试说明PCB的作用具体表现在那几个方面？为什么说PCB是进程存在的唯一标识

2.3 进程调度

进程调度

调度的层次

调度算法的评价指标有哪些？

处理机调度算法的共同目标是什么？批处理系统的调度目标又是什么？

各个调度算法的概念/特点/缺点

为什么要引入高响应比优先调度算法？他有何优点？

在抢占调度方式中，抢占的原则是什么？

时间片大小的确定

多级反馈队列调度算法的优点

选择调度方式和算法的准则，并举例

2.3 进程同步

制约关系分为哪几种？如何理解？

同步机制应遵循的规则

实现临界区互斥的方法有哪些？

什么是前驱图？为什么要引入前驱图

原语

管程的组成

2.4 死锁

死锁概念

死锁产生的原因

死锁的必要条件

死锁的处理策略

系统安全状态

在解决死锁问题的几个方法中，那种方法最易于实现？哪种方法使得资源利用率最高？

内存管理

基本概念

内存管理的任务和功能

程序转变为可执行程序的步骤

程序的链接

程序的装入

什么是静态链接？静态链接需要解决两个什么问题？

什么是装入时动态链接？装入时动态链接方式有何优点？

什么是运行时动态链接？有何优点？

连续内存分配

试说明I/0控制发展的主要推动因素是什么？

有哪几种I/O控制方式？各适用于何种场合？

试说明DMA的工作流程

什么是设备控制器？主要功能是什么？

设备控制器的组成

设备无关性的基本含义是什么？为什么要设置该层？

设备独立性的优点

使用逻辑设备名的好处

设备分配的原则

设备分配方式

设备映射表

Spooling技术的组成（假脱机技术

SPOOLing技术的特点

何谓设备虚拟？实现设备虚拟式所依赖的关键技术是什么？

假脱机系统向用户提供共享打印机的基本思想是什么？

引入缓冲的主要原因是什么？

1 计算机系统概述

操作系统的定义

操作系统是指控制和管理整个计算机系统的硬件与软件资源，合理地组织，调度计算机的工作与资源的分配，进而为用户和其他软件提供方便接口与环境的程序的集合。

操作系统的四大特性

操作系统基本特性包括：并发，共享，虚拟，异步

1. 并发：两个或多个事件在同一时间间隔内发生（计算机系统中同时存在多个运行的程序）
2. 共享：系统中的资源可供内存中多个并发执行的进程共同使用

3. 虚拟：把一个物理上的实体变为若干个逻辑上的对应物
4. 异步：多道程序环境下允许多个程序并发执行，但由于资源有限，进程的执行并不是一贯到底的，而是走走停停，以不可预知的速度向前推进。

把一段时间内只允许一个进程访问的资源称为临界资源

互斥共享：一段时间内只允许一个进程访问该资源

同时访问：若干个用户同时访问该文件

操作系统利用了多种虚拟技术来实现虚拟处理器，虚拟内存，虚拟外部设备等

操作系统的虚拟技术可归纳为：时分复用技术，如处理器的分时共享；空分复用技术，如虚拟存储器

操作系统的目标和功能

处理机管理，存储器管理，设备管理，文件管理。

操作系统作为计算机系统资源的管理者

处理机管理（又叫进程管理）

任务：进程何时创建，何时撤销，如何管理，如何避免冲突，合理共享。

功能：主要包括进程控制，进程调度，进程通信，进程同步，死锁处理。

存储器管理 存 射 保 扩

任务：给多道程序的运行提供良好的环境，方便用户使用及提高内存的利用率

功能：内存分配与回收 地址映射 内存保护与共享 内存扩充

文件管理

任务：对用户文件和系统文件进行组织管理，以方便用户使用，并保证文件的安全性

功能：文件存储空间的管理，目录管理，文件读写管理和保护

设备管理

任务：完成用户的IO请求，方便用户使用各种设备，并提高设备的利用率

功能：缓冲管理，设备分配，设备处理，虚拟设备

多道程序设计的概念和特点

多道程序设计技术允许多个程序同时进入内存并允许它们在CPU中交替运行，这些程序共享系统中的各种软硬件资源。

1. 多道。计算机内存中同时存放多道相互独立的程序
2. 宏观上并行。同时进入系统的多道程序都处于运行过程中。
3. 微观上串行。内存中的多道程序轮流占有CPU，交替执行

多道程序设计的优点和缺点

优点：资源利用率高，多道程序共享计算机资源，从而使各种资源得到充分利用；系统吞吐量大，CPU 和其他资源保持“忙碌”状态。缺点：用户响应的时间较长；不提供人机交互能力，用户既不能了解自己的程序的运行情况，又不能控制计算机。

多道程序设计技术的实现需要解决哪些问题

1）如何分配处理器。
2）多道程序的内存分配问题。
3）I/O 设备如何分配。
4）如何组织和存放大量的程序和数据，以方便用户使用并保证其安全性与一致性。

整个操作系统发展的动力

1. 不断提高计算机资源的利用率
2. 器件不断更新迭代
3. 方便用户
4. 计算机体系结构的不断发展
5. 不断提出新的应用要求。

OS的作用可表现在哪几个方面

作为用户与计算机硬件系统之间的接口：OS处于用户与计算机硬件系统之间。用户通过OS来使用计算机系统。

作为计算机系统资源的管理者：资源：处理机、存储器、I/O设备以及文件（数据和程序）。

实现了对计算机资源的抽象

是什么原因使操作系统具有异步性特征？

对于内存中的每个进程，在何时能获得处理机运行，何时又因提出某种资源请求而暂停，以及进程以怎样的速度向前推进，每道程序总共需要多少时间才能完成等，都是不可预知的，进程是以人们不可预知的速度向前推进的，此即进程的异步性。

为什么要引入实时操作系统

答：实时操作系统是指系统能及时响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时任务协调一致地运行。引入实时 OS 是为了满足应用的需求，更好地满足实时控制领域和实时信息处理领域的需要。

原语具有哪些特点？

1. 处于操作系统最底层，接近硬件
2. 具有原子性，操作一气呵成，不可以被中断
3. 运行时间较短，调用频繁。

什么是中断向量表

为了处理上的方便，通常是为每种设备配以相应的中断处理程序，并为每个设备的中断请求规定一个中断号。中断向量表把中断处理程序的入口地址和对应的中断号记录在一个表项中。

对多源中断的处理方式

1. 屏蔽中断。当处理机正在处理一个中断时，讲屏蔽掉所有的中断。直到CPU完成本次中断的处理后，再去检查是否有中断发生。
2. 嵌套中断。当同时有多个不同优先级的中断请求时，CPU优先响应最高优先级的中断请求。高优先级的中断请求可以抢占正在运行的低优先级中断的处理机。

为什么要引入中断机制？中断处理过程有几步？

为了提高多道程序运行环境中CPU和各种设备的利用率，提高系统的吞吐量。

五步：

1. 测定是否有未响应的中断信号
2. 保护被中断进程的CPU环境
3. 转入相应的设备处理程序
4. 中断处理
5. 恢复CPU的现场并退出中断

2 进程管理

2.1 进程与线程

进程

程序段，数据段，PCB构成了进程实体，进程是进程实体的运行过程，是操作系统进行处理机调度和资源分配的基本单位

进程的特征

1. 动态性。进程的实质是进程实体的执行过程
2. 并发性。多个进程同时存在于内存中，且能在一段时间内同时运行

3. 独立性。进程是一个能独立运行，独立获得资源，独立接受调度的基本单位。
4. 异步性。进程是按异步方式运行的，即按各自独立的，不可预知的速度向前推进

进程的创建步骤

1. 首先为新进程分配一个唯一得进程标识号，然后申请一个空白PCB。若PCB申请失败，则进程创建失败。
2. 然后为该进程分配运行时所必须的资源，如内存，文件，IO设备；如果所需的资源不能立即满足，则仍处于创建态。
3. 初始化PCB，主要包括初始化标志信息，初始化处理机状态信息等。
4. 最后把该进程转入就绪态并插入就绪队列。

进程的状态与转换

1. 运行态。进程在处理机上运行
2. 就绪态。进程获得了除了CPU外的一切所需资源。一旦获得CPU即可运行
3. 阻塞态。进程正在等待某一事件而暂停运行。
4. 创建态。进程正在被创建，尚未转到就绪态。
5. 结束态。进程正在从系统中消失，可能是进程正常结束或其他原因退出运行。

转换图：

就绪 to 运行 通过进程调度和进程切换上处理机

运行 to 就绪 时间片到/被更高优先级进程抢占

运行 to 阻塞 进程请求某一资源/等待某一事件发生

阻塞 to 就绪 进程等待事件到来/进程请求的资源已经分配好

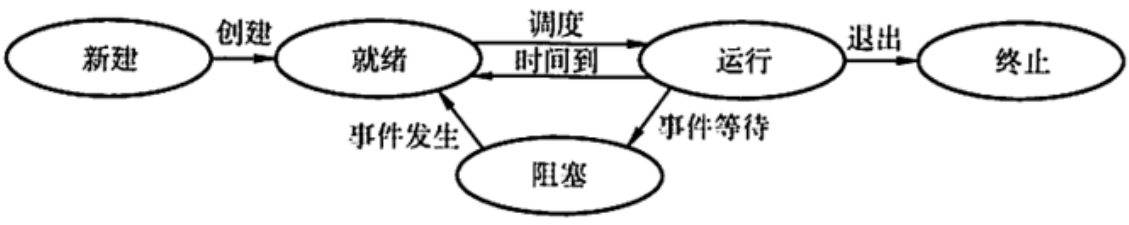


图 2.1　5 种进程状态的转换

PCB的组织方式

1. 线性方式 即将系统中所有的PCB都组织在一张线性表中
2. 链接方式 即把具有相同状态进程的PCB分别通过PCB中的链接字链接成一个队列
3. 索引方式 即系统根据所有进程状态的不同，建立几张索引表

进程的通信

高级通信方法

1. 共享存储 通信的进程之间有一块可以直接访问的共享空间，进程可以直接读写实现进程之间的信息交换
2. 消息传递 进程之间传递格式化的消息
3. 管道通信 通过缓冲区来传递消息

为什么要引入线程

线程是轻量级的进程，为了减小程序在并发执行时的时空开销，使OS有更好的并发性，提高CPU的利用率 ，于是在操作系统中引入线程

进程和线程的比较

1. 调度 线程是调度的基本单位，线程是资源分配的基本单位
2. 拥有资源 进程是资源分配的基本单位，线程不拥有系统资源，但可以使用其隶属进程的系统资源
3. 并发性 不仅进程之间可以并发执行，多个线程之间也可以并发执行
4. 系统开销 系统创建进程的开销远大于创建或撤销线程的开销
5. 独立性 每个进程都拥有独立的地址空间和资源，而线程共享它们所属进程的地址空间和资源。

用户级线程和内核级线程的区别

1. 内核支持线程是OS内核可感知的，而用户级线程是OS内核不可感知的。

2. 用户级线程的创建、撤消和调度不需要OS内核的支持，而内核级线程需要
3. 用户级线程运行在用户态，而内核级线程可以运行在任何状态下。
4. 在只有用户级线程的系统内，CPU调度还是以进程为单位。在有内核级线程的系统内，CPU调度则以线程为单位

为什么程序并发执行会产生间断性特征？

程序并发执行时，他们共享系统资源，由于资源是有限的，使这些并发执行的进程之间，形成了相互制约关系，从而使得进程在执行期间出现间断性。

程序并发执行时为什么会失去封闭性和可再现性

程序并发执行时，多个程序共享系统中的各种资源，因而这些资源的状态由多个程序改变，并且进程的执行具有异步性，最终使程序运行失去了封闭性，也会导致其失去可再现性。

在操作系统中为什么要引入进程的概念？它会产生什么样的影响。

为了使程序在多道程序环境下能并发执行，并对并发执行的程序加以控制和描述，在操作系统中引入了进程的概念。

影响：使程序的并发执行得以实行，提高了资源的利用率。

试说明PCB的作用具体表现在那几个方面？为什么说PCB是进程存在的唯一标识

PCB是进程实体的一部分，记录了进程的状态信息。有了PCB之后，程序成为一个能独立运行的基本单位，成为能与其他进程并发执行的进程。PCB和进程一一对应，OS是根据PCB对并发执行的进程进行控制和管理的，所以说PCB是进程存在的唯一标识

2.3 进程调度

进程调度

在单处理器多道程序环境中，内存中同时存在多个进程，但只有一个处理机，每次一个进程执行完后，需要操作系统通过某种调度算法来从就绪队列中选择一个进程上处理机运行。

调度的层次

1. 作业调度。按照一定的原则从外存上处于后备队列的作业中挑选-一个(或多个)，给它(们)分配内存、输入/输出设备等必要的资源，并建立相应的进程
2. 内存调度。把外存上的那些已具备运行条件的挂起态进程再重新调入内存，并修改其状态为就绪态，挂在就绪队列上等待。
3. 进程调度。按照某种算法从就绪队列中选取一个进程，将处理机分配给它。

调度算法的评价指标有哪些？

CPU利用率

系统吞吐量 单位时间内CPU完成作业的数量

周转时间 从作业提交到作业完成所经历的时间

等待时间 等待CPU的时间之和

响应时间 从用户提交请求到系统首次产生响应所用的时间

处理机调度算法的共同目标是什么？批处理系统的调度目标又是什么？

处理机调度算法的共同目标：资源利用率、公平性、平衡性、策略强制执行。批处理系统的调度目标：平均周转时间短、系统吞吐量高、处理机利用率高

各个调度算法的概念/特点/缺点

略。见思维导图

为什么要引入高响应比优先调度算法？他有何优点？

在批处理系统中，先来先服务算法（FCFS）所考虑的只是作业的等待时间，而忽视了作业运行时间。而短作业优先算法（SJF）正好与之相反，只考虑作业运行时间，而忽视了作业等待时间。高响应比优先调度算法则是既考虑了作业等待时间，又考虑作业运行时间的调度算法，因此既照顾了短作业，又不致使长作业的等待时间过长，从而改善了处理机调度的性能。

在抢占调度方式中，抢占的原则是什么？

优先权原则、短进程优先原则、时间片原则

时间片大小的确定

若选择很小的时间片，将有利于短作业。但时间片小，意味着会频繁地执行进程调度和
进程上下文的切换，这无疑会增加系统的开销。反之，若时间片选择得太长，且为使每个
进程都能在一个时间片内完成，RR算法便退化为FCFS算法，无法满足短作业和交互式用
户的需求。一个较为可取的时间片大小是略大于一次典型的交互所需要的时间，使大多数
交互式进程能在-一个时间片内完成，从而可以获得很小的响应时间 。

多级反馈队列调度算法的优点

1. 提高系统吞吐量且缩短平均周转时间，照顾短进程
2. 不必是事先估计进程的执行时间
3. 对于短进程，响应快，在前几个队列中就能完成，周转时间短
4. 对于长进程，依次在多级队列中运行，用户不必担心长期得不到处理

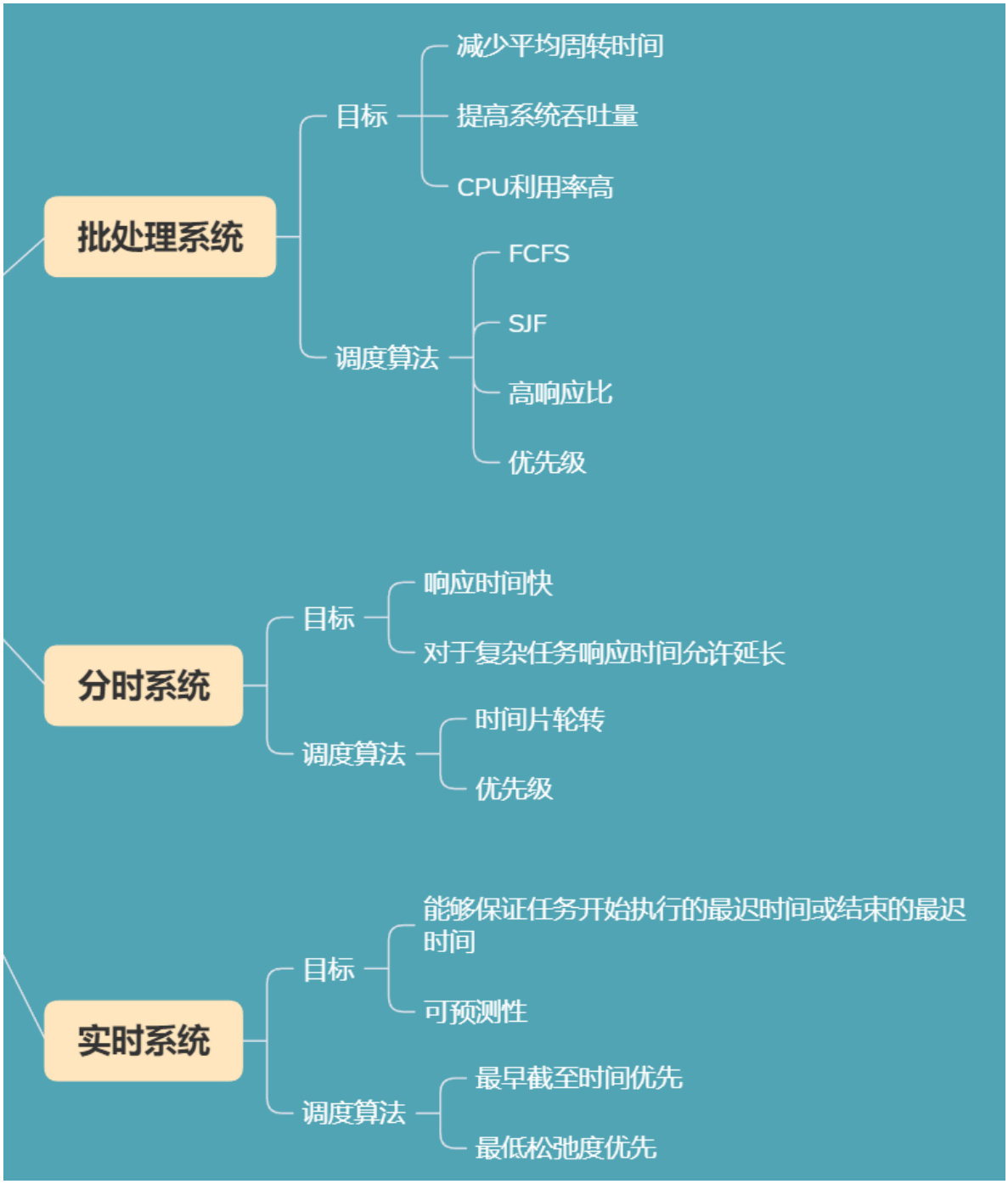
选择调度方式和算法的准则，并举例

对于如何选择调度方式和算法，在很大程度上取决于操作系统的类型及其设计目标

一些基本的原则：

1. 要提高系统资源的利用率
2. 使诸进程都获得合理的CPU时间，不会发生进程饥饿现象
3. 调度算法应尽可能保持系统资源使用的平衡性

举例：



2.3 进程同步

制约关系分为哪几种？如何理解？

分为两种：间接制约关系和直接制约关系

1. 间接制约关系：多个程序在并发执行时，由于共享系统资源，如CPU、I/O设备等，致使在这些并发执行的程序之间形成间接制约的关系。
2. 直接制约关系：某些应用程序，为了完成某任务而建立了两个或多个进程。这些进程将为完成同一项任务而相互合作。进程间的直接制约关系就是源于它们之间的相互合作。

同步机制应遵循的规则

1. 空闲让进。当无进程处于临界区时，表明临界资源处于空闲状态，应允许一个请
   求进入临界区的进程立即进入自己的临界区，以有效地利用临界资源。
2. 忙则等待。当已有进程进入临界区时，表明临界资源正在被访问，因而其它试图
   进入临界区的进程必须等待，以保证对临界资源的互斥访问。
3. 有限等待。对要求访问临界资源的进程，应保证在有限时间内能进入自己的临界
   区，以免陷入“死等”状态。
4. 让权等待。当进程不能进入自己的临界区时，应立即释放处理机，以免进程陷入
   “忙等”状态。

实现临界区互斥的方法有哪些？

软件方法：

1. 单标志法
2. 双标志先检查法
3. 双标志后检查法
4. Peterson算法

硬件方法：

1. 中断屏蔽方法
2. TestAndSet指令
3. Swap指令

什么是前驱图？为什么要引入前驱图

为了能更好地描述程序的顺序和并发执行情况，我们引入前趋图。所谓前趋图是指一个“有向无环图。图中的每个结点可用来表示一个进程或程序段，结点间的有向边描述结点之间执行的先后顺序

原语

原语是指完成某种功能且不可被分割中断执行的操作序列，通常由关中断，开中断来实现

管程的组成

1. 管程的名称
2. 局部于管程内部的共享数据结构说明
3. 对该数据结构进行操作的一组过程
4. 对局部于管程内部的共享数据设置初始值的语句

2.4 死锁

死锁概念

死锁的概念：多个进程因竞争资源而造成的一种互相等待的僵局，若无外力作用，这些进程都将无法向前推进。

死锁产生的原因

1. 进程对不可剥夺的系统资源的竞争

2. 进程推进顺序非法
3. 信号量使用不当

死锁的必要条件

1. 互斥条件 进程互斥使用资源，即在一段时间内，某资源只能被一个进程占用。
2. 请求和保持条件。进程已经保持了至少一个资源，但又提出了新的资源请求
3. 不可抢占条件。进程已获得的资源在未使用完之前不能被抢占，只能在进程使用
   完时由自己释放。
4. 循环等待条件。在发生死锁时，必然存在一个进程一资源的循环链。

死锁的处理策略

死锁预防 破坏产生死锁的4个必要条件中的一个或多个

1. 破坏互斥条件
2. 破坏不可剥夺条件 若一个已经保持某些不可剥夺资源的进程请求新的资源得不到满足时，它必须释放已持有的资源
3. 破坏请求和保持条件 采用静态分配法，一次性把进程所需要的所有资源分配给它，如果无法满足，则该进程不能运行
4. 破坏循环等待条件 采用顺序资源分配法，给系统的资源编号，规定每个进程必须按编号递增的顺序请求资源，同类资源一次申请完。

死锁避免 使用银行家算法防止系统进入不安全状态

死锁检测和解除。不采取任何措施，在发生死锁后，系统能够及时检测出死锁，然后采取某种措施解除死锁。

1. 资源剥夺法
2. 撤销进程法
3. 进程回退法

系统安全状态

所谓安全状态，是指系统能按某种进程推进顺序（P1,P2,....,Pn）为每个进程Pi分配其所需的资源，直到满足每个进程对资源的最大需求，使每个进程都可顺序完成。

在解决死锁问题的几个方法中，那种方法最易于实现？哪种方法使得资源利用率最高？

解决死锁的方法有预防死锁，避免死锁，检测和解除死锁，其中预防死锁方法最容易实现，但由于所施加的限制条件过于严格,会导致系统资源利用率和系统吞吐量降低；而检测和解除死锁方法使得系统有较好的资源利用率和吞吐量

内存管理

基本概念

内存管理的任务和功能

给多道程序提供良好的运行环境，方便用户使用内存，提高内存的利用率。

功能：

1. 内存的分配和回收
2. 地址映射
3. 内存扩充
4. 内存的保护和共享

程序转变为可执行程序的步骤

编译：由编译程序把源代码编译为多个目标模块

链接：由链接程序把编译后形成的一组模块以及所需的库函数链接在一起，形成一个完整的装入模块。

装入：由装入程序把装入模块装入内存运行。

程序的链接

静态链接 在程序运行之前，先将各目标模块及它们所需要的库函数链接成一个完整的可执行程序，以后不再拆开

装入时动态链接 将用户源程序编译后得到一组目标模块，在装入内存时，采取编装入边链接的方式

运行时动态链接 对于某些目标模块的链接，是在程序执行中需要该目标模块时才进行的

程序的装入

1. 绝对装入方式。在单道程序环境下，用户程序编译后，产生绝对地址的目标代码，此时可以采用绝对装入方式
2. 可重定位装入。在多道程序环境下，多个目标模块的起始地址通常都从0开始，程序中的其他地址都是相对于起始地址的，在装入时对目标程序中指令和数据地址进行修改。
3. 动态运行时装入。装入程序把装入模块装入内存后，并不立即把装入模块中的相对地址转换为绝对地址，而是把这种地址转换推迟到程序真正要执行时才进行。

什么是静态链接？静态链接需要解决两个什么问题？

在程序运行之前，先将各目标模块及它们所需的库函数连接成一个完整的装配模块，以后不再拆开。这种事先进行连接的方式称为静态链接方式。

要解决两个问题：对相对地址进行修改；变换外部调用符号

什么是装入时动态链接？装入时动态链接方式有何优点？

将用户源程序编译后所得到的一组目标模块，在装入内存时，采用边装入边链接的链接方式。

优点：便于修改和更新；便于实现对目标模块的共享

什么是运行时动态链接？有何优点？

将某些模块的链接推迟到程序执行时才进行。

优点：加快程序的装入过程；节省大量的内存空间。

连续内存分配

连续内存分配分为哪几类？

1. 单一连续分配
2. 固定分区分配
3. 动态分区分配
4. 可重定位分配

动态分区分配有哪些算法？

基于顺序搜索的动态分区分配算法：

1. 首次适应算法
2. 循环首次适应算法
3. 最佳适应算法
4. 最坏适应算法

基于索引搜索的动态分区分配算法

1. 快速适应算法
2. 伙伴系统
3. 哈希算法

非连续内存分配

什么是页表，页表的作用是什么？

在分页系统中，允许将进程的各个页离散地存储在内存的任一物理块中，为保证进程仍然能够正确地运行，即能在内存中找到每个页面所对应的物理块，系统又为每个进程建立了一张页面映射表，简称页表。页表的作用是实现从页号到物理块号的地址映射。

在分页系统中是如何实现地址变换的？

利用地址变换机构实现从逻辑地址到物理地址的转变换，通过页表来实现从页号到物理块号的变换，将逻辑地址中的页号转换为内存中的物理块号，然后加上逻辑地址的页内偏移即可得到物理地址。

具有快表时是如何实现地址变换的？

在CPU给出有效地址后，先去高速缓冲寄存器中的快表查询，将此页号与快表中的所有页号进行比较，若其中有与此所对应的物理块号，便将物理块号加上页内偏移得到物理地址。如在快表中未找到对应的页表项，则还需再访问内存中的页表，找到后，通过拼接得到物理地址；同时，再将此页表项存入快表的一个寄存器单元中，即：更新快表。如果此时快表已满，则通过某种算法淘汰一项，再存入。

多级页表解决了什么问题?又会带来什么问题?

多级页表解决了当逻辑地址空间过大时,页表的长度会大大增加的问题。而采用多级页表时,一次访 盘需要多次访问内存甚至磁盘，会大大增加一次访存 的时间。

较详细地说明引入分段存储管理是为了满足用户哪几方面的需要？

1. 方便编程：用户通常把自己的作业按照逻辑关系划分为若干段
2. 信息共享：段是信息的逻辑单位，方便共享信息。
3. 信息保护：对信息的逻辑单位进行保护，分段能更有效方便地实现信息保护功能
4. 动态增长
5. 动态链接

为什么说分段系统比分页系统更易于实现信息的共享和保护？

分页系统的每个页面是分散存储的，它们没有逻辑上的联系，因此如果为了实现信息共享和保护，需要建立大量的页表项，记录每一页的共享和保护信息，并且可能会出现一个页内代码出现不同保护等级的现象；而分段系统中每个段都是程序员手动设置的，是相互联系的代码的集合，在实现共享和保护时，只需为要共享和保护的程序设置一个段表项，将其中的基址与内存地址一一对应就能够实现。

分页和分段存储管理有何区别

1. 页是信息的物理单位，分页是为了实现离散分配方式，以消减内存的外部零头，提高内存利用率。段则是信息的逻辑单位，它含有一组相对完整的信息。
2. 页的大小固定且由系统决定，由系统把逻辑地址划分为页号和页内地址两部分，是由机械硬件实现的,因而在系统中只能有一种大小的的页面;而段的长度却不固定,决定于用户所编写的程序,通常由编译程序在对原程序进行编译时,根据信息的性质来划分。
3. 分页的作业地址空间是一维的,而分段作业地址空间则是二维的。

试全面比较连续分配和离散分配方式。

1. 连续分配是指为一个用户程序分配一个连续的地址空间，包括单一连续分配，固定分区分配，动态分区分配，可重定位分配。
2. 离散分配方式分为分页、分段和段页式存储管理。分页式存储管理旨在提高内存利用率，分段式存储管理旨在满足用户(程序员)的需要，段页式存储管理则将两者结合起来，具有分段系统便于实现、可共享、易于保护和动态链接等优点，又能像分页系统很好解决外部碎片及为各段可离散分配内存等问题，是比较有效的存储管理方式；

虚拟内存分配

程序的局部性原理

1. 时间局部性。程序中的某条指令一旦执行，不久后该指令可以再次执行。某数据被访问过，不久后数据可能再次被访问。产生原因是程序中存在大量的循环操作
2. 空间局部性。一旦程序访问了某个存储单元，在不久后，其附近的存储单元也将被访问。因为指令通常是顺序存放、顺序执行的，数据也一般是以向量、数组、表等形式存储的。

虚拟存储器的定义

基于局部性原理，在程序装入时，仅须将程序当前要运行的少数页面或段先装入内存，而将其余部分暂留在外存，便可启动程序执行。在程序执行过程中，当所访问的信息不在内存时，由操作系统将所需要的部分调入内存，然后继续执行程序。另一方面，操作系统将内存中暂时不使用的内容换出到外存上，从而腾出空间存放将要调入内存的信息。这样，系统好像为用户提供了一个比实际内存容量大得多的存储器，称为虚拟存储器。

请求分页的基本概念/思想

在请求分页系统中，只要求将当前需要的一部分页面装入内存，便可以启动作业运行。在作业执行过程中，当所要访问的页面不在内存中时，再通过调页功能将其调入，同时还可通过置换功能将暂时不用的页面换出到外存上，以便腾出内存空间。

在请求分页系统中，页表应包括哪些数据项？每项的作用是什么？

1. 状态位（存在位）P：它用于指示该页是否已调入内存，供程序访问时参考。
2. 访问字段A：用于记录本页在一段时间内被访问的次数，或记录本页最近已有多长时间未被访问，提供给置换算法（程序）在选择换出页面时参考。
3. 修改位M：标识该页再调入内存后是否被修改过。
4. 外存地址：用于指出该页在外存上的地址，通常是物理块号，供调入该页时参考。

试说明请求分页系统中的地址变换过程。

在进行地址变换时，首先检索快表，试图从中找出所要访问的页。若找到，便修改页表项中的访问位A，供置换算法选换出页面时参考。对于写指令，还需将修改位M置成“1”，表示该页再调入内存后已被修改。否则，去内存中查询页表，如果发生缺页中断则通过OS的调页程序将页面调入内存。最后利用页表项中给出的物理块号和页内地址形成物理地址。

何谓固定分配局部置换和可变分配全局置换的内存分配策略？

1. 固定分配局部置换。固定分配为每个进程分配一组固定数目的物理块，在进程运行期间不再改变。局部置换是指如果进程在运行中发现缺页，则只能从分配给该进程的n个页面中选出一页换出，然后再调入一页，以保证分配给该进程的内存空间不变。
2. 可变分配全局置换。可变分配是指先为每个进程分配一定数目的物理块，在进程运行期间，可根据情况做适当的增加或减少。全局置换是指如果进程在运行中发现缺页，则将OS所保留的空闲物理块取出一块分配给该进程，或者以所有进程的全部物理快为标的，选择一块换出，然后将所缺之页调入。

在请求分页系统中，常采用哪几种页面置换算法？

采用的页面置换算法有：最佳置换算法和先进先出置换算法，最近最久未使用（LRU）置换算法，Clock置换算法，最少使用置换算法，页面缓冲算法等。

试说明改进型Clock置换算法的基本原理。

因为修改过的页面在换出时付出的开销比未被修改过的页面大，在改进型Clock算法中，既考虑页面的使用情况，还要增加置换代价的因素；在选择页面作为淘汰页面时，把同时满足未使用过和未被修改过作为首选淘汰页面。

时钟置换算法

简单时钟置换算法：

给每一帧关联一个附加位，称为使用位。页面初次被调入内存或者刚被访问过时，把使用位置为1。把候选帧集合看作一个循环缓冲区，当需要替换一页时，扫描缓冲区，查找使用位被置为0的帧。每遇到使用位为1的帧时，扫描过后就将其置为0.

改进型时钟置换算法

改进型时钟置换算法在使用位的基础上增加了修改位，用来表示当前帧是否被修改，将页面进一步细分为使用过但未修改，使用过且修改过。每次循环扫描缓冲区时，有未使用过则先置换出为使用过，若全部使用过，则优先置换为修改过的页面。

在请求段表机制中，应设置哪些段表项？

段名、段长、段基址、存取方式、访问字段、修改位、存在位、增补位、外存始地址。

驻留集

驻留集是指为当前进程分配的物理页框的结合

页面分配策略

固定分配局部置换

可变分配局部置换

可变分配全局置换

页面调入时机

预调页策略：根据程序的局部性原理，在程序第一次被调入内存执行时，有程序员指定调入页面的地址信息

请求调页策略：在程序执行过程中，发生缺页中断时，需要调页程序发出调页请求，由操作系统将所需页面调入内存

抖动

抖动是指刚刚换出内存的页面又要换入内存，刚刚换入内存的页面又要换出内存。

抖动发生的原因：操作系统给某个进程分配的页面数量小于进程频繁访问的页面数量

什么是抖动？产生抖动的原因是什么？

抖动就是指当内存中已无空闲空间而又发生缺页中断时，需要从内存中调出一页程序或数据送磁盘的对换区中，而接下来刚被换出的页很快被访问，需重新调入。如此频繁更换页面，使得系统把大部分时间用在了页面的调进换出上，而几乎不能完成任何有效的的工作，我们称这种现象为“抖动”。

产生抖动的原因是系统中 同时运行的进程太多，由此分配给每个进程的物理块太少，不能满足进程正常运行的基本要求，致使每个进程运行时频繁缺页。

4 文件管理

4.1 文件系统基础

什么是文件？

文件是以计算机硬盘为载体存储在计算机上的信息集合，是用户进行输入输出的基本单位。

文件打开

系统将文件的属性从外存复制到内存中的文件打开表中，并将该表目的索引返回给用户.

文件的逻辑结构和物理结构

逻辑结构:这是从用户观点出发所观察到的文件组织形式，即文件是由一系列的逻辑记录组成的，是用户可以直接处理的数据及其结构，它独立于文件的物理特性。

物理结构:又称为存储结构。这是指系统将文件存储在外存上所形成的一种存储组织形式，是用户不能看见的。

逻辑文件的划分

有结构文件,无结构文件

有结构文件:顺序文件,索引文件,索引顺序文件.

1. 顺序文件 :串结构(记录之间的顺序和关键字无关) 顺序结构(记录之间的顺序和关键字有关)
2. 索引文件 :通过建立一张索引表来存储每个记录的长度和地址,加快检索速度
3. 索引顺序文件 :是顺序文件和索引文件的结合,将顺序文件的记录分为若干组,在索引表中为每一组的第一条记录建立一个索引项.各个组内记录可以无序,但组间一定有序.
4. 直接文件（哈希文件）:通过哈希函数直接决定记录地址.

如何提高对变长记录顺序文件的检索速度？

为变长记录文件建立一张索引表，为主文件中的每个记录在索引表中分别设置一个表项，记录指向记录的指针以及记录的长度L，索引表按关键字排序，因此其本身也是一个定长记录的顺序文件，这样就把对变长记录顺序文件的顺序检索转变为对定长记录索引文件的随机检索，从而加快对记录检索的速度，实现直接存取。

文件控制块

文件控制块是存放控制文件所需信息的数据结构，主要信息有：

1. 文件的基本信息。文件名，文件物理位置
2. 存储控制信息。文件的读写权限
3. 使用信息。修改时间，创建时间

索引节点

将文件的描述信息放入索引节点中，目录项仅由文件名和对应索引节点的地址构成

提高了对目录的检索速度

目录管理的要求

1. 实现“按名存取”
2. 提高对目录的检索速度
3. 能够支持文件共享
4. 允许文件重名

目录结构

1. 单级目录：在整个文件系统中只建立一张目录表，每个文件对应一个目录项 查找速度慢，不允许文件重名，不利于文件共享
2. 两级目录：将文件目录分为主文件目录和用户文件目录 不同用户的文件可以重名 但是两级目录结构缺乏灵活性，不能对文件分类
3. 多级目录：两级目录的推广 能够很方便地对文件进行分类，目录结构清晰，不同目录下的文件可以重名 但是在搜索文件时，需要不断读取下一级的文件目录，增加了访问磁盘的次数
4. 有向无环图目录：在树型目录的基础上加入了指向同一节点的有向边。便于文件共享。但是使系统的管理变得更加复杂

目前广泛采用的目录结构形式是哪种？它有什么优点？

树形结构目录。明显提高了对目录的检索速度和文件系统的性能；同时可以很方便地对文件进行分类，层次结构清晰，也能有效地进行文件的管理和保护。

目录查询技术

1. 线性检索法。利用用户提供的文件名，用顺序查找法从文件目录中找到该文件的目录项。
2. 哈希方法。利用用户提供的文件名，并将它变为文件目录的索引值，再利用该索引值到目录中去查找，这样可以显著地提高检索速度

在Hash检索法中，如何解决“冲突”问题？

如果在目录表的相应目录项中的文件名与指定文件名并不匹配，则表示发生了“冲突”，此时须将其Hash值再加上一个常数（该常数与目录的长度值互质），形成新的索引值，再返回到第一步重新开始寻找。

共享文件的方法

1. 基于索引节点的共享方式 采用索引节点的方式，目录项只保留文件名和索引节点地址，索引节点中有共享计数器，用来记录指向该文件的用户目录项的个数。
2. 利用符号链实现文件共享 只有文件主才拥有指向该文件索引节点的指针，共享者只拥有该文件的路径名

基于索引结点的文件共享方式有何优点？

优点是建立新的共享链接时，不改变文件拥有者关系，仅把索引结点共享计数器加1，系统可获悉了由多少个目录项指向该文件。缺点是拥有者不能删除自己的文件否则会出错。

基于符号链的文件共享方式有何优点？

在利用符号链方式实现文件共享时，只是文件主才拥有指向其索引结点的指针；而共享该文件的其他用户则只有该文件的路径名，并不拥有指向其索引结点的指针。这样，就不会发生在文件主删除一共享文件后留下一悬空指针的情况。

文件保护

文件保护通过**口令保护**，**加密保护**和**访问控制**等方式实现

访问控制：为每个文件增加一张访问控制表，以记录每个用户的访问权限

4.2 文件系统的实现

文件系统层次结构

用户调用接口：为用户提供与文件及目录有关的调用

文件目录系统：管理文件目录，管理用户进程的打开文件表

存取控制验证模块：实现文件保护功能，检查用户访问权限以确保合法性

逻辑文件系统与文件信息缓冲区：根据文件的逻辑结构将用户要读写的逻辑记录转换成文件逻辑结构内的相应块号

物理文件系统：把逻辑记录所在的相应块号转换成实际的物理地址

辅助分配模块&设备管理程序模块：分配设备，启动设备，分配缓冲区

文件分配方式

连续分配：每个文件在磁盘上占有一组连续的块

1. 优点 支持顺序访问和随机访问 实现简单存取速度快 作业访问磁盘时需要的寻道数和时间最小
2. 缺点 文件不易动态增长 频繁增删会产生外部碎片

链接分配：采用离散的分配方式

1. 优点 增删改方便 显著提高了磁盘空间的利用率（消除了外部碎片）

隐式 每个目录项都会有一个指向文件首块的指针，每个文件块都有指向下一个文件块的指针，文件块可以分布在磁盘的任何位置，通过指针依次顺序访问

1. 缺点 不能直接访问 不稳定 容易发生数据丢失 指针占用一定的空间

显式 把用于链接文件各物理块的指针从物理块的末尾提取出来，显式地放入内存中的一张链接表中 整个磁盘设置一张 FAT

1. 系统启动后，文件分配表就会被读入内存，提高了检索速度，减少了磁盘的访问次数

索引分配 把每个文件的所有物理块号放在一张索引表中

优点：能够直接访问，没有外部碎片。

缺点：索引表占用一定的存储空间

索引分配的改进方案：

1. 链接方案
2. 多层索引
3. 混合索引

混合索引分配

将多种索引分配方式相结合的分配方式。例如，系统既采用直接地址，又采用单级索引分配方式或两级索引分配方式。

优点：能够较全面地照顾到小型、中型、大型和特大型文件。

1. 对于小文件，将它们的每个盘块地址直接放入FCB，这样就可以直接从FCB中获得该文件的盘块地址，即为直接寻址。
2. 对于中型文件，可以采用单级索引方式，需要先从FCB中找到该文件的索引表，从中获得该文件的盘块地址，即为一次间址。
3. 对于大型或特大型文件，可以采用两级和三级索引分配方式。

文件存储空间管理

1. 空闲表法 为系统外存中所有的空闲盘块建立一张空闲盘快表
2. 空闲链表法 将磁盘上所有的空闲盘块链接起来组成一个链表
3. 位示图法 利用二进制的一位来表示磁盘中一个盘块的使用情况
4. 成组链接法 把顺序的n个空闲盘块号保存在第一个成组链块中，其最后一个空闲盘块(作为成组链块）则用于保存另一组空闲盘块号，如此继续，直至所有空闲盘块均予以链接。系统只需保存指向第一个成组链接块的指针

4.3磁盘的组织和管理

如何改善磁盘系统的性能

1. 首先可通过选择好的磁盘调度算法，以减少磁盘的寻道时间
2. 其次是提高磁盘I/O速度，以提高对文件的访问速度
3. 第三采取冗余技术，提高磁盘系统的可靠性，建立高度可靠的文件系统

磁盘读写时间构成

一次磁盘读写操作的时间构成

1. 寻道时间 磁头移动到指定磁道所需要的时间
2. 旋转延迟时间 1/2*r 磁头定位到某一磁道的扇区所需要的时间
3. 传输时间 从磁盘读出或写入数据所经历的时间

磁盘调度算法

- 先来先服务FCFS 磁头当前位置
- 最短寻找时间优先SSTF
- 扫描算法SCAN（电梯调度算法） 磁头移动方向 往返迂回
- 循环扫描C-SCAN 单向循环
- LOOK 往返迂回 不触碰边缘
- C-LOOK 单向循环 不触碰边缘

目前常用的磁盘调度算法有哪几种？每种算法优先考虑的问题是什么？

1. 先来先服务算法优先考虑进程请求访问磁盘的先后次序
2. 最短寻道时间优先算法优先考虑要求访问的磁道与当前磁头所在磁道距离是否最近
3. 扫描算法考虑欲访问的磁道与当前磁道间的距离，更优先考虑磁头当前的移动方向。

何谓磁盘高速缓存？在设计磁盘高速缓存时需要考虑哪些问题？

磁盘高速缓存是指在内存中为磁盘盘块设置的一个缓冲区，在缓冲区中保存了某些盘块的副本。要考虑的问题有：

(1) 如何将磁盘高速缓存中的数据传送给请求进程；
(2) 采用什么样的置换策略；
(3) 已修改的盘块数据在何时被写回磁盘。

磁盘高速缓存

磁盘高速缓存，是指在内存中为磁盘盘块设置的一个缓冲区，在缓冲区中保存了某些盘块的副本。当出现一个访问磁盘的请求时，先去查看磁盘高速缓冲器，看所请求的盘块内容是否已在磁盘高速缓存中，如果在，便可从磁盘高速缓存中去获取，这样就省去了启动磁盘操作，而且可使本次访问速度提高几个数量级。如果不在，才需要启动磁盘将所需要的盘块内容读入，并把所需盘块内容送给磁盘高速缓存，以便以后又需要访问该盘块的数据时，便可直接从高速缓存中提取。

如何将磁盘高速缓存中的数据传送给进程（数据交付方式）

1. 数据交付，这是直接将高速缓存中的数据传送到请求者进程的内存工作区中
2. 指针交付，只将指向高速缓存中某区域的指针交付给请求者进程。

如何选择磁盘高速缓存的调度算法

较为常用的算法仍然是最近最久未使用算法LRU，最近未使用算法NRU（时钟置换算法）等。除此之外还要考虑一下几点：

1. 访问频率
2. 可预见性
3. 数据的一致性

已修改的盘块数据在何时被写回磁盘

系统中专门增设了一个修改程序，使之在后台运行，该程序周期性地调用一个系统调用SYNC。其主要功能是强制性地将所有在高速缓存中已修改的盘块数据写回磁盘。

数据的一致性

保存在多个文件中的统一数据，在任何情况下都必须能保证相同

事务的定义和特性

事务是用于访问和修改各种数据项的一个程序单位。事务也可以被看做是一系列相关读和写操作。

特性：ACID

1. 原子性(Atomic) 一个事务在对一批数据执行修改操作时，应该是要么全部完成，并用修改后的数据去代替原来的数据，要么一个也不修改。
2. 一致性(Consistent)， 即事务在完成时，必须使所有的数据都保持一致状态
3. 隔离性(Isolated)， 即对一个事务对数据所作的修改，必须与任何其它与之并发事务相隔离。换言之，一个事务查看数据时数据所处的状态，要么是另一并发事务修改它之前的状态，要么是另一事务修改它之后的状态，而不会是任何中间状态的数据
4. 持久性(Durable)， 即事务完成之后，它对于系统的影响是永久性的。

并发控制

在多用户系统和计算机网络环境下，可能有多个用户在同时执行事务。由于事务具有
原子性，这使各个事务的执行必然是按某种次序依次进行的，只有在一个事务执行完后，才允许另一事务执行，即各事务对数据项的修改是互斥的。人们把这种特性称为顺序性，而把用于实现事务顺序性的技术称为并发控制。

并发控制有哪些方法？

1. 利用互斥锁实现顺序性
2. 利用互斥锁和共享锁实现顺序性

5 IO管理

IO系统的层次结构及功能

1. 用户层IO软件 用于实现用户与I/O设备交互
2. 设备独立性软件 用于实现用户程序与设备驱动器的统一接口、设备命令、设备保护，以及设备分配与释放等。
3. 设备驱动程序 与硬件直接有关，用来具体实现系统对设备发出的操作指令，驱动I/O设备工作
4. 中断处理程序 用于保存被中断进程的CPU环境，转入相应的中断处理程序进行处理，处理完后恢复现场，并返回到被中断的进程
5. 硬件

试说明IO系统的基本功能

隐藏物理设备的细节、与设备的无关性、提高处理机和I/O设备的利用率、对I/O设备进行控制、确保对设备的正确共享、错误处理

什么是中断处理程序

当有I0设备发来中断请求信号时，在中断硬件做了初步处理后，便转向中断处理程序。它首先保存被中断进程的CPU环境，然后转入相应设备的中断处理程序进行处理，在处理完成后，又恢复被中断进程的CPU环境，返回断点继续运行。

什么是设备驱动程序

它处于I/O系统的次底层，是进程和设备控制器之间的通信程序，其主要功能是，将上层发来的抽象I/O请求转换为对I/O设备的具体命令和参数，并把它装入到设备控制器中的命令和参数寄存器中。

设备驱动程序的主要任务

其主要任务是接收上层软件发来的抽象I/O要求，如read或write命令，再把它转换为具体要求后，发送给设备控制器，启动设备去执行。

设备驱动程序的功能

1. 接收由与设备无关的软件发来的命令和参数
2. 检查用户I0请求的合法性，了解I/0设备的工作状态
3. 发出I/O命令，如果设备空闲，便立即启动I/O设备，完成指定的I/O操作;如果设备忙碌，则将请求者的请求块挂在设备队列上等待。
4. 及时响应由设备控制器发来的中断请求，并根据其中断类型，调用相应的中断处
   理程序进行处理。

IO控制方式

1. 程序直接控制方式 计算机从外部设备读取的每个字，CPU需要对外设状态进行循环检查，直到确定该字已经在I/O控制器的数据寄存器中。
2. 中断驱动方式 IO 控制器从CPU接收一个读命令，然后从外部设备读数据。一旦数据读入I/O0控制器的数据寄存器，便通过控制线给CPU发出中断信号，表示数据已准备好，然后等待CPU请求该数据。I/O 控制器收到CPU发出的取数据请求后，将数据放到数据总线上，传到CPU的寄存器中。
3. DMA方式 在IO设备和内存之间开辟直接的数据交换通路，彻底释放CPU。数据传送的基本单位是数据块，所传送的数据是从设备直接到内存，仅在一个或多个数据块开始和结束时，才需要CPU干预。数据块的传送是在DMA控制器控制下完成的
4. 通道控制方式 IO通道是指专门负贵输入/输出的处理机。I/O通道方式是DMA方式的发展，它可以进一步减少CPU的干预，即把对一个数据块的读(或写)为单位的干预，减少为对一-组数据块的读(或写)及有关控制和管理为单位的干预。

DMA方式和中断方式的主要区别

DMA方式与中断方式的主要区别是：

1. 中断方式在每个数据需要传输时中断CPU，而DMA方式则是在所要求传送的一批数据全部传送结束时才中断CPU

2. 中断方式的数据传送是在中断处理时由CPU控制完成的，而DMA方式则是在DMA控制器的控制下完成的。

I/O通道与DMA方式的区别

1. DMA方式需要CPU来控制传输的数据块大小、传输的内存位置，而通道方式中这些信息是由通道控制的。
2. 每个DMA控制器对应一台设备与内存传递数据，而一个通道可以控制多台设备与内存的数据交换。

试说明I/0控制发展的主要推动因素是什么？

促使I/0控制不断发展的几个主要因素如下：

1. 尽量减少CPU对I/O控制的干预，把CPU从繁杂的I/O控制中解脱出来，以便更多地去完成数据处理任务。
2. 缓和CPU的高速性和设备的低速性之间速度不匹配的矛盾，以提高CPU的利用率和系统的吞吐量。
3. 提高CPU和I/O设备操作的并行程度，使CPU和I/O设备都处于忙碌状态，从而提高整个系统的资源利用率和系统吞吐量。

有哪几种I/O控制方式？各适用于何种场合？

1. 程序I/O方式：适用于早期的计算机系统中，并且是无中断的计算机系统；
2. 中断驱动I/O控制方式：普遍用于现代的计算机系统中；
3. DMA I/O控制方式：适用于I/O设备为块设备时在和主机进行数据交换的一种I/O控制方式；
4. I/O通道控制方式：当I/O设备和主机进行数据交换是一组数据块时通常采用I/O通道控制方式，但此时要求系统必须配置相应的通道及通道控制器。

试说明DMA的工作流程

如图 5.3(c)所示，DMA 方式的工作过程是：CPU 接收到 I/O 设备的 DMA 请求时，它给 DMA 控制器发出一条命令，同时设置 MAR 和 DC 初值，启动 DMA 控制器，然后继续其他工作。之后 CPU 就把控制操作委托给 DMA 控制器，由该控制器负责处理。DMA 控制器直接与存储器交互，传送整个数据块，每次传送一个字，这个过程不需要 CPU 参与。传送完成后，DMA 控制器发送一个中断信号给处理器。因此只有在传送开始和结束时才需要 CPU 的参与。

什么是设备控制器？主要功能是什么？

执行控制IO的电子部件称为设备控制器，控制一个或多个IO设备，以实现IO设备和计算机之间的数据交换

主要功能如下：

1. 接受和识别命令
2. 数据交换
3. 标识和报告设备的状态
4. 地址识别
5. 数据缓冲区
6. 差错控制

设备控制器的组成

1. 设备控制器与处理机的接口。该接口用于实现CPU和设备控制器之间的通信
2. 设备控制器与设备的结构。该接口用于实现设备和设备控制器之间的通信，并且一个设备控制器可以控制多个设备
3. IO逻辑。IO逻辑用于实现对设备的控制

设备无关性的基本含义是什么？为什么要设置该层？

应用程序独立于具体使用的物理设备。为了实现设备独立性而引入了逻辑设备和物理设备两概念。在应用程序中，使用逻辑设备名称来请求使用某类设备；而系统在实际执行时，还必须使用物理设备名称。

为了提高OS的可适应性和可扩展性，在现代OS中都毫无例外地实现了设备独立性，也称设备无关性。

设备独立性的优点

1. 方便用户编程
2. 使程序运行不受具体机器环境的限制
3. 便于程序移植

使用逻辑设备名的好处

使用逻辑设备名的好处是

1. 增加设备分配的灵活性
2. 易于实现IO重定向

设备分配的原则

设备分配应根据设备特性、用户要求和系统配置情况。既要充分发挥设备的使用效率，又要避免造成进程死锁，还要将用户程序和具体设备隔离开。

设备分配方式

设备分配方式有静态分配和动态分配两种。

1. 静态分配主要用于对独占设备的分配，它在用户作业开始执行前，由系统一次性分配该作业所要求的全部设备、控制器。一旦分配， 这些设备、控制器就一直为 该作业所占用，直到该作业被撤销。静态分配方式不会出现死锁，但设备的使用效率低。
2. 动态分配在进程执行过程中根据执行需要进行。当进程需要设备时，通过系统调用命令向系统提出设备请求，由系统按某种策略给进程分配所需要的设备和控制器。有利于提高设备的利用率，但若分配算法使用不当，可能会造成死锁

设备映射表

建立逻辑设备和物理设备的对应关系

Spooling技术的组成（假脱机技术

1. 输入井和输出井。这是在外存上开辟出的两个存储空间
2. 输入缓冲区和输出缓冲区。这是在内存中开辟的两个缓冲区，用于缓和CPU和磁盘之间速度不匹配的矛盾
3. 输入进程和输出进程。输入进程也称为预输入进程，用于模拟脱机输入时的外围
   控制机，将用户要求的数据从输入设备传送到输入缓冲区，再存放到输入井。
4. 井管理程序。用于控制作业与磁盘井之间信息的交换

SPOOLing技术的特点

1. 提高了IO的 速度
2. 将独占设备改造为共享设备
3. 实现了虚拟设备功能

何谓设备虚拟？实现设备虚拟式所依赖的关键技术是什么？

通过虚拟技术可将一台独占设备变换成若干台逻辑设备，供若干个用户（进程）同时使用，通常把这种经过虚拟技术处理后的设备称为虚拟设备。其实现所依赖的关键技术是SPOOLING技术。

假脱机系统向用户提供共享打印机的基本思想是什么？

对每个用户而言，系统并非即时执行其程序输出数据的真实打印操作，而只是即时将数据输出到缓冲区，这时的数据并未真正被打印，只是让用户感觉系统已为他打印；真正的打印操作，是在打印机空闲且该打印任务在等待队列中已排到队首时进行的；以上过程是对用户屏蔽的，用户是不可见的。

引入缓冲的主要原因是什么？

1. 缓和CPU与I/0设备之间速度不匹配的矛盾
2. 减少对CPU的中断频率
3. 放宽对中断响应时间的限制

4. 解决数据力度不匹配的问题
5. 提高CPU和I/0设备之间的并行性。