第9章 互连网络——An overview of network

Switched networks are replacing buses as the normal means of communication between computers, between I/O devices, between boards, between chips, and even between modules inside chips.

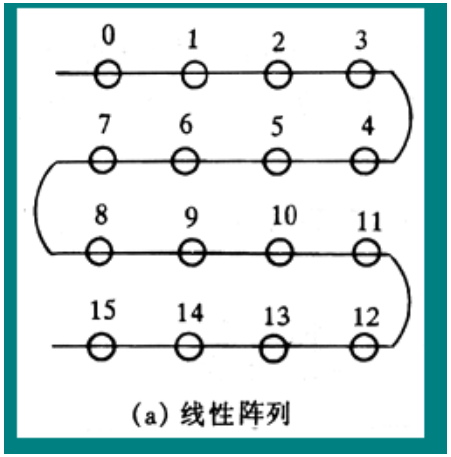
设计目标

The ultimate goal of computer architects is to design interconnection networks of the ***lowest possible cost*** that are capable of transferring the ***maximum amount of available information*** in the ***shortest possible time.***

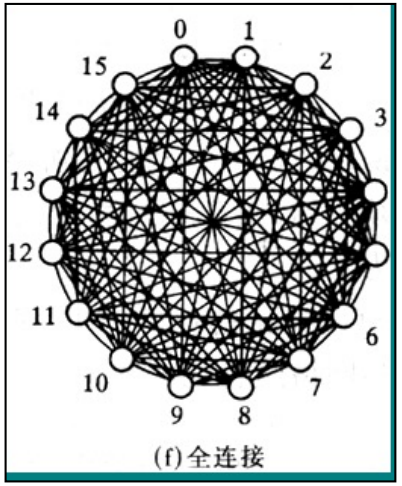
——低成本

——高带宽

——低延迟



设计空间



互连网络的类型

On-chip networks (OCNs) — Also referred to as network-on-chip (NoC), Interconnect microarchitecture functional units, register files, caches, compute tiles, and processor and IP cores within chips or multichip modules.

System/storage area networks (SANs)— used for interprocessor and processor-memory interconnections within multiprocessor and multicomputer systems, and also for the connection of storage and I/O components within server and data center environments.

Local area networks (LANs)—used for interconnecting computer systems distributed across a machine room or throughout a building or campus environment. ***Ethernet*** has a 10 Gbps standard version that supports maximum performance over a distance of 40 km.

Wide area networks (WANs)—connect computer systems distributed across the globe, which requires internetworking support. WANs connect many millions of computers over distance scales of many thousands of kilometers.

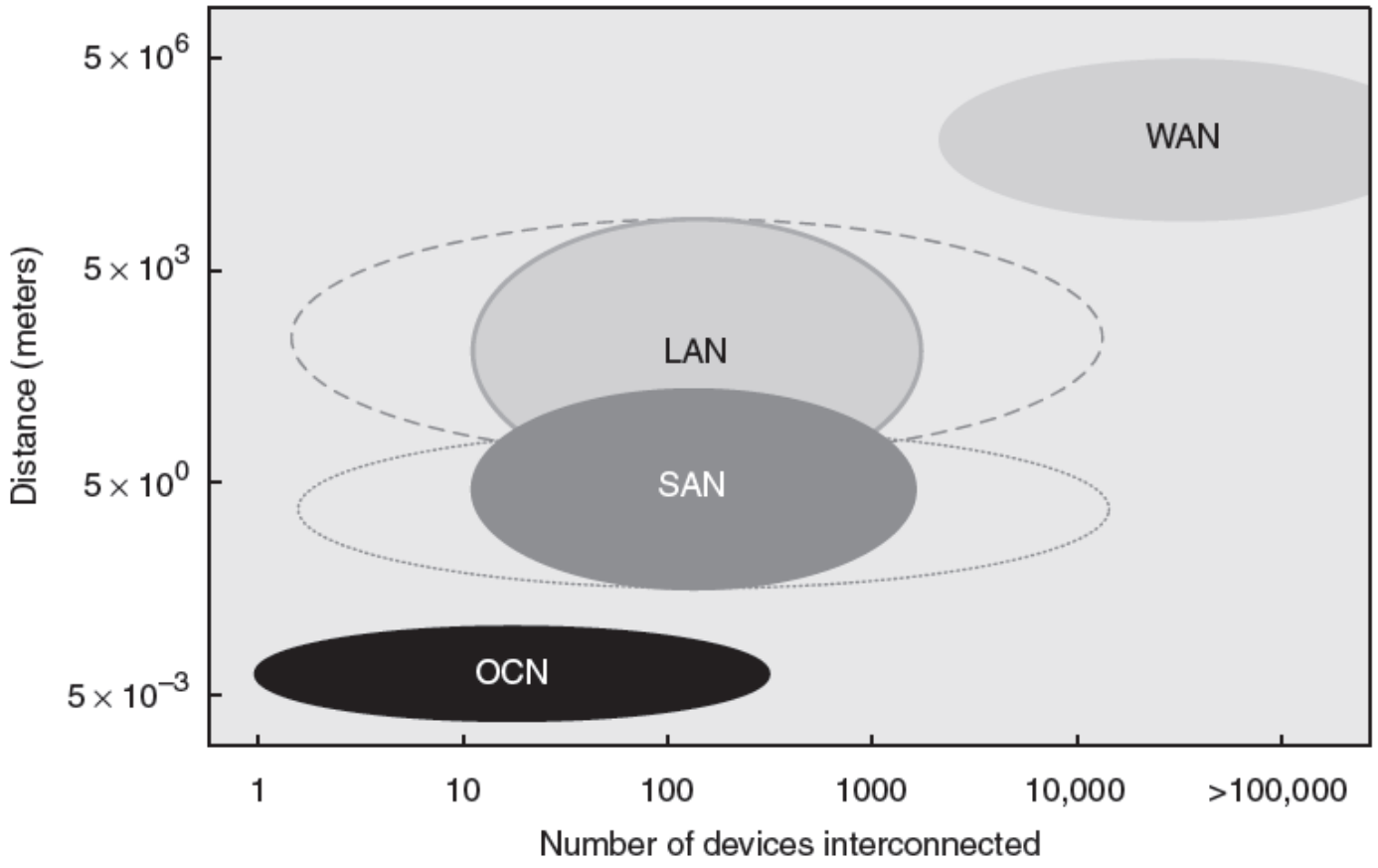


Figure F.2 **Relationship of the four interconnection network domains in terms of number of devices connected and their distance scales: on-chip network (OCN), system/storage area network (SAN), local area network (LAN), and wide area network (WAN).** Note that there are overlapping ranges where some of these networks com-

On-Chip Network: Intel Single-Chip Cloud Computer

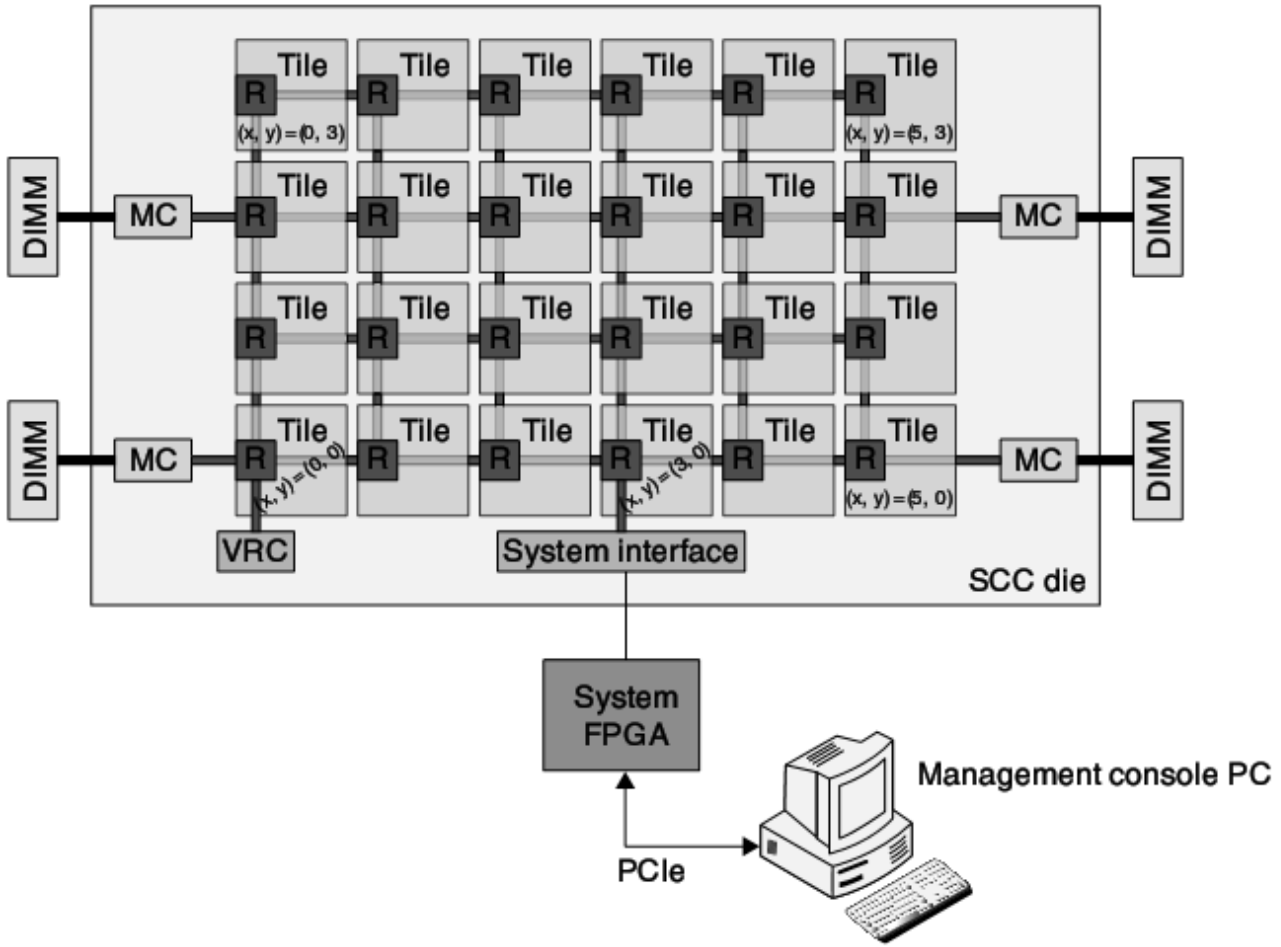
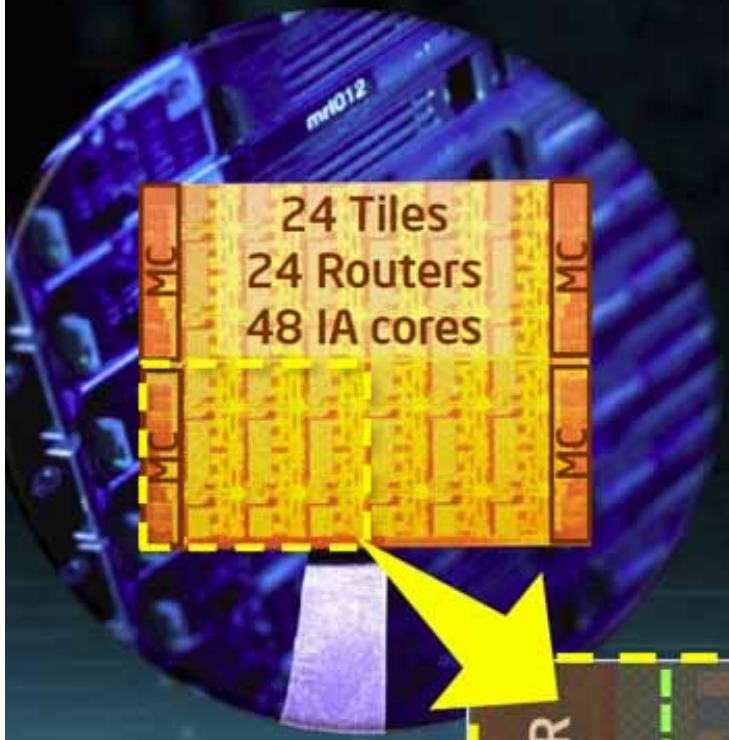
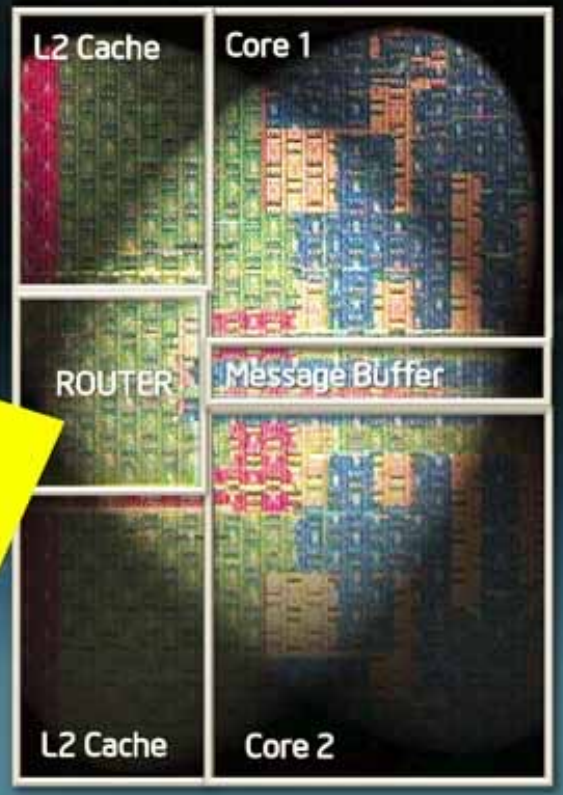


Figure F.28 **SCC Top-level architecture.** From Howard, J. et al., *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 58–59.

Inside the SCC



Dual-core SCDC Tile



- 2D mesh network with 256 GB/s bisection bandwidth
- 4 Integrated DDR3 memory controllers (64GB addressable)

Network name [vendors]	Used in top 10 supercom-puter clusters (2005)	Number of nodes	Basic network topology	Raw link bidirec-tional BW	Routing algorithm	Arbitration technique	Switching technique; flow control
InfiniBand [Mellanox, Voltair]	SGI Altrix and Dell Poweredge Thunderbird	>Millions (2^{128} GUID addresses, like IPv6)	Completely configurable (arbitrary)	4–240 Gbps	Arbitrary (table-driven), typically up*/down*	Weighted RR fair scheduling (2-level priority)	Cut-through, 16 virtual channels (15 for data); credit-based
Myrinet-2000 [Myricom]	Barcelona Supercomputer Center in Spain	8192 nodes	Bidirectional MIN with 16-port bidirectional switches (Clos net.)	4 Gbps	Source-based dispersive (adaptive) minimal routing	Round-robin arbitration	Cut-through switching with no virtual channels; Xon/Xoff flow control
$QsNet^{II}$ [Quadrics]	Intel Thunder Itanium2 Tiger4	>Tens of thousands	Fat tree with 8-port bidirectional switches	21.3 Gbps	Source-based LCA adaptive shortest-path routing	2-phased RR, priority, aging, distributed at output ports	Wormhole with 2 virtual channels; credit-based

Figure F.31 Characteristics of system area networks implemented in various top 10 supercomputer clusters in 2005.

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。

- 结点：处理器、存储模块或其它设备。
- 在拓扑上，互连网络为输入结点到输出结点之间的一组互连或映象。

9.1 互连函数

9.1.1 互连函数

变量x：输入（设x=0，1，…，N－1）

函数f(x)：输出

通过数学表达式建立输入端号与输出端号的连接关系。即在互连函数f的作用下，输入端x连接到输出端f(x)。

- 互连函数反映了网络输入数组和输出数组之间对应的置换关系或排列关系。

（有时也称为**置换函数**或**排列函数**）

- 互连函数f(x)有时可以采用循环表示

  即：（x_0 x_1 x_2 … x_{j-1}）

  表示：$f(x_0)=x_1$，$f(x_1)=x_2$，…，$f(x_{j-1})=x_0$

  j称为该循环的长度。

- 设$n=\log_2 N$，则可以用n位二进制来表示N个输入端和输出端的二进制地址，互连函数表示为：

  $f(x_{n-1}x_{n-2}\cdots x_1x_0)$

9.1.2 几种基本的互连函数

介绍几种常用的基本互连函数及其主要特征。

1. 恒等函数

- 恒等函数：实现同号输入端和输出端之间的连接。

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

2. 交换函数

- 交换函数：实现二进制地址编码中第k位互反的输入端与输出端之间的连接。

$$E(x_{n-1}x_{n-2}\cdots x_{k+1}x_kx_{k-1}\cdots x_1x_0)=x_{n-1}x_{n-2}\cdots x_{k+1}\bar{x}_kx_{k-1}\cdots x_1x_0$$

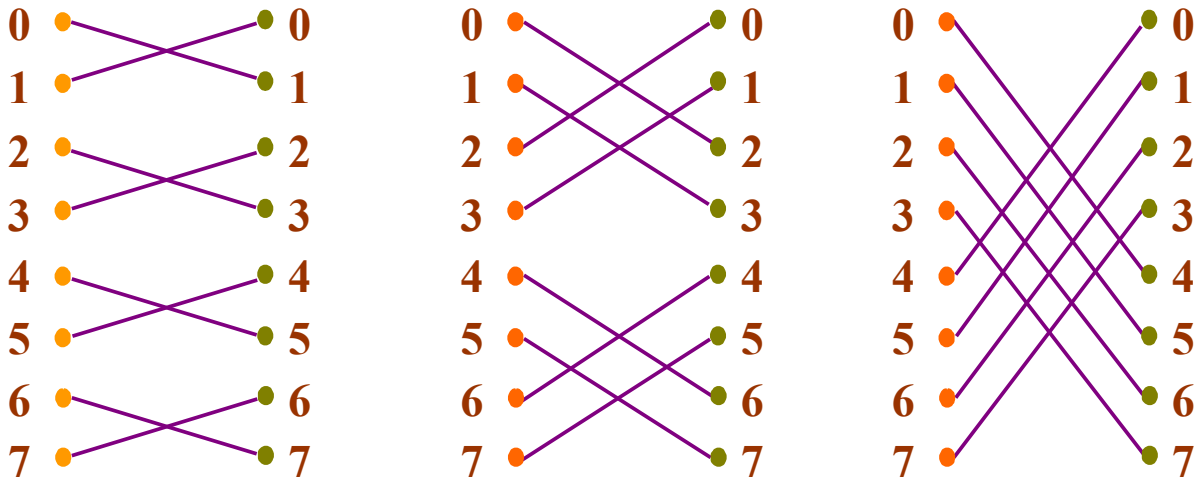
- 主要用于构造立方体互连网络和各种超立方体互连网络。
- 它共有$n=\log_2 N$种互连函数。

（N为结点个数）

- 当N＝8时，n＝3，可得到常用的立方体互连函数：

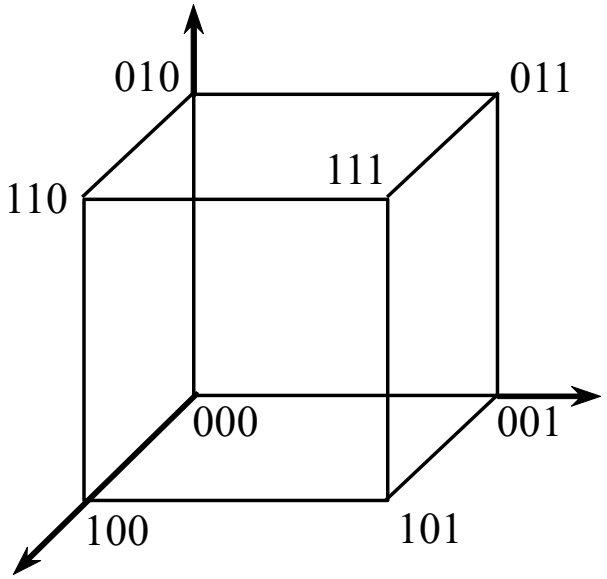
$$Cube_0(x_2x_1x_0)=x_2x_1\bar{x}_0$$
$$Cube_1(x_2x_1x_0)=x_2\bar{x}_1x_0$$
$$Cube_2(x_2x_1x_0)=\bar{x}_2x_1x_0$$

- 变换图形



(a) $Cube_0$ 交换函数　(b) $Cube_1$ 交换函数　(c) $Cube_2$ 交换函数

N=8 的立方体交换函数



立方体网络

3. 均匀洗牌函数

- 均匀洗牌函数：将输入端分成数目相等的两半，前一半和后一半按类似均匀混洗扑克牌的方式交叉地连接到输出端（输出端相当于混洗的结果）。
  - 也称为混洗函数（置换）
  - 函数关系

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

即把输入端的二进制编号循环左移一位。

- 互连函数（设为s）的第k个子函数：把s作用于输入端的二进制编号的低k位。
- 互连函数（设为s）的第k个超函数：把s作用于输入端的二进制编号的高k位。

例如：对于均匀洗牌函数

第k个子函数：

$$\sigma_{(k)}(x_{n-1}\cdots x_k \mid x_{k-1}x_{k-2}\cdots x_0) = x_{n-1}\cdots x_k \mid x_{k-2}\cdots x_0x_{k-1}$$

即把输入端的二进制编号中的低k位循环左移一位。

第k个超函数：

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

即把输入端的二进制编号中的高k位循环左移一位。

下列等式成立：

$$\sigma^{(n)}(X)=\sigma_{(n)}(X)=\sigma(X)$$

$$\sigma^{(1)}(X)=\sigma_{(1)}(X)=X$$

- 对于任意一种函数f(x)，如果存在g(x)，使得

$$f(x)\times g(x)=I(x)$$

则称g(x)是f(x)的逆函数，记为$f^{-1}(x)$。

$$f^{-1}(x)= g(x)$$

- 逆均匀洗牌函数：将输入端的二进制编号循环右移一位而得到所连接的输出端编号。

- 互连函数

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

- 逆均匀洗牌是均匀洗牌的逆函数

➢ 当N=8时，有：

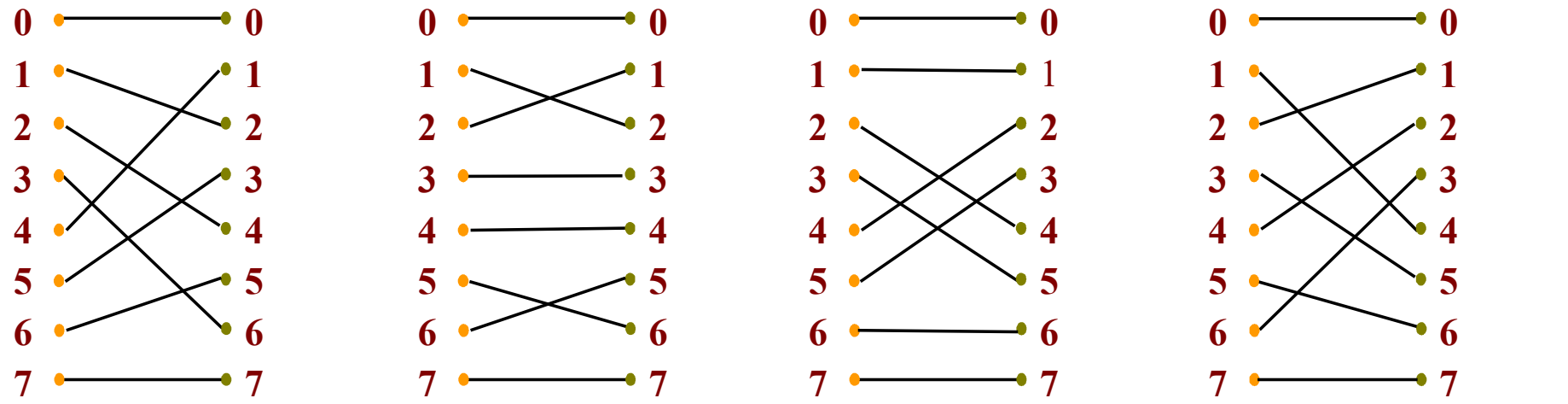
$\sigma(x_2x_1x_0)=x_1x_0x_2$

$\sigma_{(2)}(x_2x_1x_0)=x_2x_0x_1$

$\sigma^{(2)}(x_2x_1x_0)=x_1x_2x_0$

$\sigma^{-1}(x_2x_1x_0)=x_0x_2x_1$

- N=8 的均匀洗牌和逆均匀洗牌函数



(a) 均匀洗牌函数 σ　(b) 子洗牌函数 $\sigma_{(2)}$　(c) 超洗牌函数 $\sigma^{(2)}$　(d) 逆均匀洗牌函数 σ^{-1}

N=8 的均匀洗牌函数

4. 蝶式函数

- 蝶式互连函数：把输入端的二进制编号的最高位与最低位互换位置，便得到了输出端的编号。

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

- 第k个子函数

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

把输入端的二进制编号的低k位中的最高位与最低位互换。

- 第k个超函数

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

把输入端的二进制编号的高k位中的最高位与最低位互换。

- 下列等式成立

$$\beta^{(n)}(X)=\beta_{(n)}(X)=\beta(X)$$

$$\beta^{(1)}(X)=\beta_{(1)}(X)=X$$

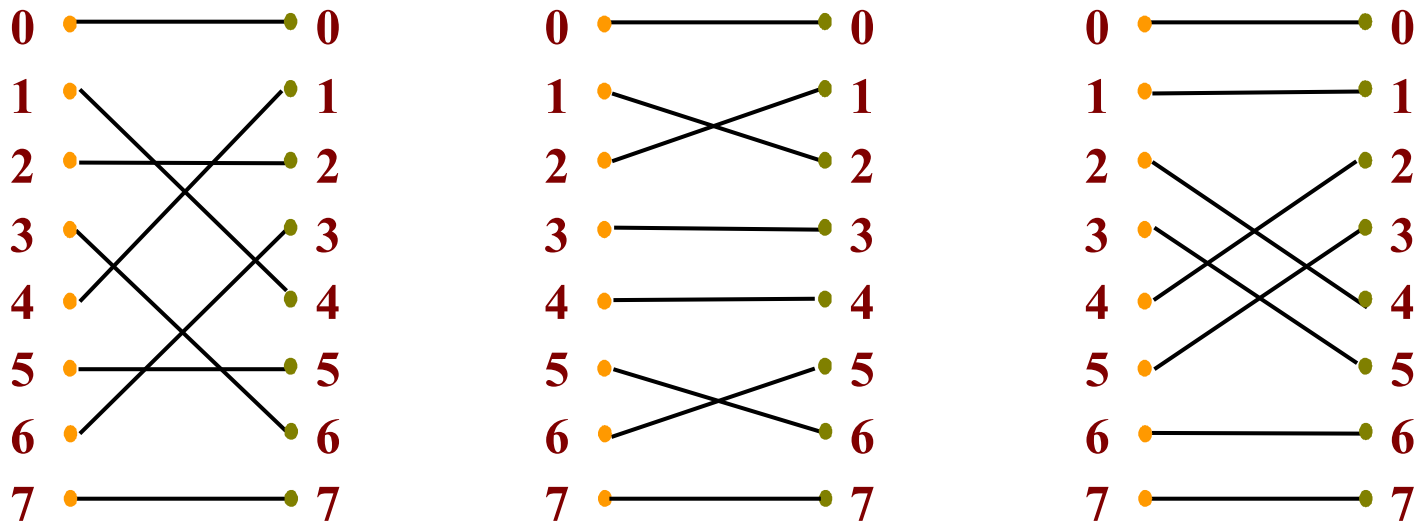
- 当N=8时，有：

$$\beta(x_2x_1x_0)=x_0x_1x_2$$

$$\beta_{(2)}(x_2x_1x_0)=x_2x_0x_1$$

$$\beta^{(2)}(x_2x_1x_0)=x_1x_2x_0$$

- 蝶式变换与交换变换的多级组合可作为构成方体多级网络的基础。



(a) $\beta = \rho$　　(b) $\beta_{(2)} = \rho_{(2)}$　　(c) $\beta^{(2)} = \rho^{(2)}$

N=8 的蝶式函数和反位序函数

5. 反位序函数

- 反位序函数：将输入端二进制编号的位序颠倒过来求得相应输出端的编号。
  - **互连函数**

$$\rho\left(x_{n-1}x_{n-2}\cdots x_1x_0\right)=x_0x_1\cdots x_{n-2}x_{n-1}$$

- 第k个子函数

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

即把输入端的二进制编号的低k位中各位的次序颠倒过来。

- 第k个超函数

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

即把输入端的二进制编号的高k位中各位的次序颠倒过来。

- 下列等式成立

$$\rho^{(n)}(X)=\rho_{(n)}(X)=\rho(X)$$

$$\rho^{(1)}(X)=\rho_{(1)}(X)=X$$

- 当N=8时，有：

$$\rho(x_2x_1x_0)=x_0x_1x_2$$

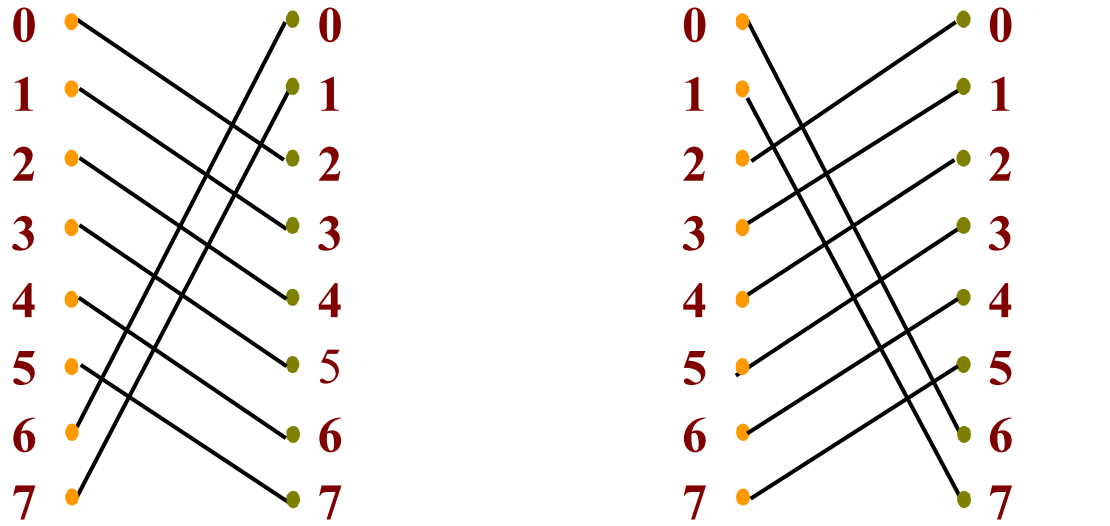
$$\rho_{(2)}(x_2x_1x_0)=x_2x_0x_1$$

$$\rho^{(2)}(x_2x_1x_0)=x_1x_2x_0$$

6. 移数函数

- 移数函数：将各输入端都错开一定的位置（模N）后连到输出端。
  - 函数式

$$\alpha(x)=（x\pm k）\bmod N \quad 1\le x\le N-1，1\le k\le N-1$$



(a) 左移移数函数 k＝2　　(b) 右移移数函数 k＝2

7. PM2I函数

- P和M分别表示加和减，2I表示2^i。
  - 该函数又称为“加减2^i”函数。
- PM2I函数：一种移数函数，将各输入端都错开一定的位置（模N）后连到输出端。
- 互连函数

$$PM2_{+i}(x) = x + 2^i \bmod N$$

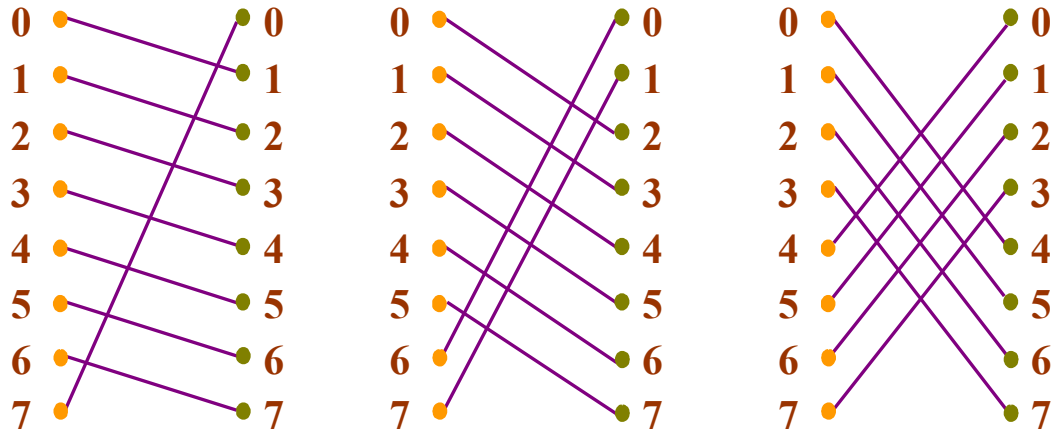
$$PM2_{-i}(x) = x - 2^i \bmod N$$

其中：

$0 \leqslant x \leqslant N-1$，$0 \leqslant i \leqslant n-1$，$n = \log_2 N$，N为结点数。

- PM2I互连网络共有2n个互连函数。

- 当N＝8时，有6个PM2I函数：
  - $PM2_{+0}$ ： (0 1 2 3 4 5 6 7)
  - $PM2_{-0}$ ： (7 6 5 4 3 2 1 0)
  - $PM2_{+1}$ ： (0 2 4 6) (1 3 5 7)
  - $PM2_{-1}$ ： (6 4 2 0) (7 5 3 1)
  - $PM2_{+2}$ ： (0 4) (1 5) (2 6) (3 7)
  - $PM2_{-2}$ ： (4 0) (5 1) (6 2) (7 3)

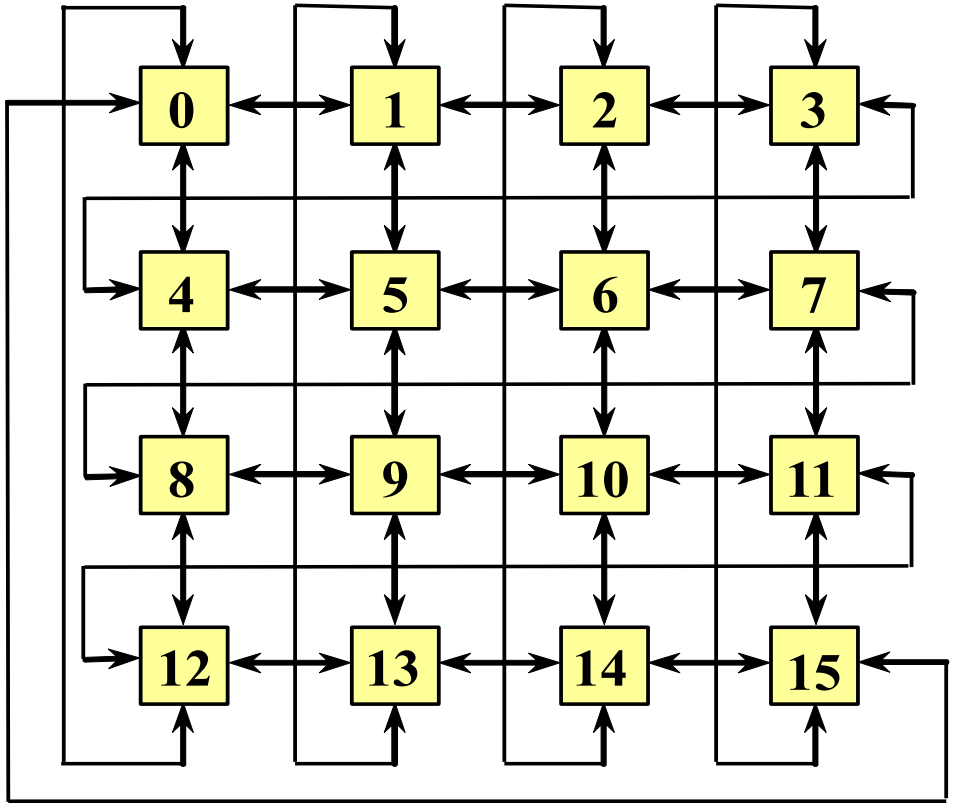


(a) $PM2_{+0}$ (b) $PM2_{+1}$ (c) $PM2_{+2}$

N=8 的PM2I函数

➢ 阵列计算机ILLIAC Ⅳ

- 采用$PM2_{\pm 0}$和$PM2_{\pm n/2}$构成其互连网络，实现各处理单元之间的上下左右互连 。



用移数函数构成ILLIAC Ⅳ 阵列机的互连网络

- $PM2_{+0}$ ： （0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15）
- $PM2_{-0}$ ： （15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0）
- $PM2_{+2}$ ：
- $PM2_{-2}$ ：

9.2 互连网络的结构参数与性能指标

9.2.1 互连网络的结构参数

1. 网络通常是用有向边或无向边连接有限个结点的图来表示。
2. 互连网络的主要特性参数有：
   - 网络规模N：网络中结点的个数。

     表示该网络所能连接的部件的数量。

- 结点度d：与结点相连接的边数（通道数），包括入度和出度。
  - 进入结点的边数叫入度。
  - 从结点出来的边数叫出度。
- 结点距离：对于网络中的任意两个结点，从一个结点出发到另一个结点终止所需要跨越的边数的最小值。
- 网络直径D：网络中任意两个结点之间距离的最大值。

  网络直径应当尽可能地小。

- 等分宽度b：把由N个结点构成的网络切成结点数相同（N/2）的两半，在各种切法中，沿切口边数的最小值。
- 对称性：从任何结点看到的拓扑结构都是相同的网络称为对称网络。

9.2.2 互连网络的性能指标

评估互连网络性能的两个基本指标：时延和带宽

1. 通信时延

指从源结点到目的结点传送一条消息所需的总时间，它由以下4部分构成：

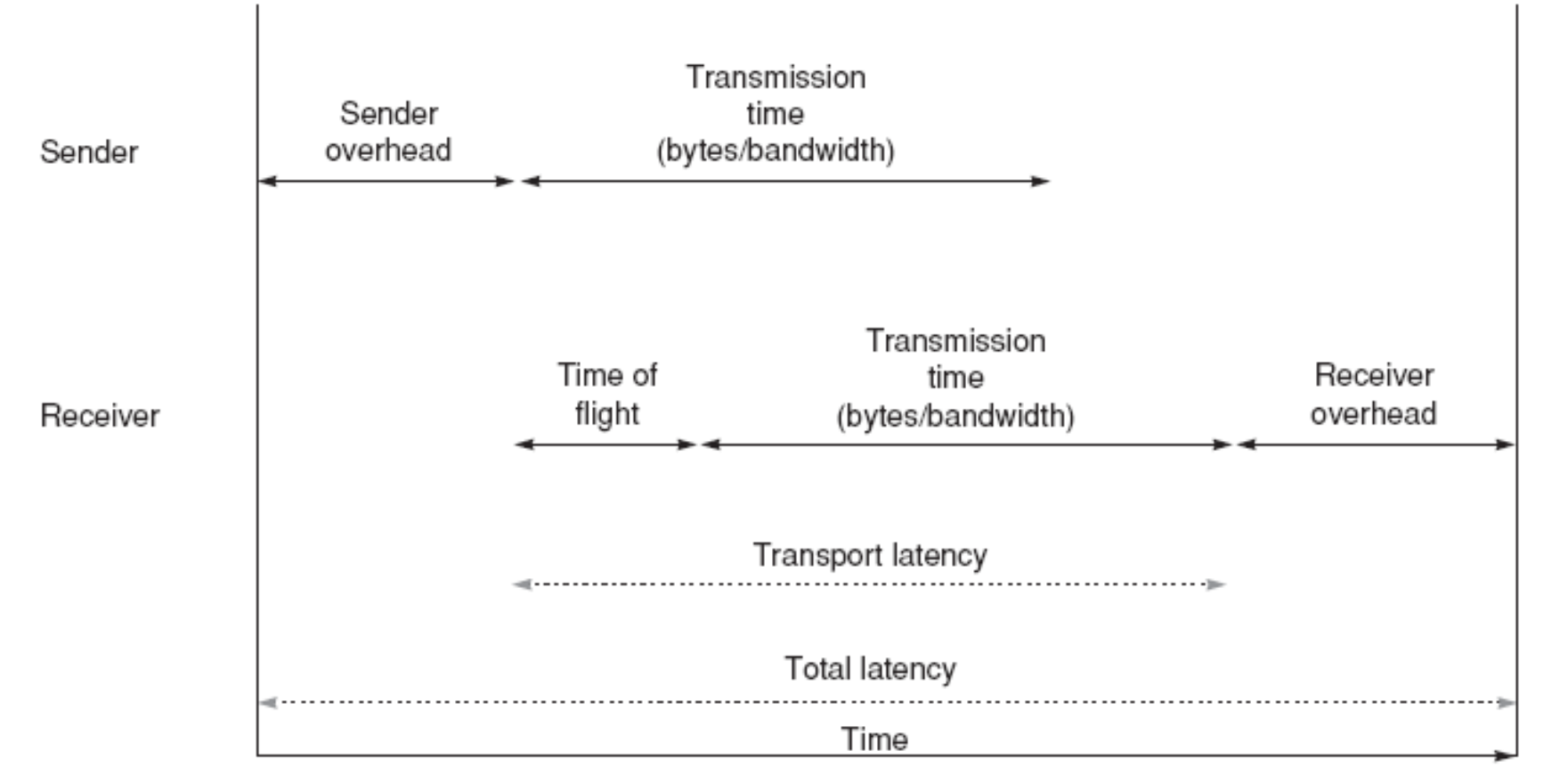
- 软件开销：在源结点和目的结点用于收发消息的软件所需的执行时间。
- 通道时延：通过通道传送消息所花的时间。
  - **通路时延 = 消息长度/通道带宽**

- 选路时延：消息在传送路径上所需的一系列选路决策所需的时间开销。
  - **与传送路径上的结点数成正比。**
- 竞争时延：多个消息同时在网络中传送时，会发生争用网络资源的冲突。为避免或解决争用冲突所需的时间就是竞争时延。

2. 网络时延

通道时延与选路时延的和。

Components of packet latency



3. 端口带宽

- 对于互连网络中的任意一个端口来说，其端口带宽是指单位时间内从该端口传送到其他端口的最大信息量。
  - **在对称网络中，端口带宽与端口位置无关。网络的端口带宽与各端口的端口带宽相同。**
  - **非对称网络的端口带宽则是指所有端口带宽的最小值。**

4. 聚集带宽

- 网络从一半结点到另一半结点，单位时间内能够传送的最大信息量。

5. 等分带宽

- 与等分宽度对应的切平面中，所有边合起来单位时间所能传送的最大信息量
- 等分宽度b：把由N个结点构成的网络切成结点数相同（N/2）的两半，在各种切法中，沿切口边数的最小值。

9.3 静态互连网络

互连网络通常可以分为两大类：

- 静态互连网络

  各结点之间有固定的连接通路、且在运行中不能改变的网络。

- 动态互连网络

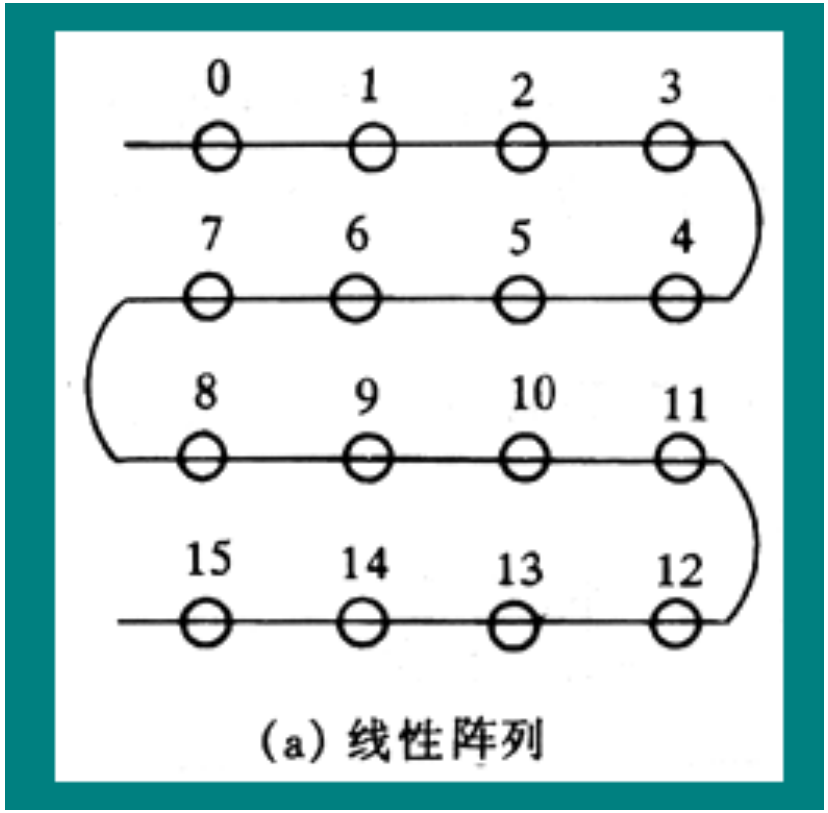
  由交换开关构成、可按运行程序的要求动态地改变连接状态的网络。

下面介绍几种静态互连网络。

（其中：N表示结点的个数）

1. 线性阵列：一种一维的线性网络，其中N个结点用N-1个链路连成一行。

- 端结点的度：1
- 其余结点的度：2
- 直径：N－1
- 等分宽度b=1

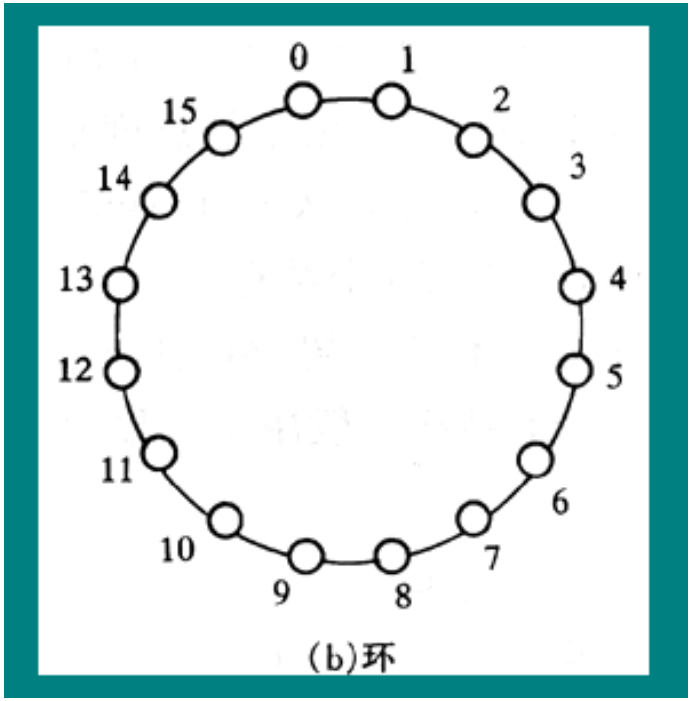


(a) 线性阵列

2. 环和带弦环

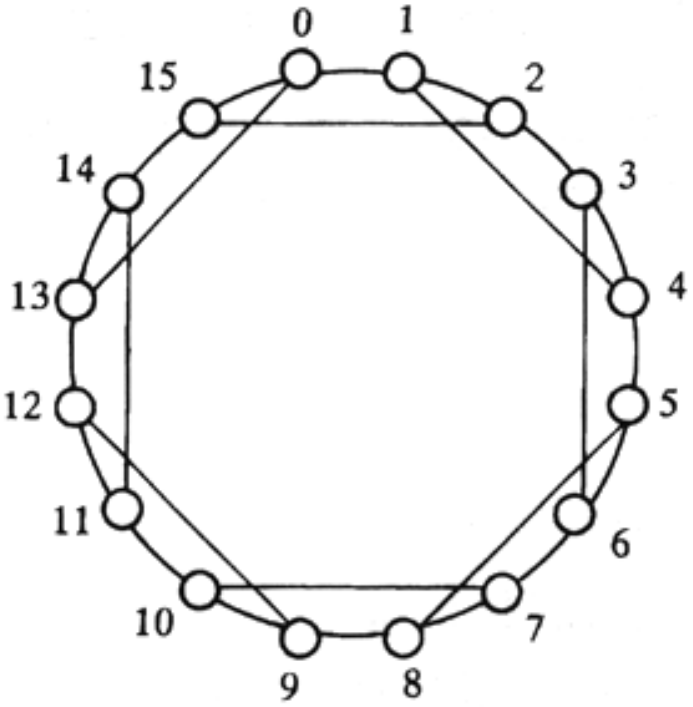
- 环：用一条附加链路将线性阵列的两个端点连接起来而构成。可以单向工作，也可以双向工作。

  - 对称
  - 结点的度：2
  - 双向环的直径：N/2
  - 单向环的直径：N
  - 环的等分宽度b=2

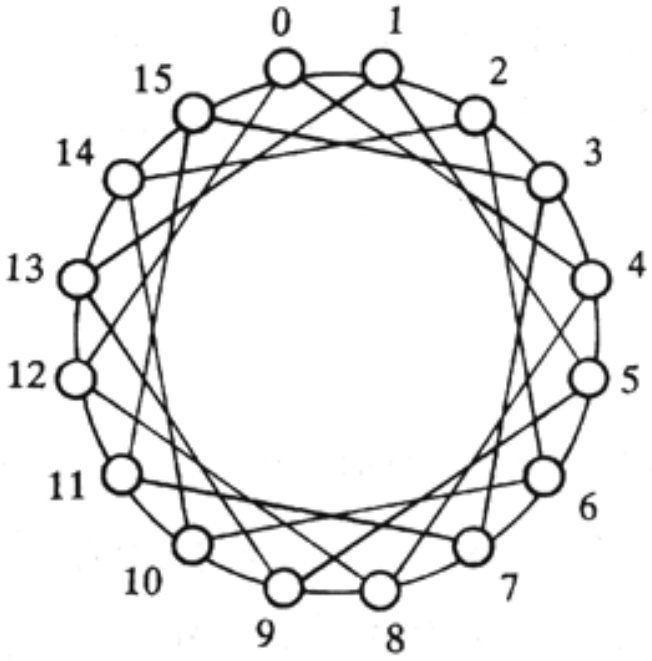


- 带弦环

增加的链路愈多，结点度愈高，网络直径就愈小。

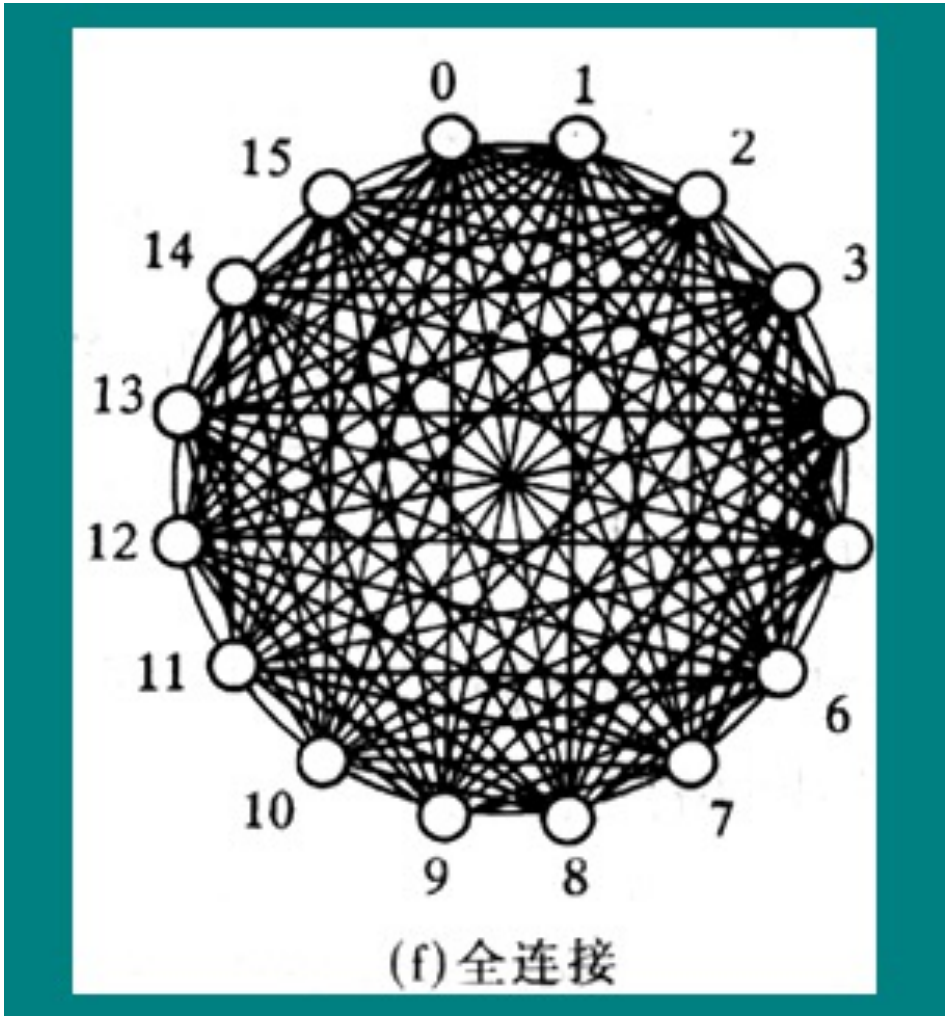


(c)度为 3 的带弦环



(d)度为 4 的带弦环(与 Illiac 网相同)

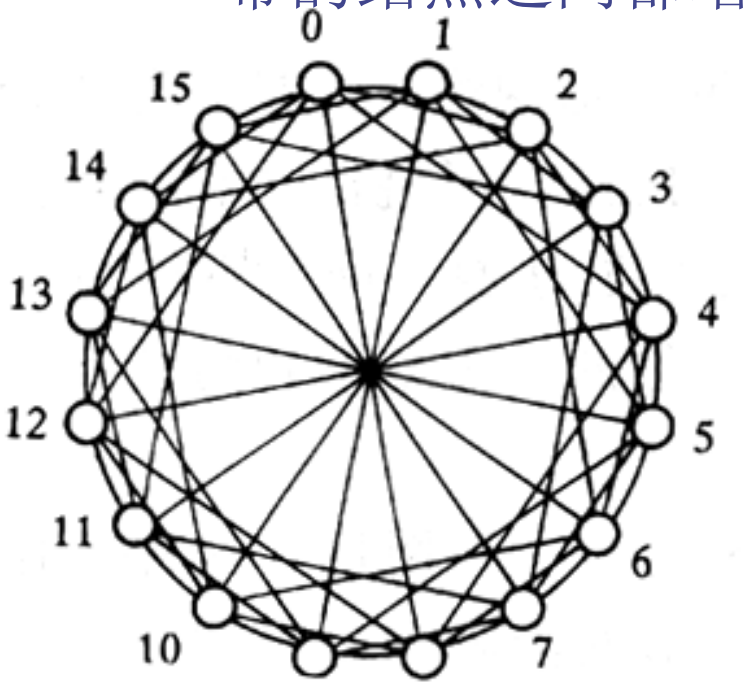
- 全连接网络
  - 结点度：15
  - 直径为1。



(f)全连接

3. 循环移数网络（PM2I，Barrel Shifter）

- 通过在环上每个结点到所有与其距离为2的整数幂的结点之间都增加一条附加链而构成。



(e)循环移数网络

N=16

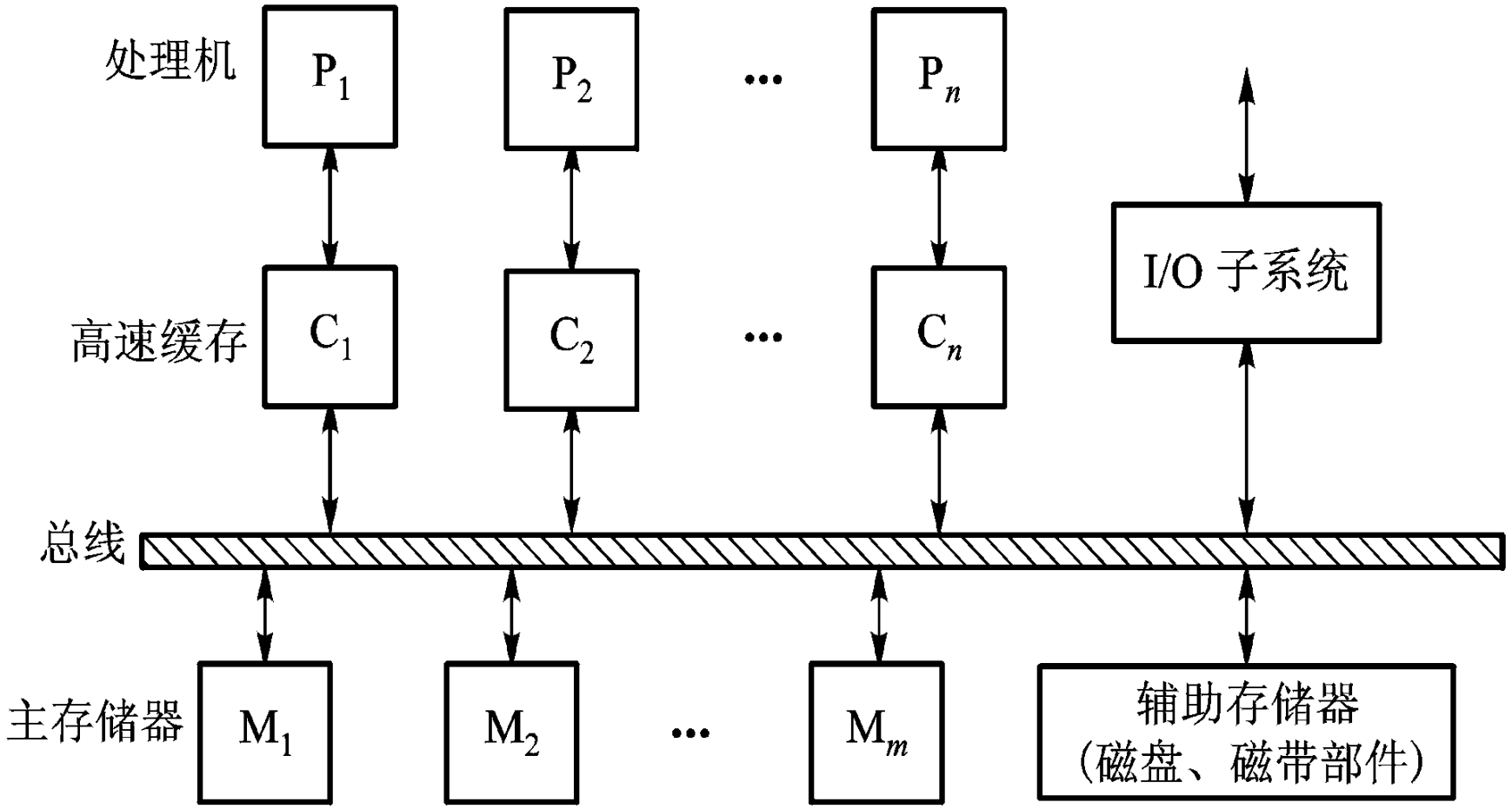
- 结点度：7
- 直径：2

9.4 动态互连网络

9.4.1 总线网络

1. 由一组导线和插座构成，经常被用来实现计算机系统中处理机模块、存储模块和外围设备等之间的互连。
   - 每一次总线只能用于一个源（主部件）到一个或多个目的（从部件）之间的数据传送。
   - 多个功能模块之间的争用总线或时分总线
   - 特点
     - **结构简单、实现成本低、带宽较窄**

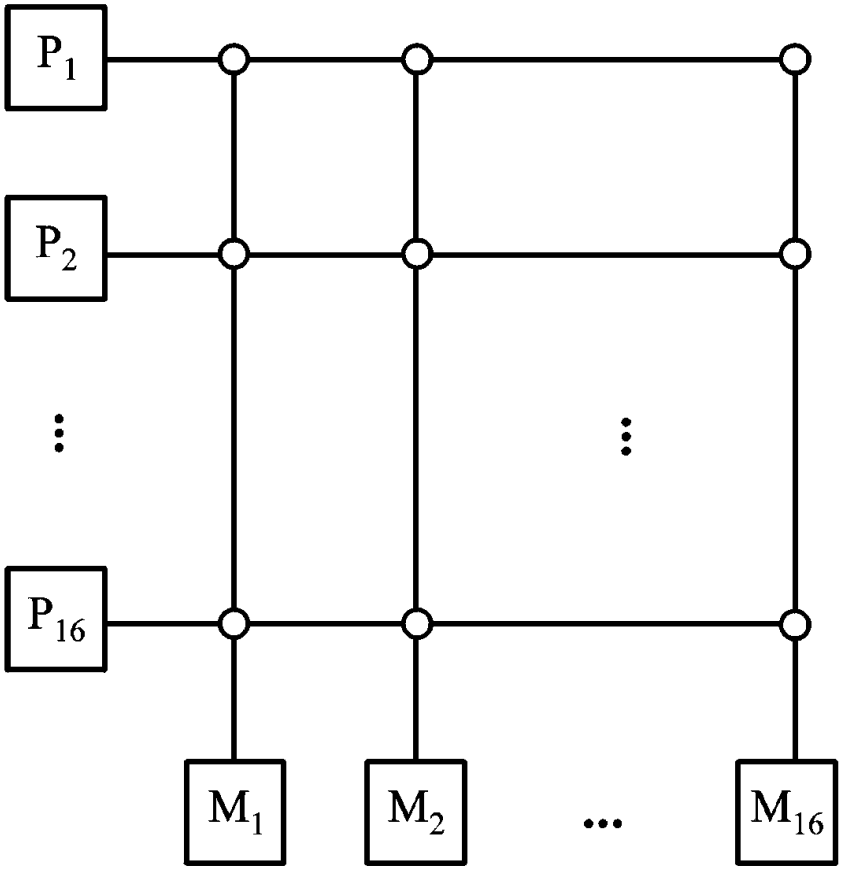
2. 一种由总线连接的多处理机系统

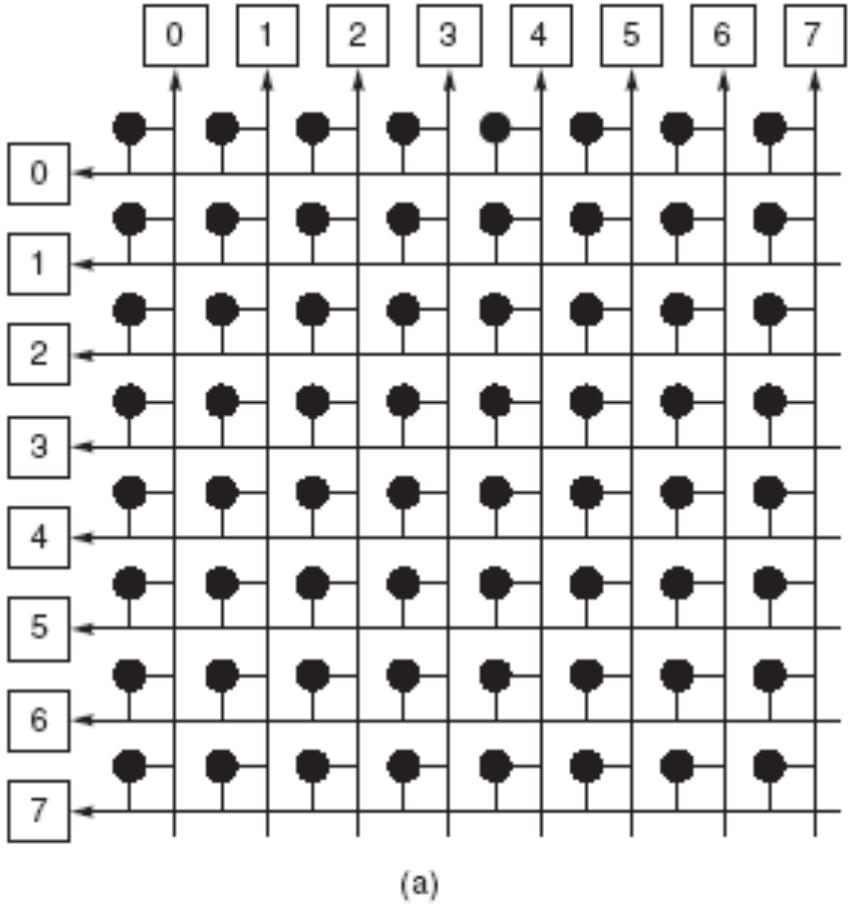


9.4.2 交叉开关网络

1. 单级开关网络

- 交叉点开关能在对偶（源、目的）之间形成动态连接，同时实现多个对偶之间的无阻塞连接。
- 带宽和互连特性最好。
- 一个n×n的交叉开关网络，可以无阻塞地实现n!种置换。
- 对一个n×n的交叉开关网络来说，需要n^2套交叉点开关以及大量的连线。
  - **当n很大时，交叉开关网络所需要的硬件数量非常巨大。**



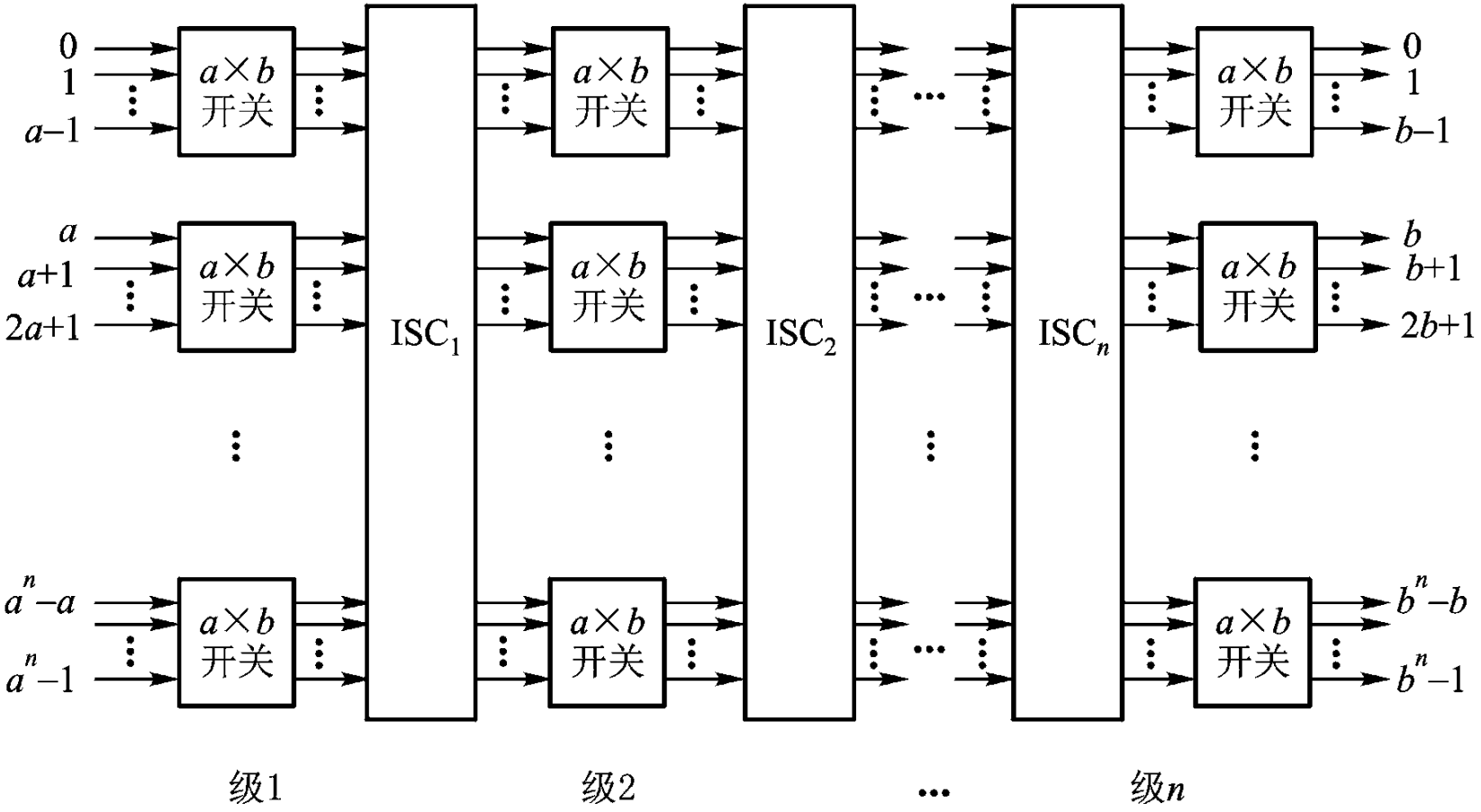


(a)

9.4.3 多级互连网络

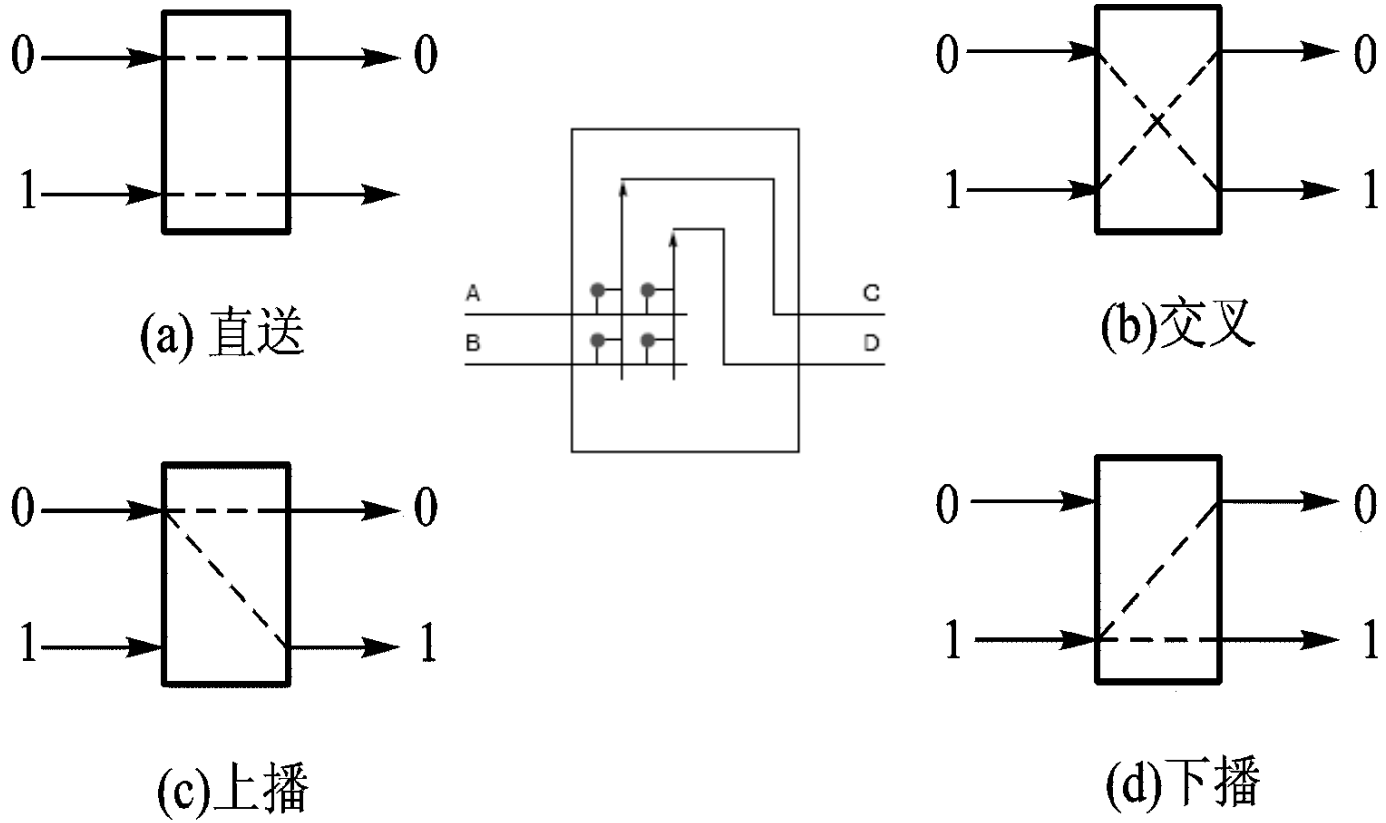
1. A common way of addressing the crossbar scaling problem consists of splitting the large crossbar switch into several stages of smaller switches interconnected

   - MIMD和SIMD计算机采用多级互连网络MIN（Multistage Interconnection Network）
   - 一种通用的多级互连网络
     - 由a×b开关模块和级间连接构成的通用多级互连网络结构
     - 每一级都用了多个a×b开关
       - a个输入和b个输出
       - 在理论上，a和b不一定相等，然而实际上a和b经常选为2的整数幂，即$a=b=2^k$，$k\geqslant 1$。
     - 相邻各级开关之间通过ISC(InterStage Connection)实现级间连接


0
1
a–1
a
a+1
2a+1
a^n–a
a^n–1
a×b 开关
ISC1
ISC2
ISCn
0
1
b–1
b
b+1
2b+1
b^n–b
b^n–1
级1
级2
…
级n

- 最简单的开关模块：2×2开关

2×2开关的4种连接方式



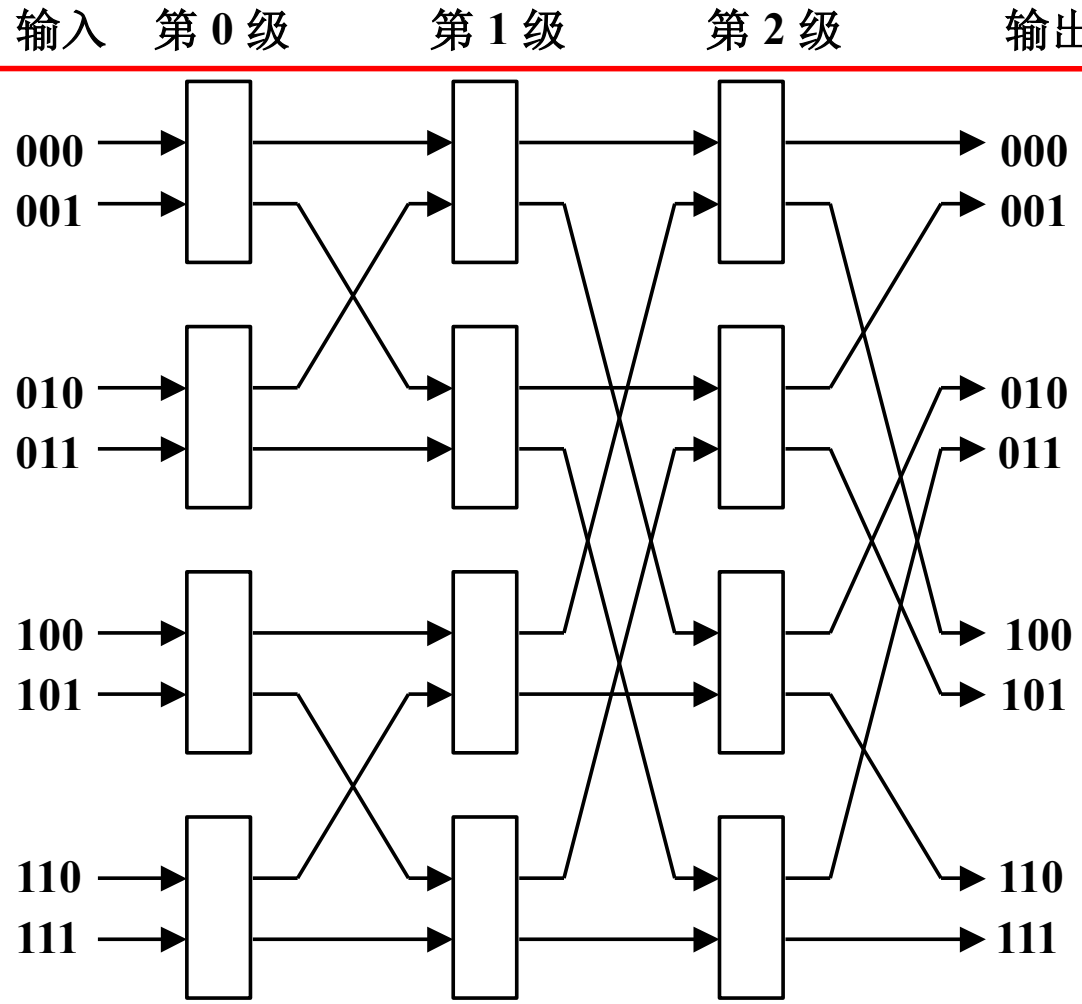
- 各种多级互连网络的区别在于所用开关模块、控制方式和级间互连模式的不同。
  - 控制方式：对各个开关模块进行控制的方式。
    - 级控制：每一级的所有开关只用一个控制信号控制，只能同时处于同一种状态。
    - 单元控制：每一个开关都有一个独立的控制信号，可各自处于不同的状态。
    - 部分级控制：第i级的所有开关分别用i＋1个信号控制，0≤i≤n－1，n为级数。
  - 常用的级间互连模式：

均匀洗牌、蝶式、多路洗牌、立方体连接等（perfect shuffle, butterfly, multiway shuffle, cube connection）

2. 多级立方体网络

- 多级立方体网络包括STARAN网络(1972)和间接二进制n方体网络等。
- 一个N输入的多级立方体网络有**$\log_2 N$**级（为什么？），每级用N/2个2×2开关模块，共需要$\log_2 N \times N/2$个开关。

级序：各级编号是0，1，……，n-1，即按升序排列。
功能：第i级实现$Cube_i$，根据控制信号决定是否“交换”。例如当0级所有开关为“交换”状态时，所有通过的数据被执行$Cube_0$交换，依此类推。

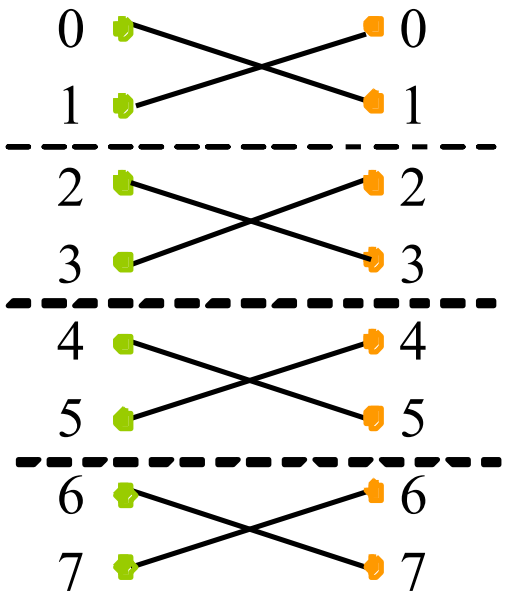


多级立方体网络画法 1

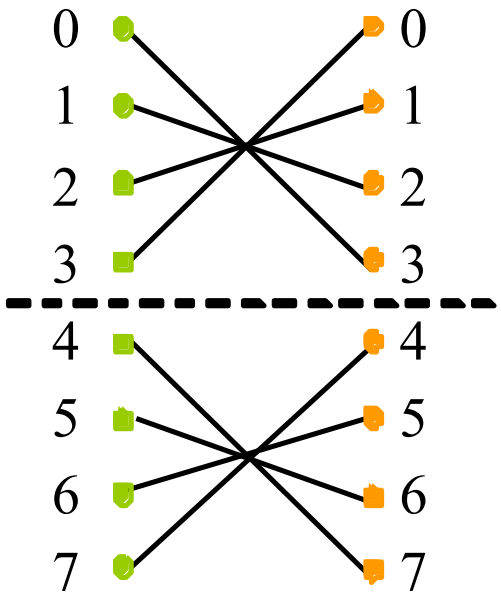
构造：第i级将输入端号仅第i位不同的数据组合在一个开关上。最后1级之后再设置1个“逆均匀洗牌”，使各输出端顺序排列。

- STARAN网络采用级控制和部分级控制。
  - 采用级控制时，所实现的是交换功能；
  - 采用部分级控制时，则能实现移数功能。
- 间接二进制n方体网络则采用单元控制。
  - 具有更大的灵活性。
- 交换
  - 将有序的一组元素头尾对称地进行交换。

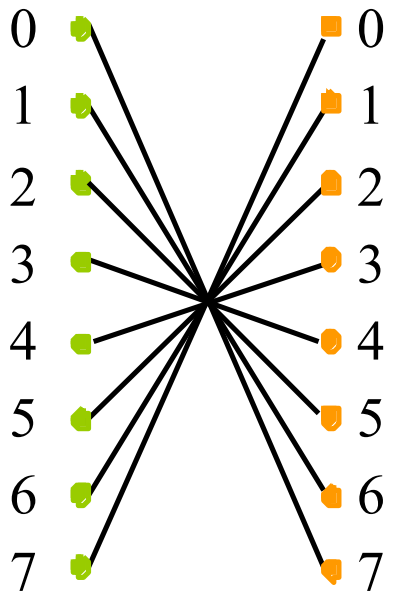
例如：对于由8个元素构成的组，各种基本交换的图形：



(a) 4 组 2 元交换



(b) 2 组 4 元交换



(c) 1 组 8 元交换

8个元素的基本交换图形

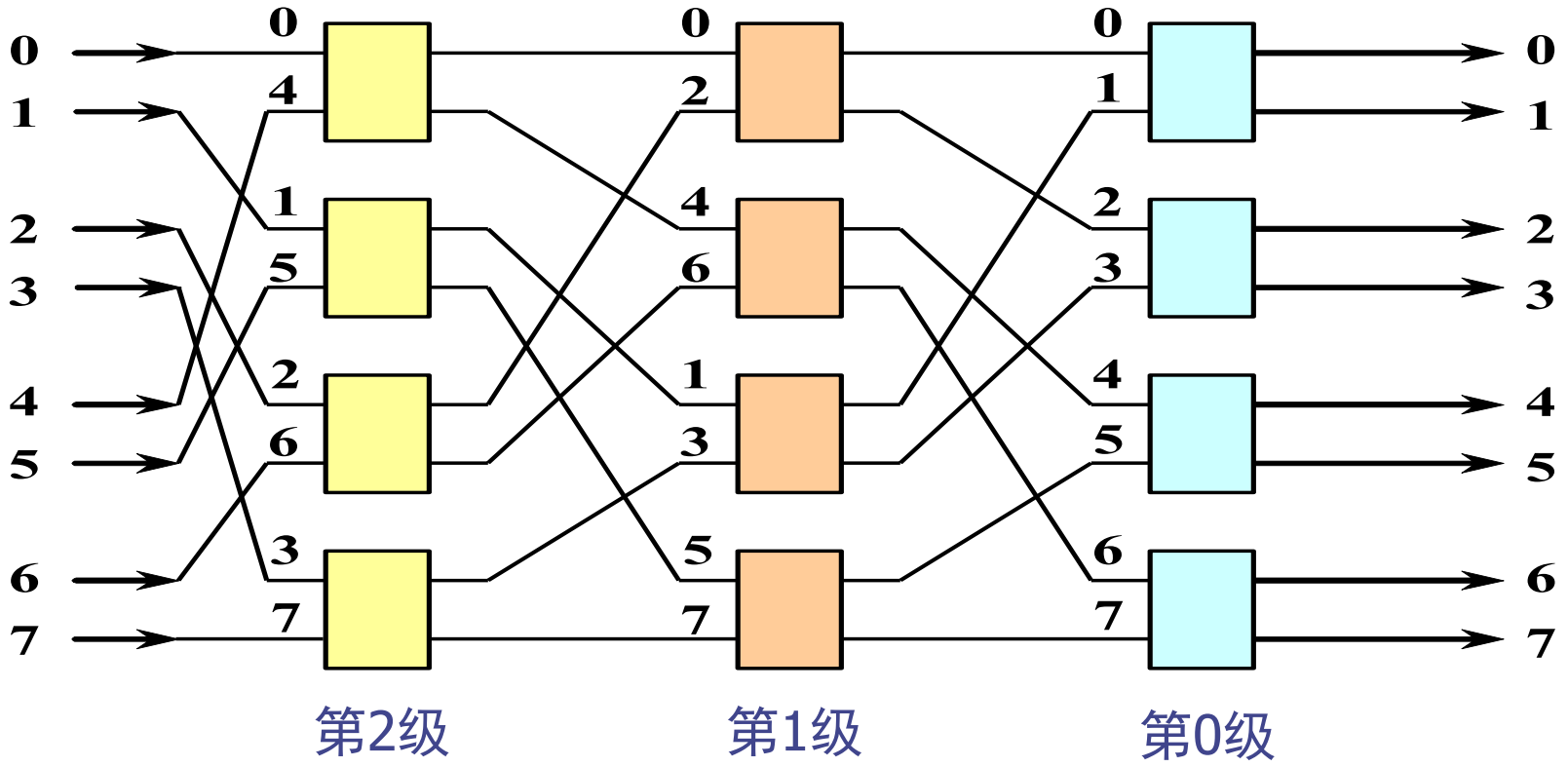
级控制信号 $k_2k_1k_0$	连接的输出端号序列（入端号序列：01234567）	实现的分组交换	实现的互连函数
000	0 1 2 3 4 5 6 7	恒等	I
001	1 0 3 2 5 4 7 6	4组2元交换	$Cube_0$
010	2 3 0 1 6 7 4 5	4组2元交换＋ 2组4元交换	$Cube_1$
011	3 2 1 0 7 6 5 4	2组4元交换	$Cube_0$＋$Cube_1$
100	4 5 6 7 0 1 2 3	2组4元交换＋ 1组8元交换	$Cube_2$
101	5 4 7 6 1 0 3 2	4组2元交换＋ 2组4元交换＋ 1组8元交换	$Cube_0$＋$Cube_2$
110	6 7 4 5 2 3 0 1	4组2元交换＋ 1组8元交换	$Cube_1$＋$Cube_2$
111	7 6 5 4 3 2 1 0	1组8元交换	$Cube_0$＋$Cube_1$＋ $Cube_2$

- 当STARAN网络用作移数网络时，采用部分级控制，控制信号的分组和控制结果。

部分级控制信号						连接的输出端号序列（入端号序列：01234567）	所实现的移数功能
第0级	第1级		第2级				
A B C D	E G	F H	I	J	K L		
1	1	0	1	0	0	1 2 3 4 5 6 7 0	移1 mod 8
0	1	1	1	1	0	2 3 4 5 6 7 0 1	移2 mod 8
0	0	1	1	1	1	4 5 6 7 0 1 2 3	移4 mod 8
1	1	0	0	0	0	1 2 3 0 5 6 7 4	移1 mod 4
0	1	1	0	0	0	2 3 0 1 6 7 4 5	移2 mod 4
1	0	0	0	0	0	1 0 3 2 5 4 7 6	移1 mod 2
0	0	0	0	0	0	0 1 2 3 4 5 6 7	不移 全等

3. Omega网络

- 级序：各级编号是n-1，……，0，即按降序排列。
  - 每级由4个4功能的2×2开关构成
  - 级间互连采用**均匀洗牌连接**方式



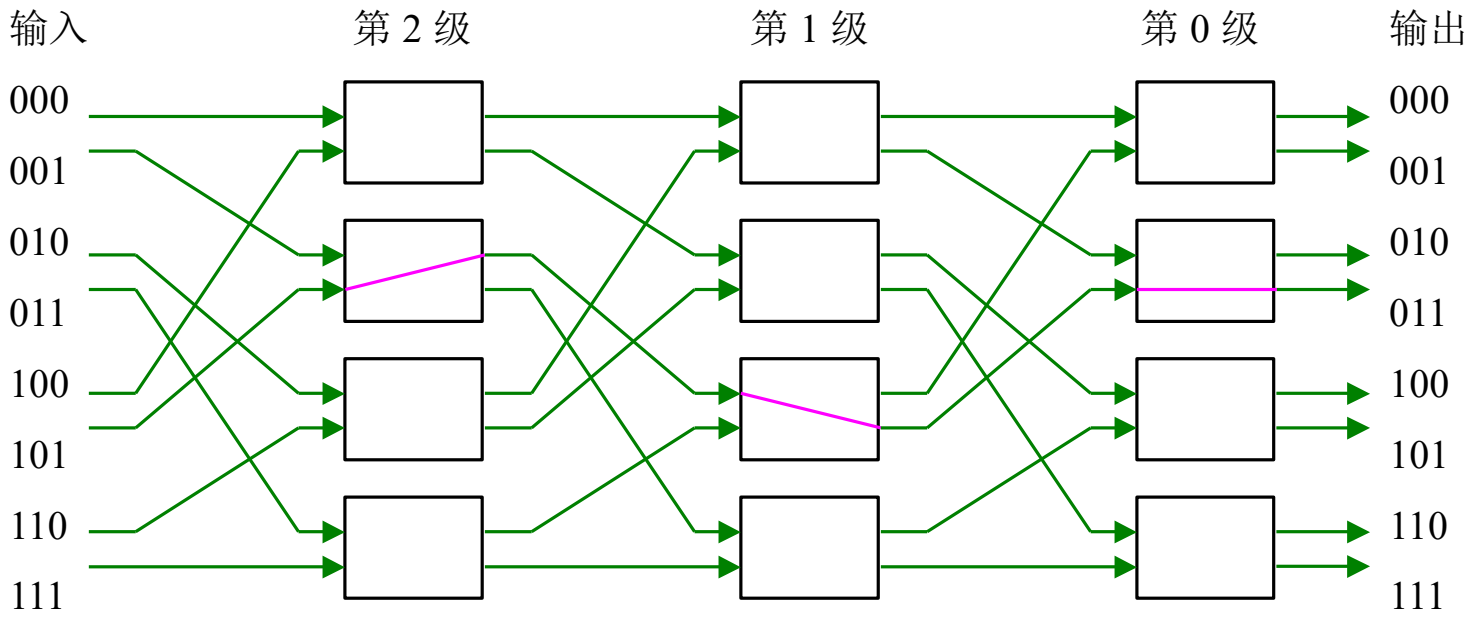
多级混洗—交换网络寻径算法（路由算法）1

目的：根据给定的输入/输出对应关系，确定各开关的状态。

名称：异或法

操作：将任一个输入端号与它要到达的输出端号作异或运算，其结果的bit_i位控制数据到达的第i级开关，“0”表示“直连”，“1”表示“交换”。

例如给定传输101B→011B，二者异或结果为110B，于是从101B号输入端开始，把它遇到的第2级开关置为“交换”，第1级开关置为“交换”，第0级开关置为“直连”。如下图红线所示。（可简记为“第i级判第i位”）

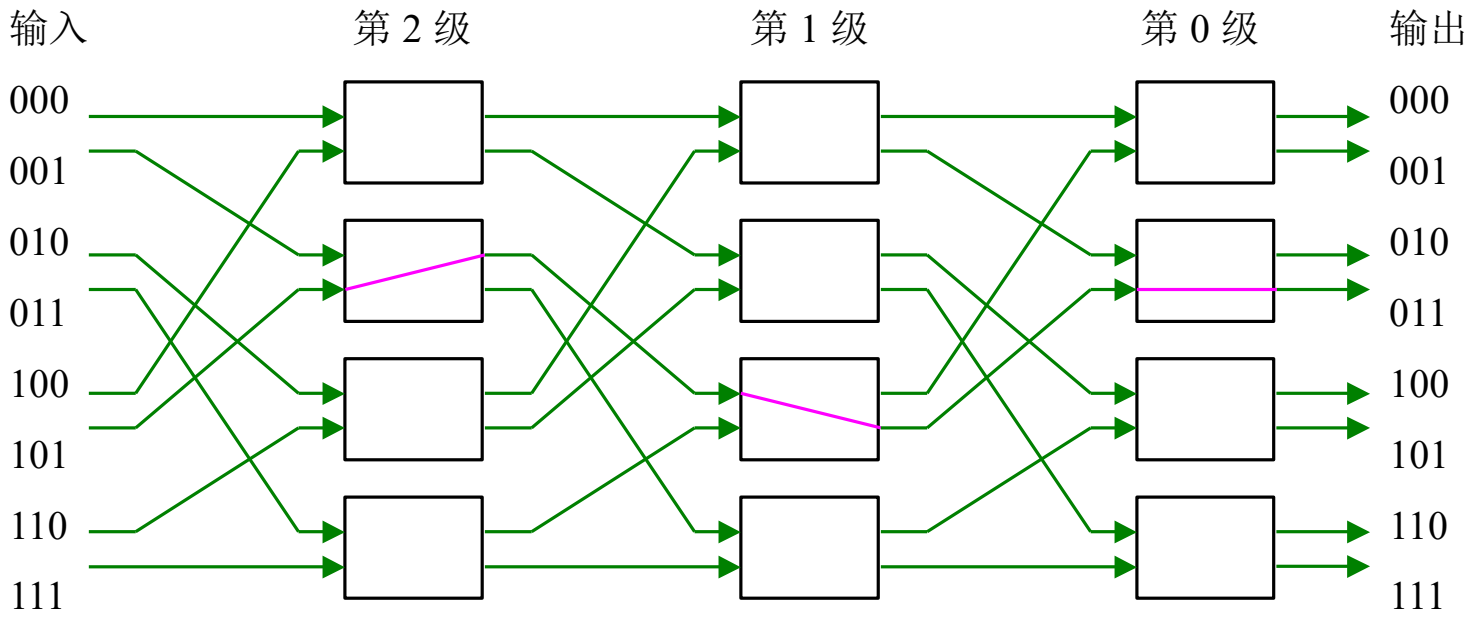


多级混洗—交换网络寻径算法（路由算法）2

名称：末址法

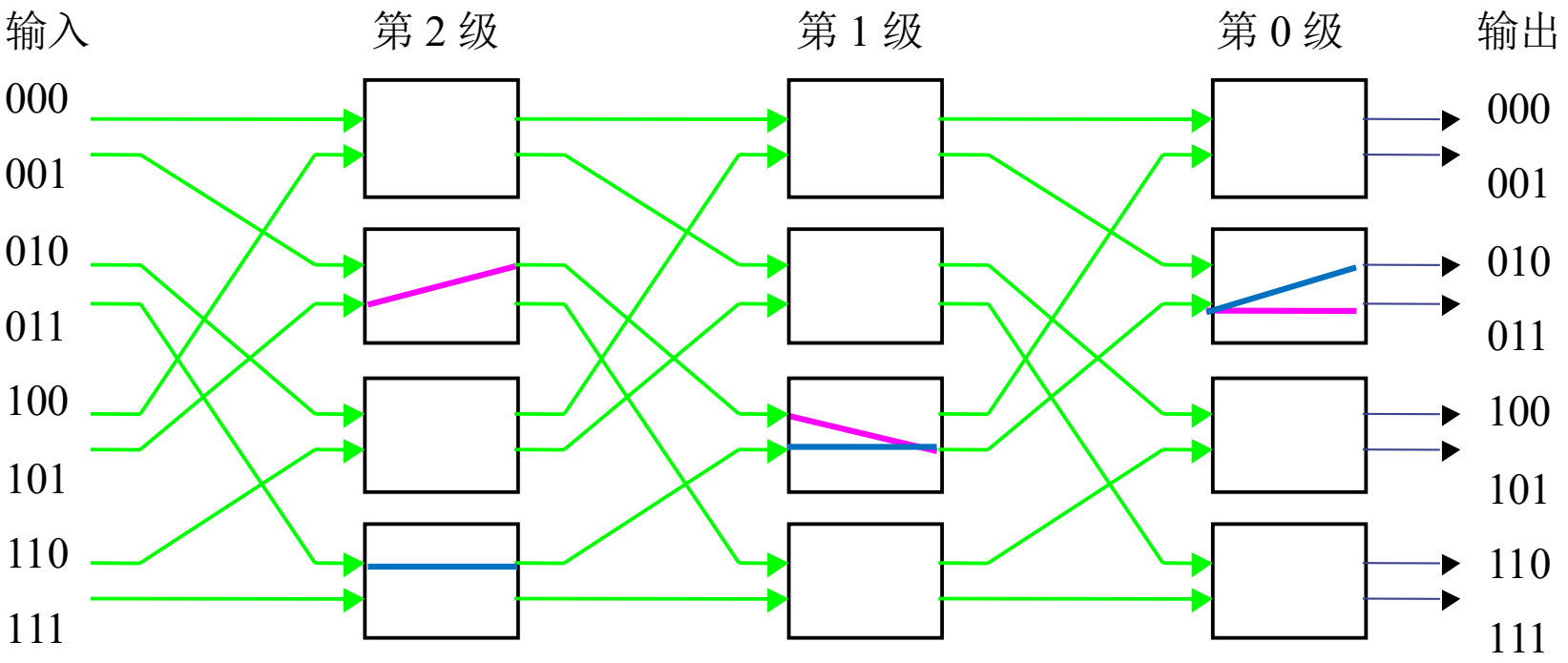
操作：输出端号二进制形式的bit_i位控制数据到达的第i级开关，“0”表示从该级开关的上出口输出，“1”表示从该级开关的下出口输出。

例如给定传输101B→011B，输出端号的二进制形式依次为0→1→1，于是从101B号输入端开始，依次选择出口为“上”→“下”→“下”。如下图红线所示。（同样简记为“第i级判第i位”）



传输101B→011B，二者异或结果为110B，路径如下图红线所示。

传输011B→010B，二者异或结果为001B，路径如下图蓝线所示。



- 一个N输入的Omega网络
  - 有$\log_2 N$级，每级用$N/2$个2×2开关模块，共需要$N\log_2 N/2$个开关。
  - 每个开关模块均采用单元控制方式。
  - 不同的开关状态组合可实现各种置换、广播或从输入到输出的其它连接。
- N=8的多级立方体互连网络的另一种画法

第9章 互连网络——An overview of network

该网络由混洗函数（shuffle）与交换函数（exchange即$Cube_0$）定义，或者说它的互连函数族只有这两个成员。shuffle也可记作σ（Σ的小写）。

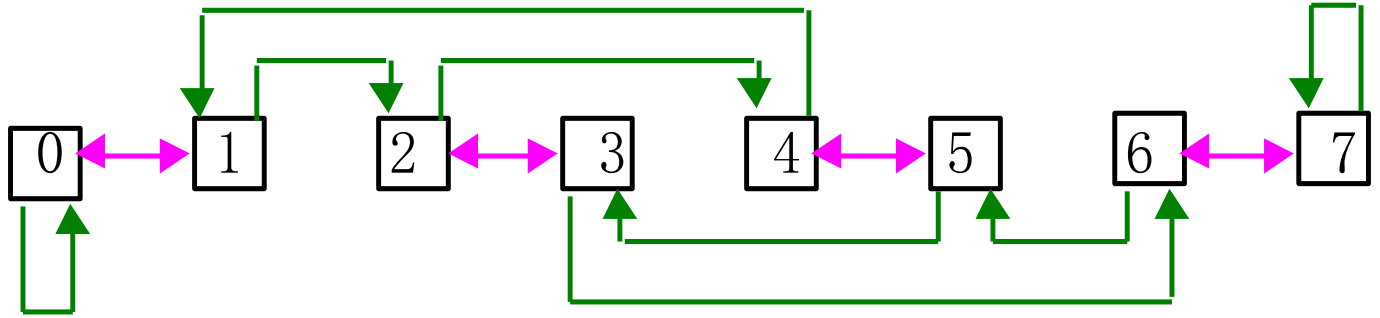
- 混洗函数定义：

$$shuffle(x) = \begin{cases} 2x \mod (N-1), & 当x < N-1 \\ N-1, & 当x = N-1 \end{cases}$$

例如：当N=8时，shuffle（0）= 0，shuffle（1）= 2，shuffle（7）= 7。

n=3的混洗网络拓扑形状如下图绿线所示，可以看出它不是一个连通图，所以还需要增加一个交换函数（图中红线所示），才能构成完整的单级混洗—交换网络。

单级混洗—交换网络的直径是2n-1。



Destination-tag routing

- In destination-tag routing, switch settings are determined solely by the message destination. The most significant bit of the destination address is used to select the output of the switch in the first stage; if the most significant bit is 0, the upper output is selected, and if it is 1, the lower output is selected. The next-most significant bit of the destination address is used to select the output of the switch in the next stage, and so on until the final output has been selected.
- For example, if a message's destination is PE 001, the switch settings are: straight, straight, crossed. If a message's destination is PE 101, the switch settings are: crossed, straight, crossed. These switch settings hold regardless of the PE sending the message.

http://en.wikipedia.org/wiki/Omega_network

XOR-tag routing

- In XOR-tag routing, switch settings are based on (source PE) XOR (destination PE). This XOR-tag contains 1s in the bit positions that must be swapped and 0s in the bit positions that both source and destination have in common. The most significant bit of the XOR-tag is used to select the setting of the switch in the first stage; if the most significant bit is 0, the switch is set to pass-through, and if it is 1, the switch is crossed. The next-most significant bit of the tag is used to set the switch in the next stage, and so on until the final output has been selected.
- For example, if PE 001 wishes to send a message to PE 010, the XOR-tag will be 011 and the appropriate switch settings are: A2 straight, B3 crossed, C2 crossed.

符号约定：起点$x = x_{n-1}\cdots\cdots x_0$，终点$y = y_{n-1}\cdots\cdots y_0$。

1. 单级立方体网

二者间路径由地址逻辑差$y_i \oplus x_i$决定。$y_i \oplus x_i$=“1”代表$Cube_i$维需要走一步，各维先后顺序可任意安排，“1”的个数即是总步数。

2. 单级混洗-交换网

二者间路径也由地址逻辑差$y_i \oplus x_i$决定。如果$y_i \oplus x_i$=“1”，表明x_i须先经n-i步shuffle到最低位，1步$Cube_0$求反，再经i步shuffle回到原位变成y_i。如有多位“1”，可以合并shuffle，具体算法须灵活设计。

3. 单级PM2I网

二者间路径由地址算术差y-x mod N决定。结果中“1”所在的位代表从x到y正向路径需要走一步的维，“1”的个数代表从x到y正向路径需要走的总步数，但此路径不一定是从x到y的最短路径，需要用某种方法降解优化，具体算法须灵活设计。

PM2I网的直径是 $\lceil n/2 \rceil$。

证明：

设起点x = $x_{n-1}.....x_0$，终点y = $y_{n-1}.....y_0$，反映二者间正向路径的地址差z = y−x = $z_{n-1}.....z_0$。z中“1”的个数代表从x到y正向行进步数。

如果 z_{n-1}、...、z_0 中有2个以上相邻的“1”，按照 Booth 算法 $2^{i+k-1}+...+2^i = 2^{i+k}-2^i$，即可把连续前进k步替换为进、退各1步（第i+k维进1步、第i维退1步），变为更短的路径。

直径是距离的最大值，而距离是一对结点间的最短路径。对PM2I网来说，包含有最多“1”且不能用 Booth 算法降解的地址差就是一个正向直径。

根据 Booth 算法的规律，仅当z由一位“1”、一位“0”交替，或者两位“1”、两位“0”交替构成时，该算法无法将其替换成含有更少的“1”的形式。

结论：任何一对最远结点的地址差中必有一半的“0”和一半的“1”，故PM2I网的直径是 $\lceil n/2 \rceil$。