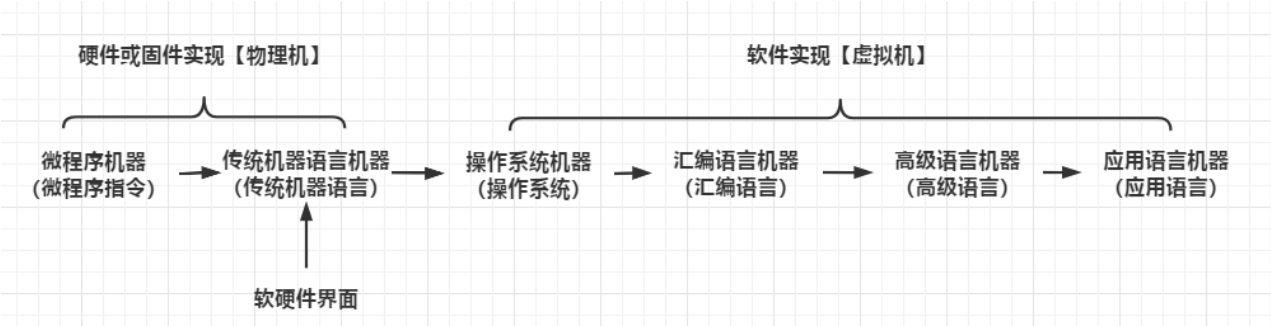
# 第一章 计算机系统结构的基础知识

- 计算机系统的层次结构：




- **翻译 和 解释**
  - 翻译：转换程序把高一级机器上的**程序**转换为低一级机器上**等效的程序** （**速度快，占用存储空间大**）
  - 解释：把高一级机器上程序的**每一条语句**转换为低一级机器上的一段**等效程序**并执行（速度慢，占用存储空间小）
- 计算机组成 和 实现
  - 计算机体系结构：【例：机器指令集确定】
  - 计算机组成：计算机系统结构的逻辑实现 【例：指令的实现方式，如取指令等具体操作】
  - 计算机实现：计算机系统结构的物理实现 【例：指令集中多有指令功能的具体电路等实现】

  **具有相同系统结构的计算机可以采用不同的计算机组成；同一种计算机组成又可以采用多种不同的计算机实现**

## 系统结构分类

- 计算机系统结构的分类
  - **Flynn分类法【依据指令流和数据流的多倍性】：SISD，SIMD；MISD；MIMD**
  - Handler分类法【依据计算机并行度和流水线】：三个层次（程序控制器PCU；算数逻辑部件ALU或运算部件PE；运算部件所包含的位级电路BLC的套数）
  - 冯氏分类法【依据最大并行度】：字串位串；字串位并；字并位串；字并位并

## ☆ Amdahl 和 CPU性能计算

- 计算机系统设计的定量管理：大概率事件优先原理；程序局部性原理；Amdahl定律；CPU性能公式
  - Amdahl定律：加快某部件执行速度所获得的系统性能(加速比)，与该部件在系统中**总执行时间的比例**有关

    $$S_n = \frac{T_0}{T_n} = \frac{1}{(1-\sum_i F_i) + \sum_i (\frac{F_i}{S_i})}$$ $S_e$：部件加速比 $F_e$：可改进比例
  - CPU性能公式

    $$CPI = \frac{执行程序所需的时钟周期数}{IC(执行程序的指令条数)}$$

    $$CPU_{时间} = IC \times CPI \times t = IC \times CPI / f$$
- 计算机系统设计的主要方法

  由上往下【专用机设计】，由下往上【软件处于被动，会造成软硬件脱节】，从中间开始【**传统机器语言机器级与操作系统机器级之间**】

- 计算机系统的性能测评：执行时间 和 速度（吞吐率, (MIPS，MFLOPS)）

  性能比较：平均执行时间，加权执行时间，调和平均值法，几何平均值法

  调和平均值：$H_m = \frac{n}{\sum \frac{1}{R_i}} = \frac{n}{\sum T_i}$ 几何平均值：$G_m = \sqrt[n]{(\prod R_i)} = \sqrt[n]{(\prod \frac{1}{T_i})}$

- 冯诺依曼结构：**存储程序**原理及**指令驱动**；**运算器为中心**；指令和数据同等对待；存储器顺序线性编址；顺序执行，程序分支由转移指令实现；

- 实现**可移植**的常用方法：**采用统一的高级语言；系列机【兼容机】；仿真和模拟**

  兼容：**向上兼容**，向下兼容，向前兼容，**向后兼容【系列机的根本特征】**

  模拟：软件方法（虚拟机），**机器语言解释**

  仿真：**较多硬件**，**微程序**直接**解释**另一台机器的指令系统（难度大）

- 并行性
  - 概念：计算机在**同一时刻或同一时间间隔**内进行多种运算或操作，称为并行性【同时性 or 并发性】
  - 处理数据角度：字串位串 `->` 字串位并 `->` 字并位串 `->` 全并行
  - 执行程序角度：指令内部并行 `->` 指令级并行 `->` 线程级并行 `->` 任务级或过程级并行 `->` 作业或程序级并行
  - 提高并行性的技术途径：**时间重叠；资源重复；资源共享（如分时系统）**
  - **多机系统**遵循时间重叠、资源重复、资源共享原理，发展为**3种不同的多处理机**：同构型多处理机、异构型多处理机、分布式系统
  - 耦合度：紧密耦合系统【直接耦合】；松散耦合系统【间接耦合】

软件是促使计算机系统结构发展最重要的因素，应用是促使计算机系统结构发展最根本的动力，而器件是促使计算机系统结构发展最活跃的因素。

# 第三章 流水线技术

- 流水线的表示：**连接图 和 时空图**
- 特点
  - 装入时间和排空时间时，流水线不满载
  - 流水线每一个段后面通常都有一个缓冲寄存器,称为**流水寄存器**。在相邻两段间传送数据，有**缓冲、隔离、同步**作用
  - 各流水段时间尽可能相等，否则将引起流水线堵塞、断流，时间最长的段为**流水线瓶颈**
  - 流水线**性能**与流水线输入端输入速度有关

## 流水线分类

- 流水线分类
  - 按流水线等级：部件级流水线；处理机级流水线；系统级流水线【宏流水线】
  - 按流水线完成的功能：**单功能流水线；多功能流水线【分为静态流水线 和 动态流水线** !! 计算题不一样】
  - 按流水线中是否有反馈回路：线性流水线；非线性流水线
  - 按任务流入流出顺序是否相同：顺序流水线；乱序流水线【无序/错序/异步】

## ☆ 流水线性能计算公式

- 性能指标

- 吞吐率：$P=\frac{n}{T_k}$ 注意点：分母单位是时间，不是时钟周期数！要乘以$\triangle t$
  - 总时间的计算
    - 如果各段时间均相等：$T_k=(k+n-1)\triangle t$
    - 各段时间不相等：$T_k=\sum_{i=1}^{k}\triangle t_i+(n-1)\max(\triangle t_1,\dots\triangle t_k)$
  - 解决流水线瓶颈的方法：**重复设置瓶颈段；瓶颈段细分【超流水线】**
- 加速比：$S=\frac{T_s}{T_k}$ $T_s$：顺序执行，$T_k$：用K段流水线的时间
- 流水线效率【流水线设备利用率】：$E=\frac{nk\triangle t}{kT_k}=\frac{n\triangle t}{T_k}=P\triangle t=\frac{S}{k}$

- 流水线的最佳段数：PCR定义为单位价格的最大吞吐率

## 5段流水线

- 经典5段流水线
  - 取指令【IF】
  - 指令译码【ID】：访问**通用寄存器**，读出所需操作数
  - 执行 / 有效地址计算【EX】
    - ALU：执行计算
    - load / store ： 形成访存有效地址
    - 分支指令：形成转移目标地址
  - **存储器访问** / 分支完成 【MEM】 【与IF段访存可能出现冲突，解决方法：采用哈佛结构，采用分离的指令存储器和数据存储器 或 指令Cache和数据Cache】
    - store：把指定的数据写入存储器单元
    - load：从存储器中读取相应的数据
    - 分支：若分支成功，将转移目标地址送入PC

      ALU在此为空操作
  - 写回【WB】：把结果数据写入**通用寄存器组**
    - ALU：结果数据来自ALU 【与ID段读寄存器可能会造成冲突，解决方法：WB写操作在前半周期，ID读操作在后半周期】
    - load：结果数据来自存储器
    - 分支 / store：空操作

分支指令和ALU指令、store指令需要**4个周期**；**load指令需要5个周期。**

## 相关与冲突

- 3种**相关类型**
  - **【真】数据相关** 【RAW / WR】
  - **名相关**：**反相关**【WAR / RW】 **输出相关**【WAW / WW】 `->` **寄存器换名**技术
  - **控制相关**【全局相关】：分支指令引起（无条件转移 / 条件转移 / 中断等）
- 3种**冲突类型**
  - **结构冲突**：需**重复设置资源** 或者 插入暂停周期

    - **数据冲突**：WAR，RAW，WAW，解决方案：定向技术（旁路/专用通路技术）；停顿（互锁机制）；依靠编译器（指令/流水线调度）
    - **控制冲突**：通过软件（编译器）来减少分支延迟：
        - **预测分支成功**：在5段流水线中，除非已知分支目标地址，否则没有任何好处【因为分支判断与地址计算在同一个周期】
        - **预测分支失败**： 若预测正确，无延迟；若预测错误，延迟1个周期（设分支指令在ID段末尾完成）
        - **延迟分支**：**延迟槽**（不管分支是否成功，都顺序执行延迟槽中的指令）

          调度策略：**从前调度【被调度指令必须与分支无关，适用于任何情况】，从目标处调度【分支成功时起作用】；从失败处调度【分支失败时起作用】**

          **分支取消机制**（若预测失败，就将分支延迟槽中的指令转化为空操作）
- MIPS流水线的实现
    - 所有**数据冲突**均在**ID段检测**到（若有冲突，就插入停顿周期）；流水线的**互锁机制**（硬件，比较寄存器地址是否相等）
    - 改进后的流水线对**分支指令**的处理提前到**ID段**进行

# 第四章 向量处理机

## 向量处理机结构及方法

- 参加运算的每个向量都需指明其基地址、位移量和向量长度。
- 向量处理的方法
    - 横向处理：不适合向量流水处理
    - **纵向处理**：适合 **存储器-存储器型** 的向量处理机
    - **纵横处理【分组处理】**：组内纵向，组间横向，适合 **寄存器-寄存器型** 的向量处理机
- 结构
    - 存储器-存储器结构
        - 流水线运算部件的输入和输出端都直接（或经**缓冲器**）与**存储器相连**
        - 源向量，目的向量和**中间结果都需要送回存储器**
        - 普通存储器的**3倍带宽**：一个时钟周期内读出两个操作数并写回一个结果。
    - 寄存器-寄存器型结构
        - 中间有存储器，但访问速度更快；且提供高带宽及多种寻址方式，支持流水线链接技术
    - Cray-1 的特点
        - 12条**单功能流水线**
        - 主存储器与V寄存器之间数据传送以**成组传送方式**进行
        - 只要不出现**向量Vi和功能部件冲突**，各Vi之间和各功能部件之间都能并行工作

## 提高向量流水处理机性能的方法

- 提高向量流水处理机性能的方法
    - 设置多个功能部件，使它们并行工作
    - 采用**链接技术**，加快一串向量指令的执行font>

    - 采用循环开采技术，加快循环的处理
    - 采用多处理机系统，进一步提高性能
- **链接技术**利用向量指令间存在**先写后读**的数据相关性，加快向量指令序列执行速度，是流水线定向技术的发展

  条件：
    - 空间
        - **无向量寄存器和功能部件使用冲突**
    - 时间
        - 只有前一条指令的第1个结果分量送入结果向量寄存器的那一个时钟周期方可链接
        - 向量指令的两个源操作数分别是两条先行指令的结果寄存器时，先行两条指令**产生运算结果的时间必须相等。**同时，两条向量指令的**向量长度也须相等。**
- 分段开采技术【向量循环】
    - 向量分段由系统**硬件和软件控制**完成，**对程序员透明**

## ☆ 向量处理机性能计算公式

- 向量处理机的性能评价
    - 一条向量指令的处理时间$T_{vp}=(s+e+n-1)T_c=(T_{start}+n)T_c$

      【s: 建立流水线所需时钟周期数；e: 流水功能部件级数；n: 向量长度；$T_{start}=s+e-1$，启动时间，**还差一个时钟周期**就产生第一个结果所需的时钟周期数】
    - 每秒百万次浮点运算**MFLOPS**

      **基于操作**而非指令，用来比较两种不同机器；衡量机器的**浮点操作性能**，不体现整体性能

      程序在不同机器上执行的**指令数目可能不同**，但执行的**浮点运算总次数相同**
    - 一组向量指令的处理时间
        - 能在一个同一时间或时间段**并行或链接**执行的一条（或多条）向量指令，称为一个编队
            - 并行编队：多条指令的执行时间取该编队中各指令的执行时间的**最大值**
            - 链接编队：多条指令的执行时间取该编队中各指令的执行时间**和**
        - 不分段开采时，总时间：$T_{all}=(\sum_{i=1}^{m}T_{start}^{(i)}+mn)T_c$
        - 分段开采时，总时间：$T_n=(\lceil\frac{n}{MVL}\rceil\times(T_{start}+T_{loop})+mn)T_c$ 【m：编队数，n：向量长度】
    - 最大性能$R_\infty=\lim\frac{向量指令序列中浮点运算次数\times时钟频率}{向量指令序执行所需的时钟周期数}$ 【峰值性能】
    - 半性能向量长度$n_{\frac{1}{2}}$

      反映为建立流水线而导致的性能损失；值**越小**，性能越好；**与具体的向量指令有关**
    - 4.向量长度临界值$n_v$

      是指对于某一计算任务而言，向量方式的处理速度优于**标量串行**方式处理速度时所需的最小向量长度。

# 第五章 指令级并行及其开发——硬件方法

## ILP相关概念

- 开发**指令级并行性【ILP】**；能降低指令执行的平均周期数CPI。属**细粒度并行性。** 细粒度并行性：处理机指令一级或操作一级并行处理。 粗粒度并行性：进程、任务、程序一级的并行处理

  途径：**资源重复【空间并行】 和 流水线技术【时间并行】**

  方法：硬件（动态） 和 软件（静态）
- 基本程序块：一串连续的代码，除入口和出口外，没有其他的分支指令和转入点

  循环级并行：使一个循环中的**不同循环体**并行执行
- 三种相关类型：**数据相关；名相关；控制相关** （相关是否会导致实际冲突的发生以及该冲突会带来多长的停顿，是流水线的**固有属性**）
- 正确执行程序，必须**保持**最关键的两个属性：**数据流 和 异常行为**

## Tomasulo和记分牌

- 调度方法：
  - 静态调度：依靠编译器；编辑期间执行；与具体的流水线结构相关；顺序执行
  - 动态调度：依靠硬件和算法；程序执行期间进行；乱序执行
    - 记分牌算法
      - 执行段采用**多功能部件**，允许多条指令在执行段中**并行操作**；尽早地执行没有结构冲突和数据冲突的指令【并没有去解决冲突，而是停顿等待冲突消失】
      - **WW冲突** 和 **资源冲突**会导致记分牌在**流出阶段**停顿。 **RW冲突**会导致记分牌在**写结果阶段**停顿 **WR冲突**会导致记分牌在**读操作数阶段**停顿
    - Tomasulo算法【公共数据总线法】
      - 通过**寄存器换名**来消除RW冲突和WW冲突，**分散控制**处理数据相关和乱序执行
      - **指令顺序流出**
      - 如该数据已经就绪，其操作数寄存器号换成**数据本身或保留站标识**，不再与寄存器有关系

        如操作数没有计算出来，则将该指令中相应的寄存器号换名为**将产生操作数的保留站标识**
      - **冲突检测和指令执行控制是分布的**
      - **计算结果通过CDB**直接从产生它的保留站传送到所有需要它的功能部件，**不用经过寄存器**
      - **消除 WAW冲突 和 WAR冲突 和 结构冲突：指令流出【只要保留站空闲就能流出】**

## 动态分支预测技术

- **动态分支预测技术**：从转移指令近期转移**是否成功**的记录，动态地预测下一次转移的方向
  - 分支历史表【BHT】（常用2位计数器）
    - 分支指令到达**（ID）段**，从BHT读出信息进行分支预测，预测正确时，**有一个周期的延迟**；如果将预测提前到取指阶段，则预测正确的情况下没有延迟；预测不正确，则需要清除指令预取缓冲器，会产生多个周期的延迟
    - 格式：[转移指令 ，历史表信息，转移目标指令]
  - 分支目标缓冲器【BTB】
    - 将**分支成功**的分支**指令的地址**和它的**分支目标地址**都放到一个缓冲区中保存

- 格式：[成功转移的分支指令地址 ， 转移目标指令地址]
- 延迟分析：

| 指令在BTB中？ | 预测 | 实际情况 | 延迟周期 |
| --- | --- | --- | --- |
| 是 | 成功 | 成功 | 0 |
| 是 | 成功 | 不成功 | 2 |
| 不是 | | 成功 | 2 |
| 不是 | | 不成功 | 0 |

  - 分支目标指令缓冲区【BTIB】：得到**转移目标指令处**的**多条指令地址**

- 基于硬件的前瞻执行

  对分支指令结果进行猜测(假设猜测总是对的),并**按猜测结果继续取指、流出和执行后续的指令**；

  执行结果写回**再定序缓冲区** ROB，等到确认后再写回寄存器或存储器

  将写结果分成：**写结果 和 指令确认**

  **乱序执行，顺序确认**；实现了**精确异常**

## ☆ 多流出技术时空图

- 多指令流出技术
  - 超标量流水线：空间并行；**同时发射**；指令**按序流出**，在流出时进行**冲突检测**；需要增加浮点寄存器的读/写端口 和 定向路径
  - 超长指令字：指令调度是由**编译器**静态完成的；**单一的控制流或控制器**: **每个时钟周期启动一条超长指令。**程序代码增加，采用**锁步机制**（任何一个操作部件停顿，整个处理机就停顿），机器代码不兼容
  - 超流水线处理机：在一个时钟周期内能够**分时流出**多条指令

比较图：

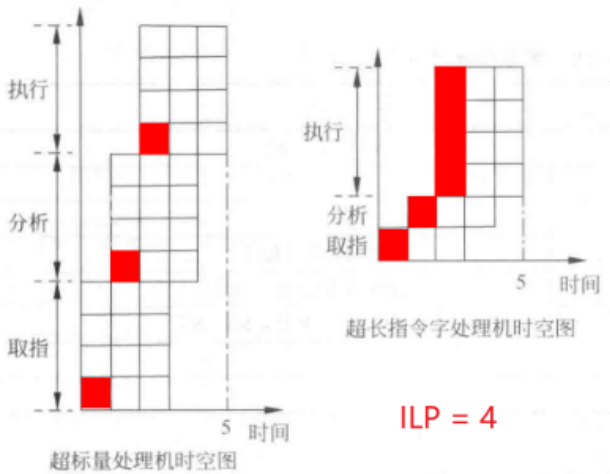



图 5.3　超标量流水处理机与超长指令字处理机的时空图

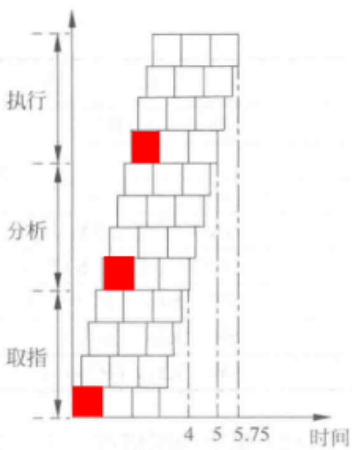



图 5.4　超流水处理机的时空图

# 第九章 互连网络

- 互连网络的三要素：**开关元件，互联结构，控制方式**；

## ☆ 互连函数一览表

- 互连函数一览表:

  **交换函数 (特例：立方体互连函数)【构造(超)立方体网络】$E$**

  $E_k(x_{n-1}x_{n-2}\dots x_{k+1}x_kx_{k-1}\dots x_1x_0) = x_{n-1}x_{n-2}\dots x_{k+1}\bar{x}_kx_{k-1}\dots x_1x_0$ 同$Cube_k$

  **均匀洗牌函数 (混洗函数)【构造Omega网络】 循环左移 $\sigma$**

  $\sigma(x_{n-1}x_{n-2}\dots x_1x_0) = x_{n-2}\dots x_1x_0x_{n-1}$

  逆均匀洗牌函数：循环右移

  **蝶式函数 交换最高&最低位【构造立方体多级网络】$\beta$**

  $\beta(x_{n-1}x_{n-2}\dots x_1x_0) = x_0x_{n-2}\dots x_1x_{n-1}$

  **反位序函数 对称变换一下 $\rho$**

  $\rho(x_{n-1}x_{n-2}\dots x_1x_0) = x_0x_1\dots x_{n-2}x_{n-1}$

  **移数函数 【构成环形网 / 方格网 / 移数网】 $\alpha$**

  $\alpha(x) = (x \pm k) \mod N$

  **PM2I函数 PM2**

  $PM2_{\pm i}(x) = (x \pm 2^i) \mod N$

  有2n个互连函数

- 互连网络的结构参数：网络规模；节点度；节点距离；网络直径；对称性；等分宽度【把由N个结点构成的网络切成结点数**相同**（N/2）的两半（线等分宽度：对应网络最大流量）】
- 互连网络的性能指标：**时延 和 带宽**（通信时延【与网络传输状态有关】，网络时延【网络硬件特征决定，与程序行为和网络传输状态无关】，端口带宽，聚集带宽，等分带宽）
- 互连网络分类
  - 静态互连网络：线性阵列；环和带弦环；循环移数网络；树形和星形；胖树形；网格形和环网形（Illiac网络）；超立方体；带环n-立方体（n-CCC）；
  - 动态互连网络：
    - 总线网络：总线和共享存储器是两大瓶颈；解决带宽问题：多总线或多层次的总线

- 交叉开关网络：**带宽和互连特性最好**；**无阻塞**连接
- 多级网络：STARAN网络和间接二进制n方体网络**【仅在控制方式上不同】**
  - 相邻各级开关之间都有**固定的级间连接**；
  - 控制方式：**级控制；单元控制；部分级控制**
  - **STARAN网络**：**级控制【交换功能】；部分级控制【移数功能】**
  - **Omega网络**：级间互连采用**均匀洗牌连接方式**；采用**单元控制**方式

## ☆ 静态互连网络特征一览表

表 9.1　静态互连网络特征一览表

| 网络类型 | 节点度 $d$ | 网络直径 $D$ | 链路数 $l$ | 等分宽度 $B$ | 对称性 | 网络规格说明 |
|---|---|---|---|---|---|---|
| 线线阵列 | 2 | $N-1$ | $N-1$ | 1 | 非 | $N$ 个节点 |
| 环型 | 2 | $[N/2]$ | $N$ | 2 | 是 | $N$ 个节点 |
| 全连接 | $N-1$ | 1 | $N(N-1)/2$ | $(N/2)^2$ | 是 | $N$ 个节点 |
| 二叉树 | 3 | $2(h-1)$ | $N-1$ | 1 | 非 | 树高 $h=[\log_2 N]$ |
| 星型 | $N-1$ | 2 | $N-1$ | $[N/2]$ | 非 | $N$ 个节点 |
| 2D 网格 | 4 | $2(r-1)$ | $2N-2r$ | $r$ | 非 | $r\times r$ 网格，$r=\sqrt{N}$ |
| Illiac 网 | 4 | $r-1$ | $2N$ | $2r$ | 非 | 与 $r=\sqrt{N}$ 的带弦环等效 |
| 2D 环网 | 4 | $2[r/2]$ 向下取整 | $2N$ | $2r$ | 是 | $r\times r$ 环网，$r=\sqrt{N}$ |
| 超立方体 | $n$ | $n$ | $nN/2$ | $N/2$ | 是 | $N$ 个节点，$n=[\log_2 N]$（维数） |
| CCC | 3 | $2k-1+[k/2]$ | $3N/2$ | $N/(2k)$ | 是 | $N=k\times 2^k$ 节点<br>环长 $k\geqslant 3$ |
| $k$ 元 $n$-立方体 | $2n$ | $n[k/2]$ | $nN$ | $2k^{n-1}$ | 是 | $N=k^n$ 个节点 |

循环移数网络　2n-1　n/2　N个节点, n=log2N

## ☆ 动态互连网络特征一览表

表 9.5　动态网络的主要特性　　硬件复杂；处理器带宽增加

| 网络特性 | 总线系统 | 多级网络 | 交叉开关 |
|---|---|---|---|
| 单位数据传送的最小时延 | 恒定 | $O(\log_k n)$ 聚集带宽最大 | 恒定 |
| 每台处理机的带宽 | $O(w/n)\sim O(w)$ | $O(w)\sim O(nw)$ | $O(w)\sim O(nw)$ |
| 连线复杂性 | $O(w)$ | $O(nw\log_k n)$ | $O(n^2w)$ |
| 开关复杂性 | $O(n)$ | $O(n\log_k n)$ | $O(n^2)$ |
| 连接特性和寻径性能 | 一次只能一对一 | 只要网络不阻塞，就可实现某些置换和广播 | 全置换，一次一个 |
| 典型计算机 | Symmetry S1，Encore Multimax | BBNTC-2000 IBM RP3 | Cray Y-MP/816 Fujitsu VPP500 |
| 说明 | 总线上假定有 $n$ 台处理机；总线宽度为 $w$ 位 | $n\times n$ MIN 采用 $k\times k$ 开关，其线宽为 $w$ 位 | 假定 $n\times n$ 交叉开关的线宽为 $w$ 位 |

- 消息传递机制
  - 消息【**长度不固定**】：结点之间进行通信的**逻辑单位**；由若干个“包”（包含寻径所需目的地址的**基本单位**；【**长度固定**】）组成；包由“片”（通路所能处理的**最小单位**；头片 + 数据片；**长度经常是受网络大小的影响**）组成
  - 四种寻径方式：**线路交换、存储转发、虚拟直通、虫孔方式**【线路交换 和 包交换】
    - 线路交换：传递消息之前，先建立物理连接，包**无需存储**；适合于具有**动态和突发性**的大规模并行处理数据的传送；通道非共享；若频繁建立通路，时间开销大　$T=\frac{L+L_r\times(D+1)}{B}$
    - 存储转发：包缓冲存储器；网络时延大，与结点距离成正比　$T_{SF}=\frac{L}{B}(D+1)$
    - 虚拟直通：只要接收到用作**寻径信息**（头部），即进行路由选择；**时延与结点数无关**　$T=\frac{L+L_h\times(D+1)}{B}\approx\frac{L}{B}$
    - 虫孔方式：只要接收到用作**寻径信息**（**头片**），即进行路由选择；各片的传送可按**流水方式**进行；**传输延迟略大于线路交换**，网络冲突较小；不同包的片不能混合在一起传送；**结点是把一个片存储到缓冲器中**【与虚拟通道的区别】　$T_{WH}=T_f\times D+\frac{L}{B}=\frac{L+L_f\times D}{B}\approx\frac{L}{B}$
  - 解决死锁问题：**虚拟通道**【物理通道由所有的虚拟通道分时地共享】
  - 流控制策略（包冲突的解决）：**虚拟直通寻径方法，阻塞流控制方法，扬弃并重发方法，阻塞后绕道传送**
  - 寻径方法：**确定性寻径方法** 和 **自适应寻径方法**
    - 确定性寻径方法：**X-Y寻径；选播和广播寻径方法**
    - 4种通信模式：**单播；选播；广播；会议**
    - 参数：**通道流量 和 通信时延** 【**存储转发网络**中关注**时延**，**虫孔网络**中关注**流量**】

# 第十章 多处理机

## 多处理机分类

- 多处理机：**MIMD**【Flynn分类法】；**能实现指令、程序、任务级并行**
- MIMD分类

    - 并行向量多处理机（PVP）：多台巨型向量处理机采用共享存储器（SM）方式互连而成
    - 大规模并行处理机（MPP）：处理器数目很多
    - 机群多处理机（Cluster）：多个完整的计算机
    - 集中式共享存储器多处理机（SMP）
    - 分布式存储器多处理机
- 根据**存储器组织结构**，现有的MIMD机器分为两类
    - **集中式共享存储器多处理机【对称共享多处理机】**
        - 各处理器**共享一个集中式的物理存储器**
        - 均匀访存模型
    - **分布式存储器多处理机**
        - 特点：
            - 存储器在**物理上是分布**的；
            - 每个结点包含：处理器; 存储器; I / O; 互连网络接口;
            - 降低对存储器和互连网络带宽要求；对**本地**存储器的**访问延迟时间小**；支持大规模；通信复杂；**处理器之间**访问**延迟较大**
        - 分类：编址方式
            - **共享地址空间**：**共享存储器通信机制**；**分布式共享存储器系统**；**非均匀访存模型**
            - **独立地址空间**：**消息传递通信机制**【类似远程进程调用】；多以**机群**的形式存在
                - 同步消息传递：请求处理器发送一个消息后一直要**等到应答结果才继续运行**
                - 异步消息传递：数据**不经请求就直接送往**数据接受方

## 通信机制

- 不同通信机制的优缺点
    - 共享存储器通信
        - 与常用的**对称式多处理机**使用的通信机制**兼容**
        - 通信**数据量较小**时，通信**开销较低**，**带宽利用较好**
        - 通过**Cache技术**来**减少远程通信的频度**，延迟以及对共享数据的访问冲突
    - 消息传递机制
        - **硬件较简单**
        - 显式通信；编程者可重点注意并行计算的主要通信开销
        - **同步**与发送消息相关联，能减少不当的同步带来错误的可能性。

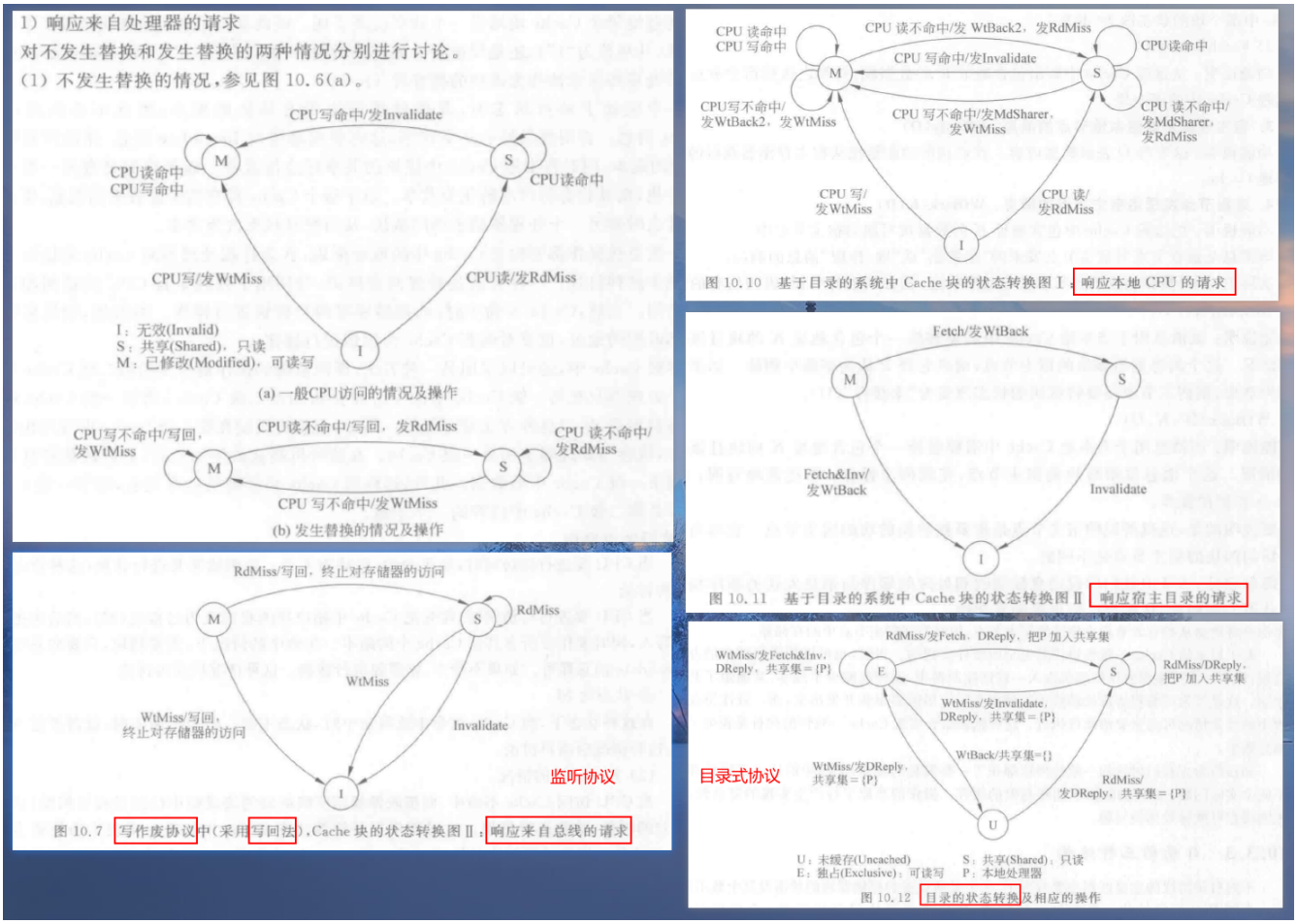
**一种通信机制的硬件模型上可建立另一种通信模式：在共享存储器上支持消息传递相对简单；反之较困难**

## 监听式协议和目录式协议

- **对称式共享存储器系统结构**
    - 特点
        - 多个处理器**共享一个存储器**
        - 支持对**共享数据和私有数据**的**Cache缓存**【有Cache的一致性问题】
    - 存储器的一致性定义：如果对存储器中某个数据项的任何**读操作**均可得到其最新写入的值，则这个存储系统是一致的
    - Cache提供的两种功能：

        - 共享数据的**迁移**：减少了对远程共享数据的**访问延迟**，也减少了对**共享存储器带宽**的要求
        - 共享数据的**复制**：减少了访问共享数据的**延迟**，也减少了访问共享数据所产生的**冲突。**
    - Cache的一致性协议：采用**不同的技术跟踪共享数据**的状态
        - **监听式协议**
            - Invalidate 与 WtMiss 的区别：**Invalidate不引起调块**
            - 每个Cache块设有一个共享位
            - 可扩放性很差
        - **目录式协议**
            - 设置目录表：记录每个可以调入Cache的数据块的状态位和若干指针位；目录承担了一致性协议操作的主要功能。
            - 任何一个数据块，可以快速地**在唯一一个处理机位置**(**宿主结点**)中找到记录该块状态的相关信息。使一致性协议**避免了广播操作。**
            - 位向量：记录哪些Cache中有副本；长度与处理器个数成正比，指定了共享集
            - 本地节点可以和宿主结点是同一个结点，也可以不是；远程节点可以和宿主结点是同一个结点，也可以不是。
            - **目录的3类结构：全映象目录；有限映象目录；链式目录**
                - 全映像目录：处理简单，速度快，**目录表最大**；目录所占用的空间与$N^2$【N: 处理器个数】成正比。
                - 有限映像目录：**位数固定**的目录项目；占用空间与$N \times \lceil \log_2 N \rceil$成正比；目录表较大；表满的时候**需要替换**
                - 链式目录：一个**目录指针链表**；链表长度不受限制【单链法，双链法 两种实现方法】
    - 两对概念区分
        - 写作废 和 写更新
            - 写作废协议：某个处理器更新其私有cache中的数据时，通知其他cache这一数据在他们中的副本无效；保证它拥有对该数据项的**唯一的访问权**
            - 写更新协议：某个处理器更新其私有cache中的数据时，将更新的数据发送给其他所有的cache，更新所有副本
        - 写直达 和 写回【cache和主存】
            - 写直达：当cache写命中时，cache与主存同时发生修改
            - 当cache命中时，只修改cache的内容，而不立即写入主存，只要当此行被换出时才写回主存

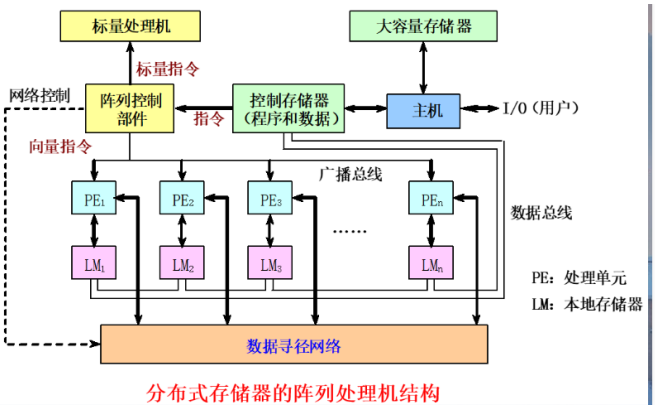
    无论是写更新还是写无效，共享存储器中的数据副本都可以通过写回或者写直达策略来维护cache一致性



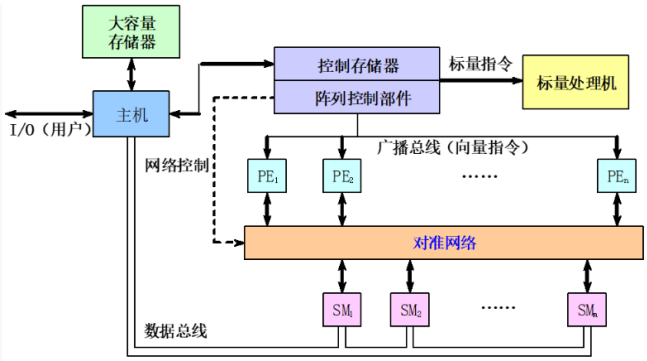

# 第十三章 阵列处理机

- **阵列处理机**，也叫**并行处理机**，是一种典型**SIMD计算机**，它由**多处理单元**(PE)按一定**互连方式**，在**同一控制部件**(CU) 控制下，对各自数据完成同一条指令规定的操作；**资源重复方法提高并行性**；利用并行性中的**同时性**；是**专用计算机**；**异构型多处理机系统**；
- 阵列处理机操作模型的五元组表示：**（N，C，I，M，R）**【N：处理单元（PE）数；C：控制部件直接执行的指令集；I：并行执行的指令集；M：屏蔽方案集；R：数据寻径功能集】
- 多处理机与并行处理机的本质差别在于并行级别的不同：多处理机实现**任务一级**的并行；而并行处理机则实现同一指令多数据流的**操作一级**的并行
- **地址屏蔽方案**只能"控制"**特定的PE子集**；**数据条件屏蔽方案**逻辑上，数据条件屏蔽能"激活"**任意PE子集**
- 阵列处理机的基本结构
  - **分布式存储器**的阵列机
    - 阵列部件、标量处理机和主机等构成
    - 特点
      - 由多个相同的处理单元PE组成，**PE有各自的本地存储器LM。**PE在阵列控制部件的统一指挥下，实现并行操作
      - 指令送到**控制部件进行译码。**指令执行顺序是**串行**进行的：分成**标量指令和向量指令**

        - **PE之间**通过数据寻径网络互连，实现通信，控制部件通过执行程序来控制数据寻径网络。
        - **PE的同步**是在**控制部件**的控制下由**硬件**实现
        - 各种阵列处理机的主要差别：**在于数据寻径网络的不同**
    - **共享存储器**的阵列机
        - 特点：
            - **共享多体并行存储器SM**通过**对准网络**与各处理单元PE相连
            - 存储模块的数目**等于或略大于**处理单元的数目。
            - **所有阵列指令都使用（长度为n的）向量操作数**（n为PE的个数）
            - **互连网络**是共享存储器**SM**和处理单元**PE**之间的**必由之路。**



分布式存储器的阵列处理机结构



共享存储器的阵列处理机结构

- Illiac IV阵列 特点：
    - 由**64**个处理单元部件PU 组成
    - 每一个处理单元有一个自己的本地存储器**PEM**
    - 任意处理单元间通信最短距离都不会超过7步
    - 一般情况，**n×n**个单元组成的阵列中，任意两个处理单元之间的**最短距离**不超过（**n - 1**）步。
    - 操作数来源：PE本身的寄存器；PEM(本地存储器)；CU的公共数据总线；PE的4个近邻
    - 阵列控制器CU与处理单元之间有4条信息通路：
        - 公共数据总线CDB： 向本地存储器（PEM）广播公共数据;
        - 模式位线： 将PE的状态传送至CU;
        - 指令控制线 （大约有200根）： 控制PE; CU总线:
        - 本地存储器（PEM）经过 CU总线把指令或数据送往阵列控制器CU
- 阵列处理机的并行算法
    - **阵列结构解有限差分方程**
        - 差分法求解的**精度与网格间距有直接的关系**，**网格越小，精度越高**，但求解所花费的**时空开销越大**
        - 可以**并行**地完成，从而提高处理速度（Illiac）
    - 矩阵加
    - 矩阵乘
    - **递归折叠求和算法**

# 计算题需要注意的问题

第一章

- Amdahl用的时候，看清楚当前部件运行时间占比是 在原始时间的占比，还是改进后总时间的占比！

(13 分)假定要将某一执行部件改进后速度提高 10 倍，改进后被改进部件执行时间占系统总时间的 50%。问改进后，获得的加速比 $S_p$ 是多少？

假设系统在改进前后的执行时间分别为 $T_0$ 和 $T_n$，则

$$S_p = T_0/T_n = \frac{1}{(1-F_e)+F_e/S_e} \quad (1)$$

由（1）式得

$$T_n = [(1-F_e)+F_e/S_e]\times T_0 \quad (2)$$

根据题意：

$$50\%\,T_n = (F_e \times T_0)/10 \quad (3)$$

由（2）式 -（1）式得

$$50\%\,T_n = (1-F_e) \quad (4)$$

$$Fe = \frac{10\times T_n}{2\times T_0} \quad (5)$$

由（4）、（5）式可得

$$S_p = T_0/T_n = 5.5$$

第三章

- 流水线性能计算的时候，需要看清楚 静态 还是 动态多功能流水线！！
- 对于一段程序的时空图，记得关注题目有没有说采用定向技术，以及指明了只有ALU等部件采用定向还是广义的定向；还有记得MEM阶段的写 后面指令 在ID阶段读 是不冲突的！

```
LOOP:   LW      R1,0(R2)
        DADDIU  R1,R1,#1
        SW      R1,0(R2)
        DADDIU  R2,R2,#4
        DSUB    R4,R3,R2
        BNEZ    R4,LOOP
```

| 指令 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LW | IF | ID | EX | M | WB | | | 前半周期写，后半周期读 | | | | | | | | | | | | | | |
| DADDIU | | IF | S | S | ID | EX | M | WB | | | | | | | | | | | | | | |
| SW | | | | | IF | S | S | ID | EX | M | WB | | | | | | | | | | | |
| DADDIU | | | | | | | | IF | ID | EX | M | WB | | | | | | | | | | |
| DSUB | | | | | | | | | IF | S | S | ID | EX | M | WB | | | | | | | |
| BNEZ | | | | | | | | | | | | IF | S | S | ID | EX | M | WB | | | | |
| LW | | | | | | | | | | | | | | | IF | S | S | IF | ID | EX | M | WB |

图 3.18　排空流水线时空图

第四章

- 并行编队，一个编队的时间取最大值；链接编队，一个编队内的指令执行时间求和
- 如果题目中指明了 “向量寄存器和功能部件之间的数据传送需要1拍”，则最开始的读存储器的一拍可以省略

4-1 在CRAY-1机器上执行下述4条向量指令（括号中给出了相应功能部件的执行时间），如果向量寄存器和功能部件之间的数据传送需要1拍，如果向量长度为64，问最快需要多少拍才能得到全部结果？

V0←存储器　　（从存储器中取数：7拍）
V2←V0+V1　　（向量加：4拍）
V3←V2<A3　　（按（A3）左移：4拍）
V5←V3∧V2　　（向量逻辑乘：2拍 ）
[ (7+1) + (1+4+1) + (1+4+1) +63 ]+ [ (1+2+1)+63 ]= 150

注意点1：　寄存器冲突

前三条指令链接执行，最后一条指令串行

- 在计算有几个编队的时候，要看清楚题目 有没有说到链接，没有就不使用链接；

  并行编队，需要注意的点：编队内不能有**功能部件冲突**，源寄存器冲突，目的寄存器冲突

```
LV    V1，  Rb     ；  取向量B
MULV  V2，V1，Fs   ；  B和标量s相乘   m=2
SV    Ra，  V2     ；  存向量A
```

第五章

- 记分牌算法中需要注意：读后写冲突！

| 指　令 | 指令状态表 | | | |
|---|---|---|---|---|
| | 流出 | 读操作数 | 执行 | 写结果 |
| L.D F6, 34(R2) | √ | √ | √ | √ |
| L.D F2, 45(R3) | √ | √ | √ | √ |
| MULT.D F0, F2, F4 | √ | √ | √ | |
| SUB.D F8, F6, F2 | √ | √ | √ | √ |
| DIV.D F10, F0, F6 | √ | 前面还没有读数据，因此还不能写 | | |
| ADD.D F6, F8, F2 | √ | √ | √ | |

  还有，数据准备好之后，读操作数如果结束了，$R_jR_k$ 要写成**no**

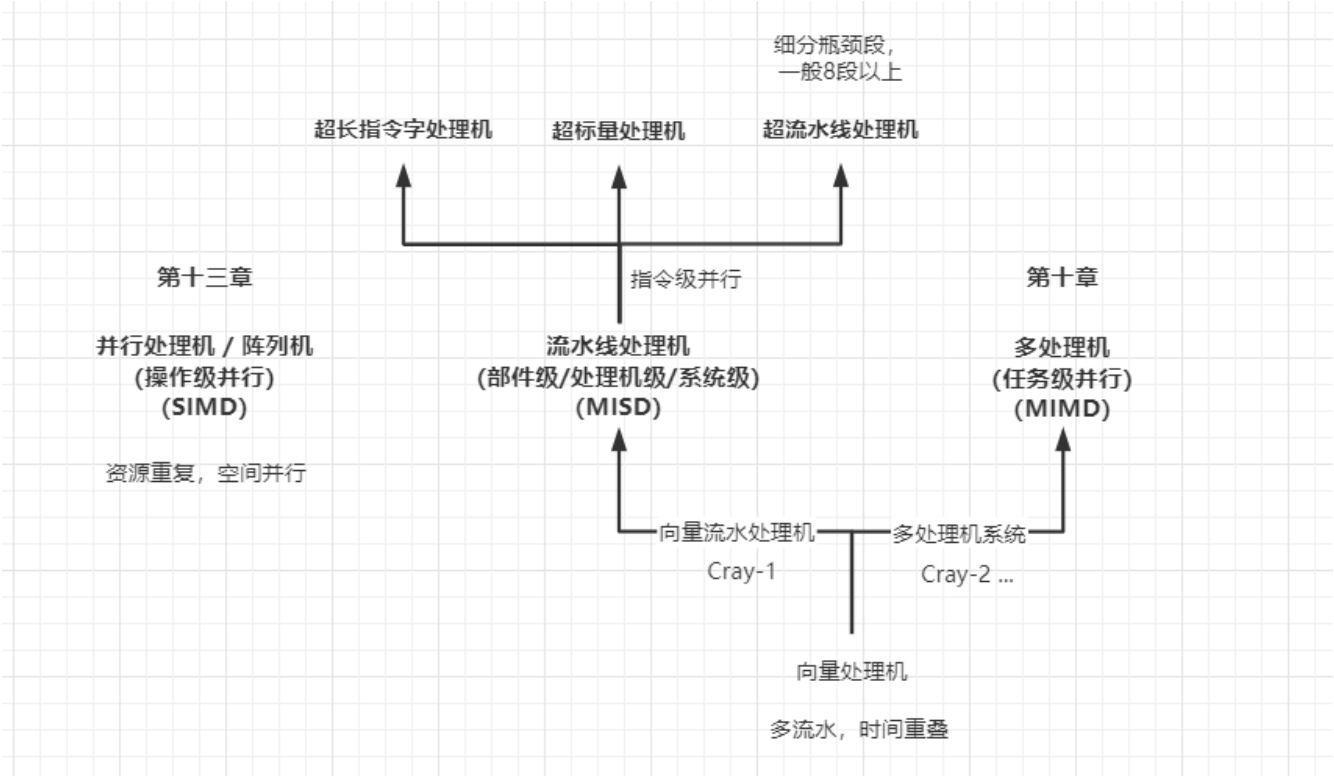
- 多流出技术的不同的时空图！

第九章

- 详见互连函数一览表

第十三章

- 递归折叠求和算法：注意两个点之间传的距离，根据互连网络的不同而不同；如果是移数网络或者立方体网络，每次缩成一半之后，两点之间的通信距离仍然是1；如果是环网，两点之间的通信距离会翻倍


细分瓶颈段，
一般8段以上
超长指令字处理机
超标量处理机
超流水线处理机
指令级并行
第十三章
并行处理机 / 阵列机
(操作级并行)
(SIMD)
资源重复，空间并行
流水线处理机
(部件级/处理机级/系统级)
(MISD)
第十章
多处理机
(任务级并行)
(MIMD)
向量流水处理机
Cray-1
多处理机系统
Cray-2 ...
向量处理机
多流水，时间重叠