

第五章 并行处理机





内容


- 1 并行性概念
- 2 并行处理机
- 3 互联网络



1 并行性概念

- 1.1 并行性定义
- 1.2 提高并行性的三条技术途径
- 1.3 并行处理的发展

1 并行性概念

- 计算机系统设计的基本任务之一就是加快指令的解释过程
- 并行的目的：提高机器处理速度
- 并行性  设备的简单重复
- 还有：重叠、先行控制、流水线技术、并发执行多道程序技术

1.1 并行性(Parallelism)

- **定义：** 在同一时刻或同一时间间隔内完成两种或两种以上的性质相同或不同的工作，只要在**时间上相互重叠**，均存在并行性
- **分类：**
 - **同时性(Simultaneity):** 两个或多个事件在**同一时刻**发生的并行性
 - **并发性(Concurrency):** 两个或多个事件在**同一时间间隔**内发生的并行性

并行性例子

- n 位加法器做串行进位并行加法，由于存在进位信号从低位到高位逐位递进的延迟时间，因此 n 位加法器的运算结果并不是在同一时刻获得，故不存在同时性，只存在**并发性**
- m 个存储器模块能同时进行存取信息，则属于**同时性**

并行性等级

- 指令内部并行：指令内部的微操作之间的并行

实现并行性是一个软、硬件功能合理分配的问题

多个过程或任务（程序段）

- 作业或程序级并行：在多个作业或程序间的并行

并行性的等级从数据处理角度

- **字串位串**：同时只对一个字的一位进行处理，不存在并行性
- **字串位并**：同时对一个字的所有位进行处理，
- **字并位串**：同时对多个字的同一位进行处理
- **字并位并**：同时对多个字的所有位或部分位进行同时处理

1.2 提高并行性的三条技术途径

(1) 时间重叠 (time-interleaving)

- 流水线
- 多个处理过程在时间上相互错开，轮流重叠使用同一套硬件的各个部件，以加快部件的周转而提高速度。时间重叠原则上不要求重复的硬件设备

(2) 资源重复 (resource-replication)

- 重复设置多个硬件部件以提高计算机系统的性能
- 随着硬件价格不断下降，从单处理机发展到多处理机，资源重复已经成为提高系统并行性的有效手段

(3) 资源共享 (resource-sharing)

- 分时系统、分布式系统
- 利用软件方法，使多个用户分时使用同一个计算机系统。例如多道程序、分时系统就是资源共享的产物。它是提高计算机系统资源利用率的有效措施



资源重复：侧重空间方面，开发并行性的同时性；

时间重叠：侧重时间方面，在处理时间上相互重叠；

资源共享：侧重软件手段，开发并行性的并发性。

1.3 并行处理的发展

(1) 单机系统中并行处理的发展

- 单处理机并行性开发的主要途径是时间重叠
- 实现时间重叠的基础是部件功能专用化
- 将一件工作按功能分割成若干联系的部分，每一部分有指定的专门部件来完成，然后按时间重叠的原则把各部分执行过程在时间上重叠起来，使所有部件依次分工完成一组同样工作


时间重叠:

- 在处理器内部能同时解释两条指令，从而提高处理器速度，这种方式称为**重叠方式**
- 如果把指令解释过程分解成多个子过程，分别由多个专用部件完成，这就是**先行控制**，可实现同时解释多条指令
- 如把功能专用化深入到处理机的执行部件内部，将该部分再分成多个专用功能段，进行流水处理，从而对数据处理的并行性由字串位并发展到全并行，这就是操作**流水线**


- 可以用一条向量指令对向量的各个元素进行流水处理。这就可以实现从指令间并行性转为指令内的并行性，这就是**向量处理机**
- 把时间重叠原理应用于任务一级，对各任务设置专用处理机，按流水线方式工作，就构成了**宏流水线**，即进入了**多处理机领域**。这种多处理机称为非对称型(asymmetrical)或异构型多处理机(heterogeneous multiprocessor system)

资源重复：

- 在高性能的单机系统中，随着硬件价格下降，资源重复也逐渐普遍起来。资源重复主要是为了提高系统的可靠性，即在关键部件上采用冗余技术
- 同时，在系统结构中采用多操作部件和多存储体，这种资源重复的结构也是为了提高系统的速度

- 
- 通过重复设置多个相同的处理单元，在一个控制器的指挥下，按照同一指令（即一条向量指令）要求，各处理机同时对各向量元素进行操作。这就是并行处理机
 - 并行处理机普遍采用阵列结构形式，故称之为阵列机


- **相联存储器**是一种按内容寻址的、具有信息处理功能的存储器，能按**字并位串**或**全并行**的方式对所有存储单元的内容进行操作。以相联存储器为核心，加上中央处理机、指令存储器和I/O接口，就可以构成以存储器并行操作为特征的**相联处理机**。它是并行处理机思想应用于相联连存储器内部。

- 
- 在任务级并行，则每个处理单元配备自己的控制器，能独立的解释、执行指令而成为一台处理机，这就是**多处理机系统**

它可以称为**对称型**（symmetrical）或**同构型多处理机系统**（homogeneous multiprocessor system）。它们由多个同类型、同功能的处理机构成，能同时处理同一作业中能并行执行的多个任务

资源共享：

- 从多道程序发展到分时系统，其实质是单机模拟多机功能。分时系统适用于多终端情况，对于远程用户，可配接远程终端
- 如果在终端内配上微处理器，使其不仅有I/O功能和通信功能，还具有一定的信息存储、分析、处理能力，这就成为智能终端

- 
- 将若干台具有独立功能的处理机(或计算机)相互连接起来，在操作系统(或分布式操作系统)的控制下，统一协调的运行，这就是**分布式处理系统**(distributed processing system)
 - 分时系统是以“集中”为特征，分布系统是以“分布”为特征；分时系统的并行性是并发性，分布系统的并行性是同时性

(2) 多机系统中并行处理的发展

- 多计算机系统 (multi-computer system)
多台独立的计算机构成的系统
- 多处理机系统 (multiprocessor system)
多台处理机构成的系统

在操作系统和并行性方面两者的区别：

- **多计算机系统：** 每台计算机有自己的操作系统；它们之间通过通道和/或通信线路实现通信，以完整文件或数据集合进行信息传递，以实现作业和任务级的并行
- **多处理机系统：** 有统一的操作系统控制，由于共享存储器，各处理机之间不但能以完整的文件或数据集合进行信息传递，也能以向量或单个数据进行通信

耦合度是反映多机系统中各机器之间物理连接的紧密程度和交互作用能力的强弱

- **最低耦合**：仅通过中间存储介质互相通信，除此之外，各机器间并**无物理连接**，也无共享的联机硬件资源。例如：磁带
- **松散耦合**（间接耦合）：机器之间是通过**通道或通信线路**实现互联，共享某些外围设备。特点：通信频带较低
- **紧密耦合**（直接耦合）：通过总线或高速开关实现互连，共享主存储器，机器间通信频率高，信息传输率和吞吐量

两种典型的松散耦合形式:

- 多台计算机通过通道共享外围设备，各个机器实现功能专用化，机器处理结果以文件和数据集合形式送到共享外设，供其他机器调用，从而获得较高的系统使用效率
- 计算机网络，各节点计算机通过计算机网络连接，在网络操作系统管理下，合理调度软、硬件资源，以求得更大范围内资源共享

多机系统中的并行性

- 多机系统也是沿着时间重叠、资源重复、资源共享三条技术途径发展

(1) 时间重叠：多个专用功能部件

- 将处理功能分解给各个专用处理机完成，实现功能专用化。各处理机之间按照时间重叠原理工作



(2) 资源重复

早期不是为了提高速度，而是为了提高可靠性

(3) 资源共享

网络化

多机系统的比较

- **同构型多处理机**是把一道程序（作业）分解为若干相互独立的程序段或任务，分别有各个处理机**并行执行**。
- **异构型多处理机**是将作业分解成**串行执行**的若干个任务，分别由不同功能的处理机分工完成，依靠流水作业的原理，对多个作业重叠地进行处理。
- **分布处理系统**各处理机尽量完成本地作业，当其资源和能力不够时才与其他处理机协同。

2 并行处理机

- 在单机系统里主要是采用时间重叠技术。把一件工作按功能分割为若干相互联系的部分，把每一部分指定给专门的部件完成，然后按时间重叠原则把各部分执行过程在时间上重叠起来，使所有部件依次分工完成一组同样的工作。
- 并行处理机主要是通过资源重复技术来实现并行处理的。它属于单指令流多数据流（SIMD）计算机一类。

并行处理机的定义

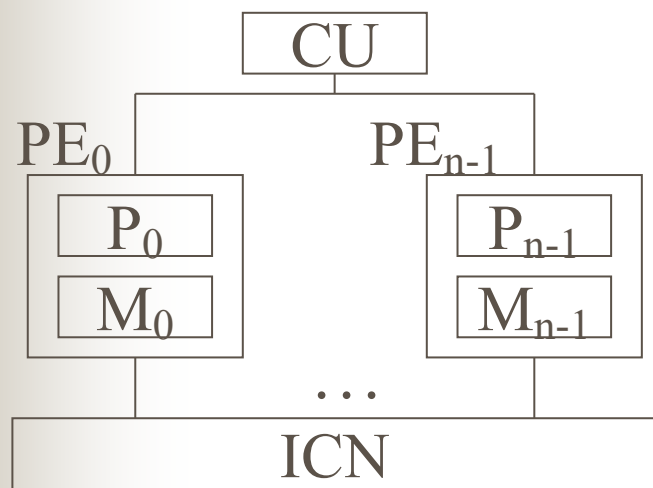
- 多个处理部件按照一定方式互联，在同一个控制部件控制下，对各自的数据完成同一条指令规定的操作
- 从控制部件角度看，指令是串行执行的
- 从处理部件角度看，数据是并行处理的
- 并行处理机也称为阵列处理机，按照Flynn分类法，它属于SIMD处理机

并行处理机工作原理

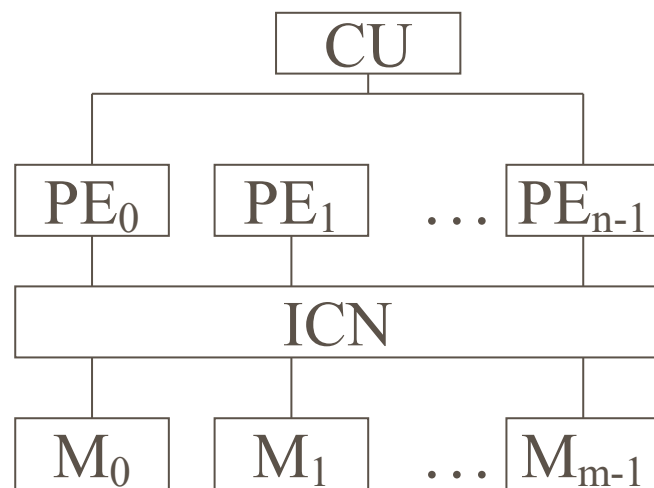
(1) 组成

通常由1个控制器(CU)，多个处理器(PE)， m 个存储模块(M)及1个互连网络(Inter Connection Network ICN)组成。

根据存储模块组成方式可有分布式和集中式两种。



分布



集中式



并行处理机两种结构的共同特点

并行处理机的两种基本结构的共同特点：

- 重复设置许多个**同样的**处理单元PE(Process Element);
- 由ICN按照一定的方式**相互连接**;
- 在**统一的控制**部件CU (Control Unit) 作用下;
- 各PE对分配来的数据并行地完成**同一条**指令所规定的操作。

并行处理的特点

- **资源重复**。利用众多的处理单元对向量所包含的各个分量同时进行运算，获得很高处理速度。
- **连接模式**。它的处理单元间是通过ICN来通信的。不同的连接模式确定了它的不同结构。
- **专用性**。它直接与一定的算法相联系，其效率取决于在多大程度上把计算问题归结为**向量数组**处理。
- **复合性**。整个系统是由三部分复合起来的一个多机系统，即多个处理单元组成**阵列**并行地处理向量；功能极强的控制部件实际上是一台标量处理机；系统的管理功能则由高性能单处理机担负。

阵列机结构

- 阵列机系统是并行处理机最常见的结构形式，它是由大量的处理机按一定规则的几何形式构成阵列形式。
- 最早阵列机是ILLIAC IV，它是由4个处理机阵列构成，每个阵列：64个PE+1个控制部件。

阵列机结构(cont.)

- 阵列机的处理属于SIMD形式（单指令流多数据流），它最适合作向量数组运算。每个处理单元相当于一个向量数组元素的运算，包括定点和浮点的多种运算操作。对于是阵列机处理单元个数的倍数的向量数组运算尤为合适。如PE=64，则16，32，64，128，256，512…阵列向量数组就很方便地使阵列机发挥最佳效能。

阵列机结构(cont.)

- 阵列机中PE之间的互联通信是由互联寄存器来实现的。当PE执行互联指令时，由本PE的互联寄存器与相邻PE互联寄存器进行信息交换。
- 阵列机的操作分公共操作和本地操作。公共操作是指阵列机中的所有PE同时执行的操作，它一般由逻辑控制器来调度。本地操作是每个PE自己的操作。

阵列机算法

■ 举例矩阵问题：

A、B两个矩阵相加，只要把A和B居于相应位置的一对分量存放在同一个处理单元存储器内。当阵列机执行加法公共操作时，每个处理单元都将处于本结点的 A_i 和 B_i 两个矩阵元素进行加法运算，其和即为矩阵和的对应元素。

■ 累加和问题：

书上有详细的举例，请自学。

3 互联网络

- 互连网络基本概念
- 单级互连网络
- 多级互连网络



互联网络是什么

- 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，用于实现计算机系统内部多个处理机或多个功能部件之间的相互连接。

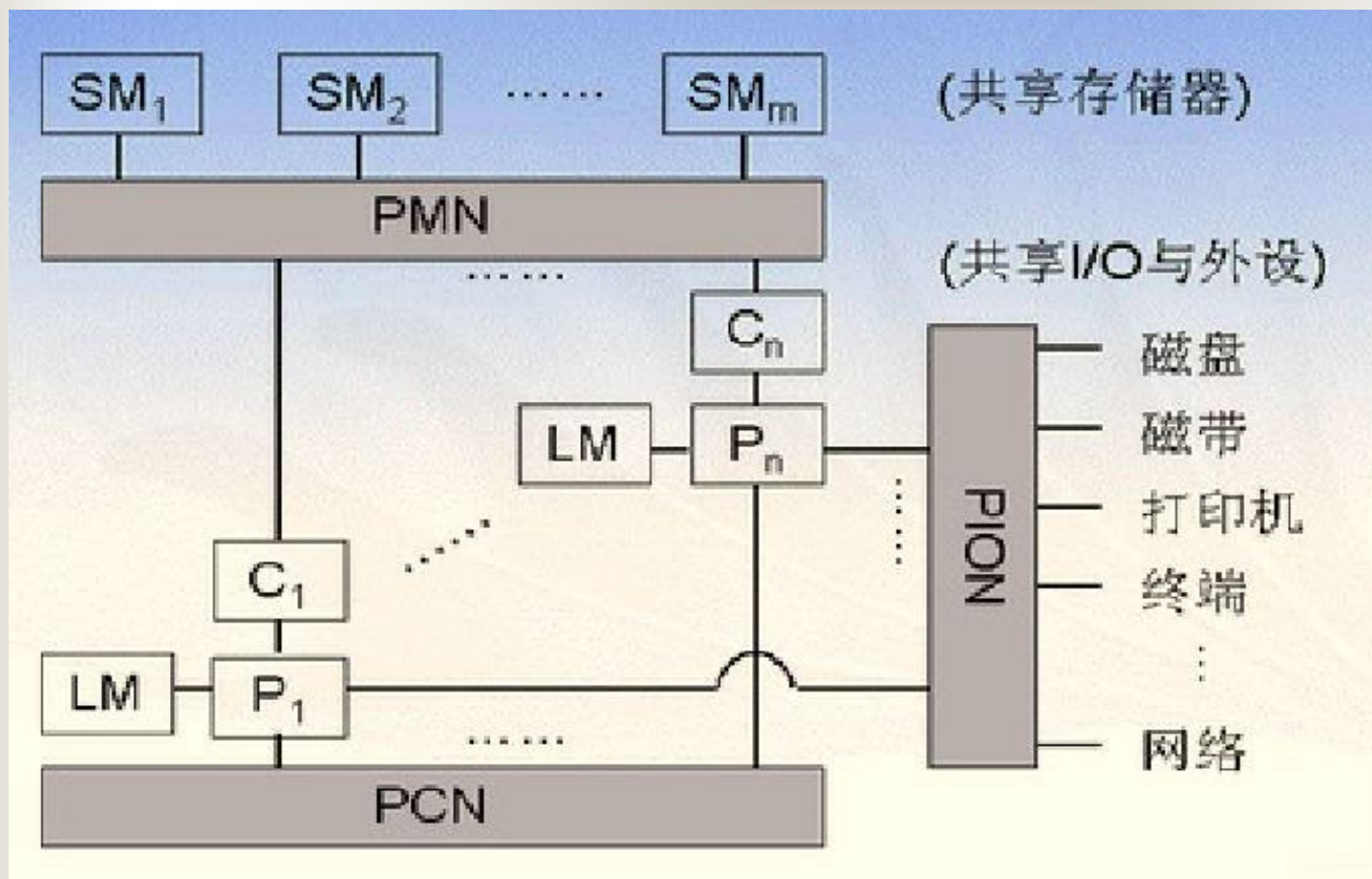


互联网络的结构特征

- | | |
|----------|-----------|
| (1) 通信方式 | 同步、异步 |
| (2) 控制策略 | 集中、分散 |
| (3) 交换方式 | 线路交换、分组交换 |
| (4) 拓扑结构 | |

互连网络的作用

- 用来实现计算机系统内部多个处理机或多个功能部件之间的相互连接
- 互连网络对整个计算机系统的性能价格比有着决定性的影响
- 互连网络由有向边或无向边连接有限个节点的组成
- 一个例子：具有本地存储器、私有高速缓存、共享存储器和共享外围设备的一般处理机系统的互联结构



- PMN: 处理器-存储器网络
- PION: 处理器-I/O网络
- PCN: 处理机之间的通信网络

互连网络的表示方法

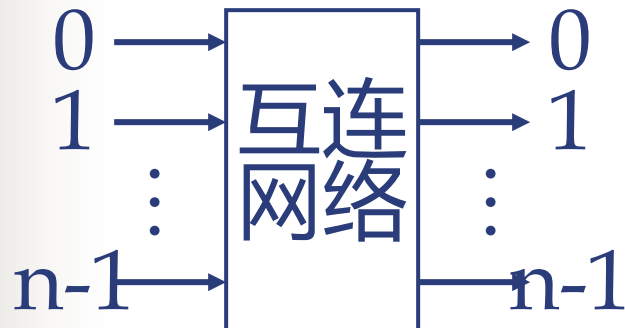
为了在输入结点与输出结点之间建立对应关系，**互连网络有三种表示方法：**

(1)互连函数表示法：

如： $f(x_{n-1} \dots x_1 x_0) = x_0 x_{n-2} \dots x_1 x_{n-1}$

(2)图形表示法

(3)输入输出对应表示法



输入: 0 1 2 3 4 5 6 7
输出: 1 0 3 2 5 4 7 6

单级互连网络

- 交换互连函数(立方体单级网络)
- 全混选单级函数 (Shuffle)
- 蝶式函数 (Butterfly)
- 反位序函数 (Bit Reversal)
- 移数函数(PM2I单级网络)

(1) 交换互联网络（立方体单级网络）

- 函数关系：把二进制结点号的某一位取反

$$E_k(x_{n-1} \cdots x_{k+1} x_k x_{k-1} \cdots x_1 x_0) = x_{n-1} \cdots x_{k+1} \bar{x}_k x_{k-1} \cdots x_1 x_0$$

当 $n=3$ 时，有3种函数，表示8个结点之间的连接关系。

$$E_0(x_2 x_1 x_0) = x_2 x_1 \bar{x}_0$$

$$E_1(x_2 x_1 x_0) = x_2 \bar{x}_1 x_0$$

$$E_2(x_2 x_1 x_0) = \bar{x}_2 x_1 x_0$$

- 由于交换函数主要用于超立方体互连网中，因此也称为超立方体函数，用Cube表示，如：Cube0、Cube1、Cube2等。

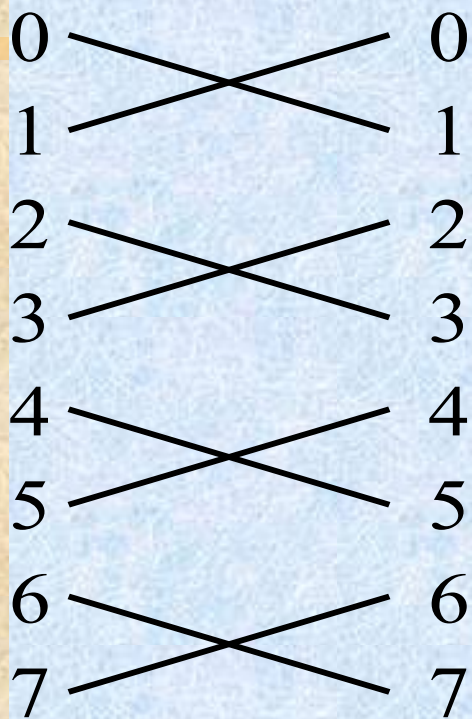
出端编码与连接的入端结点的编码有一位相反。

互连函数:

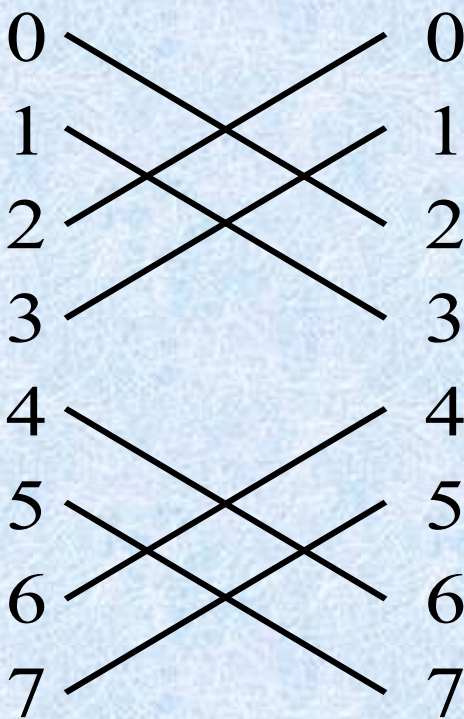
$$\text{Cube}_0 = (b_2 \overline{b_1} b_0) \quad (0, 1) \quad (2, 3) \quad (4, 5) \quad (6, 7)$$

$$\text{Cube}_1 = (\overline{b_2} b_1 b_0) \quad (0, 2) \quad (1, 3) \quad (4, 6) \quad (5, 7)$$

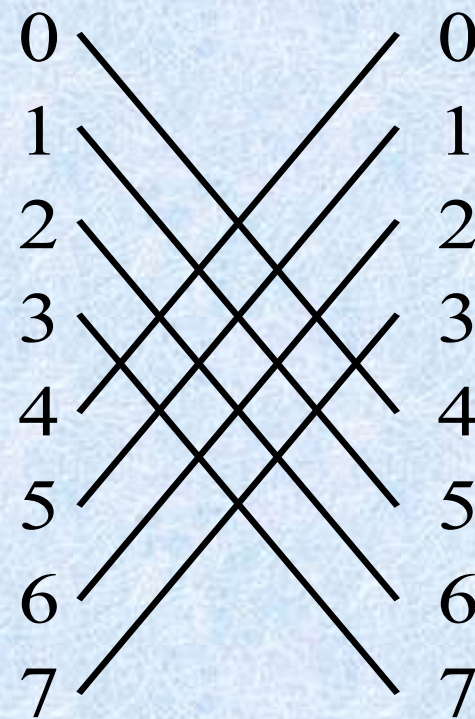
$$\text{Cube}_2 = (\overline{b_2} b_1 b_0) \quad (0, 4) \quad (1, 5) \quad (2, 6) \quad (3, 7)$$



E_0 交换函数

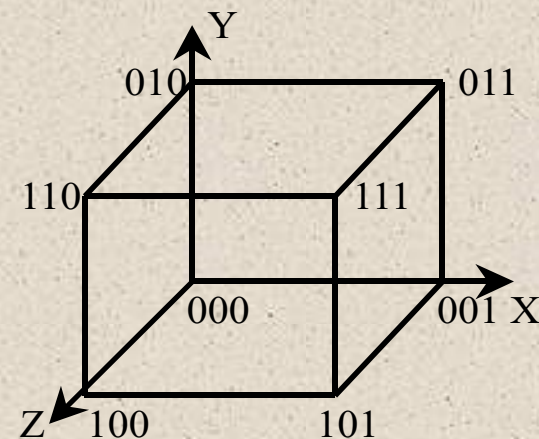


E_1 交换函数



E_2 交换函数

注意：立方体坐标编号不能标错。



立方体网



扩展成超立方体:

有 $n = \log_2 N$ 个互连函数;

$\text{Cube}_i = (b_{n-1} \dots b_i \dots b_0)$;

最大连接度 $= \log_2 N$; 结点最大间距 $= \log_2 N$ 。

应用: 几种互连函数反复调用, 任意结点间可连接。

(2) 全混洗函数 (Perfect shuffle)

函数关系：把二进制结点号循环左移一位

$$S(x_{n-1}x_{n-2}\cdots x_1x_0) = x_{n-2}x_{n-3}\cdots x_1x_0x_{n-1}$$

子混洗(subshuffle) $S_{(k)}$,最低k位循环左移一位

超混洗(supershuffle) $S^{(k)}$,最高k位循环左移一位

$$S_{(k)}(x_{n-1}x_{n-2}\cdots x_kx_{k-1}x_{k-2}\cdots x_1x_0) = x_{n-1}x_{n-2}\cdots x_kx_{k-2}\cdots x_1x_0x_{k-1}$$

$$S^{(k)}(x_{n-1}x_{n-2}\cdots x_{n-k}x_{n-k-1}\cdots x_1x_0) = x_{n-2}\cdots x_{n-k}x_{n-1}x_{n-k-1}\cdots x_1x_0$$

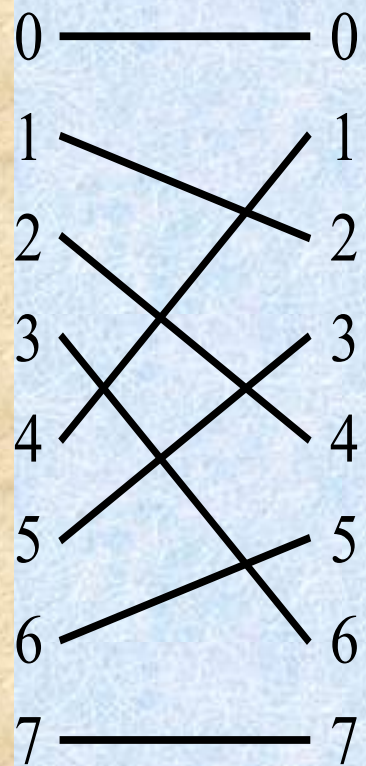
显然成立：

$$S^{(n)}(x) = S_{(n)}(x) = S(x)$$

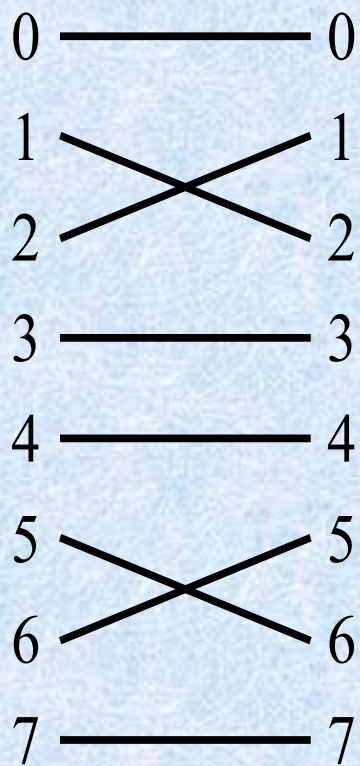
$$S^{(1)}(x) = S_{(1)}(x) = x$$

逆混洗函数：

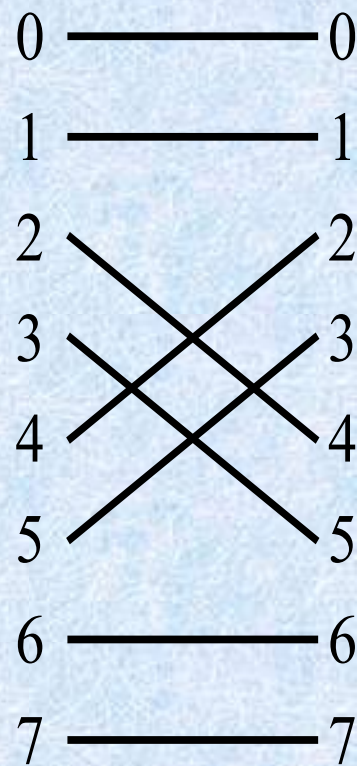
$$S^{-1}(x_{n-1}x_{n-2}\cdots x_1x_0) = x_0x_{n-1}x_{n-2}\cdots x_1$$



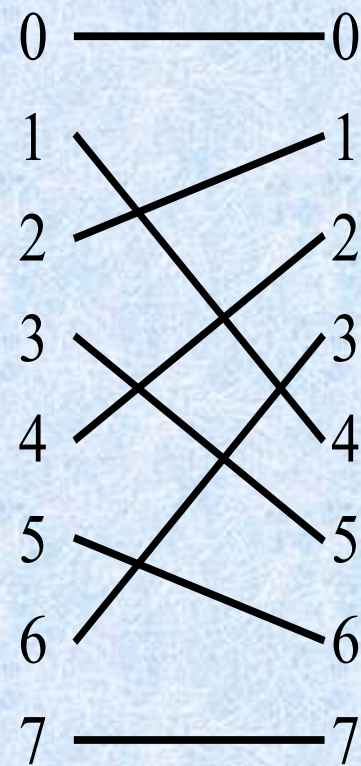
全混洗牌函数



子混洗函数 $S_{(2)}$



超混洗置 $S^{(2)}$



逆混洗置 S^{-1}

(3) 蝶式函数 (Butterfly)

蝶式函数的名称来自于FFT变换时的图形，如蝴蝶式样。函数关系：**将输入端二进制结点的最高位和最低位互换位置。**

$$B(x_{n-1}x_{n-2}\cdots x_1x_0) = x_0x_{n-2}\cdots x_1x_{n-1}$$

子蝶式(subbutterfly) $B_{(k)}$ 最低k位的高低位互换

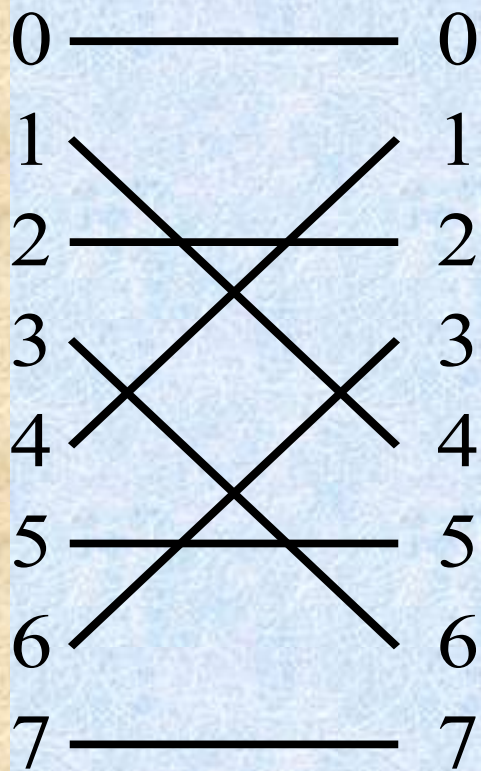
超蝶式(superbutterfly) $B^{(k)}$ 最高k位的高低位互

换 $B_{(k)}(x_{n-1}x_{n-2}\cdots x_kx_{k-1}x_{k-2}\cdots x_1x_0) = x_{n-1}x_{n-2}\cdots x_kx_0x_{k-2}\cdots x_1x_{k-1}$

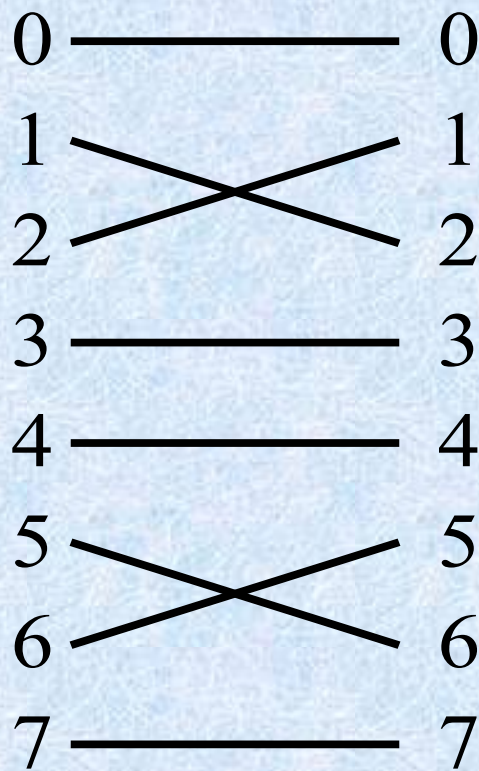
$$B^{(k)}(x_{n-1}x_{n-2}\cdots x_{n-k}\cdots x_1x_0) = x_{n-k}x_{n-2}\cdots x_{n-k+1}x_{n-1}x_{n-k-1}\cdots x_1x_0$$

显然成立： $B^{(n)}(x) = B_{(n)}(x) = B(x)$

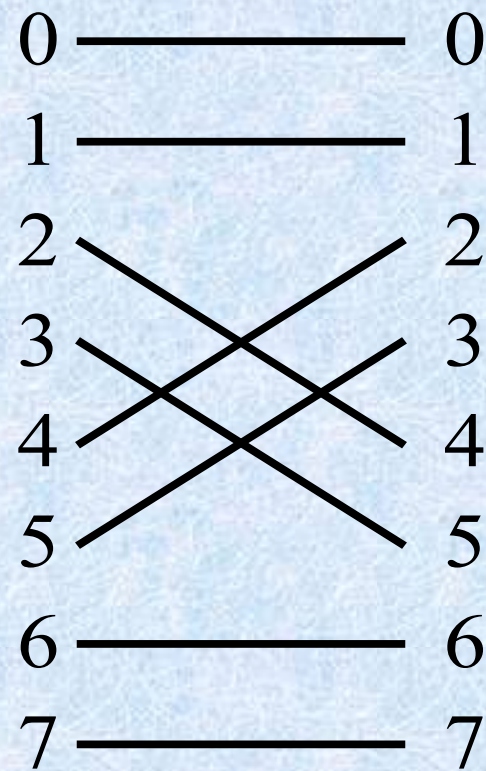
$$B^{(1)}(x) = B_{(1)}(x) = x$$



$B=R$ 函数



$B_{(2)}=R_{(2)}$ 函数



$B^{(2)}=R^{(2)}$ 函数

蝶式函数和反位序函数

(4) 反位序函数 (Bit Reversal)

函数关系：将二进制自变量的位序反过来。

$$R(x_{n-1}x_{n-2}\cdots x_1x_0) = x_0x_1\cdots x_{n-2}x_{n-1}$$

子反位序函数，最低k位的位序反过来

超反位序函数，最高k位的位序反过来

$$R_{(k)}(x_{n-1}x_{n-2}\cdots x_kx_{k-1}\cdots x_1x_0) = x_{n-1}x_{n-2}\cdots x_kx_0x_1\cdots x_{k-1}$$

$$R^{(k)}(x_{n-1}x_{n-2}\cdots x_{n-k}x_{n-k-1}\cdots x_1x_0) = x_{n-k}x_{n-k+1}\cdots x_{n-1}x_{n-k-1}\cdots x_1x_0$$

对于n=3的情况，正好有：

$$R=B, \quad R_{(2)}=B_{(2)}, \quad R^{(2)}=B^{(2)}。$$

(5) 移数函数

函数关系：将输入端向量循环移动一定的位置

$$A(x) = (x + r) \bmod 2^n, \quad 0 \leq x \leq 2^n$$

经常取 $r=2^i$ ，因此移数函数又称为**加减 2^i 函数**、**PM2I函数**等。

子移数函数：

$$\begin{aligned} A_{(k)}(x) &= (x + 2^i) \bmod 2^k \\ A^{(k)}(x) &= (x + 2^{n-k}) \bmod 2^n \end{aligned}$$

其中： $0 \leq x \leq N-1$ ， $0 \leq i, k \leq n-1$ ， $n = \log_2 N$ 。

Illiac函数包含 $PM2_{\pm 0}$ 和 $PM2_{\pm n/2}$ 等4个互连函数

每个接点与它的上下左右4个相邻接点连接

PM2I

互连函数:

$$\text{PM2}_{+i}(j) = (j + 2^i) \bmod N;$$
$$n = \log_2 N, 0 \leq i \leq n-1, 0 \leq j \leq N-1$$

$$\text{PM2}_{-i}(j) = (j - 2^i) \bmod N;$$

共有 $2n$ 个互连函数($2n-1$ 种不同)

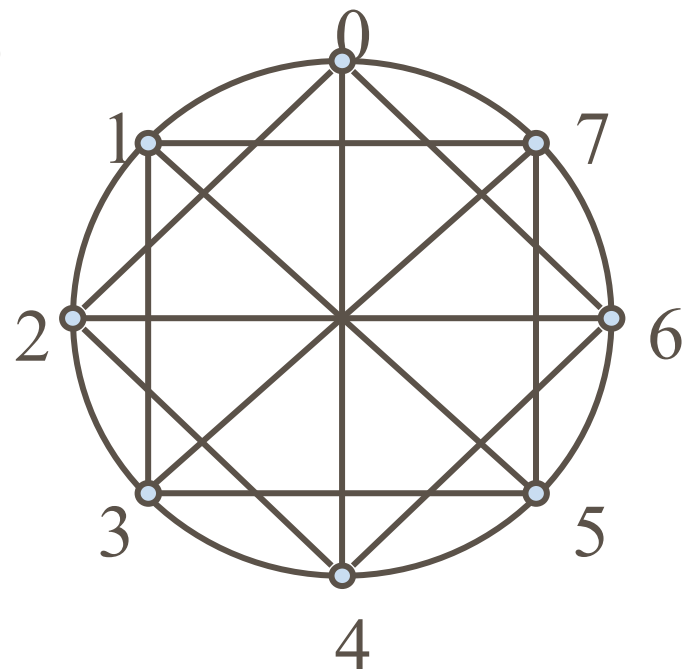
连接图:

± 0 : 顺环圆周连接;

± 1 : 顺环内接 $n/2$ 边形连接;

± 2 : 顺环内接 $n/4$ 边形连接;

$\pm (n-1)$: 顺环内直径连接。



■ N=8时，互连函数为：

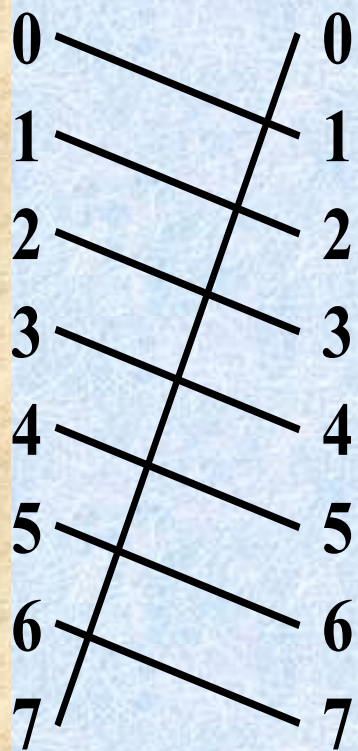
➤ $PM2_{+0}$: (01234567)

➤ $PM2_{-0}$: (76543210)

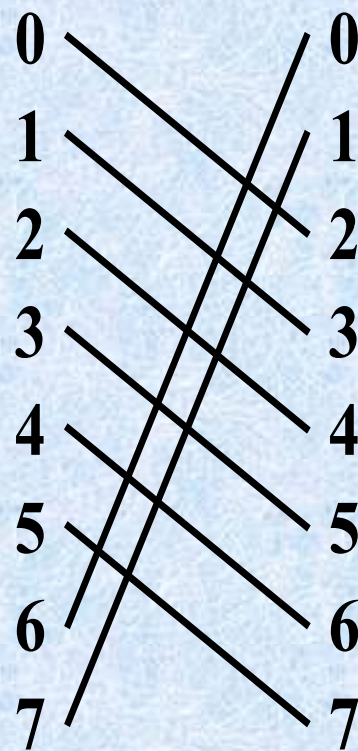
➤ $PM2_{+1}$: (0246) (1357)

➤ $PM2_{-1}$: (6420) (7531)

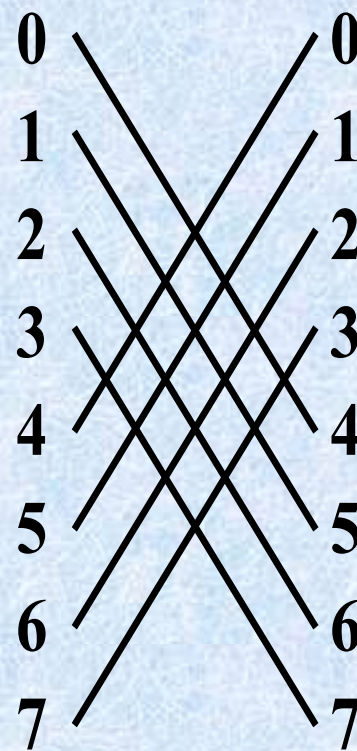
➤ $PM2_{\pm 2}$: (04) (15) (26) (37)



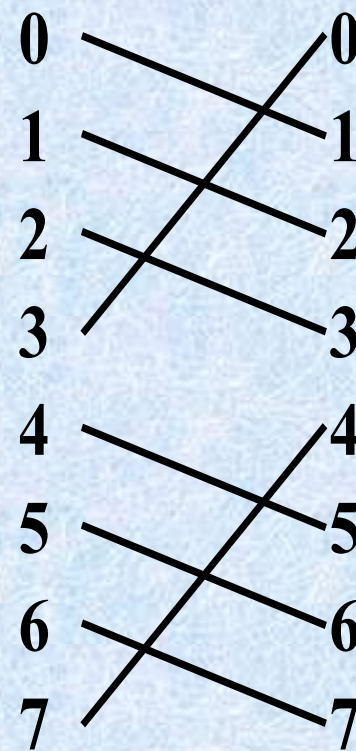
$i = 0$



$i = +1$

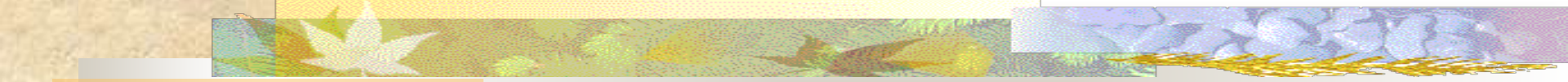


$i = +2$



$A_{(2)}$

移数函数



例：假设16个处理机的编号分别为0、1、...、15，采用单级互连网络。互连函数分别为：

(1)Cube3

(2)PM2+3

(3)PM2-0

(4)Shuffle

(5)Butterfly

(6)Reversal

第12号处理机分别与哪一个处理机相连？

解: $(12)_{10} = (1100)_2$

(1) Cube3, (2) PM2+3, (3) PM2-0,

(4) Shuffle, (5) Butterfly, (6) Reversal

1100最高位取反得0100, 4号处理机

$(12 + 8) \text{ MOD } 16 = 4$, 4号处理机

$12 - 1 = 11$, 11号处理机

1100循环左移1位得到1001, 9号处理机

1100的最高最低位交换0101, 5号处理机

1100的位序反过来为0011, 3号处理机



单级互联网络总结

(1) 单级互连网络特性

任一单级互连网络可实现部分结点(一对或几对)间的连接, 不能实现**任意多对**结点间的**同时连接**。

单级互连网络含义: 某些连接方法或拓扑结构。

(2) 单级互连网络应用

利用单级互连网络的特性作为实际ICN的拓扑结构;
通过交换开关作为ICN的可变因素;

通过交换开关多次控制实现ICN的结点间任意互连。

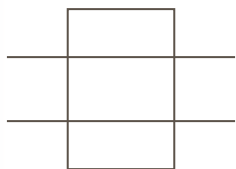
多级互连网络

- 多级互连网络采用多个相同的或不同的单级互连网络直接连接起来，实现任意节点间的直接互连
- 多级互连网络采用的关键技术：
 - ① 交换开关
 - ② 交换开关之间的拓扑连接
 - ③ 对交换开关的不同控制方式

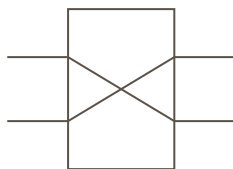
① 交换开关

- 一个 $a \times b$ 交换开关有 a 个输入和 b 个输出。最常用的二元开关： $a=b=2$ 。
- 每个输入可与一个或多个输出相连，但是在输出端必须避免发生冲突。一对一和一对多映射是容许的；但不容许有多对一映射。
- 只容许一对一映射时称为置换连接，称这种开关为交叉开关。
- 具有直通和交换两种功能的开关称为**二功能开关**，或交换开关。用一位控制信号控制。
- 具有所有4种功能的交换开关称为**四功能开关**，用两位控制信号控制。

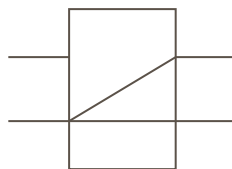
交换开关：具有传送或播送功能。



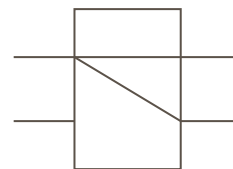
直通



交换



上播



下播

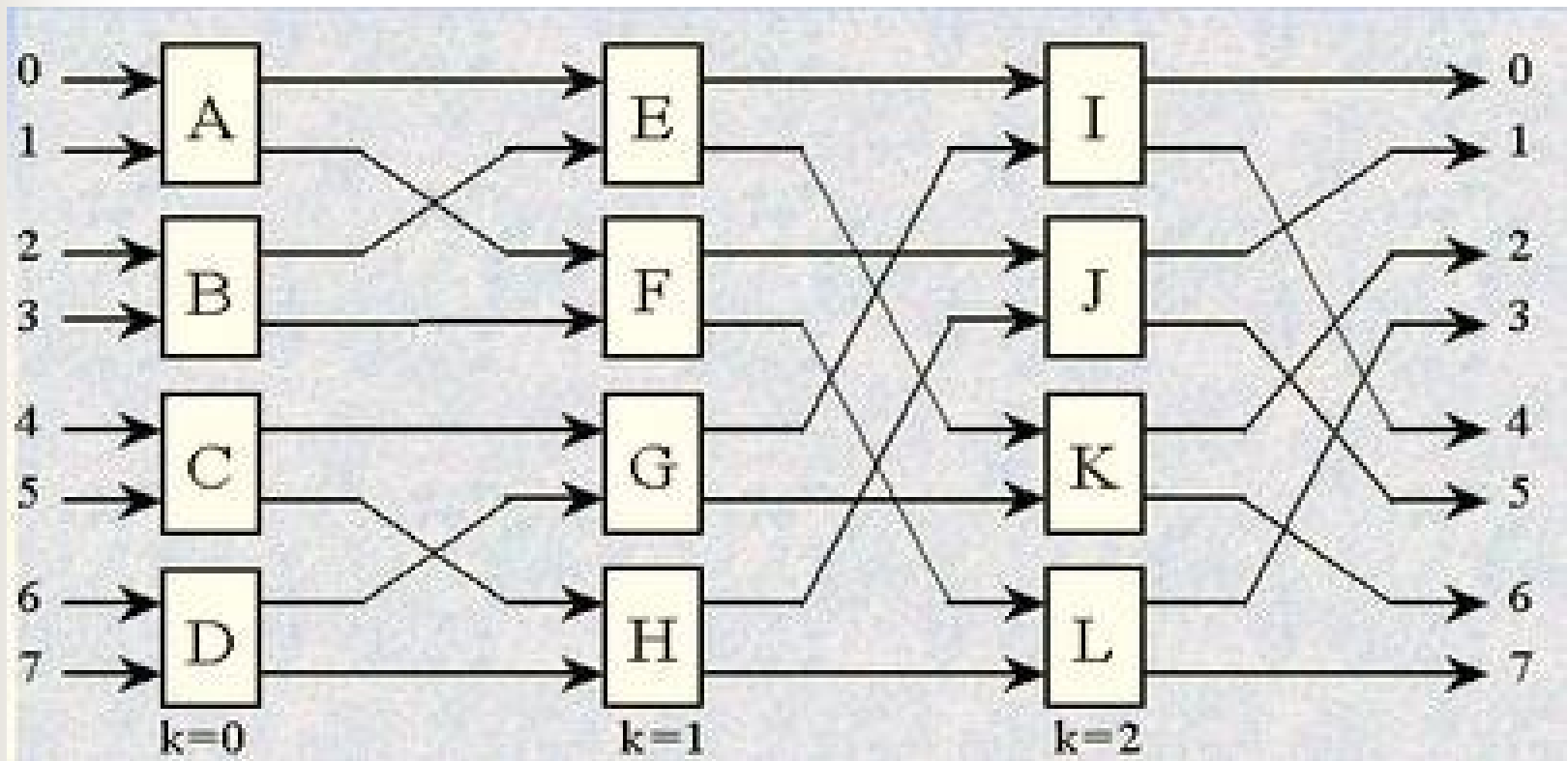
② 拓扑结构

- 前一级交换开关的输出端与后一级交换开关的输入端间的连接模式称为拓扑结构。
- 通常采用前面介绍过的互连函数实现拓扑结构。

③ 控制方式

- 有多级交换开关，每一级又有多个交换开关。
通常有三种控制方式
 - **级控制**：同一级交换开关使用同一个控制信号控制。
 - **单元级控制**：每个交换开关分别控制。
 - **部分级控制**：第 i 级使用 $i+1$ 个控制信号控制（ $0 \leq i \leq n-1$ ）
- 同一个多级互连网络分别采用三种不同的控制方式，可以构成三种不同的互连网络。

多级立方体网络STARAN

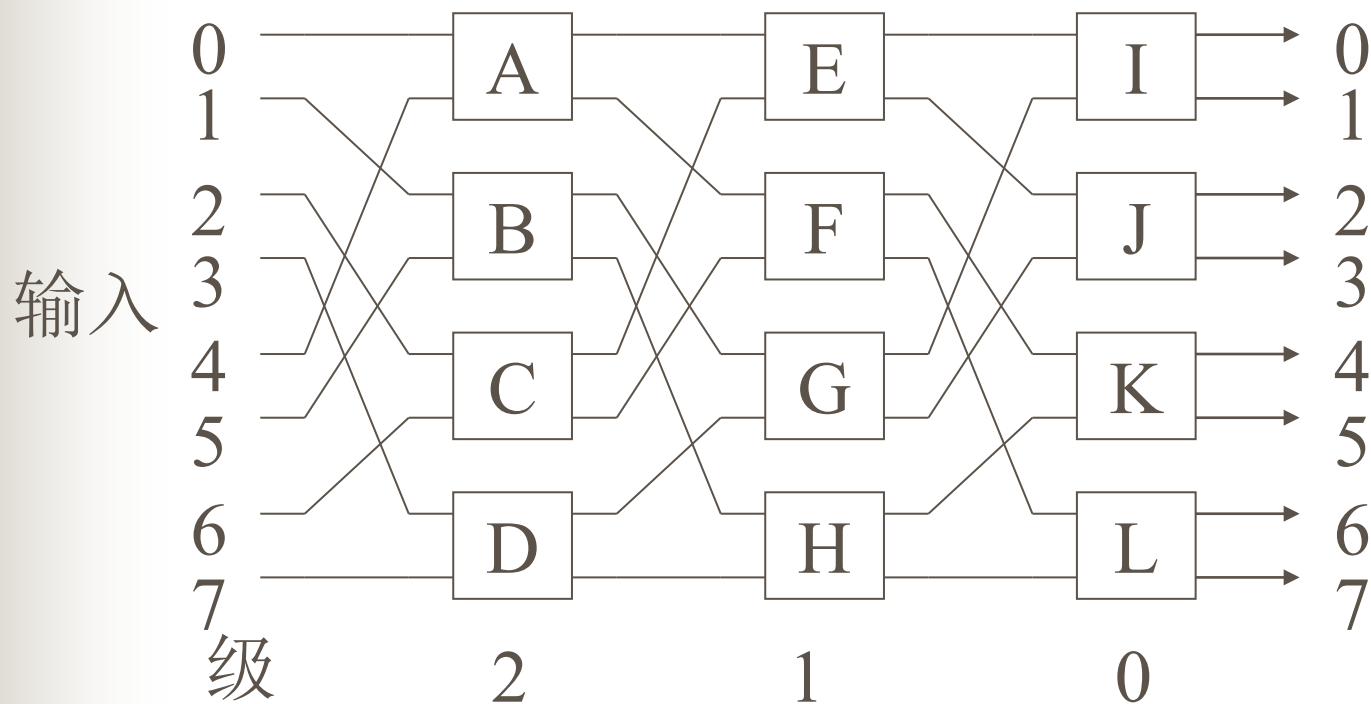


- **采用二功能开关**，总共需要开关 $n 2^{n-1}$ 个。
- **采用交换函数**，各级分别采用 E_0, E_1, \dots, E_{n-1} 函数。
 - 当所有开关都直通时，实现恒等变换。
 - 当A、B、C、D交换，其余直通实现 E_0 函数。
 - 当E、F、G、H交换，其余直通实现 E_1 函数。
 - 当I、J、K、L交换，其余直通实现 E_2 函数。
- **采用不同的控制方式**，可构成不同的互连网络。
 - 采用级控制可以构成STARAN交换网。
 - 采用部分级控制，可以构成STARAN移数网。
 - 采用级控制可以构成间接二进制n方体网。

控制：级控制(开关为1时交换功能， 否则为直通)

		级控制信号 ($k_2k_1k_0$)							
		000	001	010	011	100	101	110	111
入 端	0	0	1	2	3	4	5	6	7
	1	1	0	3	2	5	4	7	6
	2	2	3	0	1	6	7	4	5
	3	3	2	1	0	7	6	5	4
	4	4	5	6	7	0	1	2	3
	5	5	4	7	6	1	0	3	2
	6	6	7	4	5	2	3	0	1
	7	7	6	5	4	3	2	1	0
功 能		i	Cube0	Cube1	Cube0 + Cube1	Cube2	Cube0 + Cube2	Cube1 + Cube2	Cube0 + Cube1 + Cube2

多级混洗交换网络



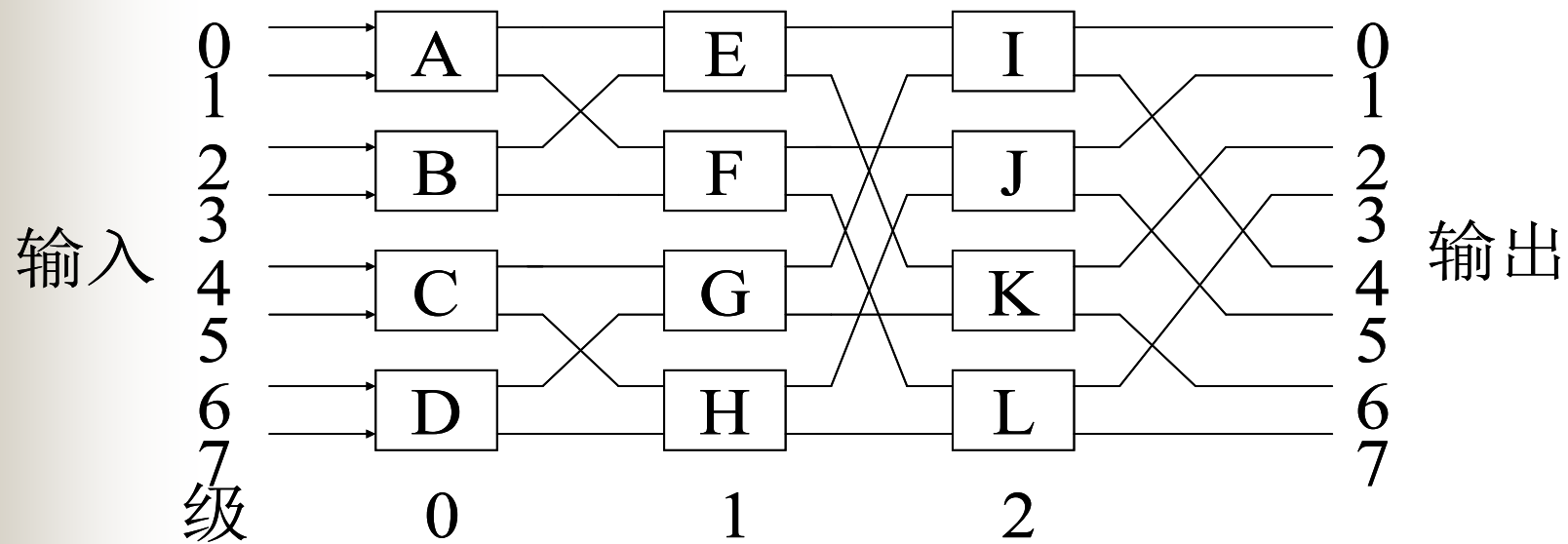
交换开关：四功能（允许实现一对多的连接）


拓扑结构：不同级均为全混洗结构；

控制方式：级控制、部分级控制、单元控制；

连接图：第 $n-1$ 级靠近入端；

例. 如图所示结点数为 $N=8$ 的多级混洗互连网络连接8个处理机, 编号为(0~7), 交换开关为二功能(直通和交换), 开关1为交换功能, 0为直通功能



- 
- ① 写出当级控制信号 $k_2k_1k_0=100$ 时，入端3连接的出端是几号？
 - ② 若实现入端为0与出端7的连接，写出级控制信号的控制字值；
 - ③ 写出实现如下连接的单元控制信号：
 $7 \rightarrow 2, 3 \rightarrow 6$
 - ④ 是否可用级控制信号同时实现如下连接？
 $6 \rightarrow 5$ 和 $3 \rightarrow 1$

超立方体寻径

- 假设有一个节点数为 $N=2^n$ 的 n 维立方体。每个结点的二进制编码为 $b=b_{n-1}b_{n-2}\dots b_1b_0$
- 源节点 $s=s_{n-1}s_{n-2}\dots s_1s_0$ ，目的节点 $d=d_{n-1}d_{n-2}\dots d_1d_0$ 。确定一条从 s 到 d 的步数最小的路径。
- 将 n 维表示成 $i=1, 2, \dots, n$ ，其中第 i 维对应于结点地址中的第 $i-1$ 位。设 $v=v_{n-1}v_{n-2}\dots v_1v_0$ 是路径中的任一结点。以下算法唯一确定一条从 s 到 d 的步数最小的路径：
 - ① 计算**方向位** $r_i=s_{i-1} \oplus d_{i-1}$ (异或)，其中 $i=1, 2, \dots, n$ 。
令 $i=1$ ， $v=s$ 。
 - ② 若 $r_i=1$ ，则从当前结点 v 寻径到下一结点 $v \oplus 2^{i-1}$ 。若 $r_i=0$ ，则跳过这一步。
 - ③ $i=i+1$ 。若 $i \leq n$ ，则转第2步，否则退出。

例如：假设有一个N= 16个结点的n=4方体，
s=0110， d=1101， 如何确定一条从s到d的步数最小的路径？

➤ 计算方向位： $i=1$ ， $v=s=0110$ ，

$$r_4 r_3 r_2 r_1 = 0110 \oplus 1101 = 1011$$

➤ 第1步： $r_1=1$ ， $v=v \oplus 2^{i-1} = 0110 \oplus 0001 = 0111$ ， $i=i+1=2$

➤ 第2步： $r_2=1$ ， $v=v \oplus 2^{i-1} = 0111 \oplus 0010 = 0101$ ， $i=i+1=3$

➤ 第3步： $r_3=0$ ， 跳过一步， $i=i+1=4$

➤ 第4步： $r_4=1$ ， $v=v \oplus 2^{i-1} = 0101 \oplus 1000 = 1101$ ，
 $i=i+1=5$ ， 结束

➤ 因此路径为： $0110 \rightarrow 0111 \rightarrow 0101 \rightarrow 1101$ 。

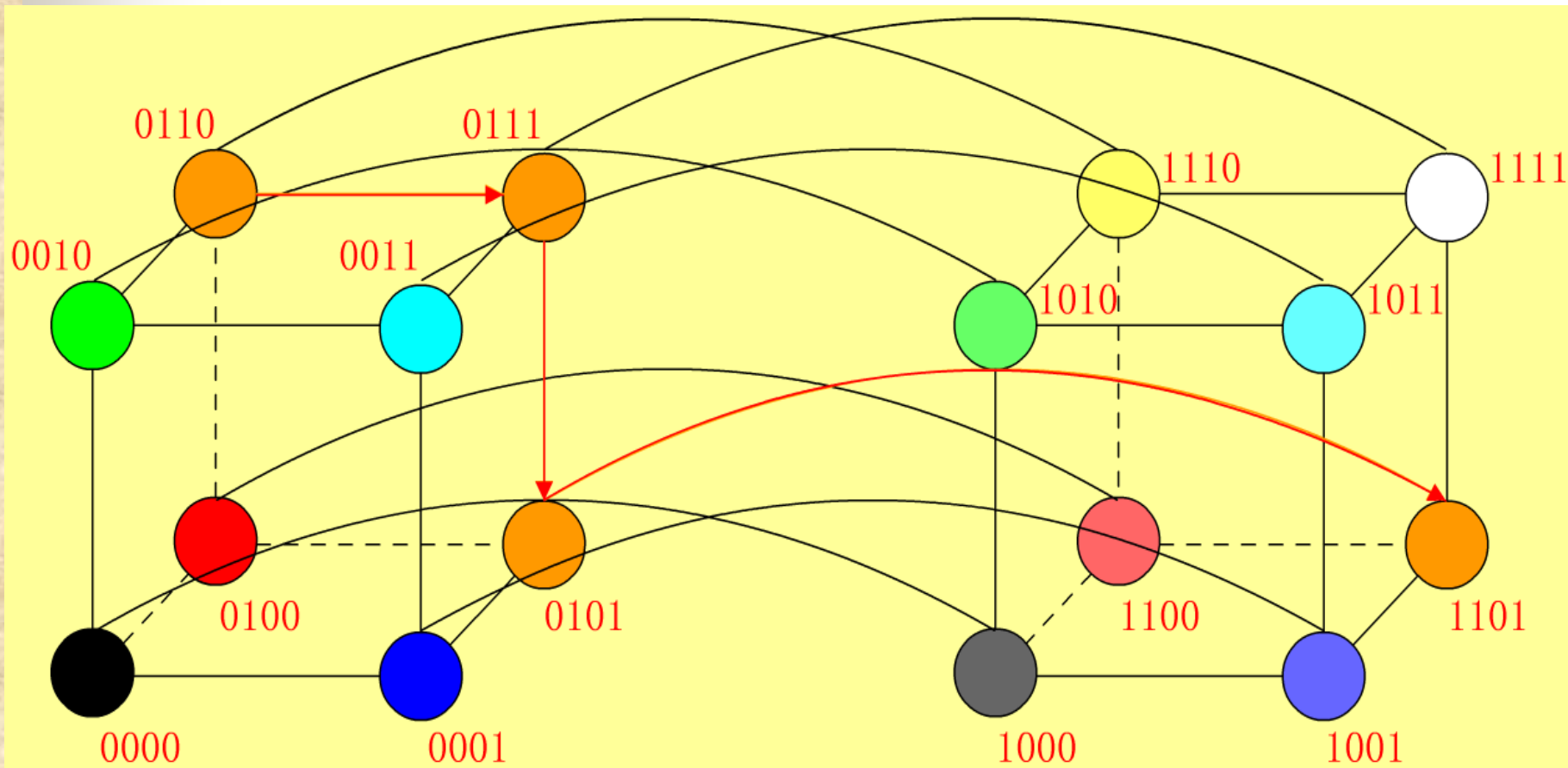


图5.18 E-立方路由算法示例



本章重点：

- 并行性概念
- 并行处理机的特点
- 典型互联网络



作业

- “习题”文件中的“第五章”部分