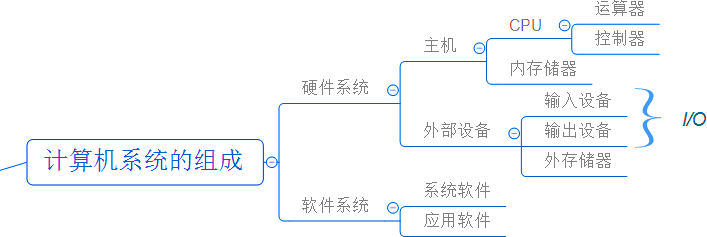
# 2018-2019（1）计算机硬件与系统软件串讲

### 1、所有概念

**2、计算机系统的基本组成（P1）**



基本的计算机硬件系统由运算器、控制器、存储器、输入设备和输出设备五大部件组成。

* **运算器**是对数据进行加工处理的部件，它主要完成数据和逻辑运算。
* **控制器**的主要功能则是从主存中取出指令并进行分析，控制计算机的各个部件有条不紊地完成指令的功能。
* **储存器**是计算机中的记忆设备，分为内部储存和外部储存。内存空间小，速度快，一般用来临时存放计算机运行时所需的程序、数据及中间结果。外存（硬盘、软盘、光盘、U盘等）空间小、速度慢，可用于长期存储信息。寄存器是 CPU 中的记忆设备，用来临时存放指令、数据及运算结果。与内存储器相比，寄存器的速度要快的多。

### 3、进位计数制及其转换（P3）

### 二进制

* 加法：逢二进一
* 减法：借一当二
* 乘法：与十进制一样

八进制：逢八进一

十进制：逢十进一

十六进制：逢十六进一

### 2——10 转换

2->10：将每一位的数乘以它的权再全部加起来；

10->2：将十进制的数写成**二进制数权的多项式**，从高到低的系数就是二进制。

### 2——8 转换

2-->8：将二进制数从小数点处开始，每三位一组，每组对应其八进制的数，按顺序排列即可；

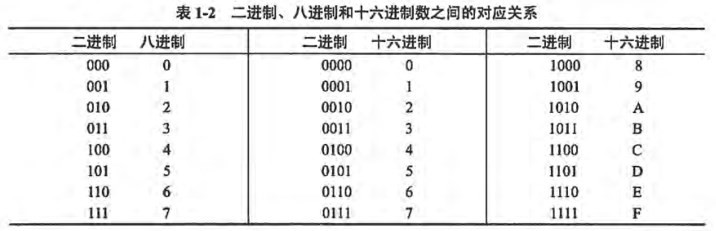
8-->2：每个八进制数字都对应其二进制数组，按顺序排列即是二进制数。

### 2——16 转换：

2-->16：将二进制数从小数点处开始，每四位一组，每组对应其十六进制的数，按顺序排列即可；

16-->2：每个十六进制数字都对应其二进制数组，按顺序排列即是二进制数。

其他：先转换为二进制再转



### 4、码制的表示及其相关问题（P5）

### 机器数：各种数据在计算机中表示的形式称为机器数。二进制，小数点隐藏不占位置，机器数对应的实际数值称为数的真值。

**原码**

最高位为符号位，1 为负，0 为正，其余的 n-1 位为数的绝对值。

**反码**

正数与原码一样，负数为原码按位求反。

**补码**

正数与原码一样，负数为反码按位求反。

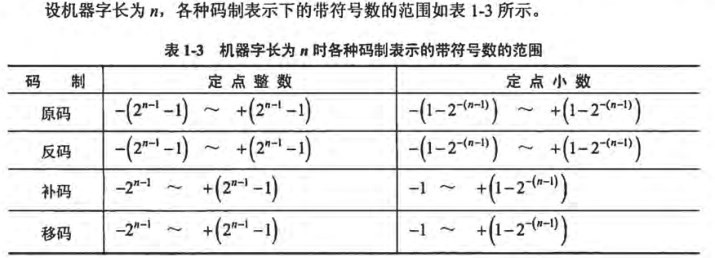
**移码**

是在数 X 上加一个偏移量来定义的，常用于表示浮点数中的阶码。

如果机器字长为n，在偏移量为 时，只要将补码的符号位取反即可。

### 5、定点数与浮点数的概念用途（P7）

**（1）定点数**

所谓定点数，就是表示数据时小数点的位置固定不变。小数点的位置通常有两种约定方式：定点整数（纯整数，小数点在最低有效数值位之后）和定点小数（纯小数，小数点在最高有效数值位之前）。

**（2）浮点数**

当机器字长为n 时，定点数的补码和移码可表示个数，而其原码和反码只能表示 个数（0 表示占用了两个编码），因此，浮点数是小数点位置不固定的数，它能表示更大范围的数。

在十进制中，一个数可以写成多种表示形式。例如。83.125 可写成或

等。同样，一个二进制数也可以写成多种表示形式。例如，二进制数

1011.10101 可以写成



其中E 称为阶码，F 为尾数，这种表示数的方法称为浮点表示法。

在浮点表示法中，阶码通常为带符号的纯整数，尾数为带符号的纯小数。浮点数的表示格式如下：

很明显，一个数的浮点表示不是唯一的。当小数点的位置改变时，阶码也相应改变，因此可以用多种浮点数形式表示同一个数。

浮点数所能表示的数值范围主要由阶码决定，所表示数值的精度则由尾数决定。为了充分利用尾数来表示更多的有效数字，通常采用规格化浮点数。规格化就是将尾数的绝对值限定在区间[0.5,1]。当尾数用补码表示时，需要注意如下问题：

A、若尾数 M>=0，则其规格化的尾数形式为 M=0.1XXXX X,其中 X 可为 0，也可为 1，即将

尾数限定在区间[0.5,1]。

B、若尾数 M<=0，则其规格化的尾数形式为 M=1.0XXXX X,其中 X 可为 0，也可为 1，即将

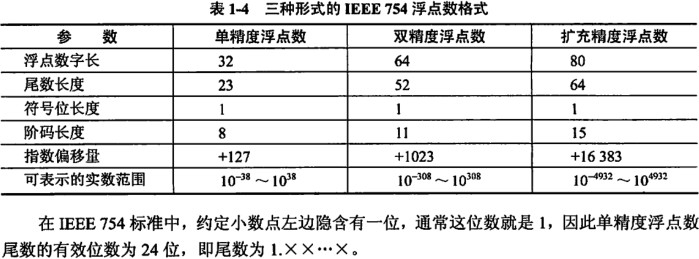
尾数 M 限定在区间[-1，,-0.5]。

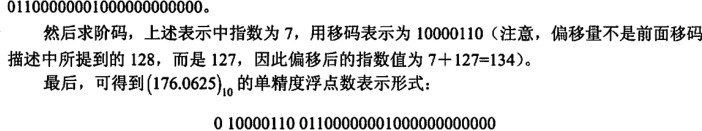
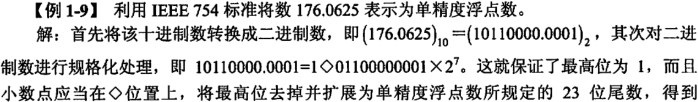
**（3）工业标准 IEEE 754**

IEEE 754 是由 IEEE 制定的有关浮点数的工业标准，被广泛采用。该标准的表示形式如下：

其中，S 为数的符号位，为 0 时表示正数，为 1 时表示负数；P 为阶码，通常用移码表示；

M 为尾数，用原码表示。

目前，计算机中主要使用三种形式的 IEEE 754 浮点数，如表 1-4 所示

****

**6、汉字编码的类型**

①输入码：数字编码、拼音码、字形码

②内部码

③字形码

**7、校验码的类型、海明码、模 2 加减计算**

校验码：奇偶校验码、海明吗、循环冗余检验码

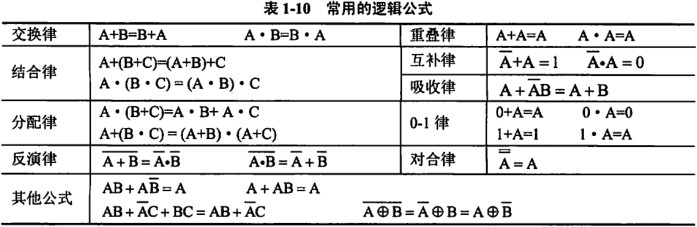
**8、与或非及逻辑表达式的化简（P17）**

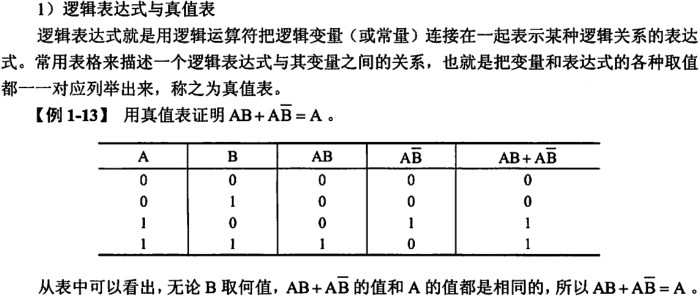
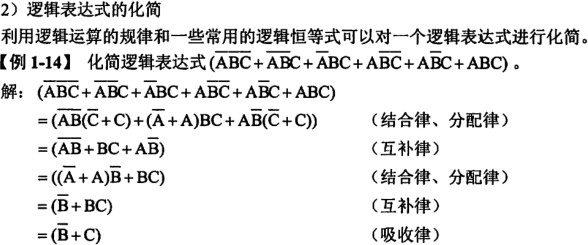
（1）“与”运算：又叫逻辑乘，运算符号常用 AND、∩、∧或“· ”表示。当且仅当A 和B 都为“真”时，A“与”B 为真。

（2）“或”运算：又称逻辑加，运算符号常用 OR、∩、∨、或“＋”表示。当且仅当 A、B都为“假”时，A“或”B 为假。

（3）“非”运算：也称逻辑求反运算，常用 A’表示对变量A 的值求反。其运算规则很简单：1’=0，0’=1。

（4）“异或”运算：又称半加运算，其运算符号常用XOR 或⊕表示。当且仅当A、B 值不同时，A“异或”B 为真。A⊕B=A’·B+A·B’。

常用的逻辑公式：



**9、机器数的加减运算及浮点运算的步骤（P18）**

**1、机器数的加减运算：**在计算机中，可以只设置加法器，将减法运算转化为加法运算来实现。

**（1）原码加、减法**

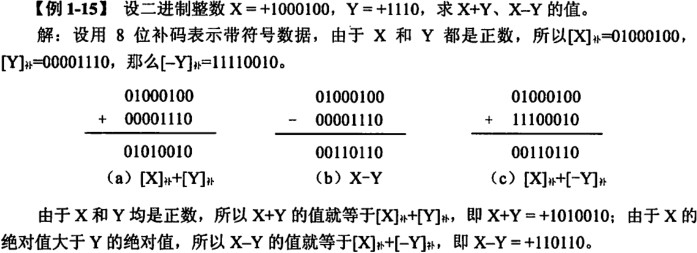
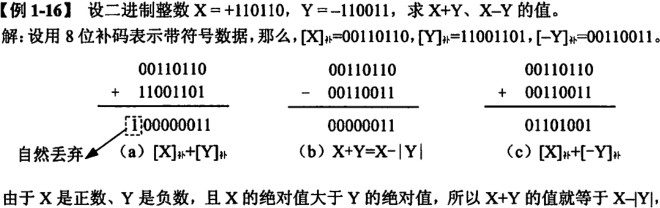
当两个相同符号的原码数相加时，只需将数值部分直接相加，运算结果的符号与两个加数的符号相同。若两个加数的符号相异，则应进行减法运算。其方法是：先比较两个数绝对值的大小，然后用绝对值大的绝对值减去绝对值小的绝对值，结果的符号取绝对值大的符号。因此，原码表示的机器数进行减法是很麻烦的，所以在计算机中很少被采用。

**（2）补码加、减法**

①补码加法的运算法则是：和的补码等于补码求和，即[X+Y]补=[X]补+[Y]补。

②补码减法的方法是：差的补码等于被减数的补码加上减数取负后的补码。因此，在补码表示中，可将减法运算转换为加法运算，即[X-Y]补=[X]补+[-Y]补。

③由[X]求[-X]的方法是：[X]的各位取反（包括符号位），末尾加 1。



因此，补码加减运算的规则如下。

①参加运算的操作数用补码表示。

②符号位参加运算，则两个数的补码直接相加；若进行相减运算，则将减数连同符号位一起变反后与被减数相加。

③运算结果用补码表示。

与原码减运算相比，补码减运算的过程要简便得多。在补码加减运算中，符号位和数值位一样参加运算，无须作特殊处理。因此，多数计算机都采用补码加减运算。

**（3）溢出及判定**

在确定了运算的字长和数据的表达方法后，数据的范围也就确定了。一旦运算结果超出所能表示的数据范围，就会发射溢出。发生溢出时，运算结果肯定是错误的。

只有当两个同符号的数相加（或者是相异符号数相减）时，运算结果才有可能溢出。

**2、浮点运算**

浮点加减运算

设有浮点数 的运算过程如下。

**（1）对阶，使两个数的阶码相同**。令 把阶码小的数的尾数右移 K 位，使其阶码加上 K。

**（2）求尾数和（差）。**

**（3）结果规范化并判断溢出。**若与那算结果所得的尾数不是规范化的数，则需要进行规范化处理。当尾数溢出时，需要调整阶码。

**（4）舍入。**在对结果右规使，尾数的最低位将因移出而丢掉。另外，在对阶过程中也会将尾数右移使最低位丢掉。这就需要进行舍入处理，以求得最小的运算误差。舍入处理的方法如下。

①截断法。将要保存的数据末味右边的数据全部截去，不管数据是 0 还是 1。

②末位恒 1 法。将要保留的末位数据恒置 1，不管右移丢掉的数据是 0 还是 1。

③0 舍 1 法。舍去的数据为 0 时，保持末位原始状态。若舍去的数据为 1，则将末位

加 1.

**（5）溢出判别。**以阶码为准。若阶码溢出（超出最大值），则运算结果溢出；若阶码下溢（小于最小值），则结果为 0，否则结果正确无溢出。

**10、计算机 CPU 的组成及执行指令的步骤**

CPU主要由运算器、控制器（Control Unit，CU）、寄存器组和内部总线组成。

执行指令的步骤：①取指令；②指令译码；③按指令操作码执行；④形成下一条指令地址。

**11、存储系统的类型和层次结构**

存储器的分类：

①按存储器所处的位置，分为内存和外存（辅存）。

②按材料分为磁存储器、半导体存储器和光存储器。

③按工作方式可分为读写存储器和只读存储器。

④按访问方式可分为地址访问存储器与内容访问存储器。

⑤按寻址方式可分为随机存储器、顺序存储器以及直接存储器。

存储系统的层次结构：

一般情况下，可用三级结构进行描述：高速缓存、主存储器与辅助存储器。内存容量：bit、byte、KB、MB、GB、TB、PB、EB、ZB、YB、NB、DB

**12、IO 接口的功能与分类**

功能：选址功能、传送命令的功能、传送数据的功能、反映设备状态的功能。

接口类型：

①按照数据传送方式分类：并行接口、串行接口

②按照功能选择灵活性分类：可编程接口与不可编程借口

③按照时序控制方式分类：同步接口与异步接口

④按照数据传送的控制方式分类：中断借口以及 DMA 接口

**13、CPU 与外设之间交换数据的方式**

① 接程序控制 ②中断方式 ③直接存储器存取方法 ④通道控制方法

**14、指令种类、格式与寻址方式**

**指令种类：**①数据传送指令 ②IO 指令 ③算术运算指令 ④逻辑运算指令 ⑤移位操作指令

⑥程序控制指令 ⑦串操作类指令 ⑧处理机控制类指令 ⑨数据转换类指令

**寻址方式：**①立即寻址 ②直接寻址 ③寄存器寻址 ④寄存器间接寻址 ⑤间接寻址 ⑥基址寻址 ⑦变址寻址

**15、操作系统的特征功能类型**

（1）四个**特征**：并发性、共享性、虚拟性和不确定性

（2）五个**功能**：进程管理（处理机管理）、文件管理、存储管理、设备管理和作业管理

（3）七种**类型**：

* 批处理操作系统（单道：一次只有一个作业装入内存执行；多道：多个作业装入内存执行，3个特点是多道、宏观上并行、微观上串行运行）
* 分时操作系统（多用户的通用系统。4个特点：多路性、独立性、交互性和及时性）
* 实时操作系统（专用系统。实时控制系统：数据自动采集、飞机自动驾驶等；实时信息处理系统：飞机订票系统、情报检索系统等）
* 网络操作系统（共享网络资源）
* 分布式操作系统（是网络操作系统的更高级形式+透明性、可靠性、高性能。eg. 谷歌搜索）
* 微型计算机操作系统（eg. Windows、Mac OS、Linux）
* 嵌入式操作系统（运行在嵌入式智能芯片环境中，5个特点：微型化、可定制、实时性、可靠性、易移植性）

**16、程序与进程的不同（书本没有）**

①进程是程序的一次执行；

②一个进程可以执行一个或几个程序；

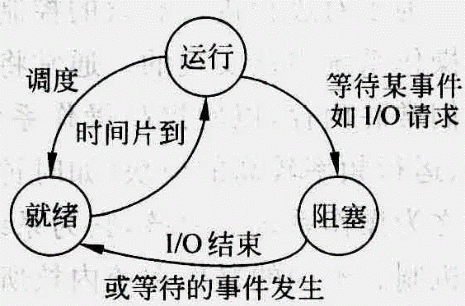
③程序可以作为软件资源长期保留，进程具有生命期

④进程具有并发性，程序不具备这种特征。

⑤进程是一个独立运行的单位，也是系统进行资源分配和调度的一个独立单位，进程具有独立性。

**17、进程的状态及其转换**

* **三态模型**

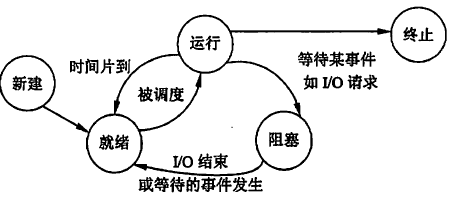


运行（Running）：当一个进程在处理机上运行时，称该进程处于运行状态。显然，对于单处理机系统，处于运行状态的进程只有一个。

就绪（Ready）：一个进程获得了除处理机外的一切所需资源，一旦得到处理机即可运行，则称此进程处于就绪状态。

阻塞（Blocked）：也称等待或睡眠状态，一个进程正在等待某一事件发生（例如，请求IO而等待IO完成等）而暂时停止运行，这时即使把处理机分配给该进程，它也无法运行，故称该进程处于阻塞状态。

* **五态模型**



新建态：对应于进程刚刚被创建且没有被提交的状态，并等待系统完成创建进程的所有必要信息。

终止态：第一个阶段等待操作系统进行善后处理，第二个阶段释放内存。设置终止态的目的是防止系统进行善后处理时引起资源分配不当等问题。

* **处理机**包括中央处理器（CPU），主存储器，输入-输出接口，加接外围设备就构成完整的计算机系统。

**18、临界区管理原则和PV操作（必考）**

（1）临界区(Critical Section CS )是进程中对临界资源实施操作的那段程序。对互斥临界区管理的4条原则如下。 （一次只能供一个进程使用）

* **有空即进。**当无进程处于临界区时，允许进程进入临界区，并且只能在临界区运行有限的时间。
* **无空则等。**当有一个进程在临界区时，其他需要进入临界区的进程必须等待，以保证进程互斥地访问临界资源。
* **有限等待。**对要求访问临界资源的进程，应保证进程等待有限时间后进入临界区，以免陷入“饥饿”状态。
* **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。
* **（2）PV操作**

信号量是一个整形变量，根据控制对象的不同赋不同的值。信号量可分为公用信号量和私用信号量两类。

* 公用信号量：实现进程间的互斥，初值=1或资源的数目
* 私用信号量：实现进程间的同步，初值=0或某个整数

信号量S的物理意义：

* S>=0时表示某资源的可用数
* s<0时其绝对值表示阻塞队列中等待该资源的进程数。

PV操作是实现进程同步与互斥的常用方法。

* **P操作**的定义：S=S-1，若S>=0，则执行P操作的进程继续执行；若S<0，则置该进程为阻塞状态，并将其插入阻塞队列。

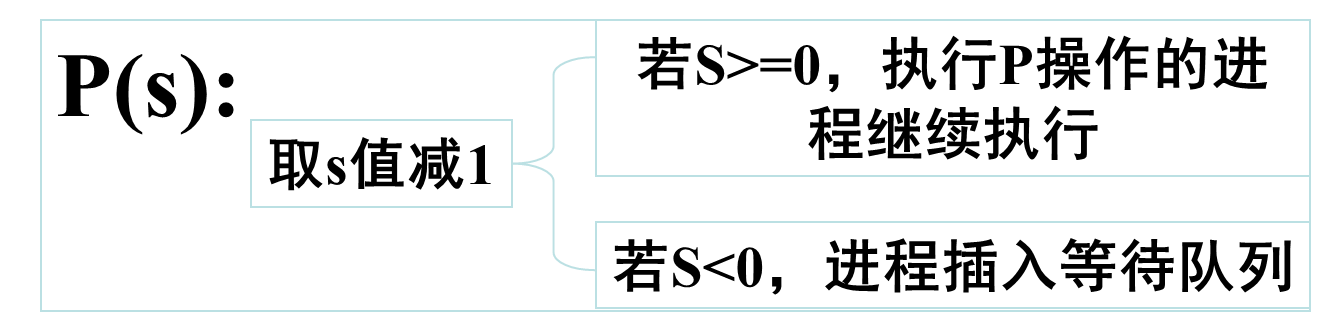
**Procedure P（Var S：Semaphore）； // P操作表示申请一个资源**

**Begin**

**S:=S-1；**

**If S<0 then  W(S)   { 执行P操作的进程插入等待队列 }**

**End；**



* **V操作**定义：S=S+1，若S>0则执行V操作的进程继续执行；若S<0，则从阻塞状态唤醒一个进程，并将其插入就绪队列，执行V操作的进程继续执行。

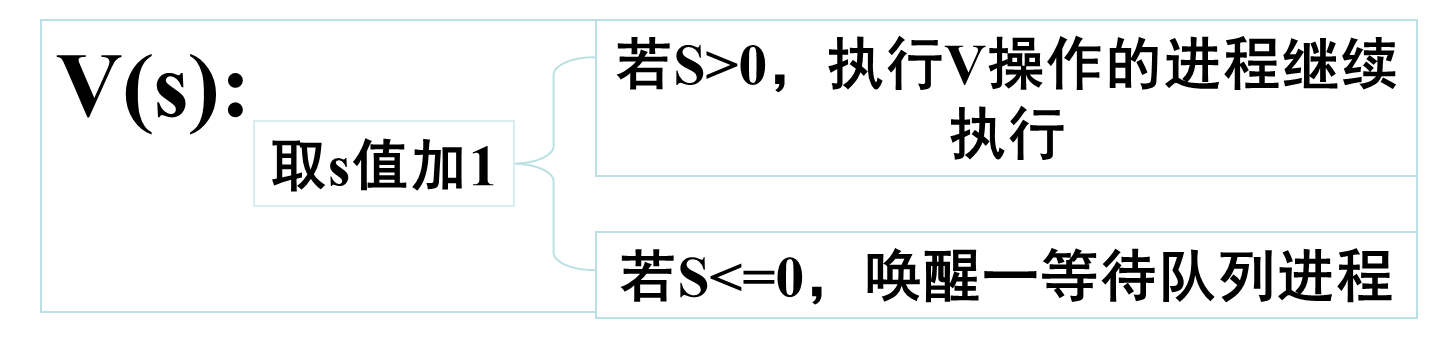
**Procedure V（Var S：Semaphore）； // V操作表示释放一个资源**

**Begin**

**S:=S+1；**

**If S<=0 then R(S) { 从阻塞队列中唤醒一个进程 }**

**End；**

****

**（3）利用PV操作实现进程的互斥**

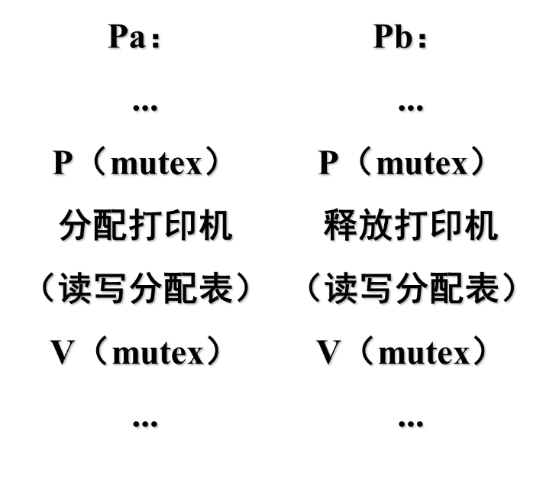
令信号量mutex的初值为1，当进入临界区时执行P操作，退出临界区时执行V操作。这样利用PV操作实现进程互斥的代码段如下：

P（mutex）

　 临界区

V（mutex）

**例：**打印机分配，互斥信号量mutex（初值为1），Pa为分配进程，Pb为释放进程。



**（4）利用PV操作实现进程的同步**

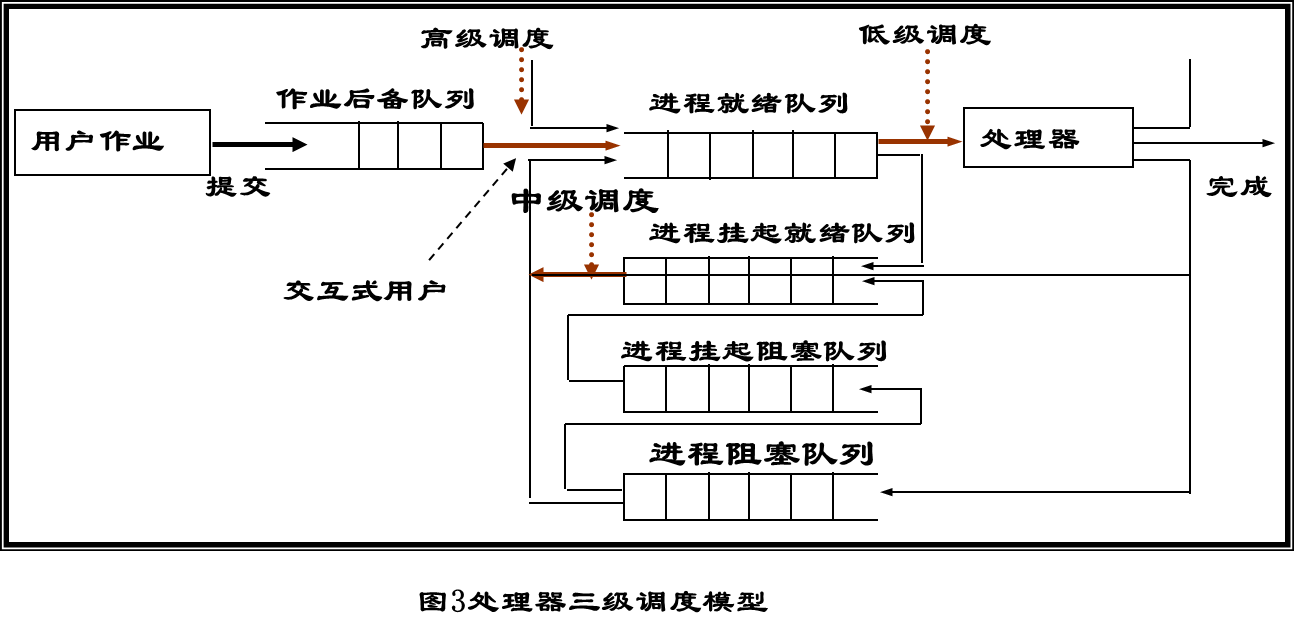
**例：**供者和用者对单缓冲区的同步。信号量：S1——缓冲区空否（初值为1），S2——缓冲区满否（初值为0）

****

**19、进程三级调度与进程调度算法**

（1）三级调度

* **高级调度。**又称“长调度”、“作业调度”，它决定处于输入池中的哪个后备作业可以调入主系统做好运行的准备，成为一个或一组就绪进程。系统中一个作业只需经过一次高级调度。
* **中级调度。**又称“中程调度”、“对换调度”，它决定处于交换区中的就绪进程哪个可以调入内存，以便直接参与对CPU的竞争。
* **低级调度。**又称“短程调度”、“进程调度”，它决定处于内存中的就绪进程哪个可以占用CPU，是操作系统中最活跃、最重要的调度程序，对系统的影响很大。



（2）进程调度方式

指当有更高优先级的进程到来时**如何分配CPU**。分为可剥夺和不可剥夺两种。

（3）进程调度算法

* **先来先服务（First Come First Served，FCFS）**

FCFS按照作业提交或进程变为就绪状态的先后次序分配CPU，即每当进入进程调度时，总是将就绪队列队首的进程投入运行。FCFS调度法比较有利于长作业，有利于CPU繁忙的作业；而不利于I/O繁忙的作业。FCFS算法主要用于宏观调度。

* **时间片轮转**

主要用于微观调度。通过时间片轮转，提高进程并发性和响应时间特性，从而提高资源利用率。有固定时间片和可变时间片两种选择。

* **优先级调度**

让每一个进程都有一个优先数，系统在调度时总选择优先级高的占用CPU。优先级调度分为静态优先级和动态优先级。确定优先级的因素有进程类型（系统进程优先级较高）、对资源的需求（如对CPU和内存需求较少的进程优先级较高）和用户要求（如紧迫程度和付费多少）。

* **多级反馈调度**

多级反馈队列算法是时间片轮转算法和优先级算法的综合与发展。其优点是照顾短进程以提高系统吞吐量、缩短了平均周转时间；照顾I/O型进程以获得较好的I/O设备利用率和缩短响应时间；不必估计进程的执行时间，动态调节优先级。

* **20、死锁的原因及条件**

**原因：**竞争资源或非法的进程推进顺序

**4个必要条件：**

* 互斥条件
* 请求保持条件
* 不可剥夺条件
* 环路条件

**21、可变分区管理算法（P78）**

**（1）首次适应法**

首先将空闲分区按照地址递增的顺序组织成空闲分区链。为作业分配内存时，系统根据作业大小，从空闲分区链的第一个空闲分区开始查找，只要找到第一个满足作业大小的空闲分区，从该空闲分区中分割一部分分配给作业，另一部分仍作为空闲分区；如果空闲分区链全部查找完也不能满足作业要求，则系统不能为作业分配内存。

该分区分配算法首先利用内存中的低地址空闲分区，保留了高地址空闲分区。缺点是：A、系统每次都是从链首开始查找空闲分区，低地址段的大空闲分区被分配或被分割，剩下了小空闲分区或空闲分区“碎片”；B、系统每次从链首开始查找空闲分区，查找过程可能花费的时间长。

**（2）循环首次适应法**

与首次适应法相同，循环首次适应法也按照地址递增的顺序组织空闲分区链。为作业分配内存时，系统不是从空闲分区链的第一个空闲分区开始查找，而是从空闲分区链上，上次为作业分配分区后的位置开始查找，找到第一个满足作业大小的空闲分区，分割并分配该空闲分区。如果找到空闲分区链的链尾还没有找到，系统可以再从链首开始查找。

该分区分配算法虽然克服了首次适应算法的缺点，使得空闲分区的分布更加均匀，查找空闲分区所需要的时间更短，但是，空闲分区链中的小分区或“碎片”问题仍然不能解决。

**（3）最优适应法**

该算法扫描整个空闲分区链，从中挑选出一个满足进程要求的最小分区进行分配，避免分割大空闲分区，使得内存“碎片”更小。为了配合查找空闲分区，空闲分区链需要按照分区大小递增的顺序组织，每次查找时从链首的最小分区开始，直到找到第一个满足作业要求的分区为止。因此，被分配的空闲分区是大小最适合的分区。

该分区分配算法克服了首次适应算法和循环首次适应算法的缺点，是一种较优的分区分配算法。遗憾的是该算法由于找到的空闲分区是最小能够满足要求的分区，从得到的空闲分区中分割了分配给进程的部分后，剩余的空闲分区很小，这一部分很小的“碎片”，难以再次利用。

**（4）最坏适应法**

与最优适应法相反，最坏适应法是挑选满足作业要求的最大分区，使得分配的空闲分区分配给作业后剩下的部分比较大，能够再作为空闲分区进行分配。减少了内存中“碎片”的大小和个数。配合该分区分配算法，空闲分区链需要按照分区大小递减的顺序组织，每次从链首最大的分区开始分配，只要能够满足作业要求就分配。这样，为作业分配内存后，剩下的“碎片”最大。

该算法存在的问题是最后会导致系统缺乏较大的空闲分区。

**（5）快速适应法（书里没讲）**

设置多个空闲分区链，对经常用到的长度的空闲分区，设置单独的空闲分区链，每个空闲区链，按照空闲分区的长度进行归类。这样，系统中存在多个空闲分区链，存在多个空闲分区表。在系统中增加一个空闲分区管理索引表，该表的每一表项对应一个空闲分区表。由于空闲分区管理索引表有每个空闲分区链的长度范围和开始指针。为作业分配内存时，首先根据作业大小查找空闲分区管理索引表，得到空闲分区表的起始指针，然后再从空闲分区表相应的空闲分区链中为作业分配一个空闲分区。

该算法的优点是能够快速得到空闲分区，不会分割空闲分区，并能够保留大的空闲分区，对大的作业也不会产生内存“碎片”。该算法的缺点是回收分区较困难，算法复杂，系统的开销较大。

**22、页式存储管理原理（P80）**

* **（1）分页原理**
* 将一个进程的地址空间划分成若干个大小相等的区域，称为**页**。
* 相应地，将**主存**空间划分成与页相同大小的若干个物理块，称为**块或页框**。
* 为进程分配主存时，可将进程中若干页分别装入多个不相邻接的块中。

（2）地址结构

31 12 11 0

|  |  |
| --- | --- |
| **页号P** | **页内地址W** |

（3）页表

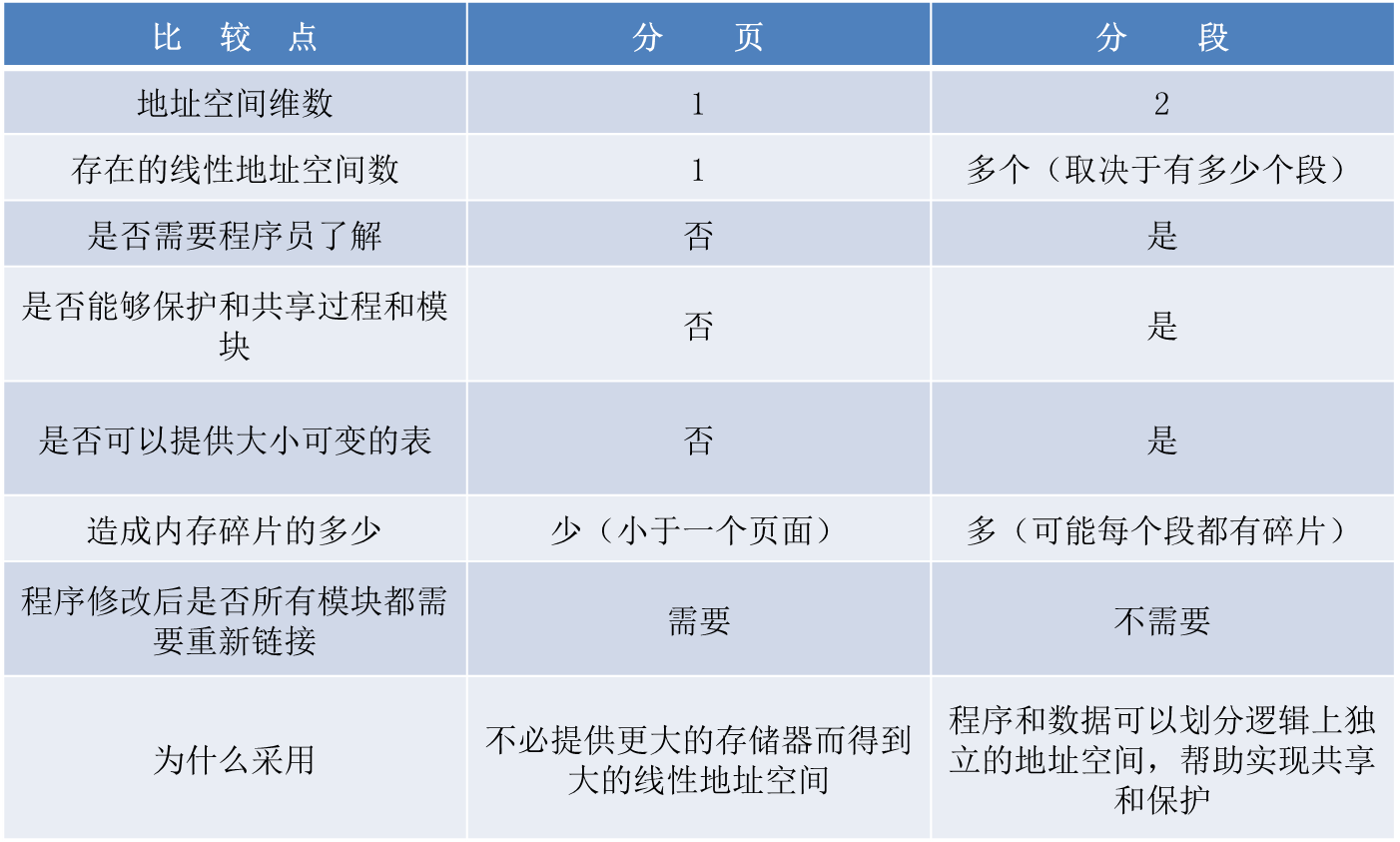
将进程的每一页离散地分配到内存的多个物理块中后，系统应保证能在内存中找到每个页面所对应的物理块。为此，系统为每个进程建立了一张页面映射表，简称**页表**。

（4）地址变换机构

基本任务是利用页表把用户程序中的逻辑地址变换成内存中的物理地址。

**23、段式与页式管理的不同**

在实现程序和数据的共享时，常常以信息的逻辑单位为基础。分页系统中的每一页只是存放信息的物理单位，其本身没有完整的意义，因而不便于实现信息共享，而段却是信息的逻辑单位，有利于信息的共享和保护。



**24、程序局部性原理及虚拟存储器的实现**

* 在一段时间内，程序的执行仅局限于某个部分。相应地，它所访问的存储空间也局限于某个区域内。

**（1）时间局限性。**如果程序中的某条指令一旦执行，则不久的将来该指令可能再次被执行。如果某个存储单元被访问，则不久以后该存储单元可能再次被访问。产生时间局限性的**原因是程序中存在大量的循环操作。**

**（2）空间局限性。**一旦程序访问了某个存储单元，则在不久的将来，其附近的存储单元最有可能被访问，即程序在一段时间内所访问的地址，可能集中在一定的范围内，其典型**原因是程序是顺序执行的。**

* **虚拟存储器的实现**

（1）请求分页系统

（2）请求分段系统

（3）请求段页式系统

* **25、页面置换算法（P83-85例题）**

（1）最佳（Optimal）置换算法

（2）先进先出（FIFO）置换算法

（3）最近最少使用（LRU）置换算法

（4）最近未用（NUR）置换算法

**26、设备管理技术**（提高设备的利用率）**及SPOOLING技术（P85-88）**

（1）通道技术

引入通道的目的是使数据的传输独立于CPU，将CPU从繁琐的I/O工作中解脱出来。

（2）DMA技术（直接内存存取）

指数据在内存与输入输出设备之间实现直接成块传送。

（3）缓冲技术

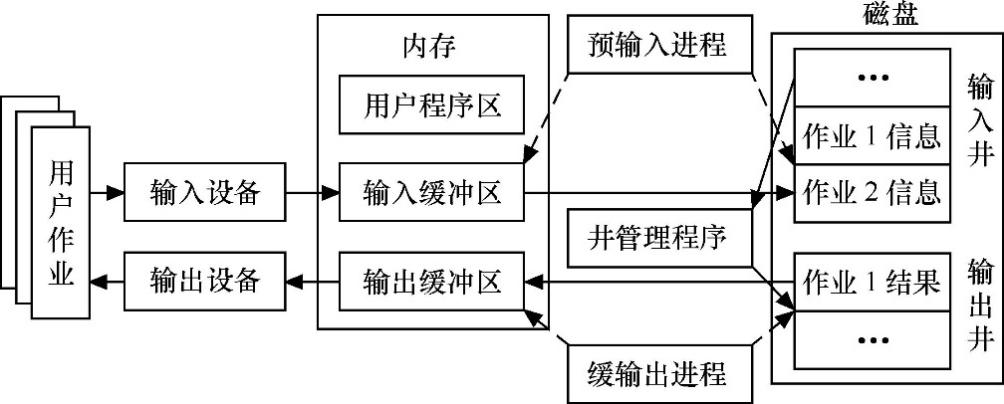
可提高外设利用率，尽可能使外设处于忙状态。

* （4）Spooling技术

是关于慢速字符设备如何与计算机主机交换信息的一种技术，通常称为“假脱机技术”。

Spooling系统由**“预输入程序”“缓输出程序”“井管理程序”以及输入和输出井**组成。

特点：提高了输入/输出速度，将独占设备改造为共享设备，实现了虚拟设备功能。



27、常见磁盘调度算法（不作为重点）

（1）磁盘驱动调度

例如，一个请求磁盘I/O的磁盘进程队列，分别在下列柱面上阻塞：98，181，35，112，24，144，45，57。如果磁头最初的位置在42号柱面。

* **先来先服务**

磁头移动的顺序为：42→98→181→35→112→24→144→ 45→57

* **最短寻道优先**

磁头移动的顺序为：42→45→35→24→57→98→112→144→181（按照最短路径）

* **扫描算法（SCAN）**

磁头移动的整个顺序为：42→35→24→0→45→57→98→112→144→181

* **单向扫描调度算法（CSCAN）**

磁头总是从0柱面开始移动，直到柱面的另一端；再回到0柱面重新开始，磁头移动的整个顺序为：42→45→57→98→112→144→181→199→0→24→35

（2）旋转调度算法

**28、文件系统的功能**

（1）按名存取

（2）统一的用户接口

（3）并发访问和控制

（4）安全性控制

（5）优化性能

（6）差错恢复

**29、文件的逻辑结构和物理结构（P91-92）**

**逻辑结构**可分为两大类：**（1）有结构的记录式文件**，它是由一个以上的记录构成的文件，故又称为记录式文件，记录的长度可分为定长和不定长两类。**（2）无结构的流式文件**，它是由一串顺序字符流构成的文件。

**物理结构**指文件的内部组织形式，常见的有4类，**（1）连续结构，也称顺序结构**，在对文件中的记录进行批量存取时效率最高；**（2）链接结构，也称串接结构**，每个物理块设有一个指针指向下一个物理块；**（3）索引结构**，系统为每个文件建立一张索引表；**（4）多个物理块的索引表**，根据一个文件大小的不同，其索引表占用物理块的个数不等。

**30、文件的目录结构（P93）**

（1）一级目录结构

（2）二级目录结构，由主文件目录和用户目录组成

（3）多级目录结构，也称为树型目录结构

31、文件的存取方法和存储空间管理方法

文件的**存取方法**是指读写文件存储器上的一个物理块的方法，有**（1）顺序存取法；（2）直接存取法**，允许用户随意存取；**（3）按键存取法，**是直接存取法的一种，根据被称为“键”的数据项内容存取。

（外部存储器）**存储空间的管理方法**有**（1）位示图，**每一位对应文件存储器上的一个物理块，取值0和1分别表示空闲和占用；**（2）空闲区表，**将一个连续未分配区域称为“空闲区”，空闲表中包含序号、空闲区的第一块号、空闲块的块数和状态等信息；**（3）空闲块链，**每个空闲物理块中有指向下一个空闲物理块的指针，所有空闲物理块构成一个链表，记录头指针即可，节省空间；**（4）成组链接法，**UNIX系统中，每100个空闲块为一组，每组的第一个空闲块登记了下一组的物理盘块号和空闲块总块数。

**32、作业调度算法**

作业状态（提交、后备、执行、完成）的划分比进程状态的划分更粗，进程是作业全部状态中的一个阶段体现。作业调度将作业从后备状态转换到内存执行状态。作业执行状态包含作业所对应进程的就绪、运行和阻塞状态。

（1）单道批量处理

* **先来先服务**

先启动等待时间最长的作业，是一种非抢占式算法

* **短作业优先（Shortest-Job-First，SJF）**

先启动运行时间最短的作业，是一种抢占式算法

* **响应比高者优先**



在每次调度前都要计算所有备选作业的响应比，然后选择响应比最高的作业执行。该算法比较复杂，系统开销大。响应比高，可能是因为作业等待时间长，也可能是因为作业需要处理时间短，它能够客观地对待长作业和短作业。

（2）多道批量处理

* **优先级调度算法**

用户指定优先级，优先级高的作业先启动

* **均衡调度算法**

根据系统的运行情况和作业本身的特性对作业进行分类。作业调度程序轮流地从这些不同类别的作业中挑选作业执行。

（3）作业调度算法性能的衡量指标（例题：P101+课件2.2.4）

周转时间=完成时间-提交时间

周转系数=周转时间/执行时间

**就平均周转时间和平均周转系数来说，最短作业优先算法最小，FCFS算法最大，相应比高者优先算法居中。**

**33、文件的保护方式（P96）**

（1）存取控制矩阵

（2）存取控制表

（3）用户权限表

（4）密码

34、计算机网络基础：

IP地址、子网掩码，超网，协议、Internet基础，安全基础。

简答：

执行指令的步骤（P28）

CPU与外设之间交换数据的方式（P38）

产生死锁的4个必要条件（P75）

名词解释

进程：进程（Process）是程序的一次执行。进程通常由程序、数据和进程控制块（Process Control Block，PCB）组成。其中，PCB是进程存在的唯一标志。

地址重定位：将逻辑地址转换成主存物理地址的过程称为地址重定位。

课堂提问：

分页管理（P79）

Spooling系统的组成和结构（P88）

可能考一道10分的题

页面置换算法（P83）

名词解释/简答

Spooling技术

选择

磁盘驱动调度（P88）

一些概念：

（1）进程控制是由操作系统内核(Kernel)中的原语实现的。

（2）原语(Primitive)是指由若干条机器指令组成的、用于完成特定功能的程序段。原语的特点是在执行时不能被分割，即原子操作要么都做，要么都不做。

（3）进程间的**同步**是指进程间完成一项任务时直接发生相互作用的关系。

进程间的**互斥**是指系统中各进程互斥使用临界资源。

（4）文件是具有符号名的、在逻辑上具有完整意义的一组相关**信息项**的集合。

**一、 单项选择题（本大题共20小题，每小题1分，共20分）**

* 1. 计算机中，表示信息的最小单位是（ ）?

|  |  |  |  |
| --- | --- | --- | --- |
| A. Byte | B. Word | C. Bit | D. KB |

**二、名词解释（本大题共5小题，每小题3分，共15分）**

1. 地址重定位：将逻辑地址转换成主存物理地址的过程称为地址重定位。

**三、简答题（本大题共5小题，每小题6分，共30）**

1. 请把237.625转换为二进制数据。

**四、实验题（15分）**

**五、应用题（20分）**

1、桌上有一空盘，允许存放一只水果。爸爸可向盘中放苹果，也可向盘中放桔子，儿子专等吃盘中的桔子，女儿专等吃盘中的苹果。规定当盘空时一次只能放一只水果供吃者取用，请用P、V原语实现爸爸、儿子、女儿三个并发进程的同步。

设置三个信号量S,So,Sa ，初值分别为1，0，0。分别表示可否向盘中放水果，可否取桔子，可否取苹果。

