操作系统概论-02323（2017年张琼声版本）

第一章：操作系统简介

操作系统概念：操作系统是一种复杂的系统软件，是不同程序代码、数据结构、初始化文件的集合，可执行。

操作系统是提供计算机用户与计算机硬件之间的接口，并管理计算机软件和硬件资源，并且通过这个接口使应用程序的开发变得简单、高效。

接口是两个不同部分的交接面。接口分为硬件接口和软件接口，计算机的所有功能最终都是由硬件的操作来实现的，计算机屏蔽了对硬件操作的细节。

操作系统完成的两个目标：

与硬件相互作用，为包含在所有硬件平台上的所有底层可编程部件提供服务。

为运行在计算机系统上的应用程序（即用户程序）提供执行环境

现代计算机特点是支持多任务，，一方面保证用户程序的顺利执行，另一方面使计算机系统资源得到高效的利用，保证计算机系统的高性能

操作系统的功能：处理机管理、内存管理、设备管理、文件管理。

* 操作系统的发展：

无操作系统--单道批处理系统--多道批处理系统--微机操作系--实时操作系统

无操作系统阶段：电子管，无存储设备，第一台：1946年宾夕法尼亚大学的「埃尼阿克」

单道批处理系统：晶体管，磁性存储设备，内存中有一道批处理作业，计算机资源被用户作业独占。

吞吐量是指单位时间内计算机系统处理的作业量

多道程序系统：集成电路芯片，出现了分时操作系统（多个终端）。

微机操作系统：第一台Intel公司顾问GaryKildall 编写的CP/M系统，是一台磁盘操作系统，用于Intel8080.

实时操作系统：广泛应用于各种工业现场的自动控制、海底探测、智能机器人和航空航天等。

* 批处理、实时、分时系统的优缺点比较：

单道批处理系统：自动性、顺序性、单道性。优点：减少了等待人工操作的时间

缺点：CPU资源不能得到有效的利用。

多道批处理系统：多道性、无序性、调度性、复杂性。优点：能够使CPU和内存IO资源得到充分利用，，提高系统的吞吐量。缺点：系统平均周转时间长，缺乏交互能力。

分时系统：多路性、及时性、交互性、独立性。优点：提供了人机交互，可以使用户通过不同终端分享主机。缺点：不能及时接收及时处理用户命令。

实时操作系统（用户实时控制和实时信息处理）：多路性、独立性、及时性、交互性、可靠性。在实时系统中，往往采取多级容错措施来保证系统安全和数据安全。

操作系统产品：主机操作系统（批处理、事务处理（银行支票处理或航班预订）、分时处理），微机操作系统，服务器操作系统、嵌入式操作系统（物联网操作系统）

操作系统特征：并发（多个事件在同一时间间隔内同时发生）、共享、虚拟、异步

操作系统功能：

内存管理：任务是为多道程序的运行提供良好的运行环境，方便用户使用内存，提高内存利用率，以及从逻辑上扩充内存实现虚拟存储。它具有内存分配、内存保护、地址映射和内存扩充（借助与虚拟存储技术）等功能。

进程管理

文件管理：存储空间的管理-目录管理-文件的读写管理和权限控制

设备管理

提供用户接口：命令接口，图形用户接口，程序接口

操作系统体系结构：

简单的监控程序模型—单体结构模型—层次结构模型—客户服务器模型与微内核结构—动态可扩展结构模型

单体内核是操作系统中最早、最常见的体系结构（UNIX/MS-DOS/Linux/MAC OS X/BSD）

层次结构最经典的例子Dijjkstra的THE系统

指令的执行：程序是指令的集合，程序的执行就是按照某种控制流执行指令的过程。一个单一指令需要的处理称为指令周期，包括取指周期和执行周期

第二章：进程管理

程序的顺序执行特点：顺序性，封闭性、可再现性

程序的并发执行特点：间断性、失去封闭性、不可再现性

进程的概念：

进程是允许并发的程序在某个数据集合上的运行过程

进程是正文段、用户数据段和进程控制块共同组成的执行环境。正文段存放被执行的机器指令，用户数据段存放进程在执行时要操作的用户数据，进程控制块存放程序的执行环境，操作系统通过这些描述和管理进程。

进程代表了程序的执行过程，是一个动态的实体，它随着指令的执行而不断变化，在某个特定时刻的进程内容被称为进程映像。

进程的特征：并发性、独立性、异步性、动态性、结构特征。

* 进程和程序的区别：

程序是静态的，进程是动态的

程序是永久的，进程是暂时存在的

程序和进程存在的实体不同。程序是指令的集合，进程是由正文段、用户数据段、进程控制块组成

进程和程序的联系：

进程是程序的一次执行，进程总是对应至少一个特定的程序，执行程序的代码，一个程序可以对应多个进程。

进程控制块：

进程实体存在的标志是操作系统管理进程所使用的数据结构—进程控制块

进程控制块是进程实体的一部分，是操作系统中最重要的数据结构，进程控制块中记录了操作系统所需要的，用户描述进程情况以及控制进程运行所需要的全部信息，进程控制块是操作系统感知进程存在的唯一标志。

进程控制块中的信息：进程标识符信息、处理机状态信息、进程调度信息、进程控制信息

进程的状态：就绪态、执行态，阻塞态

转换：

进程的组织：链接方式、索引方式、进程队列

进程的控制：进程的创建----阻塞----唤醒----终止

创建的条件：1）用户登录2）作业调度3）提供服务4）应用请求

阻塞的条件：1）请求系统服务2）数据尚未到达3）无工作可做4）启动某种操作

* 操作系统内核

操作系统内核是计算机硬件的第一次扩充，内核执行操作系统与硬件密切相关，执行频率高的模块，常驻内存。

操作系统内核的功能：1）支撑功能2）资源管理功能

支撑功能包括：中断处理、时钟管理和原语操作，原语操作是一组在执行过程中不能中断的操作

资源管理功能包括：进程管理、存储器管理和设备管理

中断：中断是改变计算机执行指令顺序的一种事件，这种事件与CPU芯片内外部硬件电路产生的电信号相对应。

中断的目的：能有效提高CPU的利用率，改善系统性能，支持系统的异步性。引用中断机制前，采用的是反复轮询的方式，来检测本次I/O是否结束。

中断类型1）同步中断（内部中断或异常）2）异步中断（外部中断）

同步中断是当指令执行时由CPU控制单元产生的，如除法出错，调试、溢出、浮点出错等

异步中断是由其他硬件设备随机产生的，可分为外部可屏蔽中断（I/O设备产生）和外部不可屏蔽中断（紧急事件产生，硬件故障等）

引起中断的原因：1）人为设置中断2）程序性事故3）I/O设备4）硬件故障5）外部事件

单重中断的处理过程：CPU在反复执行指令的过程中，每执行完一条执行，都会检查是否有外部中断的到来，如果有中断信号，则转中断处理。

* 时钟管理：

计算机的很多活动都是由定时测量来控制的，两种定时测量：1）保存当前的系统时间和日期2）维持定时器，操作系统依靠时钟硬件和时钟驱动程序来完成上述两种测量

时钟硬件（可编程间隔定时器）的功能：按照指定的时间间隔产生时钟中断，测量逝去的时间，并触发与时间有关的操作

时钟软件（时钟驱动程序）功能：1）维护日期和时间2）递减当前进程在一个时间片内的剩余执行时间，并检查是否为0，防止进程运行超时3）对CPU的使用情况记账 4）递减报警计数器

操作系统内核可以利用时钟机制防止一个进程垄断CPU或者其他资源

两个时钟源：实时时钟（RTC/CMOS）和OS时钟.

* 系统调用：系统调用是一群事先定义好的模块，他们提供一条管道让应用程序或用户能由此得到核心程序的服务。

系统调用是系统程序与用户程序之间的接口

系统调用与一般函数调用的区别：

1. 系统调用运行在系统态，一般函数运行在用户态
2. 系统调用与一般函数的执行过程不同，系统调用中断时，由系统找相应的系统调用子程序
3. 系统调用要进行『中断处理』，比一般函数多了一些系统开销

* 进程同步：

操作系统同步机制的主要任务就是保证在多任务共享系统资源的情况下，程序执行能得到正确的结果。同时，同步机制需要解决进程执行的协调问题。

进程同步的概念：在多任务系统中，进程一般存在资源共享关系和相互合作的关系。进程同步有两个任务：1）对具有共享资源关系的进程，保证以互斥的方式访问临界资源。临界资源是必须以互斥方式访问的共享资源。2）对具有相互合作关系的进程，要保证相互合作的诸进程协调执行。

同步机制应遵循的准则：1）空闲让进2）忙则等待3）有限等待4）让权等待

* 信号量机制（wait signal）对不同的共享资源设置称为信号量的变量，用信号量的取值标识资源的使用状况，或某种事件的发生。

1. 整型信号量机制：

用整型变量值来标记资源的使用情况。若整型量>0，说明有可用资源；若整型量<=0，说明资源忙，进程必须等待。对于一次只允许一个进程访问的临界资源，可定义一个用户互斥的整型信号量，并将其初始化为1，整型信号量的值只能通过两个特定的原子操作wait和signal来改变。

Var s integer;

Wait(s){ //申请资源

While s<=0 do no-op;

S=s-1; //占用资源

}

signal(s){ //释放资源

s=s+1;

}

整型信号量的互斥：初始变量为1

整型信号量的协调：初始变量为0

总结：1）整型信号量的值只能由wait和signal操作改变

1. wait和signal的操作都是原子操作，即在这两个操作中对信号量的访问是不能被中断的
2. 原子操作可以通过关中断来实现
3. 整型信号量机制的实例：linux中的自旋锁SpinLock
4. 不同的资源对应不同的信号量，并不是系统中所有资源都用同一个信号量标识
5. 记录型信号量机制：

代码：

Type semaphore = record

Value : integer; //资源数量

L : list of process； //阻塞队列

Procedure wait(s)

Var s : semaphore;

Begin

s.value = s.value-1; //申请资源

if s.value <0 then block(s.L) //此时资源无，自我阻塞进入阻塞队列

end

procedure signal(s)

var s:semaphore;

begin

s.value=s.value +1; //释放一个资源

if s.value <=0 then wakeup(s.L); //释放后发现还有阻塞，则唤醒阻塞中的进程

end

记录型信号量的优点是不存在「忙等」，采取了「让权等待」的策略

1. AND型信号量的机制

基本思想是将进程在整个运行过程中所需要的所有资源一次性的全部分配给进程，待进程使用完之后再一起释放。只要还有一个资源不能分配给该进程，其他所有可能为之分配的资源也不分配给它。

管程：管程是描述共享资源的数据结构和在数据结构上的共享资源的管理程序的集合

进程通信：进程之间的高级通信机制分为：共享存储器系统、消息传递系统、管道通信系统。

线程：在操作系统中，进程是进行资源分配和独立执行的基本单位，为了进一步提高程序的并发性，减少系统开销，在操作系统中引入了线程的概念。

线程是进程中的一个实体，是被系统独立调度和分派的基本单位。线程在运行中存在间断性，也有就绪、执行、阻塞三种形态。

第三章：进程调度与死锁

进程调度的功能是按照某种策略或算法从就绪态进程中为当前空闲的cpu选择在其上运行的新进程。

选择调度方式和算法的若干准则：

1. 周转时间短 周转时间是指从作业被提交给系统开始，到作业完成为止

系统的平均周转时间T等于N各作业的周转时间之和除以n

T=(t1+t2+t3+…+tn)/n

作业的周转时间T与系统为它提供的服务时间TS之比为W，W=T/TS，被称为带权周转时间，那么n个作业的平均带权周转时间为：

T=(t1/ts1+t2/ts2+…+tn/tsn)/n

服务时间Ts是一个作业在CPU上执行的总时间

2) 响应时间快 响应时间是指从用户提交一个请求开始直至系统首次产生响应的时间为止的一段时间

3）截止时间的保证 截止时间是指某个任务必须开始执行的最迟时间，或必须完成的最迟时间

4） 系统吞吐量高

5）处理机利用率好

* 调度算法：

1. 先来先服务（FCFS）从就绪列的队首选择最先到达就绪队列的进程，FCFS适合长进程，不利于短进程，适合CPU繁忙性进程，不适合IO繁忙性进程。
2. 短进程优先调度算法（SPF）短进程优先算法能有效降低进程的平均等待时间，提高系统的吞吐量
3. 优先调度算法（PSL） 类型：非抢占式优先权调度算法、抢占式优先权调度算法；优先权的类型：静态优先权和动态优先权。抢占静态最容易饥饿
4. 时间片轮转调度算法（RR）

时间片大小的确定考虑的因素：

系统对响应时间的要求，响应时间越短，时间片取值应该越小。

就绪队列中进程的数目

系统的处理能力

5） 多级队列调度 不同的队列优先权不同，调度算法也可能不同。

6） 多级反馈队列调度 队列优先权越高，时间片越短，时间片通常成倍增长

实时系统中的调度：

基本条件：1）提供必要的调度信息2）系统处理能力强3）采用抢占式调度机制4）具有快速切换机制

常用的调度算法：1）最早截至时间优先（EDF） 2）最低松弛度优先（LLF）

多处理器调度：

多处理器系统的类型：紧密耦合、松弛耦合、对称处理器系统、非对称处理器系统

进程调度方式：1）自调度2）成组调度3）专用处理器分配

自调度：采用自调度的系统中有一个公共的就绪队列，任何一个空闲的处理器都可以从该就绪队列中选择一个进程或者一个线程运行。在多处理器环境下，FCFS是较好的自调度算法

自调度优点：1）易移植 2）有利于提高CPU的利用率

缺点：1）瓶颈问题 2）低效性 3）程序切换频繁

* 死锁：死锁是由多个进程竞争共享资源而引起的进程不能向前推进的僵死状态

产生死锁的原因：竞争死锁资源且分配资源的顺序不当

产生死锁的必要条件：1）互斥2）请求保持3）不剥夺4）环路等待

处理死锁的方法：预防死锁、避免死锁、检测并解除死锁和忽略死锁

死锁的避免：资源分配的状态分为安全状态和不安全状态，不安全状态不一定产生死锁，但是系统进入安全状态以后，就可以避免死锁的产生，所以避免死锁的实质在于使系统处于安全状态。

银行家算法：

基本思想：一个进程提出资源请求后，系统进行资源的试分配。然后检测此次分配是否处于安全状态，若安全则按分配方案分配资源，否则不分配资源。

试分配过程：

available[] 可用数量

max[] 最大数量

allocation[] 已分配的资源数

need[] 还需要某资源的数量

1. 先进行试分配
2. request i <= need i
3. request i <= available i

满足上述条件进行试分配

1. available = available –request i

allocation = allocation + request i

need i = need i – request i

然后安全检测

wrok[] = available

finish[] = false

当need I <= work时，work = work +allocation I finish [] =true

若对于所有的finish[] =true都成里，则说明处于安全状态

第四章：内存管理

内存管理的目标：1）实现内存分配、内存回收等操作 2）提高内存利用率和内存的访问速度（即充分利用现有的内存资源，为应用程序提供方便的内存使用方式和一个快速、安全且充分大的存储器）

程序的链接和装入：

链接要解决的问题是将编译后的目标模块装配成一个可执行程序，分为静态链接和动态链接。

1. 静态链接：在程序运行前，用链接程序将目标模块链接成一个完整的装入模块，任务：一时对逻辑地址进行修改，二是变换外部调用符号

优点：运行速度较快 缺点：可执行文件较大，占用空间大，系统开销大，程序开发不够灵活，修改一个模块会导致整个程序重新链接

1. 动态链接：可将某些目标模块的链接推迟到这些模块中的函数要被调用时再进行。优点：节省内存和外存空间，方便程序开发。缺点：增加了运行的时间开销，使程序运行时的速度变慢。

程序装入：

装入方式：绝对装入方式、可重定位装入（静态装入方式）和动态运行时装入方式

绝对装入方式：编译程序已知程序在内存中的位置，编译时产生物理地址的目标代码，装入程序按照装入模块的物理地址将程序和数据装入内存

可重定位装入方式：编译时不知道程序在内存中的位置，那么编译时就必须生成可重定位的代码，其中的地址都是逻辑地址，在程序装入内存时，再把逻辑地址映射为物理地址。程序装入时对目标程序中的指令和数据地址修改的过程称为重定位。

静态装入方式的特点：1）编译程序使目标模块的地址从0开始 2）程序装入时，装入程序根据内存的使用情况将装入模块装入到内存的某个位置，并对模块进行重定位。物理地址=有效逻辑地址+程序在内存中的起始地址

动态运行时装入：当一个进程在被换出之前的内存地址与后来被从外存调入时所在的内存位置不同，这时，地址映射延迟到进程执行时再进行

* 连续分配的存储管理方式：

类型：1）单一连续区分配方式2）固定分区分配方式 3）动态分区分配方式

单一连续区分配方式：适用于单用户单任务系统，内存分为系统区和用户区

固定分区分配方式：将用户内存空间分配成若干固定大小的区域，每一个区域运行一道用户程序；分区的数量是固定的，大小也是固定的

每个分区大小相等的分配方式缺点是内存利用率比较低，主要用于一个计算机去控制多个相同对象的场合，如冶炼炉

分区大小不等：可以更好的利用内存

分区结构：分区编号，分区大小，分区起始地址，分区状态

在一些实时系统中，固定分区的分配方式还是简单而有效的。

动态分区分配方式：用户分区的数量和大小都是动态变化的

分配原理：系统初始只有一个大的空闲分区，当进程请求内存资源时，系统根据请求资源的大小分配一片空闲区域给进程，当运行一段时间后，空闲分区可能会散布在不连续的区域，这时系统会维护一个记录当前空闲分区情况的数据结构，当进程请求内存时，系统从所有空闲分区中找一个合适大小的空间给进程。

数据结构：空闲分区表和空闲分区链

空闲分区链可以动态的为每个分区建立一个节点，每个节点包括分区大小、分区起始地址、指向前一个空闲分区节点的指针、指向后一个空闲分区节点的指针。每个节点占用的内存可以动态分配、动态回收。

动态分区分配算法：

1. 首次适应算法FF

要求空闲分区链以地址递增的顺序进行链接，每次从链首开始查找，低地址空间可能会被反复划分 缺点：造成空间浪费，内存碎片

1. 循环首次适应算法NF

不再每次从链首开始查找，而是从上一次查找的空闲分区的下一个空闲分区开始查找，每次应设置一个起始查找指针，指示下一次查找的分区

优点：空闲区分布均匀，查找开销小， 缺点：缺少空闲大的分区

1. 最佳适应算法BF

为了方便查找，把所有空闲区按照空闲分区的大小递增的顺序进行排列，总是把大小和进程所请求的内存空间大小最接近的空闲分区分配给进程。

优点：避免了大材小用，提高了内存的利用率 缺点：容易留下难以利用的小空闲区

* 基本分页存储管理方式：

把进程离散的存储在内存中物理地址不连续的区域，这种内存管理方式称为离散内存管理方式。离散内存管理分配内存空间的管理方式：分页存储管理，分段存储管理、段页式存储管理

基本概念：

页：将一个进程的逻辑地址空间分成若干个大小相等的片，称为页。

页框：将物理内存地址分成与页大小相同的若干个存储块，称为页框或页帧

分页存储：为进程分配内存时，以页框为单位将进程中的若干页分别装入多个可以不相邻的页框中。

页内碎片：进程的最后一页一般装不满一个页框，而形成了不可利用的碎片，称为「页内碎片」，是一种内部碎片

页表：实现页号到页框号的映射，在基本的分页机制中，每个进程有一个页表，进程的每一页在页表中有一个对应的页表项，页表在内存中连续存放。

分页存储管理方式的地址结构：

|  |  |
| --- | --- |
| 页号P | 页内偏移量W |

若用m位表示逻辑地址，页大小为2的n次方字节，则用低n位表示页内偏移量w，用高m-n位表示页号P。

公式：P=INT(A/L) W=MOD(A/L) A为逻辑地址 L是页大小

分页地址变换：实现逻辑地址到物理地址的转换

公式：物理地址=页框号x页框大小+页内偏移量

为了减少CPU在有效访问内存时间上的开销，提高访问内存的速度，引入了快表机制。

快表：也称转换后援缓冲（TLB）是为了提高访存速度而采用的专用缓存，存放最近被访问过的页表项。

* 两级页表和多级页表：

页表再分页，就形成了两级或多级页表。

两级页表：将页表再分页，使得每个页表分页的大小与内存页框的大小相同，并为它们编号。

逻辑地址结构：

|  |  |  |
| --- | --- | --- |
| 页目录号p1 | 页号p2 | 页内偏移地址d |

页目录号实际是一个索引值，，根据p1从页目录表项中找到页表所在的页框号，页号P2是页表中的偏移量，根据p2可以知道应该从页表中的第p2项找到进程页所在的页框号。

公式：进程A的物理地址=进程页所在的页框号x页框大小 + 页内偏移地址d

* 基于分页的虚拟存储系统：

虚拟存储技术实现的基本思想是：只把进程的一部分装入内存，在进程执行的过程中，CPU访问内存时如果发现所访问的内容不在内存中，则通过异常处理将所需要的内容从外存调入内存。

虚拟存储技术的好处：1）提高内存利用率 2）提高多道程序度 3）把逻辑地址空间和物理地址空间分开，程序员不用关心物理内存的容量对编程的限制。

虚拟存储技术的特征：1）离散性2）多次性 3）对换性 4）虚拟性

页表：页表是请求分页系统最重要的数据结构，其作用是描述记录页的各种数据结构，包括在实现逻辑地址到物理地址映射时需要的页号和页框号的对应关系。同时增加了请求换入和页置换时需要的数据。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 页框号 | 状态位P | 访问字段A | 修改位M | 保护位 |

状态位：表示页是否在内存中 访问字段：记录页最近是否被访问过

修改位：表示页最近是否被修改过 保护位：访问权限，1可读可写，0只读

缺页异常机构：主要作用是在访问内存过程中发现缺页产生缺页异常信号，使CPU中断当前控制流的执行，转去执行缺页异常处理程序，完成请求调页。

* 页分配策略：问题，最少页框数？如何淘汰页？分配算法？

最少页框数：是指能保证进程正常运行所需要的最少页框数。操作系统为进程分配的页应该大于或者等于最少页框数。

页分配和置换策略：

页分配策略：固定分配策略和可变分配策略

页置换策略：局部置换和全局置换。1）局部置换是指发生置换时，只从请求置换的进程本身的内存页中选择一个淘汰页，腾出内存空间，调入请求页。2）全局置换是指发生置换时，从所有进程的内存页中选择被淘汰的页。

两种组合：1）固定分配局部置换 2）可变分配局部置换 3）可变分配全局置换

分配算法：

1. 平均分配算法 n进程m页框，则分配INT[m/n]个页框，余数放入缓冲
2. 按比例分配算法 为进程分配的页框数=进程页数/所有进程页数的总和x页框数
3. 考虑优先权的分配算法

* 页调入策略：

1）系统可以在进程需要是将页调入内存 有利于提高内存的利用率，但是对系统的时间性能不利

2）采用预先调入页的策略 将预计不久之后会被访问的也预先调入内存

* 页置换算法：

1. 最佳置换算法ORA：该算法选择以后永远不会被访问的页或者很长时间不会被访问的页作为换出页（看后面谁最长时间不会被访问到就换出）
2. 先进先出置换算法FIFO：最简单。该算法是为每个页记录该页调入内存的时间，当选择换出页时，选择进入内存时间最早的页（用指针指示当前调入新页时，应淘汰的那页在队列中的位置，换出后，指针指向下一个应淘汰的页）
3. 最近最久未使用的LRU置换算法：性能较好的算法。该算法是选择最近最久未使用的页换出（看前面谁进来的时间最久，最长时间没被访问过）

其他置换算法附件引用位算法简单clock算法改进型clock算法最少使用置换算法页缓冲算法

请求调入和置换技术都是以时间换空间的技术

缺页率对有效访存时间的影响：

有效访问时间是成访存所用的时间。假设P为缺页率，ma为存储器访问时间，根据实际性能取ma=100ms=0.1us

有效访存时间=（1-P）x ma + P x 缺页异常时间

引入工作集机制是为了能有效降低缺页率，从而提高访存的时间效率

抖动：由于多道程序度太高，运行进程的大部分时间都用于进行页的换入、换出，而几乎不能完成任何有效工作的状态称为抖动。

抖动的预防：1）采取局部置换策略2）在CPU调度程序中引入工作集算法 3）挂起若干进程

* 分段存储管理

引入分段机制的优点是方便编程、分段共享、分段保护、动态链接以及存储空间的动态增长

分段：系统为每个进程建立一个段表，段表的每一个表项记录了的信息包括段号、段长和该段的基址，段表存放在内存中。

段表：段表是操作系统维护的用于支持分段存储管理地址映射的数据结构

|  |  |  |
| --- | --- | --- |
| 段号 | 段长 | 段基址 |

段基址就是段在物理内存中的起始地址，段大小也称段界限。

* 分页和分段的主要区别：

联系：分段和分页都属于离散分配方式，都需要通过数据结构和硬件的配合实现逻辑地址到物理地址的映射。

区别：页是按物理单位划分的，分页的引入是为了提高内存的利用率和支持虚拟存储；分段是按逻辑单位划分的，引入分段是为了方便程序员编程。

页的大小是固定的，段的大小是不固定的

分页的地址是一维的，分段的地址是二维的

第五章：文件系统

文件结构：1）无结构字节序列 2）固定长度记录序列 3）树形结构

文件的类型：正规文件、目录文件、字符设备文件、块设备文件

正规文件包含用户信息，一般分为ASCII文件和二进制文件。

文件存取：顺序存取和随机存取，随机存取又称为直接存取

文件的操作：CREATE/DELETE/OPEN/CLOSE/READ/WRITE/APPEND/SEEK/getattributes/setattributes/rename

目录：目录是文件系统中实现按名访问文件的重要数据结构。

目录文件的结构：属性放在目录项中、放在i节点中

目录结构：单层目录---两级目录---树形目录

优缺点比较：

单层目录：也称为根目录。优点：结构简单 缺点：搜索效率低，不适合多用户系统

两级目录：优点：解决了文件重名问题和共享问题，查找时间降低 缺点：增加了系统的存储开销。

树形目录：即多级目录 优点：便于文件分类，层次结构清晰，便于管理和保护，解决了重名问题，查找速度加快 缺点：查找一个文件需要多次访问磁盘影响速度，结构相对复杂。

路径名：绝对路径名、相对路径名。只要路径的第一个字符是分隔符，就是绝对路径。

目录操作：create/delete/opendir/closedir/readdir/rename

文件系统的实现：

文件系统通常是以2的n次方个连续的扇区为单位对文件进行磁盘空间的分配，把分配给文件的连续扇区构成的磁盘块称为簇

分配方式：连续分配：就是把每个文件作为一连串连续数据块存储在磁盘上

优点：实现简单，读性能好 缺点：随着时间的推移，磁盘会变得零碎

使用磁盘链接表的分配：该方法为每个文件构造簇的链接表，每个簇开始的几个字节存放下一个簇的簇号，其他地址存放数据，每个文件可以存放在不连续的簇内。 优点：可以充分利用每个簇，不会因为磁盘碎片浪费存储空间，管理也比较简单 缺点：随机存取相当缓慢

使用内存的链接表分配：将文件所在的磁盘的簇号存放在内存的表中。

i结点：该方法为每个文件赋予一个被称为i结点的数据结构，其中列出了文件属性和文件块的磁盘地址

磁盘空间管理：

记录空闲块方式：空闲簇链接表位图

公式：

1. 块号=字号x字长+位号
2. 柱面号=块号/柱面上的块数
3. 磁头号=（块号mod柱面上的块数）/ 磁道上的扇区数
4. 扇区号=（（块号mod柱面上的块数）mod磁道上的扇区数

第六章：I/O设备管理

I/O系统的组成：IO设备，与设备相连的控制器，通道（用于大型系统中专门用于I/O的专用计算机）

I/O系统的结构：分为微机I/O系统、主机I/O系统

I/O设备的分类：

按照传输速率分为：1）低速设备，如键盘鼠标2）中速设备，如打印机3）高速设备，如磁带机，磁盘机，光盘机

按照信息交换的单位分类：1）块设备，如磁盘2）字符设备，如终端、打印机、通信端口、鼠标

按照设备的共享属性分为：1）独占设备，如打印机 2）共享设备，如硬磁盘3）虚拟设备，通过虚拟技术把一台物理设备变成若干逻辑设备

* 设备控制器：

设备控制器是CPU与I/O设备的接口，接收I/O命令并控制设备完成I/O工作

设备控制器是一个可编址设备，链接多个设备时可有多个设备地址。

设备控制器的功能：1）接收和识别命令 2）数据交换 3）设备状态的了解和报告 4）地址识别 5）数据缓冲 6）差错控制

设备控制器的组成：逻辑结构由3部分组成

1）设备控制器与处理机的接口：数据线、控制线和地址线

2）设备控制器与设备的接口：接口中的3类信号为数据、状态和控制信号

3）I/O逻辑：主要由指令译码器和地址译码器两部分功能部件组成。

* I/O通道：通道用于大型主机系统控制I/O设备，与控制设备结合，与微机和小型机的设备控制器有对等的功能。即用来替代微机、小型机的设备控制器，实现大型主机系统的I/O设备控制功能，提供操作系统与I/O设备间的接口。
* I/O控制方式：1）早期轮询控制方式2）中断控制方式3）DMA控制方式

轮询：这种控制方式使CPU经常处于由于输入/输出而造成的循环测试状态，造成CPU极大的浪费，影响整个系统的吞吐量。

中断：中断控制方式能使CPU与I/O设备在某些时段上并行工作，提高CPU利用率和系统的吞吐量。

DMA：为了进一步提高CPU与I/O的并行程度，引入了DMA控制方式。DMA控制需要特殊结构的设备控制器，DMA的控制器逻辑结构组成：主机与DMA的接口，DMA与设备的接口，以及I/O控制逻辑。

为了实现主机与设备控制器之间数据的传送，在DMA控制器中设计了4类寄存器：命令/状态寄存器CR、内存地址寄存器MAR、数据寄存器DR、数据计数器DC

* 缓冲管理：

缓冲区是用来保存两个设备之间或者设备与应用程序之间传输数据的内存区域

缓冲的引入：在数据到达速率与数据离去速率不同的地方，都可以引入缓冲区

引入缓冲区的原因：1）处理数据流的生产者与消费者之间的速度差异

2）协调传输数据大小不一致的设备。

引入缓冲区除了可以缓和CPU与I/O设备之间速度不匹配的矛盾，还能降低对CPU中断频率的要求，放宽对中断响应时间的限制，提高CPU与I/O设备的并行性。

单缓冲---双缓冲---循环缓冲---缓冲池

缓冲区可以工作在收容输入、提取输入、收容输出和提取输出四种工作方式下

* 设备分配：

设备的固有属性：可分为独占设备、共享设备、可虚拟设备

设备分配的算法：1）先来先服务 2）基于优先权的分配算法

设备的分配方式：

1）安全分配方式：这种分配方式摒弃了造成死锁的条件之一「请求和保持」条件，从而使设备的分配是安全的。缺点：

2）不安全分配：优点：一个进程可同时操作多个设备，使进程推进迅速。缺点：不安全，可能造成死锁。

设备独立性：其含义是应用程序独立于具体使用的物理设备。

实现设备独立性带来的好处：

应用程序与物理设备无关，系统变更外围设备时不需要修改应用程序

易于处理输入输出设备的故障

提高了系统的可靠性，增加了设备的灵活性

独占设备的分配程序：分配设备---分配控制器---分配通道

* SPOOLing技术

原理：在多道程序环境下，利用一道程序来模拟脱机输入时的外围控制机功能，把低速I/O设备上的数据传送到高速输出磁盘上，再用另外一道程序来模拟脱机输出时的外围控制机的功能，把高速磁盘上的数据传送到低速输出设备上。这种在联机情况下实现同时外围操作称为SPOOLing技术。

SPOOLing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SPi和输出进程SPo 4）请求I/O队列

SPOOLing技术的特点：1）提高了I/O速度 2）将独占设备改造为共享设备 3）实现了虚拟设备的功能

* I/O软件原理

I/O软件的总体目标是将软件组织成一种层次结构，低层软件用来屏蔽硬件的具体细节，高层软件则主要是为用户提供一个简洁、规范的界面。

应用程序与设备管理软件的构成：

1）用户层软件 2）与设备无关的软件层 3）设备驱动程序 4）中断处理程序（底层） 其中设备驱动程序包括设备服务程序和中断处理程序

设备管理软件的功能：1）实现IO设备的独立性 2）错误处理 3）异步传输 4）缓冲管理 5）设备的分配和释放 6）实现IO控制方式

设备驱动程序：设备驱动进程是I/O进程与设备控制器之间的通信程序，其主要任务是接收上层软件发出的抽象的I/O请求，把它们转换为具体要求之后，发送给设备控制器，启动设备去执行。

* 磁盘管理：

磁盘管理的重要目标是提高磁盘磁盘空间利用率和磁盘访问速度。

磁盘设备可包括一个或多个物理盘片，每个盘面被组织成若干个同心环，这种环称为磁道，每条磁道被划分为若干个扇区，一个物理记录存储在一个扇区上，磁盘上存储的物理记录数目是由扇区数、磁道数、及磁盘面数所决定的

磁盘的类型：硬盘和软盘、单片盘和多片盘、固定头磁盘和移动头磁盘

固定头磁盘：可进行并行读写，这种磁盘主要用于大容量的磁盘上

移动头磁盘：只能串行读写，广泛应用于中小型磁盘设备上

磁盘的访问时间：

寻道时间：磁臂移动到指定磁道上所经历的时间

旋转延迟时间：将指定扇区移动到磁头下面所经历的时间

传输时间：把数据从磁盘读出或向磁盘写入数据经历的时间

磁盘访问时间中，寻道时间和旋转延迟时间基本上都与所读/所写数据的多少无关，而且寻道时间和旋转延迟时间通常占据了访问时间中的大头

* 磁盘调度：磁盘调度的一个重要目标是使磁盘的平均寻道时间最少

1. 先来先服务：根据进程请求访问磁盘的先后顺序进行调度 优点：公平简单 缺点：平均查寻道时间可能过长
2. 最短寻道时间优先：与当前磁头所在的磁道距离最近优先
3. 扫描（SCAN）算法：

进程「饥饿」现象，scan算法又称为点滴调度算法

1. 循环扫描（CSCAN）算法：规定磁头单向移动
2. NStepSCAN和FSCAN调度算法

提高磁盘I/O速度的方法：

1）提前读 2）延迟写 3）优化物理块的分布 4）虚拟盘 5）磁盘高速缓存