流水线

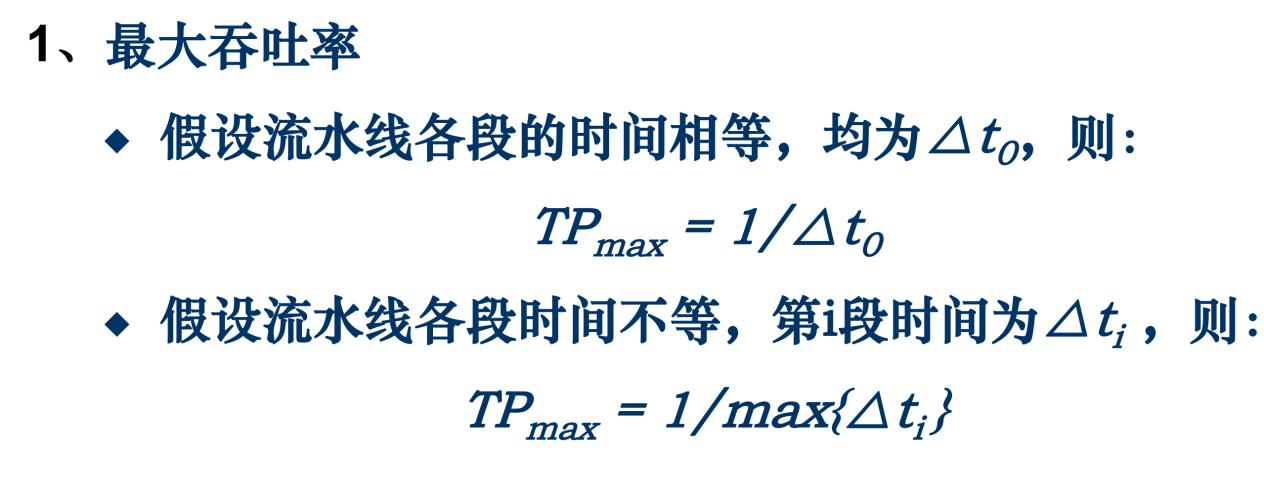
6.2流水线性能分析

6.2.1 吞吐率

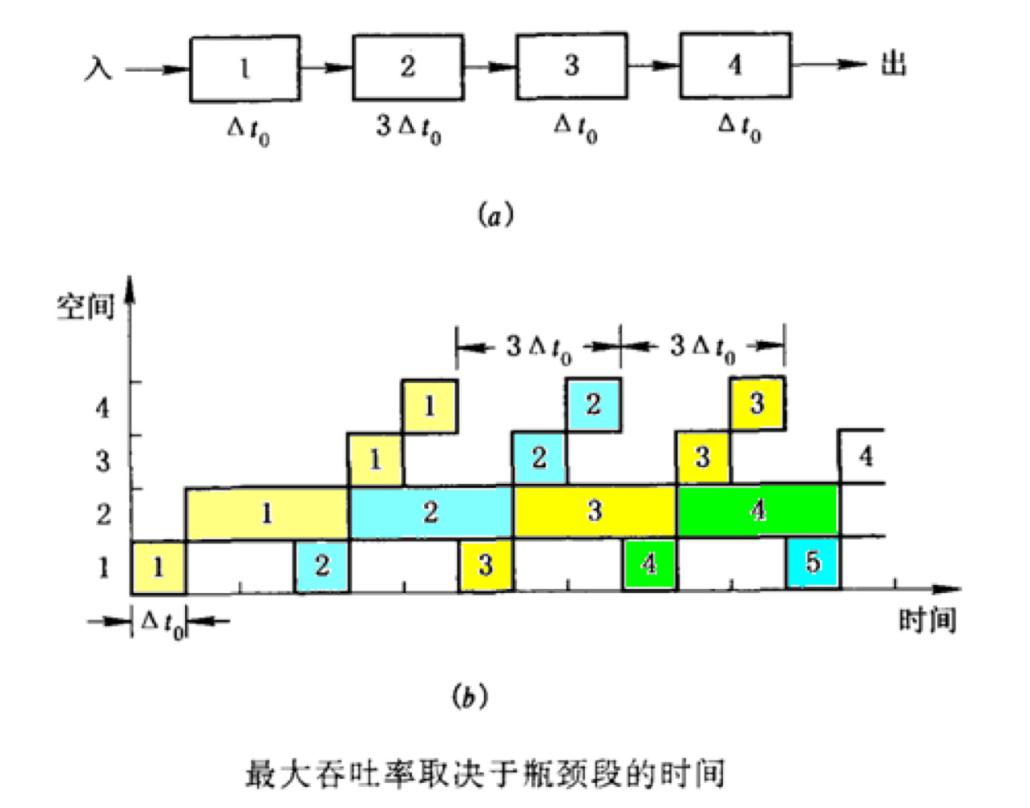
吞吐率是指单位时间流水线完成的任务数或输出结果的数量

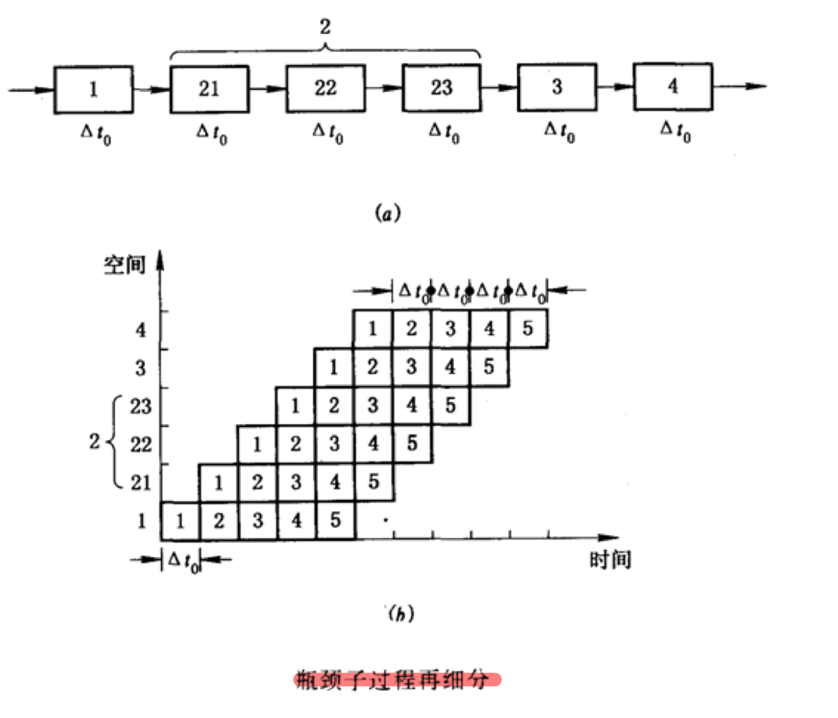
TPmax 最大吞吐率 流水线达到稳定状态后的吞吐率

TP 流水线由m段组成，完成n个任务的吞吐率称为实际吞吐率

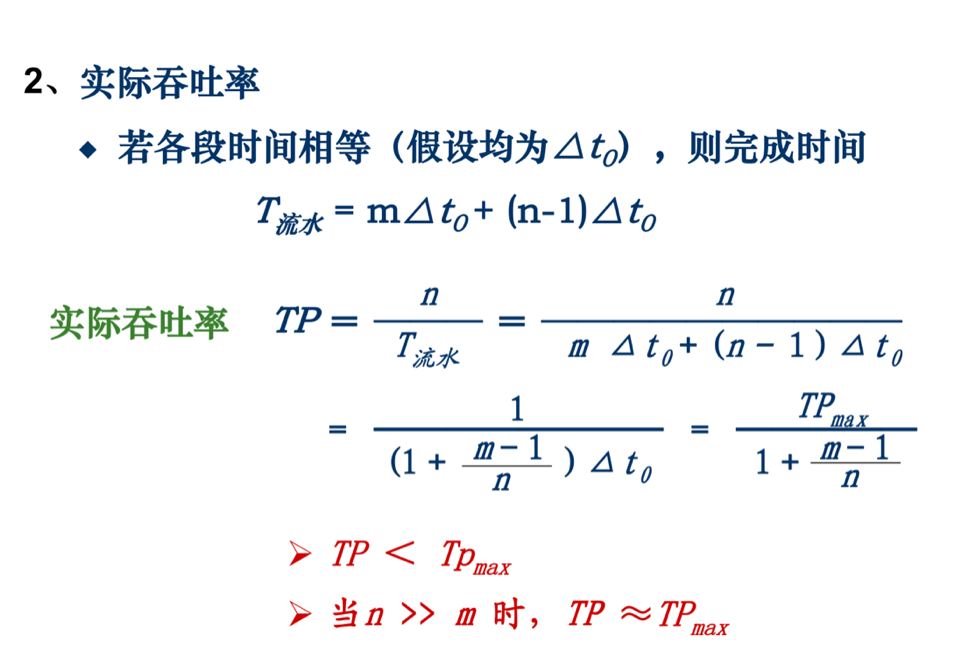


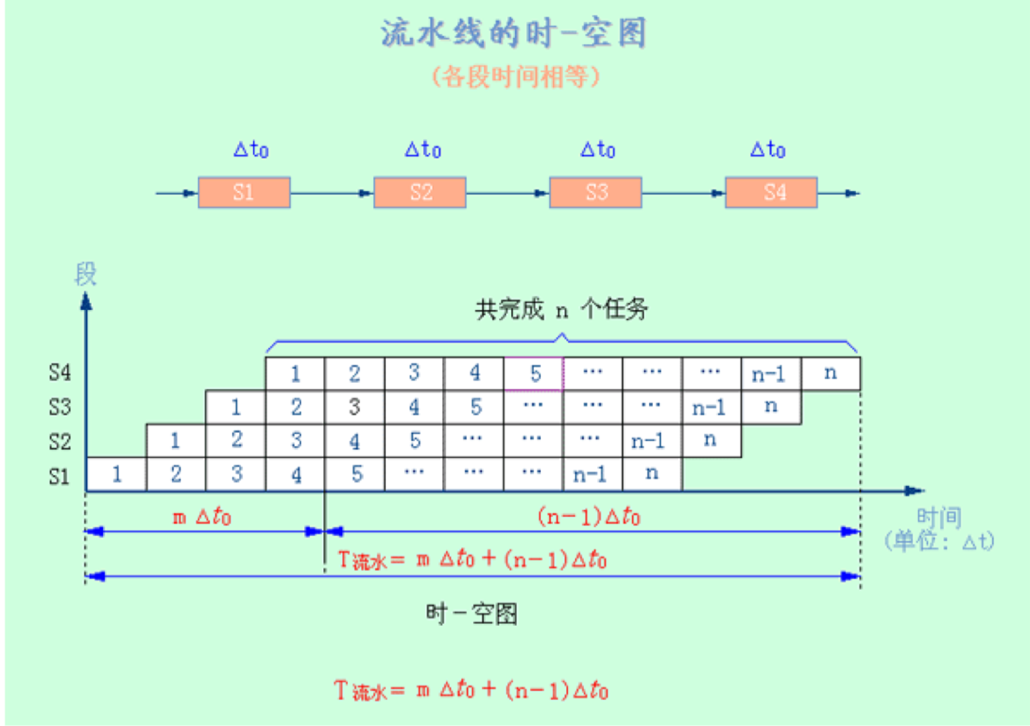
最大吞吐率取决于流水线最慢一段所需的时间，该段是流水线的瓶颈，可以通过细分瓶颈段和重复设置瓶颈段的方式来消除瓶颈

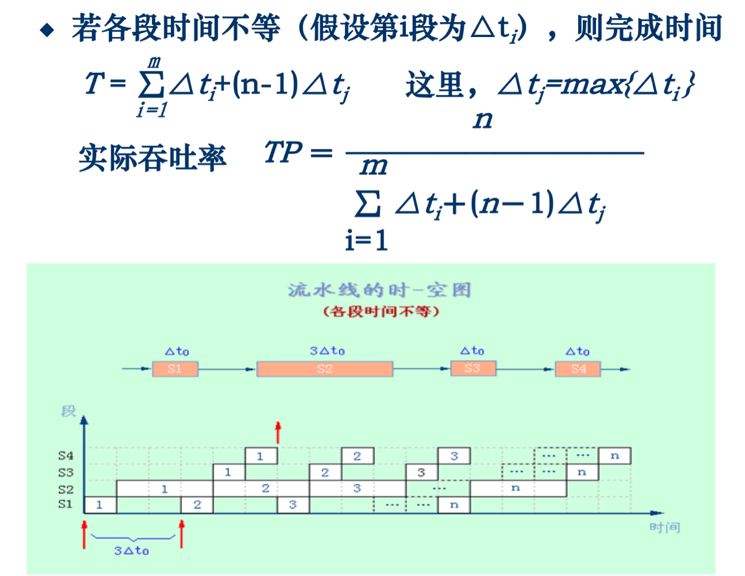








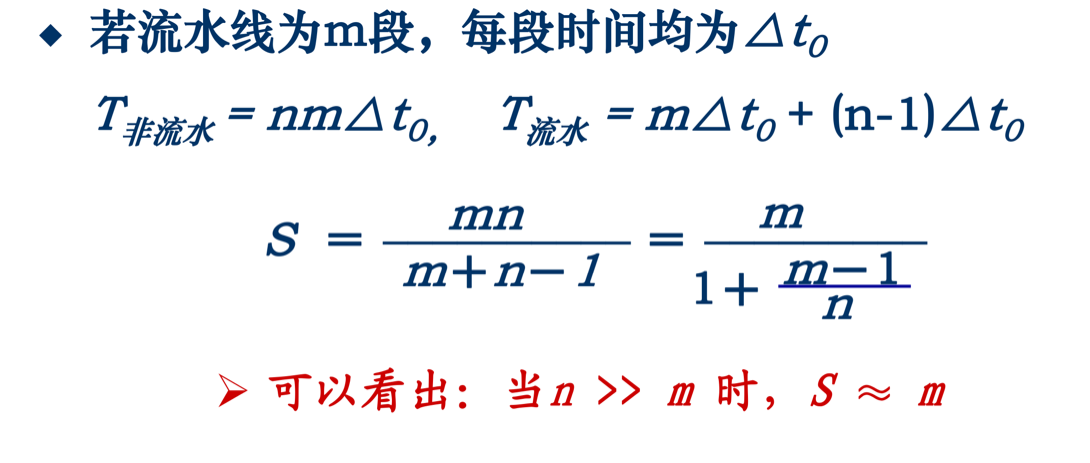




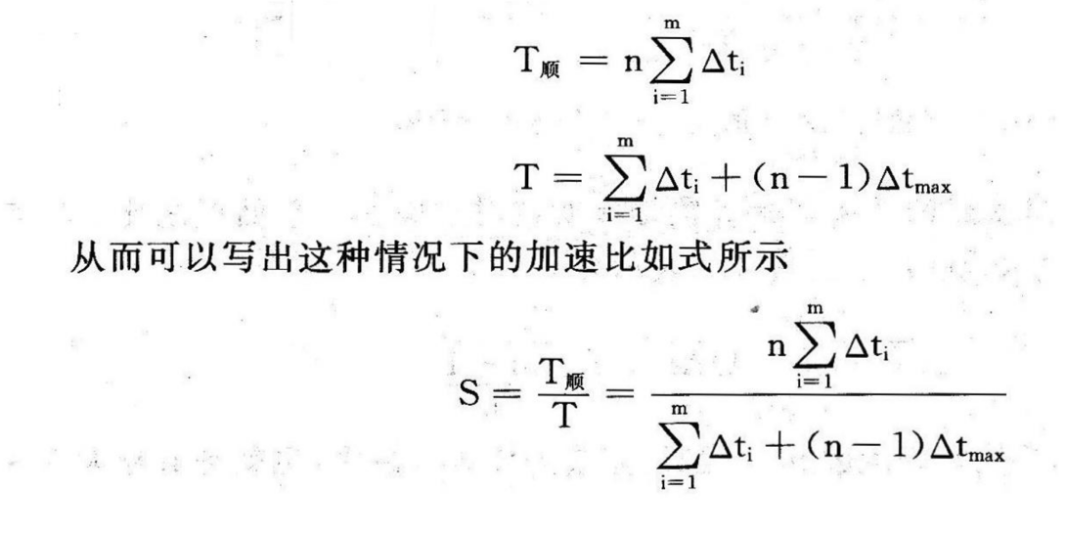
6.2.2 加速比

流水线速度与等功能的非流水线的速度之比

加速比S = T非流水/T流水

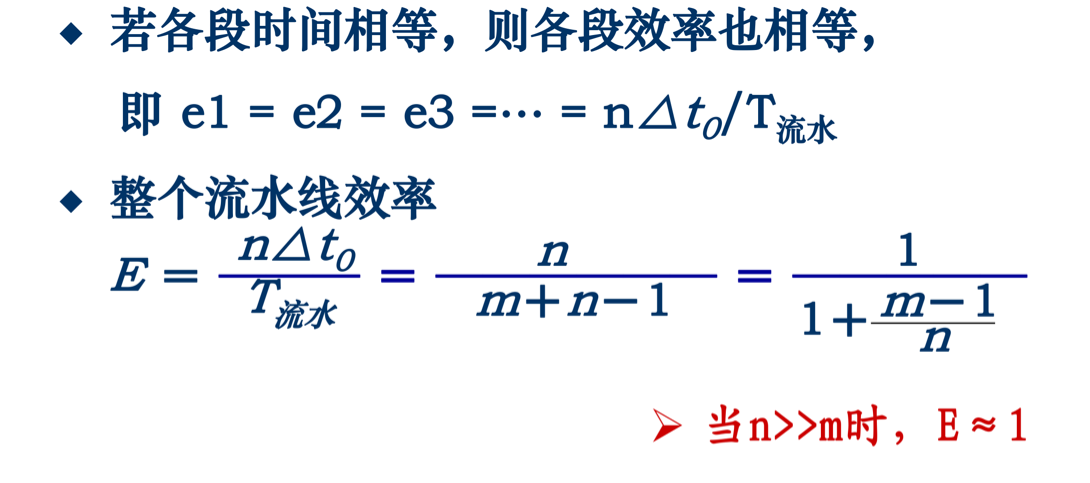


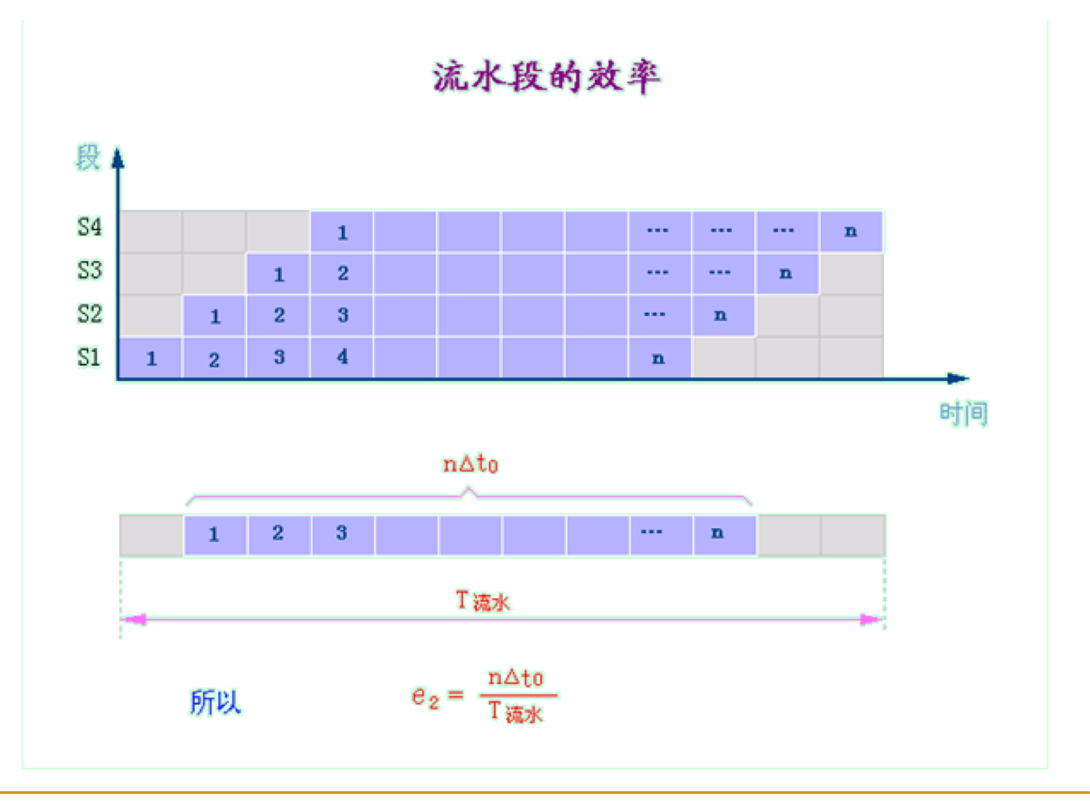
如果各段时间不相等的流水线



6.2.3 效率

效率是指流水线的设备利用率



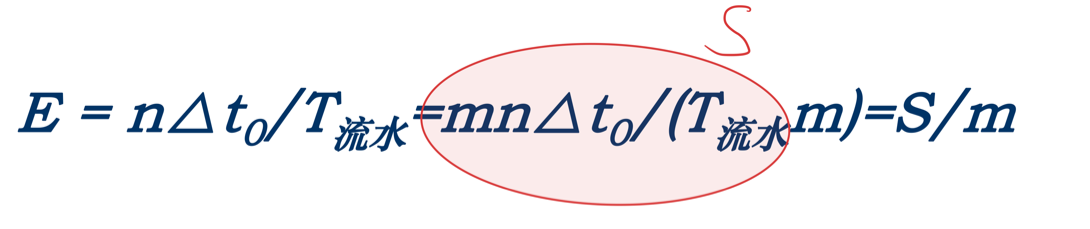


实际上效率就是n个任务所占时空区与m个段总的时空区之比

所以各段时间不等时：E = n个任务占用的时空区/m个段占用的时空区

效率、吞吐率和加速比的关系

1 效率是实际加速比S与最大加速比m之比



2 delta t0不变 效率与吞吐率成正比，提高效率也会提高吞吐率



3 m和delta t0不变，加速比和吞吐率成正比，提高加速比，吞吐率也提高



流水线性能问题总结：

流水线不能减少单条指令执行时间，只能提高吞吐率。

增加深度可以提高流水线性能，但深度受限于延迟和额外开销

需要使用高速锁存器作为流水线寄存器

指令相关会限制流水线性能

6.3流水线相关

6.3.1概述

流水线相关是指向邻近的两条指令存在某种关联，后一条不能再原先指定的时钟周期开始执行

消除相关的基本方法：暂停

相关的类型：

1 结构相关

指令重叠执行过程中，硬件资源满足不了重叠执行的要求，发生资源冲突

2 数据相关

一条指令需要等待前面指令的结果产生才能继续执行，所以不能与产生结果的指令重叠执行

3 控制相关

流水线遇到分支指令或者其他会改变pc值的指令

6.3.2 结构相关

流水线机器中，为了使各种指令顺利重复执行，要把功能部件流水化，把资源重复设置

产生结构相关的常见原因及解决方法：

功能部件不是全流水（设置所有功能单元完全流水化）

重复设置的资源数量不足（设置足够多的硬件资源，比如相互独立的指令存储器和数据存储器）

但是以上方法硬件代价大，有些设计方案允许结构相关存在，为了降低成本、减少功能单元延迟

6.3.3 数据相关

一条指令的结果还未有效生成就被作为后续指令的操作数

产生原因 流水线重叠执行改变读写顺序，使之与非流水时的顺序不同

数据相关分类

两条指令i和j，访问同一寄存器R，若i先进流水线写后读（RAW） 则会有四种访问顺序

1 写后读(RAW) i写j读

如果j在i完成前就从R读数据，会产生错误

（最常见的数据相关，制约cpu性能）

2 写后写(WAW) i写j写

j在i前完成写操作，产生错误结果

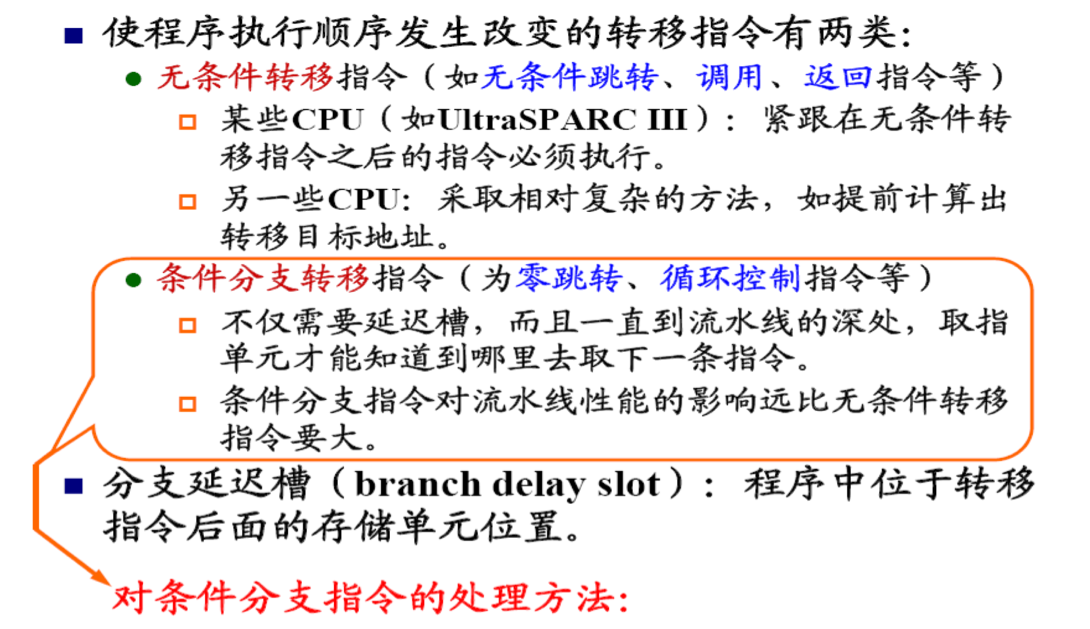
3 读后写（WAR）i读j写

j先将数据写入R，导致i读的结果错误

4 读后读（RAR）

不产生数据相关

6.3.4控制相关



1 冻结流水线

一旦监测到分支指令，就在转移目标地址确定前清除或保存所有紧随分支指令后的指令，当分支指令确定出新的pc值后，才依据新pc值填充流水线（严重影响性能）

2 预取分支目标(prefetch branch target)

条件分支指令被识别后，除了紧随的指令，其分支目标也被预取，并且保存到分支指令被执行，这样如果分支跳转，已预取的目标指令可以立刻执行

3 多流(multiple streams)

在条件分支的两路同时启动取指操作并保存指令，分支指令执行时，真正执行的通路可以立即获得（对预取分支目标的发展）

4 循环缓冲器(loop buffer)

一个高速小型的存储器，保存最近获取的n条指令。分支发生时首先检查分支目标是否在缓冲器中，如果在就从缓冲器中获取指令。

好处：

1 分支发生前，顺序获取的指令已经在缓冲器中。

2 若目标地址仅仅在分支指令的后几个单元处，当分支发生时，目标将已在缓冲器中，例如对if-then, if-then-else语句有利

3 循环缓冲器足够大到容纳一个循环中的全部指令时，这些指令只需要从内存中读出一次

分支预测

1 静态分支预测

1.1预测方法：

预测分支不会发生（预测分支失败，大多数条件分支指令用于出错检测时使用，因为错误不常发生，预测分支失败是合理的）

预测分支总是发生，如果程序包含较多循环，这种预测机制是有效的，因为n次循环仅有一次机会出现分支转移失败。一旦分支转移成功，可以开始从分支目标地址处取指令执行。

由编译器预测：某些处理器中设计了一组新的条件转移指令，这些指令中有一位可以由编译器设置，当编译器认为会发生转移 时(或者不会发生时)就设置该位。当遇到这样的指令时，取指单元就直接按照指令中 的指示采取行动。

剖面法：实际运行该程序（模拟器上），获得不同分支指令发生转移的信息，发送给编译器。

1.2预测错误如何取消已经执行和将要执行的指令

第一种方法：执行，直到它们将要修改计算机的状态，结果保存在临时寄存器中，当得知预测正确时再恢复正常状态。

第二种方法：把马上要被覆盖的寄存器的原值保存在临时寄存器中

2 动态分支预测

通过记录分支指令近期运行历史作为预测依据，提高分支预测准确性

2.1 延迟分支/延迟转移

流水线遇到分支指令时，按照正常方式处理，同时执行延迟槽中的指令，编译器在延迟槽中放入有用的指令，称为延迟槽调度。

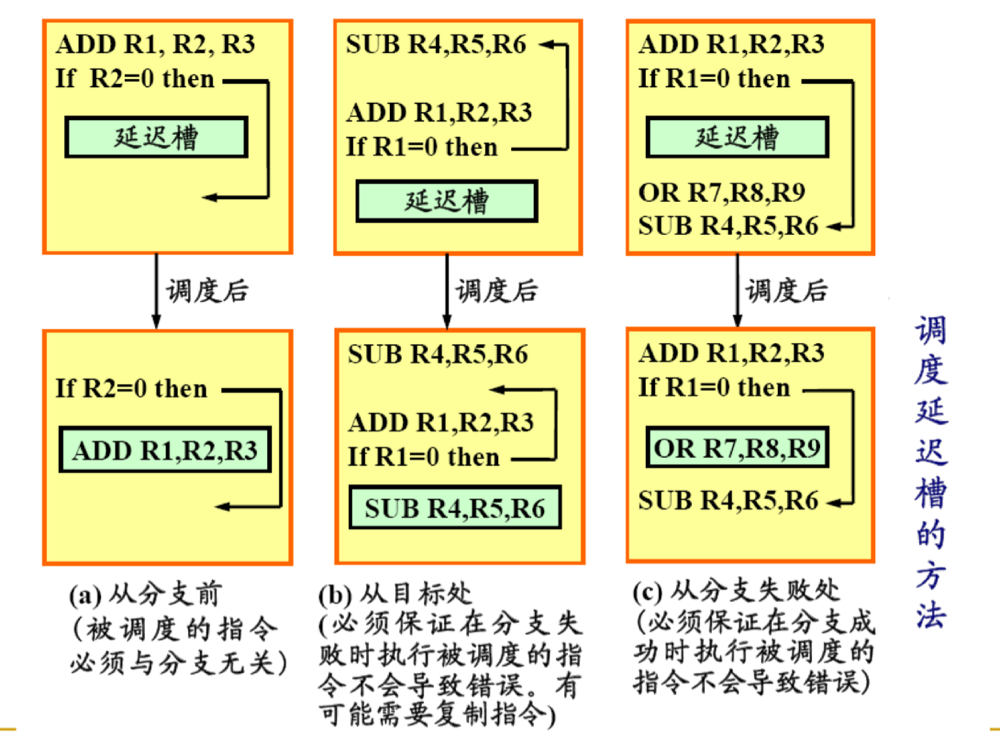
采用该方法的限制：放入延迟槽的指令要满足一定条件，预测器要有预测分支是否成功的能力

三种调度：

1 从分支前调入

2 从目标前调入

3 从失败处调入



全局性相关：转移指令引起的控制相关，对流水线吞吐率效率影响比较大。

局部性相关：数据相关

6.4 指令级高度并行的超级处理器

6.4.1 指令级并行概念

不相关的指令在流水线中重叠执行，这种指令间的并行性称为指令级并行。可以通过依赖硬件，动态地发现和开发指令级并行；或者依靠软件技术，在编译阶段静态发现并行。

6.4.2指令流水线限制

增加指令发射宽度和流水线的深度，会导致CPU功耗（功耗是限制处理器发展首要因素）上升，需要复杂硬件和高频时钟支持。

突破限制的途径：流水线深度、更深层解决流水线中的相关、多核CPU、多指令流水线，提高指令级并行技术

6.4.3提高指令级并行的技术

乱序执行

寄存器重命名

推测执行

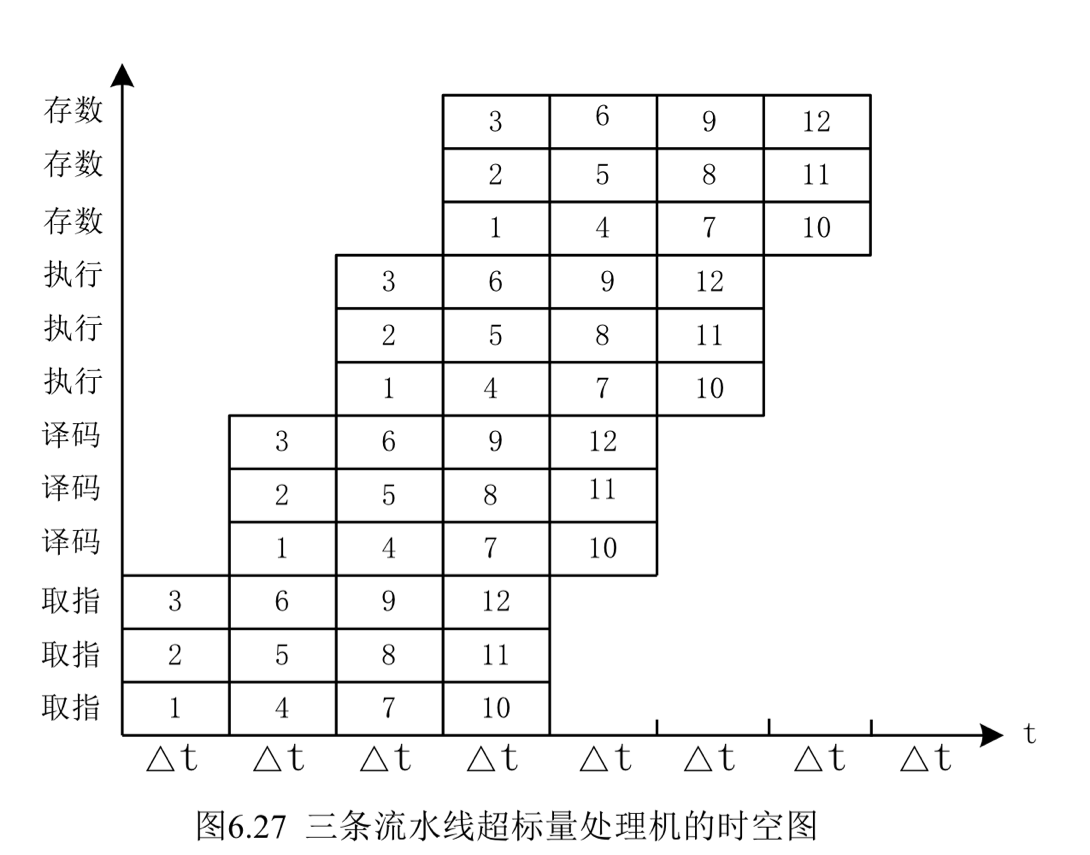
6.4.4 多指令流出（发射）技术

一个时钟周期内流出多条指令 CPI<1 , IPC>1

三种结构：

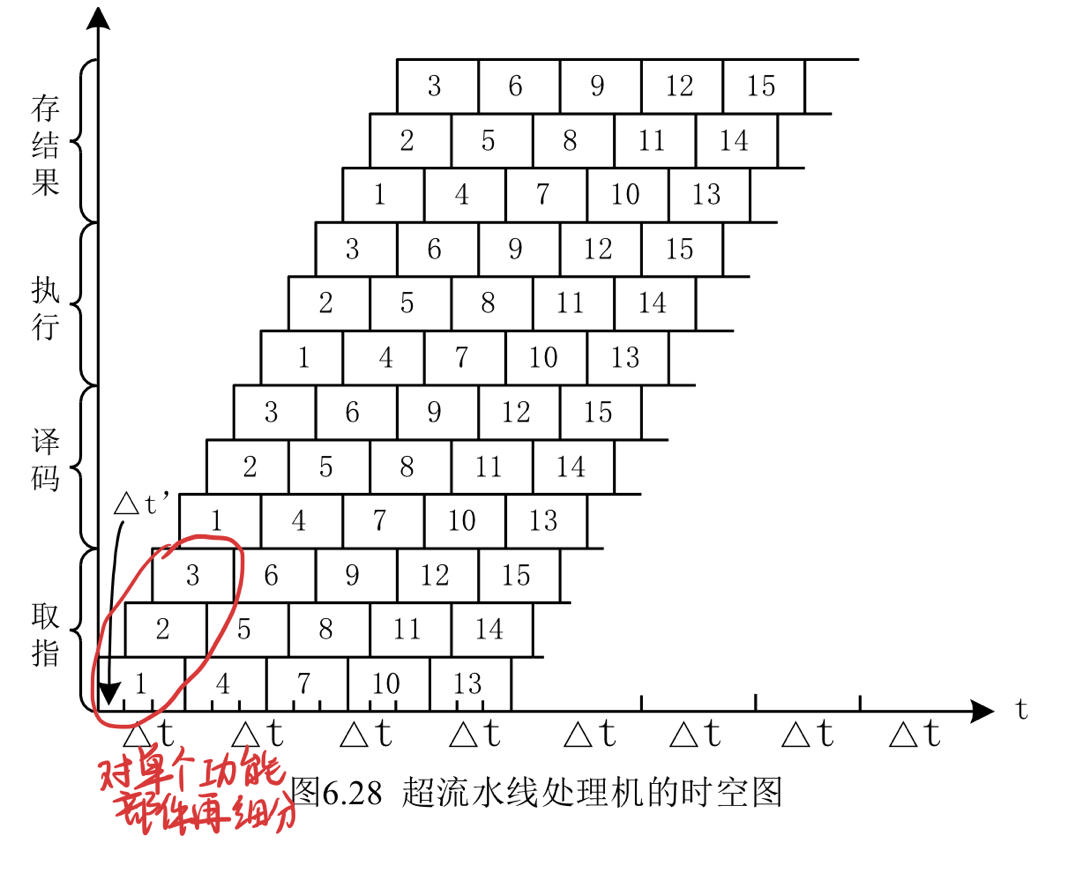
超标量(super scalar) 性能最高

每个时钟周期流出的指令数不等，可以静态或者动态调度



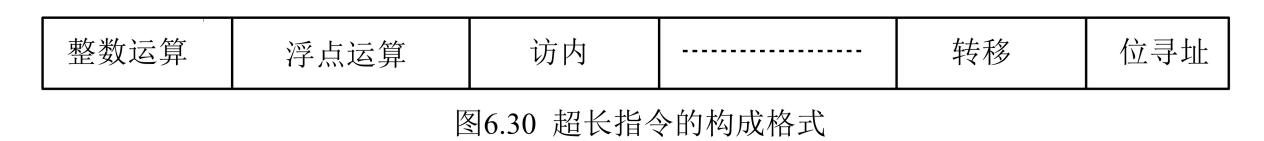
超流水(super pipeline)

将每个功能部件进一步流水化，特别是取指令或指令流出被分解成多个段，使得一个功能部件在一拍中可以处理多条指令。

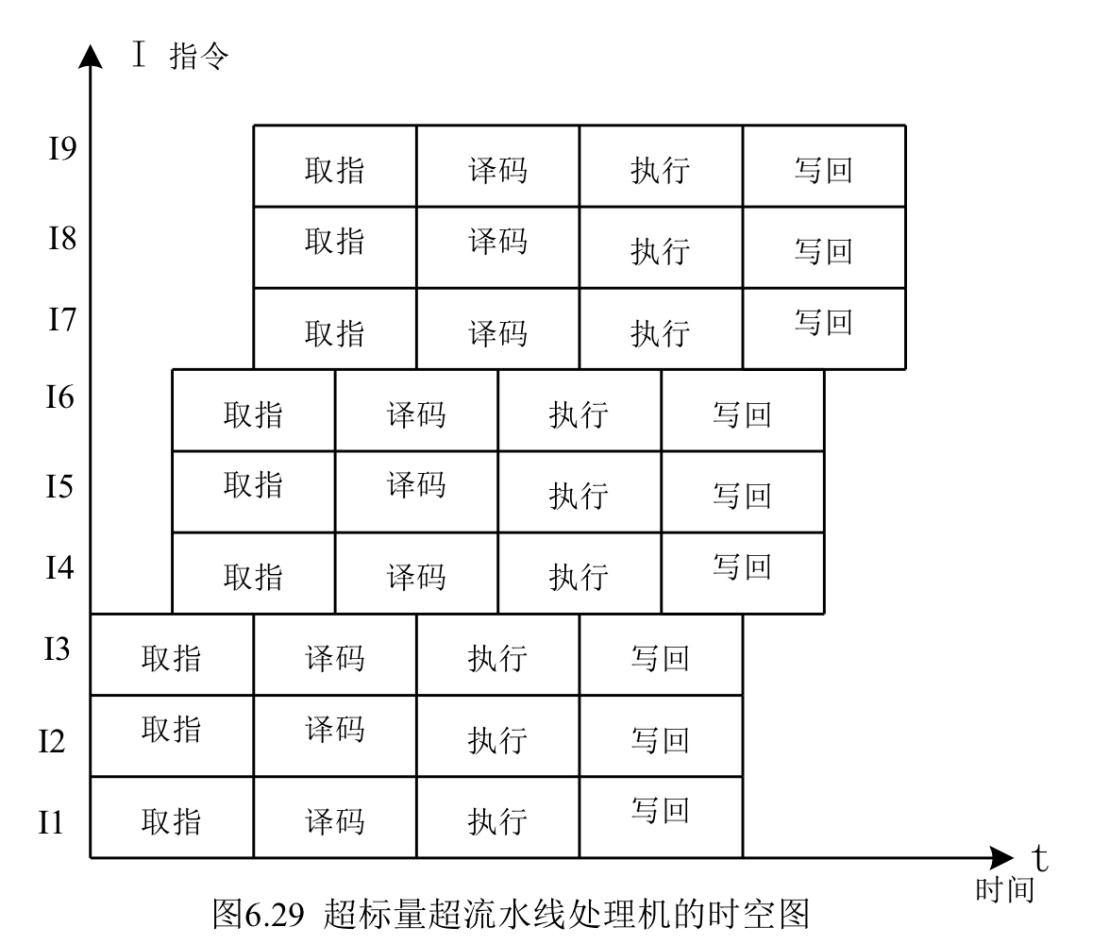


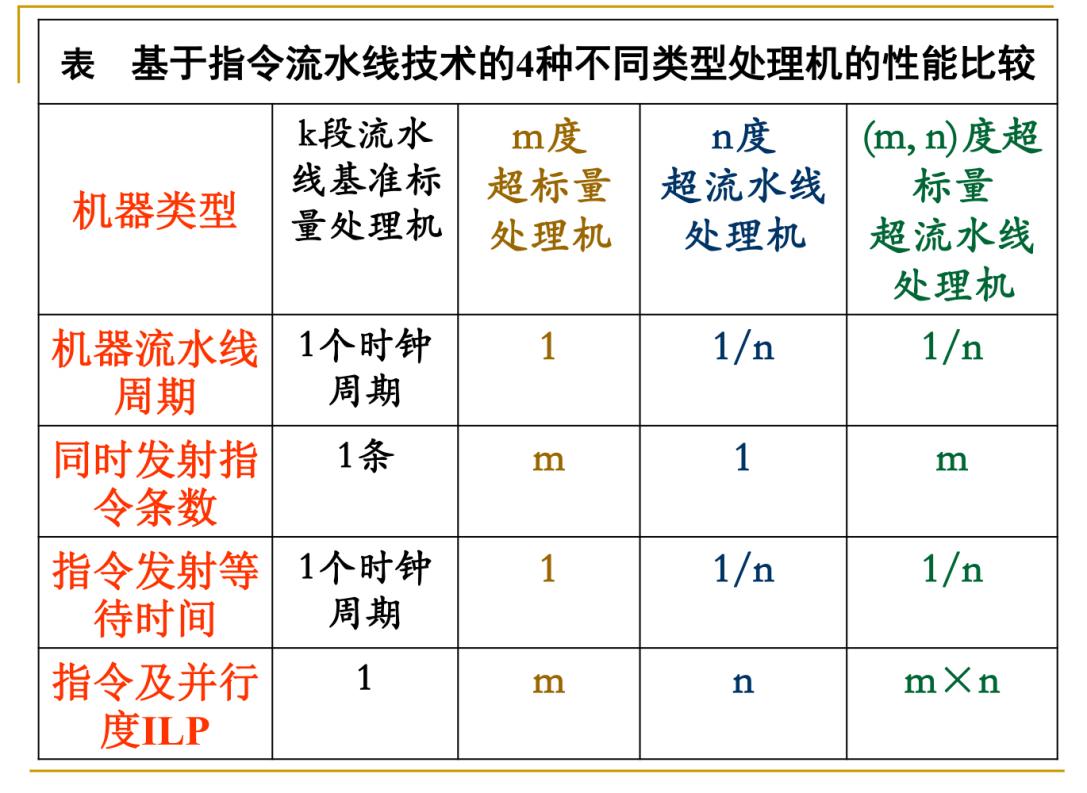
超长指令字VLIW(very long instruction word)

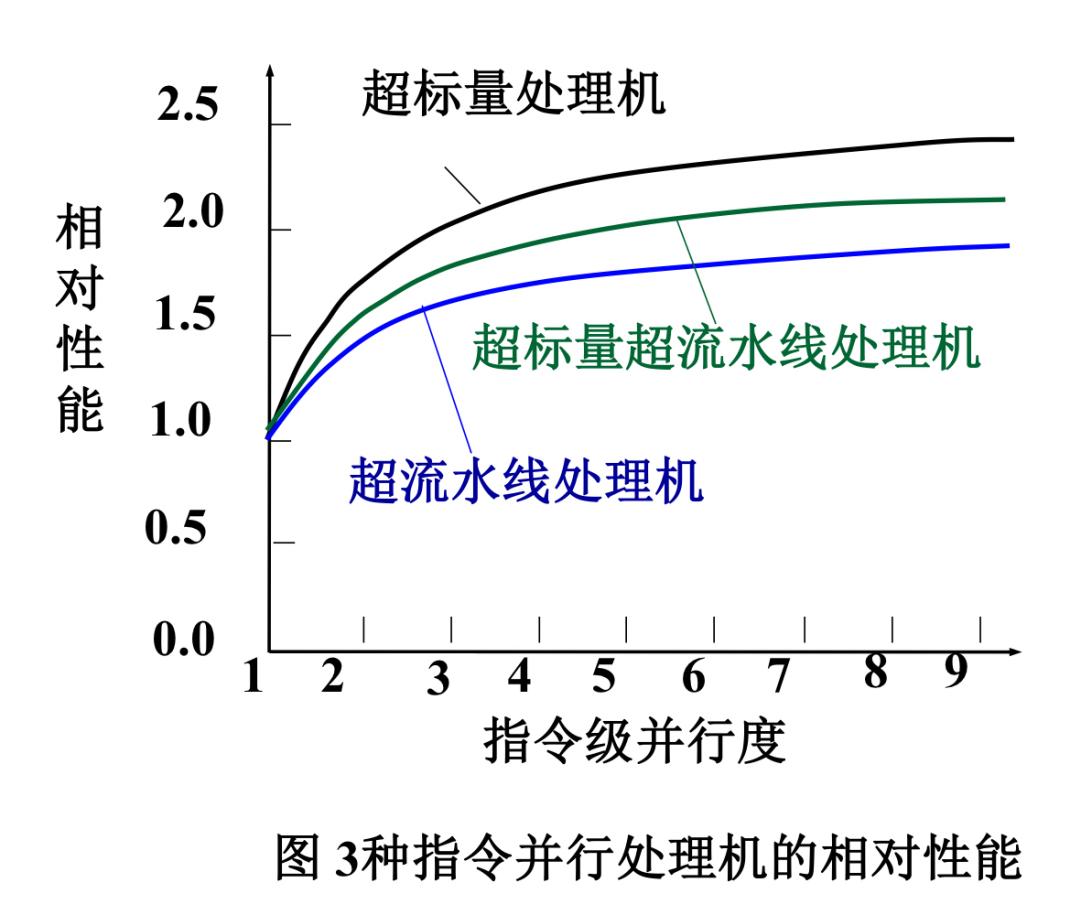
每个时钟周期流出的指令数是固定的，它们构成一条长指令（混合指令包，只能通过编译静态调度）



**超标量超流水**







8输入输出系统

8.1总线

总线系统是计算机系统的重要组成部分。

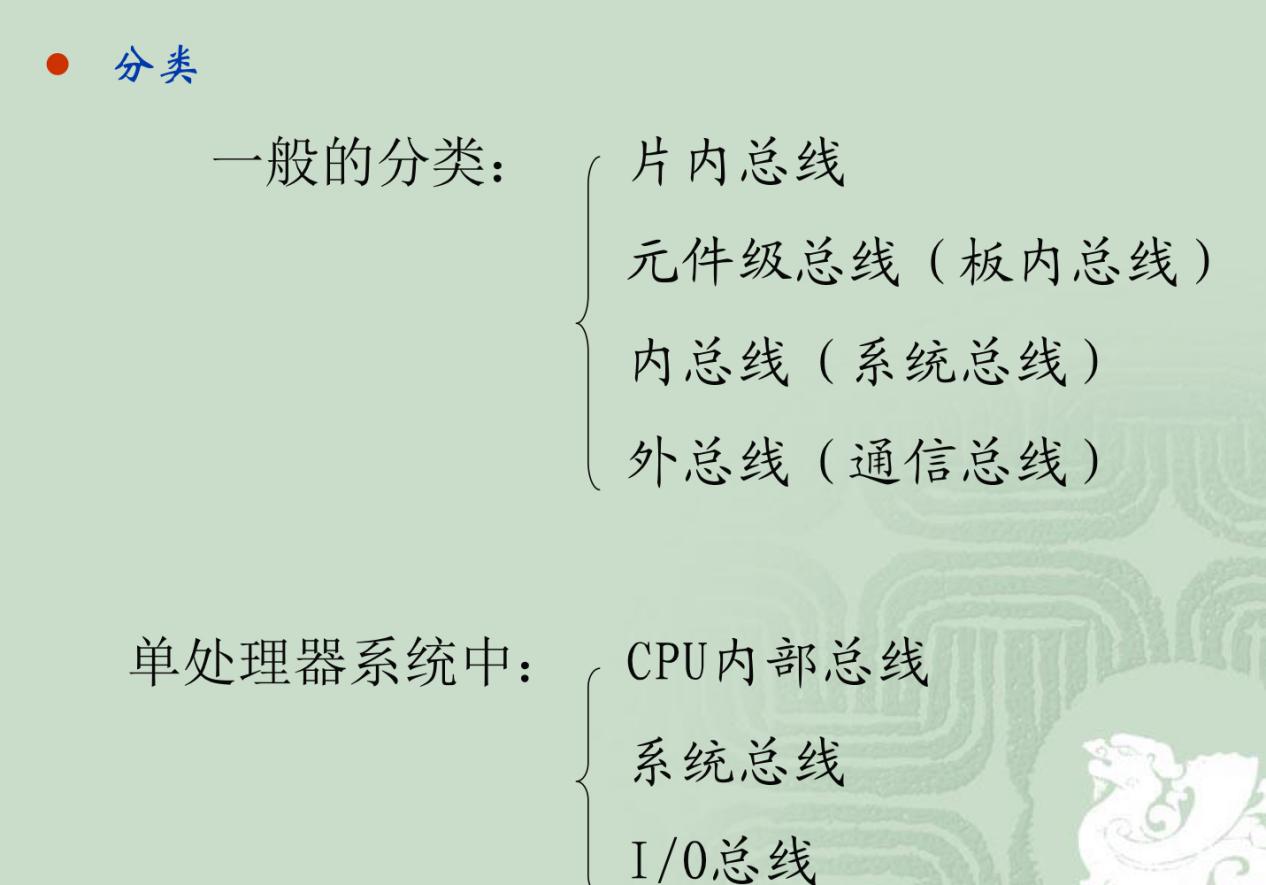
总线技术是计算机系统的重要技术，其结构和性能对计算机系统的性能有很大的影响。

8.1.1 概述

1 总线的分类

广义概念：连接两个及以上数字元件的信息通路。

计算机中的概念：计算机系统多个功能部件之间进行数据传送的公共通路。



2 结构

单总线结构

双总线结构

多总线结构

3 性能描述

总线带宽：总线的最大数据传输率（每秒传输的字节数）。

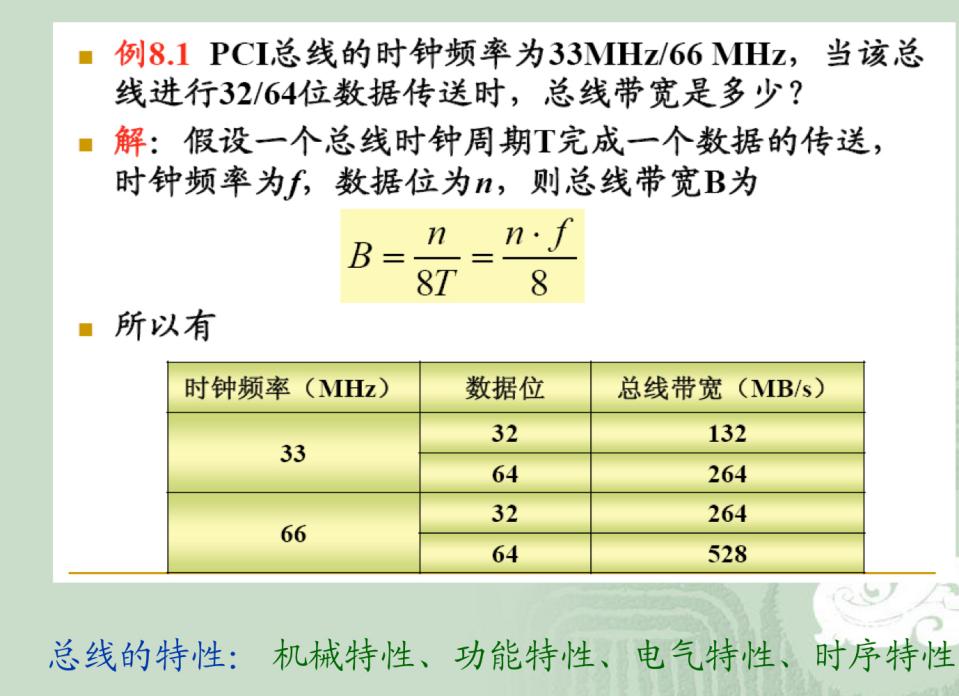
总线宽度：总线的线数，决定了总线占用的物理空间和成本（地址线宽度，数据线宽度）。

总线负载：连接在总线上的最大设备数。

总线频率

是否支持猝发传送等。

总线带宽 = 数据线宽度 \* 总线频率



8.1.2 总线的信息传送

1 传送方式

串行传送：一个数据的各位利用一条数据线依次进行传输

并行传送：一个数据的各位利用多条数据线同时进行传输

同步总线：包括收发双方公用的时钟（总线时钟）和一个固定的协议（用于与时钟相关联的通信） 同步定时方式不需要应答，时序简单，实现方便。总线速度以慢速设备而定，且总线不宜过长。

异步总线：总线中没有时钟线，使用握手协议，允许总线周期长度可变，支持慢速与快速设备在同一总线连接。总线控制复杂，分为非互锁、半互锁、全互锁。

2 总线的仲裁

计算机各系统功能部件之间在争用资源的基础上进行工作，当多个设备提出占用总线的请求时，以某种方式选择其中一个设备占用总线，需要使用总线仲裁部件。

集中式仲裁

分布式仲裁

8.2 输入输出基本原理

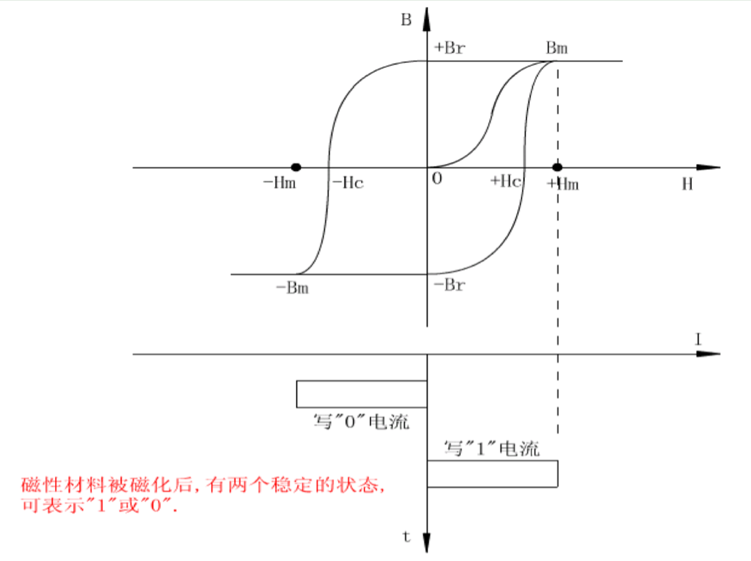
8.2.1 外部设备

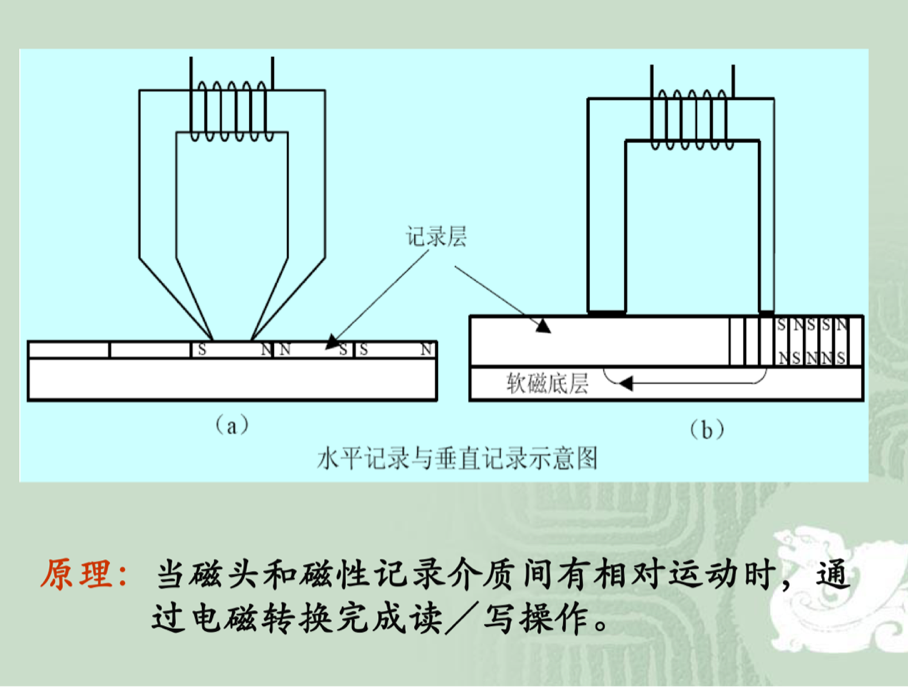
输入、输出设备，外部存储设备（外部存储器、辅助存储器）

外部存储器：磁存储器、光存储器

一 磁记录原理与记录方式

1. 磁记录原理：硬磁、剩磁

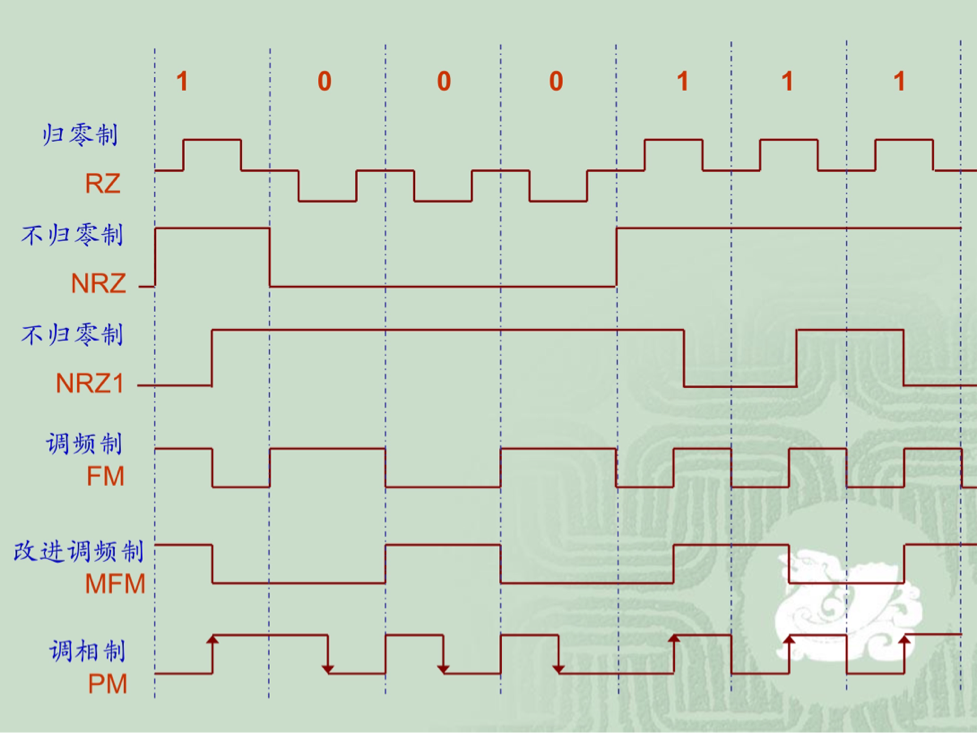




1. 磁记录方式

是一种编码方法，按照某种方案将二进制信息变换成存储介质磁层中一个磁化翻转状态的序列，使I/O控制电路能实现转换。

对提高记录密度可靠性有重要影响。



1. 性能评价

自同步能力：从单个磁道读出的脉冲序列中提取同步时钟脉冲的难易程度。

R = 最小磁化翻转间隔 / 最大磁化翻转间隔

编码效率（记录密度）：位密度与最大磁化翻转密度之比。

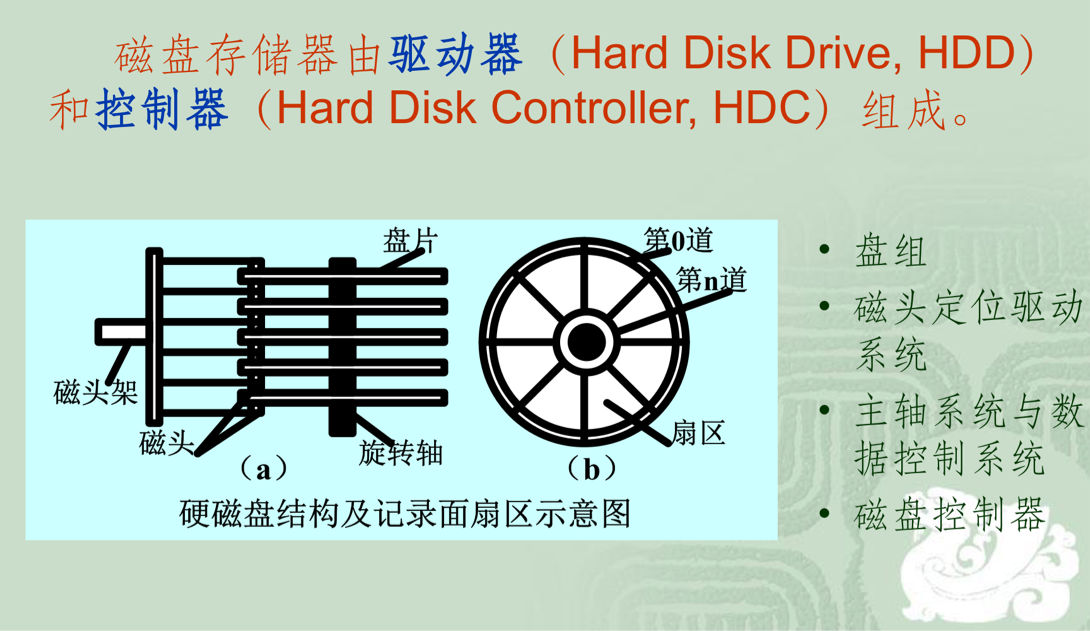
二 硬磁盘存储器

1 硬磁盘存储器的种类

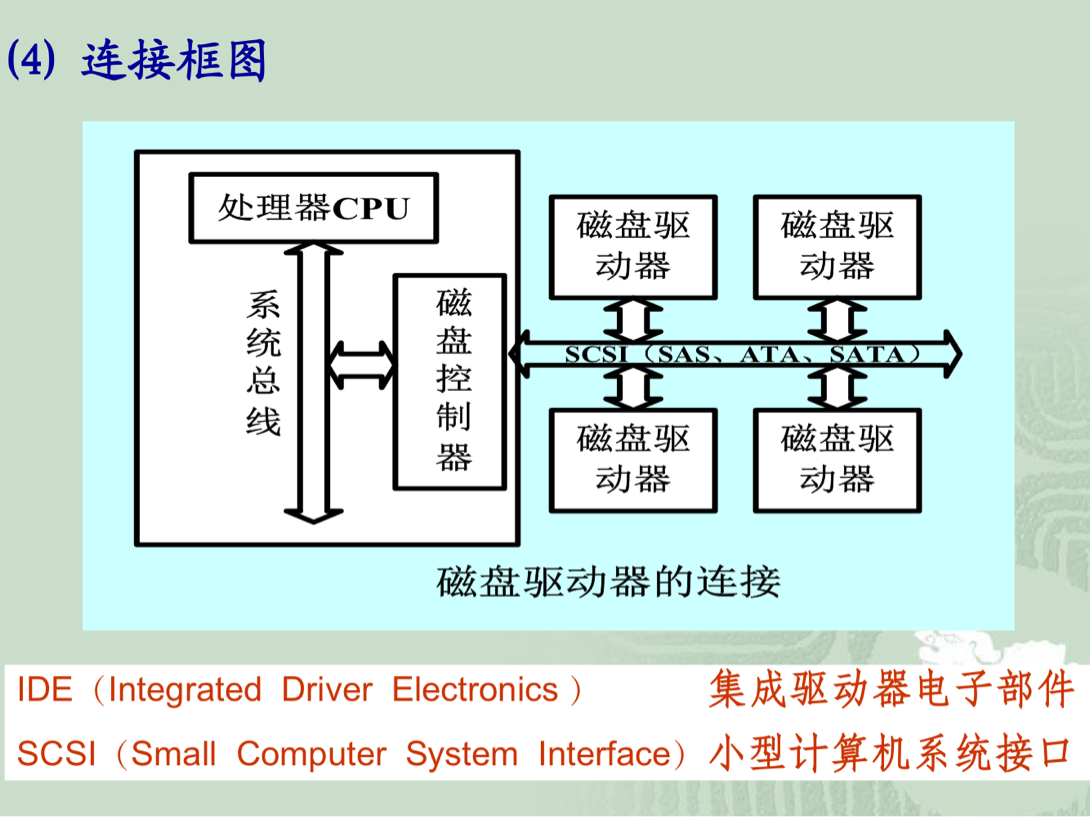
工作方式：固定磁头、移动磁头

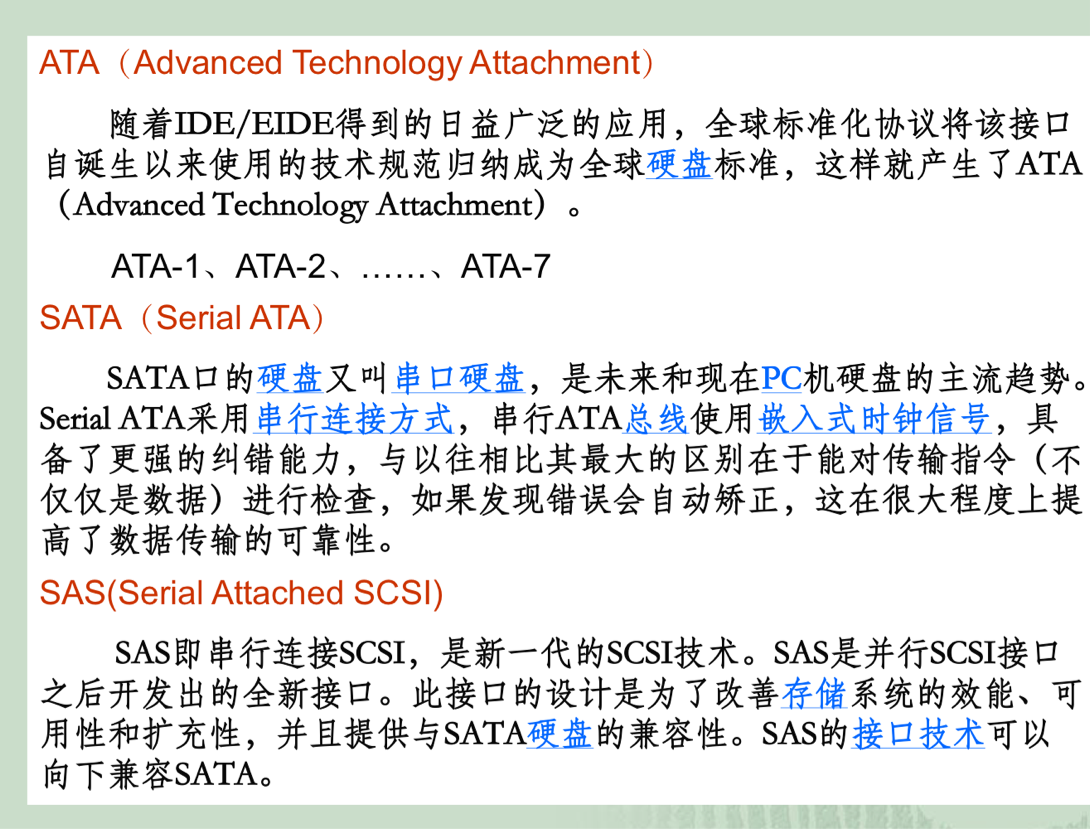
盘片是否可以更换：固定盘存储器、可换盘存储器

2 硬磁盘存储器的结构





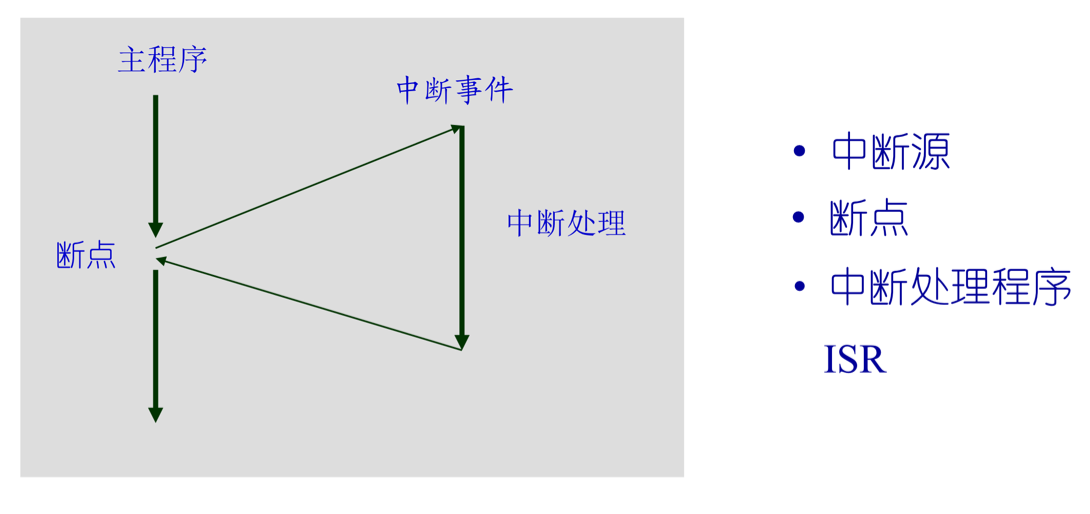


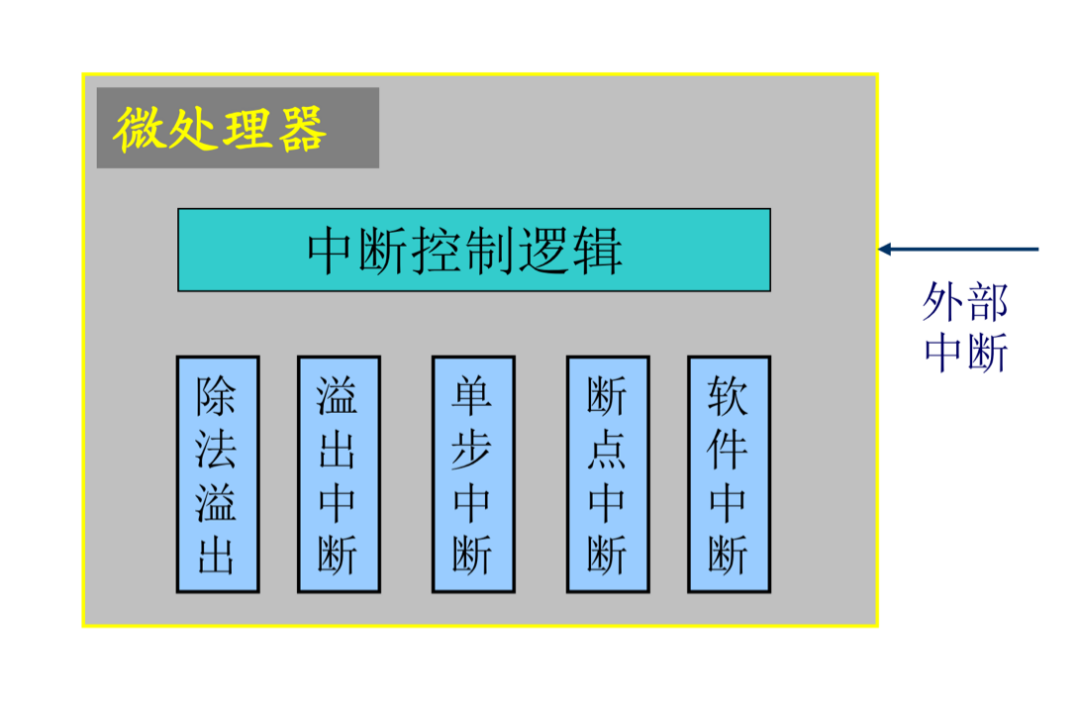


8.3 中断方式

8.3.1 中断的基本概念

中断：某事件引起CPU暂停当前程序的运行，转入对所发生事件的处理，处理结束后又返回程序打断处接着执行的过程。





中断源：引起中断的事件（内部错误、外设请求等）。

外部中断的一般过程：

中断请求：外设->CPU

中断响应：CPU ->外设

中断承认

断点保护

中断源识别

中断处理：中断服务程序 ISR

中断返回：断点恢复 返回

中断的软硬件技术

1 中断请求信号的有效性：信号形式应满足CPU要求

电平类中断请求信号：

保持至CPU发现

CPU响应后撤销

边沿类中断请求信号：

中断请求信号的锁存和撤销

2 中断响应的的条件（以可屏蔽外部中断为例）

CPU对内部中断源提出的中断请求必须接受，对外部中断源提出的中断请求是否响应取决于外中断类型和响应条件。

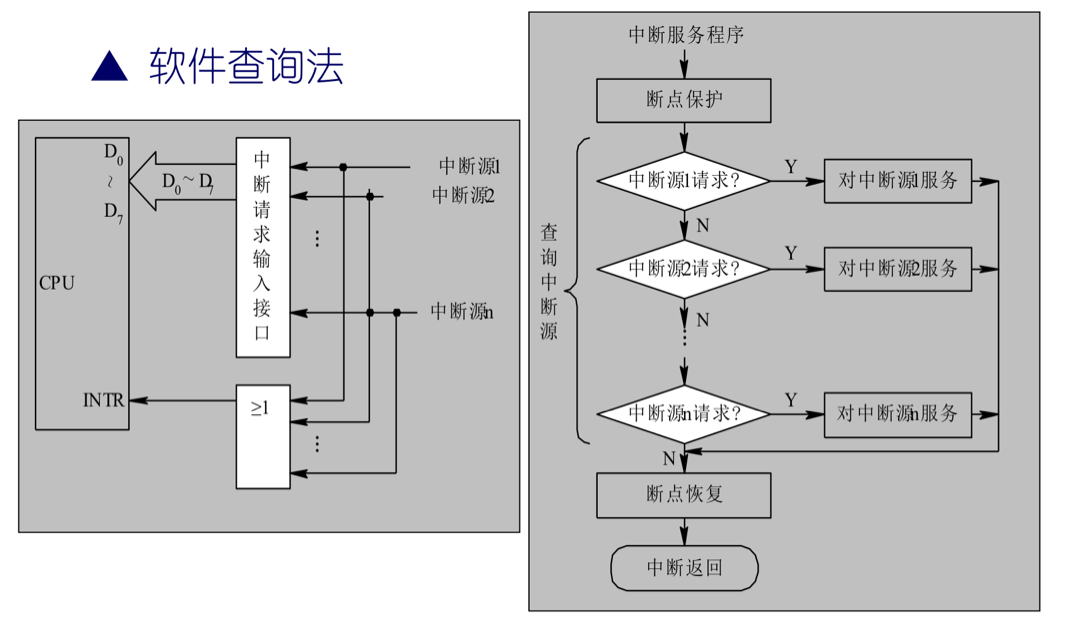
一条指令结束：CPU在每条指令的最后一个时钟周期检测中断请求。

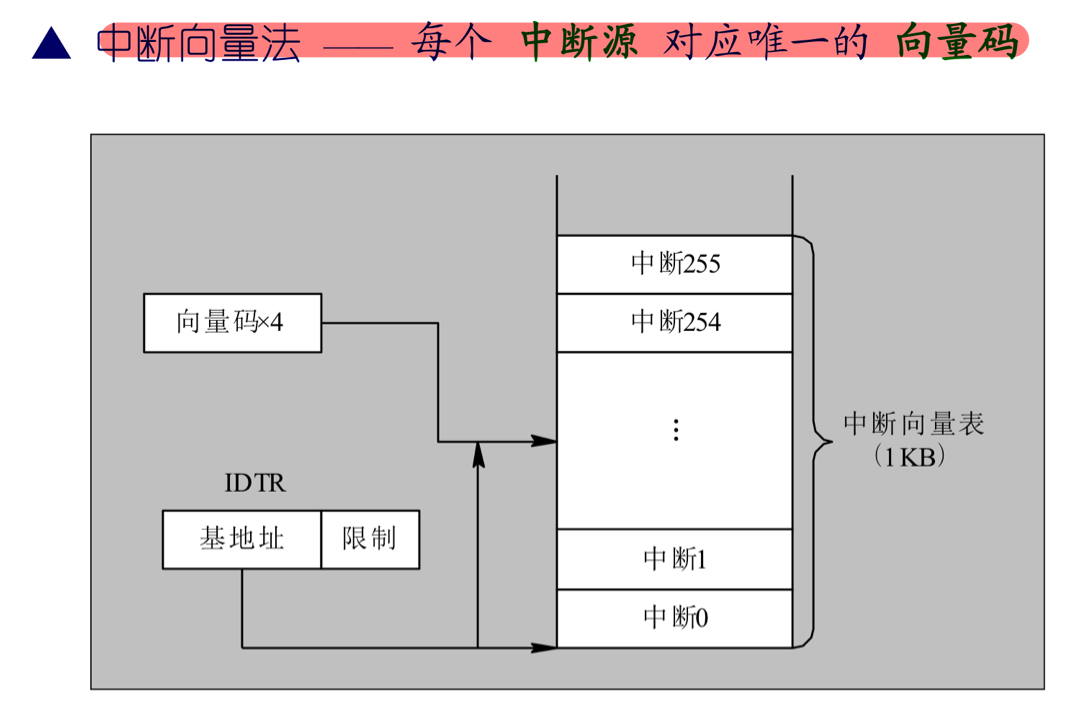
特殊指令

CPU开中断

无更紧迫的事务

3 中断源识别





4 断点保护及恢复

断点信息：断点地址状态。

断点保护：保护断点信息

PSW、CS、IP压栈 （CPU硬件自动完成）

其他寄存器（AX、BX等）压栈 （中断服务程序完成）

断点恢复：恢复断点信息

PSW、CS、IP弹栈 （CPU硬件自动完成）

其他寄存器（AX、BX等）弹栈 （中断服务程序完成）

5 中断优先级

1 软件查询

2 利用可编程中断控制器（PIC）

3 硬件链式优先级

4 硬件优先级编码比较电路

6 中断嵌套

当高优先级中断可以中断低优先级中断时，会产生中断嵌套。

