◎第一章 操作系统引论

1、设计现代 OS 的主要目标是：**有效性、方便性、可扩展性、开放性**。

2、操作系统的五大功能是：**处理机管理、存储器管理、设备管理、文件管理、用户管理**。

3、操作系统的基本特性是：**并发性、共享性、异步性、虚拟性。**其中最基本特征是**并发**和**共享。**最重要的特征是**并发性** 。**虚拟性：**时分复用技术、空分复用技术

4、操作系统的作用：**作为用户与计算机硬件系统之间的接口、OS 作为计算机系统资源的管理者、OS 用作扩充机器**。**分类：**批处理系统，分时系统，实时系统

5、以下不是微内核 OS 特点的是 ABCD

A、足够小的内核 B、应用“机制与策略分离”的原理

C、基于客户/服务器模式 D、采用面向对象技术

注：微内核 OS 运行效率并不高，它还有一个特点是基于客户/服务器模式

★第二章 进程管理

1、简述进程的定义：进程是进程实体的运行过程，是系统进行资源分配和调度的一个独立单位

★2、说明进程与程序的区别（进程 4 个特征：动态、并发、独立、异步）

（1）**动态性：**程序是指令的有序集合，其本身没有任何运行的含义，它是一个静态的概念。而进程是程序在处理机上的一次执行过程，它是一个动态概念。

（2）程序的存在是永久的。而进程则是有生命期的，它因创建而产生，因调度而执行，因得不到资源而暂停，因撤消而消亡。

（3）程序仅是指令的有序集合。而进程则由程序段、相关数据段.进程控制块（PCB）组成。

（4）进程与程序之间不是一一对应。

（5）**并发性：**多个进程实体同存于内存中，且能在一段时间内同时运行。并发性是进程的重要特征，也是 OS 的重要特征。引入进程的目的也正是为了使其进程实体能和其它进程实体并发执行；而程序(没有建立 PCB)是不能并发执行的。

（6）**独立性：**在传统的 OS 中，独立性是指进程实体是能独立分配资源和独立接受调度，能独立运行的基本单位。凡未建立 PCB 的程序都不能作为一个独立的单位参与运行。

（7）**异步性：**进程实体按异步方式运行,推进次序每次不一定相同。

3、程序段、相关数据、**PCB（进程控制块）**三部分构成了进程实体。

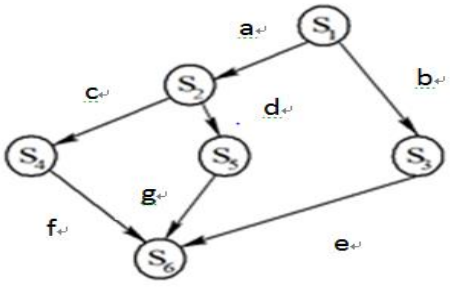
4、引起挂起状态的原因有：**终端用户的请求、父进程的请求、负荷调节的需要、操作系统的需要。**

5、原语(Primitive)是由若干条指令组成的，用于完成一定功能的一个过程，原语在执行中不允许被中断，原语的作用是**实现进程的通信和控制。**常见的几种元语：**创建原语 create()**（功能：创建一新进程）、**阻塞原语 block()**（功能：将进程由执行状态转为阻塞状态）、**唤醒原语 wakeup()**（功能：将进程由阻塞状态变为就绪状态）、**挂起原语 suspend()**（功能：将指定进程或处于阻塞状态的进程挂起）、**激活原语 active()**（功能：将指定进程激活）。

6、同步机制应遵循的四条规则是：**空闲让进、忙则等待、有限等待、让权等待。**

★7、试写出相应的程序来描述右图所示的前驱图

```
P1(){S1;signal(a);signal(b);}
P2(){wait(a);S2;signal(c);signal(d);}P3\P4\P5\P6 略
Main{semaphore a,b,c,d,e,f,g;
      a.value=b.value=c\d\e\f\g.value=0;
      cobegin
      P1();P2();P3();P4();P5();P6();
      coend}
```



7、**程序并发执行的特征：**1.间断性 2.失去封闭性 3.不可再现性

8、管程的定义

一个管程定义了一个数据结构和能为并发进程所执行的一组操作，这组操作能同步进程和改变管程中的数据。

9、高级通信机制分类：**共享存储器系统、消息传递系统、管道通信系统、客服机-服务器系统**

10、操作系统中引入进程的目的是**为了使多个程序能并发执行，以提高资源利用率和系统吞吐量**，在操作系统中再引入线程，则是为了**减少程序在并发执行时所付出的时空开销。**

11、进程与线程的比较

	进程	线程
引入目的	能并发执行，提高资源的利用率和系统吞吐量.	提高并发执行的程度，减小开销，进一步提高资源的利用率和系统吞吐量.
并发性	较低	较高
基本属性（调度）	资源拥有的基本单位—进程	独立调度/分派的基本单位—线程
基本状态	就绪； 执行；等待	就绪；执行；等待
拥有资源	资源拥有的基本单位—进程	无资源分配
系统开销	创建/撤消/切换时空开销较大	创建/撤消/切换时空开销较小
系统操作	创建，撤消，切换	创建，撤消，切换
存在标志	进程控制块PCB	线程控制块TCB
关系	单进程单线程；单进程多线程；多进程单线程；多进程多线程	

13、当一个进程完成了特定任务后，系统回收这个进程所占的**主存空间**和取消该进程的**进程控制块（PCB）**就撤销了该进程。

14、当一个进程独占处理器顺序执行时，具有两个特性：**封闭性**和**可再现性**。

15、对信号量 S 的操作只能通过**原语**操作进行，对应每一个信号量设置了一个等待队列。

16、在操作系统中，进程是一个**资源分配**的基本单位，也是一个独立运行和**调度**的基本单位。

17、**PCB：**它是就进程实体的一部分.是操作系统中最重要的记录性数据结构,PCB 中记录了操作系统所需的用于描述进程的当前情况以及控制进程运行的全部信息信号量的应用

★第三章 处理机调度与死锁

1、一个作业从提交开始，往往要经历三级调度：**高级调度、低级调度、中级调度**。

高级调度：又称为作业调度或长程调度，其主要功能是根据某种算法，把外存上处于后备队列中的那些作业调入内存。它调度的对象是作业。

低级调度：又称进程调度或短程调度。主要功能：保存处理机的现场信息、按某种算法选取进程、把处理器分配给进程。常采用**非抢占（非剥夺）方式**和**抢占（剥夺）方式**两种。它所调度的对象是进程（或内核级线程）。

中级调度：又称中程调度。在内存和外存对换区之间按照给定的原则和策略选择进程对换，以解决内存紧张问题，从而提高内存的利用率和系统吞吐量。

2、选择调度方式和调度算法的准则

面向用户的准则：（1）周转时间短(2) 响应时间快(3) 截止时间的保证(4) 优先权准则

面向系统的准则：(1) 系统吞吐量高(2) 处理机利用率好(3) 各类资源的平衡利用

3、常见的调度算法有：**先来先服务调度算法（FCFS）、短作业/进程优先调度算法（SJF/SPF）、高优先权优先调度算法、基于时间片的轮转调度算法。**

4、进程调度算法采用时间片轮转法时，时间片过大会使轮转法转为**先来先服务调度算法（FCFS）**。

5、若使当前运行的进程总是优先级最高的进程，则应该选择进程**高优先权优先**调度算法

6、在响应比最高者优先的作业调度算法中，当各个作业等待时间相同时，**运行时间短**的作业将得到优先调度；当各个作业要求运行的时间相同时，**等待时间长**的作业得到优先调度。

7、一个理想的作业调度算法应该是既能**提高系统的效率**，又能使系统的作业**及时得到结果**。

8、常用的几种实时调度算法有：**最早截止时间优先算法（EDF）、最低松弛度优先算法（LLF）**

最早截止时间优先算法：该算法是根据任务的开始截止时间来确定任务的优先级。开始截止时间越早，其优先级越高

最低松弛度优先算法：该算法是根据任务紧急（或松弛）的程序，来确定任务的优先级。任务的紧急度越高，其优先级越高，并使之优先执行

9、死锁的定义：指多个进程在运行过程中因争夺资源而造成的一种僵局（deadly-Embrace)，若无外力作用，这些进程都将无法向前推进。

10、产生死锁的原因有两个：**竞争资源、进程间推进顺序非法**。

★11、产生死锁的四个必要条件是**互斥条件、请求和保持条件、不剥夺条件、循环等待条件**。

12、处理死锁的基本方法是：**预防死锁、避免死锁、检测死锁、解除死锁**。

13、什么是系统的安全状态，避免死锁的实质是什么？

系统的安全状态是指在某一时刻，系统能按某种进程顺序(p_1，p_2,…，p_n)来为每个进程 Pi 分配其资源,直到满足每个进程对资源的最大需求,使每个进程都可顺利地完成，则称此时的系统状态为安全状态.称序列<p_1, p_2,…，p_n>为安全序列。如果一个系统在安全状态，就没有死锁。如果一个系统处于不安全状态，就有可能死锁。避免死锁的实质：确保系统不进入不安全状态。

14、确定系统处于 S 为死锁状态的充分条件是：**当且仅当 S 状态的资源分配图是不可完全简化的。**

15、常用的解除死锁方法有两种:**资源剥夺法、撤消进程法**。

资源剥夺法：当发现死锁后,从其进程剥夺足够数量的资源给死锁进程,以解除死锁状态。

撤消进程法：采用强制手段从系统中撤消一个/一部分死锁进程,并剥夺这些进程的资源供其它死锁进程使用。

例解：⑴该状态是安全的，因为存在一个安全序列< $P_0P_3P_4P_1P_2$>。下表为该时刻的安全序列表。 ⑵若进程提出请求 Request(1，2，2，2)后，系统不能将资源分配给它，若分配给进程 P2，系统还剩的资源情况为（0，4，0，0），此时系统中的资源将无法满足任何一个进程的资源请求，从而导致系统进入不安全状态，容易引起死锁的发生。

★第四章　存储器管理

❖ **本章内容：**存储器的层次结构、连续分配方式、基本分页存储管理方式、基本分段存储管理方式、虚拟存储器的基本概念、页面置换算法

❖ **主要考点：**存储器的层次结构、★地址变换、虚拟存储器的概念、特征、**★★页面置换算法**

1、多级存储器结构：Cpu 寄存器，高速缓存，主存储器，磁盘缓存，磁盘，可移动存储介质。

速度快，价格高，容量小→→→→容量大，不可执行

程序的装入：绝对装入方式，可重定位装入方式，动态运行装入方式

程序的链接：静态链接方式，装入时动态链接，运行时动态链接

2、在动态分区式内存分配算法中，倾向于优先使用低地址部分空闲区的算法是**首次适应算法**；能使内存空间中空闲区分布较均匀的算法是**循环首次适应算法**。

3、连续分配方式：单一连续，固定分区，动态分区，动态可重定位分区/分配

固定分区分配：存储管理方法:将内存用户空间分为若干固定大小的区域。除 OS 占一区外，其余的一个分区装入一道程序。分区的大小事先必须确定

主要特点：管理简单，但因作业的大小并不一定与某个分区大小相等，从而使一部分存储空间被浪费。所以主存的利用率不高。

动态分区分配：存储管理方法: 在作业进入内存时，根据作业的大小动态地建立分区，并使分区的大小正好适应作业的需要

主要特点: 管理简单，只需小量的软件和硬件支持，便于用户了解和使用。进程的大小与某个分区大小相等，从而主存的利用率有所提高。

分区分配算法: 首次适应算法、循环首次适应算法、最佳适应算法、最坏适应算法、快速适应算法

4、动态存储分配时，要靠硬件地址变换机构实现**重定位**。

5、为了实现进程对换，系统必须实现三方面功能：**对换空间的管理、进程的换入和进程的换出**。

6、分页的作用是**实现从页号到物理块号的地址映射**。

分段的的优点：方便编程，信息共享，信息保护，动态增长，动态链接。

7、简述分页和分段有何相同点和不同点

相同点：分页和分段都采用离散分配的方式，且都要通过地址映射机构来实现地址变换

不同点：

（1）从功能上看，页是信息的物理单位，分页是为实现离散分配方式，以消减内存的外零头，提高内存的利用率，即满足系统管理的需要，而不是用户的需要；而段是信息的逻辑单位，它含有一组其意义相对完整的信息，目的是为了能更好地满足用户的需要。

（2）页的大小固定且由系统确定，而段的长度却不固定，决定于用户所编写的程序。

（3）分页的作业地址空间是一维的，而分段的作业地址空间是二维的。

8、虚拟存储器的定义

虚拟存储器是指仅把作业的一部分装入内存便可运行作业的存储管理系统，它具有请求页调入功能和页置换功能，能从逻辑上对内存容量进行扩充。

9、虚拟存储器具有**多样性、对换性、虚拟性**三大主要特征。

10、在存储管理中常用**虚拟存储器**方式来摆脱主存容量的限制。

11、**虚拟存储器的实现方法：**1.分页请求系统 2.请求分段系统。

◎5 章 1、I/O 系统设备分类 ：按使用特性分类：存储设备，输入/输出设备

按传输速度的高低 ：低速设备（键盘、鼠标），中速设备（打印机），高速设备（磁盘）

按信息交换的单位： 块设备（磁盘）， 字符设备（交互式终端、打印机）

按设备共享属性：独占设备， 共享设备（磁盘）， 虚拟设备

2、所谓**设备控制器**，是一块能控制一台或多台外围设备与 CPU 并行工作的硬件

3、**设备控制器基本功能：**1 接收和识别命令 数据交换 标识和报告设备的状态 地址识别 数据缓冲差错控制 。2 检查用户 I/O 请求的合法性，了解 I/O 设备的状态，传递有关参数，设置设备的工作方式。 3 发出 I/O 命令 。4 及时响应由控制器或通道发来的中断请求，并根据其中断类型调用相应的中断处理程序进行处理 。5 对于设置有通道的计算机系统，驱动程序还应能够根据用户的 I/O 请求，自动地构成通道程序。

4、 **I/O 通道类型：**字节多路通道、数组选择通道、数组多路通道。**主要目的**是为了建立独立的 IO 操作,不仅使数据的传送能独立于 cpu,而且也希望有关对 IO 操作的组织,管理及结束处理尽量独立,以保证 cpu 有更多的时间去进行数据处理。

5、 **设备驱动程序的特点：**1.驱动程序汉族要是指在请求 I/O 的进程与设备控制器之间的一个通信和转换程序 2.驱动程序与设备控制器和 I/O 设备的硬件特性紧密相关，因而对不同类型的设备应配置不同的驱动程序 3.驱动程序与 I/O 设备所采用的 I/O 控制方式紧密相关 4.由于驱动程序与硬件紧密相关，因而其中的一部分必须用汇编语言书写。5.驱动程序应允许可重入。6.驱动程序不允许系统调用。

6、 设备驱动程序的主要功能：1）接收由设备独立性软件发来的命令和参数，并将命令中的抽象要求转换为具体要求；2）检查用户 I/O 请求的合法性，了解 I/O 设备的状态，传递有关参数，设置设备的工作方式；3）发出 I/O 命令；4）及时响应由控制器或通道发来的中断请求，并根据其中断类型调用相应的中断处理程序进行处理；5）对于设置有通道的计算机系统，驱动程序还应能根据用户的 I/O 请求，自动地构成通道程序。

7、设备处理方式：1.为每一类设备设置一个进程，专门用于执行这类设备的 I/O 操作。2.在整个系统中设置一个 I/O 进程，专门用于执行系统中所有各类设备的 I/O 操作。3.不设置专门的设备处理进程，而只为各类设备设置相应的设备处理程序，供用户进程或系统进程调用

8、设备独立性：指用户设备独立于所使用的具体物理设备。即在用户程序中要执行 I/O 操作时，只需用逻辑设备名提出 I/O 请求，而不必局限于某特定的物理设备。

9、为什么要引入设备独立性？如何实现设备独立性？

引入设备独立性，可使应用程序独立于具体的物理设备，是设备分配具有灵活性。另外容易实现 I/O 重定向。

为了实现设备独立性，必须在设备驱动程序之上设置一层设备独立性软件，用来执行所有 I/O 设备的公用操作，并向用户层软件提供统一接口。关键是系统中必须设置一张逻辑设备表 LUT 用来进行逻辑设备到物理设备的映射，其中每个表目中包含了逻辑设备名、物理设备名和设备驱动程序入口地址三项；当应用程序用逻辑设备名请求分配 I/O 设备时，系统必须为它分配相应的物理设备，并在 LUT 中建立一个表目，以后进程利用该逻辑设备名请求 I/O 操作时，便可从 LUT 中得到物理设备名和驱动程序入口地址。

10、主存储器与外围设备之间的数据传送控制方式有**程序直接控制、中断驱动方式、DMA 方式**和**通道控制方式。**

11、使用缓冲区能有效地缓和 **I/O** 设备和 **CPU** 之间速度不匹配的矛盾。

12、缓冲区的设置可分为**单缓冲、双缓冲、多缓冲**和**缓冲池**。

14、SPOOLing：即同时联机外围操作，又称脱机操作。在多道程序环境下，可利用多道程序中的一道程序，来模拟脱机的输入输出功能。即在联机条件下，将数据从输入设备传送到磁盘，或从磁盘传送到输出设备。

15、SPOOLing 系统特点：提高了 I/O 的速度、将独占设备改造成共享设备、实现了虚拟设备功能。

16、SPOOling 系统的组成:1.输入井和输出井 2.输入缓冲区和输出缓冲区 3.输入进程和输出进程 4.I/O 请求队列

◎第六章 文件管理

1、文件定义：是指记录在外存上的具有文件名的一组相关信息的集合

文件属性：文件名、文件类型、文件长度、文件的物理位置、文件的建立日期以及用户对该文件的存取权限等

文件的特点：文件具有保存性、文件是按名存取、文件的内容是一组信息的集合

2、文件的逻辑结构分为**流式文件（无结构文件）和记录式文件（有结构文件）**二种。

3、文件的物理结构分为**顺序文件、索引文件和索引顺序文件**三种。

4、文件系统中，用于文件的描述和控制并与文件一一对应的是**文件控制块 FCB**。

5、对目录进行查询的方式常见的有两种：**线性检索法和 Hash 方法**。

6、目前广泛采用的目录结构是**树型目录结构**。

7、Hash 检索法有何优点？又有何局限性？

在 Hash 检索法中，系统利用用户提供的文件名并将它变换为文件目录的索引值，再利用该索引值到目录中去查找，这样能有效地提高目录的检索速度，但 Hash 检索法也有局限性即对于使用了通配符的文件名，系统是无法使用 Hash 检索法检索目录的。

8、**顺序文件的优缺点:**优点:a 对诸记录进行批量存取时,存取效率最高 b 只有顺序文件才能存储在磁带上并能有效的工作;缺点 a 交互应用时性能差 b 增加或修改一个记录比较困难,为了解决这一问题可以为顺序文件配置一个运行记录文件(log file)或称为事物文件(transactionFile)把试图增加删除或者修改的信息记录于其中,规定每个一定时间例如 4 个小时,将运行记录文件于原来的主文件加以合并,产生一个按关键字排序的新文件

9、**索引文件：**使用索引文件的主要问题是，它除了有主文件外，还须配置一张索引表，而且每个记录都要有一项索引项，因此提高了存储费用。

10、**文件分配方式：**

连续分配连续分配要求为每一个文件分配一组相邻接的盘块。一组盘块的地址定义了磁盘上的一段线性地址。连续分配的主要优点如下：1.顺序访问容易。2.顺序访问速度快。 连续分配的主要缺点如下：1.要求有连续的存储空间。2.必须事先知道文件的长度。

链接分配（隐式链接，显示链接）隐式链接：用采用隐式链接分配方时，在文件目录的每个目录项中，都须含有指向链接文件的第一个盘块和最后一个盘块的指针。

显示链接:这是指吧链接文件各物理块的指针，显示的存放在内存的一张链接表中。该表整个磁盘仅设置一张。

11、**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 文件共享 4.允许文件重名。

八、九章

1、**接口：**字符显示式联机用户接口，图形化联机用户接口，脱机用户接口

2、**联机命令的类型 ：**磁盘访问类、文件操作类、目录操作类、其他命令

3、**系统调用：**在操作系统核心中设置了一组用于实现各种系统功能的子程序（过程），并将他们提供给应用程序调用。应用程序利用一种系统调用命令，去调用所需的系统过程。是用户程序取得 OS 服务的唯一途径。是一种特殊的过程调用。

4、**系统调用类型：**(1)进程控制类系统调用(2)文件操纵类系统调用 (3)进程通信类系统调用

代码题

1 利用记录型信号量解决生产者—消费者问题

```
Int in=0,out=0; ■
      Item buffer[n]; ■
   ■  semaphore mutex=1,empty=n,full=0;
      Void proceducer(){do{ ■
                producer an item nextp; … ■
                wait(empty);   wait(mutex); ■
                buffer(in) =    nextp;
■               in    =   (in+1) %n; ■
                signal(mutex); ■   signal(full); ■
               }while(TRUE)}
      Void consumer(){do{
                wait(full); ■   wait(mutex); ■
                nextc   =    buffer(out); ■
                out     =   (out+1) %n; ■
                signal(mutex); ■      signal(empty); ■
                consumer the item in nextc;~
                }while(TRUE)}
Void main(){cobegin proceducer();consumer(); coend}
```

3 利用管程解决生产者——消费者问题

```
Monitor producer-consumer{
 Int in=0,out=0; ■
      Item buffer[n];
   Condition notfull,notempty;
Int count;public:
 Void put(item nextp){
       if (count>=n ) cwait(notfull);
    Buffer[in]=nextp;          in:=(in+1)% n;
       Count++; csignal(notempty)}
 Void get(item nextc){
   if (count<=0 ) cwait(notempty);
      nextc=buffer[out];   out:=(out+1)%n; count--;
    Csignal(notfull);}
  {in=0;out=0;count=0;}}PC
Void Producer(){item x;while(true){~
    produce an item in nextp   ; PC.put(x) ;}}
Void Consumer(){iten x;while(true){
    PC.get(x); consumer the item in nextc;~ }}
Void main(){cobegin proceducer();consumer(); coend}
```

2 小和尚老和尚提水进程：

```
（定义）Var mutex_jar,mutex_well: semaphore := 1,1
        empty, full, buckets :semaphore := 30, 0, 5;
        小和尚 young_monk :begin    repeat
       wait(empty); wait(buckets);
        go to the well;
        wait(mutex_well);
         get water;
         signal(mutex_well);
        go to the temple;
       wait(mutex_jar);
        pure the water into the jar;
        signal(mutex_jar);
         signal（buckets);
       signal(full);
        until false;
        end
       老和尚 old_monk :begin    repeat
        wait(full); wait(buckets);wait(mutex_jar);
            get water;
         signal(mutex_jar);signal(buckets);
          signal(empty); until false;
        end
```

4 读者-写者问题

```
semaphore rmutex=1,wmutex=1;int rdcount=0
读者 void reader(){do{wait(rmutex);
if(rdcount=0)wait(wmutex);rdcount++;
signal(rmutex);~~perfor read operation;~~
wait(rmutex);rdcount--;
if(rdcount=0)signal(wmutex);signal(rmutex);
}while(TRUE);}
写者 void writer(){do{wait(wmutex);
perform write operation;signal(wmutex);
}while(TRUE);}
Void main(){cobegin
 reader();writer();
 coend}
```