**第1章 绪论 习题**

1. 指令在计算机系统中是如何执行的？请简述执行过程。

计算机在运行时，先从内存中取出第一条指令，通过控制器的译码，按指令的要求，从存储器中取出数据进行指定的运算和逻辑操作等加工，然后再按地址把结果送到内存中去。接下来，再取出第二条指令，在控制器的指挥下完成规定操作，依此进行下去。直至遇到停止指令，程序与数据一样存贮，按程序编排的顺序，一步一步地取出指令，自动地完成指令规定的操作是计算机最基本的工作原理。这一原理最初是由美籍匈牙利数学家冯.诺依曼于1945年提出来的，故称为冯.诺依曼原理。

1. 操作系统在计算机系统中的地位是什么？

操作系统是计算机系统中硬件层的首次扩充，是用户和计算机系统之间的接口，是其他软件和硬件之间的接口，是最基本，最重要的系统软件，其他的用户和软件都要在操作系统的支持下才能使用计算机系统进行各种事务的处理.

1. 操作系统为用户提供了哪些接口，它们是如何工作的？

操作系统为用户提供两个接口：命令接口和系统调用。

命令接口：系统为用户提供的。利用操作命令来组织和控制作业的执行或管理计算机系统。

系统调用：编程人员使用系统调用。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。

1. 多道程序系统的主要特点是什么？

①多道--内存中同时存放两道或两道以上的程序,它们共享CPU和系统中的其他资源。

②宏观上并行—在一个时间段内,多个作业都在同时运行。

③微观上串行—在某一个时刻,只有一道作业真正在CPU上运行,即:各作业都在管理程序的控制下在一台计算机上交替地执行。

1. 什么是计算机操作系统？它的主要功能包括哪些？

操作系统是一组主管并控制计算机操作、运用和运行硬件、软件资源和提供公共服务来组织用户交互的相互关联的系统软件程序。

操作系统的功能：

处理器管理，主要控制和管理CPU的工作

存储管理，主要进行内存的分配和管理

设备管理，主要管理基本的输入输出设备

文件管理，负责对计算机文件的组织、存储、操作和保护等

进程管理（作业管理），对计算机所进行的操作进行管理

1. 嵌入式操作系统的主要特点是什么？

[嵌入式操作系统通过虚拟化技术实现虚拟机管理抽象层,对硬件资源抽象以支撑“软件定义机器”,达到针对业务的实时性,确定性等支持多样应用场景的特点.](http://www.baidu.com/baidu.php?url=0s0000aPch_Smgbn9R6v86ZHpqXGuM_AxTLphcNTn_th6f7QJWEoFFeTRKGPUhqWORJsx_yPNpA5fR-gQjURSmCzdH3HrW5OzVQwrrO26R8RgF1QRZXZH8tzo8SSbaAWnLZwDGDk5NHwwNyWsh6IiAu6TYTbgVF88O_1fOmqLH1eEjxOOUmBDQBQ5os0sISu58e7p-P-oTVtIYafwMQAW9q9DYCr.DR_i5BCF3DYpQCYfUuX9mhrMFuELNvIMQvTXNzI5QIJyAp7WGe5ZGLJ0.U1Yk0ZDqkeUcltJq_VMgqtSd1U5HlteEspoPSQQudOx5VTv1VTHZSQZs0ZKGm1Yk0ZK1pyI85iY0IjLZV_WozovzvIxL1egP_sKGUHYznWR0u1dEugK1n0KdpHdBmy-bIykV0ZKGujY1nsKWpyfqPjnd0AdY5HDsnH-xnH0kPdtznjRkg1bsn1Kxn1msnfKopHYs0ZFY5HfvrfKBpHYkPH9xnW0Yg1Rsndt4njc0UynqnH0krNtknjDLg1csPH7xn1DdPjb4PWcLnWKxn0KkTA-b5H00TyPGujYs0ZFMIA7M5H00mycqn7ts0ANzu1Ys0ZKs5Hn4njc3nHfvrfK8IM0qna3snj0snj0sn0KVIZ0qn0KbuAqs5H00ThCqn0KbugmqTAn0uMfqn0KspjYs0Aq15H00mMTqnH00UMfqn0K1XWY0mgPxpywW5gK1QyIlpZ940A-bm1dcHbc0TA9YXHY0IA7zuvNY5Hnkg1nknNtzP0KYIgnqnHDkrjbzPj04nHc3PjcdPjfdnWn0ThNkIjYkPWTkPHDzP1bznj0v0ZPGujdWPjF9uAP9rj0snjDknAnz0AP1UHY1PH77rjmvnDRsrR7af17j0A7W5HD0TA3qn0KkUgfqn0KkUgnqn0KlIjYs0AdWgvuzUvYqn7tsg1Kxn7ts0Aw9UMNBuNqsUA78pyw15HKxn7tsg1Kxn0Ksmgwxuhk9u1Ys0AwWpyfqn0K-IA-b5HmY0A71TAPW5H00IgKGUhPW5H00Tydh5H00uhPdIjYs0A-1mvsqn0K9uAu_myTqnfK_uhnqn0KbmvPb5Rm3f1DdwDndfYR3fWfswjTLnDPKnRDsfYujwH9jfWKarbuZ5HD0IZF9uARqn0KBuA-b5H7jrj6dPHuDPj63wWRsfH0Ln1-awWn4nWTkPRuaPWnk0AqW5HD0mMfqn0KEmgwL5H00ULfqn0KETMKY5H0WnanWnansc10Wna3snj0snj0WnaPDw-fWnanVc108nj0snj0sc1D8nj0snH0sc10WnansQW0snj0sn0KBmy4omyPW5H0Wn0K3TLwd5Hf1n161rjc40Z7xIWYsQWnYg108njKxna3sn7tsQWckg108njKxn7tsQW0sg100mMPxTZFEuA-b5H00ThqGuhk9u1Ys0APv5fKGTdqWTADqn0KWTjYs0AN1IjYs0APzm1YvPj6vn0&us=newvui&xst=TjY1rH0zrjDYPWbKm1dArDnkPRwjPRP7rDcYnDfLP1KjfH7KnDPAfYR3fYcsfWGAw1Yk0ycqnRn3rjRdPbfYrj9APHKKnjT1rRFAn1bzP1Ddwbcvn1DKT1YkPjD1PHndnWb4PjRvrjfsrHc3n-tznWNxn07L5TXvzrE2EUQudOx5VTv10gDqkeUcltJq_VMgqtSd1U5HlteEspoPSQQudOx5VTv1VTHZSQZs0gRqPjn1rjn3nWbKIjYkPWTkPHDzP1bz0ydk5H0an0cV0yPC5yuWgLKW0ykd5H0Kmv3qPWcsrHRknjwxPjPxUvNVgvwM0Hn4PjRYnjfsPWc&cegduid=Pjn1rjn3nWb&solutionId=20513043&word=&ck=3506.15.206.465.177.512.227.1249&shh=www.baidu.com&sht=62095104_43_oem_dg&wd=&bc=110101" \t "_blank)

1. 实时操作系统的主要特点是什么？

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。虽然，多道批处理系统和分时系统已能获得较为令人满意的资源利用率和响应时间，从而使计算机的应用范围日益扩大，但它们仍然不能满足以下某些应用领域的需要。

实时控制系统，当把计算机用于生产过程的控制，以形成以计算机为中心的控制系统时，系统要求能实时采集现场数据，并对所采集的数据进行及时处理，进而自动地控制相应的执行机构，使某些(个)参数(如温度、压力、方位等)能按预定的规律变化，以保证产品的质量和提高产量。

也可将计算机用于对武器的控制，如火炮的自动控制系统、飞机的自动驾驶系统，以及导弹的制导系统等；实时信息处理系统，该系统由一台或多台主机通过通信线路连接到成百上千个远程终端上，计算机接收从远程终端上发来的服务请求。

根据用户提出的请求对信息进行检索和处理，并在很短的时间内为用户做出正确的响应。如：网上订票，网上购物等。

1. 批处理系统有什么优势？在现代操作系统中，它还有用武之地吗？

批处理系统的主要优点是系统吞吐量大,资源利用率高,系统开销小。现代操作系统中，仍然有批处理任务的需求，可以在后台处理大量的计算任务。

1. 嵌入式操作系统与物联网操作系统的区别与联系，请谈谈你的理解。

嵌入式操作系统具有专门的功能，通常通过包括操作系统的定制硬件和软件与其他设备具有专有接口。

物联网系统在设计上是开放的，由标准构建块组成，并使用开放接口连接到其他设备。并非所有物联网系统的构建方式都相同，但核心技术是相似的。

10、Linux内核开发时需要注意什么问题？

内核模块内不能使用stdlib,stdio等C标准库，内核在lib/string.c下实现了一些常用函数strcpy.memset之类的函数，这些是可以在内核模块内使用的。

内核不支持浮点运算，说是提高性能避免用户空间，内核空间切换时需要保存FPU。

内核模块内调用函数，函数参数大小不能大于1Kb(1024bytes); 模块内定义的函数局部变量大小不能超过1Kb(1024bytes);否则会报警告frame size larger than 1024bytes。

内核模块申请空间标志问题，在中断程序中申请内存使用GFP\_ATOMIC标志，以避免休眠等待。

中断访问资源时，要用spin\_lock 或其他并发控制机制，保护资源以免冲突，造成系统紊乱。中断中使用并发控制要避免造成休眠。如：mutex 互斥锁会造成休眠不能在中断中使用。

11、Linux操作系统内核数据结构选择的原则是什么？

如果你对数据集合的主要操作是遍历数据，就使用链表。事实上没有数据结构可以提供比线性算法复杂度更好的算法遍历元素，所以你应该用最简单的数据结构完成简单工作。另外，当性能并非首要考虑因素时，或者当你需要存储相对较少的数据项时，或者当你需要和内核中其他使用链表的代码交互时，也该优先选择链表

如果你的代码符合生产者/消费者模式，则使用队列，特别是如果你想（或者可以）要一个定长缓冲。队列会使得添加和删除项的工作简单有效。同时队列也提供了先入先出（FIFO）语义，而这也正是生产者/消费者用例的普遍需求。另一方面，如果你需要存储一个大小不明的 (可能很多项）的数据集合，那么链表可能更合适，因为你可以动态添加任何数量的数据项

如果你需要映射一个UID到一个对象，就使用映射。映射结构使得映射工作简单有效，而且映射可以帮你维护和分配UID 。Linux的映射接口是针对UID到指针的映射，它并不适合其他场景。但是如果你在处理发给用户空间的描述符，就考虑一下映射吧

如果你需要存储大最数据，并且检索迅速，那么红黑树最好。红黑树可确保搜索时间复杂度是对数关系，同时也能保证按序遍历时间复杂度是线性关系。虽然它比其他数据结构复杂一些, 但其内存开销情况并不是太糟。但是如果你没有执行太多次时间紧迫的查找操作，则红黑树可能不是最好选择。这种情况最好使用链表

12、谈谈你对未来国产操作系统发展的看法与建议。

**第2章 操作系统结构 习题**

1. 请简要描述一下操作系统的设计目标是什么？

设计操作系统的主要目标：

(1)方便性（极大方便了用户，使计算机变得易学易用）

(2)有效性（一层含义：提高系统资源的利用率；二层含义：提高系统的吞吐量）

(3)可扩充性（OS广泛采用微内核技术，能够方便地添加新的功能模块）

(4)开放性（遵循国际标准，使得各种软硬件能够彼此兼容）

1. 举例说明机制与策略的关系。

Unix/Linux的接⼝设计有⼀句通⽤的格⾔“提供机制⽽不是策略”。

⼤部分的编程问题都可以被切割成两个部分：“需要提供什么功能（机制）”和“怎样实现这些功能（策略）”。Unix/Linux抽象出对这些对象的执⾏机制：

程序的执⾏机制 -- 进程；

内核函数的执⾏机制 -- 线程；

中断服务程序的执⾏机制 -- 中断信号触发；

中断下半部的执⾏机制 -- softirq, tasklet, ⼯作队列；

系统调⽤的执⾏机制 -- 软中断（int $0x80触发）

而策略就是实现以上机制的方法

1. **操作系统为用户提供的接口有哪几种？分别有什么特点？**
2. 解释系统调用的原理，参数传递采用什么形式？

[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F) [内核](https://baike.baidu.com/item/%E5%86%85%E6%A0%B8)提供一系列具备预定功能的[多内核](https://baike.baidu.com/item/%E5%A4%9A%E5%86%85%E6%A0%B8)[函数](https://baike.baidu.com/item/%E5%87%BD%E6%95%B0)，通过一组称为系统调用（system call)的接口呈现给用户。系统调用把[应用程序](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F)的请求传给[内核](https://baike.baidu.com/item/%E5%86%85%E6%A0%B8)，调用相应的内核[函数](https://baike.baidu.com/item/%E5%87%BD%E6%95%B0)完成所需的处理，将处理结果返回给应用程序。

向系统传递参数有三种常用方法：

1）直接通过寄存器来传递参数。

2）当参数数目比较多的时候，这些参数通常存在内存的某个区域中，内存地址通过寄存器来传递）。

3）参数也可通过程序压入堆栈，并通过操作系统弹出。

1. 系统调用函数与API函数有什么区别与联系？

两者都可以被开发人员使用来访问内核的底层功能，而程序员更喜欢根据 API 来编程。其主要原因有：

（1）程序的可移植性。程序员根据 API 设计程序，以希望程序能在任何支持同一 API 的系统上编译并执行；但是直接使用系统调用，无法做到直接移植。

（2）对程序员而言，实际系统调用比 API 更为注重细节且更加难用。

尽管如此，在 API 的函数和内核中的相关系统调用之间常常还是存在紧密联系的。事实上，许多 POSIX 和 Windows 的 API 还是类似于 UNIX、Linux 和 Windows 操作系统提供的系统调用。

1. 在Linux系统中如何自定义实现一个系统调用，请简述过程。

（1）获取root权限（即管理员权限）

（2）准备工作,安装相关编译程序

（3）添加系统调用函数

（4）设置系统调用号

（5）设置对应系统调用的头文件

（6）编译内核

1. 微内核结构的操作系统有什么优势？

(1)提高了系统的可扩展性;

(2)增强了系统的可靠性;

(3)可移植性

(4) 提供了对分布式系统的支持

(5)融入了面向对象技术

1. 指令的执行分为用户模式与内核模式，这两种模式有什么特点？

内核模式：在内核模式下,执行代码具有对底层硬件的完全且不受限制的访问.它可以执行任何CPU指令并引用任何内存地址.内核模式通常保留给操作系统的最低级别,最受信任的功能.内核模式下的崩溃是灾难性的; 他们将停止整个PC.

用户模式：在用户模式下,执行代码无法直接访问硬件或参考内存.以用户模式运行的代码必须委托给系统API才能访问硬件或内存.由于这种隔离所提供的保护,用户模式下的崩溃始终是可恢复的.您计算机上运行的大多数代码都将以用户模式执行.

1. 请简述你对虚拟机的理解。

10、Linux系统内核动态模块加载机制有什么优势？

内核模块动态加载的技术是指：可以让操作系统内核在需要时载入和执行的代码，同时在不需要的时候可以卸载。这种机制扩展了操作系统的内核功能，却不需要重新启动系统，。

**第3章 进程与线程 习题**

* 1. 操作系统中为什么要引入“进程”？

进程是计算机中的程序关于某数据集合上的一次运行活动，是系统进行资源分配和调度的基本单位，是操作系统结构的基础。在操作系统中引入进程，是为了实现多个程序的并发执行。传统的程序不能与其他程序并发执行，只有在为之创建进程后，才能与其他程序(进程)并发执行。

* 1. 进程有哪些基本状态？他们之间的转换是怎样发生的？

通常,进程有五种基本状态，除了新建和完成状态之外，还有3种可相互转化的状态

·等待态:等待某个事件的完成; ·就绪态:等待系统分配处理器以便运行; ·运行态:占有处理器正在运行。这些状态可相互转换：

运行态→等待态 往往是由于等待外设,等待主存等资源分配或等待人工干预而引起的。

等待态→就绪态 则是等待的条件已满足,只需分配到处理器后就能运行。

运行态→就绪态 不是由于自身原因,而是由外界原因使运行状态的进程让出处理器,这时候就变成就绪态。例如时间片用完,或有更高优先级的进程来抢占处理器等。

就绪态→运行态 系统按某种策略选中就绪队列中的一个进程占用处理器,此时就变成了运行态

* 1. 通常有哪些事件会导致创建一个进程？

引起进程创建的原因有以下几种：

1）操作系统准备接纳一个新的批处理作业时，由作业调度程序为用户作业创建相应的新进程。

1. 终端用户登录到系统时会创建进程。

3）操作系统为用户创建一个服务进程。如，用户请求打印一个文件，操作系统可以创建一个打印进程来完成。

4）现有进程可以派生其子进程，并与其并行执行任务。

4、请简述创建一个进程的过程。

创建一个新进程，操作系统调用进程创建原语来实现，进程创建原语需要完成以下步骤：

（1）给新进程分配一个唯一的进程标识符，并申请一个空白的PCB(PCB是有限的)。若PCB申请失败则创建失败。

（2）给进程分配空间分配包括程序、数据、用户栈等。

（3）将新进程插入就绪队列和进程隶属关系族群中。

1. 创建或扩充其它数据结构，如为进程创建记账文件等。

5、当使用fork()创建一个子进程的时候，子进程有什么特点？父进程与子进程被调度的顺序是怎样的？

**fork()函数**功能：从原进程中创建一个新进程，新进程为子进程，原进程为父进程。该函数有两个返回值：子进程中返回0，父进程中返回进程id。

创建新的子进程成功后，将父进程的文本段、数据段、堆栈都复制一份给子进程，但**子进程有自己独立的空间**，子进程对这些内存的**修改并不会影响父进程空间的相应内存**。

父进程与子进程被调度的顺序是随机的。

6、进程控制块的作用是什么？

进程控制块(PCB)是进程存在的惟一标志，是操作系统进程管理中一种重要的[数据结构](https://so.csdn.net/so/search?q=%E6%95%B0%E6%8D%AE%E7%BB%93%E6%9E%84&spm=1001.2101.3001.7020" \t "_blank)。PCB存放进程的管理和控制信息的数据结构称为进程控制块。它是进程管理和控制的重要的数据结构，在进程创建时建立PCB，并伴随进程运行的全过程，直到进程撤销而撤销。

7、为什么处理器的执行需要用户态和内核态两种模式？

当执行操作系统程序时,处理机处于核心态。它有较高的特权,可以执行所有的指令,包括一般用户程序中不能使用的特权指令,从而能对所有寄存器和内存进行访问、启动I/O操作等。

用户程序是在用户态下执行,它的权限较低,只能执行指令集中非特权指令。设置这两种不同状态的目的是为了保护操作系统程序(特别是其内核部分),防止受到用户程序的损害。

1. 什么是线程？试述线程与进程的区别？

**线程是轻量级进程，是操作系统进行调度的最小单位。**一个线程是一个任务（一个程序

段）的一次执行过程。线程不占有内存空间，它包括在进程的内存空间中。在同一个进程内，多个线程共享进程的资源。一个进程至少有一个线程。

根本区别：进程操作系统分配资源的基本单位，而线程是任务调度和值行的基本单位

在开销方面：每个进程都有独立的代码和数据空间（程序上下文） 程序之前的切换会有较大的开销，线程可以看做是轻量级的进程，同一类线程共享代码和数据空间，每个线程都有自己独立的运行栈和程序计数器，线程之间切换的开销小。

所处环境：在操作系统中能同时运行多个进程（程序）；而在同一个进程（程序）中有

多个线程同时执行（通过CPU调度，在每个时间片中只有一个线程执行）

内存分配方面：系统在运行的时候会给每个进程分配不同的内存空间，对于线程而言，

除了CPU以外，系统不会为线程分配内存（线程所用资源来自其所属进程的资源） 线程组之间只能共享资源。

包含关系：没有线程的进程可以看做是单线程的，如果一个进程内有多个线程，则执

行过程不是一条线的，而是多条线（线程）共同完成的；线程是进程的一部分，所以线程也被称为轻权进程或者轻量级进程

1. 请解释什么是写时拷贝？

写时拷贝在数据第一次写入到某个存储位置时，首先将原有内容拷贝出来，写到另一位置处，然后再将数据写入到存储设备中，该技术只拷贝在拷贝初始化开始之后修改过的数据。

传统的fork()系统调用直接把所有的资源复制给新创建的进程。这种实现过于简单并且效率低下，因为它拷贝的数据或许可以共享，Linux的fork()使用写时拷贝(copy-on-write)页实现。写时拷贝是一种可以推迟甚至避免拷贝数据的技术。内核此时并不复制整个进程的地址空间，而是让父子进程共享同一个地址空间。只用在需要写入的时候才会复制地址空间，从而使各个进行拥有各自的地址空间。也就是说，资源的复制是在需要写入的时候才会进行，在此之前，只有以只读方式共享。这种技术使地址空间上的页的拷贝被推迟到实际发生写入的时候

1. 线程池技术引入的原因是什么？有什么优点与缺点？

线程池的出现正是着眼于减少管理线程的开销而产生的技术。

线程池的性能损耗优于线程（通过共享和回收线程的方式实现），但是**也会有如下一些缺点**：

* 线程池不支持线程的取消、完成、失败通知等交互性操作。
* 线程池不支持线程执行的先后次序排序。
* 不能设置池化线程（线程池内的线程）的Name，会增加代码调试难度。
* 池化线程通常都是后台线程，优先级一般不会设置的比较高。
* 池化线程阻塞会影响性能（阻塞会使语言运行库CLR错误地认为它占用了大量CPU。CLR能够检测或补偿（往池中注入更多线程），但是这可能使线程池受到后续超负荷的印象）。
* 线程池使用的是全局队列，全局队列中的线程依旧会存在竞争共享资源的情况，从而影响性能。

11、Linux系统中创建子进程的调用fork()与vfork()有什么区别与联系？

区别一：

   fork函数创建子进程后，父子进程同时运行，运行先后顺序取决于linux内核的进程调度。

  vfork函数创建子进程后保证子进程先运行，当子进程调用exit退出后，父进程在执行

区别二：

  fork函数创建的子进程直接拷贝父进程的空间，子进程的变量修改不会影响父进程的变量。

  vfork函数创建子进程是直接使用父进程的存储空间，不进行拷贝，所以当子进程修改了变量的值时，父进程的值也会随之改变。

**第4章 进程同步 习题**

1. 什么是进程间的互斥？什么是进程同步关系？

答：进程互斥是进程之间的间接制约关系。在多道系统中，每次只允许一个进程访问的资源称为临界资源，进程互斥就是保证每次只有一个进程使用临界资源。

进程同步是进程间共同完成一项任务时直接发生相互作用的关系。为进程之间的直接制约关系。在多道环境下，这种进程间在执行次序上的协调是必不可少的。

1. 信号量的物理意义是什么？

信号量的物理意义是:当信号量的值大于零时,表示 可用资源的数量 ; 当信号量值小于零时,其绝对值为 等待使用信号量所代表资源的进程的数量。

1. 硬件机制解决进程互斥问题的方式有哪些？它们有什么缺陷？

1）中断禁止：用于单CPU，使用屏蔽中断的方法，屏蔽时无法切换进程，达到进程互斥的目的。此方法显而易见的效率低；屏蔽期间无法响应外部请求；无法切换进程；无法工作在多处理器环境

2）专用机器指令

在中断禁止机制难以奏效的情况下，我们还有另一种基于硬件的方案——使用专用的机器指令，这些指令可以实现程序对相同内存单元的互斥访问。

缺点，如：导致CPU空耗。当一个进程等待进入临界区时，会不断地持续检测，消耗处理器时间；导致饥饿进程。由于各种原因，当有多个进程都在等待进入临界区时，在某些极端情况下有可能某个进程永远无法进入临界区，发生饿死现象。

1. 忙等待的含义是什么？在操作系统中还有哪些其他形式的等待？忙等待能完全避免吗？

所谓忙等待是不断测试等待一个条件满足。

除了忙等待之外,操作系统中还有一种等待是让进程在等待状态阻塞等待一个条件满足,当条件满足时,通过中断告知进程。

忙等待的进程浪费CPU的时间,降低了效率,并且有可能导致优先级反转。但是并不能完全避免忙等待。如果在等待时间不是很长,并且系统内部进程的数量小,并且对CPU利用率要求不高的无优先级或者不同优先级的进程不会有同一个临界区的小型系统而言,可以考虑使用忙等待的方式解决简单的问题。另外,在多CPU的小型系统中,忙等待所导致的优先级反转的问题并不存在,因此这样的有优先级的小型系统也可以使用忙等待。

5、 进程之间存在哪几种相互制约关系？各是什么原因引起的？下列活动分别属于哪种制约关系？

（1）若干同学去图书馆借书；

（2）两队举行篮球比赛；

（3）流水线生产的各道工序；

（4）商品生产和社会消费。

进程之间存在两种制约关系，即同步和互斥。 同步是由于并发进程之间需要协调完成同一个任务时引起的一种关系，为一个进程等待另一个进程向它直接发送消息或数据时的一种制约关系。 互斥是由于并发进程之间竞争系统的临界资源引起的，为一个进程等待另一个进程已经占有的必须互斥使用的资源时的一种制约关系。

（1）是互斥关系，同一本书只能被一个学生借阅，或者任何时刻只能有一个学生借阅一本书。

（2）是互斥关系，篮球是互斥资源。

（3）是同步关系，一个工序完成后开始下一个工序。

（4）是同步关系，生产商品后才能消费。

6、设有两个优先级相同的进程P1和P2如下。信号量S1和S2的初值均为0，试问P1、P2并发执行结束后，x=?，y=?，z=?

〈进程P1〉 〈进程P2〉

y:=1; x:=1;

y:=y+2; x:=x+1;

V(S1); P(S1);

z:=y+1; x:=x+y;

P(S2)； V(S2)；

y:=z+y; z:=x+z;

解:因为P1和P2是两个并发进程,所以进程调度程序调度P1和P2的顺序是不确定的。

这里不妨假设P1先执行。进程P1执行到语句P(S2)时,S2=-1,进程P1阻塞。此时,y=3,z=4。当进程调度程序调度到进程P2时,由于进程P1已执行了V(S1),进程P2在执行P(S1)时并未阻塞而继续执行,当执行到V(S2)时,将P1唤醒,分成两种情况:

(1)然后执行P2最后一个语句z:=x+z,此时x=5,z=9。当进程P1再次被调度时,继续执行P1的最后一个语句,此时y=12,最终结果是:x=5,y=12,z=9。

(2)如果当P2进程执行到V(S2)时,将P1唤醒,然后P2进程被中断,此时x=5,y=3,z=4。P1进程开始执行然后执行最后一个语句y:=z+y,此时x=5,y=7,z=4。然后P2进程被调度,执行z:=x+z,此时x=5,y=7,z=9。

如果P2先执行,则执行结果与上面相同。

7、有三个并发进程input,copy,output，input进程负责从输入设备读入信息并把信息放入到缓冲区buffer1中，copy进程负责把buffer1中的内容加工后复制到缓冲区buffer2中；output进程负责从buffer2中取出信息并送到打印机中输出，请使用WAIT，SIGNAL操作写出上述三个进程的同步算法。

EmptyB1、EmptyB2表示buffer1与buffer2缓冲区中的空闲单元的数目，

其初值为 N1与N2

Full\_B1、Full\_B2表示buffer1与buffer2满缓冲单元的数目，其初值为 0

output进程：

Repeat

Wait(Full\_B2);

从buffer2中取出一个数据Y；

打印Y；

Signal(EmptyB2);

Until false;

copy进程：

Repeat

Wait(EmptyB2);

Wait(Full-B1);

从buffer1中取出一个数据；

将该数据拷贝到buffer2中；

Signal(Full\_B2);

Signal(EmptyB1);

Until false;

Input进程：

Repeat

读入信息X；

Wait(EmptyB1);

X存入Buffer1中；

Signal(Full\_B1);

Until false;

8、a, b两点之间是一段东西向的单行车道，现要设计一个自动管理系统，管理规则如下：当ab之间有车辆在行驶时，同方向的车可以同时驶入ab段，但另一方向的车必须在ab段外等待；当ab之间无车辆在行驶时，到达a点（或b点）的车辆可以进入ab段，但不能从a点与b点同时驶入；当某方向在ab段行驶的车辆驶出了ab段且暂无车辆进入ab段时，应让另一方向等待的车辆进入ab段行驶。请用信号量为工具，对ab段实现正确管理以保证行驶安全。

首先，设置互斥信号量Sab=1，用于a、b点的车辆互斥进入ab段；

然后，分别设置共享变量ab=0用于记录当前ab段上由a点进入的车辆数量；共享变量ba=0用于记录当前ab=段上由b点进入车辆的数量；

最后，设置互斥信号量S1=1用于ab段的车辆互斥访问共享变量ab；设置互斥信号量S2=1用于ba段的车辆互斥访问共享变量ba

semaphore S1=1,S2=1,Sab=1;

int ab=ba=0;

Pab： Pba：

Repeat Repeat

Wait(S1) Wait(S2)

abcount=abcount+1; bacount=bacount+1;

if abcount==1 then wait(Sab) if bacount==1 then wait(Sab)

signal(S1) signal(S2)

进入车道行驶； 进入车道行驶；

Wait(S1) Wait(S2)

abcount=abcount-1; bacount=bacount-1;

if abcount==0 then signal(Sab) if bacount==0 then signal(Sab)

signal(S1) signal(S2);

until false; until false;

1. 在公共汽车上，司机与售票员的工作流程分别为，司机：启动车辆→正常运行→到站停车→启动车辆….；售票员：关车门→售票→开车门→关车门….。为保证乘客安全，司机与售票员要密切配合，协调工作。请用信号量来实现司机与售票员之间的同步。

约束：怎么密切配合协调工作才能保证安全呢？

a) 关车门之后再启动车辆；利用前驱图解释

b) 到站停车之后再开车门；

2 根据约束定义信号量；

关车门和启动车辆需要一个信号量进行同步S1；到站停车和开车门之间需要一个信号量进行同步S2；

3 建立几个进程呢？

a) 为司机建立一个进程Driver；

b) 为售票员建立一个进程Conductor；

Conductor：

Repeat

关车门；

售票；

开车门；

Until false;

Driver：

Repeat

启动车辆；

正常行驶；

到站停车；

Until false;

4 加入同步关系：

Conductor：

Repeat

关车门；

Signal(s1);

售票；

Wait(s2)

开车门；

Until false;

Var s1,s2:semorphore=0,0;

Driver：

Repeat

Wait (s1);

启动车辆；

正常行驶；

到站停车；

Signal(s2)

Until false;

10、假如一个家庭有以下成员“爸爸、妈妈、两个儿子、两个女儿；桌上有一个空盘，最多可以容纳两个水果。爸爸可向盘中放入苹果，妈妈可向盘中放入橘子，两个儿子专吃盘中的橘子，两个女儿专吃盘中的苹果。试用信号量描述各个家庭成员的行为。

Semaphore empty, orange, apple,

Empty=2; orange=apple=0,

void mother( )

{

while(true)

{

wait(empty);

put an orange;

signal(orange)；

}

}

void father( )

{

while(true)

{

wait(empty);

put an apple;

signal(apple)；

}

}

void son ( )

{

while(true)

{ wait(orange);

eat the orange;

signal(empty);

}

}

void daughter( )

{

while(true)

{ wait(apple);

eat the apple;

signal(empty);

}

}

1. 在一个飞机订票系统中，多个用户共享一个数据库。多用户同时查询是可以接受的，但若一个用户要订票需更新数据库时，其余所有用户都不可以访问数据库。请画出用户查询与订票的逻辑框图。要求：当一个用户订票而需要更新数据库时，不能因不断有查询者的到来而使他长期等待。

答：本题是典型的读者一写者问题。查询操作是读者,订票操作是写者,而且要求写者优先。

为了达到这一控制效果,可以引入一个变量rc,用于记录当前正在运行的读者进程数。每个读者进程进入系统后需对rc值加1。当rc值由O变为1时,说明是第一个读者进程进入,因此需要该读者进程对控制写者进程的信号量Srw进行P操作,以便与写者进程互斥运行:当rc值由非0值增加时,说明不是第一个读者进程,此时控制写者进程的信号量已经过P操作控制禁止写者进程进入,因此不需要再次对该信号量进行P操作。当读者进程退出时,需对rc做减1操作。如发现减1后m值变为0,说明是最后一个读者进程退出,因此需要该读者进程对控制写者进程的信号量Srw进行V操作,以便使写者进程能够进入。资源计数变量rc也是一个临界资源,需要用信号量Src对它进行互斥访问控制。为了提高写者的优先级,我们还增加了一个信号量S,用以在写进程到达时封锁其后续的读者进程。用户查询与订票的逻辑框图如图所示。

查询者 定票者

wait (S)

wait(S)

wait(Src)

wait (Srw)

rc=rc+1

更新数据库中的数据

N

rc=1?

signal (Srw)

Y

signal (S)

wait (Srw)

wait (Src)

wait (S)

在数据库中查询所需的信息

wait (Src)

Rc=rc-1

N

Rc=0?

Y

signal(Srw)

signal (Src)

12、一个理发店由一个有N张椅子的等候室和一个放有一张理发椅的理发室组成。若没有要理发的顾客，则理发师就去睡觉，若一个顾客走进理发店且所有的椅子都被占用了，则该顾客就离开理发店，若理发师正在为人理发，则该顾客就找一张空椅子坐下等待，若理发师在睡觉，则顾客就唤醒他。试用信号量设计一个协调理发师和顾客的程序。

Semaphore costomers=0; 等候的顾客数（不包括正在理发的）

Semaphore barbers=0; 等候顾客的理发师数

Semaphore mutex=1;

Int waiting =0; 等候的顾客数（还没有理发，实际是customers的备份，为了读取信号量的当前值）；

Void customers(void)

{ wait(mutex);

if(waiting<N)

{ waiting = waiting + 1;

signal(customers);

signal(mutex);

wait(barbers);

get\_hair( );

}

else {V(mutex);}

}

Void barber(void)

{ while (true)

{ wait(customers);

wait(mutex);

waiting = waiting – 1 ;

signal(barbers);

signal(mutex);

cut\_hair( );

}

提⽰：考虑⼀下理发师（barber）重复的下列活动：（1）睡觉；（2）为顾客理发；

顾客（customers）重复的下列活动：（3）在椅⼦上等候；（4）理发；

离开；

显然，理发师在（1）处要考察是否有顾客等候理发，如果没有，理发师睡觉；在（2）处理发师等待最先进⼊理发店的顾客唤醒，开始理发。

顾客在（3）处先看是否有座位，没有则离开；等候理发的顾客在（4）处被理发师唤醒（最先理发的顾客要唤醒理发师）；理发结束后离开。

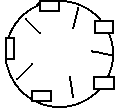
在这两个活动中，从资源的⾓度来看，理发师是顾客争⽤的资源，⽤信号量barber表⽰，初值为0；除此以外，顾客还要争⽤n 张椅⼦，信号量customers表⽰等候理发的顾客数，初值为0；最后设置信号灯变量mutex ⽤于这两个活动对资源barber、customers的互斥，初值为1。

13、指出下列哲学家就餐问题的算法在什么情况下会导致死锁，并改进此算法，使它不会产生死锁。算法描述：

五个哲学家在一张圆桌上进行思考和吃饭。哲学家思考时，并不影响他人。只有当他吃饭时，他才试图拿起左右两根筷子（一根一根的拿起）。如果筷子已在他人手上，则需等待。只有当他同时拿起左右两根筷子时，才可以吃饭。如图4.11所示：

程序描述为：（第i个哲学家,i=0,1,2,3,4）

Var chopstick: array[0..4] of semaphore; /\* 各信号量初值均为1\*/

 Repeat

P(chopstick[i]); /\* P操作，拿左筷子\*/

P(chopstick[i+1 mod 5]); /\* P操作，拿右筷子\*/

Eat();/\*吃饭\*/

V(chopstick[i]); /\*V操作，放下左筷子\*/

V(chopstick[i+1 mod 5]); /\* V操作，放下右筷子\*/ 图7.1

Think();/\*思考\*/

Until false;

图4.11哲学家进餐示意图

解：

若某个时刻，每位哲学家都刚好拿起身边的筷子想要吃饭，同时都没办法获得另一只筷子，此时算法就会陷入死所状态。

如果相邻的两位哲学家在同时都想吃面条时，申请筷子的顺序不一样，如一个先申请右手的筷子，另一个则先申请左手的筷子，这样就不可能形成环路等待的情况。因此，我们可以将前面的算法作一些改进，如做一些这样的规定：编号为奇数的哲学家先申请右手的筷子，然后再申请左手的筷子；编号为偶数的哲学家先申请左手的筷子，然后再申请右手的筷子。此时算法可以描述如下：

void Pi( )

{semaphore S[5]={1,1,1,1,1};

int i=0;

while(true)

{Think;

if(i mod 2 == 0)

{

wait(ChopStick [(i+1) mod 5]);

wait(ChopStick [i]);

eat noodles;

signal(ChopStick [i])；

signal(ChopStick [(i+1) mod 5])；

}

else

{

wait(ChopStick [i]);

wait(ChopStick [(i+1) mod 5]);

eat noodles;

signal(ChopStick [(i+1) mod 5])；

signal(ChopStick [i])；

}

}

}

**第5章 死锁 习题**

1. 什么是死锁？请举一个生活中发生死锁的例子？

死锁是指两个或两个以上的进程在执行过程中，由于竞争资源或者由于彼此通信而造成的一种阻塞的现象，若无外力作用，它们都将无法推进下去，此时称系统处于死锁状态或系统产生了死锁，这些永远在互相等待的进程称为死锁进程；

举例:一根独木桥上有两个人迎面走，相遇时，都在等着对方让路，但是谁也不同意退回去让对方先走，导致谁也到不了对岸，两个人就是两个程序，他们都占有桥这个资源不愿放手，于是便一直处于等待状态，于是形成死锁。

1. 什么原因引起了死锁？产生了死锁的四个条件是什么？

引起死锁的原因是资源竞争，死锁的发生必须具体以下四个条件：

（1）互斥条件。指进程的共享资源必须保持使用的互斥性，即任何一个时刻只能分配给一个进程使用，互斥条件是形成死锁最根本的原因，因为如果资源不要求排它性地使用，那么一定不会造成请求资源而无法满足的局面。

（2）占有且等待条件。一个进程占有了某些资源之后又要申请新的资源而得不到满足时，处于等待资源的状态，且不释放已经占用的资源。

（3）不可剥夺条件。任何进程不能抢夺另一个进程所占用的资源，即已经被占用的资源只能由占用进程自己来释放。

（4）环路条件。存在一组进程P1,P2,…,Pn，其中每个进程分别等待另一个进程所占用的资源，形成环路等待条件。

1. 解除死锁的三种方法是什么，请分别描述各自的思想。

解决死锁问题常采用的措施有：

死锁的预防。通过设置某些限制条件，去破坏产生死锁的四个必要条件中的一个或几个，来预防死锁。（常见的有预先分配策略和有序分配策略）

死锁避免。在资源的动态分配过程中，用某种方法防止系统进入不安全状态，从而避免死锁。

死锁动态检测与恢复。系统检测机构及时地检测出死锁的发生，然后采取某种措施解除死锁。

* 1. 不安全状态是否必然导致系统进入死锁状态？

不是必然导致进入死锁

5、某系统A,B,C,D这4类资源供5个进程共享，进程对资源的需求和分配情况如表所示。现在系统还剩资源A类1个，B类5个，C类2个和D类0个，按照银行家算法回答下面问题：

|  |  |  |
| --- | --- | --- |
| 进程 | 已占有资源数目 | 最大需求数 |
| A B C D | A B C D |
| P1 | 0 0 1 2 | 0 0 1 2 |
| P2 | 1 0 0 0 | 1 7 5 0 |
| P3 | 1 3 5 4 | 2 3 5 6 |
| P4 | 0 6 3 2 | 0 6 5 2 |
| P5 | 0 0 1 4 | 1. 6 5 6 |

1）现在系统是否处于安全状态？

2）如果现在进程P2提出需要(0,4,2,0)个资源的请求，系统能否去满足它的请求？

解：现在可用资源向量为：(1,5,2,0)，计算出每个进程仍需资源的矩阵如下所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 进程 | 已占有资源数目 | 最大需求数 | 尚需资源数 |
| A B C D | A B C D | A B C D |
| P1 | 0 0 1 2 | 0 0 1 2 | 0 0 0 0 |
| P2 | 1 0 0 0 | 1 7 5 0 | 0 7 5 0 |
| P3 | 1 3 5 4 | 2 3 5 6 | 1 0 0  2 |
| P4 | 0 6 3 2 | 0 6 5 2 | 0 0 2 0 |
| P5 | 1. 0 1 4 | 0 6 5 6 | 0 6 4 2 |

在那个地址,进程P1已经拥有足够资源,在执行完后可归还C和D资源,同时,进程P3能够取得足够的资源完成执行,其后,P2、P4、P5能够依次取得足够的资源来完成,因此,用银行家算法分派资源时,系统出于平安状态。安全序列为:P1,P3,P2,P4,P5

（2）P2 进程提出资源请求 (0,4,2,0),采用银行家算法，假设资源获得分配，则相关矩阵如下表：现在可用资源向量为：(1,1,0,0)

|  |  |  |  |
| --- | --- | --- | --- |
| 进程 | 已占有资源数目 | 最大需求数 | 尚需资源数 |
| A B C D | A B C D | A B C D |
| P1 | 0 0 1 2 | 0 0 1 2 | 0 0 0 0 |
| P2 | 1 4 2 0 | 1 7 5 0 | 0 3 3 0 |
| P3 | 1 3 5 4 | 2 3 5 6 | 1 0 0  2 |
| P4 | 0 6 3 2 | 0 6 5 2 | 0 0 2 0 |
| P5 | 1. 0 1 4 | 0 6 5 6 | 0 6 4 2 |

P1进程可执行完毕，然后释放资源，则可用资源向量为：(1,1,0,0)+(0,0,1,2)=(1,1,1,2)

P3进程可执行完毕，然后释放资源，则可用资源向量为：(1,1,1,2)+(1,3,5,4)=(2,4,6,6)

P2进程可执行完毕，然后释放资源，则可用资源向量为：(2,4,6,6)+(1,4,2,0)=(3,8,8,6)

P4进程可执行完毕，然后释放资源，则可用资源向量为：(3,8,8,6)+(0,6,3,2)=(3,14,11,8)

P5进程可执行完毕，然后释放资源，则可用资源向量为：(3,14,11,8)+(0,0,1,4)=(3,14,12,12)

此时系统是安全状态，所以P2进程的请求是可以分配的。

6. 设有三个进程P1,P1,P3，各按如下所示顺序执行程序代码：

进程P1 进程P2 进程P3

↓ ↓ ↓

P(s1) P(s3) P(s2)

P(s2) P(s1) P(s3)

… … …

V(s1) V(s3) V(s2)

V(s2) V(s1) V(s3)

↓ ↓ ↓

其中s1,s2,s3是信号量，且初值均为1.

在执行时能否产生死锁？如果可能产生死锁，请说明在什么情况下产生死锁？并给出一个防止死锁产生的修改办法。

答:可能产生死锁。因为当进程P1执行P(s1),进程P2执行P(s3),进程P3执行P(s2)后,三个资源(即信号量s1,s2,s3)被三个进程分别占用,接下来任何一个进程都无法得到所申请的资源,于是都无限地循环等待,造成死锁。

一个防止死锁产生的办法是:进程申请信号量时,按序申请。如下图所示

进程P1 进程P2 进程P3

↓ ↓ ↓

P(s1) P(s1) P(s2)

P(s2) P(s3) P(s3)

┉ ┉ ┉

V(s1) V(s1) V(s2)

V(s2) V(s3) V(s3)

↓ ↓ ↓

1. 有三个进程P1，P2和P3并发工作。进程P1需用资源S3和S1；进程P2需用资源S1

和S2；进程P3需用资源S2和S3。回答：  
(1)若对资源分配不加限制，会发生什么情况?为什么?  
(2)为保证进程正确工作，应采用怎样的资源分配策略?为什么?

解：

(1)若对资源的分配不加限制系统可能出现死锁现象。因为比如P1 进程占了S3 资源P2 进程占了S1 资源P3 进程占了S2 资源此时系统已经没有S1 、S2 和S3资源了而P1 则申请S1 资源P2 申请S2 资源P3 申请S3 资源此时显然系统处于死锁状态了。

(2)为了保证进程正确地工作系统应该采取一定的资源分配策略来限制死锁发生的必要条件。 ①限制请求和保持条件的发生。可以要求进程只有一次性申请到了所有的它所需的所有资源后才能得到运行否则将放弃它申请到的资源。 ②限制不剥夺条件的发生。可以允许系统中的进程在申请了系统中已经没有了的资源后那么可以从其他没有在运行的进程中抢夺该资源。 ③限制环路等待条件的发生。可以要求进程按照资源的编号来顺序地申请资源即P1 进程先申请S1 再申请S3， P2 进程先申请S1 再申请S2， P3 进程先申请S2 再申请S3 。 以上三种资源分配策略都限制一个或多个死锁发生的必要条件可以预防死锁的发生。

8、设系统有三种类型的资源，数量为(4，2，2)，系统中有进程A，B，C按如下顺序请求资源：  
 进程A申请(3，2，1)  
 进程B申请(1，0，1)  
 进程A申请(0，1，0)  
 进程C申请(2，0，0)  
请你给出一种防止死锁的资源剥夺分配策略，完成上述请求序列，并列出资源分配过程，指明哪些进程需要等待，哪些资源被剥夺。

解：

① 分配策略为:当进程Pi申请ri类资源时,检查ri中有无可分配的资源:有则分配给Pi;否则将Pi占有的资源全部释放而进入等待状态。(Pi等待原占有的所有资源和新申请的资源)

② 资源分配过程: 剩余资源

进程A:(3,2,1) (1,0,1)

进程B:(1,0,1) (0,0,0)

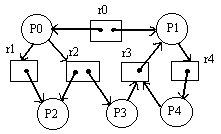
进程A:(0,1,0)(不满足) (3,2,1)

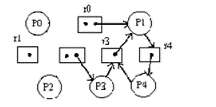
A的所有资源被剥夺,A处于等待

进程C:(2,0,0) (1,2,1)

C,B完成之后,A可完成。

1. 化简图6的资源分配图，并说明有无进程处于死锁状态。



化简结果为右图：

右图是不可完全化简的，故有死锁，

死锁的进程为：P1、P3、P4。

10、针对哲学家进餐问题，请给出本章例题中未给出代码的无死锁解决方案。

**第6章 处理器调度 习题**

1. 简述三种类型的处理器调度。

处理器调度分为：高级调度，中级调度，低级调度

高级调度主要任务：在多道批处理操作系统中，从输入系统的一批作业中按照预订的调度策略挑选若干作业进入主存，为其分配所需资源，并创建作业的响应用户进程后便完成高级调度任务。

中级调度：根据主存资源决定主存中所能容纳的进程数目，并根据进程的当前状态来决定辅助存储器和主存中进程的对换。

低级调度：根据某种原则决定就绪队列中的哪个进程或者内核级线程获得处理器，并将处理器出让给它使用。

1. 抢占式调度和非抢占式调度有何区别？

|  |  |  |
| --- | --- | --- |
| **编号** | **抢占式调度** | **非抢占式调度** |
| 1 | CPU 在一定时间内分配给进程。 | CPU被分配给进程，直到它结束执行或切换到等待状态。 |
| 2 | 抢占式调度的执行过程在执行过程中被中断。 | 非抢占式调度的执行过程在执行过程中不会中断。 |
| 3 | 抢占式调度通常将进程从就绪状态切换到运行状态，反之亦然，并维护就绪队列。 | 非抢占式调度不会将进程从运行状态切换到就绪状态。 |
| 4 | 如果一个高优先级的进程频繁到达就绪队列，那么低优先级的进程就得等待很长时间，它可能要饿死。 | 如果 CPU 分配给突发时间较长的进程，那么突发时间较短的进程可能不得不饿死。 |
| 5 | 抢占式调度非常灵活，因为关键进程在进入就绪队列时被允许访问 CPU，无论当前正在执行什么进程。 | 非抢占式调度是刚性的，即使关键进程进入就绪队列，运行 CPU 的进程也不会受到干扰。 |
| 6 | 抢占式调度与成本相关，因为它必须维护共享数据的完整性。 | 非抢占式调度与成本无关。 |

1. 周转时间和响应时间有何区别？

响应时间：从提交第一个请求到产生第一个响应所用时间。衡量分时系统的响应性能。  
周转时间：从作业提交到作业完成的时间间隔。  
平均周转时间：衡量不同调度算法对相同作业流的调度性能

1. 多级反馈轮转调度算法中，为什么对不同就绪队列中的进程使用不同长度的时间片？

有以下几个原因：（1）、运行时间短的进程只需要经过前面几级队列就能得到结果，且它们被优先调度，有利于提高系统的吞吐率；（2）对运行时间长的进程在进入了低级就绪队列后可以得到较长的时间片，以减少调度次数来保证系统效率；（3）对经常使用外围设备的进程来说，每次等到外围设备传输结束后总是链入第一级就绪队列，它们会被优先调度，有利于处理器与外围设备以及外围设备之间的并行工作，从而提高系统资源的使用效率。因此，在分级调度算法中，对不同就绪队列中的进程规定使用不同长度的时间片。

1. 硬实时任务与软实时任务有什么区别？

硬实时任务是指系统必须满足任务对截止时间的要求，否则可能出现难以预测的结果。如：运载火箭的控制等。

软实时任务是指它的截止时间并不严格，偶尔错过了任务的截止时间，对系统产生的影响不大。如：网页内容的更新，火车售票系统等。

1. 周期性实时任务与非周期性实时任务有什么区别？

|  |  |
| --- | --- |
| 周期性实时任务 | 非周期性实时任务 |
| 定期任务会在一定的时间间隔后自我重复 | 非周期性实时任务可以随机发生。 |
| 这些任务由时钟中断控制。 | 这些任务不受时钟中断控制 |
| 两个连续任务发生的时间间隔不能为零。 | 两个连续任务发生之间的时间间隔可以为零。 |
| 周期性任务通常包括软实时任务和硬实时任务。 | 临时任务通常包括软实时任务。 |
| 定期任务可以轻松满足所有周期性任务实例的截止日期。 | 满足非定期任务的所有实例的最后期限是相当困难的。 |
| 定期任务包括绝大多数内部任务。 | 非周期性实时任务包括与用户的交互任务。 |
| 示例：按时间间隔从传感器获取信息。 | 示例：分布式系统中的日志记录任务。 |

7、在单道批处理系统中，有四个进程进入系统，进入时间及所需计算时间如下表所示。现忽略进程调度所花时间。当第一个进程进入系统后就可开始调度

|  |  |  |
| --- | --- | --- |
| 进程 | 进入时间 | 所需计算机时间 |
| 1 | 8：00 | 2小时 |
| 2 | 8：30 | 30分钟 |
| 3 | 9：00 | 6分钟 |
| 4 | 9：30 | 12分钟 |

1. 将分别采用“先来先服务”和“短作业优先”调度算法时，各个进程的开始时间，完成时间，周转时间分别填入下面的表中

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 先来先服务 | | | 短作业优先 | | |
| 进程 | 开始时间 | 完成时间 | 周转时间 | 开始时间 | 完成时间 | 周转时间 |
| 1 |  |  |  |  |  |  |
| 2 |  |  |  |  |  |  |
| 3 |  |  |  |  |  |  |
| 4 |  |  |  |  |  |  |

(2)采用“先来先服务”调度算法时，平均周转时间为：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

采用“短作业优先”调度算法时，平均周转时间为：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

解：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 先来先服务 | | | 短作业优先 | | |
| 作业 | 开始时间 | 完成时间 | 周转时间 | 开始时间 | 完成时间 | 周转时间 |
| 1 | 8：00 | 10：00 | 120分 | 8：00 | 10：00 | 120分 |
| 2 | 10：00 | 10：30 | 120分 | 10：18 | 10：48 | 138分 |
| 3 | 10：30 | 10：36 | 96分 | 10：00 | 10：06 | 66分 |
| 4 | 10：36 | 10：48 | 78分 | 10：06 | 10：18 | 48分 |

采用“先来先服务”调度算法时，平均周转时间为： 103.5分钟

采用“短作业优先”调度算法时，平均周转时间为： 93分钟

8、对于下列三个进程，采用不可强占的调度方式，当第一个进程进入系统后开始调度，它们的提交时间及执行时间由下表给出。分别计算采用先来先服务，最短作业优先和最高响应比调度算法时它们的调度顺序和平均周转时间。

|  |  |  |
| --- | --- | --- |
| 进程号 | 到达队列时间（时） | 需运行时间（小时） |
| 1 | 10：00 | 2：00 |
| 2 | 10：10 | 1：00 |
| 3 | 11：50 | 0：50 |

解：（1）先来先服务调度算法：

执行顺序：1----2-----3

平均周转时间：（120+170+120）/3 = 137分钟

（2）最短作业优先

执行顺序：1----3-----2

平均周转时间：（120+220+60）/3 = 133分钟

（3）最高响应比

进程1最先执行，当12时执行完毕，此时

进程2的响应比：（60+110）/60=28.3

进程3的响应比：（50+10）/50=1.2

因此进程2先执行

执行顺序：1----2-----3

平均周转时间：（120+170+120）/3 = 137分钟

9、某分时系统的时间片长度为100毫秒。设有A、B、C三个进程，它们各自需占用200毫秒CPU时间才能够完成运行。这三个进程在该分时系统中以字母次序轮流占用时间片（按字母次序轮转到某字母时，如果相应的进程不在系统中，则跳过该字母所对应的进程），操作系统占用的时间忽略不计。A、B、C进程进入系统的时间分别为第0毫秒、第50毫秒、第250毫秒。试问：

（1）C进程在第几毫秒开始运行？

（2）C进程的周转时间是多少毫秒？

这三个进程的运行时间如下图所示：

C进程在400毫秒开始运行

C进程的周转时间是350毫秒

B进入时间

100ms

200ms 300ms

400ms

500ms

600ms

C进入时间

C完成时间

A

B

C

10、考虑一组非周期性任务，下表给出了它们的执行简表，请给出这组任务的调度图

|  |  |  |  |
| --- | --- | --- | --- |
| 进程 | 到达时间 | 执行时间 | 开始截止时间 |
| A | 10 | 20 | 100 |
| B | 20 | 20 | 30 |
| C | 40 | 20 | 60 |
| D | 50 | 20 | 80 |
| E | 60 | 20 | 70 |

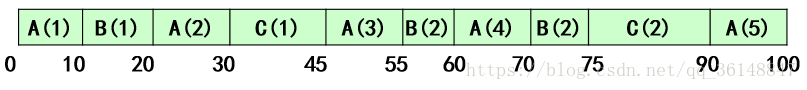
答：每一方格代表10个时间单元。

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 最早期限 |  | A | A |  | C | C | E | E | D | D |  |  |
| 有自愿空闲时间的最早期限 |  |  | B | B | C | C | E | E | D | D | A | A |
| 先来先服 |  | A | A |  | C | C | D | D |  |  |  |  |

11、考虑一组周期性任务，下表给出了它们的执行简表，请给出这组任务的调度图

|  |  |  |  |
| --- | --- | --- | --- |
| 进程 | 到达时间 | 执行时间 | 完成截止时间 |
| A（1） | 0 | 10 | 20 |
| A（2） | 20 | 10 | 40 |
| … | … | … | … |
| B（1） | 0 | 10 | 50 |
| B（2） | 50 | 10 | 100 |
| … | … | … | … |
| C（1） | 0 | 15 | 50 |
| C（2） | 50 | 15 | 100 |
| … | … | … | … |

“最早完成最后期限调度”是一种可抢占的调度，通过在每个可抢占点上优先调度最后期限最邻近的进程，可以满足系统的所有要求。分析即可得到如下调度图。



**第7章 内存管理 习题**

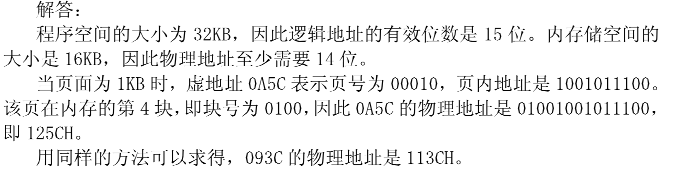
1、名词解释

地址重定位、 局部性原理、覆盖与交换

1. 存储保护的目的是什么？在页式存储管理中如何进行存储保护的？

答:存储保护的目的就是为了保护主存中各区域内的信息不被破坏,它包括存储的权限,存储不可跨越区域等等。为实现存储保护,必须由硬件和软件配合实现。比如在页式管理中,要由操作系统提供页表,硬件机构确定页的使用必须是安全的,如不能访问不属于自己的页等。

3、某虚拟存储器的用户空间共有32个页面，每页1KB，主存16MB。 假定某时刻为用户的第0，1，2，3页分别分配的页帧号为5，10，4，7，试将虚拟地址0A5C和093C变换为物理地址。



4、在一个请求分页系统中，采用LRU页面置换算法时，假如一个进程的页面访问序列为4，3，2，1，4，3，5，4，3，2，1，5，当分配给该进程的页帧数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是多少？

答:当分配给该作业的物理块数M为3时,缺页9次,缺页率:9/12=3/4:

当分配给该作业的物理块数M为4时,缺页10次,缺页10/12=5/6.

5、什么是系统抖动? 产生抖动的原因是什么?

.在请求分页存储管理中，可能出现这种情况，即对刚被替换出去的页，立即又要被访问。需要将它调入，因无空闲内存又要替换另一页，而后者又是即将被访问的页，于是造成了系统需花费大量的时间忙于进行这种频繁的页面交换，致使系统的实际效率很低，严重导致系统瘫痪，这种现象称为抖动现象。

产生该现象的主要原因是置换算法选择不当。

6、简述页式存储管理和段式存储管理的区别

答案:

(1)页是信息的物理单位,是系统管理的需要。段是信息的逻辑单位,它含有一组意义相对完整的信息,分段的目的是为了更好地实现共享,满足用户的要求。

(2)页的大小固定且由系统确定。(即将逻辑地址划分为页号和页内地址是由机器硬件实现的)段的长度不固定,因段而异,它取决于用户所编写的程序(通常由编译程序在对源程序进行编译时根据信息的性质来划分)。

(3)分页的进程空间是一维的,分段式进程空间是二维的。

(4)分页系统很难实现过程和数据的分离。不便于实现共享,也无法分别对过程和数据提供保护。分段系统确可以很容易实现共享与保护。

7、内存保护是否可以完全由软件来实现？为什么？

可以。存储保护的基础是动态地址翻译。而这个翻译既可以由硬件实现，也可以由软件实现，只不过软件实现起来效率低一些而已。事实上，历史上的MULTICS操作系就是由软件来实现存储保护的。

8、用可变分区方式管理主存时，假定主存中按地址顺序依次有五个空闲区，空闲区的大小为32K，10K，5K，228K，100K现有五个作业J1，J2，J3，J4和J5它们各需主存1K，10K，108K，28K，115K若采用首次适应分配算法能把这五个作业按J1～J5的次序全部装入主存吗？

答: （1）不行。

列表模拟J1~J5进入内存情况如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 初始空闲分区状态 | J1进入后空闲分区的状态 | J2进入后空闲分区的状态 | J3进入后空闲分区的状态 | J4进入后空闲分区的状态 | 没有满足J5运行条件的空闲分区 |
| 32K | 31K | 21K | 21K | 21K |  |
| 10K | 10K | 10K | 10K | 10K |  |
| 5K | 5K | 5K | 5K | 5K |  |
| 228K | 228K | 228K | 120K | 92K |  |
| 100K | 100K | 100K | 100K | 100K |  |

（2）

以J1,J2,J3,J5,J4的次序装入这五个作业可使主存空间利用率最高。

以上述顺序模拟装入过程列表如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 初始空闲分区状态 | J1进入后空闲分区的状态 | J2进入后空闲分区的状态 | J3进入后空闲分区的状态 | J5进入后空闲分区的状态 | J4进入后空闲分区的状态 |
| 32K | 31K | 21K | 21K | 21K | 21K |
| 10K | 10K | 10K | 10K | 10K | 10K |
| 5K | 5K | 5K | 5K | 5K | 5K |
| 228K | 228K | 228K | 120K | 5K | 5K |
| 100K | 100K | 100K | 100K | 100K | 72K |

这样可以将五个作业全部装入内存，使得内存利用率最高。

9、对一个将页表放在分页系统的内存中：

　　（1）如果访问内存需要0.2μs，有效访问时间为多少？

　　（2）如果增加一个快表，且假定在快表中找到页表项的几率高达90％，则有效访问时间又是多少（假定查找快表需花的时间为0）？

　 分析：每次访问数据时,若不使用快表,则需要两次访问内存,即先从内存的页表中读出页对应的块号,然后再根据形成的物理地址去存取数据;使用快表时,若能从快表中直接找到对应的页表项,则可立即形成物理地址去访问相应的数据,否则,仍需两次访问内存。

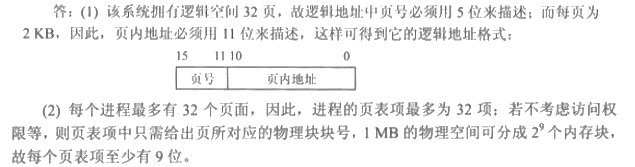
解：

(1)有效访问时间为:2×0.2=0.4μs

(2)有效访问时间为:0.9×0.2+(1-0.9)×2×0.2=0.22μs

10、某系统采用分页存储管理方式，拥有逻辑空间32页，每页2K，拥有物理空间1M

（1）写出逻辑地址的格式（2）若不考虑访问权限等，进程的页表项有多少项？每项至少有多少位？

解：（1）由于该系统拥有32个逻辑页面,所以页号必须用5位表示；而每页2K,页内地址需要11位来描述.

（2）进程的页表项最多为32项；若不考虑访问权限,每个页表项至少9位.

11、在分页存储管理系统中，逻辑地址的长度为16位，页面大小为4096字节，现有一逻辑地址为2F6AH，且第0、1、2页依次存放在页帧5、10、11中，问相应的物理地址是多少？

解：

4096B=212B ，16位寻址一共216B 。分页存储.共分的页:216/212=24=16，因此共分16页。

0页的地址范围 0 – FFFH； 1页的地址范围 1000H - 1FFFH； 2页得地址范围 2000H - 2FFFH； 11页 B000H – BFFFH； 第15页 F000H - FFFFH

2F6AH=10 1111 0110 1010 ，在2页的范围对应物理块11， 所以物理地址为: 2F6AH - 2000H + B000H = F6AH + B000H= BF6AH

12、在分页虚拟存储管理系统中，为什么说一条指令执行期间可能产生多个中断？

答：因分页虚拟管理方式中,只要进程的部分页在内存,该进程就能执行,而在执行过程中发现所要访问的指令或者数据不在内存时,则产生缺页中断,将所需的页面调入内存。在分页虚拟存储管理系统中,一条指令(如Copy A to B )可能跨了两个页,而其中要访问的操作数可能也跨了两个页。当要执行这类指令,而相应的页都不在内存时,就将产生多次缺页中断(如Copy A to B可能产生6次缺页中断)。

13、在分页虚拟存储管理系统中，假定系统为某进程分配了四个页帧（将开始4页先装入主存），页的引用顺序为：7，1，2，0，3，0，4，2，3，0，3，2，7，0，1，若采用FIFO调度算法、LRU调度算法时分别产生多少次缺页中断？

1）FIFO调度算法页框（页帧）变化情况：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 1 | 2 | 0 | 3 | 0 | 4 | 2 | 3 | 0 | 3 | 2 | 7 | 0 | 1 |
| 7 | 7 | 7 | 7 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
|  | 1 | 1 | 1 | 1 | 1 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
|  |  | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 7 | 7 | 7 |
|  |  |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| F | F | F | F | F |  | F |  |  |  |  |  | F |  | F |

产生缺页中断8次。

2）LRU调度算法页框（页帧）变化情况：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 1 | 2 | 0 | 3 | 0 | 4 | 2 | 3 | 0 | 3 | 2 | 7 | 0 | 1 |
| 7 | 7 | 7 | 7 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 1 |
|  | 1 | 1 | 1 | 1 | 1 | 4 | 4 | 4 | 4 | 4 | 4 | 7 | 7 | 7 |
|  |  | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
|  |  |  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F | F | F | F | F |  | F |  |  |  |  |  | F |  | F |

产生缺页中断8次。

14、现有一分页虚拟存取管理系统，其页表保存在寄存器中若有一个可用的空页或被替换的页未被修改，则它处理一个缺页中断需要8ms，如果被替换的页已被修改，则处理一个缺页中断需要20ms。内存存取时间为1μs，假定70％被替换的页被修改过，为保证有效存取时间不超过2μs，可接受的最大缺页率是多少？

分析：因为页表放在寄存器里，所以访问页表的时间可以忽略不计,则存取时间就包括内存存取时间和处理缺页中断的时间

答：如果用P表示缺页率，则有效访问时间不超过2μs,可表示为：

(1-p)×1μs＋p×(0.7×20ms＋0.3×8ms＋1μs)≤2μs

因此可计算出：p≤1/16400≈0.00006

15、Linux用户空间（进程）是否有高端内存概念？

  用户进程没有高端内存概念。只有在内核空间才存在高端内存。用户进程最多只能够访问3G物理内存，而内核进程能够访问所有物理内存。

16、Linux用户进程能访问多少物理内存？内核代码能访问多少物理内存？

以32位系统为例， 用户空间的进程只能访问整个虚拟地址空间的0-3GB部分，内核代码能访问0-4GB空间

**第8章 文件系统 习题**

1. 什么是文件管理

文件管理是[操作系统](https://baike.baidu.com/item/%E6%93%8D%E4%BD%9C%E7%B3%BB%E7%BB%9F/192?fromModule=lemma_inlink)的五大职能之一，主要涉及文件的逻辑组织和物理组织，目录的结构和管理。所谓文件管理，就是操作系统中实现文件统一管理的一组软件、被管理的文件以及为实施文件管理所需要的一些数据结构的总称（是操作系统中负责存取和管理文件信息的机构）从系统角度来看，[文件系统](https://baike.baidu.com/item/%E6%96%87%E4%BB%B6%E7%B3%BB%E7%BB%9F/4827215?fromModule=lemma_inlink" \t "_blank)是对文件[存储器](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8%E5%99%A8/1583185?fromModule=lemma_inlink)的存储空间进行组织，分配和回收，负责文件的存储，检索，共享和保护。从用户角度来看，文件系统主要是实现"按名取存"，文件系统的用户只要知道所需文件的文件名，就可存取文件中的信息，而无需知道这些文件究竟存放在什么地方。

2、文件系统实现按名存取主要是通过怎样的机制来实现的？

答:由于文件采用目录管理,仅仅给出文件的名字属性就可以定位该文件的文件目录项,就可以使用该文件,这就是按名存取。

文件系统采用树型方式管理的,目录文件也是采用文件的方式管理,给出文件及其路径就可以从文件系统的根查找各个节点的目录文件找到该文件。按名存取的本质就是找到指定文件的文件项。

3、列出并简单定义文件的几种常见逻辑组织形式。

文件的逻辑组织通常分为两种形式，即有结构文件和无结构文件。

其中常见的记录式文件就是有结构的文件，逻辑结构包括堆结构、顺序结构、散列结构等。

4、请解释文件控制块与文件的关系

文件控制块——用于控制和管理文件的数据结构,其中包括文件名、文件类型、位置、大小等信息。

文件控制块与文件一一对应,即在文件系统内部,给每个文件唯一地设置一个文件控制块,核心利用这种结构对文件实施各种管理

5、对目录执行的典型操作有哪些？

搜索,创建文件,删除文件,显示目录,修改目录

6、列出几种常见的记录组块方式。

我们常有三种记录组块方法（记录组织成数据块简称）

1. 固定组块法
2. 可变长跨块组块法
3. 可变长非跨块组块法

7、列出并简单定义三种文件的物理结构，各自的特点是什么。

文件的物理结构及其特点如下:

(1)连续文件:连续文件又称为顺序文件,它是把逻辑文件中的信息顺序地存放到一组邻接的物理盘块中而形成的物理文件。显然,这种文件结构保证了文件中逻辑记录的顺序与外存中文件占用盘块的顺序的一致性。连续文件的最大优点是顺序存取速度快,其缺点是由于存储文件要求有连续的存储空间,不便于文件的动态增长,且容易使外存产生碎片,降低了外存的利用率。

(2)链接文件:把一个逻辑上连续的文件分散存放在多个不要求连续的盘块中,再使用链接指针将这多个可能不连续的盘块链接起来,这样形成的物理文件称为链接文件。链接文件消除了外存的碎片,提高了外存的利用率,同时使文件很容易实现动态增长。根据对链接指针处理方式的不同,链接文件又可分为隐式链接和显示链接两种。

(3)索引文件:仍然是把一个逻辑上连续的文件分散存放在多个不要求连续的盘块中,并为该文件建立一张索引表,每个逻辑块占一个表项,按逻辑块号排列,表项内容为该逻辑块所对应的物理盘块号。然后再把索引表本身存放在另一个盘块中,把这个存放索引表的盘块称为索引块,并把索引块的盘块号作为文件的物理地址填入文件目录项中。

8、假设某个文件由长度为80个字符的100个逻辑记录组成，磁盘存储空间被划分成长度为2048个字符的块，为了有效地使用磁盘空间，可以采用成组方式把文件存放在磁盘上，请回答下列问题：

1. 改文件至少占用多少个磁盘存储块
2. 若该文件是以链接结构形式保存在磁盘上的，现用户要求使用第28个逻辑记录，写出系统为满足用户要求而应做的主要工作。

解：1）每块能存放的记录个数为「2048/80」=25个

100/25=4 至少占用4个磁盘存储块

2）0-24号记录存放在起始物理块中，25,26,27这几个逻辑记录存放在第2个物理块，因此第28个逻辑记录编号为27，位于第2个物理块的第3条记录。

9、举例说明数据分解的操作系过程

就以上题为例,假定某个文件由长度为 80个字符的 100个逻辑记录组成,磁盘存储空间被划分成长度为2048个字符的块,为有效地使用磁盘空间,采用成组方式把文件存放到磁盘上,现用户要求每次读一个逻辑记录到他的工作区中,当对该逻辑记录处理后,要求把下一个逻辑记录读人到工作区,直到连续读出8个记录。

由于主存储器与外存之间的信息交换是以块为单位的,所以应当在主存中开辟一个2048字节的缓冲区。由上题可知每块中含有25个记录,文件一共占用4块,用户在指明了要读人记录的记录号N后,根据公式[N/25」得到该记录应该在4块的哪一块中,将该块读人到缓冲区中,并根据公式(N/25)得到该记录是块内的第几个记录(式中[]表示取整,()表示取余数),将该记录从缓冲区拷贝到用户工作区,然后 N+l,如果[(N+1)/25]=[N/25],则将缓冲区中的下一个记录拷贝到用户工作区,而无需启动 I/O操作,否则,按照新的块号将一个数据块读进缓冲区,并将记录从缓冲区拷贝到用户工作区。

10、简要说明文件保护与文件保密的定义域区别。

（1）文件的保护 文件的保护是指防止文件被破坏。造成文件可能被破坏的原因有时是硬件故障、软件失误引起的，有时是由于共享文件时引起错误，应根据不同的情况采用不同的保护措施。 ①防止系统故障造成的破坏 为了防止各种意外破坏文件，可以采用建立副本和定时转储的方法来保护文件。 ②防止用户共享文件时造成的破坏 为了防止不同用户使用文件时破坏文件，可规定各用户对文件的使用权限。例如：只读、读/写、执行、不能删除等。对多用户可共享的文件采用树形目录结构，能得到某级目录权限就可得到该级目录所属的全部目录和文件，按规定的存取权限去使用目录或文件。

（2）文件的保密 文件的保密是指防止他人窃取文件。“口令”和“密码”是两种常见的方

法。一旦为文件在目录中设置口令后，文件使用者必须提供口令，只有提供的口令与设置的口令一致时才可使用该文件，否则无法使用。“密码”是把文件信息翻译成密码形式保存，使用时再解密。密码的编码方式只限文件主及允许使用该文件的用户知道，但这种方法增加了文件编码和译码的开销。

11、当用户想要读取一个文件时，需要怎么操作才能正确完成读取过程，为什么需要这些操作？

若要读取一个文件，首先必须要打开指定文件，打开一个特定文件接下来，我们来看看打开文件的具体过程：fd=fopen()是一个系统调用，用于根据文件名打开一个文件，返回该文件的文件描述符，文件打开后进程便可以根据文件描述符fd进行其他操作，比如读，写，关闭等操作。

12、虚拟文件系统的功能与优势是什么？

答：VFS的功能包括：纪录可用的文件系统的类型；将设备同对应的文件系统联系起来；处理一些面向文件的通用操作；涉及到针对文件系统的操作时，VFS把它们影射到与控制文件、目录以及inode相关的物理文件系统。

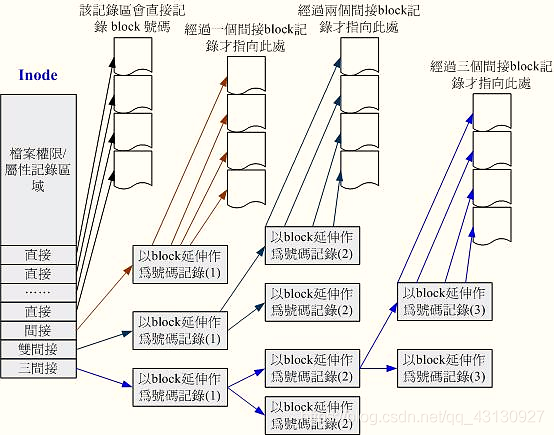
虚拟文件系统（VFS）是物理文件系统与服务之间的一个接口层，它对Linux的每个文件系统的所有细节进行抽象，使得不同的文件系统在[Linux](https://baike.baidu.com/item/Linux?fromModule=lemma_inlink" \t "_blank)核心以及系统中运行的其他进程看来，都是相同的。严格说来，VFS并不是一种实际的[文件系统](https://baike.baidu.com/item/%E6%96%87%E4%BB%B6%E7%B3%BB%E7%BB%9F/4827215?fromModule=lemma_inlink" \t "_blank)。它只存在于[内存](https://baike.baidu.com/item/%E5%86%85%E5%AD%98/103614?fromModule=lemma_inlink)中，不存在于任何外存空间。VFS在系统启动时建立，在系统关闭时消亡。

13、假定一个文件系统组织方式与MS-DOS相似，在FAT中可有64K个指针，磁盘的盘块大小为512B，试问该文件系统能否指引一个512MB的磁盘？

解：

512MB/512B=1M个盘块。每个盘快都应该要有一个指针,所以64K个指针不能支持一个512MB的磁盘。

14、在UNIX中，每个i节点中有10个直接地址和一、二、三级间接索引。若每个盘块512B，每个盘块地址4B，则一个1MB的文件分别占用多少间接盘块？25MB的文件呢？



* 10个直接盘块存放的容量为： 10\*512B = 5KB
* 一个盘块中可放的盘块地址数为：512B/4B = 128
* 一次间接索引存放的容量为：128\*512B = 64KB
* 二次间接索引存放的容量为：128\*128\*512B = 8192KB
* 三次间接索引存放的容量为：128\*128\*128\*512B = 1048576KB

1MB为1024KB，1024KB-64KB-5KB=955KB，955×1024B/512B=1910，所以1MB的文件分别占用128个一次间接盘块和1910个二次间接盘块。

25MB为：25×1024KB--64KB-5KB-8192KB=17339KB, 17339×1024B/512B=34678 所以25MB的文件分别占用128个一次间接盘块和128^2=16384个二次间接盘块， 34678个三次间接盘块.

15、假设某文件系统的硬盘空间为500MB，盘块大小为1KB，采用显示链接分配，请回答以下问题：

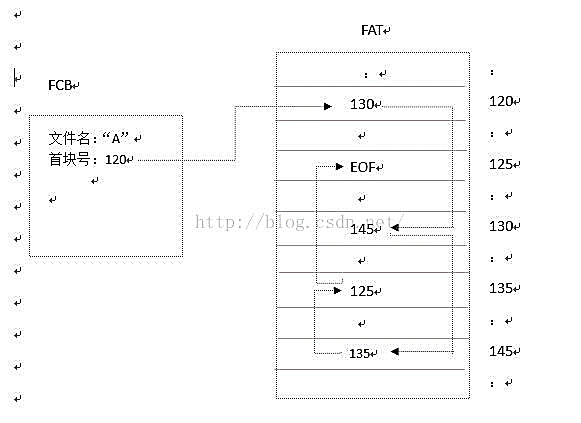
(1)其FAT表（文件分配表）需占用多少存储空间？

(2)如果文件A占用硬盘的盘块号依次为120、130、145、135、125共五个盘块，请画图示意文件A的FCB与FAT表的关系以及FAT表中各盘块间的链接情况。

解：

(1) 共500MB/1KB=500K个盘块，为保存最大的盘块号500K，该FAT表至少需要19位，扩展为半个字节的整数倍后，可知每个FAT表项需20位，即2.5字节。因此，FAT需占用的空间大小为： 2.5×500KB=1250KlB (取20位或24位皆可)

（2）



**第9章 IO管理 习题**

1. 分析并简述I/O设备与控制器之间的关系。

“I/O设备一般是由执行I/O操作的机械部分和执行控制I/O的电子部件组成。 通常将这两部分分开,执行I/O操作的机械部分就是一般的I/O设备,而执行控制I/O的电子部件则称为设备控制器或适配器(adapter)。

1. 简述中断控制方式与DMA方式进行内存与外设之间数据传输的基本原理，并对比其优缺点。

要点：

所谓DMA方式就是在进行数据传送时，CPU让出总线的控制权，由硬件中的DMA控制器接管直接控制总线。在DMA.控制器的控制下，数据不经CPU而直接在内存和外设之间传送，提高大批量数据交换的速度，从而提高计算机系统的数据传输效率。在传送过程中，是直接依靠硬件在主存和I/O设备之间传送数据，传送期间不需要CPU程序干预。主要用于几个方面：

● 用于磁盘等高速外存的数据决传送；

● 用于高速通信设备的数据帧传送；

● 用于高速数据采集；

● 用于动态存储器别新。

DMA方式与中断方式相比较，都具有随机性，都能实现主机与I/O在一段时间内的并行操作但是DMA方式必须通过硬件完成高速数据传送，而简单的批量数据传送中断方式则通过服务程序完成中、低速I/O传送，并能处理复杂的随机事态。而且，对于中断I/O来说，它是以字（节）为单位进行的I/O的，每完成一个字（节）的I/O，控制器便要向CPU请求一次中断。换言之，采用中断驱动I/O方式时的CPU，是以字（节）为单位进行千预的。如果将这种方式用于块设备的I/O，是极其低效的。

1. 为什么希望用双缓冲区而不是单缓冲区来提高I/O的性能？

系统为用户进程设置两个缓冲区。以输入操作为例，输入设备先将数据输入第一个缓冲区，第一个缓冲区满后，转向第二个缓冲区。此时操作系统可以从第一个缓冲区中取出数据送到用户进程，对数据进行处理。采用双缓冲可以减少或消除用户进程的等待时间，提高系统效率。

1. 为什么要引入缓冲技术？设置软件缓冲区的原则是什么？

其主要原因可归结为以下几点：

（1）缓解设备之间速度差异的矛盾。

（2）缓冲区可以缓解设备之间传输数据大小不一致的矛盾

（3）支持应用程序I/O的复制语义

设置缓冲区的原则是:如果数据到达率与离去率相差很大, 则可采用单缓冲方式; 如果 信息的输入和输出速率相同 (或相差不大) 时, 则可用双缓冲区; 对于阵发性的输入、 输出, 可以设立多个缓冲区。

1. 设备驱动程序的主要功能是什么？它在系统中处于什么样的地位？

答:设备驱动程序的功能主要有:接受用户的I/O请求;取出 请求队列中队首请求,将相应设备分配给它;启动该设备工作, 完成指定的I/O操作;处理来自设备的中断。

设备驱动程序在系统中处于核心空间,位于设备控制器的上 层,目的是对核心I/O子系 统隐藏各个设备控制器的差别。

1. 请解释什么是设备独立性原则？

设备独立性，即[应用程序](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F/5985445?fromModule=lemma_inlink)独立于具体使用的[物理设备](https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E8%AE%BE%E5%A4%87/2692368?fromModule=lemma_inlink)。为了实现设备独立性而引入了逻辑设备和物理设备这两个概念。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。因此，系统须具有将逻辑设备名称转换为某物理设备名称的功能，这非常类似于[存储器管理](https://baike.baidu.com/item/%E5%AD%98%E5%82%A8%E5%99%A8%E7%AE%A1%E7%90%86/6627083?fromModule=lemma_inlink)中所介绍的[逻辑地址](https://baike.baidu.com/item/%E9%80%BB%E8%BE%91%E5%9C%B0%E5%9D%80/3283849?fromModule=lemma_inlink)和[物理地址](https://baike.baidu.com/item/%E7%89%A9%E7%90%86%E5%9C%B0%E5%9D%80/2129?fromModule=lemma_inlink)的概念。

1. 什么是虚拟设备，SPOOLing系统的功能与原理是什么？

当系统只有一台输入设备或一台输出设备的情况下，可允许两个以上的作业并行执行，并且每个作业都感觉到获得了供自己独占使用的输入设备和输出设备，操作系统采用的这种技术为用户提供了虚拟设备。 SPOOLing技术借助磁盘和通道实现了输入/输出过程的共享。当用户提出输入/输出请求时，系统及时响应，此时用户会认为已独占输入/输出设备；但事实上，有多道作业同时进入该过程，并分别占用各个阶段。可假设如下情况：第一道作业提出打印申请，得到响应后正在打印机上输出；此时第二道作业提出输出请求，系统响应后将其送入磁盘输出井，一旦第一道作业打印结束，第二道作业可马上开始打印；接着第三道、第四道作业也源源不断地提出输出请求并得到响应，先后进入输出井及占用打印机。若系统控制得好，可令整个过程被数道作业共享，而每一个进入输出过程的作业都会认为自己在独占打印机。可以说，该系统向用户提供了多台打印机。8、设备分配的原则是什么？

9、I/O软件分为多个层次，请说明以下各工作是在哪一层完成的？

1. 向设备存储器写命令
2. 检查用户是否有权使用设备
3. 将二进制整数转换成ASCII码以便打印。

解答:I/O软件一般分为用户层软件、设备独立性软件、设备驱动程序和中断处理程序。

(1)向设备寄存器写命令是在设备驱动程序中完成的

(2)检查用户是否有权使用设备属于设备保护,因此是在设备独立性软件中完成的。

(3)将二进制整数转换成ASCII码的格式打印是通过I/O库函数(如C的printf()中就有打印格式的控制字符串)完成的,因此属于用户层软件。

10、内部中断和外部中断产生的原因是什么？中断处理的主要流程是什么？

在微机系统中，对于外部中断，中断请求信号是由外部设备产生，并施加到CPU的NMI或INTR引脚上，CPU通过不断地检测NMI和INTR引脚信号来识 别是否有中断请求发生。对于内部中断，中断请求方式不需要外部施加信号激发，而是通过内部中断控制逻辑去调用。无论是外部中断还是内部中断，中断处理过程 都要经历以下步骤：　请求中断→响应中断→关闭中断→保留断点→中断源识别→保护现场→中断服务子程序→恢复现场→中断返回。

11、Linux系统中把一个中断处理过程分为哪几部分？为什么采用这种技术？

linux中断分为：上半部（顶半部）与下半部（底半部）

上半部：上半部就是中断处理函数，那些处理过程比较快，不会占用很长时间的处理就可以放在上半部完成。

下半部：如果中断处理过程比较耗时，那么就将这些比较耗时的代码提出来，交给下半部去执行，这样中断处理函数就会快进快出。

目的：

Linux 内核将中断分为上半部和下半部的主要目的就是实现中断处理函数的快进快

出，那些对时间敏感、执行速度快的操作可以放到中断处理函数中，也就是上半部。剩下的所有工作都可以放到下半部去执行，比如在上半部将数据拷贝到内存中，关于数据的具体处理就可以放到下半部去执行。

**第10章 大容量存储器 习题**

1. 进程读取硬盘数据所耗费的时间由哪几部分构成？

①寻道时间,启动磁臂的时间s与磁头移动n条磁道所花费的时间之和 。

②旋转延迟时间,是指定扇区移动到磁头下面所经历的时间。

③传输时间 ,指把数据从磁盘读出或向磁盘写入数据所经历的时间。

1. 现有一个6400块的文件，块的大小与扇区相同且从头开始存放。假设柱面和文件块都是从1开始排序。试问该文件的第3681块应在哪个柱面第几道的第几扇区。第79柱面7磁道7扇区存放的是文件的第几块？

解：

因为文件块的编号从1开始编号，所以要先减去一

(3681-1)/(8\*8)=57……这是柱面号（从1开始的话要加一，即58）  
(3681-1)%(8\*8)/8=4……这是盘面号（从1开始的话要加一，即5）  
(3681-1)/%(8\*8)%8=0……这是扇区号（从1开始的话要加一，即1）  
所以该文件的第3681块应在58柱面第5盘面的第1扇区  
  
“第79柱面7磁道7扇区存放的是文件的第几块”这句话是错误的，柱面号和磁道号为同一概念，所以应修改为“第79盘面7磁道7扇区存放的是文件的第几块”

根据上述规则，第79盘面7磁道7扇区应转换为78、6、6

6\*(8\*8)+78\*8+6+1=1015

第79盘面7磁道7扇区存放的是文件的第1015块

3、假设计算机系统采用CSCAN（循环扫描）磁盘调度策略，使用2KB的内存空间记录16384个磁盘块的空闲状态。

（1）请说明在上述条件下如何进行磁盘块空闲状态管理。

（2）设某单面磁盘旋转速度为每分钟6000转。每个磁道有100个扇区，相邻磁道间的平均移动时间为1ms。若在某时刻，磁头位于100号磁道处，并沿着磁道号大的方向移动（如下图所示），磁道号请求队列为50、90、30、120，对请求队列中的每个磁道需读取1个随机分布的扇区，则读完这4个扇区点共需要多少时间？要求给出计算过程。

（3）如果将磁盘替换为随机访问的Flash半导体存储器（如U盘、SSD等），是否有比CSCAN更有效的磁盘调度策略？若有，给出磁盘调度策略的名称并说明理由；若无，说明理由。

解:

(1)采用位示图法管理磁盘空闲块,每一位表示一个磁盘块的状态,共需要16384/32=512 个字节即2KB的空间,正好可放在系统提供的内存中。

(2)采用CSCAN调度算法,访问磁道的顺序和移动的磁道数如下表所示:

故移动的磁道数为20+90+20+40=170,所花的时间为170ms。由于转速为6000r/min,则平均旋转延迟为5ms,总的旋转延迟时间为20ms,读取一个扇区的平均时间为0.1ms,故读取4个扇区所花的时间为0.4ms。综上所述,读取磁道上所有扇区所花的总时间为170+5\*4+0.4=190.4ms。

(3)采用先来先服务(FCFS)调度策略更高效,因为Flash半导体存储器的物理结构不需要考虑寻道时间和旋转延迟,可直接按I/O请求的先后顺序服务。

4、某文件系统空间的最大容量为4TB，以磁盘块为基本分配单位，磁盘块大小为1KB。文件控制块（FCB）包含一个512B的索引表区。请回答下列问题：

（1）假设索引表区仅采用直接索引结构，索引表区存放文件占用的磁盘块号。索引表项中块号最少占多少字节？可支持的单个文件最大长度是多少字节？

（2）假设索引表区采用如下结构：第0~7字节采用 <起始块号，块数> 格式表示文件创建时预分配的连续存储空间，其中起始块号占6B，块数占2B；剩余504字节采用直接索引结构，一个索引项占6B，则可支持的单个文件最大长度是多少字节？为了使单个文件的长度达到最大，请指出起始块号和块数分别所占字节数的合理值并说明理由。

解：

遇到这种类型的题目，首先要求出系统的总块数，因为总大小为4TB.而ITB=2B，所以总大小为（4╳2）B=2B。每块大小为1KB.即2B。由此可以得出文件系统，存储空间的总块数为2/2=232块。

要表示这么多块，块号需要32个二进制位、即32/8B=4B，因此索引表项中的块号最少占4B。每个索引区为512B.可以存放128个索引表项，而1第一题采用直接索引结构，即每个索引表项指向个磁盘块。所以，最大文件长度为128╳1KB=128KB由此可以得到1）的答案：

文件系统存储空间共有块数=（4╳2）/2=2。为表示2个块号，索引表项占32/8B=4B。

512可存放2个索引表项，故最大文件长度=2╳2B=2B=128KB。接下来分析2）题，题中给出的条件有两个：①0~7B采用另种格式后面内分析。②剩余504B采用直接索引结构且索引项占6B。条件②比较简单，先予以解决。

剩余的504B可以表示504B/6B=84个块（内为是直接索引），所以这部分能够表示的大小为84KB。再看前面的8个字节，其中6B为起始块号，2B为块数，由于块的大小不变，

因此起始块号位数并无影响，只关注这部分能表示多少块即可。2B共有2╳8=16个：进制位，即可以表示2个块，因此这部分能够表示的大小为2KB=64MB。相加即可得到这种方法可支持的单个文件最大长度。

要改变表示的文件大小最大值，只能从前8个字节入手（因为后面的504字节只能固定表示84KB），而根据之前的分析，块大小不变，只需要让块数增加就可以扩充文件大小，于是可以将8字节中更多的字节表示块数。因为4字节就是32位，足够表示空间内所有的块（一共就2块），所以只要有4B表示块数就足够，因此得到结论：起始块号占4B.块数占4B即可使文件长度达到最大。

于是根据刚才的分析可以得到2）的答案：块号占6B，块数占2B的情况下，最大文件长度=（2╳2+（504/6）╳2）B=64MB+84KB.合理的起始块号和块数所占字节数分别为4、4（1、7或2、6或3、5），因为块数占4B或以上，就可以表示4TB人小的文件长度，达到文件系统的空间1：无限。

5、系统中磁头停留在磁道号为100的磁道上，这时先后有4个进程提出了磁盘访问请求，要访问磁盘的磁道号按申请到达的先后顺序依次为：55、120、39、110。移动臂的运动方向：沿磁道号递减的方向移动。若分别采用FCFS磁盘调度算法、SSTF算法、SCAN算法时，所需寻道长度分别为多少？

解：

FCFS算法：100-55+ 120-55+ 120-39+ 110-39 = 262

SSTF算法：110-100 +120-110 + 120-55 + 55-39 = 101

SCAN算法： 100-55+ 55-39 +110-39 + 120-110 = 142

1. 某操作系统的磁盘文件空间共有500块，若用字长为32位的位示图管理盘块空间，试问：

(1)位示图需要多少个字？

(2)如果b（盘块号）,i，j从1开始计数，第i字第j位对应的块号是多少？

(3)如果b（盘块号）,i，j 从0开始计数，第i字第j位对应的块号是多少？

**（1）位示图需要的字数计算：INT（500/32）=16个字。**

**（2）块号b=(i-1)\*32+j**

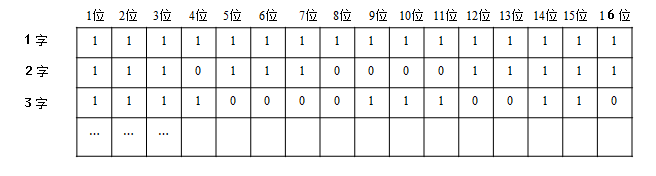
**（3）块号b=i\*32+j**

7、有3200个磁盘块可用来存储信息，如果用字长为16位的字来构造位示图，若位示图部分内容如图所示。

(1)位示图共需多少个字？

(2)若某文件长度为3200B，每个盘块为512个字节，采用链接结构且盘块从1开始计数，系统将为其分配哪些磁盘块？试具体说明分配过程。

(3)若要释放磁盘的第300块，应如何处理？



解： 1）3200/16=200

2）3200/512 =7 ； 分配的磁盘块为20、24、25、26、27、37和38

分配过程为：顺序检索位示图，从中找到第一个值为0的二进制位，得到行号i=2，列号j=4 ;计算出找到的第一个盘块是 B1 = （2-1)\*16+4=20

第二个值为0的二进制位，得到行号i=2，列号j=8. ;计算出找到的第一个盘块是

B2=(2-1)\*16+8=24

………….

修改位示图，令Map[2,4]=Map[2,8] = …..=1,并将对应块20、24……. 分配出去。

3）计算出磁盘第300块所对应的二进制位的行号i和列号j :

i = (300-1)/16 + 1 = 19 j = (300-1)MOD16 + 1= 12 修改位示图，令Map[19，12]=0， 表示对应块为空闲块

8、有一计算机系统利用下图所示的位示图（行号、列号都从0开始编号）来管理空闲盘块。如果盘块从1开始编号，每个盘块的大小为1KB。

(1) 现要为文件分配两个盘块，试具体说明分配过程。

(2) 若要释放磁盘的第300块，应如何处理？

（编号从0开始）

答:(1)为某文件分配两个盘块的过程如下:

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1

1 1 1 1 1 1 0 1 1 1 1 0 1 1 1 1

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

第一、顺序检索位示图,从中找到第一个值为0的二进制位,得到其行号为i1=2,列号为j1=2;第二个值为0的二进制位,得到其行号为i2=3,列号j2=6。

第二、计算出找到的两个空闲块的盘块号分别分:

B1=i1\*16+j1+1=2\*16+2+1=35

B2=i2\*16+j2+1=3\*16+6+1=55

第三、修改位示图,令map[2,2]=map[3,6]=1,并将对应块35、55分配出去。

(2)释放磁盘的第300块时,应进行如下处理:

第一、计算出磁盘第300块所对应的二进制位的行号i和列号j:

i=(300-1)/16=18 j=(300-1)%16=11

第二、修改位示图,令:map[18,11]=0,表示对应块为空闲块。