍然占用了大量 CPU时间；在外围设备较多时， 由于中断次数的急剧增加， 可能造成 CPU无法响应中断而出现中断丢失的现象；如果外围设备速度比较快，可能会出现 CPU来不及从数据缓冲寄存器中取走数据而丢失数据的情况。

DMA方式是在外围设备和内存之间开辟直接的数据交换通路进行数据传送。它的优点是除了在数据块传送开始时需要 CPU的启动指令，在整个数据块传送结束时需要发中断通知 CPU进行中断处理之外， 不需要 CPU的频繁干涉。它的缺点是在外围设备越来越 多的情况下， 多个 DMA控制器的同时使用， 会引起内存地址的冲突并使得控制过程进一步复杂化。

通道方式是使用通道来控制内存或 CPU和外围设备之间的数据传送。 通道是一个独立与 CPU的专管输入／输出控制的机构， 它控制设备与内存直接进行数据交换。它有自己的通道指令， 这些指令受 CPU启动，并在操作结束时向 CPU发中断信号。该方式的优点是进一步减轻了 CPU的工作负担， 增加了计算机系统的并行工作程度。缺点是增加了额外的硬件，造价昂贵。

3. 什么是通道 ?试画出通道控制方式时的 CPU 通道和设备的工作流程图。

答：通道是一个独立与 CPU的专管输入／输出控制的机构， 它控制设备与内存直接进行数据交换。 它有自己的通道指令， 这些指令受 CPU启动，并在操作结束时向 CPU发中断信号。



4. 什么是中断 ?什么叫中断处理 ?什么叫中断响应 ？

答： 中断是指计算机在执行期间， 系统内发生任何非寻常的或非预期的急需处理事件，使得 CPU暂时中断当前正在执行的程序而转去执行相应的事件处理程序，待处理完毕后又返回原来被中断处继续执行的过程。

CPU转去执行相应的事件处理程序的过程称为中断处理。

CPU收到中断请求后转到相应的事件处理程序称为中断响应。

5. 什么叫关中断 ?什么叫开中断 ?什么叫中断屏蔽 ？

答：把 CPU内部的处理机状态字 PSW的中断允许位清除从而不允许 CPU响应中断叫做关中断。

设置 CPU内部的处理机状态字 PSW的中断允许位从而允许 CPU响应中断叫做开中断。

中断屏蔽是指在中断请求产生之后， 系统用软件方式有选择地封锁部分中断而允许其余部分的中断仍能得到响应。

6. 什么是陷阱 ?什么是软中断 ?试述中断、陷阱和软中断之间异同。

答 :陷阱指处理机和内存内部产生的中断， 它包括程序运算引起的各种错误，如地址非法、校验错、页面失效。存取访问控制错、从用户态到核心态的切换等都是陷阱的例子。软中断是通信进程之间用来模拟硬中断的一种信号通信方式。

8. 什么是缓冲 ?为什么要引入缓冲 ？

答 :缓冲即是使用专用硬件缓冲器或在内存中划出一个区域用来暂时存放输入输出数据的器件。

引入缓冲是为了匹配外设和 CPU之间的处理速度， 减少中断次数和 CPU的中断处理时间，同时解决 DMA或通道方式时的数据传输瓶颈问题。

13. 什么是 I/O 控制?它的主要任务是什么 ？

答：I/O 控制是指从用户进程的输入输出请求开始，给用户进程分配设备和启动有关设备进行 I/O 操作，并在 I/O 操作完成之后响应中断， 直至善后处理为止的整个系统控制过程。

15. 设备驱动程序是什么？为什么要有设备驱动程序？用户进程怎样使用驱动程序？

答 :设备驱动程序是驱动外部物理设备和相应 DMA控制器或 I/O 控制器等器件，使之可以直接和内存进行 I/O 操作的子程序的集合。 它们负责设置相应设备有关寄存器的值，启动设备进行 I/O 操作，指定操作的类型和数据流向等。

设备驱动程序屏蔽了直接对硬件操作的细节， 为编程者提供操纵设备的友好接口。

用户进程通过调用设备驱动程序提供的接口来使用设备驱动程序。