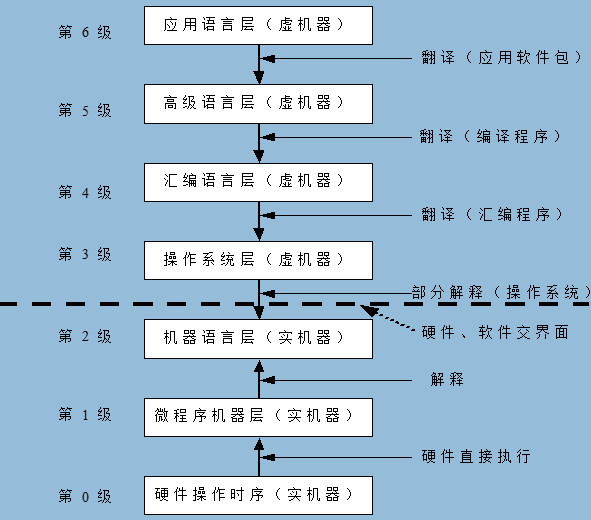
**第一章 计算机系统结构的基础知识**

**1.1计算机系统结构基本概念**

推动计算机性能高速增长因素：（1）计算机实现技术发展（2）计算机系统结构创新

* + 1. **计算机系统的层次结构**

**将计算机系统分为多级层次结构目的：**分清各级层次结构彼此之间的界面，明确各自的功能，以便构成合理、高效的计算机系统。



**1. 机器：**能存储和执行程序算法和数据结构的集合体。

① 有自己的语言，即可编程。

② 有实现语言的手段，即可执行程序。

通常将具体的由硬件/固件实现的机器称为**物理机（实机器）**。

**2.虚拟机：**利用软件手段构成的机器，也可称为**虚机器**。虚拟机是在实机器的基础上，开发各种软件，扩充机器功能而实现的。

**3.解释：**将一种语言的一条指令或一个语句转换为等效的另一种语言的一串指令后立即执行。**特点：**占用存储空间少，执行效率低，速度慢。（**边转换边执行**）

**4.翻译：**将一种语言的源程序全部转换为等效的另一种语言的程序后，才开始执行。**特点：**执行效率高，速度快，占用存储空间大。（**先转换再执行**）

**1.1.2 计算机系统结构的定义**

**1.经典定义：**计算机系统结构是机器语言（含汇编语言）程序员所看到的计算机的属性，即概念性结构与功能特性。

**计算机系统结构的实质：**确定软硬件的交界面。其界面之上是软件的功能，界面之下是硬件和固件的功能。

**2.广义定义：**计算机系统结构包括计算机设计的三个方面：指令系统结构、组成、硬件。

**3.透明性：**本来存在的事物或属性，从某种角度看似乎不存在。计算机系统在不同层次上，有着不同的属性，高层次的用户并不需要了解低层次属性的具体实现，即存在透明性。

**4.系统结构包含内容：**（1）指令系统（2）数据表示（3）操作数的寻址方式（4）寄存器的构成定义（5）中断机构（6）存储体系和管理（7）I/O结构（8）机器工作状态定义和切换（9）信息保护

**1.1.3 计算机组成和计算机实现**

**1.计算机组成定义：**计算机组成是指计算机系统结构的**逻辑实现**。包括物理机器级内部的数据流、控制流的形成与逻辑设计。

**2.计算机组成内容：**（1）数据通路宽度（2）功能部件的设置（3）系统内部并行性的确定（4）各种操作对功能部件的共享程度（5）可靠性技术的应用（6）缓冲、排队、预估、预测技术的应用（7）存储器逻辑组织方式及速度的确定（8）控制器的逻辑结构及组成方式（9）功能部件的互连方式及总线结构（对机器语言程序员透明）

**3.计算机实现定义：**计算机实现是指计算机组成的物理实现。

**4.计算机实现内容：**（1）CPU、主存等各种部件的物理结构（2）器件的集成度与速度的选择（3）信号传输技术的确定（4）器件、模块、插件、底板的划分与连接（5）电源、地线、通风、冷却等技术（6）整机的装配与调试

**5.透明性的体现：**对于**系统程序员**，系统结构是不透明的，组成和实现是透明的。对于**应用程序员**，系统结构、组成和实现均是透明的。

**6.系统结构、组成、实现直接关系：**

（1）同一系统结构可由多种组成技术实现，同一组成可有多种物理实现手段。（**系列机：**具有相同系统结构，而有不同组成与实现的一类机器）

（2）系统结构、组成、实现是互相影响的。计算机组成上取决于系统结构，下受限于实现技术。组成与实现又直接推动系统结构的发展。

（3）系统结构、组成、实现的内容是动态变化的。

**1.1.4 系统结构分类**

**1.Flynn分类：**SISD、SIMD、MISD、MIMD

**2.Flynn分类法缺陷：**（1）对MISD 类型的合理性计算机学术界上有争论。（2）对于标量及向量流水计算机的分类归属不很明确。（**注：**有学者将标量流水视为SISD类型，将向量流水视为SIMD类型。）

**1.2计算机系统的设计**

**1.2.1 计算机系统设计的定量原理**

**1.以经常性事件为重点：**对大概率事件赋予优先处理权和资源使用权，以获得全局最优的结果。是计算机设计中最重要、使用最广泛的设计原则。

**2. Amdahl定律**

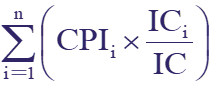
**3.CPU性能**

**（1）响应时间：**从用户向计算机系统输入一个任务后，直到获 得所需要的结果所需的等待时间，即完成一个任务所需的全部时间。

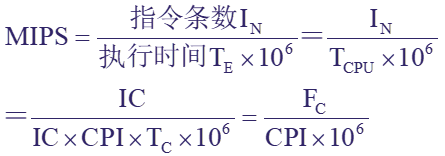
**（2）CPU时间：**CPU影响程序的时间，其中不包括I/O等待时间以及CPU转去运行其他程序所花的时间。

用户CPU时间 TCPU 可表示为：TCPU＝IC×CPI×TC

对指令进行分类：

CPI可表示为：(ICi/IC)反映了第i种指令在程序中所占的比例。

**4.MIPS和MFLOPS**

**（1）MIPS每秒百万条指令：**

**（2）MFLOPS每秒百万次浮点运算：**上述公式指令条数改为浮点运算次数即可（适用于衡量向量机性能）

**5.程序访问局部性原理：**程序执行时所访问的存储器地址分布不是随机的，而是相对地簇聚。（统计表明程序执行时间的90%都是在执行程序中10%的代码；即程序在执行过程中，往往频繁重新使用那些最近已被使用过的数据和指令）

**（1）时间局部性：**程序即将用到信息很可能就是目前正在使用的信息。

**（2）空间局部性：**程序即将用到信息很可能与目前正在使用信息在空间上相邻或临近。

**1.3计算机系统的性能评测**

略，PPT62-66系列及

**1.4计算机系统结构的发展**

**1.4.1 推动系统结构发展的主要因素**

1.**软件**是促进系统结构发展的**最重要的因素**

2.**应用**是促进系统结构发展的**最根本的动力**

3.**器件**是促进系统结构发展的**最活跃的因素**

**1.4.2 软件对系统结构的影响**

**1.软件可移植性：**一个软件可不经修改或只需少量修改便可由一台机器上移植到另一台机器上去运行，差别只是执行时间的不同。

**2.采用统一标准的高级语言：**采用与机器型号无关、可满足各种应用需要的通用高级语言进行软件的编程。

**3.采用系列机：**预先确定好一种系统结构(软硬件界面)，然后软件工作者依此进行系统软件设计；硬件设计者根据不同性能、价格要求，采用各种不同的组成和物理实现技术，向用户提供不同档次的机器。（统一的机器语言）

软件兼容：（1）向上兼容（2）向下兼容（3）向前兼容（4）向后兼容

**4.采用模拟和仿真技术：（模拟机器语言解释执行；仿真微程序解释执行）**

**（1）模拟：**模拟方法是指用软件方法在一台现有的计算机上实现另一台计算机的指令系统。（宿主机与虚拟机）

**（2）仿真：**用一台机器的微程序直接解释另一台机器的指令系统。（宿主机与目标机）

**（3）模拟与仿真区别：**

1）仿真方法中仿真微程序是存放在控存中(模拟方法中模拟程序存放在主存中)，因而实际上是有部分硬件(或固件)参与解释过程。因此仿真工作速度要比纯软件的模拟方法快。

2）用微程序仿真方法实现解释执行时，由于微程序机器级结构更依赖计算机的系统结构，再加上编写仿真微程序较为费时和复杂，因此**仿真只能在系统结构差距不大的机器之间使用**，对于系统结构差别较大的机器难于完全用仿真方法来实现软件移植。

3）仿真与模拟相结合：使用频率高指令用仿真，频率低的用模拟

**1.5计算机系统结构中并行性的发展**

**1.5.1 并行性的概念**

**并行性(parallelism)：**是指在同一时刻或是同一时间间隔内完成两种或两种以上性质相同或不相同的工作。

**同时性：**两个或多个事件在同一时刻发生的并行性。

**并发性：**两个或多个事件在同一时间间隔内发生的并行性。

**1.5.2 提高并行性的技术途径**

**1.时间重叠：**让多个处理过程在时间上相互错开，轮流重叠地使用同一套硬件设备的各个部分，以加快硬件周转而赢得速度。

**2.资源重复：**通过重复设置硬件资源，大幅度地提高计算机系统的性能。

**3.资源共享：**是一种软件方法，它使多个任务按一定时间顺序轮流使用同一套硬件设备。

**1.5.3单机系统中并行性的发展**

**1.时间重叠原理的应用：**在发展高性能单处理机过程中，起主导作用的是时间重叠原理。（部件功能专用化）

**2.资源重复原理的应用：**在单处理机中，资源重复原理的运用已十分普遍。（1）多体存储器（2）多操作部件（3）阵列处理机（并行处理机）

**3.资源共享原理的应用：**在单处理机中，资源共享的概念实质上是用单处理机模拟多处理机的功能，形成所谓虚拟机的概念。（分时系统）

**第二章 指令系统的设计**

**2.1指令系统结构的分类**

**指令系统：**机器所具有的全部指令的集合。指令系统反映了计算机的基本功能，是机器语言程序员所看到的机器的主要属性之一。指令系统是软件与硬件之间的一个主要分界面，也是软件与硬件之间互相沟通的桥梁。

**~~2.1.1 指令系统的分类方法（分类五个因素）~~**

~~1.在CPU中操作数的存放方法。（堆栈、累加器、通用寄存器）~~

~~2.在每条指令中显式指明的操作数个数。~~

~~3.操作数的寻址方式。~~

~~4.指令系统所提供的操作类型。~~

~~5.操作数的类型与大小。~~

**2.1.2 指令系统结构的三种类型**

**1.堆栈型：**指令中的操作数默认存放在堆栈的栈顶。

**2.累加器型：**指令中的操作数默认存放在累加器中。

**3.通用寄存器型：**指令中明确给出操作数所在的寄存器。

（1）RM结构（2）RR结构（3）MM结构

**2.2指令系统设计和优化**

**2.2.1指令系统功能设计**

确定软、硬件功能分配，即确定哪些基本功能应该由硬件实现，哪些功能由软件实现比较合适。

**硬件实现的特点：**速度快、成本高、灵活性差

**软件实现的特点：**速度慢、价格便宜、灵活性好

**2.2.2 指令系统结构的两种设计策略**

**1.RISC精简指令系统计算机：**尽可能地把指令系统简化，不仅指令的条数少，而且指令的功能也比较简单。

**2.CISC复杂指令系统计算机：**增强指令功能，把越来越多的功能交由硬件来实现，并且指令的数量越来越多

**2.2.3指令格式优化**

**1.指令由操作码和操作数两部分组成:**

操作码部分：说明操作种类、操作数的数据类型。

操作数部分：说明操作数地址、地址附加信息和寻址方式标志。

**2.定长指令编码格式：**指令字长固定。指令规整，译码简单，会出现冗余码点。适用于寻址方式和操作类型非常少时。

**3.变长指令编码格式：**指令字长可变。冗余少，平均指令长度短，指令不规整，译码复杂，不同指令执行时间不一致。适用于寻址方式和操作类型非常多时。

**4.混合型指令编码格式：**通过提供一定类型的指令字长，期望能够兼顾降低目标代码长度和降低译码复杂度两个目标。

**2.3.4 指令操作码的优化**

1.哈夫曼编码：略，PPT20-31

**第三章 流水线技术**

**3.1流水线的基本概念**

**3.1.1 什么是流水线**

**1.流水方式：**把一个重复的过程分为若干个子过程，由专门功能部件来实现。把多个处理过程在时间上错开，每个子过程都可以有效地在其专用功能段上与其它子过程同时执行。

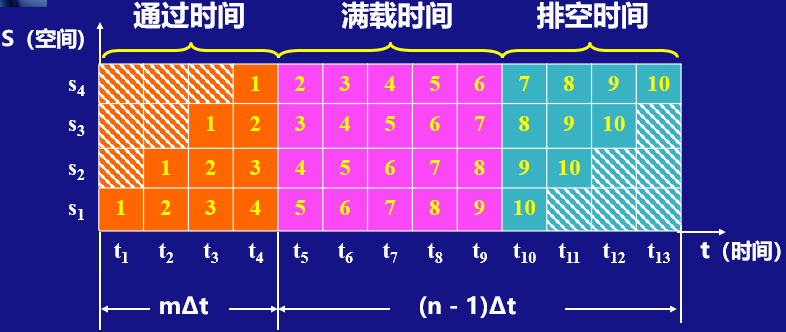
**2.流水线的级（段）：**流水线中的每个子过程及其功能部件称为流水线的级或段，段与段相互连接形成流水线。

**3.流水线的深度：**流水线中的段数。

**3.1.2 流水线的通过时间和排空时间**

**1.通过时间：**第一个任务从进入流水线到流出结果所需的时间。

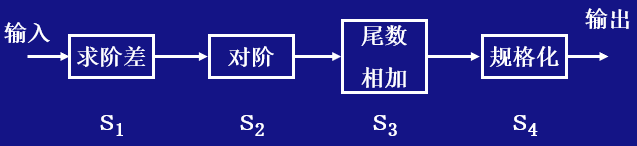
**2.排空时间：**最后一个任务从进入流水线到流出结果所需的时间。



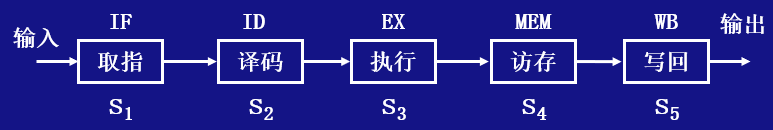
**3.2流水线的分类**

**3.2.1 按处理级别分**

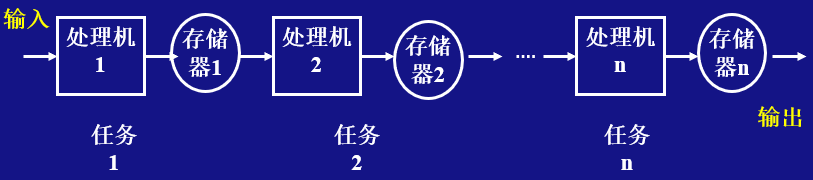
**1.部件级——运算操作流水线：**将复杂的算逻运算分段组成流水工作方式。



**2.处理机级——指令流水线：**把一条指令解释过程分成多个子过程组成流水工作方式。



**3.系统级——处理机间流水线（宏流水线）：**将系统中多个处理机串联起来，对同一数据流进行不同的处理，每个处理机完成整个任务中的一部分专门任务。



**3.2.2 按功能分**

**1.单功能流水线：**只完成一种固定功能的流水线。如浮点加法或乘法流水线。

**2.多功能流水线：**同一流水线的各功能段可进行不同的连接，使流水线在不同时间或同一时间内完成不同的功能。

**3.2.3 按工作方式分**

**1.静态流水线：**在同一时间内，流水线只能以一种功能方式工作。静态流水线可以是单功能流水线也可以是多功能流水线。当是多功能流水线时，则从一种功能方式变为另一种功能方式时，**必须先排空流水线**，然后为另一种功能设置初始条件后方可使用。

**2.动态流水线：**在同一时间内，可以将流水线中的不同功能段连接成不同的功能子集（前提条件是功能部件的使用不发生冲突），以完成不同的运算功能。

**3.2.4 按连接方式分**

**1.线性流水线：**从流水线的输入到输出，每个功能段只允许经过一次，流水线中不存在反馈回路。一般的流水线均属于线性流水线。

**2.非线性流水线：**流水线各功能段之间除了串行连接外，还存在反馈回路，因此从流水线的输入到输出过程中，某些功能段将被数次通过。非线性流水线适合于进行线性递归运算。

**3.2.5 按任务流动方式分**

**1.顺序流动流水线：**输出端的任务流出与输入端的任务流入顺序完全相同。

**2.乱序流动流水线：**输出端的任务流出与输入端的任务流入顺序不完全相同。在乱序流动流水线中，当某任务阻塞时，后面的任务可绕过它继续流动。

**3.2.6 标量处理机与向量流水处理机**

**1.流水线处理机：**指令执行部件中采用了流水线的处理机。

**2.标量处理机：**处理机不具有向量数据表示和向量指令，仅对标量数据进行流水处理。

**3.向量流水处理机：**具有向量数据表示和向量指令的流水处理机。

**3.3流水线的性能指标**

**略，PPT 21-55**

**1.流水线瓶颈问题解决方法：**

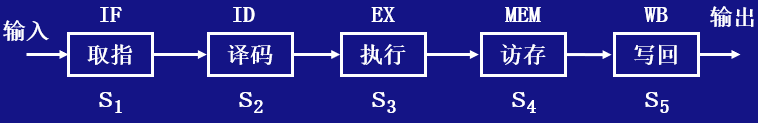
**（1）细分瓶颈段：**将瓶颈子过程进一步细分成若干个子子过程，使每一个子子过程与其他子过程时间相等。

**（2）重复设置瓶颈段：**在瓶颈段，并联设置多套功能段部件，使它们轮流工作。

**2.吞吐率、加速比和效率的关系：**E＝Tp·Δt＝Sp/m＝Sp/Spmax（**多功能流水线不适用**）

**3.4流水线的相关与冲突**

**3.4.1 经典5段流水**

**1.**

**2.**

**3. 采用流水线方式实现时需解决的问题：**

（1）保证不会在同一时钟周期要求同一个功能段做两件不同的工作。

（2）避免IF段的访存（取指令）与MEM段的访存（读/写数据）发生冲突。可以采用分离的指令存储器和数据存储器；一般采用分离的指令Cache和数据Cache。

（3）ID段和WB段可能会访问同一寄存器，产生冲突。可将**写操作**安排在时钟周期的**前半拍完成**，将**读操作**安排在时钟周期的**后半拍完成**。

（4）流水线为了能够每个时钟周期启动一条新的指令，就必须在每个时钟周期进行PC值的增量操作，并保留新的PC值。这种操作必须**在IF段完成**，以便为取下一条指令做好准备。但分支指令也可能需要改变PC的值，而且是在MEM段进行，这会导致冲突。

**3.4.2 相关**

**相关：**两条指令之间存在某种依赖关系。如果两条指令相关，则它们就有可能不能在流水线中重叠执行或者只能部分重叠执行。

**1.数据相关（真数据相关）：**先写后读关系，具有传递性，反映了数据的流动关系。

**2.名相关：**两条指令使用相同的名，但是它们之间并没有数据流动，则称这两条指令存在名相关。（**反相关：**先读后写关系；**输出相关：**先写后写关系）可通过换名技术消除。（编译器静态实现，硬件动态实现）

**3.控制相关：**指由分支指令引起的相关。

**3.4.3 流水线冲突**

**流水线冲突：**由于相关的存在，使得流水线中的下一条指令不能在指定的时钟周期执行。

**冲突带来的问题：**（1）错误的执行结果（2）流水线出现停顿

**约定：**当一条指令被暂停时，在该暂停指令之后流出的所有指令都要被暂停，而在该暂停指令之前流出的指令则继续进行

**1.结构冲突：**

**（1）概念：**当有多条指令进入流水线后在同一机器周期内争用同一功能部件所发生的冲突。（资源冲突、资源相关）

**（2）产生原因：**1）功能部件不是完全流水；2）资源份数不够

**（3）解决方案：**1）插入暂停周期（气泡）；2）设置相互独立的指令存储器和数据存储器或设置相互独立的指令Cache和数据Cache

**2.数据冲突：**

**（1）概念：**当相关的指令靠得足够近时，它们在流水线中的重叠执行或者重新排序会改变指令读/写操作数的顺序，使之不同于它们非流水实现时的顺序，则发生了数据冲突。

**（2）写后读冲突（RAW）：**也称为先写后读相关、流相关、WR相关、RAW相关等。RAW是最常见的数据冲突，对应于**真数据相关**。

**（3）写后写冲突（WAW）：**也称写写相关、WW相关、WAW相关、写后再写相关等。对应于**名相关中的输出相关**。一般乱序流水线才会发生该冲突。

**（4）读后写冲突（WAR）：**读后写冲突也称先读后写相关、变量名相关、反相关、RW相关、WAR相关等。对应于**名相关中的反相关**。一般乱序流水线才会发生该冲突。

**3.解决数据冲突的方法**

**（1）时间后推法：**遇到数据冲突时，就停顿后继指令的运行，直至前面指令的结果生成并写入寄存器。

**（2）采用定向技术：**又称**旁路技术**或**相关专用通路技术**。在执行过程中建立直接的专用通道，将执行结果直接送往需要的执行部件。

**（3）优化编译：**让编译器重新组织指令顺序来消除冲突（又称**指令调度**或**流水线调度**）

**4.控制冲突**

**（1）概念：**控制冲突即控制相关，主要是由转移指令引起的，是一种全局性的相关。当转移发生时，将使流水线的连续流动受到破坏。因此比起数据相关来，控制相关会使流水线丧失更多的性能，使流水线的吞吐率和效率严重下降。

**（2）处理分支指令最简单方法：**“冻结”或“排空”（即等待分支结果，会带来3个周期的分支延迟）

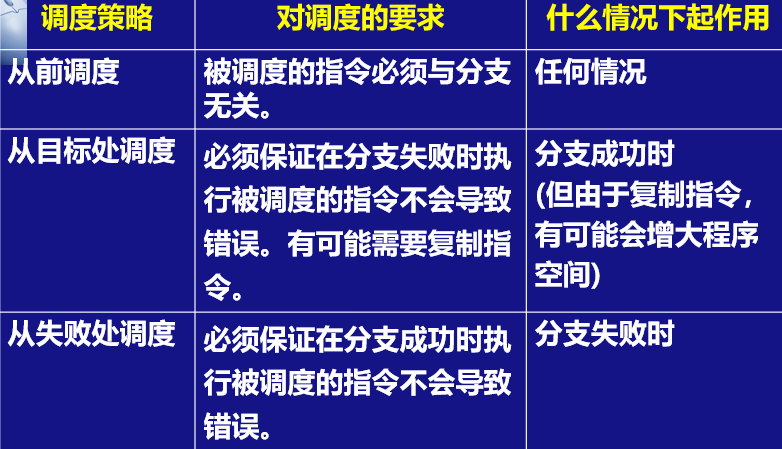
**（3）减少分支延迟的方法：**在流水线中尽早判断出分支转移是否成功。尽早计算出分支目标地址。（分支指令完成提前到ID段，此时分支延迟为1个周期）

**~~（4）静态转移预测：~~**~~当流水线译码出某指令为条件转移指令时，在其所需的条件码建立之前~~**~~，先将一个方向猜测为成功路径，通常是选取发生概率较高的路径，并在转移条件码生成之前只对这个方向上的若干条指令进行预取、译码和取操作数~~**~~，按该方向继续流动。若猜测失败，必须返回原分支点重新执行另一分支方向。~~

~~1）预测分支失败~~

~~2）预测分支成功：该方法需要先指导转移地址，对减少延迟没有好处~~

~~3）延迟分支：加入“延迟槽”~~

~~~~

**3.5非线性流水线调度**

**略，ppt 110-133**

**3.6向量处理机**

**3.6.1向量流水处理的主要特点**

1.在向量操作中，每个当前结果向量元素的计算与以前结果向量元素的计算是相互独立的，有利于发挥流水线的性能，允许向量流水线有较深的深度。

2.一条向量指令相当于一个标量循环，可降低对指令访问带宽的要求，也消除了由循环转移可能引起的控制相关。

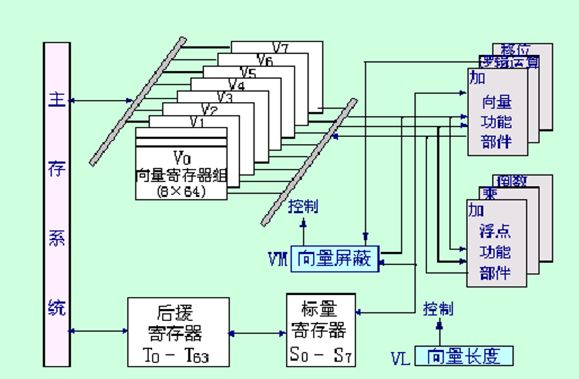
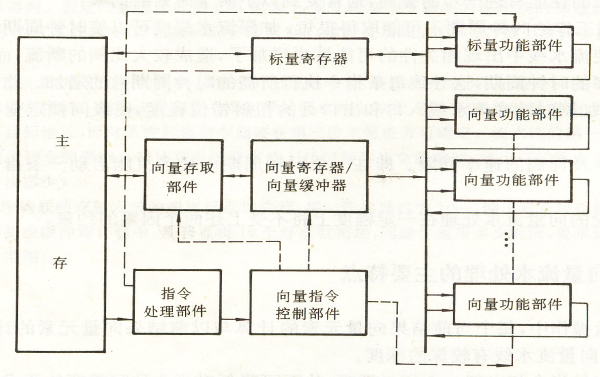
3.若向量指令所要访问的向量元素均相邻，则可利用多模块、交叉存取的方法加快向量元素的存取速度，减少访存等待时间的开销。

**3.6.2向量机的基本系统结构**

**~~1.存储器—存储器工作方式向量机：~~**~~向量操作源向量和目的向量都取或存到主存中。~~

**~~2.寄存器—寄存器工作方式向量机：~~**~~向量操作源向量和目的向量都取或存到向量寄存器。~~

3.向量机主要由**标量流水部件**和**向量流水部件**组成:（1）向量功能部件（2）向量存取部件（3）向量寄存器或向量缓冲部件（4）向量控制器（5）标量寄存器（6）标量处理部件



**~~4.向量处理与标量处理比较：~~**~~ppt 156-160~~

**3.6.3向量处理方式**

**1.横向加工方式：**逐个求出结果向量的各个元素（横向加工方式只适合于标量循环算法，不适合于向量流水处理，向量处理时会频繁出现相关和功能转换）

**2.纵向加工方式：**先对所有元素执行一种相同的运算，再对所有元素执行另一种相同的运算。（**特点：**（1）较高的吞吐率，适用于M-M型工作方式；（2）对主存带宽要求高）

**3.纵横向加工方式：**将向量元素分成长度为某个固定值的若干组，每个组内采用纵向加工方式，组之间采用横向加工方式。（适合于R-R型工作方式；向量长度超过向量寄存器长度时，必须对向量进行分组）

**3.6.4 Cray-1向量机**

**1.特点：**只要不出现**向量寄存器冲突（Vi冲突）**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**，大大加快了向量指令的处理。

**2.Vi冲突：**并行工作的各向量指令的源向量或结果向量使用了相同的Vi。

**3.功能部件冲突：**并行工作的各向量指令要使用同一个功能部件。

**3.6.5增强向量处理性能的方法**

**1.多功能部件的并行操作：**在向量机中采用独立的多个功能部件并行工作。

**2.并行操作约束条件：**只要没有**Vi冲突**和**功能部件冲突**即可并行工作。

**3.6.6 提高向量处理机性能的方法**

**1.设置多个功能部件：**设置的多个独立功能部件能并行工作，并各自按流水方式工作，从而形成了多条并行工作的运算操作流水线。

**2.链接技术**

**（1）链接特性：**当相邻两条指令存在**先写后读**相关时，只要前一条指令的结果向量的第一个元素产生并存入结果向量寄存器，就可以将它作为下一条指令的源操作数，启动下一条指令的运算，从而将两个或两个以上的功能部件链接起来。（**标量流水定向传送方法**在向量寄存器中的应用。）

**（2）链接实例：**ppt 156-166

**（3）使用链接技术的必要条件：**

1）指令间存在先写后读相关，且指令间不存在Vi冲突和功能部件冲突

2）只有当前一指令的第一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，若错过该时刻就无法进行链接。（当一条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，要求**先行的两条指令产生运算结果的时间必须相等**，才可链接）

3）进行链接的向量指令的向量长度必须相等，否则就不可能链接。

**3.分段开采技术**

**（1）概念：**当向量的长度大于向量寄存器的长度时，必须把长向量分成长度固定的段，然后循环分段处理，每一次循环只处理一个向量段。（由软硬件控制，对程序员透明）

**（2）分段开采实例：**ppt 170-174

**3.6.7 向量处理机的性能评价（重点，必考！）**

**略，见ppt 175-200**

**1.向量处理时间：**执行一条向量长度为n的向量指令所需的时间为：

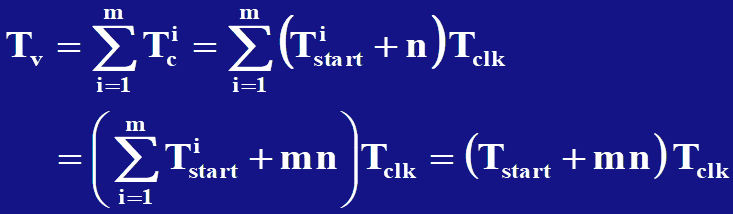
(其中：Tstart为启动时间或称延迟，n为向量长度)

**2.向量编队：**

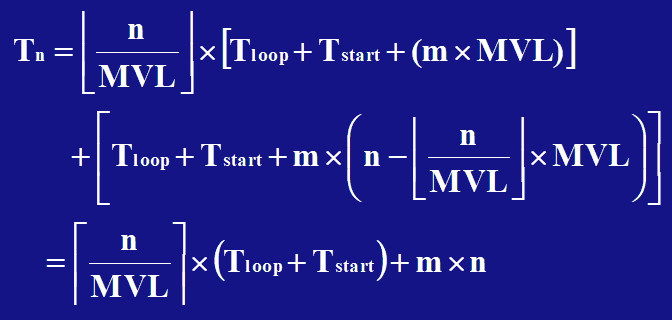
**（1）概念：**把几条能在同一个时钟周期内一起开始执行的向量指令集合称为一个编队。

**（2）组成编队条件：**同一编队指令不存在任何**功能部件冲突**、**Vi冲突**、**写后读冲突**

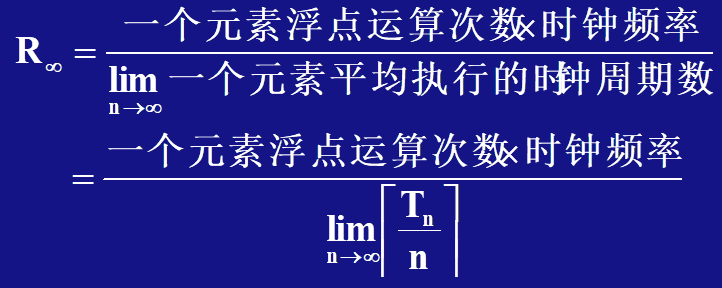
**（3）编队运行时间：**

（其中m为编队个数，n为向量长度）

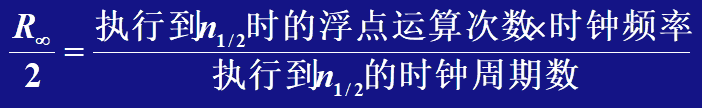
**（4）带分段开采的编队运行时间：**

（其中Tloop为分段开采所需的额外的时间开销，MVL为向量寄存器的长度）

**3.峰值性能：**表示当向量长度为无穷大时的向量流水线的最高性能，也称峰值性能



**4.半性能长度：**n½ 指向量处理机运行性能达到其峰值性能一半时必须满足的向量长度。



**5.向量长度临界值：**nv表示向量流水方式的工作速度优于标量串行方式工作时所需的向量长度临界值。

**第四章 指令级并行**

**4.1指令级并行**

基本概念：指令之间存在的潜在并行性称为指令级并行。

**4.1.1流水线处理机的实际CPI**

流水线处理机的实际CPI就是理想流水线的CPI加上各类停顿的时钟周期数：

CPI流水线=CPI理想+停顿结构冲突+停顿数据冲突+停顿控制冲突©

**~~4.1.2 IPC（Instructions Per Cycle）~~**

**~~IPC：~~**~~每个时钟周期内完成的指令条数。IPC是CPI的倒数。~~

**4.1.3 开发指令级并行常用方法**

**1.循环级并行：**循环程序不同迭代之间存在的并行性。最简单常用的指令级并行方法。

**2.开发循环级并行方法：**采用循环展开技术或采用向量指令和向量数据表示

**3.解决相关与流水线冲突问题（指令相关，则不能并行执行或只能部分重叠进行）**

**（1）相关：**是程序固有的一种属性，它反映了程序中指令之间的相互依赖关系。

**（2）三种相关：**数据相关、名相关、控制相关

**（3）冲突：**由于相关的存在，使得指令流中的下一条指令不能在指定时钟周期执行。

**（4）三种冲突：**结构冲突、数据冲突、控制冲突

**（5）解决相关与冲突的方法：**（1）保持相关，避免产生冲突（通过指令调度，静态调度或动态调度）（2）代码变换，消除相关

**4.2循环展开和指令调度**

**重点！略，ppt 9-23**

**1.循环展开技术：**把循环体代码复制多次并按顺序排列，然后相应调整循环的结束条件。

**2.循环展开和指令调度时要注意的问题：**

**（1）保证正确性：**需要注意循环控制和操作数偏移量的修改

**（2）注意有效性：**只有能够找到不同循环体之间的无关性，才能有效地使用循环展开

**（3）注意对存储器数据的相关性分析：**访存地址不同则相互独立，可以对调

**（4）注意新的相关性**

**4.3指令的动态调度（Tomasulo算法）**

**重点！略，ppt 27-72**

Tomasulo算法的核心是通过硬件进行动态调度，**允许乱序执行指令**。

**Tomasulo算法核心思想：**记录和检测指令相关，操作数一旦就绪就立即执行，把发生**RAW冲突**的可能性减少到最小；通过寄存器换名来消除**WAR冲突**和**WAW冲突**。

**第五章 存储层次**

**5.1 存储器的层次结构**

**5.1.1 从单级存储器到多级存储器**

需要解决容量、速度、价格之间的矛盾

**1.存储器层次结构：**把多种容量、速度、价格不同的存储器用硬件、软件或软硬结合的方式连接起来构成一个统一的整体，并能自动实现不同存储器之间的信息调度。

**2.依据：程序访问的局部性原理。**将计算机当前**频繁访问、活跃的程序和数据**存放在**高速小容量**的存储器中；将**近期不用、非活跃的程序和数据**存放在**速度较慢但容量较大、价格较低**的存储器中，当需要时再将它们自动调入高速存储器。

**3.存储层次结构技术希望：**

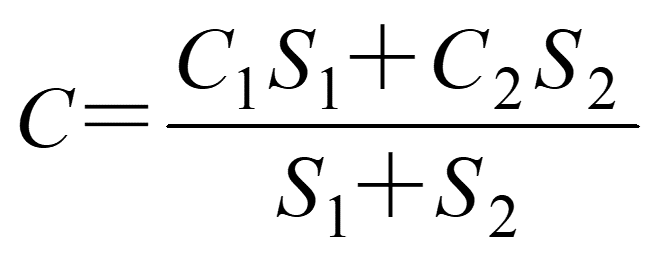
访问时间：TA≈min（TA1，TA2，… TAn）

存储容量：S≈max（S1，S2，… Sn）

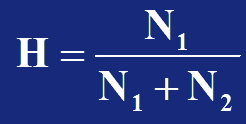
价格：C≈min（C1，C2，… Cn）

**5.1.2 存储层次性能参数（两级存储为例，M1为高速小容量，M2为低速大容量）**

**1.每位价格C：**

（其中C为价格，S为容量）

**2.命中率H和失效率F：**

（其中N1为访问M1次数，N2为访问M2次数）

**3.评价访问时间：**

TA＝TA1＋FTM=HTA1＋FTA2（其中TA1为访问M1时间，TA2为访问M1时间，TM为失效开销）

**5.1.3 “Cache－主存”和“主存－辅存”层次**

**1. Cache — 主存层次：**从CPU看速度是Cache的，容量是主存的

**（1）设置Cache的目的：**弥补主存速度的不足。

**（2）Cache — 主存层次的实现手段：**硬件。

**（3）Cache — 主存层次信息的传送单位：**块。块的大小通常为几十个字节。

**（4）失效处理：**等待调块。

**（5）**Cache对**应用程序员**和**系统程序员**均**完全透明**。

**（6）CPU可直接访问主存：**设置直接通道，Cache失效直接访存

**~~2.主存 — 辅存层次：~~**~~从整体看速度是主存的，容量是辅存的~~

**~~（1）设置辅存的目的：~~**~~解决存储器容量的问题。~~

**~~（2）主存 — 辅存层次的实现手段：~~**~~软硬件结合。~~

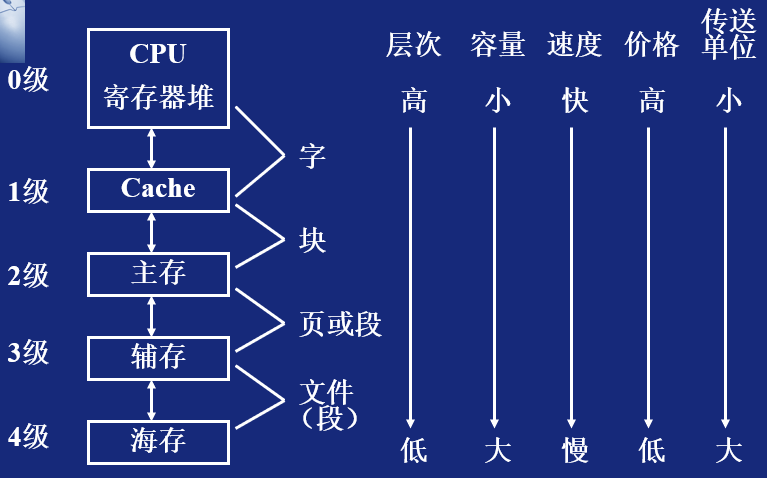
**~~（3）主、辅存之间的信息传送单位：~~**~~以页或段为单位。~~

**~~（4）失效处理：~~**~~程序切换。~~**~~失效：~~**~~在主存中没有找到要访问的程序或数据。~~

**~~（5）~~**~~主存 — 辅存层次对~~**~~应用程序员~~**~~是~~**~~完全透明~~**~~的，对~~**~~系统程序员~~**~~是~~**~~部分透明~~**~~的。~~

**~~（6）~~**~~对辅存的访问是通过访问内存实现的。~~

**~~3.多级存储器层次结构~~**

~~~~

**5.1.4 实现存储层次应考虑的问题**

1.当把一个块调入高一层(靠近CPU)存储器时，可以放到哪些位置上（**映像规则**）

2.当所要访问的块在高一层存储器中时，如何找到该块（**查找算法**）

3.当发生失效时，应替换哪一块（**替换算法**）

4.当进行写访问时，应进行哪些操作（**写策略**）

**5.2 Cache的基本知识**

**5.2.1 Cache基本工作原理和结构**

**1.Cache的基本结构：**

**（1）Cache存储阵列：**由高速存储器构成，用于存放主存信息的副本。容量小于主存，但编址方式、物理单元长度均与主存相同。

**（2）地址映像变换机构：**用于实现主存地址与Cache地址转换的部件，一般由相联存储器组成，用以记录主存内容存入Cache时两者地址的对应关系。

**（3）替换策略实现机构：**根据一定的算法，用硬件实现块的替换。

**2. Cache的工作过程**

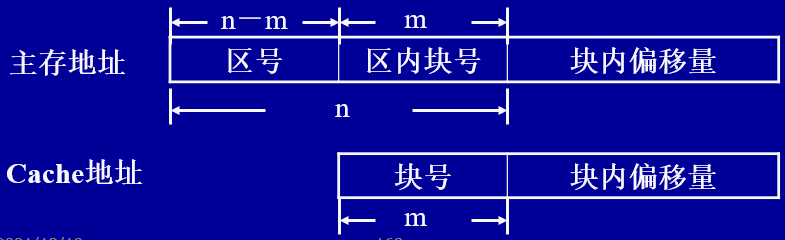
（1）当CPU需要进行访存时，首先给出主存实地址。

（2）地址映像变换机构接收到主存实地址后，根据块号判定所访问的信息字是否在Cache中：若在(Cache命中)，通过地址变换机构变换为Cache块地址，在根据块内地址，对Cache进行存取；若不在(Cache不命中)，则通知访问Cache块失效。然后通过CPU与主存之间的**直接数据通路**访问主存，将被访问字直接送给CPU，并将包含该字的新块装入Cache。若Cache巳满，则通过替换策略实现机构，调出某一Cache块，然后装入所需的块。

**5.2.2 地址映像与变换**

**1.直接映像方式：**每个主存地址映像到Cache中的一个固定地址

**（1）地址变换方式：**

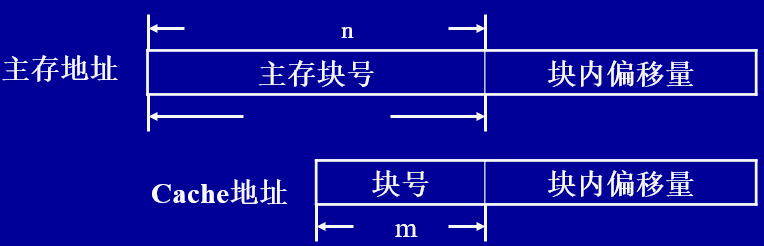


**标识Cache容量：**2m×(n-m+1)（即块数×(标识tag位数+有效位位数)）

**（2）特点：**硬件线路简单；地址变换快；一个主存块只对应于一个Cache块，没有替换策略问题；块的冲突率高；Cache利用率低

**2.全相联映像方式：**任何主存块可映像到任意一个Cache块。

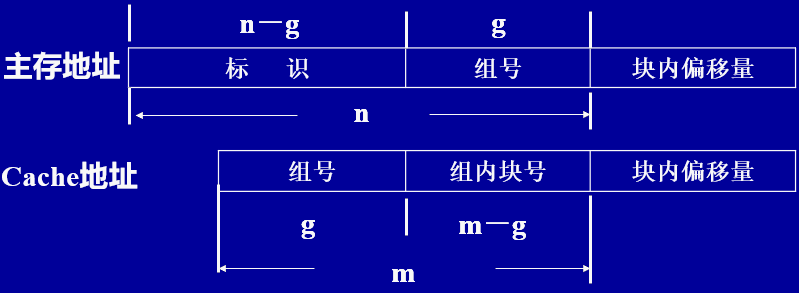
**（1）地址变换方式：**



**（2）特点：**块冲突概率小，Cache命中率高全相联方法只有在Cache中的块全部装满后才会出现块冲突；Cache利用率高；硬件复杂，成本高；相联访问影响访问速度。

**3.组相联映像方式：**将主存块和Cache块按固定大小分组，各组之间采用直接映像，组内各块之间采用全相联映像。

**（1）地址变换方式：**



如果组的大小为1时就变成了**直接映像**；如果组的大小为整个Cache的尺寸时就变成了**全相联映像**。如果一个**组里有k块**，则这种组相联映像方式称为**k路组相联**。

**5.2.3 替换算法及主存更新策略**

**1.替换策略及实现：**

**（1）直接映像：**一个主存块只对应一个Cache块，块的替换很简单，不需要替换算法。

**（2）全相联和组相联映像：**因为一个主存块可选择多个Cache块，所以需要采用替换算法，以便尽可能避免替换掉马上就要用到的信息。

**2.常用替换算法：**

**（1）随机法：**随机地选择被替换的块。简单、易于用硬件实现，但有没有考虑Cache块过去被使用的情况，反映不了程序的局部性，所以易导致命中率降低。

**（2）FIFO：**将Cache中最早调入的块作为被替换的块。容易实现。但因为最先进入的块，很可能是经常要用到的块。所以不能正确地反映程序的局部性。

**（3）LRU：**选择近期最少被访问的块作为被替换的块。LRU能较好地反映程序的局部性，因而其失效率较低。LRU比较复杂，硬件实现比较困难。

**5.2.4 写策略（更新算法）**

**1.写操作的问题：**

**（1）“写”在所有访存操作中所占的比例：**load 26%、store 9%；“写”在所有访存操作中所占的比例：9％/(100％＋26％＋9％)≈7％（其中100％指的是读指令）；“写”在访问数据Cache操作中所占的比例：9％/(26％＋9％)≈25％

（2）“写”操作必须在确认是命中后才可进行

（3）“写”访问有可能导致Cache和主存内容的不一致（直接写Cache或直接写主存）

**2. 解决主存与Cache内容不一致的方法**

**（1）写直达法：**当CPU执行写操作时，利用主存与CPU之间的直接通道，在写Cache的同时也将数据写入主存。也称为全写法。

**（2）写回法（Write-Back WB法）：**当CPU执行写操作时，只写Cache不写主存，只有当被写Cache数据块要被替换出去时才将已修改过的Cache块写回主存。（需增加dirty位）

**（3）写回法vs写直达法：可靠性方面**，写直达法优于写回法；**与主存的通信量**，写回法少于写直达法；**控制的复杂性**, 写直达法比写回法简单；**硬件实现的代价**, 写回法比写直达法低；**写速度方面**，写回法比写直达法速度快

**（4）总结：**写回法速度快，所使用的存储器带宽较低；写直达法易于实现，一致性好。

**5.2.5 数据Cache、指令Cache及一体化Cache**

**1.分离Cache和混合Cache失效率比较：**因为指令的访问局部性特点更加明显，所以对相同的容量，指令Cache的命中率比数据Cache的高。实践证明，将一体化的Cache分离成指令Cache和数据Cache有助于提高Cache的命中率。（指令Cache 75%；数据Cache 25%）

**2. 访问数据和指令时的不同特征：**（1）对数据的访问有读操作和写操作，对指令访问仅是读操作。所以数据Cache需要进行写操作，而指令Cache不需要进行写操作。（2）读入指令Cache的指令在块替换出去时不需将Cache中的指令写回主存，而读入数据Cache的数据，在进行块替换时，通常需要写回主存。

**5.2.6 Cache性能分析**

**ppt 57-71**

**平均访存时间=**指令所占的百分比×（指令命中时间＋指令失效率×失效开销）＋ 数据所占的百分比×（数据命中时间＋数据失效率×失效开销）

**CPU时间**＝IC×[CPI＋访存次数/指令数×失效率×失效开销]×时钟周期时间（其中“访存次数/指令数”为**每条指令的平均访存次数**；“访存次数/指令数×失效率”为**每条指令的平均失效次数**）

**5.2.7 改进Cache的性能**

根据平均访存时间公式，从三个方面考虑：降低失效率、减少失效开销、减少命中时间

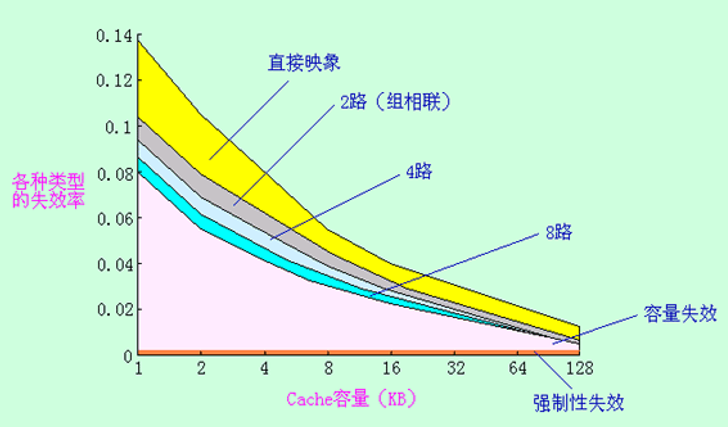
**5.3 降低Cache失效率基本方法**

**1.三种Cache失效：**

**（1）强制性失效：**第一次访问某个块时，一定不在Cache中，需从下一级调入

**（2）容量失效：**如果程序执行时所需的块不能全部调入Cache中，则当某些块被替换后，若又重新被访问，就会发生失效

**（3）冲突失效：**在组相联或直接映像Cache中，若太多的块映像到同一组(块)中，则会出现该组中某个块被别的块替换，然后又被重新访问的情况。



**2.从上图可以看出：**相联度越高，冲突失效就越少；强制性失效和容量失效不受相联度的影响；强制性失效不受Cache容量的影响，但容量失效却随着容量的增加而减少

**5.3.1增加Cache块大小**（减少强制性失效（还有预取算法））

**1.失效率与Cache块大小关系：**（1）对于给定Cache容量，当块大小增加时，失效率开始下降，后来又上升了（减少了强制性失效；块增大，块数目较少，可能会增加冲突失效）；（2）Cache容量越大，失效率达到最低的块大小就越大；（3）增大块大小，会**增加失效开销**

**2. Cache块的大小对Cache命中率的影响：**当块很小时，因为访存的空间局部性未充分利用，命中率不高；随着块的增大，因为块越大调入的相邻数据越多，空间局部性得到利用，命中率越来越高。但块增大后，块的数量就要减少，异块访问的失效率增加，数据时间局部性原理就不能充分利用，所以当块增大到一个最佳点后，命中率反而会随着块的增大而减少。

**5.3.2提高相联度**（减少冲突失效）

**1.**从实际应用角度看，采用**相联度超过8的方案的实际意义不大**。因为8路组相联映像在降低失效率方面的作用已与全相联一样有效。

**2. 2:1Cache经验规则：**大小为N的直接映像Cache的失效率约等于大小为N/2的2路组相联Cache的失效率。

**3.**提高相联度是以**增加命中时间**为代价的。

**5.3.3提高容量**（减少容量失效）

降低失效率的最直接方法，但会增大成本，且**可能会增加命中时间（一般不考虑）**

**~~5.3.4 Victim Cache~~**~~（减少冲突失效）~~

**~~1.基本概念：~~**~~也称“牺牲”Cache，是一种减少冲突失效次数而不影响时钟频率的方法。~~

**~~2.基本思想：~~**~~在Cache和它从下一级存储器调数据的通路之间设置一个全相联的小Cache，用于存放被替换出去的块(称为Victim)，以备重用。~~

**~~3.工作过程：~~**~~将因冲突而被替换出去的块（即Victim）暂存到Victim Cache中。每当发生失效时，在访问下一级存储器之前，先检查Victim Cache中是否含有所需的块。若有，就将该块与Cache中的某块作交换。~~

**~~4.作用：~~**~~有效地减小冲突失效，特别是小容量的直接映像数据Cache，作用尤其明显。~~

**第六章 输入输出系统**

**6.1 I/O系统性能与CPU性能**

**1.I/O系统包括：**I/O设备、I/O设备与处理机的连接

**2.I/O系统是计算机系统中的一个重要组成部分：**完成计算机与外界的信息交换；给计算机提供大容量的外部存储器

3.I/O系统的性能如果不能与CPU的性能相匹配，就会成为整个系统的性能瓶颈。

**4.I/O系统按要完成工作可分为：**存储I/O系统；通信I/O系统

**5.评价I/O系统主要参数：**连接特性；I/O系统容量；响应时间和吞吐率

**6.2 I/O系统的可靠性、可用性和可信性**

**6.2.1系统的可靠性**

**1.概念：**系统从某个初始参考点开始一直连续提供服务的能力。

**2.衡量指标：**

**（1）平均无故障时间MTTF：**表示系统平均能够正常运行多长时间才会出现一次故障，也称平均失效前时间

**（2）平均修复时间MTTR：**表示由于故障和失效所需的平均维护时间，即系统中断服务的时间

**（3）平均失效间隔时间MTBF：**MTBF=MTTF+MTTR（一般MTTF>>MTTR，所以**MTBF=MTTF**）

**（4）失效率：**1/MTBF

**6.2.2系统的可用性**

**1.概念：**系统正常工作的时间在连续两次正常服务间隔时间中所占的比率。

**2.衡量指标：**可用性=MTTF/MTBF

**6.2.3系统的可信性**

**1.概念：**服务的质量。即在多大程度上可以合理地认为服务是可靠的。（不可度量）

**6.3 通道处理机**

**6.3.1 通道的基本概念**

**1.通道处理器：**可以看作是一台能够执行有限I/O指令，并且能够被多台外围设备共享的小型DMA专用处理机。专门负责整个计算机系统的输入/输出工作。

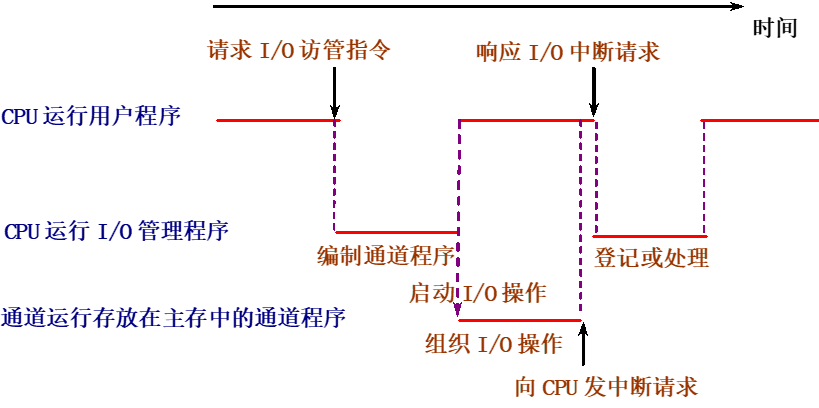
**2.通道特点：**（1）通道有自己的指令系统，能够独立执行用通道命令编写的输入输出控制程序；（2）通道可根据需要控制多种不同的设备；（3）每个通道可以连接多个外部设备，每个外设对应一个子通道；（4）通道通过数据通道与设备的控制器进行通信。

**3.设备控制器：**用于接收通道控制器的命令并向设备发出控制命令。一个设备控制器可控制多个同类的设备，只要这些设备是轮流工作的。

**4.带有通道I/O系统：**主机——通道——设备控制器——设备 四级连接方式。

**5.通道功能：**（1）接收CPU发来的I/O指令，并根据指令要求选择指定的外设与通道相连接；（2）执行通道程序；（3）给出外设中要进行读/写操作的数据所在的地址；（4）给出主存缓冲区的首地址；（5）控制外设与主存缓冲区之间的数据传送的长度；（6）指定传送工作结束时要进行的操作；（7）检查外设的工作状态是否正常，并将该状态信息送往主存指定单元保存；（8）在数据传输过程中完成必要的格式变换

**6.通道工作过程：**



**6.3.2通道的种类**

**1.字节多路通道：**

**（1）概念：**一种简单的共享通道，用于连接多台**低速或中速的设备**，可以依靠通道与CPU之间的高速数据通路分时地为多台设备服务。字节交叉模式工作。

**（2）字节交叉模式：**连接在通道上各个设备轮流占用一个很短的时间片传输一个字节。

**2.选择通道：**

**（1）概念：**只有一套完整的硬件，以独占的方式工作，逐个轮流地为物理上连接的几台**高速外设**服务。

**（2）工作方式：**一段时间内单独为一台外设服务，但在不同的时间内可以选择不同的设备。一旦选中某一设备，通道就进入“忙”状态直到该设备的数据传输工作全部结束为止。其传送的**数据宽度是可变**的，它为一台外设传送完数据后才转去处理其他外设。

**3.数组多路通道**

**（1）概念：**将字节多路通道和选择通道的特性结合起来。一个通道可带有多个子通道，各子通道以成组交叉模式轮流使用通道。其同样适用于**以数组为单位的高速外设**。

**（2）成组交叉模式：**利用通道传送完一组数据（数据块）后让出通道。

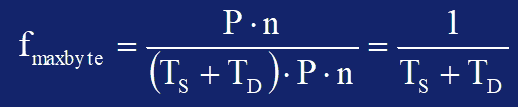
**（3）工作方式：**选择一个高速设备后，先向其发出一个寻找的命令，然后在这个设备寻找期间可以为其他设备服务。在设备寻找完成后再与其真正建立数据连接，并一直维持到一个数据块传输完毕。其传送的**数据宽度是定长**的。

**6.3.3 通道中的数据传送过程与流量分析**

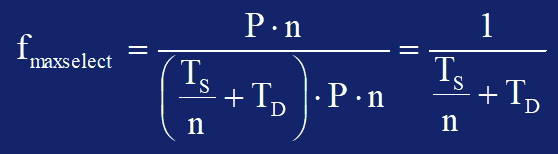
**通道流量（通道吞吐率、通道数据传输率）：**通道在数据传送期间单位时间所传送的数据量。通常用单位时间传送的字节数表示。

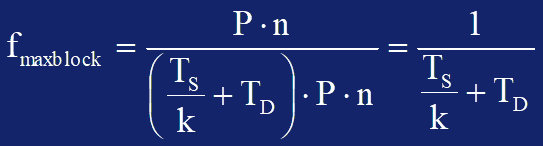
**通道极限流量（通道最大流量）：**通道在数据传送期间单位时间所允许传送的最大数据量。通常用单位时间传送的最大字节数表示。

**1.通道极限流量**

**（1）字节多路通道：**

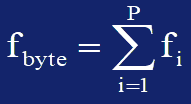
（其中TS为设备选择时间，TD为传送一个字节所需时间，P为通道上连接设备台数，n为每一个设备传送字节数）

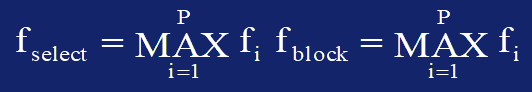
**（2）选择通道：**

**（3）数组多路通道：**

（其中k为一个数据块中包含字节数）

**2.通道实际流量**

**（1）字节多路通道：**

**（2）选择通道和数组多路通道：**

**3.通道流量设计要求：**实际流量<=极限流量

**6.3.4 I/O系统流量分析**

**略，ppt 43-57**

**1.基本设计要求：**（1）I/O系统的流量为系统中各通道的流量之和；（2）高速设备优先响应，同速设备由系统指定或依次响应；（3）I/O系统的流量表示了I/O系统对主存带宽的要求。一般要求主存带宽是I/O系统的流量的两倍以上。

**2.在实际设计最大通道流量时，应留有一定的余量：**如果一个字节多路通道的最大流量正好等于连接在这个通道上的所有设备的流量之和，当所有设备的数据传送请求集中出现时，有可能要丢失数据。

**第七章 并行处理、互连网络及多处理机**

**7.1 互连网络的基本概念**

**7.1.1 互联网络的功能和特性**

**1.基本概念：**互连网络是一种由开关元件按照一定的拓扑结构和控制方式构成的网络，用来实现计算机系统中结点之间的相互连接。而在拓扑上，互连网络是从输入结点到输出结点之间的一组互连或映象。

**2.结点：**可以是处理器、存储模块或者其他设备

**7.1.2 互连网络的特性参数**

**1.网络规模：**网络中的结点数。用于表示网络中所能连接的部件的多少。

**2.连接度（结点度）：**网络中与结点相连接的边数目。用于表示结点所需要I/O端口数。

**3.连接数（链路数）：**网络中所有结点之间连接的数量，即通路（边数）的总和。

**4.结点距离：**网络中任意两结点之间相连的最少边数。

**5.网络直径：**网络中任意两结点之间距离的最大值。

**6.等分宽度：**当某一网络被切成相等的两半时，沿切口的最小边数（通道），用b表示。

**7.对称性：**若从任何结点看，网络的拓扑结构都是一样的话，则称此网络为对称网络。对称网络较易实现，编程也比较容易。

**8.路由：**在网络通信中对路径的选择与指定。

**7.2 互联网络的结构**

**~~7.2.1 静态互联网络~~**

**~~1.线性阵列：~~**

**~~（1）概念：~~**~~一维网络，连接最简单拓扑结构，其中N个结点用N－1条链路连成一行。~~

**~~（2）网络参数：内部结点度~~**~~为2，~~**~~端结点度~~**~~为1。~~**~~网络直径~~**~~为N－1。~~**~~等分宽度~~**~~为1。~~

**~~2.环和带弦环~~**~~：~~

**~~（1）环型网络的概念：~~**~~用一条附加链路将线性阵列的两个端结点连接在一起，就构成了环型网络，简称环。（~~**~~单向环：~~**~~只有一个方向；~~**~~双向环：~~**~~两个方向，一个方向故障另一个方向还可工作）~~

**~~（2）单向环参数：~~**~~结点度为2。网络直径为N-1。等分宽度为2。~~

**~~（3）双向环参数：~~**~~结点度为2。网络直径为N/2向下取整。等分宽度为2。~~

**~~（4）带弦环：~~**~~将环的结点度提高，就得到带弦环~~

**~~（5）全连接网：~~**~~将带弦环度数提高至N-1（网络参数：~~**~~结点度~~**~~为N-1,~~**~~网络直径~~**~~为1，~~**~~等分宽度~~**~~为(N/2)~~~~2~~~~）~~

**~~（6）循环移数网络：~~**~~将环上每个结点到与其~~**~~距离为2整数幂~~**~~的结点之间都增加一条附加链就构成了循环移数网络。（网络参数：N=2~~~~n~~~~则：结点度为2n-1，网络直径为n/2）~~

**~~3.树形和星形网络~~**

**~~（1）树形网络概念：~~**~~顶部一个结点称为根，底部结点称为叶子，其它的称为中间结点~~

**~~（2）k层完全平衡二叉树参数：~~**~~最大结点度3。结点数N＝2~~~~k~~~~－1。网络直径2(k－1)。~~

**~~（3）星形网络概念：~~**~~2层树，结点度很高。星形结构多用于有集中监督结点的系统中。~~

**~~（4）星形网络参数：~~**~~结点度为N-1。网络直径为2。等分宽度为[N/2]。~~

**~~（5）胖形树：~~**~~胖树的通道宽度从叶结点往根结点上行方向逐渐增宽，它更象真实的树，其向树根方向的枝叉变得愈来愈粗。~~**~~传统二叉树的主要问题之一：~~**~~根部的通信最忙，成为通向根结点的瓶颈问题。而胖树的提出使根结点的瓶颈问题得到了缓解。~~

**~~4.网格形和环网形~~**

**~~（1）网格形网络参数：~~**~~k维网格网，结点数为 N＝n~~~~k~~~~。内部结点度为2k，边结点和角结点的结点度为3或2。网格直径为 k(n－1)。~~

**~~（2）环形网：~~**~~环形网沿阵列每行和每列都有环形连接,可以看做是一种直径更短的网格。环形网的拓扑结构将环形和网格组合在一起，并能向高维扩展。（普通2D环网和ILLIAC网）~~

**~~（3）2D环网参数：~~**~~结点数为N=n\*n。结点度为4。网络直径为2\*[n/2]。等分宽度为2n~~

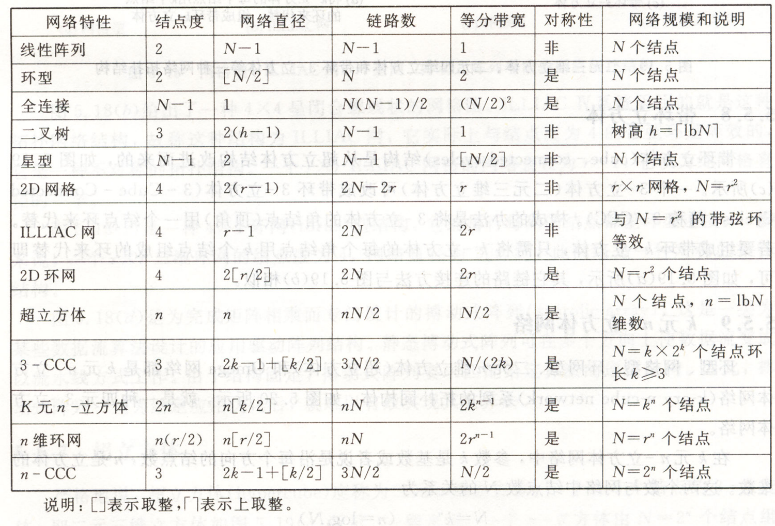
**~~（4）ILLIAC网参数：~~**~~结点数为N=n\*n。结点度为4。网络直径为n-1。等分宽度为2n~~

**~~5.超立方体~~**

**~~（1）概念：~~**~~超立方体是一种二元n维立方体（n－立方体）结构。通常一个n维立方体由 N＝2~~~~n~~ ~~个结点组成，结点分布在n维上，每维有两个结点。其实一个不可扩展网络。~~

**~~（2）超立方体网络参数：~~**~~结点数为N=2~~~~n~~~~。结点度为n。网络直径为n。等分宽度为N/2~~

~~6.静态网络特性一览表~~

~~~~

**~~7.常见静态互连网络的特性总结~~**

~~（1）结点度越大，网络的连接性越好，但网络的连接越复杂。因此在能实现所有结点连接的条件下，结点度越小越好，网络时延越小越好。~~

~~（2）等分带宽越大，网络的带宽越大。~~

~~（3）网络直径越大，通信的时间延迟越大~~

~~（4）对称性会影响可扩展性和寻径效率。~~

~~（5）网络的总价格随结点度和链路数增大而上升。~~

~~（6）网络直径小是网络的一种优点。但有时结点间平均距离可能是一种更好度量指标。~~

~~（7）等分带宽可以用较宽的通道宽度来扩大。~~

**7.2.2 互连函数和互连网络**

（互联函数表示方法：四种方法。略，ppt 34-39）

**1.交互互联网络（Exchange）：**

**（1）函数：**在网络规模为N的互连网络中，交换互连网络的互连函数称为**交换函数**： 其中0≤k≤n－1,n＝log2N。即把输入端 x的二进制编码的**第k位变反**就可得到对应的输出排列。

**（2）特性：**设**网络结点数为N**，则交换互连网络中共有**n＝log2N种互连函数**。在单级交换互连网络中，从一个结点到任何另一个结点所需的**最短距离为n**。

**（3）立方体互连网络（Cube）：**当N＝8，n＝log28＝3时，交换互连网络就是常用的立方体互连网络（Cube）。Cube0是x0取反，Cube1是x1取反，Cube2是x2取反。



**2.均匀混洗互连网络（Shuffle）**

**（1）函数：** 即：将输入端x的二进制编码**循环左移一次**就可得到输出端的二进制编码

**（2）逆均匀洗牌（逆全混洗）函数：** 即：将输入端x的二进制编码**循环右移一次**就可得到输出端的二进制编码



**（3）特点：**编号为全“0”和全“1”的输入端PE无法与网络中的其他PE相连。

**（4）应用：**均匀洗牌函数在实现多项式求值、矩阵转换和FFT等并行运算及并行排序方面都得到广泛的应用。

**3.蝶式互连网络（Butterfly）**

**（1）函数：** 即:将输入端x的二进制编码的**最高位与最低位互相交换位置**就可得到输出端的二进制编码。

**4.反位序互连网络(Bit Reversal permutation)**

**（1）函数：**即:将输入x的**二进制编码的位序颠倒**就可得到输出端的二进制编码。

**（2）与蝶式互连网络的关系：**当N＝8时，B(x)函数等于R(x)函数。



**5. PM2I互连网络**

**（1）函数：**PM2I互连网络是“加减2i”单级网络的简称，又称循环移数网。它的互连函数为移数互连网络的特例：PM2±i(j)＝(j±2i)mod N（式中：0≤i≤n－1，0≤j≤N－1，n＝log2N，N为网络中的结点数）PM2I互连网络中共有2n个互连函数。（或者说2n-1个）

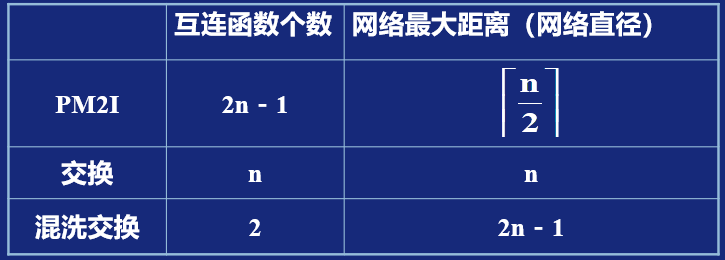
**（2）特性：**在通用情况下PM2＋(n－1)＝PM2－(n－1)。在N相同的情况下， PM2I网络比立方体网络具有更好的连接性。

**6.混洗交互网络**

**（1）函数：**混洗交换互连网络由均匀洗牌函数S(x)和交换函数E0(x)复合而成。（**实际上是或者混洗或者交换**）。PSE0=Exchange0[shuffle(xn-1…x0)]=Exchange0[xn-2…x0xn-1]

**（2）特性：**最远的全“0”和全“1”两个PE间的连接需要n次交换和n－1次混洗才能实现，所以PSE(0)的最大距离为2n－1。

**（3）三种基本网络的比较：**



**7.3 动态互连网络**

**概念：**动态互连网络能够根据需要借助控制信号对连接通路加以重新组合，实现所要求的通信模式。

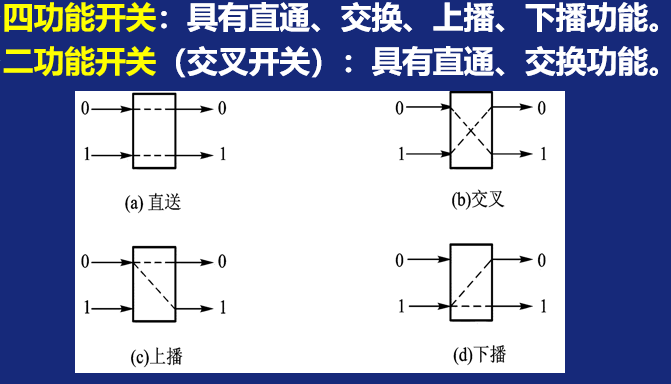
**7.3.1多级互连网络 MIN**

**1.概念：**MIMD和SIMD计算机都使用多级互连网络MIN。

2.单级互连网络通常只能实现有限几种基本连接，不能实现任意处理单元之间连接。为实现任意处理器间的互连，可采用的方法：（1）**利用时间重复性**，将同一套单级互连网络循环使用。（节省器材，但控制困难且速度慢）（2）**利用空间重复性**，将多套单级互连网络串接使用。（速度快，更加灵活，但成本增加）

**3.决定多级互联网络特性的因素：**

**（1）交换开关：**交换开关是组成互连网络的基本单元。一个开关模块一般是n个输入、n个输出，且n=2k。输入与输出**只允许一对多和一对一映射**，不允许多对一映射。只允许一对一映射的开关称为**交叉开关**。

**（2）拓扑结构：**拓扑结构就是**级间连接模式ISC**。用于表示网络间各级输入端与输出端相互连接的规则或连接模式。将各种单级互连网络进行不同组合，可构成各种具有不同连接特性的多级互连网络。

**（3）控制方式：1）级控：**同一级的所有交换开关都受同一控制信号控制，对一个n级网络就需要n个控制信号；**2）单元控制：**每一个交换开关都有一个单独的控制信号，各自处于不同的状态；**3）部分级控（组控制）：**对不同的级采用不同数量的控制信号。

**4.阻塞网络与非阻塞网络**

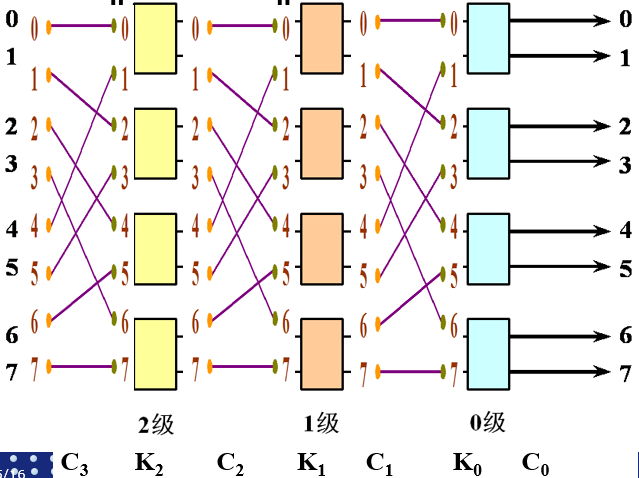
**（1）阻塞网络：**网络中一对以上的输入输出端同时实现互连时，可能会出现路径冲突。即**不能实现输入端和输出端之间的任意连接**。

**（2）非阻塞网络：**无需因增加新的连接而改变原有的连接路径，**消息通信不会阻塞**。即**可以实现输入端和输出端之间的任意连接**。

**7.3.2 多级混洗交换网络（OMEGA网络）**

**1.概念：**一个N输入端OMEGA网络通常由n级相同的网络组成(n＝log2N) 。网络的每一级都包含一个全混洗拓扑和随后一列N/2个**四功能交换开关**，因此共需要(Nlog2N)/2个开关模块，每个开关模块均采用**单元控制**方式。

2.级间连接从输入端到输出端的编号依次是：Cn，Cn-1，……C1，C0。其中Cn ,……C1为全混洗置换，C0为恒等置换。如下图为N=8，n=3时的OMEGA网络



**3.互连网络中的阻塞冲突问题：**一个输入、输出为N的互连网络，其一共有N！排列。若网络中使用二功能开关，则当所有开关处于不同状态的总数最多为2开关数=NN/2,此时网络显然会发生冲突，是阻塞网络。

**7.3.3多级立方体网络（这里以级控制的STARAN网络为主）**

**1.概念：**多级立方体网络由n个单级立方体Cubei网络串连构成，其中0≤i≤n－1。其通常采用**二功能交换开关**构成。

**2.常见的多级立方体网络：STARAN网络**和**间接二进制n方体网络**

**（1）相同点：**1）均采用二功能交换开关；2）当第 k 级（0≤k≤n－1）交换开关处于交换状态时，实现的是E(x)k互连函数。

**（2）不同点：**STARAN网络采用**级控和部分级控**。间接二进制n方体网络采用**单元控制**。

**3.STARAN网络特点：**

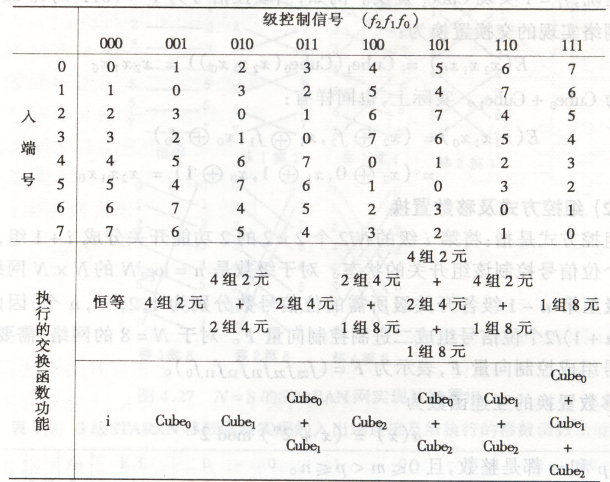
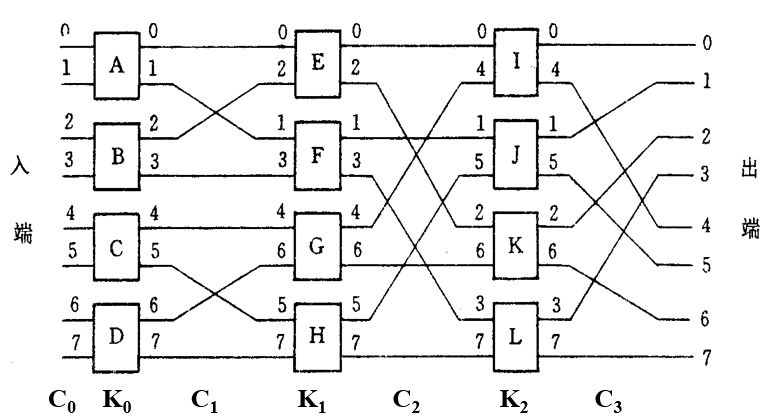
（1）一个N×N的STARAN网络的级数是n＝log2N，每级有N/2个开关，整个网络的开关数为(N/2)log2N。

（2）采用2×2的二功能开关，2个功能为直送和交叉。通常用**“0”控制开关为直送**，**“1”控制开关为交叉**。

（3）开关级的级号编排从网络的输入端到输出端，依次为K0，K1，…，Kn－1。

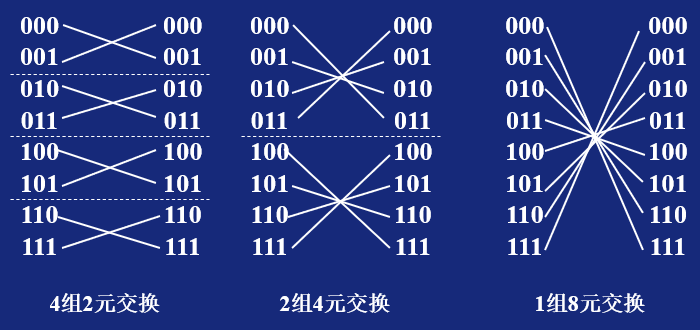
（4）级间连接模式从网络输入端到输出端依次表示为C0，C1，…，Cn。其中C0为恒等置换，C1~ Cn-1为子蝶式置换（分别为B2,…Bn；注：子蝶式Bk表示低k位的最高位和最低位互换），Cn为逆混洗置换。

4.**STARAN网络开关控制方式（级控为例且N=8）：**



**5.交换功能：**

4组2元=Cube0；2组4元=Cube0+Cube1；1组8元=Cube0+Cube1+Cube2



**7.3.4 OMEGA网络与多级立方体网络**

**1.两者之间联系：**

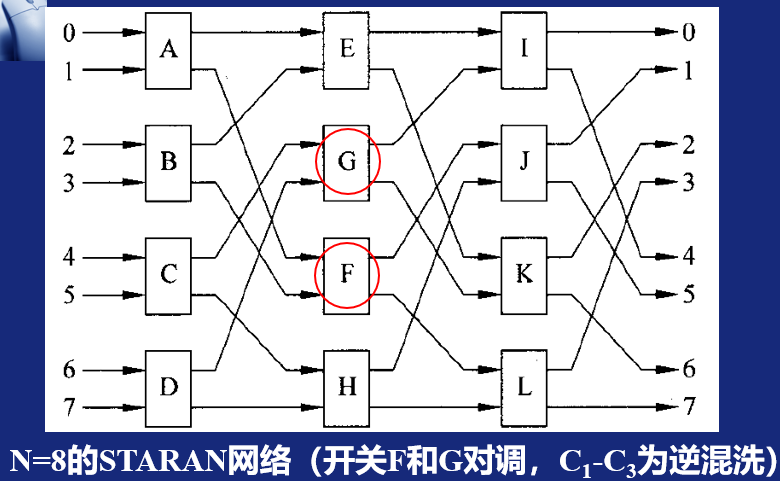
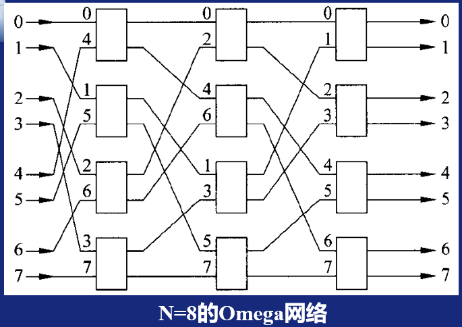
（1）n级OMEGA网络的互连函数可以表示成混洗交换函数：PSE(n)＝(ε(0)σ(n))n

（2）在多级立方体网络中，各级的编号顺序与OMEGA网络正好相反。

（3）当采用**级控方式**时，OMEGA网络成为STARAN网络的逆网络。

（4）当采用**单元控制方式**时，OMEGA网络成为间接二进制n方体网络的逆网络。

**2. N=8的多级立方体网络和Omega网络的关系**

（1）数据流向相反。

（2）OMEGA网络采用四功能交换开关，多级立方体网络采用二功能交换单元。

（3）由于OMEGA网络采用四功能交换开关，因此允许同时实现一个入端到多个出端的连接，具有更多的灵活性。