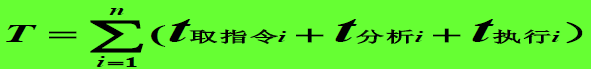
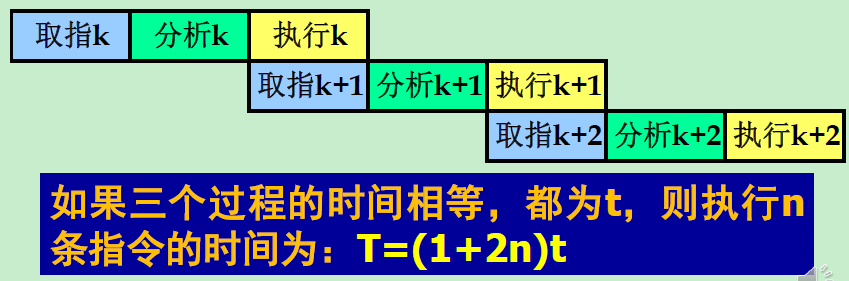
# 第五章 流水线和向量处理机

*Summarized by Pengxiang Li*

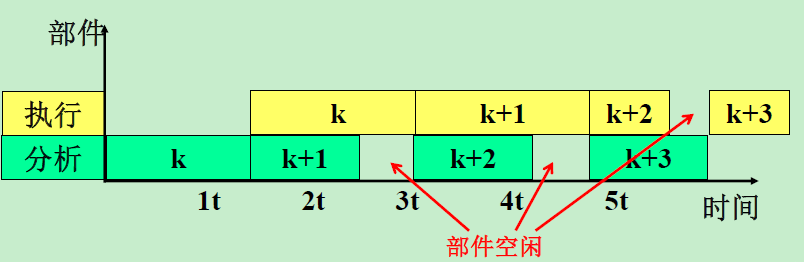
## 5.1 重叠方式

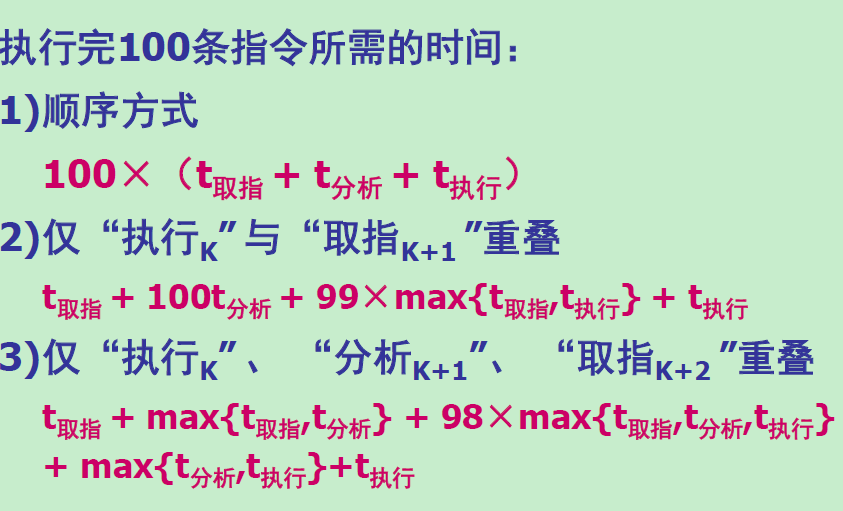
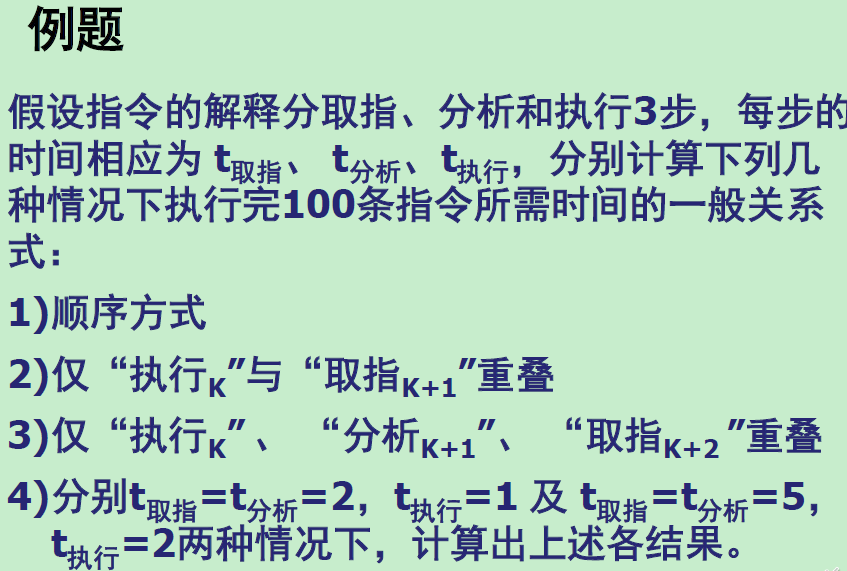
### 5.1.1 重叠原理和一次重叠

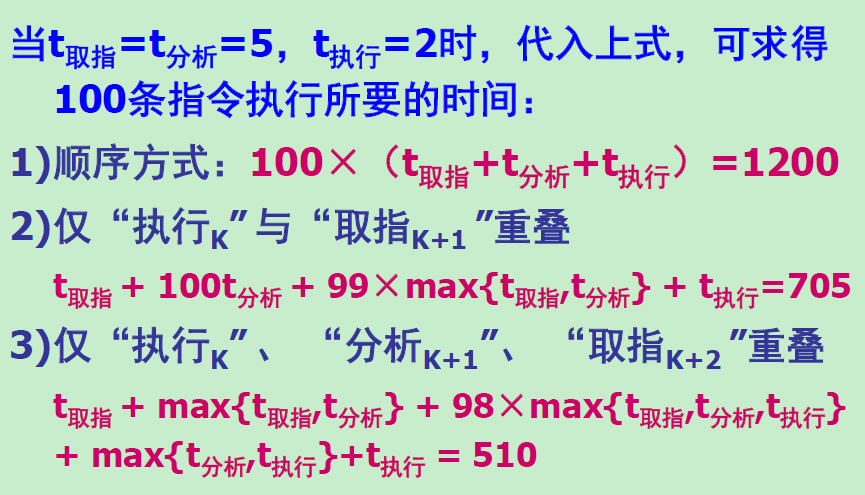
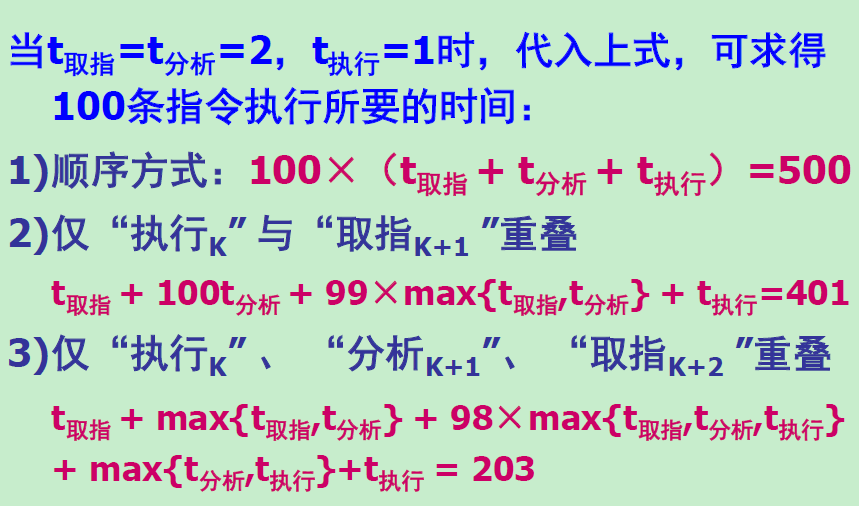
1. 基于时间重叠技术
2. 重叠解释执行两条或多条指令，加快程序的执行
3. 顺序解释执行方式：
   1. 指令之间串行执行 其指令内微操作也串行
   2. 执行n条指令所用的时间为：
      1. 
      2. 如果每段时间都为t，则执行n条指令所用的时间为： **T=3nt**
   3. 优点：简单、易于实现
   4. 缺点：速度难以提高，部件利用率低
4. **一次重叠执行方式**
   1. **每次只重叠执行 两条 指令 故称为 一次重叠**
   2. **特点： 在第 K 条指令完成之前就开始处理第 K+ 1 条指令 重叠执行两条指令**
      1. 
   3. **优点：**
      1. **虽不能加快一条指令的执行 但能加快相邻两条指令 以至一段程序的执行加快相邻指令或一段程序的执行 。 指令的执行时间缩短了近二分之一**
      2. **提高了部件利用率**
   4. **缺点：**
      1. **需要增加一些硬件**
      2. **控制过程较为复杂**
5. **二次重叠执行方式**
   1. **每次重叠执行三条指令**



* 1. **如果三过程的时间相等 执行 n 条指令的时间为： T=( 2 +n)t**
  2. **采用二次重叠执行方式能够使指令的执行时间缩短近三分之二。**

1. **重叠执行对计算机组成的要求**
   1. **问题一：功能部件冲突**
      1. **为实现 执行 k 与 分析 k+ 1 的重叠，必须在硬件上保证有独立的取指、指令分析和指令执行部件。**
      2. **解决：花费一定的硬件代价 分别设置独立的 取指部件 、 指令分析部件 、 指令执行部件；**
      3. **三个子过程都有独立的控制器：存储控制器 、指令控制器 、 运算控制器**
   2. **问题二：访存冲突**
      1. **同一时间取指k+1和分析k都要访存，主存在同一时间只能访问一个存储单元。**
      2. **解决方法（四种）：**
         1. **采用 两个独立编址 、 可同时访问的的存储器 分别存放操作数和指令 。 如果再规定执行指令所需要的操作数和执行结果只写到通用寄存器 那么 取指令、 分析指令和执行指令就可以同时进行。 但增加了主存总线控制的复杂性和软件设计的麻烦**
         2. **维持操作数和指令的混存在主存中，但是设置两个cache：指令cache和数据cache*（哈佛结构）***
         3. **维持操作数和指令的混存 采用 多体交叉存储器 采取低位交叉存取方式 让第 k 条指令的操作数和第 k+ 1 条指令存放在不同的存储体。 但这种方法有一定的局限性 不能从根本上解决冲突**
         4. **在主存和指令分析部件之间增设 FIFO的 指令缓冲寄存器 又被称为 先行指令缓冲站 。 利用主存的空闲 预先将下一条或下几条指令取入指缓 而不必访问主存储器 。 这样就能够使取指令 、 分析指令和执行指令重叠起来执行**
   3. **问题三：同步**
      1. **执行 和 分析 所需要的时间常常不完全相同 指令分析部件和指令执行部件经常要相互等待 从而造成功能部件的浪费**
      2. 
      3. **解决方法：采取先行控制技术**
         1. **先行控制 Look ahead 技术的 关键 是缓冲技术和预处理技术 以及这两者的相结合 。**
         2. **预处理技术： 早些取来指令和操作数。把进入运算器的指令都处理成寄存器 寄存器型 RR 型 指令 它与缓冲技术相结合 为进入运算器的指令准备好所需要的全部操作数；**
         3. **缓冲技术： 在工作速度不固定的功能部件之间增设指令缓冲栈 、先行读数栈 、 先行操作栈和后行写数栈等 以平滑它们的工作**
         4. **优点：节省硬件，简化控制**
         5. **缺点：必须注意安排位操作，使得执行和分析时间尽可能相等，从而使得重叠效率较高。**

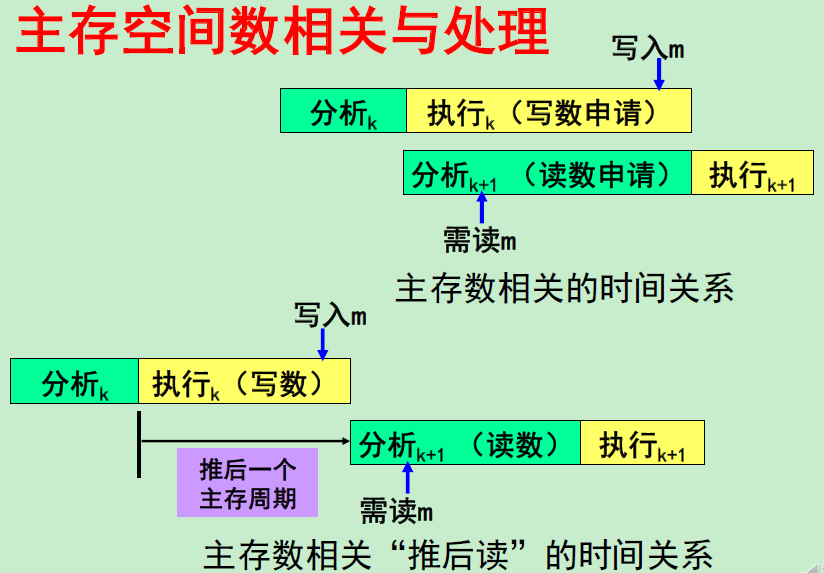




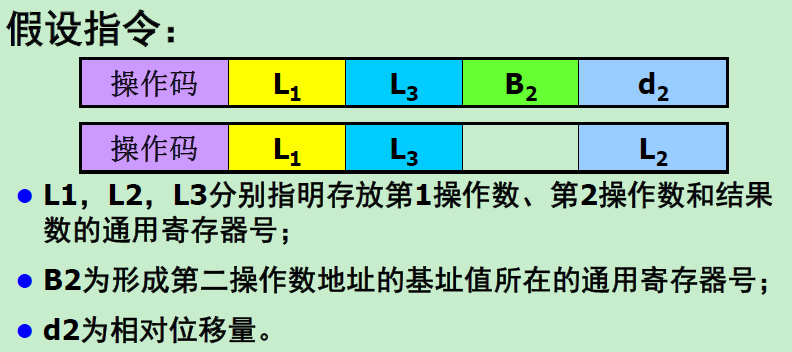
**解题技巧：重叠的时间取最大值，其他根据重叠代入公式。**

### 5.1.2 相关处理

1. **数据相关：**第k指令和第k+1条指令的数据地址之间有关联。例如：第k+1条指令的源操作数地址刚好为第k条指令存放运算结果的地址。数据相关不仅发生在主存空间，也可能发生在通用寄存器空间。
2. **指令相关**：因为指令在程序的执行过程中允许被修改造成的。例如经过第k条指令修改第k+1条指令。
   1. 指令相关的 原因：Von Neumann 型计算机的指令允许修改
   2. 解决方法：
      1. 不允许修改指令
         1. 优点：可以实现程序的可再入和递归调用
         2. 缺点：灵活性较差
      2. 允许修改指令
         1. 改变指令的执行方式，将指令相关转换为数据相关，统一按照数据相关处理。
3. 主存空间数据相关的处理
   1. 主存空间数据相关： 相邻两条指令之间出现对同一单元要求先写后读的关联，可采取延后后续指令对相关单元的读操作的方法。



1. 通用寄存器组数相关的处理
   1. 通用寄存器可能存放操作数、运算结果，也可能存放形成访问操作数物理地址的变址和基址，因此通用寄存器组的相关又有**操作数相关**和**变址/基址的相关**。通用寄存器 数相关与通用寄存器的基址值或变址值相关的处理是不同的

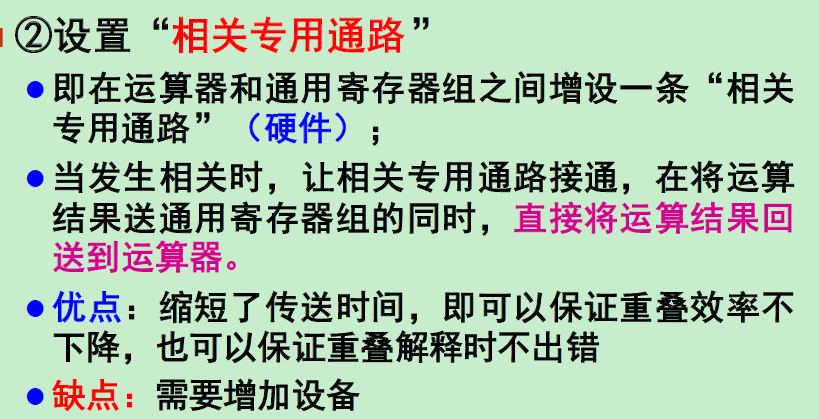


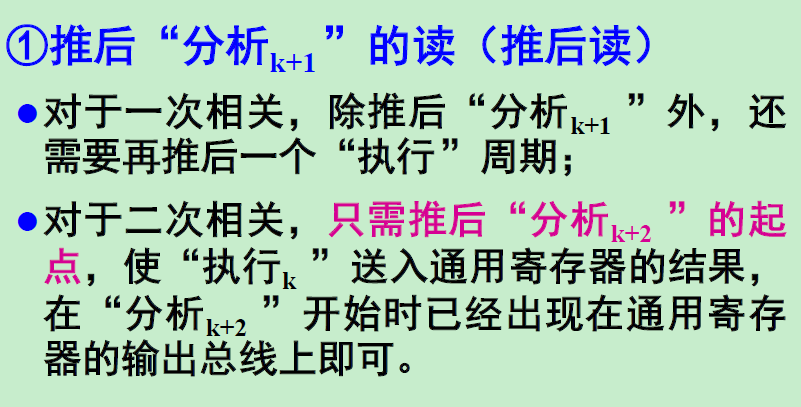
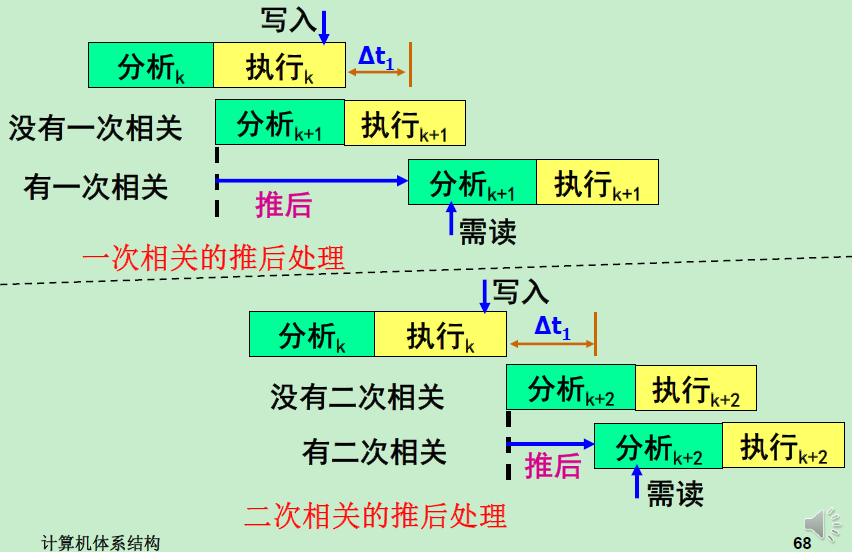
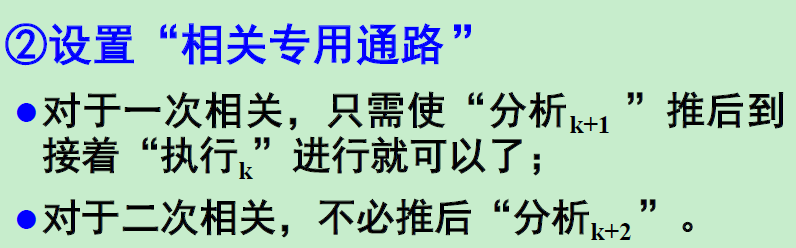


* 1. 操作数相关的处理：
     1. 解决方法**（两个方法填空）**：
        1. **推后“分析k+1”读**

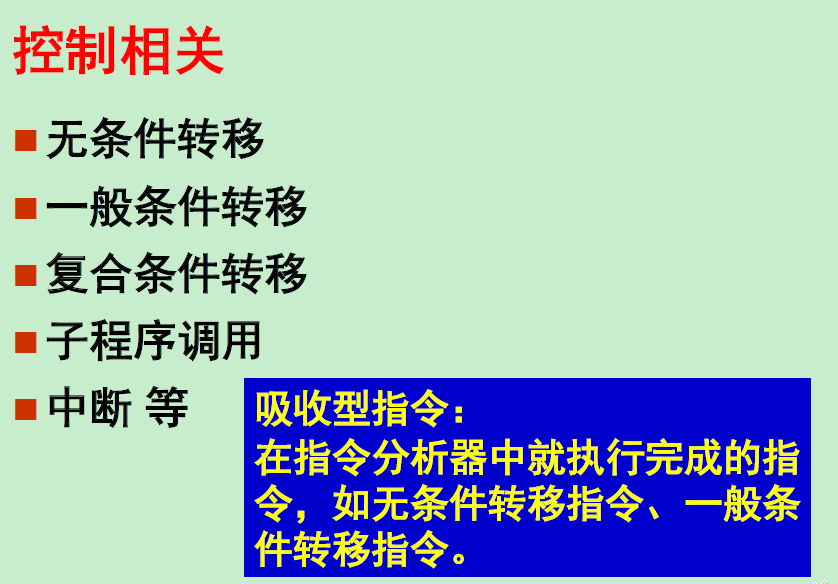


* + - 1. **设置相关专用通路**



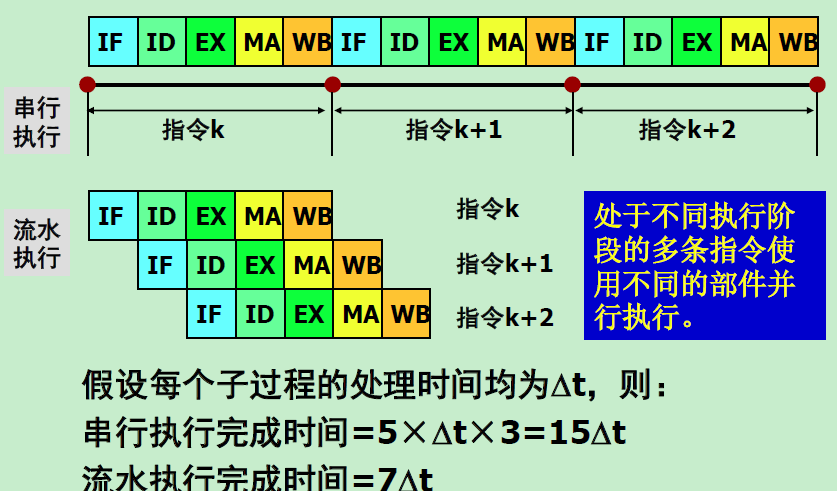
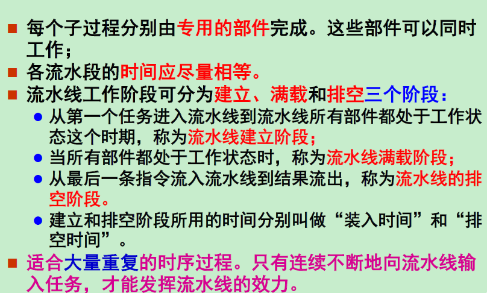
* 1. 通用寄存器组基址值/变址值相关处理
     1.  
     2. 

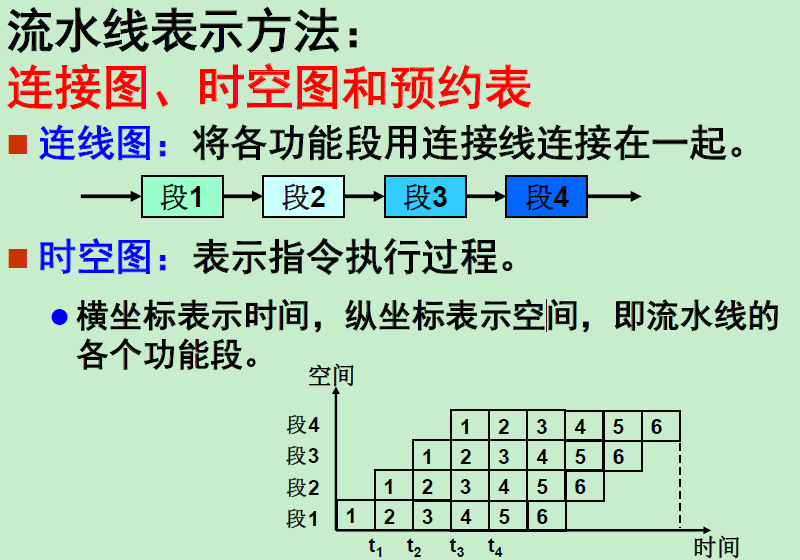
1. 控制相关



## 5.2 流水方式

### 5.2.1 流水线的基本概念

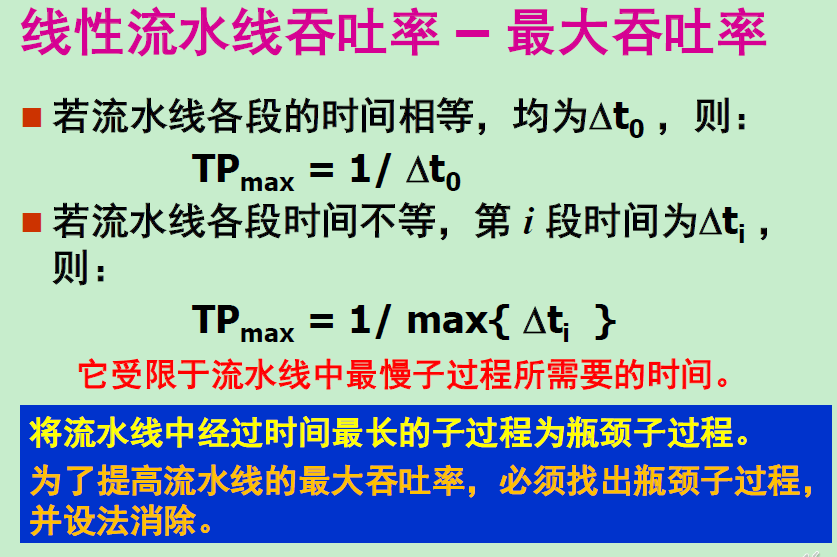
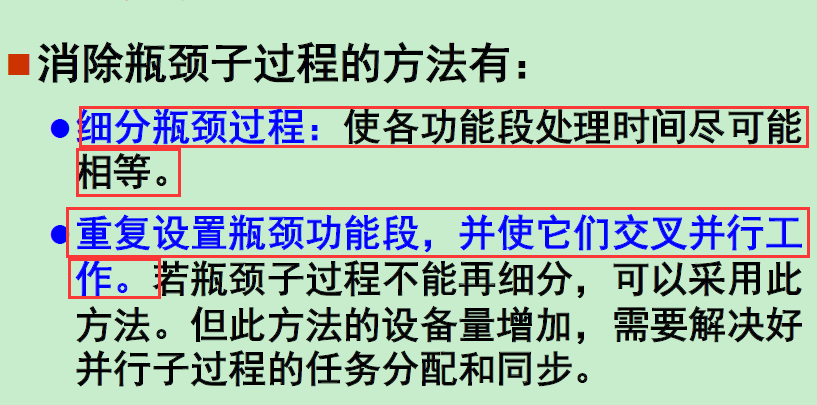
1. 流水线是重叠的引申
   1. 一次重叠是把一条指令的解释分为两个子过程，流水则是分为m个子过程。
   2. 
   3. 流水线是一条基于时间重叠的并行处理技术，不能够加快一条指令的执行，但是可以加快多条指令的执行。
      1. 由多个有联系的子过程组成。子过程称为流水线的“级”或“段“
      2. 流水线的段数称为“深度”
2. 流水线特点（四个）
   1. 只有连续提供同类任务才能充分发挥流水线的效率
   2. 在流水线的每一个流水线段中都要设置一个流水锁存器
   3. 各流水段的时间应该尽量相等，否则将引起流水线的阻塞
   4. 流水线需要有“装入时间”和“排空时间”
3. 流水线的表示方法

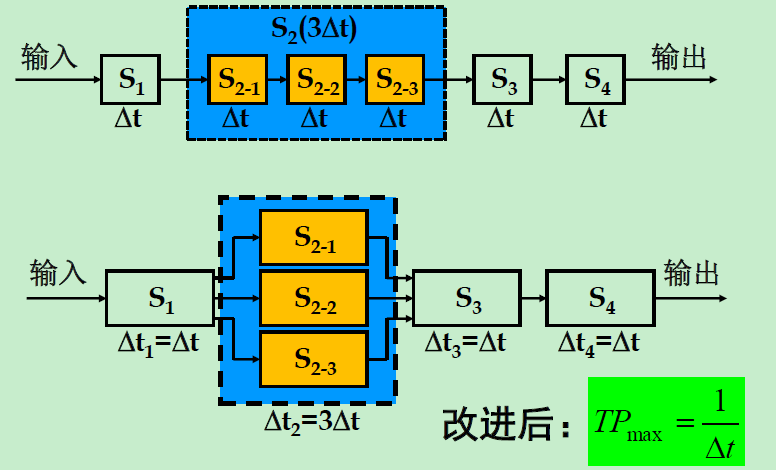
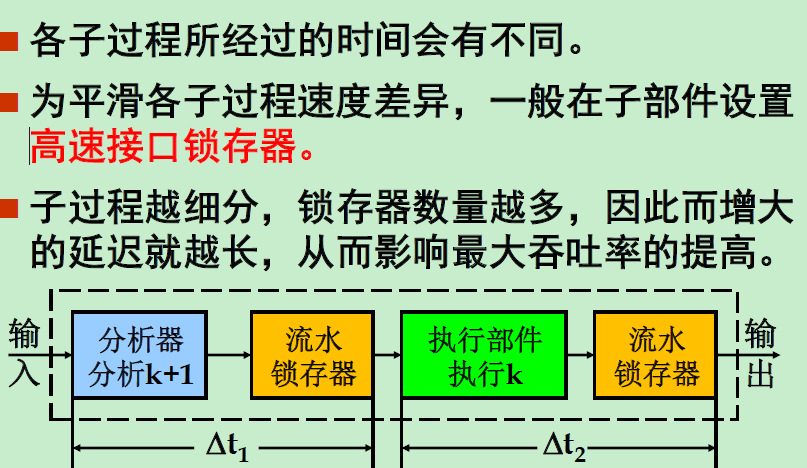


1. 流水线的分类
   1. 按照流水线的处理级别的高低（向上向下扩展的思想）来区分，分为部件级、处理机级、系统级。
      1. 部件级流水线：
         1. 构成部件的子部件之间的流水，如浮点加法流水线
      2. 处理机级流水线
         1. 构成处理机的各个功能部件之间的流水
            1. 例如先行控制器的处理机中，各功能部件之间的流水线。
            2. 例如先行控制器的处理机中各功能部件之间的流水线
      3. 系统级流水线
         1. 处理机之间的流水线，指的是构成计算机系统的多个处理机之间的流水，又称宏流水。
   2. 按照流水线具有的功能分类，可以分为单功能流水线和多功能流水线
      1. 单功能流水线
         1. 可以将但功能流水线组合起来，完成多功能流水线
      2. 多功能流水线
         1. 同一流水线的各个段之间可以有多种不同的连接方式，以实现多种不同的运算或功能
   3. 从功能部件的连接关系上对多功能流水线分类。分为静态和动态流水线。
      1. 静态：在某一时间段内，只能按一种功能的连接流水，全部流空后才能换成另一种功能。是用于处理一串相同的运算操作。
      2. 动态：各功能可根据需要连接流水。能提高利用率吞吐率，但控制复杂成本高。
   4. 按流水线各功能段之间是否有反馈回路分类。
      1. 线性流水线没有反馈回路
      2. 非线性流水线有反馈回路
   5. 按机器具有的数据表示分类。标量/向量
   6. 按照任务流出顺序与流入顺序是否相同分类。顺序/乱序

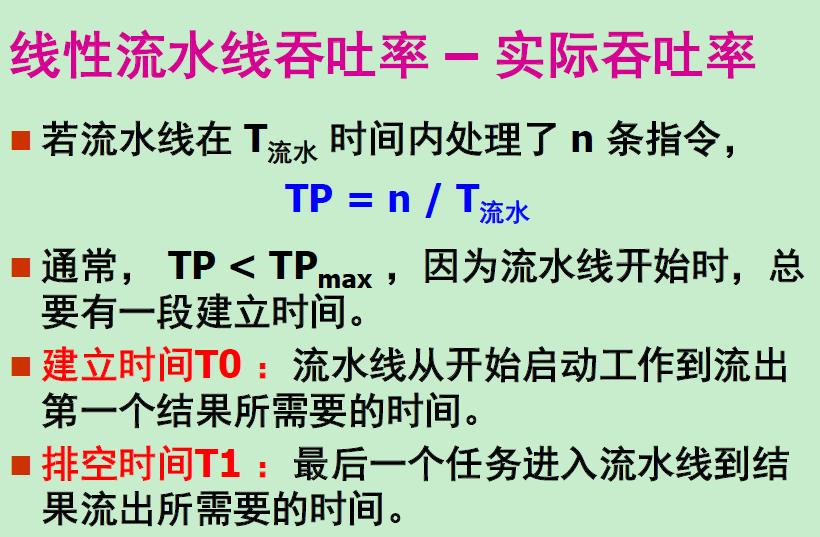
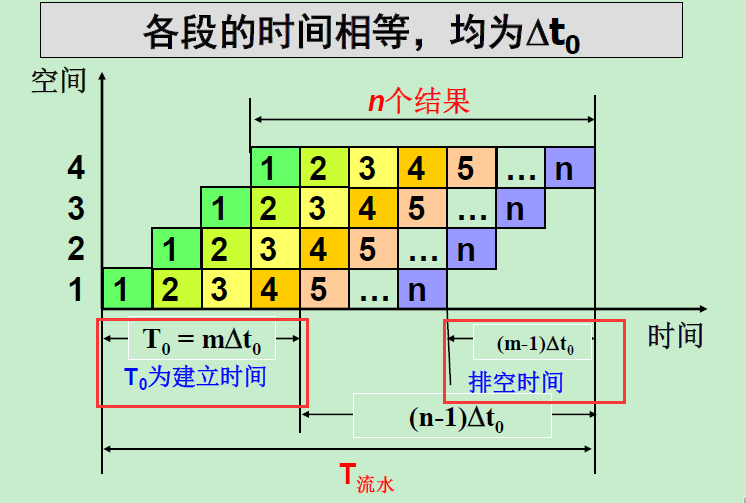
### 5.2.2 流水线处理机的主要性能

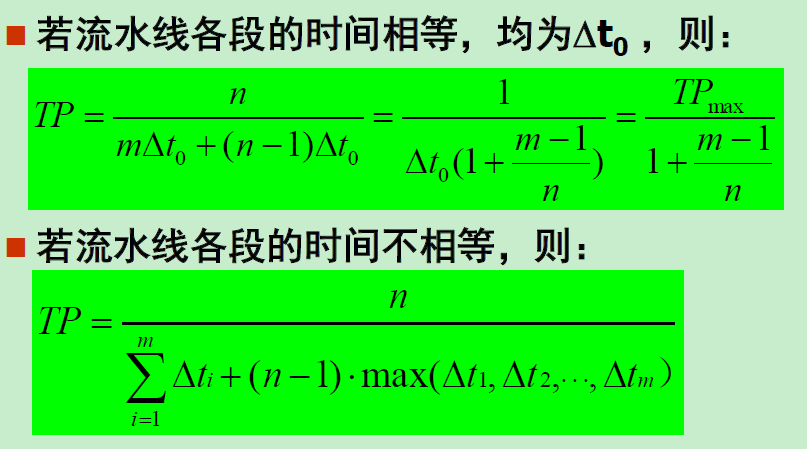
1. 吞吐率（Though Put, TP）：流水席那在单位时间内能够流出的任务数或者结果数，即完成的任务或输出的结果数量。它受限于最慢子过程经过的时间这个最慢子过程又叫做瓶颈过程。
   1. 最大吞吐率

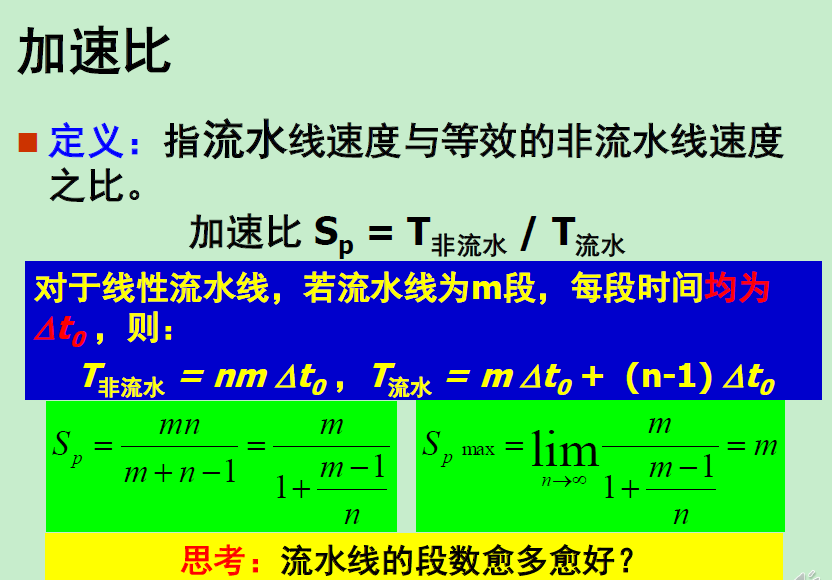
 

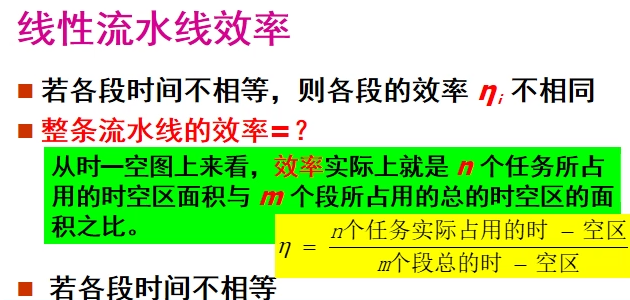
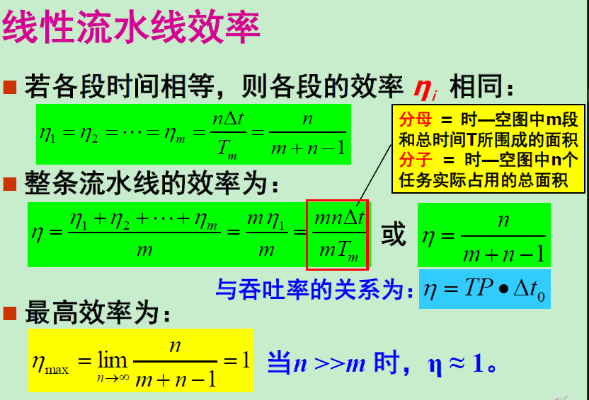
* + 1. 满负荷时可达到最大吞吐率
  1. 实际吞吐率

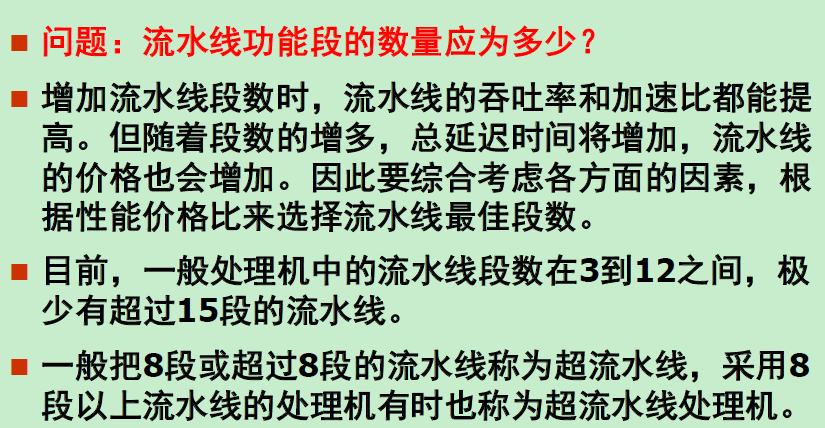
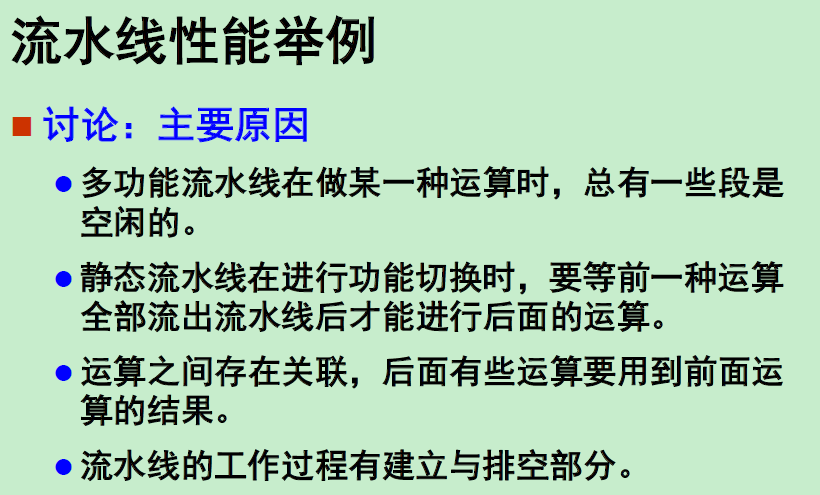
 

1. 加速比（Speed Ratio，SP）表示流水线方式相对非流水线顺序串行方式速度提高的比值。非流水线顺序串行连续完成n个任务需要n\*m\*Δt时间因此加速比为：

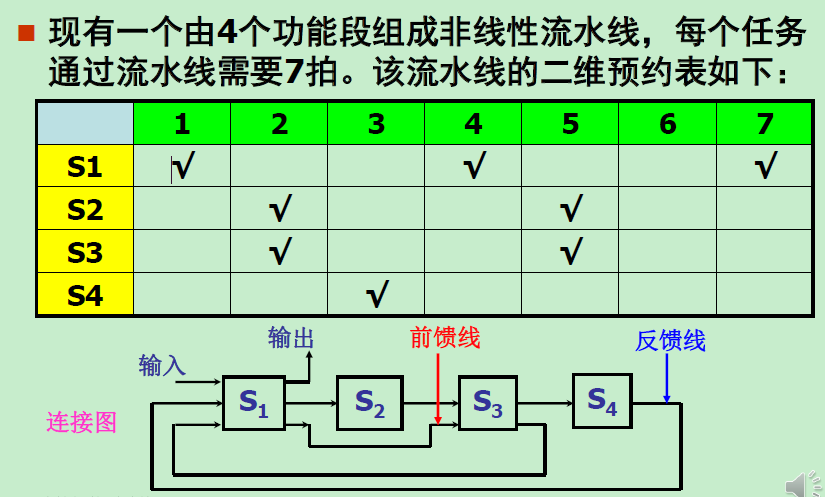
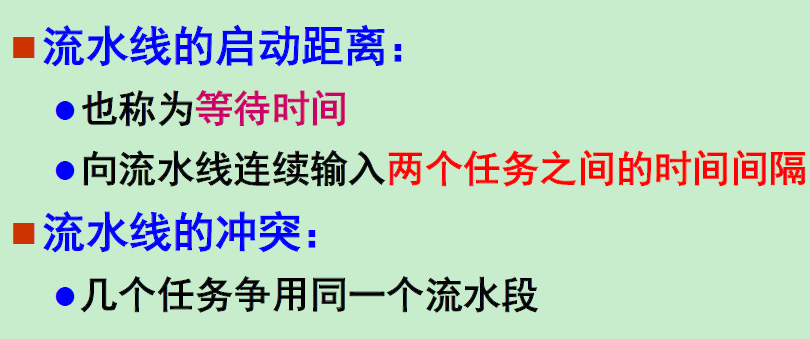
1. 效率：流水线中的设备实际使用时间占整个运行时间之比，也称为流水线的设备时间利用率。

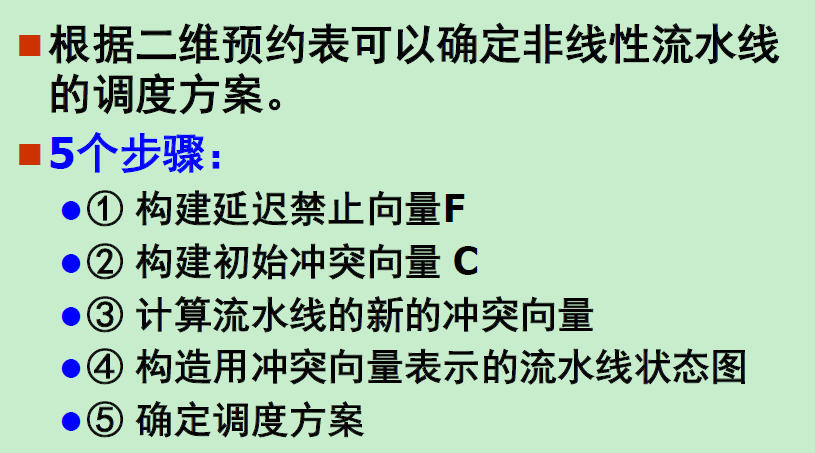
* 1. 各个段时间相等时，效率与吞吐率成正比

### 5.2.3 流水线调度（最难）

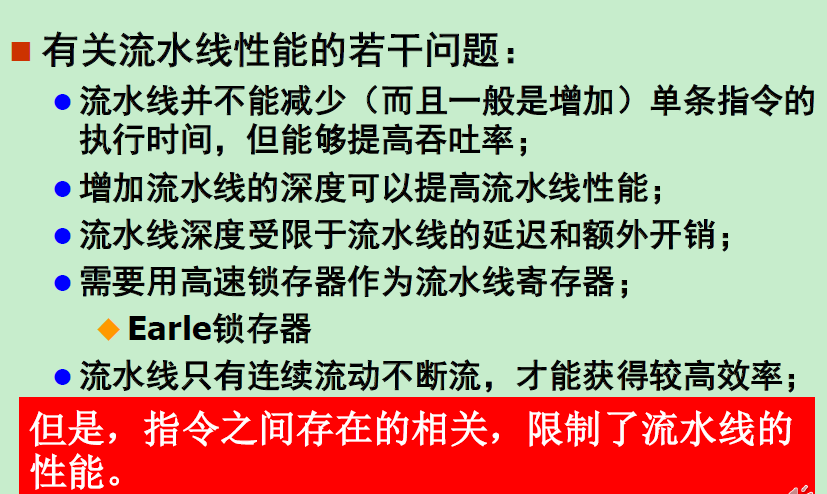
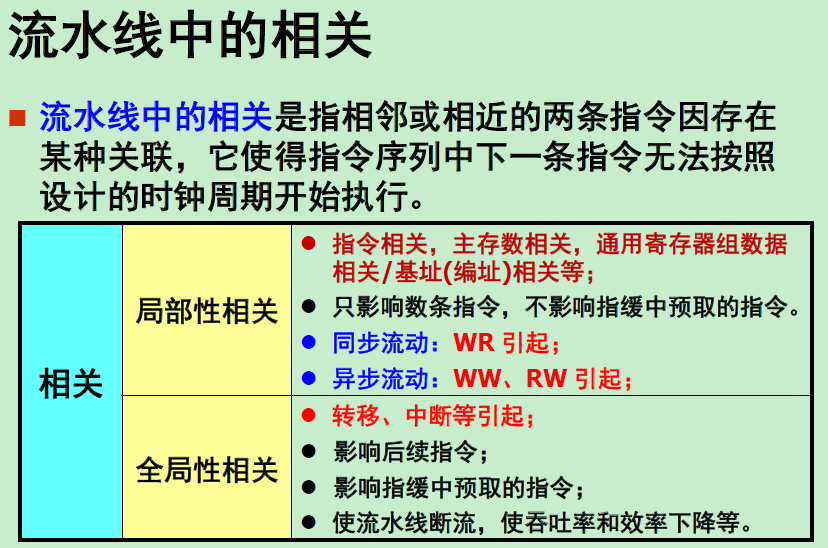
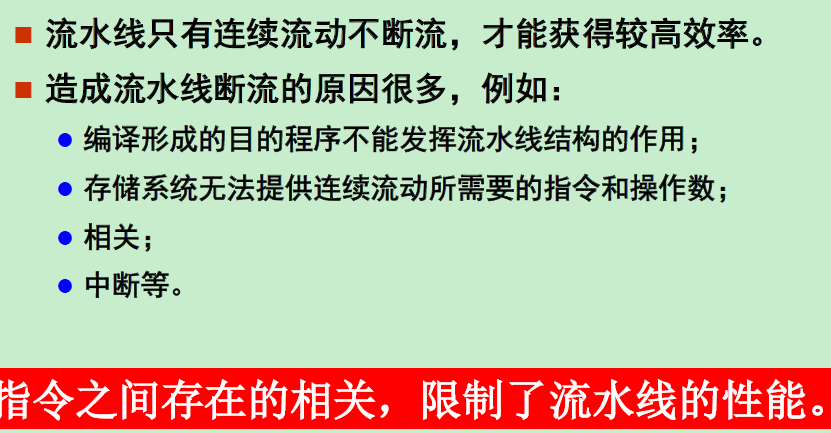
1. 由于功能部件一般只有一套，所以难免会发生资源冲突。
   1. 资源冲突：几个任务征用一个功能段
2. 线性流水线的调度：线性流水线无反馈，每个任务只通过每个段一次，不会发生冲突，因此 只需每隔 Δt （或数个 Δt ）输入一个任务即可。
3. 非线性流水线的调度：非线性流水线段间有反馈回路，一个任务在流水执行的全过程中，可能会多次通过同一段，或越过某些段。
   1. 非线性流水线的调度的任务：找出一个最小的循环周期，按照这个周期向流水线输入新任务，流水线的各个功能段都不会发生冲突，而且流水线的吞吐率和效率 最高。
   2. 在一般情况下 ，间隔 的拍数当然应该越小越好，而且往往不是一个 常数。
   3. **二维预约表法：（重点）**
      1. 假设有一个由 K 段组成的 单功能 非线性流水线，每个任务通过流水线需要 N拍。
      2. 利用类似时空图的方法，可以得到一个任务使用流水线各段的时间关系表（二维预约表）。
      3. 如果任务在第 n 拍用到第 K 段，就在相应第 n 列和第 K 行的交叉点处用 √ 表示。

 几拍右移几位

平均延迟最小的启动循环被称为**最小启动循环**。

### 5.2.4 流水机器的相关处理和控制机构

1. 局部性相关的处理
   1. 解决方法：**推后读** 和 **设置相关专用通路**
   2. **推后读** 分为同步流动和异步流动
      1. 顺序流动方式（同步流动方式）：任务流出流水线与流入流水线的顺序**一致。控制简单，但是相关后吞吐率和效率会下降。**
      2. 异步流动方式：任务流出流水线与流入流水线的顺序**不一致**
   3. **设置相关专用通路**
      1. 采用硬件的方法，可以免去相关单元的写入和读出两个子程序
      2. 缺点：硬件耗费大，控制复杂
2. 全局性相关的处理
   1. 原因： 由分支转移指令 引起的相关
      1. 影响后续指令
      2. 影响指令缓冲器中的指令
   2. 对流水线的影响是全局的，包括：
      1. 指令缓冲器中的指令或许全部要被作废
      2. 流水线断流，吞吐率和效率下降
   3. 解决方法：
      1. 为了使流水线的吞吐率和效率不至于下降太多，关键是能够争取的判断相关，即正确判断转移分支
      2. 无条件转移指令的转移分支是已知的
      3. 条件转移指令的转移分支是不确定的，需要等指令流出流水线之后才能够确定。
   4. 条件转移指令的转移分支的判断：
      1. 猜测法
      2. 加快和提前形成条件码
      3. 采取延迟转移
      4. 加快短循环程序的处理
3. 流水机器的中断处理
   1. 流水机器的中断处理主要就是如何处理好**断点现场的保存和恢复**
   2. **解决办法：**
      1. 不精确断点法
      2. 精确断点法

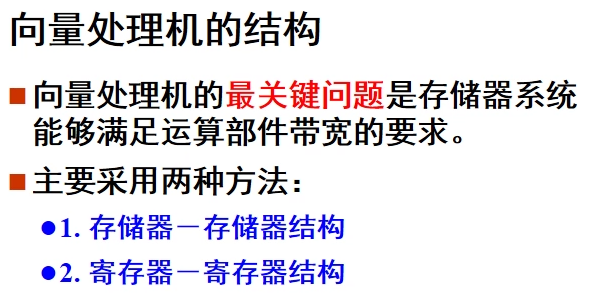
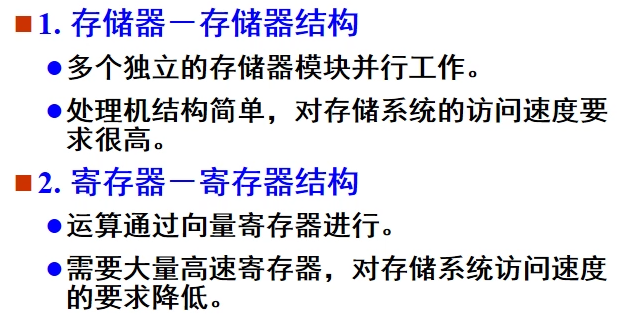
## 5.3 向量的流水处理与向量处理机

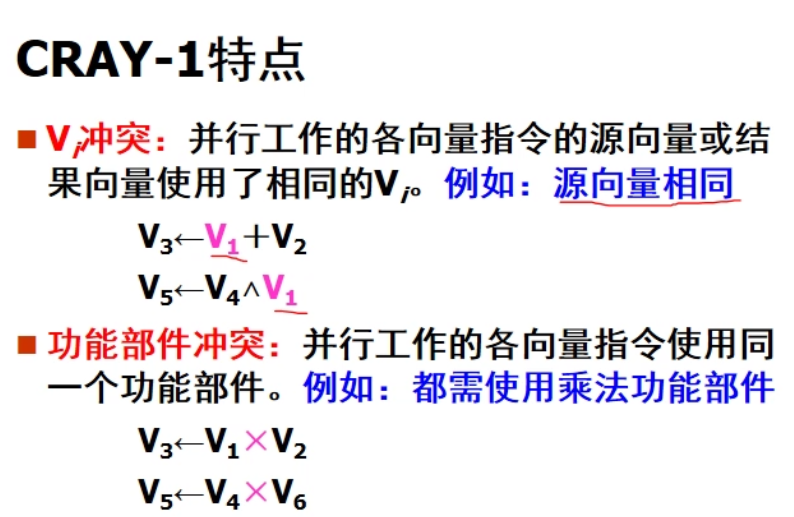
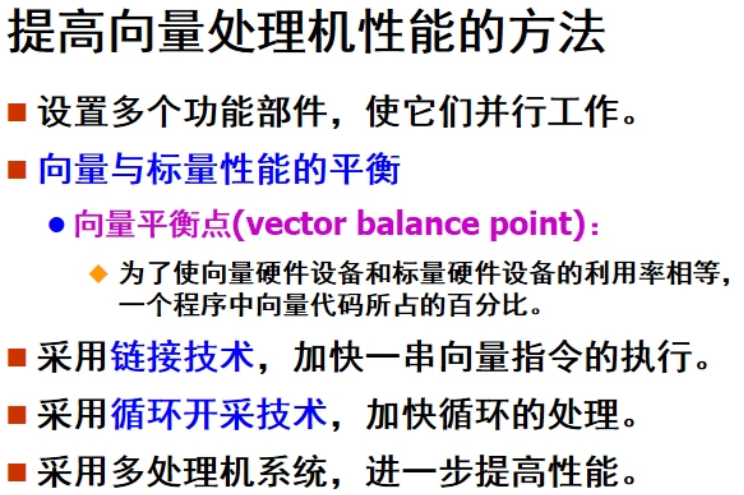
### 5.3.1向量的流水处理

1. N个互相独立的数叫做向量。由于向量中各个元素很少相关，且一般都是进行相同的运算或者处理，所以与标量运算相比，向量运算更加能够发挥出流水线的效能。
2. 向量的处理方式主要有三种：
   1. 水平（横向）处理方式
      1. 一个一个元素的完全计算
   2. 垂直（纵向）处理方式
      1. 先完成两个向量的完整计算
   3. 分组（纵横）处理方式
      1. 以n为一组，当向量长度大于向量寄存器长度n时，需要分组处理
      2. N = nk+r
      3. 组内采用纵向，组间采用横向

### 5.3.2 向量流水处理机

1. 向量处理机的指令系统
   1. 标量型
   2. 向量型
2. 向量流水处理机的结构

## 5.4 指令级高度并行的超级处理机



指令级的并行性（ILP）：指令序列中的并行性  
思想：可同时流出多个指令/操作

表示：ILP(M,N) = m \* n

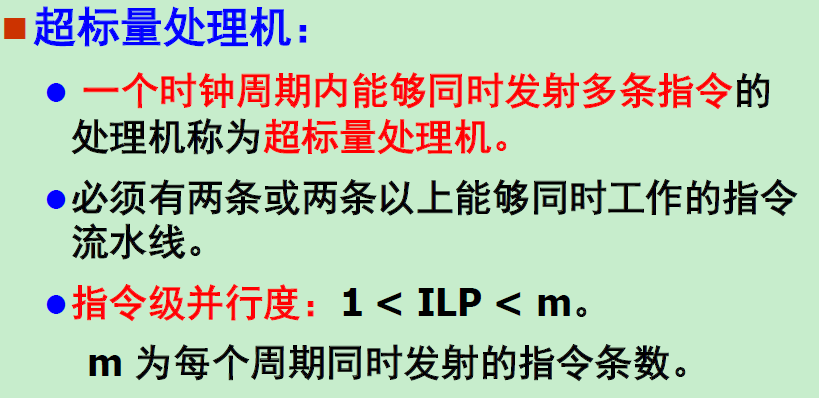
其中 m：每个时钟启动的次数

n: 每次启动的指令/操作的个数

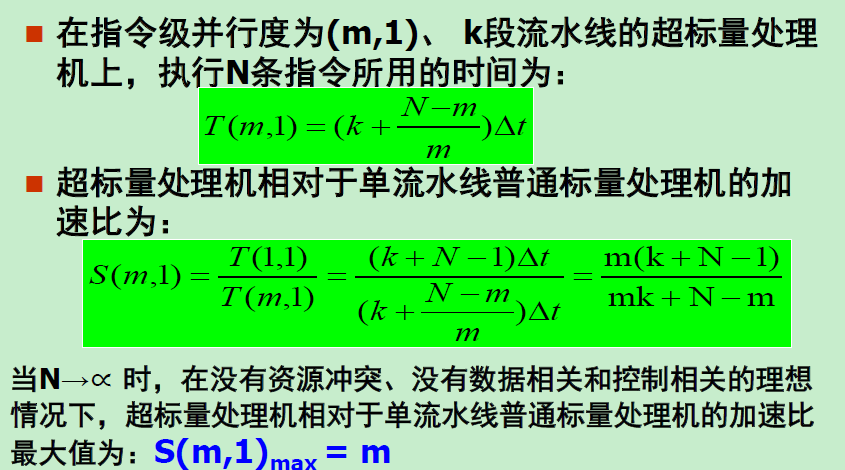
特征:IPL \* CPI = 1

### 5.4.1 超标量处理机

1. 本质就是在不同的流水线中执行不相关指令的能力。
   1. 常规的标量流水线单处理机是在每个Δt时间内解释完成一条指令。称这种流水机的度为1.
   2. 超标量处理机采用m条指令流水线，每个Δt时间内解释完成m条指令。称这种流水机的度为m.（m大于一）**超标量处理机的指令级并行度大于1**
2. 单发射和多发射
   1. 单发射
      1. 每个周期只取一条指令、只译码一条指令，只执行一条指令，只写回一个运算结果。
      2. 每个时钟周期平均执行一条指令， ILP的期望值为 1
   2. 多发射
      1. 每个周期同时取多条指令、同时译码多条指令，同时执行多条指令，同时写回多个运算结果。
      2. 每个时钟周期平均执行多条指令， ILP的期望值大于 1
3. 超标量处理机



1. **超标量处理机性能（重点 要会算）**

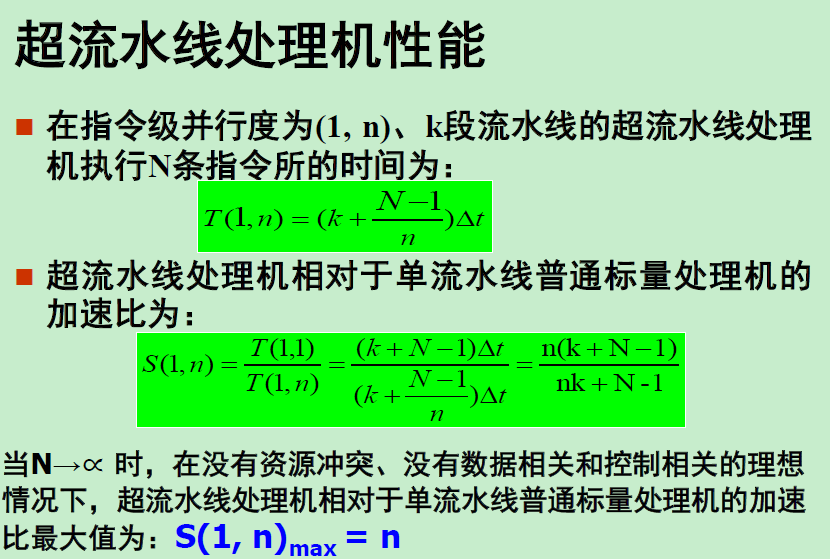
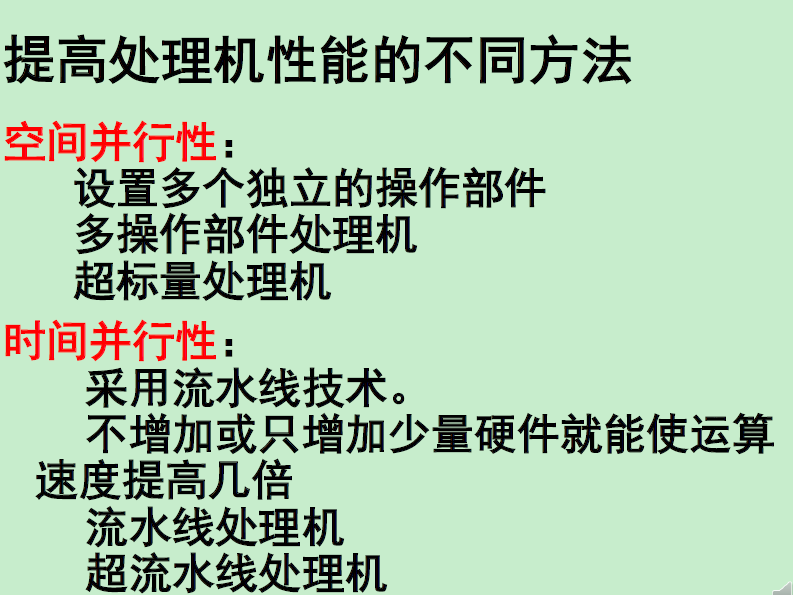


### 5.2.4 超流水线处理机

1. 一个周期内能够 分时发射 多条指令的处理机称为超流水线处理机 。

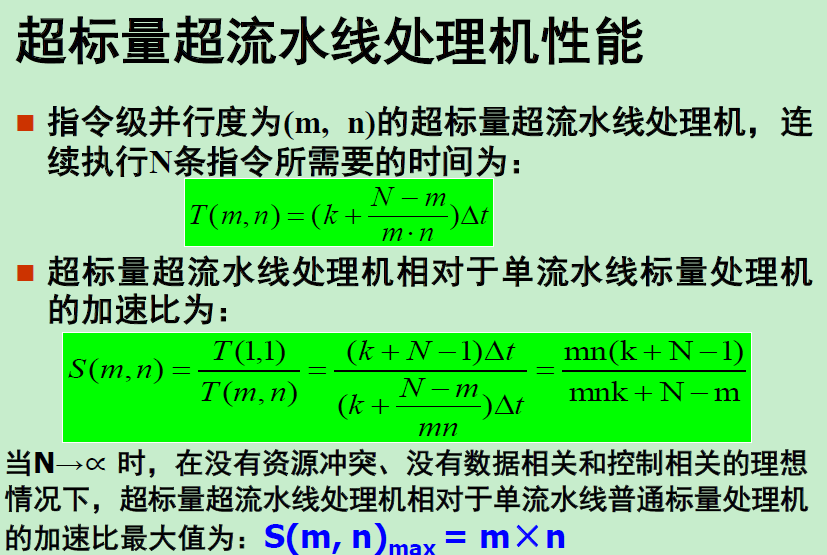
指令流水线有 8 个或更多功能段的流水线处理机称为超流水线处理机 。

1. 指令执行时序：每隔 1 /n 个时钟周期发射一条指令 流水线周期为 1 /n 个时钟周期
2. 没有**高速时钟机制** 超流水线处理机是无法实现的

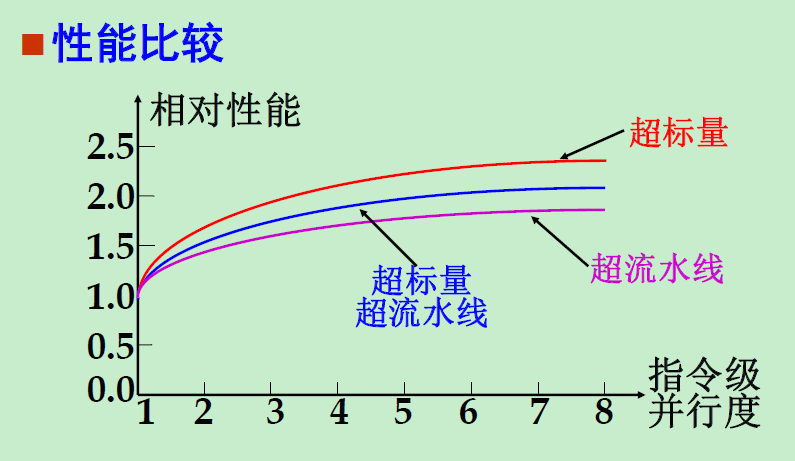
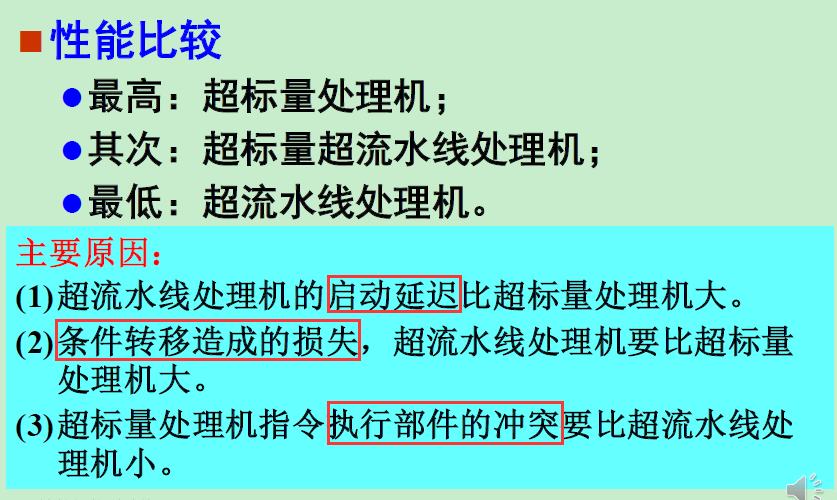


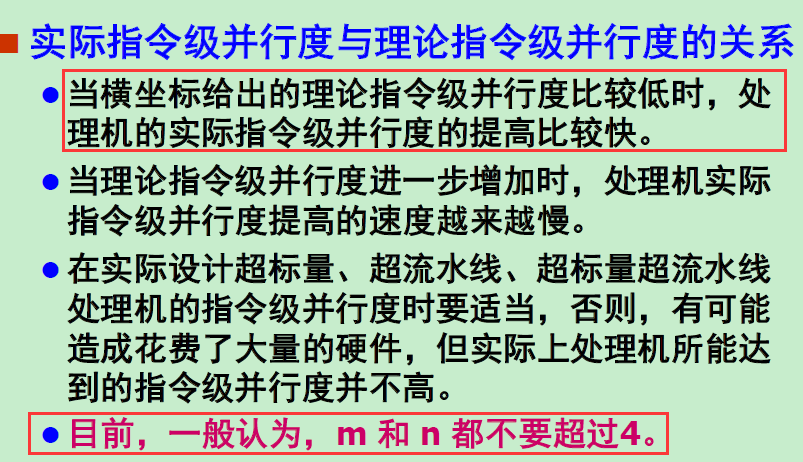
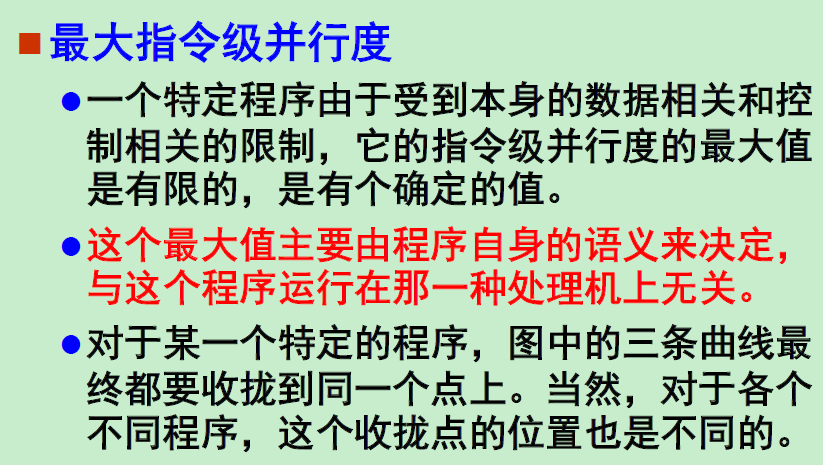
### 5.4.3 超标量流水处理机

1. 超标量超流水线处理机在一个时钟周期内分时发射指令 n 次 每次同时发射指令 m 条每个时钟周期总共发射指令 m \* n 条
2. 性能

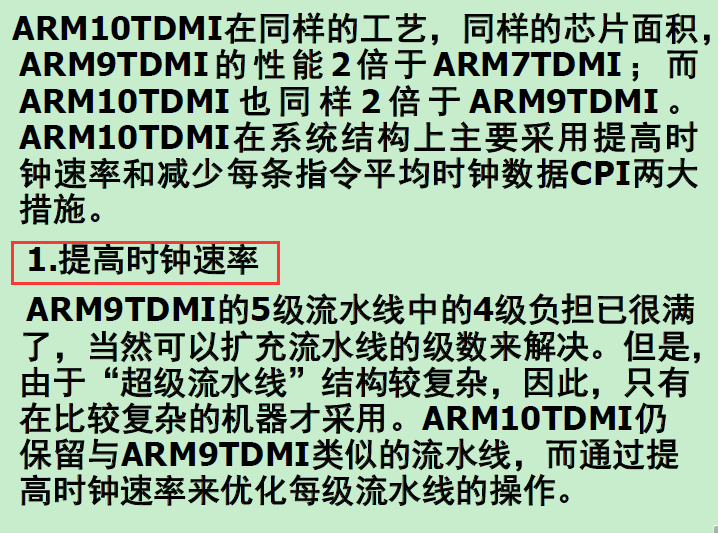


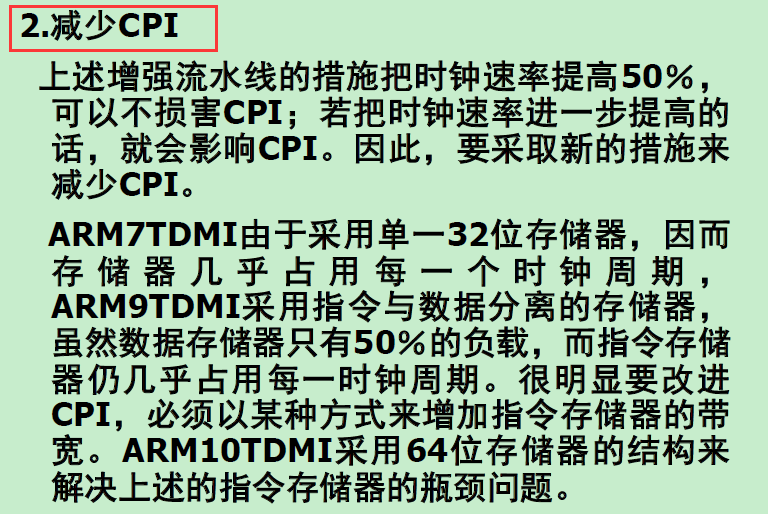
### 5.4.4 三种指令级并行处理机性能比较

## 5.5 ARM（个人认为了解即可）



[[1]](#footnote-1)

参考资料

1. 计算机体系结构简明教程.蒋本珊 马忠梅 郑宏 [↑](#footnote-ref-1)