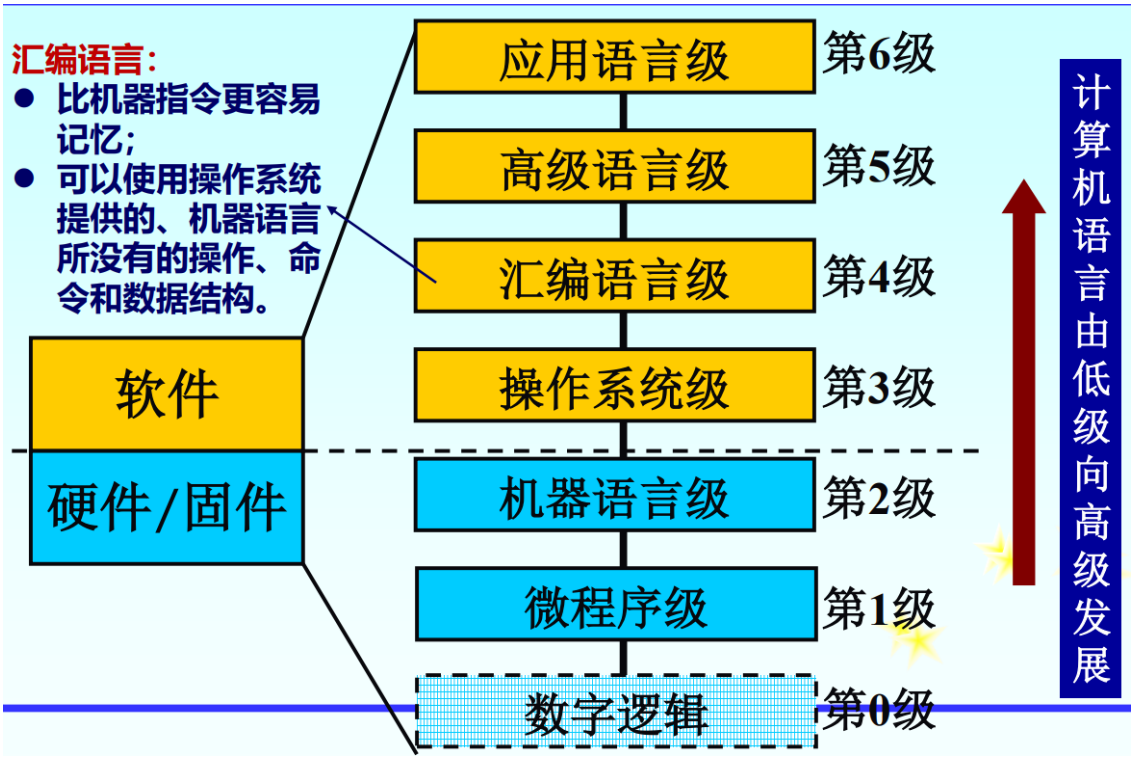
计算机体系结构

第1章 计算机系统结构的基本概念

1. 计算机系统 = 软件 + 硬件 + 固件，软件、硬件逻辑功能上等效
    - 计算机系统结构主要研究**软硬件功能分配**和对**软硬件界面的确定**
2. 从**使用语言**角度划分多级层级结构：



    - 第0、1、2级为硬件层次，第3-6级为系统软件（虚拟机）
    - 机器指令可直接硬件实现或微指令程序**解释**实现，高级程序语言可用汇编语言甚至位置了程序**解释**实现
3. 翻译与解释
    - 翻译：高级机器语言先**整体变换**到低级机器的等效程序，再在低级机器上运行

      解释：对高级机器语言每条语句**逐条解释**并运行
    - 解释速度慢，占用存储空间少
    - 通常1、2级解释，3-6级翻译
4. 计算机体系结构
    - 层次结构中**传统机器级**的系统结构，界面位于2、3级之间
    - 计算机系统结构是对计算机系统中各级界面的划分、定义及其上下的功能分配
    - **指令集结构**是软硬件之间的主要界面
    - 传统机器语言程序员能看到的计算机属性：数据表示、寻址方式、存储器组织、指令系统、存储系统组织、终端结构、管态与用户态的定义和切换、机器级IO结构、信息保护方式和保护结构
5. 计算机组成：
    - 计算机系统结构的**逻辑实现**，包括机器内的数据流和控制流的组成以及逻辑设计等
    - 包括：数据通路宽度、专用部件设置、各种操作对部件的共享程度、功能部件的并行度、控制机构的组成方式、缓冲和排队技术、预估预判技术
6. 计算机实现：计算机组成的**物理实现**

7. 计算机系统结构、计算机组成、计算机实现：
    - 区分：“有无”、需要事先约定的内容属于系统结构，具体逻辑设计属于组成
    - 关系：系统结构要考虑组成和实现的发展；组成要考虑系统结构和实现，决定于系统结构，受限于实现；组成与实现不是被动的，折中权衡； 实现是物质基础
8. 软硬件取舍基本原则：高的性能价格比，组成技术尽可能减少限制，软硬件合理分配
9. 计算机系统设计思路：从多级层次结构出发，由上往下、由下往上、从软硬件界面向两边
10. 计算机设计量化准则：
    - Amdahl定律：系统中某一部件采用某种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
        - 加速比S_n：$\text{加速比}=\frac{\text{改进后的性能}}{\text{改进前的性能}}$ 或 $\text{加速比}=\frac{\text{改进前的总执行时间}}{\text{改进后的总执行时间}}$
        - 加速比决定因素：可改进比例$F_e(<1)$、性能提高比$S_e(>1)$

$$Fe=\frac{\text{可改进部分占用的时间}}{\text{改进前整个任务的执行时间}}$$

$$Se=\frac{\text{改进前改进部分的执行时间}}{\text{改进后改进部分的执行时间}}$$

        - 改进后整个任务执行时间：$T_n=T_0\cdot(1-F_e+\frac{F_e}{S_e})$
        - 改进后系统加速比：$S_n=\frac{T_0}{T_n}=\frac{1}{(1-F_e)+\frac{F_e}{S_e}}$
    - CPU性能公式：$\text{CPU时间}=\frac{\text{IC}\times\text{CPI}}{\text{时钟频率}}$
        - CPU时间：$\text{CPU时间}=\frac{\text{CPU时钟周期数}}{\text{时钟频率}}$
        - 指令时钟数CPI：每条指令执行所用的时钟周期数，$\text{CPI}=\frac{\text{CPU时钟周期数}}{\text{IC}}$，IC为程序执行的指令数，多种指令时CPI可通过各种指令按频率加权求得
        - IPC：每个时钟周期执行的指令数，等于CPI倒数
        - 主频Fz：CPU时钟频率，CPU时钟周期等于主频倒数
    - 计算机系统性能衡量标准：
        - 吞吐量：单位时间处理请求数量
        - 响应时间：系统对请求做出响应的时间
        - 指令执行速度：$\text{MIPS}=\frac{\text{指令条数}}{\text{执行时间}\times10^6}=\frac{\text{Fz}}{\text{CPI}}=\text{IPC}\times\text{Fz}$
        - 峰值指令速度：MIPS
        - 等效指令速度：Gibson法：加权求和

$$\text{等效指令执行时间} T = \sum_{i=1}^{n} (W_i \times T_i)$$

$$\text{等效指令速度} MIPS = 1 / \sum_{i=1}^{n} \frac{W_i}{MIPS_i}$$

$$\text{等效} CPI = \sum_{i=1}^{n} (CPI_i \times W_i)$$

    - 算术平均时间、加权平均时间、归一化平均时间

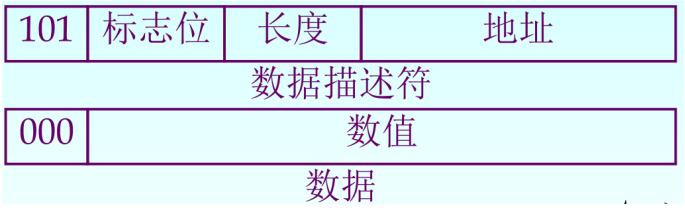
11. 软件可移植性技术：
    - 同一的高级语言
    - 采用系列机思想：同一厂家生产的具有相同系统结构，不同组成和实现的一系列计算机系统
        - 软件兼容：**向后、向上兼容**，**向后兼容最重要**，前后表示时间，上下表示机型
    - 采用模拟与仿真方法：在一台现有计算机上实现另一台计算机的指令系统
        - 模拟：全部用软件实现，A机器通过**解释方法**实现B，A为宿主机，B为虚拟机
        - 仿真：软件、硬件、固件混合实现，用A机器的**微程序**解释执行B，A为宿主机，B为目标机
12. 并行性：
    - 含义：**同时性**（时间重叠）、**并发性**（分时）
    - 划分：
        - 执行程序角度并行性等级由低到高：指令内部、指令之间、任务或进程之间、作业或程序之间
        - 处理数据角度并行性等级由低到高：位串字串、位并字串、位片串字并（同时对许多字的同一位(位片)进行处理）、全并行
        - 计算机信息加工角度并行性划分：存储器操作并行（并行存储系统）、处理器操作步骤并行（流水线处理机）、处理器操作并行（阵列处理机）、指令任务作业并行（多处理机）
    - 并行性开发途径：时间重叠、资源重复、资源共享
    - 多机系统耦合度：最低耦合（脱机处理系统）、松散耦合（通道通信线路互联）、紧密耦合（总线高速开关互联）
13. 计算机系统分类法：
    - 弗林Flynn分类法：根据**指令流**、**数据流**的**多倍性**特征对计算机系统进行分类
        - 分类：单指令流单数据流SISD、单指令流多数据流SIMD、多指令流单数据流MISD、多指令流多数据流MIMD
    - 库克Kuck分类法：根据**指令流**、**执行流**分类
        - 分类：单指令单执行流SISE，单指令多执行流SIME，多指令单执行流MISE（多道程序系统），多指令多执行流MIME
    - 冯泽云分类法：用**最大并行度**分类，最大并行度$P_m = \text{位宽}m \times \text{字宽}n$

第2章 数据表示与指令系统

1. 数据表示：
    - 计算机系统结构研究的首要问题：数据类型中哪些用硬件实现，哪些用软件实现，以及实现方法
    - **数据类型 = 数据表示 + 数据结构**
        - 数据表示：可以被**硬件直接识别**和**指令系统直接调用**的数据类型

        - 数据结构：结构化数据的组织方式，必须通过**软件映像**变换成机器具有的各种数据表示来实现
    - 高级数据表示：
        - 自定义数据表示：数据自身表明数据类型，可缩短机器语言与高级语言对数据属性说明间的语义差距
            - 带标识符的数据表示（存在一起）：

            - 数据描述符（分开存放）：



        - 向量数组数据表示：具有n个数据的数组，常用于向量处理机
        - 堆栈数据表示：高效实现堆栈数据结构，堆栈机器
    - 引入数据表示的原则：缩短程序运行时间，减少CPU与主存的通信量，通用性利用率是否高

2. 寻址方式：寻找操作数及其他信息的地址的技术
    - 体系结构研究的重点：分析各种寻址技术的优缺点，选择和确定寻址技术
    - 研究内容：编址方式、寻址方式、定位方式
    - 研究对象：寄存器、主存储器、堆栈、输入输出设备
    - 编址方式：对各种存储设备进行编码的方法
        - 编址单位：字编址（编址单位=访问单位）、字节编址（编址单位<访问单位）、位编址、块编址
        - 分类：
            - 分类编址：部件分类，每类从0开始单独编址，多个零地址空间
            - 统一编址：各种部件统一编成1个零地址空间
            - 隐含编址：事先约定编址方式，无零地址空间
        - 大多数计算机采用**分类编址**，按**字节编址**，将**通用寄存器、主存、堆栈**分类编址
    - 寻址方式：指令寻找或访问所需要的数据的方法
        - 根据是否指明寻址方式分类：显式（指令中设置专门的寻址方式字段）、隐式（用操作码隐含约定寻址方式）

        - 根据分类编址划分三类寻址：寄存器寻址、主存寻址、堆栈寻址
        - 具体寻址方式：立即寻址、直接寻址、间接寻址（间接地址在主存储器中，无偏移量）、相对寻址、变址寻址（基地址在变址寄存器中，有偏移量）
    - 定位技术：程序装入主存时，进行逻辑地址空间到物理地址空间的变换，即进行程序的定位
        - 直接定位：程序装入主存前，物理地址已经确定，直接装入即可
        - 静态再定位：软件方法将逻辑地址变换到固定的物理地址
        - 动态再定位：执行指令时，才形成访问主存的物理地址
    - 信息存储：任意存储、按字节数或位数的整数倍存储

3. 指令系统的设计与优化
    - 指令 = 操作码 + 地址码
        - 操作码 = 操作种类 + 操作数描述

        - 地址码 = 地址 + 地址附加信息 + 寻址方式
    - 指令操作码优化
        - 操作码三种编码方式：**固定长度编码**、**Huffman编码**、**扩展编码**
            - 等长扩展码表示：'-'间隔为指令位数表示，'/'间隔为指令条数表示
        - 信息源熵：信息源包含的平均信息量，$H=-\sum p_i log_2 p_i$
            - 信息冗余量：$R=1-\frac{H}{实际平均码长}$
            - 实际平均码长：$L=\sum p_i l_i$ 其中：p_i —第i个操作码出现的概率 l_i —第i个操作码的长度
    - 指令地址码优化：用一个比较短的地址表示一个比较大的虚拟地址空间
        - 举例：间接寻址、变址寻址、寄存器间址寻址
    - 指令字优化：可变长操作码与可变长地址码配合

4. 指令系统发展与改进
    - 改进思路：
        - 增强指令系统功能：复杂指令集计算机CISC
        - 简化指令系统：精简指令集计算机RISC
    - CISC：
        - 目标：强化指令功能，减少程序指令条数
        - 优化实现：
            - 面向目标程序：利用哈夫曼思想改进指令，增设强功能复合指令
            - 面向高级语言：统计使用频度来改进指令，面向编译、优化代码生成来改进指令，缩小指令系统与各种语言的语义差异，让机器具有多种指令系统，发展高级语言计算机
            - 面向操作系统：统计使用频度来改进指令，增设专用于OS的新指令，用硬件或固件实现OS的某些功能，由专门处理机完成OS
        - 问题：
            - **20%-80%律**：最常用的是简单指令，占总数20%，但使用频率占80%
            - 指令执行的CPI长
    - RISC：
        - 核心思想：**减少指令执行的CPI**
        - RISC为使优化编译器便于生成优化代码，应具有：三地址指令格式、较多的寄存器、对称的指令格式
        - RISC关键技术：
            - **重叠寄存器窗口技术**：让每个过程使用一个有限数量的寄存器窗口，并让各过程窗口部分重叠
            - **延时转移技术**：调整指令顺序，避免无条件转移指令或成功条件转移指令造成指令预取浪费，流水线断流等问题
                - 限制条件：被移动的指令与移动过程经过的指令不能有数据相关、被移动的指令不破坏条件码
            - **指令取消技术**：指令延时技术许多情况下找不到可用来调整的指令
                - 向后转移（循环程序）：循环体第一条语句安排在循环体前后出现两次
                - 向前转移（if-then）：转移不成功则执行下条指令，否则取消下条指令
                - 隐含转移：if-then语句中then部分只有一条指令，则if条件取反，如果取反后条件成立则取消下条指令，否则执行

- **指令流调整技术**：变量重命名消除数据相关，提高流水线效率
- **优化编译系统设计的技术**：使用大量寄存器，编译程序优化寄存器的分配和使用

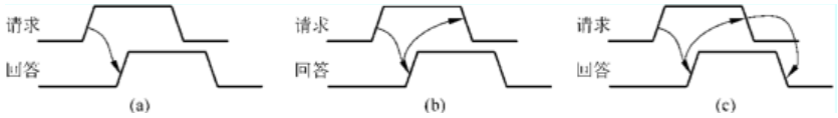
第3章 输入输出系统

1. 输入输出系统概述：
   - 范围：处理机和主存储器之外的部分，包括输入输出设备、设备控制器、与输入输出操作有关的软硬件
   - 特性与组织：
     - **异步性**（各设备按照自己的时钟工作） - **自治控制**（输入输出系统独立于处理机之外）
     - **实时性**（处理机必须及时处理设备） - **层次结构**（离处理机由近到远：I/O处理机、标准接口、设备控制器）
     - **与设备无关性**（独立于设备的标准接口） - **分类处理**（面向字符设备、面向数据块设备）
   - 三个发展阶段对应三种形式：
     - 程序控制输入输出：CPU直接进行I/O操作，包括全软件的、程序查询状态驱动的、中断驱动等
     - 直接存储器访问DMA：DMA开始和结束时需要处理及参与，传送过程中不用
     - I/O处理机：专门处理机完成外设I/O操作，实现CPU、I/O设备操作并行
2. 磁盘阵列RAID：将一组磁盘驱动器用某种逻辑方式联系起来，形成一个磁盘驱动器
   - 优点：成本功耗低，传输速率高，提供容错功能，价格便宜
   - RAID分级：

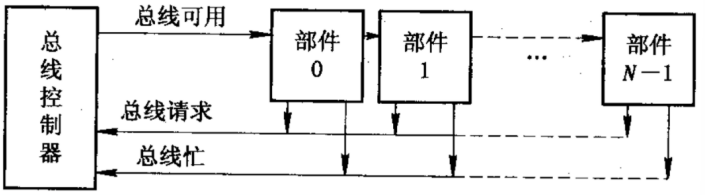
RAID级别	名称	数据磁盘数	可正常工作的最多失效磁盘数	检测磁盘数
RAID0	无冗余无校验的磁盘阵列	8	0	0
RAID1	镜象磁盘阵列	8	1	8
RAID2	纠错海明码磁盘阵列	8	1	4
RAID3	位交叉奇偶校验的磁盘阵列	8	1	1
RAID4	块交叉奇偶校验的磁盘阵列	8	1	1
RAID5	无独立校验盘的奇偶校验磁盘阵列	8	1	1
RAID6	双维无独立校验盘的奇偶校验磁盘阵列	8	2	2

3. 总线设计：
   - 特点：总线式一组能为多个部件**分时共享**的公共信息传送线路
   - 总线事务：总线上一对设备之间的一次信息交换的过程，包括**地址阶段**和**数据阶段**
   - 总线周期：完成一次总线操作的时间
   - 总线类型：
     - 按信息传送方向分：单向传输总线、双向传输总线（半双向、全双向）
     - 按用法分：专用、非专用
   - 总线性能指标：
     - 总线的数据宽度：I/O设备取得I/O总线后传送数据的**总量**，不同于数据通路宽度（数据总线线数）

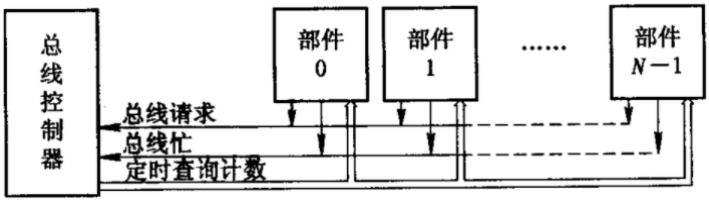
        - 总线的数据通路宽度：总线的线数，满足性能要求情况下应尽量减少
        - 总线带宽：总线最大数据传输速率，$总线带宽 = 总线宽度 \times 总线频率$
        - 总线负载：连在总线上的最大设备数量
        - 总线复用：不同时段总线同一根线路传输不同信号
        - 总线猝发传输：一个总线周期传输地址连续的多个数据
    - 总线定时控制：**同步定时**、**异步定时**（也称应答方式，包括不互锁、半互锁、全互锁）



    - 总线通信技术：
        - 同步通信：两部件之间信息传送是通过定宽、定距的系统时标进行的
        - 异步通信：分单向控制、双向控制
    - 总线仲裁：按照某种优先次序裁决，保证同一时间只有一个高优先级申请者取得总线使用权
        - **集中式**：总线控制逻辑基本集中放在一起
            - **链式查询方式**：3根控制线：1根总线忙BS、1根总线请求BR、1根总线批准BG



            - **计数器定时查询方式**：$2+\log_2 n$根控制线：1根总线忙BS、1根总线请求BR、$\log_2 n$根地址计数



            - **独立请求方式**：2n+1根控制线：1根总线忙BS、n根总线请求BR、n根总线批准BG
        - **分布式**：无中央仲裁器，总线控制逻辑分布到总线各个部件中，每次总线操作，只能有一个主方占用总线控制权，但同一时间里可以有一个或多个从方
            - **自举分布式**：设备优先级固定，使用前在总线请求线上发送请求信号，检测有否有优先级更高的设备
            - **冲突检测分布式**：设备需要使用时，先侦听总线是否忙，不忙则置忙并使用，发生冲突则都停止传输
            - **并行竞争分布式**：每个设备有仲裁号，设备根据仲裁算法决定是否占用总线

4. 通道处理机：
    - 通道种类：
        - **字节多路通道**：连接**多台低速设备**，有**多个子通道**，每个子通道连接一个设备控制器，每次传一个字节，采用**字节交叉方式**轮流服务
        - **选择多路通道（高速通道）**：物理上可连接多台设备，但**同一时刻只能选择一台设备**独占通道进行数据传输，每次传输一个**变长数据块**，逐个为多个外设服务
        - **数组多路通道**：连接多台高速设备，有多个子通道，分时共享整个通道，每次传送定长数据块
    - 通道流量分析：
        - 通道极限流量：通道能到达的最大通道流量，与选择一次设备时间T_s和传送一个字节时间T_D有关

$$f_{MAX.BYTE}=\frac{p\cdot n}{(T_S+T_D)\cdot p\cdot n}=\frac{1}{T_S+T_D}\text{字节/秒}$$

$$f_{MAX.SELETE}=\frac{p\cdot n}{(T_S/n+T_D)\cdot p\cdot n}=\frac{1}{T_S/n+T_D}\text{字节/秒}$$

$$f_{MAX.BLOCK}=\frac{p\cdot n}{(T_S/k+T_D)\cdot p\cdot n}=\frac{1}{T_S/k+T_D}\text{字节/秒}$$

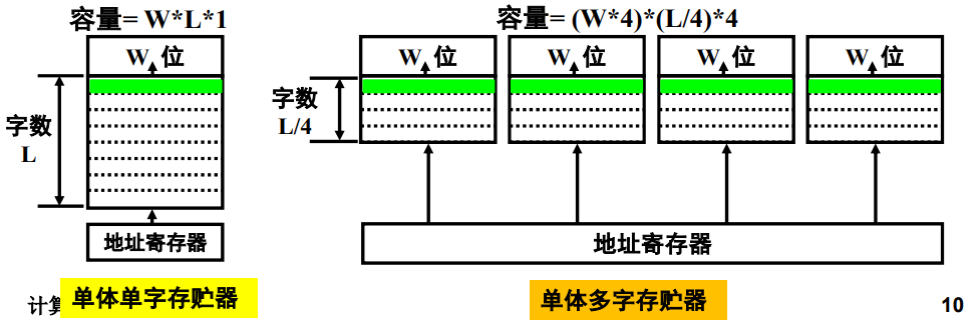
- 实际流量：

$$f_{BYTE}=\sum_{i=1}^{p}fi \qquad f_{SELETE}=\max_{i=1}^{p}fi \qquad f_{BLOCK}=\max_{i=1}^{p}fi$$

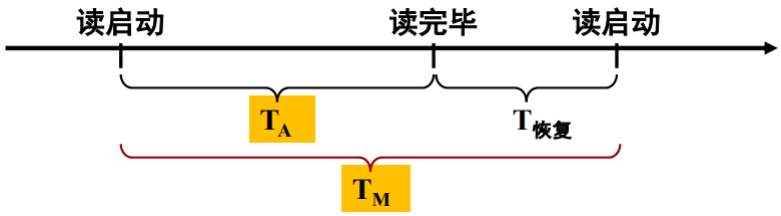
- 多通道极限流量等于各通道极限流量之和，多通道实际流量等于各通道实际流量之和

第4章 存储体系

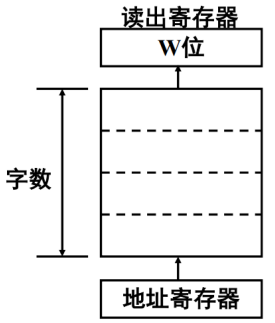
1. 存储体系概念
   - 存储系统：**两个或两个以上**的速度、容量、价格不同的存储器采用硬件、软件或软硬件结合的办法联接成的一个系统
   - 存储器性能指标：
     - 容量S_m = 字长W × 每个存储体字数L × 并行工作的存储体个数m



     - 速度：
       - 访问时间T_A，存储周期T_M

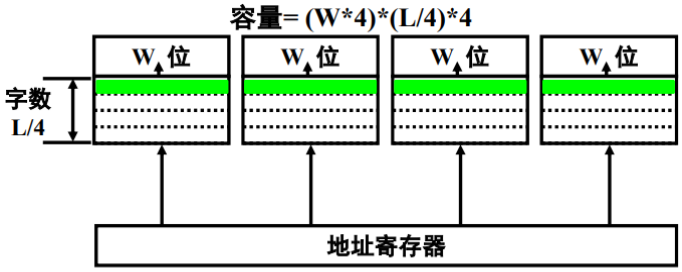


       - 频宽B_M：每秒数据传输量
     - 价格
   - 提高存储器性能方法：改进工艺和技术、构成并行主存系统（并行和重叠技术，eg: 多体单字交叉存储器）、使用存储器系统（主存、辅存）、存储体系
2. 并行存储系统：
   - 特点：一个存储周期内可以访问到多个数据
   - 类型：
     - **单体单字存储器**：一次可以访问一个存储器字，$B_M=\frac{W}{T_M}$



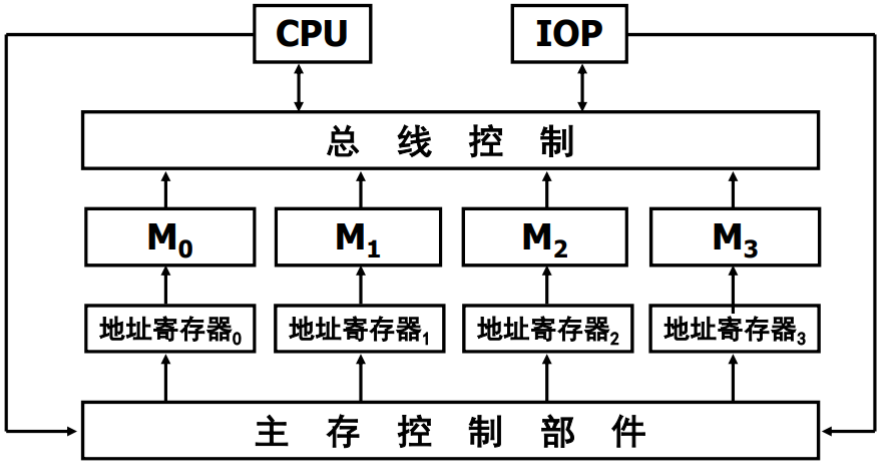
单体单字存贮器

- **单体多字存储器**：**一个**存储器，增加存储器字长，$B_M = m \times \frac{W}{T_M}$



单体多字存贮器

- **多体单字（多体交叉）存储器**：由**多个**容量较小、字长较短的相同存储器芯片组成，每个存储器有自己的地址译码与读写控制电路，字长都是CPU字长，多个存储体并行工作，$B_M = m \times \frac{W}{T_M}$，适合流水线处理



多体单字交叉存贮器

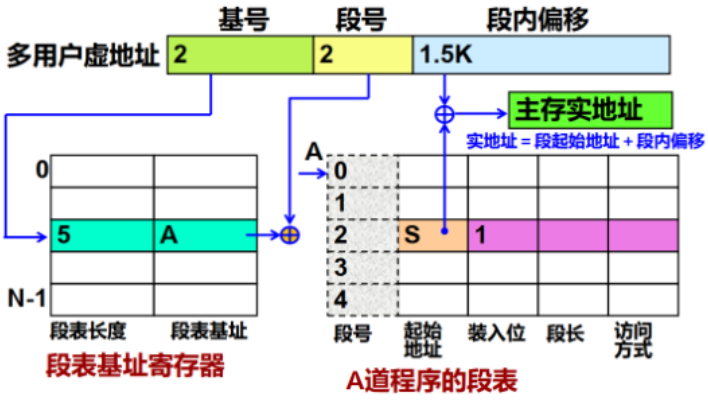
- **多体多字交叉存储器**：结合单体多字和多体单字

- 缺点：访问冲突大，取指令冲突、读操作数冲突、写数据冲突、读写冲突
  - 减少分体冲突：CPU字在主存中按模m交叉编址
    - 高位交叉：地址码高位区分存储体号，扩大存储器容量，主要使用（程序局部性原理）
    - 低位交叉：地址码低位区分存储体号，提高存储器访问速度，有效解决访问冲突

3. 存储体系：
   - 对应用程序员透明，速度接近最快的存储器，容量接近最大的存储器，单位容量价格接近最便宜的存储器
   - 典型分支：
     - **虚拟存储系统**：主存-辅存存储层次，解决**容量**问题，对应用程序员透明，对系统程序员不透明

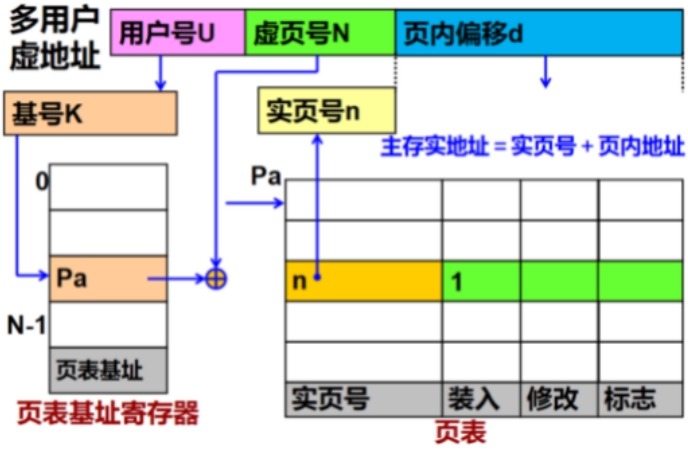
        - **Cache存储系统**：Cache-主存存储层次，解决**速度**问题，对应用程序员和系统程序员都透明
    - 程序局部性：
        - 时间局部性：未来要用到的信息可能是当前正在使用的（程序循环结构）
        - 空间局部性：未来要用到的信息可能是当前信息的相邻信息（程序顺序结构）
    - 多级存储层次的两个原则：
        - 一致性原则：同一信息在不同层次中的值保持相同
        - 包含原则：高层次存储器中信息一定包含在低层次存储器中
    - 性能参数（以二级存储层次为例）：每位价格c，命中率H，等效访问时间T_A
        - 每位价格c：$c=\dfrac{c_1S_{M_1}+c_1S_{M_1}}{S_{M_1}+S_{M_2}}$，即按容量对每位价格加权
        - 命中率H：CPU产生的逻辑地址能在M1中访问到的概率
            - 提高命中率：预取技术，预取后命中率$H'=\dfrac{H+n-1}{n}$，

              $n=$ 数据块大小 $\times$ 数据重复使用次数
        - 等效访问时间T_A：
            - M1、M2同时启动访问：$T_A=HT_{A1}+(1-H)T_{A2}$
            - M1、M2非同时启动访问：$T_A=HT_{A1}+(1-H)(T_{A1}+T_{A2})=T_{A1}+(1-H)T_{A2}$
            - 访问效率$e=\dfrac{T_{A1}}{T_A}=\dfrac{T_{A1}}{HT_{A1}+(1-H)T_{A2}}=\dfrac{1}{H+(1-H)\times\dfrac{T_{A2}}{T_{A1}}}$，与命中率、两级存储器速度比有关，越接近1越好
4. 虚拟存储系统（主存-辅存层次）：
    - 三种地址空间：程序空间与程序地址、主存空间与主存地址、辅存空间和辅存地址
    - 程序重定位：程序空间的程序地址转换为主存空间的物理地址，包括静态重定位、动态重定位
    - 虚拟存储器采用**地址映像表机构**实现程序的动态定位，主要有段式管理、页式管理、段页式管理
        - 段式管理：程序按逻辑意义分段，每段程序从0开始编址，可映射到主存任意位置
            - 多用户虚地址：（基号，段号，段内偏移）
            - 每道程序一个段表记录地址映射：（段号，起始地址，段长，装入位，访问方式）
            - 管理n道程序：n个段表基址寄存器



        - 页式管理：将主存空间和程序空间分成等大的页，按页进行调入、调出和管理
            - 多用户虚地址：（用户号，页号，页内偏移）

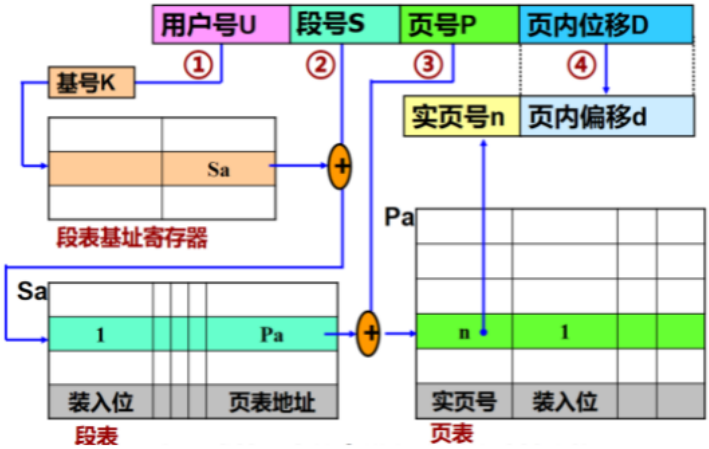
- 每道程序一个页表记录地址映射：（虚页号，实页号，装入位，修改位，专用位）
- 管理n道程序：n个页表基址寄存器



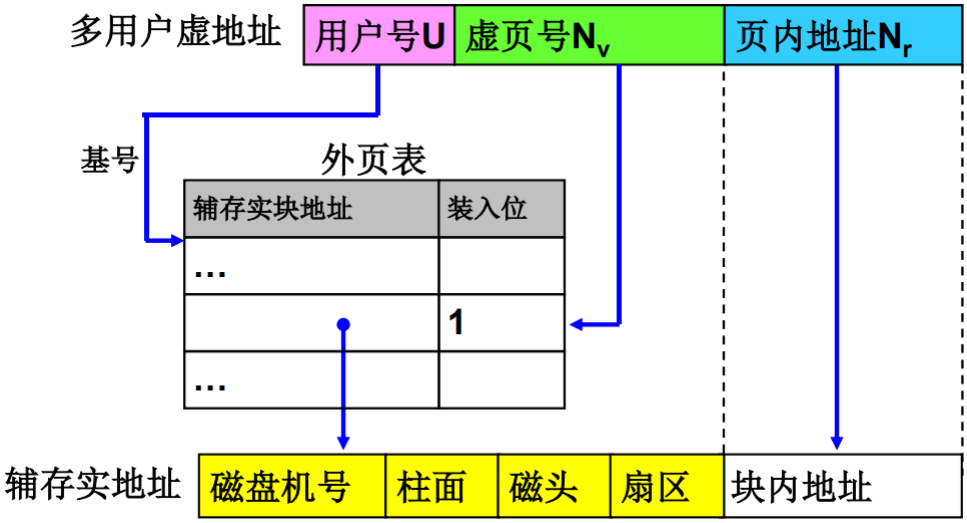
- 页表本身也按页管理，页表过大则采用多级页表，

$$\text{页表级数} = \frac{\text{虚拟内存划分的总页数}}{\text{一个页表可存储的页表项总数}}$$

- 段页式管理：程序按模块分段，段内分成固定大小的页
  - 多用户虚地址：（用户号，段号，页号，页内偏移）
  - 每道程序通过**一个段表**和**一组页表**管理，每个段有一个页表
  - 管理n道程序：n个段表基址寄存器

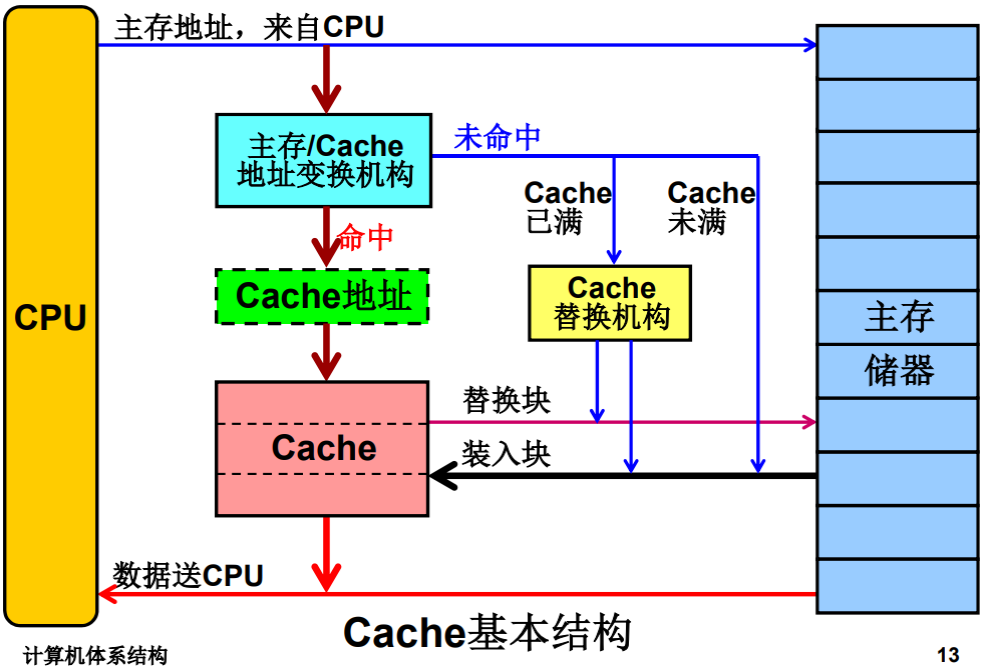


5. 页式虚拟存储系统需要解决的问题：
   - 地址映像与变换
   - 页面故障：虚存未装入主存，需要从辅存调入页，称为**外部地址变换**，记录虚存到辅存变换的页表称为**外页表**，外页表通常存在**辅存**中

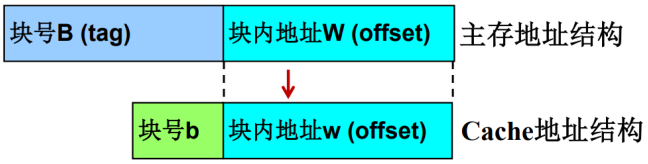


- 页面替换：
  - 算法：随机算法、先进先出FIFO算法、最近最少使用LFU算法、最近最久未使用LRU算法（最常用）、最优替换OPT算法
  - 颠簸：下次就要使用的页面本次被替换出去而发生的连续页面故障的现象
  - 堆栈型替换算法：对分配主存页面个数$m \leq n$，在任何时刻t，主存页面数集合B_t都满足关系：$B_t(m) \subseteq B_t(n)$
    - 命中率随着实页数的增加而增加
    - LRU和OPT是堆栈型，FIFO不是
- 提高等效访问速度：
  - 影响因素：减小主存访问时间，提高命中率，关键是提高**内部地址变换**的速度（用户虚地址 -> 主存实地址）
  - 方法：
    - **目录表**：相联存储器存放部分页表
    - **快慢表**：快表在高速存储器，慢表在主存，查表时同时查快慢表
    - **散列函数**：相联访问（20位）散列变换（硬件实现）为地址访问（5-8位），采用**相等比较器**避免散列冲突，从而增大快表容量

6. 高速缓冲存储器（Cache-主存层次）：
   - 基本结构：



   - Cache特点：
     - Cache-主存层次对所有程序员透明
     - Cache访问主存优先级高于其它系统访问主存的优先级
     - Cache与CPU、CPU与主存之间都有直接通路
   - 存在问题与解决：
     - **定位问题**：**地址映像规则与地址变换**，建立Cache块与主存块之间的对应关系，通常使用直接映像或组相联映像



     - **全相联映像与变换**：主存中任意块可装入Cache中任意块，冲突概率最低

        - 目录表：（主存块号，有效位），由相联存储器构成，大小等于Cache块数
    - **直接映像与变换**：主存中每一块只能装入Cache中唯一指定的块位置
        - 装入$Cache$块号 $=$ 主存块号 $\mod$ $Cache$块数
        - 主存地址：（区号E，区内块号k，块内地址offset）
        - 目录表：（主存区号，有效位）
    - **组相联映像与变换**：Cache分组，选组时用直接映像，组内用全相联映像
        - 装入$Cache$组号 $=$ 主存块号 $\mod$ 组数
        - 主存地址：

标记 (Tag)	组号(g位) Set	块内地址

        - Cache地址：

组号(g位) Set	组内块号(s位)	块内地址

        - 块表：2^g行，2^s列，查询时行内同时比较
        - 组内N块称为N路组相联
- **块冲突问题**：Cache替换算法
    - 算法：随机算法、FIFO、LRU
    - 实现方法：**堆栈法**（硬堆栈）、**比较对法**（逻辑电路、触发器），都是硬件实现
- **Cache一致性问题**：
    - 问题：写问题，包括CPU写Cache未写入主存，或IO写主存未写入Cache
    - 写命中策略：
        - 写直达法：CPU同时对Cache、主存执行写操作
        - 写回法：CPU只写Cache不写主存，Cache设置修改位，仅当替换时写回主存
    - 写不命中策略：
        - 不按写分配法：不命中时将要写的内容直接写入主存，不调块
        - 按写分配法：不命中时将要写的内容直接写入主存，再将块调入Cache
    - 一般策略：写直达法+不按写分配，写回法+按写分配

- Cache预取算法：
    - 按需取：Cache不命中时取一个块
    - 恒预取：无论是否命中，都将下一个块取入
    - 不命中预取：仅不命中时，将本块和下一个块取入

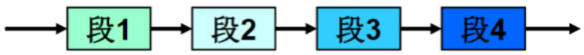
第5章 流水线和向量处理机

1. **重叠方式**：相邻指令之间让取指、分析、执行各部分的操作在时间上重叠进行，而指令内部的位操作之间仍为顺序串行
    - 速度对比：**顺序执行**：3nt，**一次重叠**：2nt+t，**二次重叠**：nt+2t
    - 二次重叠需要解决的问题：
        - 功能部件冲突：需要设置独立的取指、分析、执行部件
        - 访问主存冲突：
            - 分别设置独立编址的数字存储器和指令存储器
            - 采用多体交叉主存结构，需降低分体冲突：
                - 低位交叉存取：降低概率，不能解决冲突
                - 设置两个独立的Cache：指令Cache、数据Cache
                - 先行控制技术：缓冲技术和预处理技术
            - 在主存和指令分析部件之间增设FIFO指令缓冲寄存器，主存空闲时间预取指令存入指令缓冲器
    - 相关问题：数据相关、指令相关

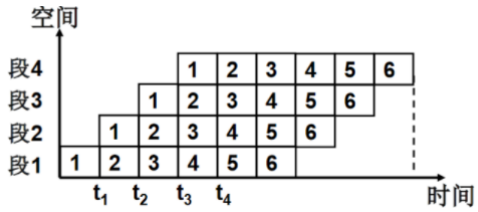
- 指令相关：延迟转移技术
  - 不允许修改指令
  - 允许修改指令，改变指令的执行方式，将指令相关转换为数据相关，统一按照数据相关处理
- 主存数据相关：推后读，同一单元要求先写后读
- 通用寄存器组相关：
  - 操作数相关、基址/变址相关
  - 推后读，或设置相关专用通路

2. **流水方式**

- 流水线技术：流水是重叠的延申，将指令的执行过程分解为多个子过程
- 概念：
  - 是一种基于时间重叠的并行处理技术
  - 每个子过程称为阶段，段的数目称为**流水线深度**
  - 每个流水段末尾或开端有**流水寄存器**，为简化一般不画
- 特点：
  - 只有连续提供同类任务才能充分发挥流水线效率
  - 流水线每个流水段要设置一个流水锁存器
  - 各流水段时间应尽量相等
  - 流水线需要装入时间和排空时间
- 流水线表示方法：**连接图**、**时空图**、**预约表**
  - 连接图：



  - 时空图：



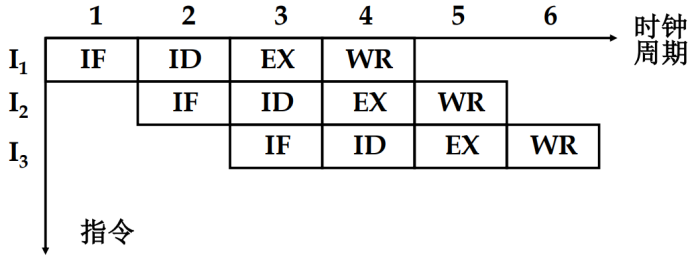
- 流水线分类：
  - 按处理级：部件级、处理机级、系统级
  - 流水线功能：单功能、多功能
  - 同一段时间能否执行不同类指令：静态多功能、动态多功能
  - 是否有反馈回路：线性流水线、非线性流水线
- 性能指标：
  - **吞吐率**：单位时间处理的指令条数 $TP = \frac{n}{T_K}$
    - 受限于瓶颈子过程，改进方法为**瓶颈段细分**或**瓶颈段重复设置**
    - 分为最大吞吐率和实际吞吐率
  - **加速比**：非流水线串行方式与流水线方式速度的比值 $S_p = T_{非流水} / T_{流水}$
  - **效率**：流水线设备实际使用时间与整个运行时间的比值

$$\eta = \frac{n个任务实际占用的时 - 空区}{m个段总的时 - 空区}$$

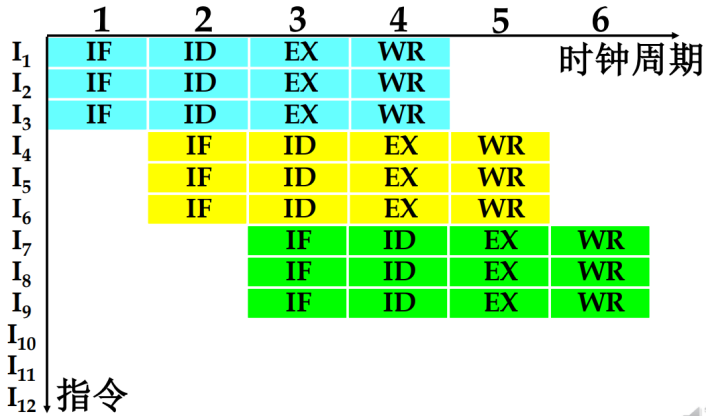
- 非线性流水线调度：解决功能段使用冲突问题
  - 方法：**二维预约表法**

- 流程：
  1. 构建禁止向量F=（调用同一段的节拍间隔数的集合）
  2. 由禁止向量构造初始冲突向量$C=(C_mC_{m-1}\dots C_1)$
  3. 向右移位，移位次数为等于0位的位数，然后与**初始冲突向量**按位或得到新的冲突向量
  4. 构造用冲突向量表示的流水线状态图
  5. 确定调度方案（最小启动循环、恒定循环等）

3. 向量的流水处理：
   - 水平处理方式：逐个求向量元素
   - 垂直处理方式：逐个计算各个计算阶段的所有数组元素，实现方法：多体交叉存储、向量寄存器组
   - 分组处理方式：向量太长时分组装入，寄存器组内采用垂直处理方式，组间采用水平处理方式
4. 指令级高度并行的超级处理机：$ILP(m,n)=m*n$，m表示每个时钟的启动次数，n表示每次启动的指令/操作个数
   - **超标量处理机(m, 1)**：设置m条指令流水线，每个时钟周期取m条指令
     - 利用了重复物理部件的空间并行性
     - 单发射处理机：每个周期只取一条指令，只译码一条指令，只执行一条指令，只写回一个结果，取指部件和译码部件各设置一套



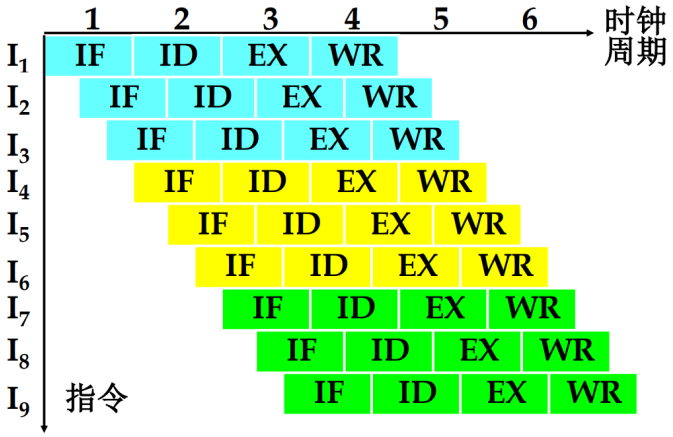
     - 多发射处理机：每个周期同时取多条指令、同时译码多条指令，同时执行多条指令，同时写回多个运算结果，需要多个取指令部件，多个指令译码部件和多个写结果部件



     - 相对单流水线普通标量处理机加速比，k段流水，N为指令条数：

$$S(m,1)=\frac{T(1,1)}{T(m,1)}=\frac{(k+N-1)\Delta t}{(k+\frac{N-m}{m})\Delta t}=\frac{\mathrm{m(k+N-1)}}{\mathrm{mk+N-m}}$$

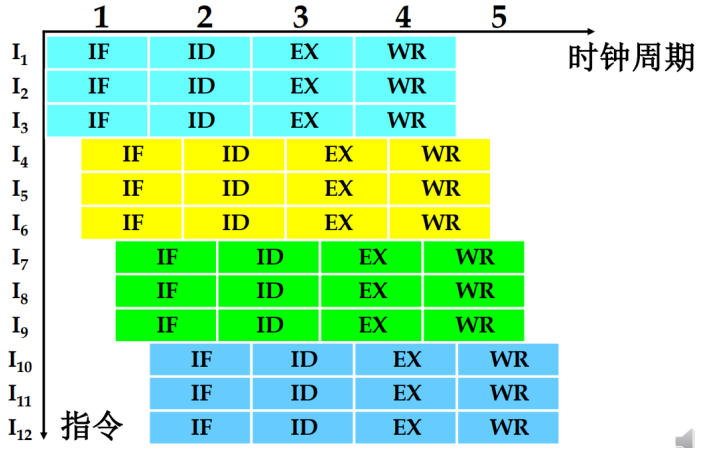
   - 超流水处理机(1, n)：一个时钟周期内发射多条指令的处理机，时钟周期变为原来的$\frac{1}{n}$，每个周期仍发出一条指令



- 利用时间并行性
- 相对单流水线普通标量处理机加速比，k段流水，N为指令条数：

$$S(1,n)=\frac{T(1,1)}{T(1,n)}=\frac{(k+N-1)\Delta t}{(k+\frac{N-1}{n})\Delta t}=\frac{\mathrm{n(k+N-1)}}{\mathrm{nk+N-1}}$$

- 超标量超流水处理机(m, n)：一个时钟周期内发射指令n次，每次发出m条

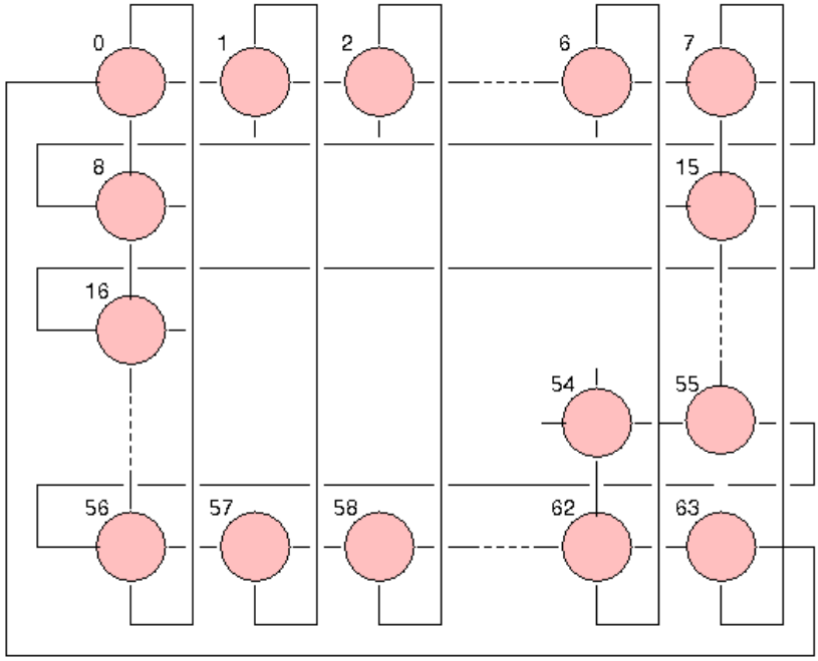


- 加速比：

$$S(m,n)=\frac{T(1,1)}{T(m,n)}=\frac{(k+N-1)\Delta t}{(k+\frac{N-m}{mn})\Delta t}=\frac{\mathrm{mn(k+N-1)}}{\mathrm{mnk+N-m}}$$

第6章 并行处理机（阵列处理机）与互联网络

1. 并行处理机（SIMD）：
   - 原理：多个处理单元PE按一定方式互连，在一个控制单元CU的控制下，对各自数据完成同一指令规定的操作
   - 根据存储器组成方式分为：**分布式存储器的阵列处理机、集中式共享存储器的阵列处理机**
   - 特点：依靠资源重复，利用的是同时性
2. 并行处理机的并行算法：
   - ILLIAC IV：PU_i与PU_{i-8}、PU_{i+8}、PU_{i-1}、PU_{i+1}相连

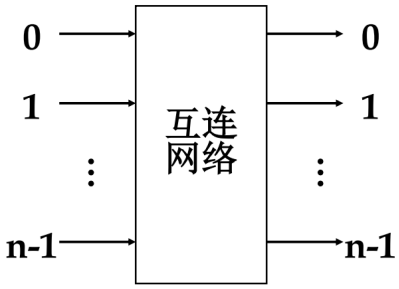


3. 互连网络ICN：
   - 表示方法：
     - 互连函数表示法：

$$\text{二进制：} f(x_{n-1}\cdots x_1x_0)=x_0x_{n-2}\cdots x_1x_{n-1}$$

$$\text{十进制：} f(\text{入端号}j) = \text{出端号}i$$

     - 图形表示法：



     - 输入输出对应表示法：

$$\begin{pmatrix} 0 & 1 & \cdots\cdots & N\text{-}1 \\ f(0) & f(1) & \cdots\cdots & f(N\text{-}1) \end{pmatrix}$$

   - 主要特性：
     - 网络规模（节点个数）
     - 节点度：与结点相连接的边数称为结点度
     - 距离：两个节点之间最少边数
     - 网络直径：节点距离最大值
     - 节点线长：物理距离
     - 对称性
   - 性能参数
     - 频带宽度：传输信息的最大速率
     - 传输时间：消息长度/频宽
     - 飞行时间：第一位信息到达接收方所花的时间
     - 传输时延：飞行时间与传输时间之和
     - 发送方开销：处理器把消息放到互连网络的时间

        - 接收方开销：处理器把消息从网络取出来的时间
        - 总时延 = 发送方开销 + 飞行时间 + 消息长度/频宽 + 接收方开销
    - 种类：
        - 静态互连网络：不能实现任意点到点的连接
            - 双向环网$\{PM2_{+0}, PM2_{-0}\}$
        - 循环互连网络：多次使用同一单级互联网络网络实现任意节点互联
        - 多级互连网络：将多套相同的单级互联网络连接
        - 全排列互连网络
        - 全交叉开关网络
        - 动态互连网络
    - 基本单级互连网络：
        - 立方体互连网络：$Cube_i$表示二进制右起第i位取反，其余位相同的点连接
        - PM2I单级互连网络：$PM2_{\pm i}$，加减2^i单元互连
        - 混洗交换单级互连网络：
            - 全混洗：$Shuffle(P_{n-1}...P_0) = P_{n-2}...P_0P_{n-1}$
            - 交换：$Cube_0$
        - 蝶形单级网络：$Butterfly(P_{n-1}P_{n-2}...P_1P_0) = P_0P_{n-2}...P_1P_{n-1}$
    - 多级互连网络：交换开关连接单级互连网络，交换开关允许一对一、一对多，不允许多对一
        - 多级立方体网络：第i级使用$Cube_i$编写入端出端
        - 多级混洗交换网络

第7章 多处理机与多计算机

1. 多处理机结构：
    - 分类
        - 共享存储器结构：
            - 物理共享：UMA
            - 逻辑共享：NUMA、COMA
        - 分布式存储器结构：MPP
2. 多核处理器：集成两个或以上独立处理单元（核）的处理器
    - 不一致性问题
        - 原因：共享可写数据、进程迁移、I/O传输
        - 解决：
            - 基于硬件：监听协议（监听总线）、基于目录的协议（适用于非总线连接）
            - 基于软件