1. 基本概念
2. 什么是虚拟计算机？

定义： 从不同角度所看到的计算机系统的属性是不同的

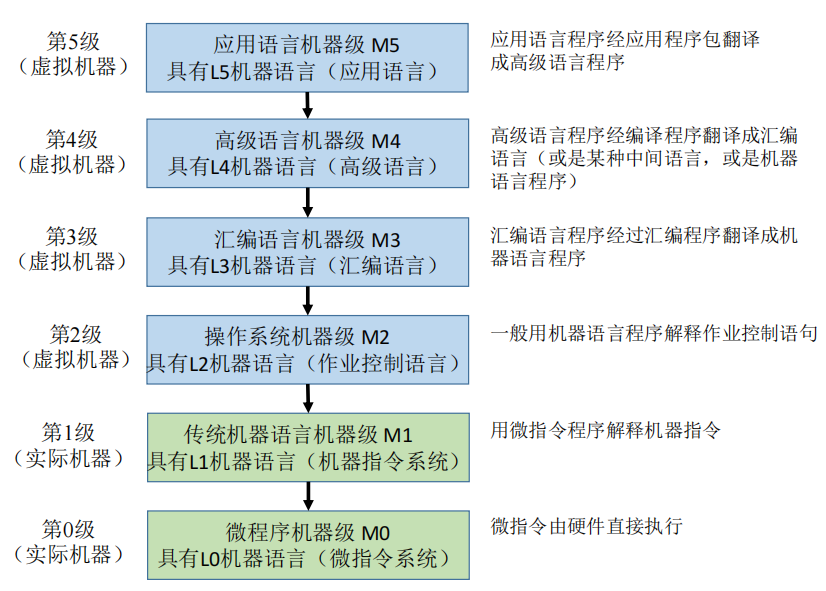
主要观察角度包括：  
• 应用程序员  
• 系统程序员

• 硬件设计人员

对计算机系统的认识通常只需要在某一个层次上

1. 计算机系统的层次结构

共6级。



1. 计算机系统结构的定义

（定义一）**Amdahl**于**1964**年在推出**IBM360**系列计算机时提出 ：

程序设计者所看到的计算机系统的属性**,** 即概念性结构和功能特性  
 • 程序员：系统程序员（包括：汇编语言、机器语言、编译程序、操作系统）  
 • 看到的：编写出能在机器上正确运行的程序所必须了解到的

（定义二）研究软硬件功能分配和对软硬件界面的确定

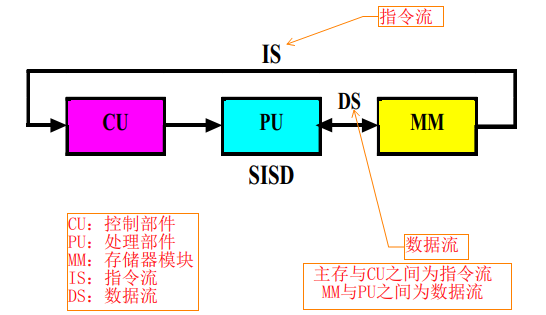
• 计算机系统由软件、 硬件和固件组成， 它们在功能上是同等的。  
 • 同一种功能可以用硬件实现， 也可以用软件或固件实现。  
 • 不同的组成只是性能和价格不同, 他们的系统结构是相同的。

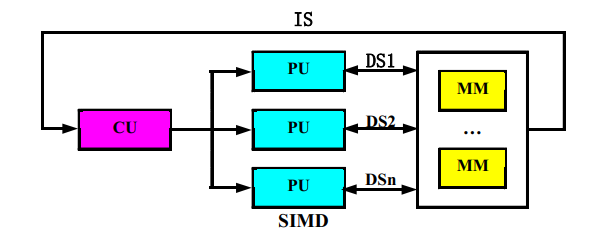
1. 计算机系统的分类（Flynn分类法）

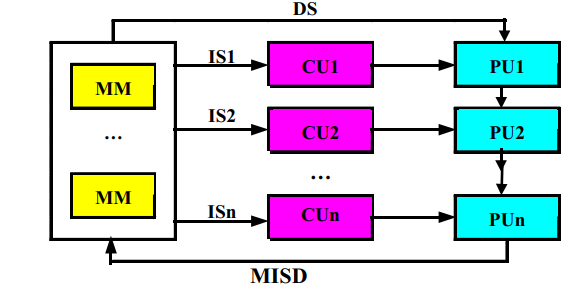
按照指令流和数据流的多倍性特征进行分类  
• 指令流：机器执行的指令序列  
• 数据流：由指令流调用的数据序列  
• 多倍性**(multiplicity)**：在系统性能瓶颈部件上同时处于同一执行阶段的指令或数据的最大可能个数

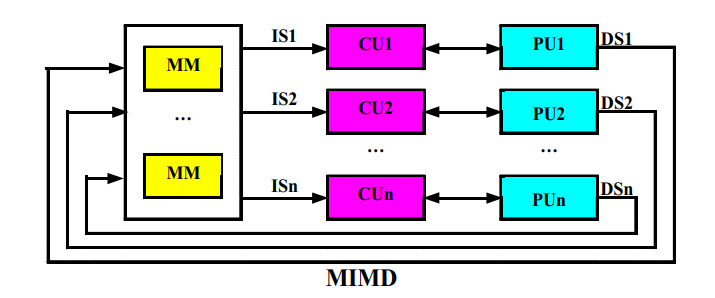
四种类型:<PPT:21-24>

1. 单指令流单数据流 **SISD(Single Instruction Single Datastream)**

 **2)** 单指令流多数据流 **SIMD(Single Instruction Multiple Datastream)**

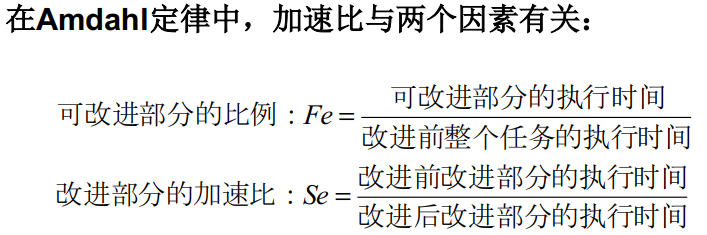
 **3)** 多指令流单数据流 **MISD(Multiple Instruction Single Datastream)**

 **4)** 多指令流多数据流 **MIMS(Multiple Instruction Multiple Datastream)**

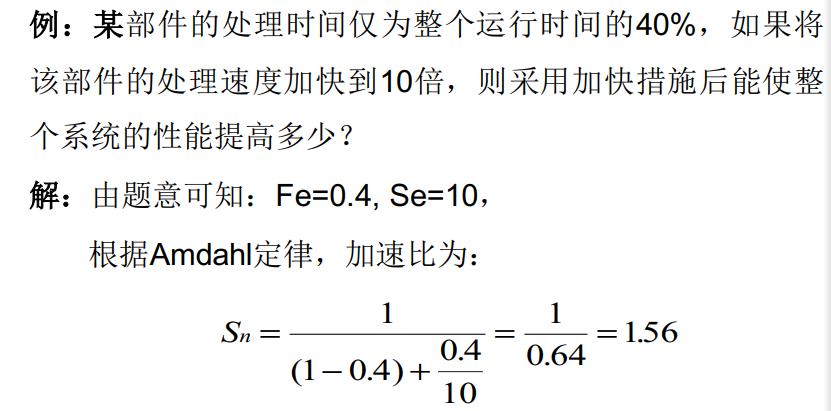


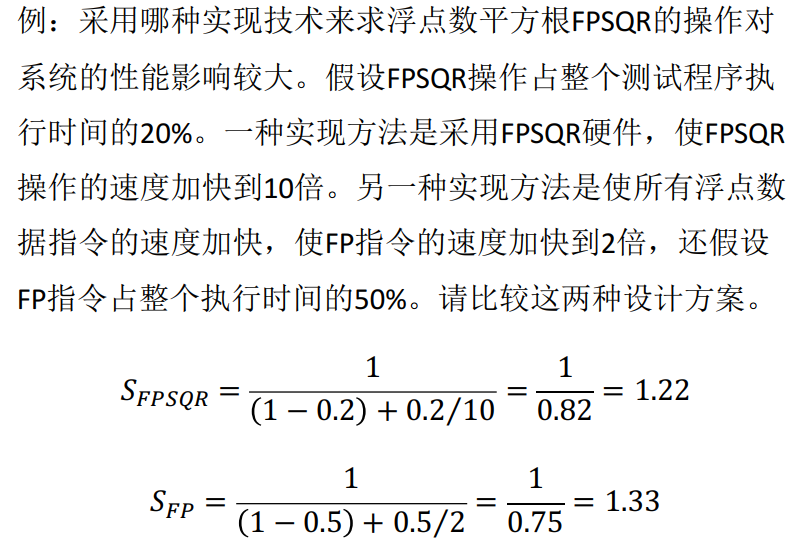
5. Amdahl定律及其应用

**Amdahl**定律：系统中某一部件由于采用更快的执行方式后，整个系统性能的提高与这种执行方式的使用频率或占总执行时间的比例有关。

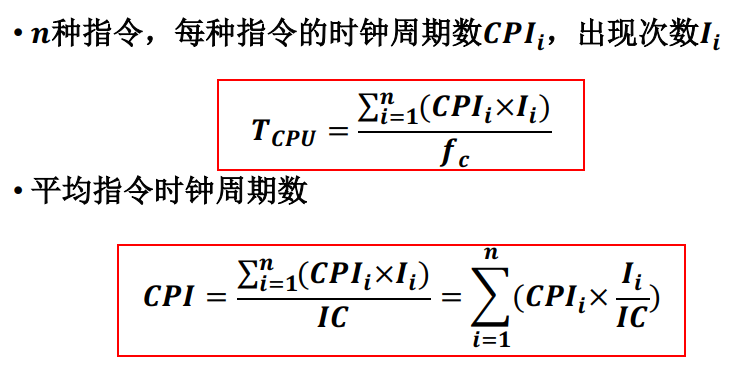
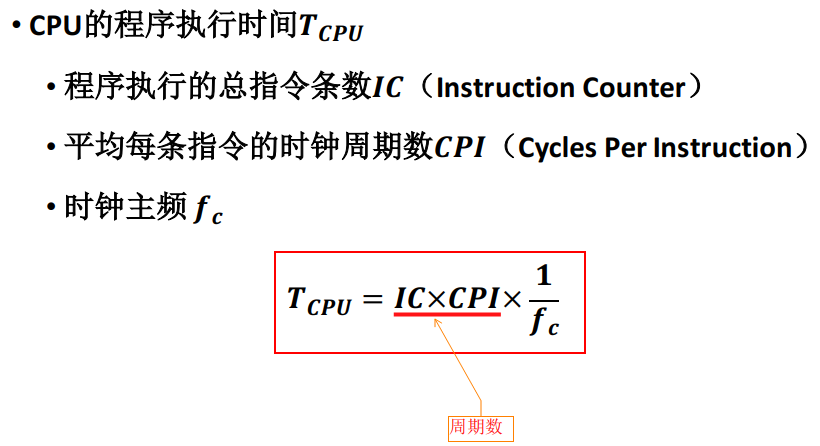


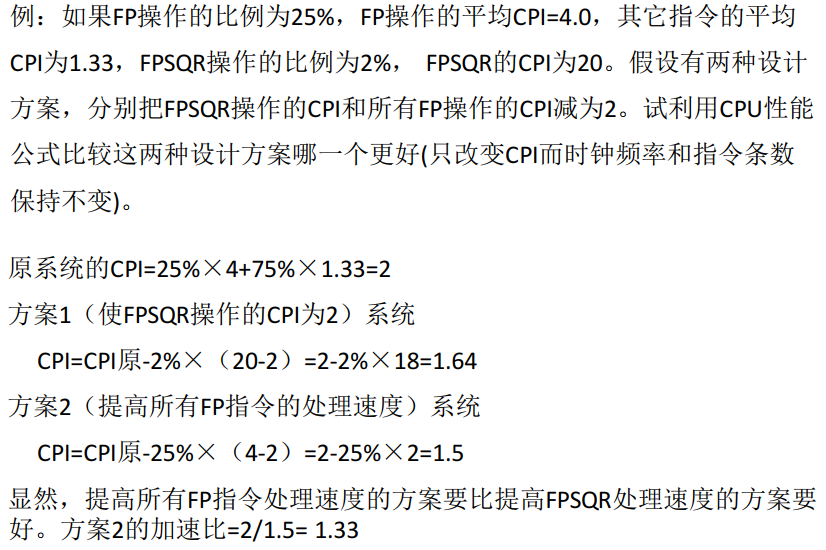






6.CPU性能公式





7.程序访问的局部性规律

• 局部性分时间上的局部性和空间上的局部性  
• 时间局部性：程序中近期被访问的信息项很可能马上将被再次访问。  
• 空间局部性：指那些在访问地址上相邻近的信息项很可能会被一起访问。

1. 指令系统
2. 数据表示与数据类型的定义与区别

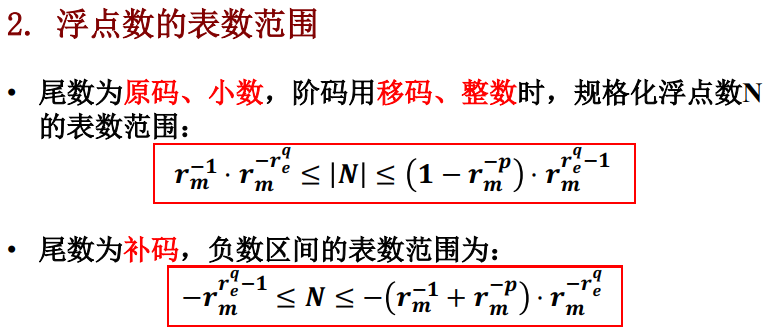
数据类型：文件、图、表、树、队列、阵列、链表、栈、向量、串、实数、整数、布尔树、字符等

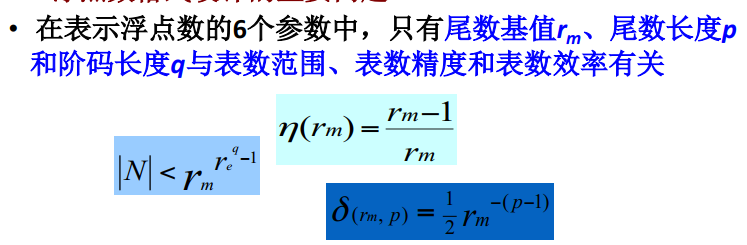
数据表示：计算机硬件能够直接识别、可以被指令系统直接调用的那些数据类型。

数据结构：串、队、栈、向量、阵列、链表、树、图等软件要处理的各种数据结构，反映了应用中要用到的各种 数据元素或信息单元之间的结构关系

数据表示和数据结构都是数据类型的子集。

1. 浮点数的表示方式、表数范围、表数精度和表数效率，浮点数格式设计





在字长确定之后，尾数基值***rm***取**2**或**4**，浮点数具有最大表数范围和最高表数精度。

1. 基本的寻址方式和定位方式

寻址方式：

立即数寻址

寄存器寻址

主存寻址

直接寻址

间接寻址

变址寻址

程序定位：将程序的逻辑地址转化为主存的物理地址

原因：• 程序的独立性  
• 程序的模块化设计  
• 数据结构在程序运行过程中， 其大小往往是变化的  
• 有些程序本身很大， 大于分配给它的主存物理空间

方式：直接定位（程序装入主存前，程序中的指令和数据的主存物理就已经确定 ）

静态定位（程序装入主存的过程中随即进行地址变换， 确定指令和数据的主存物理地址 ）

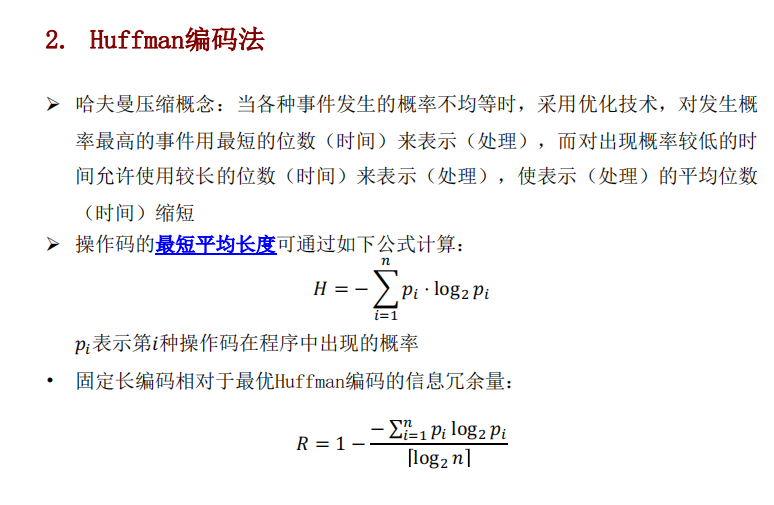
动态定位（在程序执行过程中， 当访问到相应的指令或数据时才进行地址变换， 确定指令和数据的主存物理地址 ）

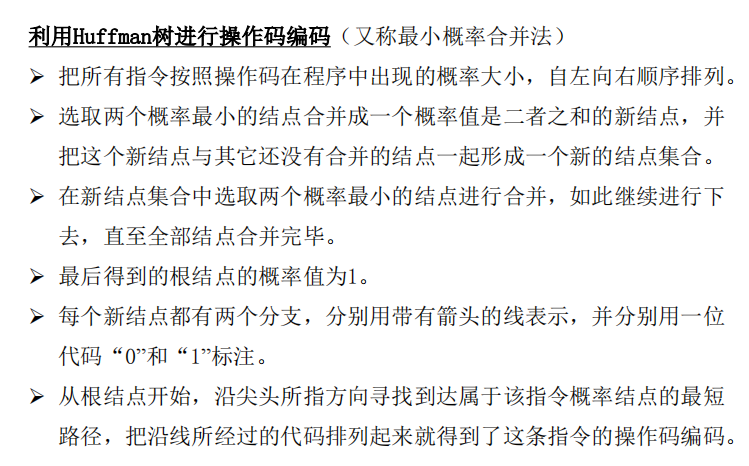
1. 操作码优化设计

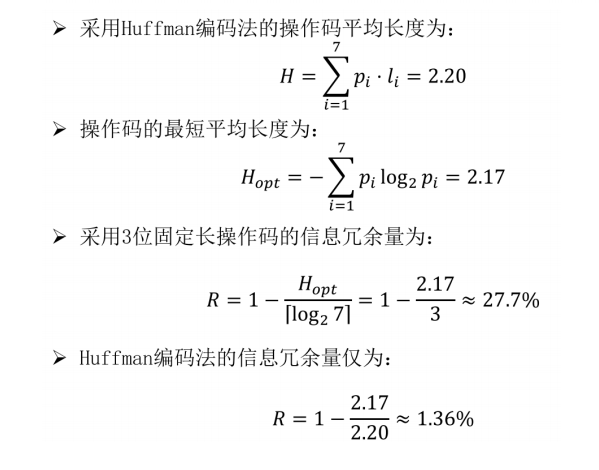
固定长操作码

主要优点：规整， 译码简单  
主要缺点： 浪费信息量（操作码的总长位 数增加）

Huffman 编码







扩展编码法(由固定长操作码与**Huffman**编码法相结合形成 )

5.RISC 的基本原理和思想、实现的关键技术

基本原理：使流水线处理能高效率执行；使优化编译器便于产生优化代码

减少CPI（指令平均执行周期数）是RISC思想的精华

硬件方面：  
 采用硬布线控制逻辑  
 减少指令和寻址方式的种类  
 使用固定的指令格式  
 采用LOAD/STORE结构  
 指令执行过程中设置多级流水线等  
 软件方面：  
 十分强调优化编译技术的作用。

实现的关键技术：

1.延时转移技术

为了使指令流水线不断流，在转移指令之后插入一条没有数据相关和控制相关的有效指令，而转移指令被延迟执行，这种技术称为延迟转移技术。

采用指令延迟转移技术时，指令序列的调整由编译器自动进行，用户不必干预。

2.指令取消技术（分为两种情况）

**(1)** 向后转移（适用于循环程序）  
实现方法：循环体的第一条指令安放在两个位置，分别在循环体的前面和后面。如果转移成功，则执行循环体后面的指令，然后返回到循环体开始；否则取消循环体后面的指令

效果：能够使指令流水线在绝大多数情况下不断流；对于循环程序，绝大多数情况下转移是成功的；只有最后一次出循环时转移不成功

1. 向前转移**(IF THEN )**

实现方法：如果转移不成功，执行转移指令之后的下条指令，否则取消下条指令。

效果：转移成功与不成功的概率**,** 通常各**50**％  
主要优点：不必进行指令流调整

3.重叠寄存器窗口技术

原因：在**RISC**中，子程序比**CISC**中多，因为传送参数而访问存储器的信息量很大

实现方法：设置一个数量比较大的寄存器堆，并把它划分成很多个窗口。在每个过程使用的几个窗口中：  
 • 有一个窗口是与前一个过程共用  
 • 有个窗口是与下一个过程共用

效果：可以减少大量的访存操作。

4.指令流调整技术

目标：通过变量重新命名消除数据相关，提高流水线效率

5.硬件为主固件为辅

微程序实现的优点是便于实现复杂指令，便于修改指令系统，增加了机器的灵活性，但是执行速度低。

以硬联逻辑为主来实现指令系统

1. 存储系统
2. 存储系统的定义和评价标准（价格、容量和速度）

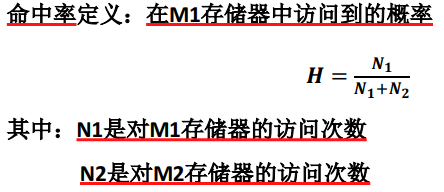
定义：两个或两个以上速度、容量和价格各不相同的存储器用硬件、软件、或软件与硬件相结合的方法连接起来成为一个存储系统。这个存储系统对应用程序员是透明的，并且，从应用程序员看，它是一个存储器，这个存储器的速度接近速度最快的那个存储器，存储容量与容量最大的那个存储器相等，单位容量的价格接近最便宜的那个存储器。

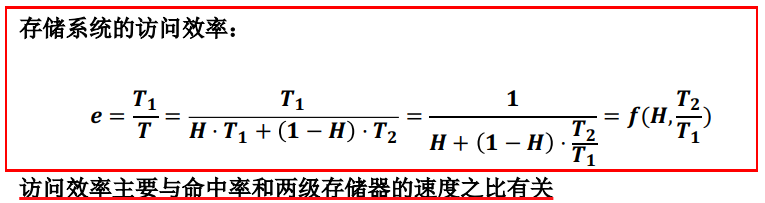
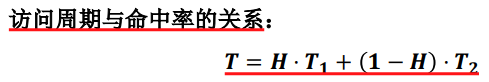
T：存储周期 S：容量 C：每位的价格

容量：（要求）提供尽可能大的地址空间，能够随机访问

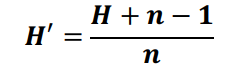
（方法有两种）  
Ø只对系统中存储容量最大的那个存储器进行编址，其他存储器只在内部编址或不编址，例如Cache存储系统  
Ø另外设计一个容量很大的逻辑地址空间，把相关存储器都映射这个地址空间中，例如虚拟存储系统

速度：（提高存储系统速度的两条途径）  
1) 提高命中率H，  
2) 是两个存储器的速度不要相差太大



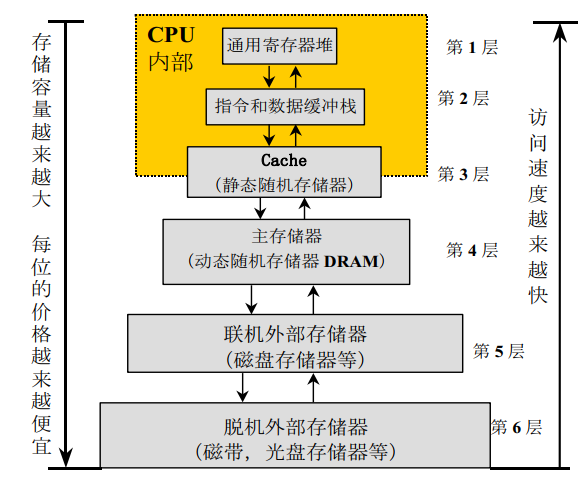
  
2. 采取预取技术提高命中率

方法：不命中时，把**M2**存储器中相邻多个单元组成的一个数据块取出来送入**M1**存储器中。



n为数据块大小与数据重复使用次数的乘积，H是原来的命中率，H’是新的命中率

3.存储系统的层次结构、地址变换与映像的定义

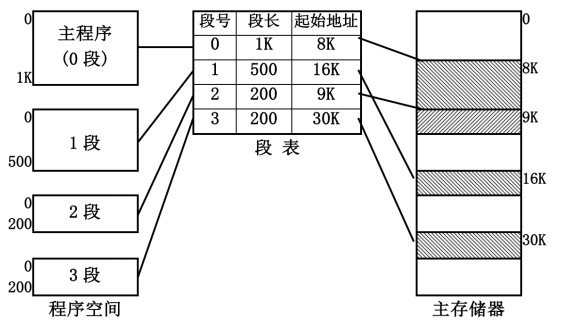


地址变换：在程序运行时，把虚地址变换为主存实地址

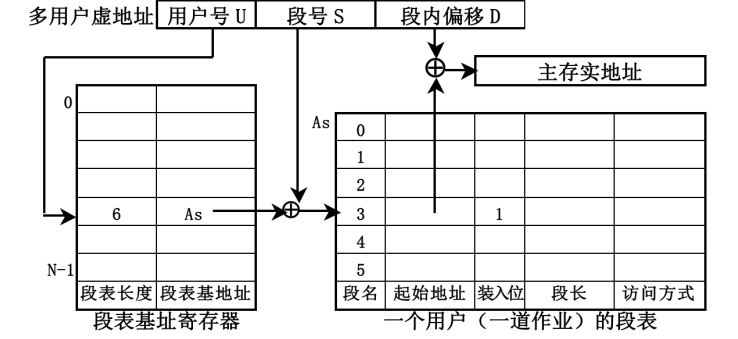
地址映象：把虚拟地址空间映像到主存地址空间  
4. 虚拟存储器工作原理（页式、段式、段页式）

段式虚拟存储器：

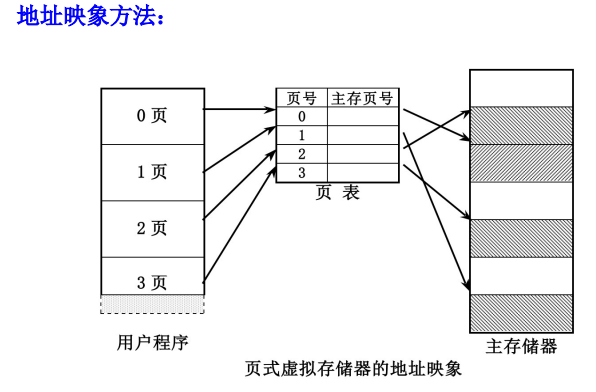
地址映象方法： 每个程序段都从0地址开始编址，长度可长可短，可以在程序执行过程中动态改变程序段的长度。

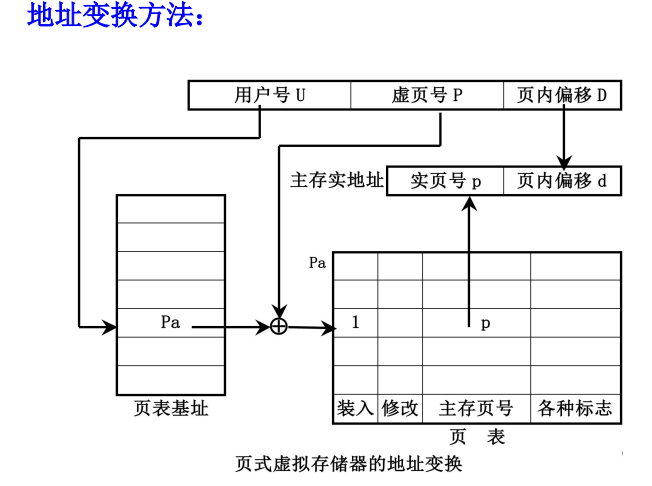


地址变换方法： 由用户号找到基址寄存器，读出段表起始地址，与虚地址中段号相加得到段表地址，把段表中该段的起始地址与段内偏移**D**相加就能得到主存实地址。

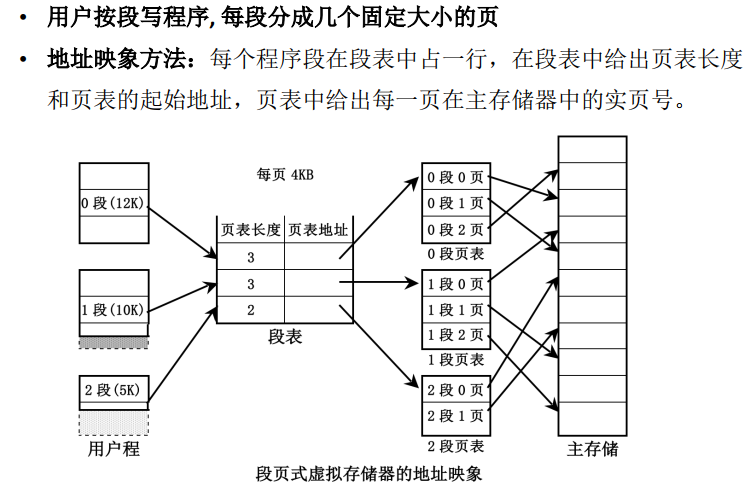


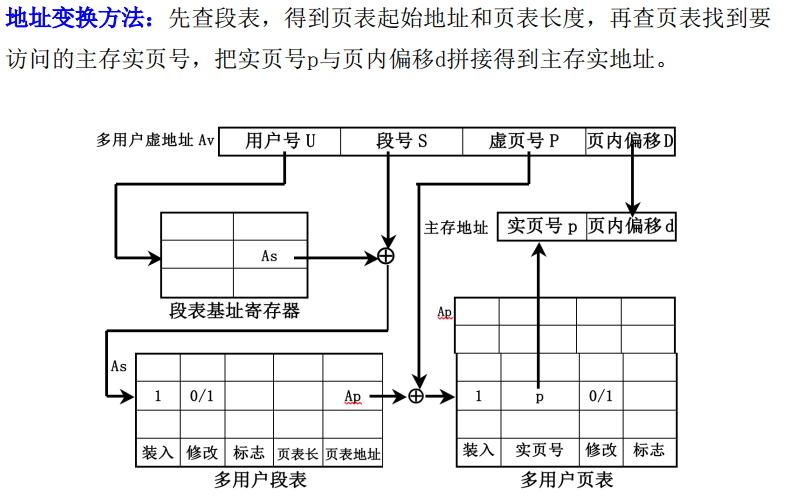
页式虚拟存储器：

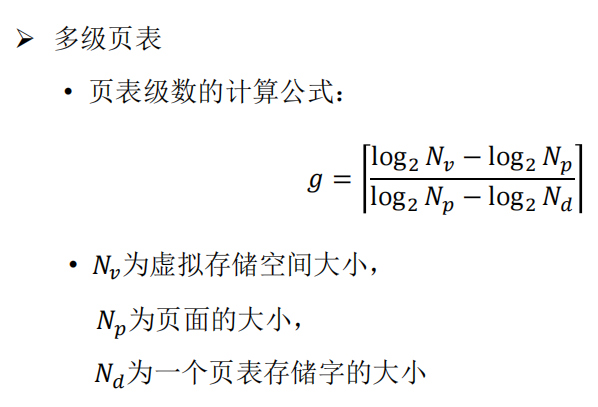
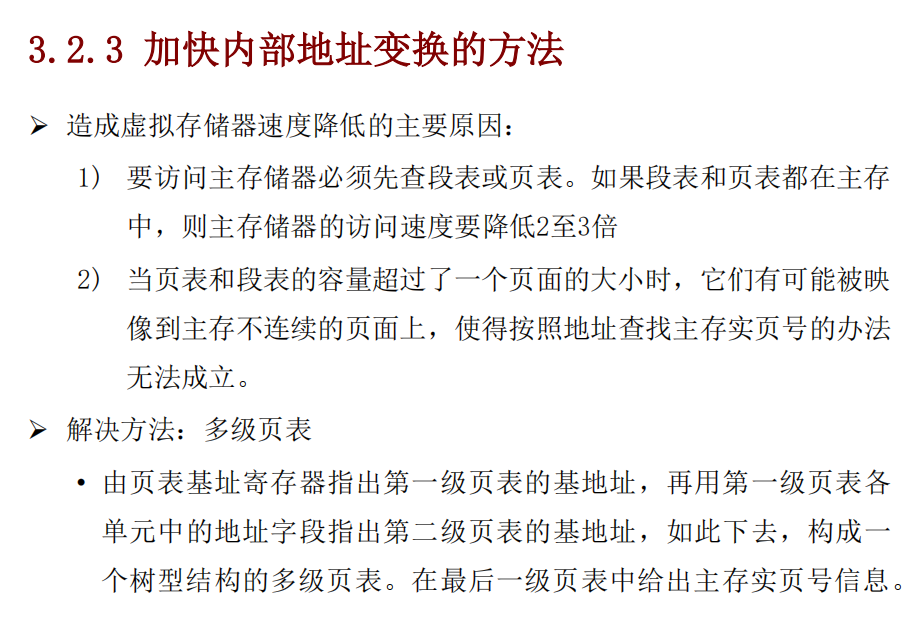


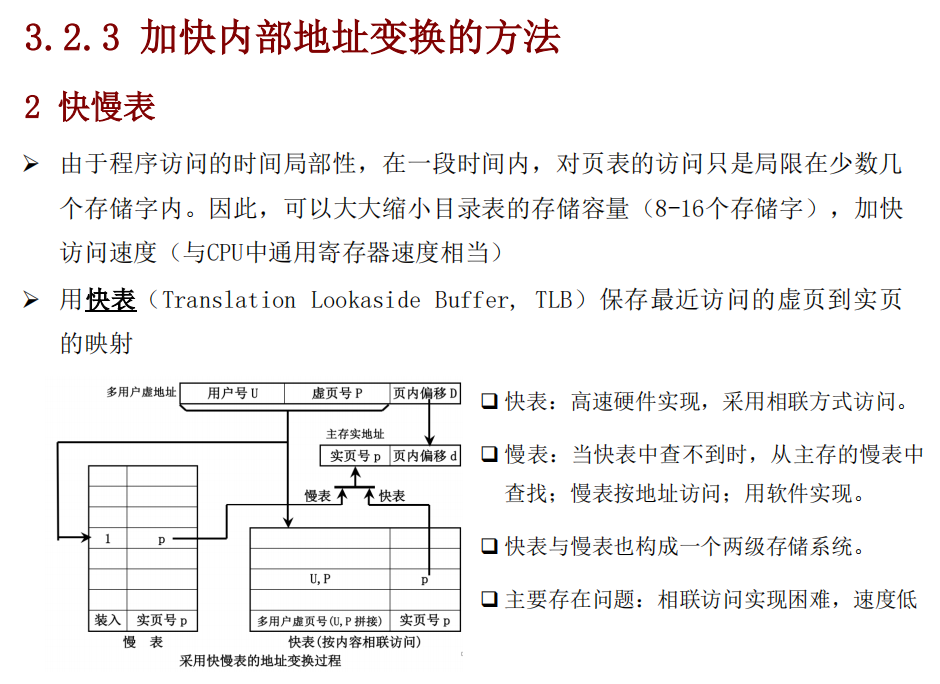


段页式虚拟存储器：



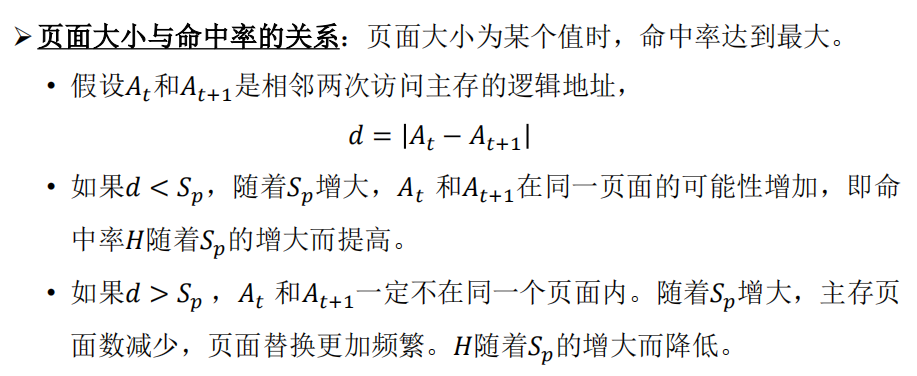


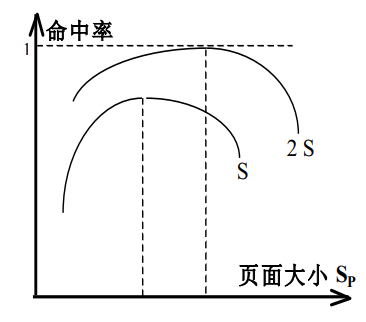


  
5. 主要的页面替换算法

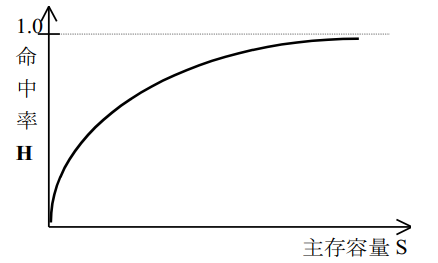
FIFO、LRU  
6. 页面大小、主存容量和页面调度算法对主存命中率的影响

页面大小：页面大小为某个值时，命中率达到最大

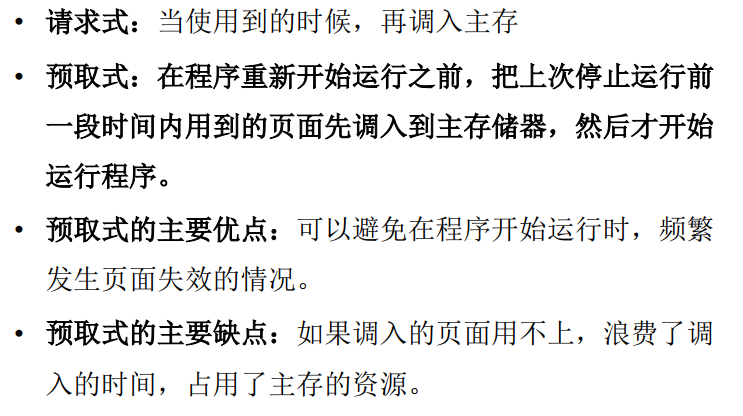




主存容量：单调上升

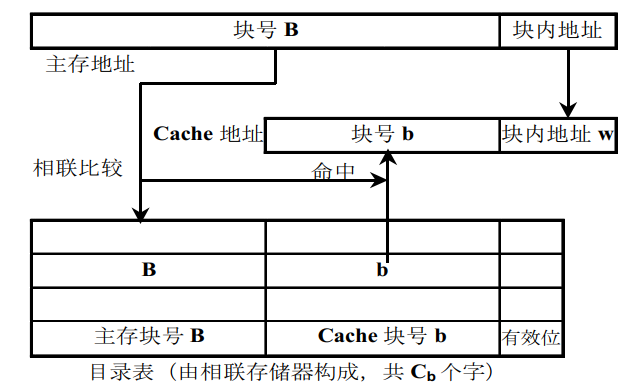


页面调度算法：

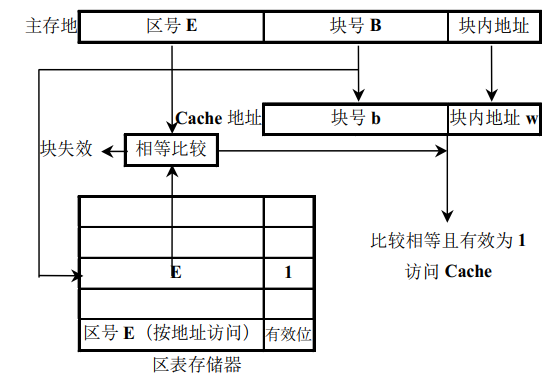
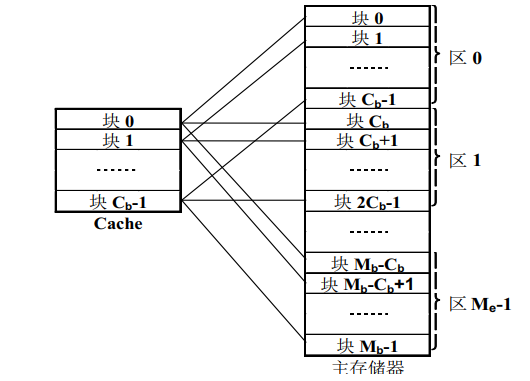
  
7. Cache 存储系统的基本工作原理（直接映像、全相联、组相联）

目录表 区表存储器 块表

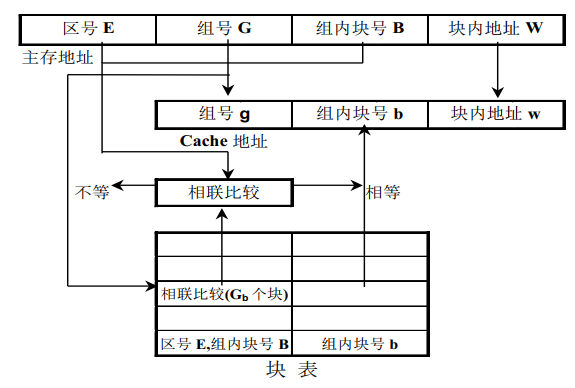
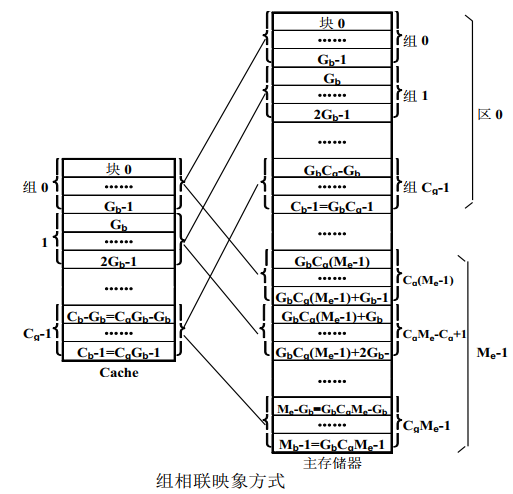
全相联：



直接相联：

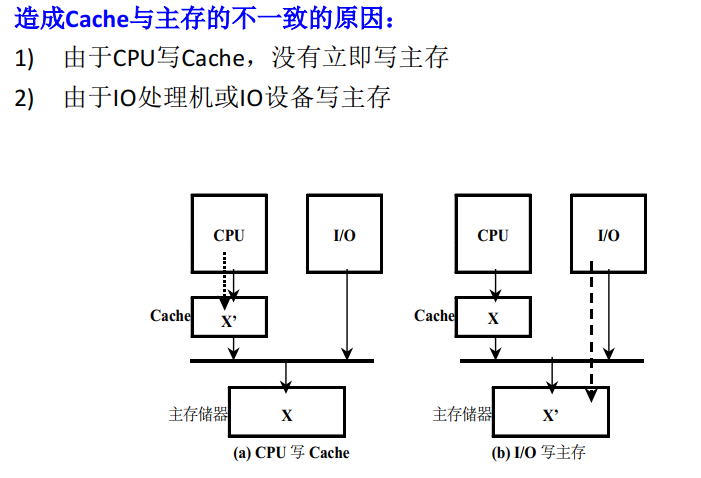


组相联：



当要访问cache时，用主存地址中的组号G按地址访问块表存储器

块表的容量与cache的大小相等  
8. Cache 的一致性问题

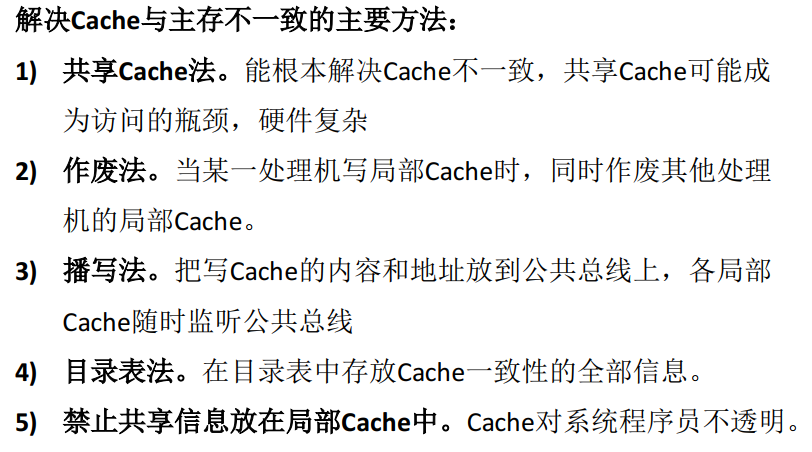


• **Cache**的更新算法  
**1)** 写直达法，写通过法， **WT(Write-through)**。 **CPU**的数据写入**Cache**时，同时也写入主存。  
**2)** 写回法，抵触修改法， **WB(Write-Back)**。 **CPU**的数据只写入**Cache**，不写入主存，仅当替换时，才把修改  
过的**Cache**块写回主存。

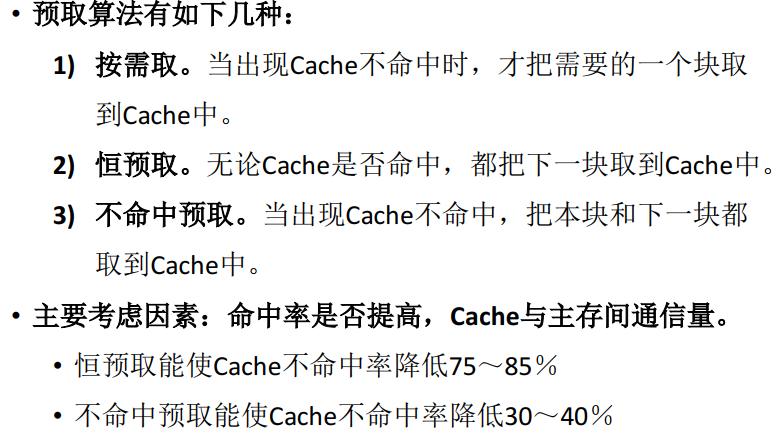
• 写回法与写直达法的优缺点比较：  
**1)** 可靠性，写直达法优于写回法。写直达法能够始终保证**Cache**是主存的副本。如果**Cache**发生错误，可以从  
主存得到纠正。

1. 与主存的通信量，写回法少于写直达法。
2. 控制的复杂性**,** 写直达法比写回法简单。
3. 硬件实现的代价**,** 写回法要比写直达法好。

写**Cache**的两种方法：  
**1)** 不按写分配法：在写**Cache**不命中时，只把所要写的字写入主存。  
**2)** 按写分配法：在写**Cache**不命中时，还把一个块从主存读入**Cache**。  
目前，在写回法中采用按写分配法，在写直达法中采用不按写分配法。



9. Cache 的预取算法



问题：ps2的2（4）cache的地址流

1. 输入输出系统
2. 基本的输入输出方式（程序控制输入输出方式、中断输入输出方式、 DMA 方式）

<书：P212-217>

程序控制输入输出方式（状态驱动输入输出方式、应答输入输出方式、查询输入输出方式、条件驱动输入输出方式 ）

四个特点：

1. 何时对何设备进行输入输出操作受CPU控制  
    2) CPU要通过指令对设备进行测试才能知道设备的工作状态。空闲、准备就绪、忙碌等  
    3) 数据的输入和输出都要经过CPU  
    4) 用于连接低速外围设备，如终端、打印机等

（P213、P214图）

中断输入输出方式  
定义：当出现来自系统外部， 机器内部， 甚至处理 机本身的任何例外的， 或者虽然是事先安排的， 但出现在现行程序的什么地方是事先不知道的事件时， CPU暂停执行现行程序， 转去处理这些事件， 等处理完成后再返回来继续执行原先的程序。  
特点：  
 **1) CPU**与外围设备能够并行工作。  
 **2)** 能够处理例外事件。  
 **3)** 数据的输入和输出都要经过**CPU**。  
 **4)** 用于连接低速外围设备。

直接存储器访问方式  
 直接存储器访问方式(DMA：Direct Memory Access)， 主要用来连接高速外围设备。 如磁盘存储器， 磁带存储器、 光盘辅助存储器， 行式打印机等。

（P215图）

DMA方式具有如下特点：  
1) 外围设备的访问请求直接发往主存储器， 数据的传送过程不需要CPU的干预。  
2) 全部用硬件实现， 不需要做保存现场和恢复现场等工作。  
3) DMA控制器复杂， 需要设置数据寄存器、 设备状态控制寄存器、 主存地址寄存器、 设备地址寄存器和数据交换个数计数器及控制逻辑等。  
4) 在DMA方式开始和结束时， 需要处理机进行管理。

输入输出设备的工作流程（P216）

目前使用的DMA方式：

1) 周期窃取方式：  
• 在每一条指令执行结束时， CPU测试有没有DMA服务申请。  
• 借用CPU完成DMA工作流程。 包括数据和主存地址的传送， 交换个数计  
数器减1， 主存地址的增值及一些测试判断等。  
• 周期窃取方式的优点是硬件结构简单， 比较容易实现。  
• 缺点是在数据输入或输出过程种实际上占用了CPU的时间。  
2) 直接存取方式：  
• DMA控制器的数据传送申请直接发往主存储器  
• 整个工作流程全部用硬件完成。  
• 优点与缺点正好与周期窃取方式相反。  
3) 数据块传送方式：  
• 在设备控制器中设置一个比较大的数据缓冲存储器。 设备控制器与主  
存储器之间的数据交换以数据块为单位， 并采用程序中断方式进行。  
• 采用数据块传送方式的外围设备有软盘驱动器、 行式打印机、 激光打  
印机、 卡片阅读机、 绘图仪等。

2. 中断源分类和优先级

种类：1) 由外围设备引起的中断。 低速外围设备每传送一个字节申请一次中断；高速外围设备的前、后处理。  
2) 由处理机本身产生的中断。 如算术溢出，除数为零，数据校验错等。  
3) 由存储器产生的中断。 如地址越界、页面失效、访问存储器超时等。  
4) 由控制器产生的中断。 如非法指令、堆栈溢出、时间片到、切换到特权态。  
5) 由总线产生的中断。 输入输出总线出错,存储总线出错等。  
6) 实时过程控制产生的中断。  
7) 实时钟的定时中断。  
8) 多处理机系统中，从其它处理机发送来的中断。  
9) 程序调试过程中，由断点产生的中断。  
10) 硬件故障中断。  
11) 电源故障中断。

中断优先级：

安排中断优先顺序主要由下列因素来决定：  
• 中断源的急迫性。  
• 设备的工作速度。  
• 数据恢复的难易程度。  
• 要求处理机提供的服务量。

中断优先级与中断服务顺序  
• 要求：响应速度快，灵活性好。  
• 做法：由硬件排队器决定中断优先级，通过软件设置中断屏蔽码改变中断服务顺序。

3.中断处理过程及其软硬件分配

软硬件分配：

恰当分配中断系统的软硬件功能，是中断系统最关键问题。

主要考虑的两个因素：  
• 中断响应时间： 中断响应时间是一个非常重要的指标。  
• 灵活性： 硬件实现速度快，灵活性差；软件实现正好相反

中断处理过程：

（ 表示一般用硬件实现， 表示一般用软件实现， 表示可以用硬件实现， 也可以用软件实现）  
• 现行指令结束， 且没有更紧急的服务请求  
• 关CPU中断  
• 保存断点， 主要保存PC中的内容  
• 撤消中断源的中断请求  
• 保存硬件现场， 主要是PSW及SP等  
• 识别中断源  
• 改变设备的屏蔽状态  
• 进入中断服务程序入口  
• 保存软件现场， 在中断服务程序中使用的通用寄存器等  
• 开CPU中断， 可以响应更高级别的中断请求  
• 中断服务， 执行中断服务程序  
• 关CPU中断  
• 恢复软件现场  
• 恢复屏蔽状态  
• 恢复硬件现场  
• 开CPU中断  
• 返回到中断点

（必须用硬件实现的有： 保存中断点和进入中断服务程序入口。 必须用软件实现的有： 中断服务和返回到中断点。）

4.中断响应时间和服务顺序

中断响应时间  
• 定义：从中断源向处理机发出中断服务请求开始，到处理机开始执行这个中断源的中断服务程序时为止，这一段时间称为中断响应时间。  
• 影响中断响应时间的因素主要有4个： (前2个属于处理机设计，后2个属于中断系统)  
 1) 最长指令执行时间。有些指令的执行时间很长，甚至无法预测。  
 2) 处理其它更紧急的任务所用时间。如处理DMA请求等。  
 3) 从第一次关CPU中断到第一次开CPU中断所经历的时间。中断系统的软件与硬件功能分配，主要就是要考虑这一段内要所的事情用  
软件来实现，还是用硬件来实现。  
 4) 通过软件找到中断服务程序入口所用时间。主要是第1和第3两部分。其中，第1部分是指令系统设计时考虑的问题，在中断系统  
的设计中，主要考虑第3部分。

服务顺序？？？？？？？

5. 中断屏蔽（设置中断屏蔽位和改变处理机优先级）

（P231-233）

注：优先级决定响应顺序；屏蔽码改变服务顺序

两种方法的差别有：  
(1)两者使用的概念不同。  
前者使用中断屏蔽，  
后者使用中断优先级。  
(2)需要屏蔽码的位数不同。  
前者所需要的屏蔽位数比较多，  
*n*： log2*n*(3)可屏蔽的中断源数量和种类不同。  
前者可以任意屏蔽掉一个或几个中断源，  
后者只能屏蔽掉比某一个优先级低的中断源。

6. 通道处理机的作用和功能

作用：

使CPU摆脱繁重的输入输出负担和共享输入输出接口

能够负担外围设备的大部分输入输出工作，包括管理所有按字节传输方式工作的低速和中速外围设备，按数据块传输方式工作的高速外围设备，对DMA接口的初始化，设备故障的检测和处理等。

主要功能：

• 接受CPU发来的指令，选择一台指定的外围设备与通道相连接。  
• 执行CPU为通道组织的通道程序。  
• 管理外围设备的有关地址。  
• 管理主存缓冲区的地址。  
• 控制外围设备与主存缓冲区之间数据交换的个数。  
• 指定传送工作结束时要进行的操作。  
• 检查外围设备的工作状态，是正常或故障。  
• 在数据传输过程中完成必要的格式变换。

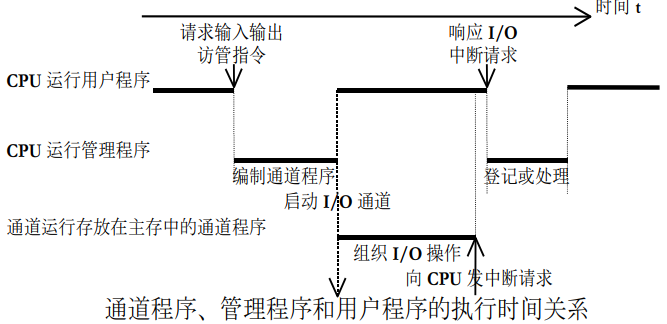
7. 通道处理机的工作过程和种类

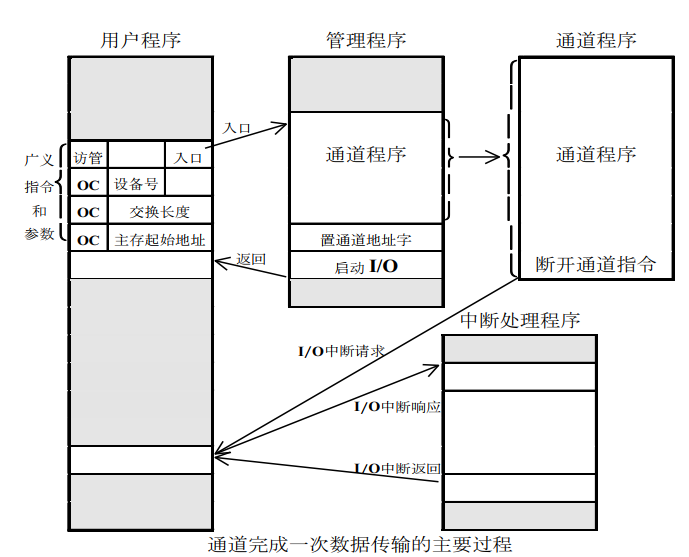
工作过程：

• 通道完成一次数据输入输出的过程分为三步：  
• 在用户程序中使用访管指令进入管理程序，由CPU通过管理程序组织一个通道程序，并启动通道。  
• 通道处理机执行通道程序， 完成指定的数据输入输出工作。  
• 通道程序结束后再次调用管理程序进行处理。

每完成一次输入输出工作， CPU只需要两次调用管理程序。

图：<书：P235\P236>





种类：

* 字节多路通道

为多台低中速的外围设备服务

有多个子通道， 每个子通道连接一个控制器

<图：书P239-图4.20>

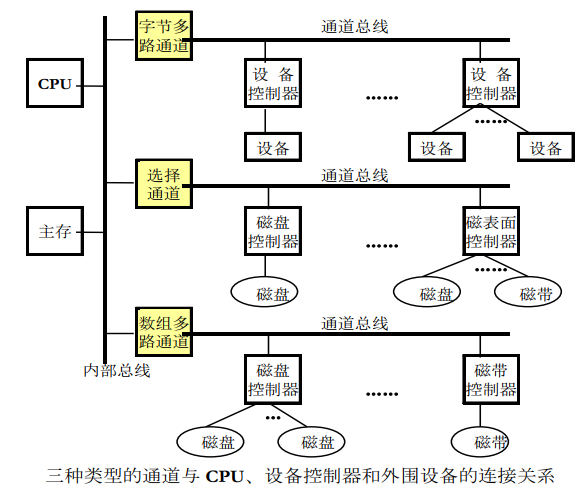
* 选择通道

为高速外围设备服务  
只有一个以成组方式工作的子通道

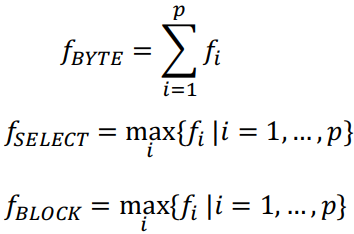
<图：书P240-图4.21>

* 数组多路通道

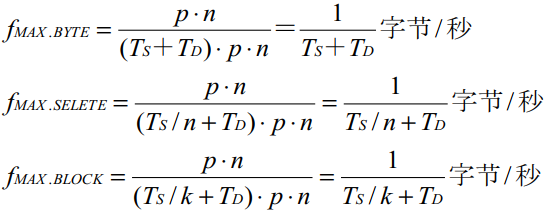
• 字节多路通道和选择通道的结合。  
• 每次为一台高速设备传送一个数据块，并轮流为多台外围设备服务。  
• 从磁盘存储器读出文件的的过程分为三步：定位、找扇区、读出数据  
• 数组多路通道的实际工作方式是：在为一台高速设备传送数据的同时，有多台高速设备可以在定位或者在找扇区。  
• 与选择通道相比，数组多路通道的数据传输率和通道的硬件利用都很高，控制硬件的复杂度也高。



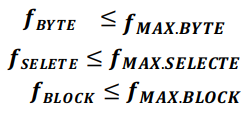
1. 通道处理机流量分析<P243-例4.1>

• 通道流量：单位时间内能够传送的最大数据量。又称通道吞吐率， 通道数据传输率等。  
• 通道最大流量： 通道在满负荷工作状态下的流量。  
• 通道流量与连接在通道上的设备的数据传输率的关系如下：  
 

三种通道的最大流量计算公式：



为保证通道不丢失数据，通道的实际流量应不大于通道最大流量



解决可能出现的数据丢失问题可以采用的方法：

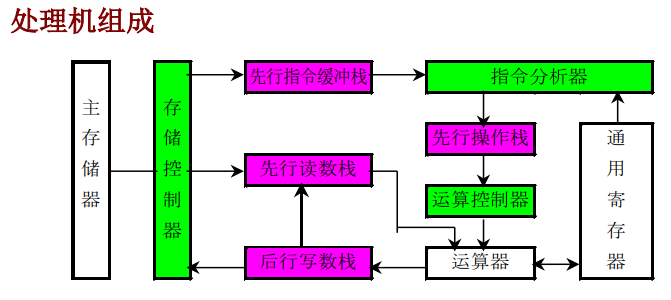
方法一：增加通道的最大工作流量。  
 方法二：动态改变设备的优先级。  
 方法三：增加缓冲存储器。

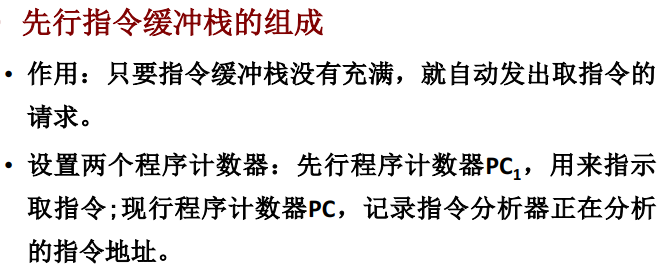
1. 标量处理机
2. 指令的重叠执行方式
3. 先行控制技术的基本结构

处理机结构：

三个独立的控制器：存储控制器、指令控制器、运算控制器

四个缓冲栈：先行指令缓冲栈、先行读数缓冲栈、先行操作栈、后行写数栈

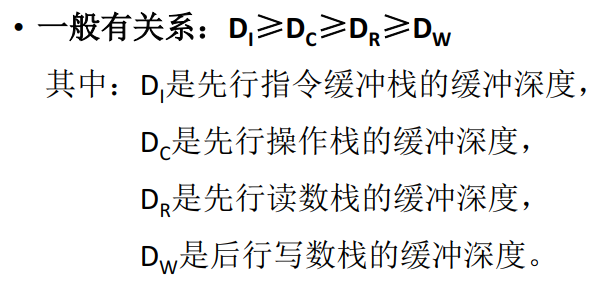




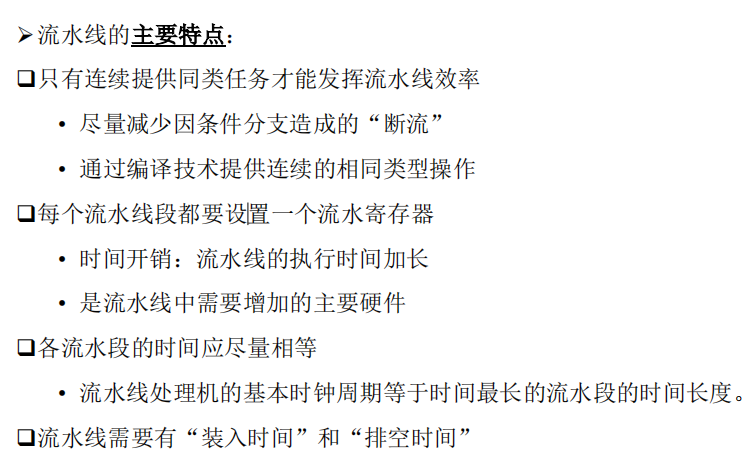
1. 缓冲深度的设计方法 (PPT例子，，，看不懂！！！)

以静态分析为主，通过模拟来确定缓冲深度。

1. 先行指令缓冲栈的设计(PPT:20-25)
2. 其他缓冲栈的设计原则



1. 流水线的基本原理（时空图）



1. 流水线的分类（线性流水线和非线性流水线、单功能和多功能流水线、静态和动态 流水线）

线性流水线：每个流水段流过且仅流过一次

非线性流水线：某些流水段之间有反馈回路或者前馈回路

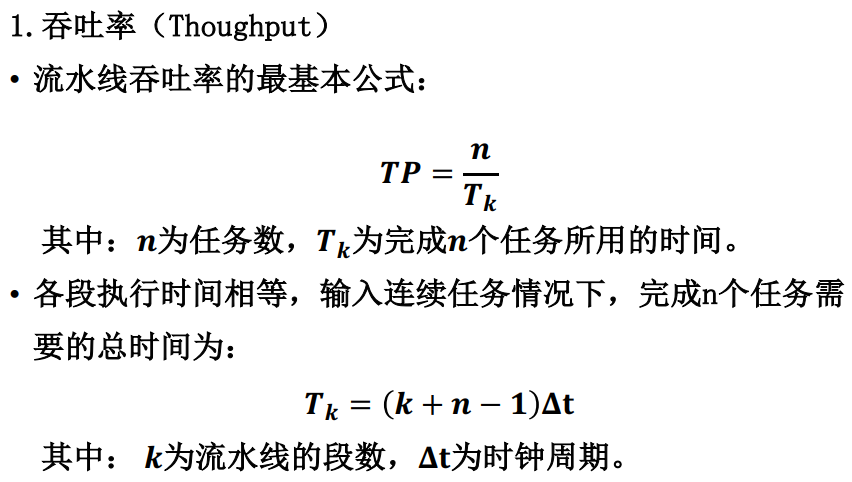
单功能流水线：只能完成一种固定功能的流水线

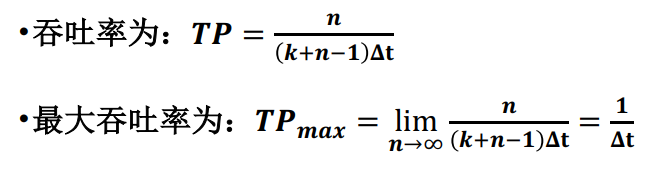
多功能流水线：流水线的各段通过不同连接实现不同功能。

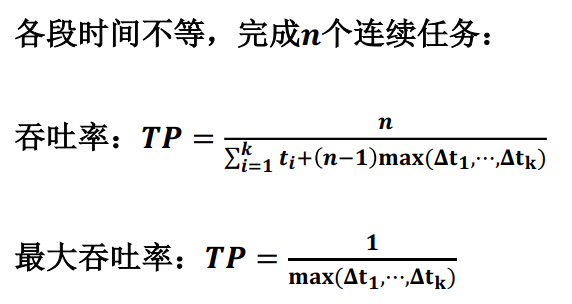
静态流水线：同一时间内，各个功能只能按照一种方式连接，实现一种固定的功能。

动态流水线：在同一段时间内，各段可以按照不同的方式连接，同时执行多种功能。

1. 线性流水线的性能分析（吞吐率、加速比、效率）



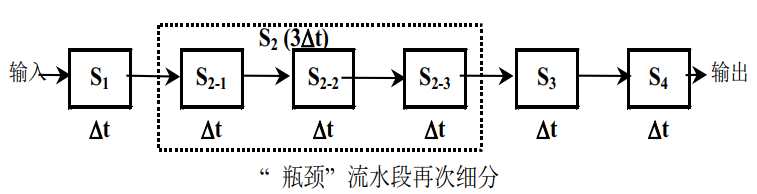




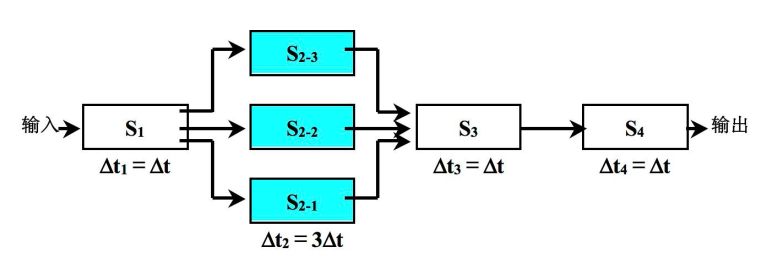
瓶颈部分：当流水线各个功能段执行时间不相等时，流水线吞吐率主要由执行时间最长的那个功能段决定，这个段就是流水线瓶颈部分。

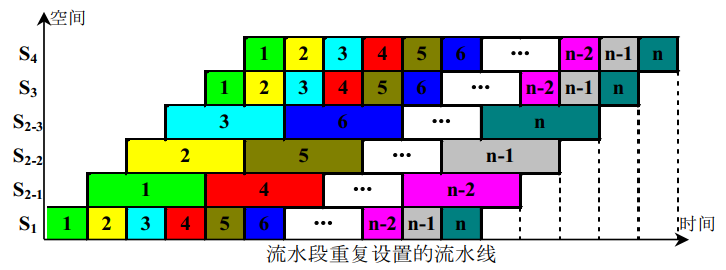
流水段各段执行时间不相等（瓶颈部分）的解决办法：

1. 将瓶颈部分再细分（如果可分的话）

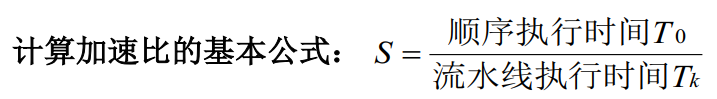


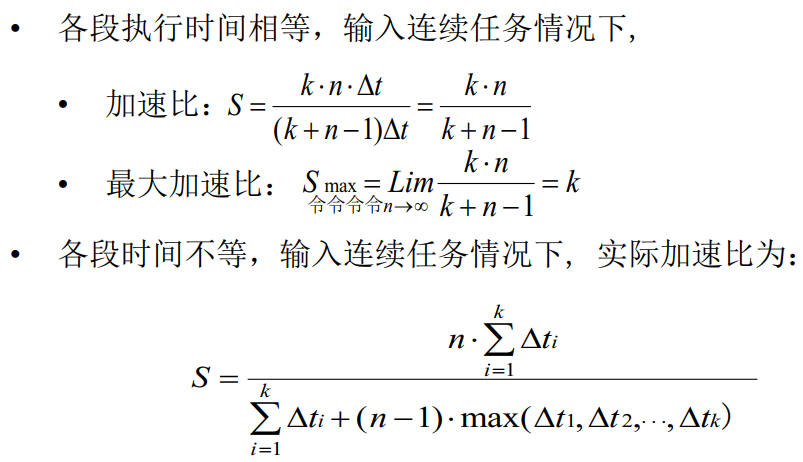
1. 瓶颈流水段重复设置：增加分配器和收集器（增加该段部件）





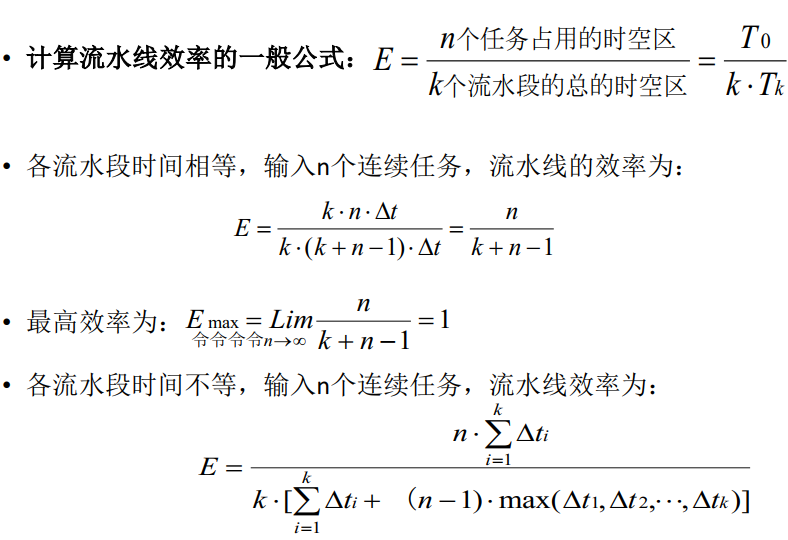
加速比：





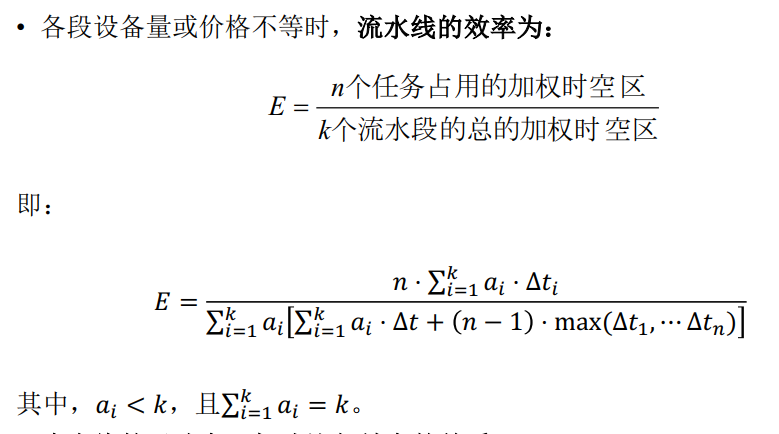
当流水线段数增加时，需要连续输入的任务数也必须增加

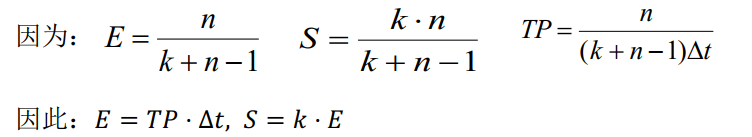
效率：



n个任务所占的时空区就是顺序执行n个任务所使用的总的时间T0

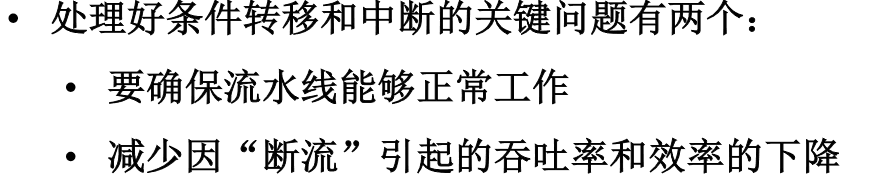
Tk是流水线完成n个任务所使用的总时间（分母部分是完成n个任务所用的时间与k个功 能段所围成的总面积，而分子部分是n个任务实际上占用的有效面积）





1. 非线性流水线的调度
2. 全局相关

由条件转移或者程序中断引起的相关称为全局相关



条件转移指令可能采取的措施：

1. 延迟转移技术和指令取消技术

指令延迟技术是在遇到转移指令时，依靠编译器把一条或者几条没有数据相关和控制相关的指令调度到转移指令的后面

应该尽量减少指令取消

1. 静态转移预测技术

在处理机的硬件和软件设计完成之后，转移预测的方向就已经确定了

1. 动态转移预测技术

根据近期是否转移成功的历史记录来预测下一次转移的方向

（在指令cache中记录转移历史信息；转移目标地址缓冲栈；转移目标指令缓冲栈）

1. 提前形成条件码

在运算完成之前或者在运算中间就形成条件码

程序中断可能采取的措施：

不精确断点：让已经进入流水线的所有指令都执行完成，断点就是最后进入流水线的那条指令的地址

可能的问题：程序的执行结果可能出错；在程序执行过程中通常要设置断点，程序员很难调试

精确断点：设置一定数量的后援寄存器，把整个流水线中所有指令的执行结果和现场保存下来。

1. 向量处理机
2. 什么是向量处理机

向量处理机是具有向量数据表示和向量指令系统的处理机。

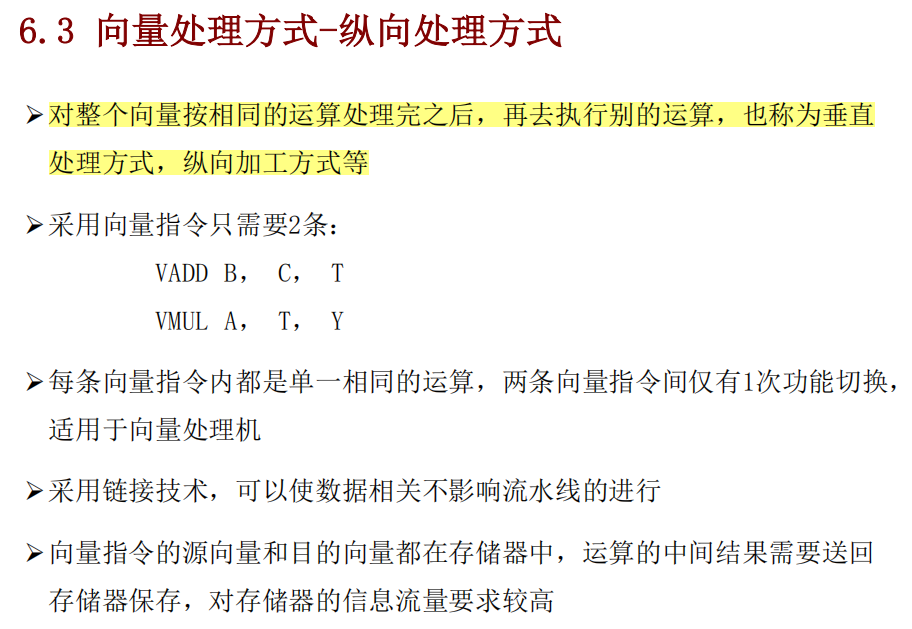
向量处理机是解决数值计算问题的一种高性能计算机。

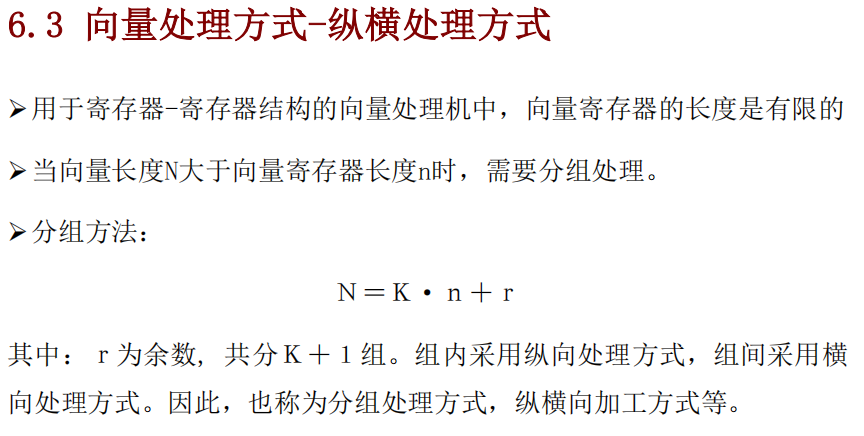
1. 向量处理方式 横向纵向纵横

横向处理方式，又称为水平处理方式，横向加工方式等。向量计算是按行的方式从左至右横向地进行。

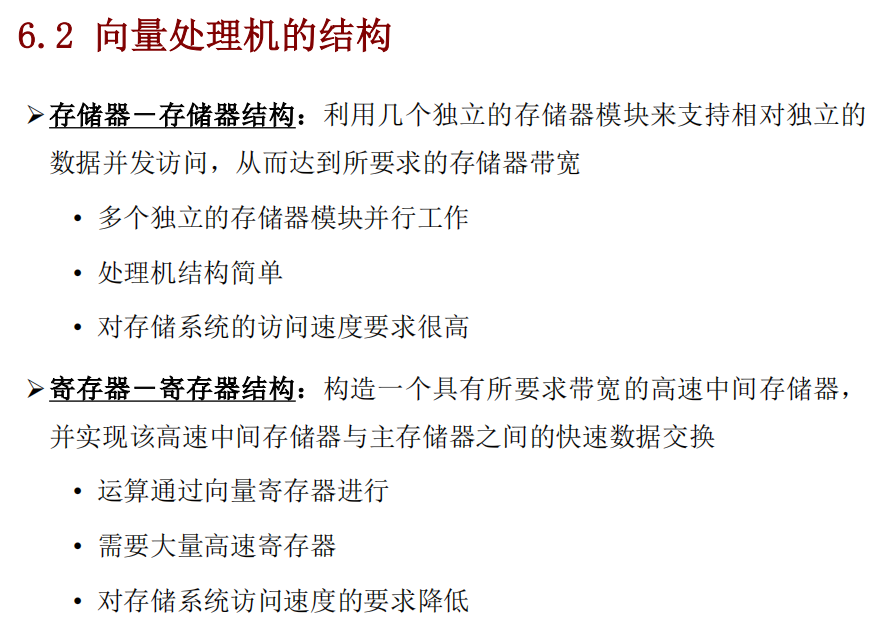
纵向处理方式，又称为垂直处理方式，纵向加工方式等。向量计算是按列的方式自上而下纵向地进行。

纵横处理方式，又称为分组处理方式，纵横向加工方式等。横向处理和纵向处理相结合的方式。





1. 向量处理机结构



1. 向量链接技术



当前一条指令的结果寄存器可以作为后继指令的操作数寄存器时，多条有数据相关的向量指令并行执行，这种技术称为两条流水线的链接技术。

当前一个向量功能部件产生第一个结果并送到结果向量寄存器入口时， 将该结果立即送往下一个功能部件的入口，开始后续的向量操作。此后依次得到的中间结果都按此处理。这样，前面功能部件的结果元素一产 生，就可以立即被后面的功能部件所使用，而不用等结果向量全部产生后再使用。



1. 性能评价掌握 会计算

1. 编队