8.2 对称式共享存储器系统结构

- 多个处理器共享一个存储器。
- 当处理机规模较小时，这种计算机十分经济。
- 近些年，能在一个单独的芯片上实现2～8个处理器核。

  例如：Sun公司 2006年 T1 8核的多处理器
- 支持对共享数据和私有数据的Cache缓存

  私有数据供一个单独的处理器使用，而共享数据则是供多个处理器使用。
- 共享数据进入Cache产生了一个新的问题

  Cache的一致性问题

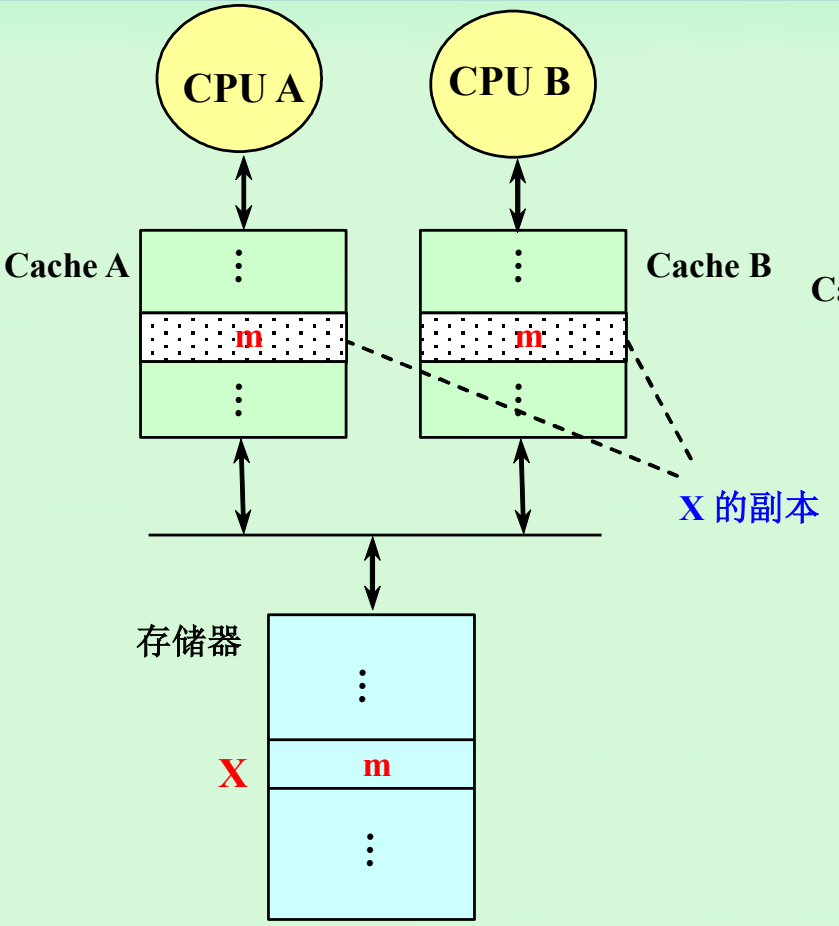
8.2.1 多处理机Cache一致性

1. 多处理机的Cache一致性问题

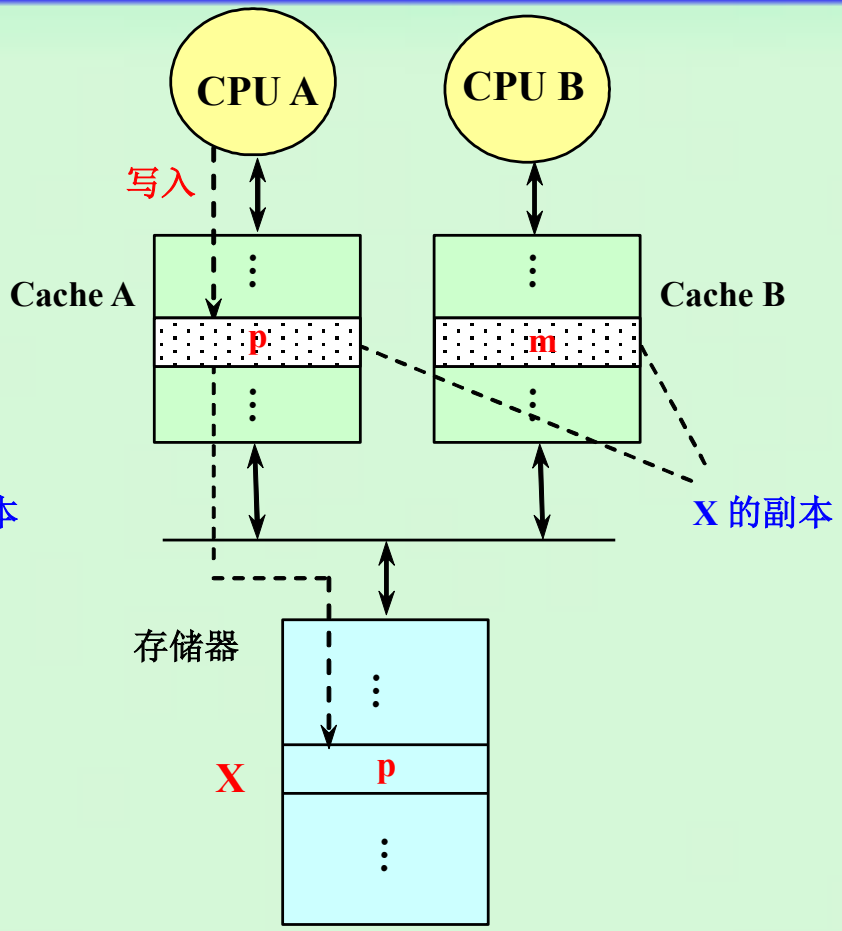
- 允许共享数据进入Cache，就可能出现多个处理器的Cache中都有同一存储块的副本，
- 当其中某个处理器对其Cache中的数据进行修改后，就会使得其Cache中的数据与其他Cache中的数据不一致。

例 由两个处理器（A和B）读写引起的Cache一致性问题

动画



（a） CPU A 写入前



（b） CPU A 将 p 写入 X，p≠m

2. 存储器的一致性

如果对某个数据项的任何读操作均可得到其最新写入的值，则认为这个存储系统是一致的。

- 存储系统行为的两个不同方面
  - What：读操作得到的是什么值
  - When：什么时候才能将已写入的值返回给读操作
- 需要满足以下条件
  - 处理器P对单元X进行一次写之后又对单元X进行读，读和写之间没有其他处理器对单元X进行写，则P读到的值总是前面写进去的值。

- 处理器P对单元X进行写之后，另一处理器Q对单元X进行读，读和写之间无其他写，则Q读到的值应为P写进去的值。
- 对同一单元的写是串行化的，即任意两个处理器对同一单元的两次写，从各个处理器的角度看来顺序都是相同的。（写串行化 ）

➢ 在后面的讨论中，我们假设：

- 直到所有的处理器均看到了写的结果，这个写操作才算完成；
- 处理器的任何访存均不能改变写的顺序。就是说，允许处理器对读进行重排序，但必须以程序规定的顺序进行写。

8.2.2 实现一致性的基本方案

在一致的多处理机中，Cache提供两种功能：

- 共享数据的迁移

  减少了对远程共享数据的访问延迟，也减少了对共享存储器带宽的要求。

- 共享数据的复制

  不仅减少了访问共享数据的延迟，也减少了访问共享数据所产生的冲突。

一般情况下，小规模多处理机是采用硬件的方法来实现Cache的一致性。

1. Cache一致性协议

在多个处理器中用来维护一致性的协议。

- 关键：跟踪记录共享数据块的状态
- 两类协议（采用不同的技术跟踪共享数据的状态）
  - 目录式协议（directory）

    物理存储器中数据块的共享状态被保存在一个称为目录的地方。
  - 监听式协议（snooping）
    - 每个Cache除了包含物理存储器中块的数据拷贝之外，也保存着各个块的共享状态信息。

- Cache通常连在共享存储器的总线上，当某个Cache需要访问存储器时，它会把请求放到总线上广播出去，其他各个Cache控制器通过监听总线（它们一直在监听）来判断它们是否有总线上请求的数据块。如果有，就进行相应的操作。

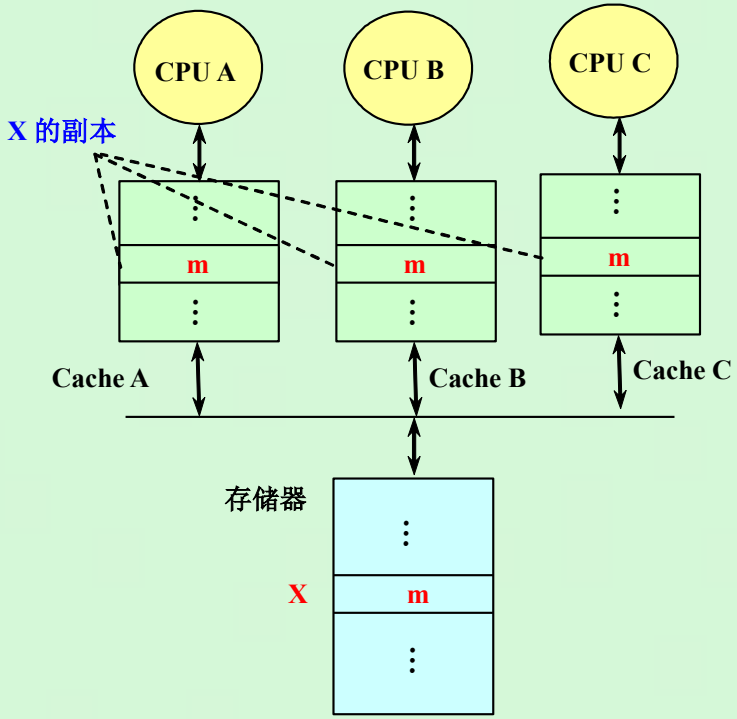
2. 有两种写协议来保证Cache一致性中的写操作。

➢ 写作废协议

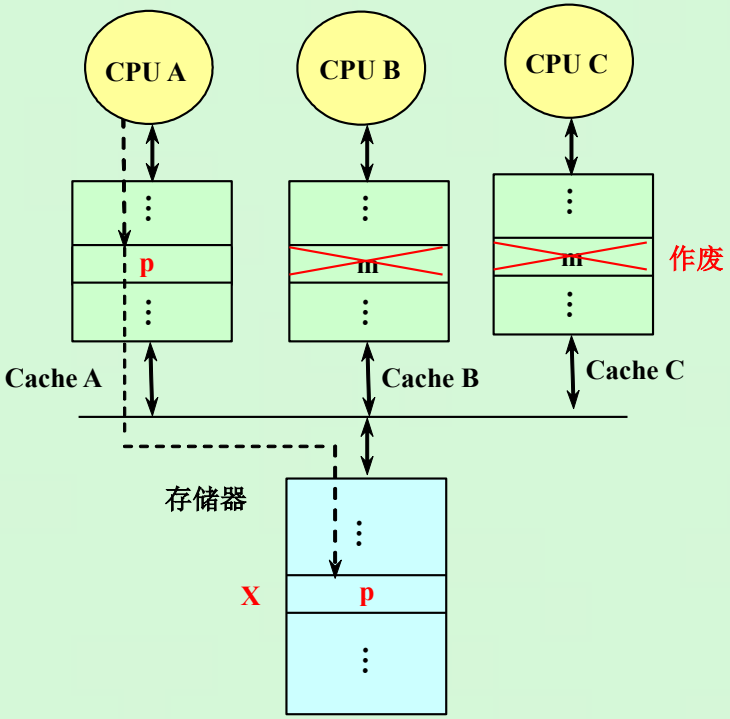
在处理器对某个数据项进行写入之前，保证它拥有对该数据项的唯一的访问权。(作废其他的副本)

例 监听总线、写作废协议举例（采用写直达法） 动画

初始状态：CPU A、CPU B、CPU C都有X的副本。在CPU A要对X进行写入时，需先作废CPU B和CPU C中的副本，然后再将p写入Cache A中的副本，同时用该数据更新主存单元X。



（a） CPU A 写入前



（b） CPU A 将 p 写入 X 后，作废其他 Cache 中的副本

➢ 写更新协议

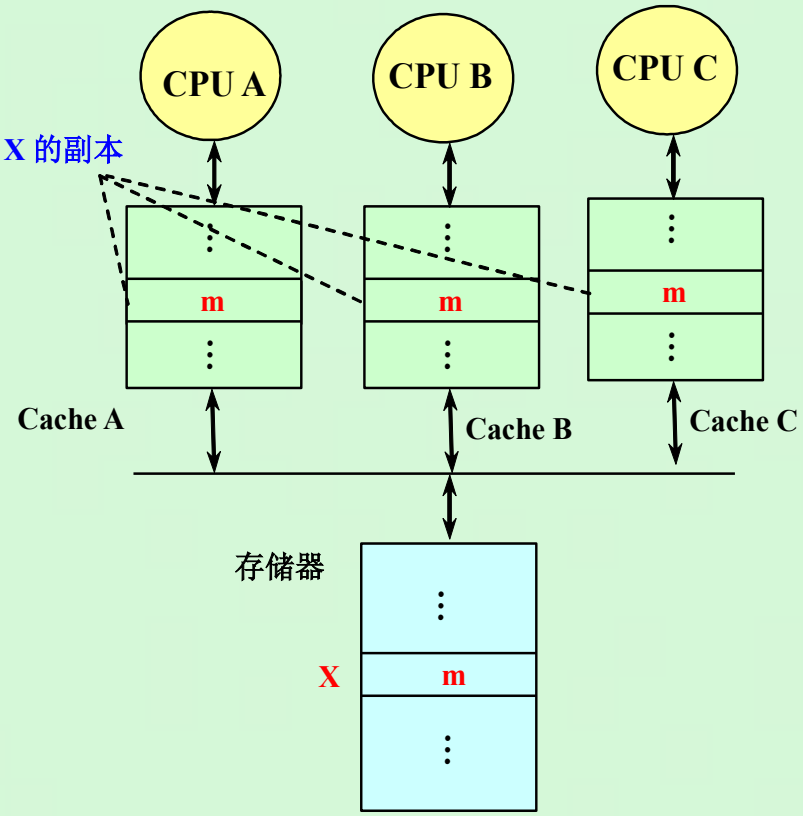
当一个处理器对某数据项进行写入时，通过广播使其他Cache中所有对应于该数据项的副本进行更新。

例 监听总线、写更新协议举例（采用写直达法）

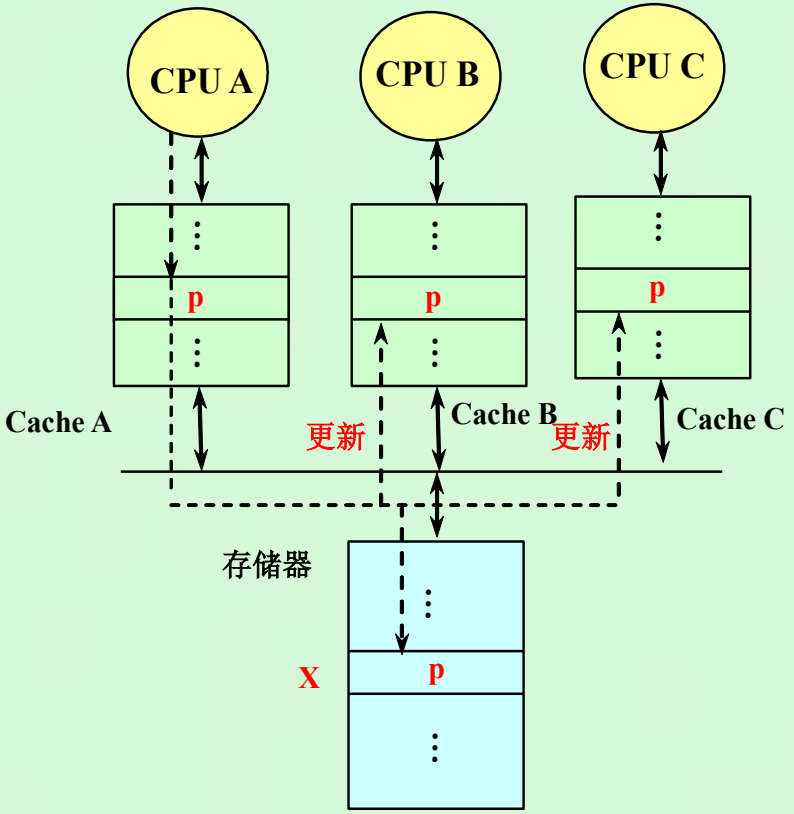
假设：3个Cache都有X的副本。

当CPU A将数据p写入Cache A中的副本时，将p广播给所有的Cache，这些Cache用p更新其中的副本。

由于这里是采用写直达法，所以CPU A还要将p写入存储器中的X。如果采用写回法，则不需要写入存储器。



（a） CPU A 写入前



（b） CPU A 将 p 写入 X 后，更新其他 Cache 中的副本

- 写更新和写作废协议性能上的差别主要来自：
  - 在对同一个数据进行多次写操作而中间无读操作的情况下，写更新协议需进行多次写广播操作，而写作废协议只需一次作废操作。
  - 在对同一Cache块的多个字进行写操作的情况下，写更新协议对于每一个写操作都要进行一次广播，而写作废协议仅在对该块的第一次写时进行作废操作即可。

    写作废是针对Cache块进行操作，而写更新则是针对字（或字节）进行。
  - 考虑从一个处理器A进行写操作后到另一个处理器B能读到该写入数据之间的延迟时间。

    写更新协议的延迟时间较小。

8.2.3 监听协议的实现

1. 监听协议的基本实现技术

- 实现监听协议的关键有3个方面
  - 处理器之间通过一个可以实现广播的互连机制相连。

    通常采用的是总线。
  - 当一个处理器的Cache响应本地CPU的访问时，如果它涉及全局操作，其Cache控制器就要在获得总线的控制权后，在总线上发出相应的消息。
  - 所有处理器都一直在监听总线，它们检测总线上的地址在它们的Cache中是否有副本。若有，则响应该消息，并进行相应的操作 。

- 写操作的串行化：由总线实现

（获取总线控制权的顺序性）

2. Cache发送到总线上的消息主要有以下两种：
   - **RdMiss**——读不命中
   - **WtMiss**——写不命中
- 需要通过总线找到相应数据块的最新副本，然后调入本地Cache中。
  - 写直达Cache：因为所有写入的数据都同时被写回主存，所以从主存中总可以取到其最新值。
  - 对于写回Cache，得到数据的最新值会困难一些，因为最新值可能在某个Cache中，也可能在主存中。

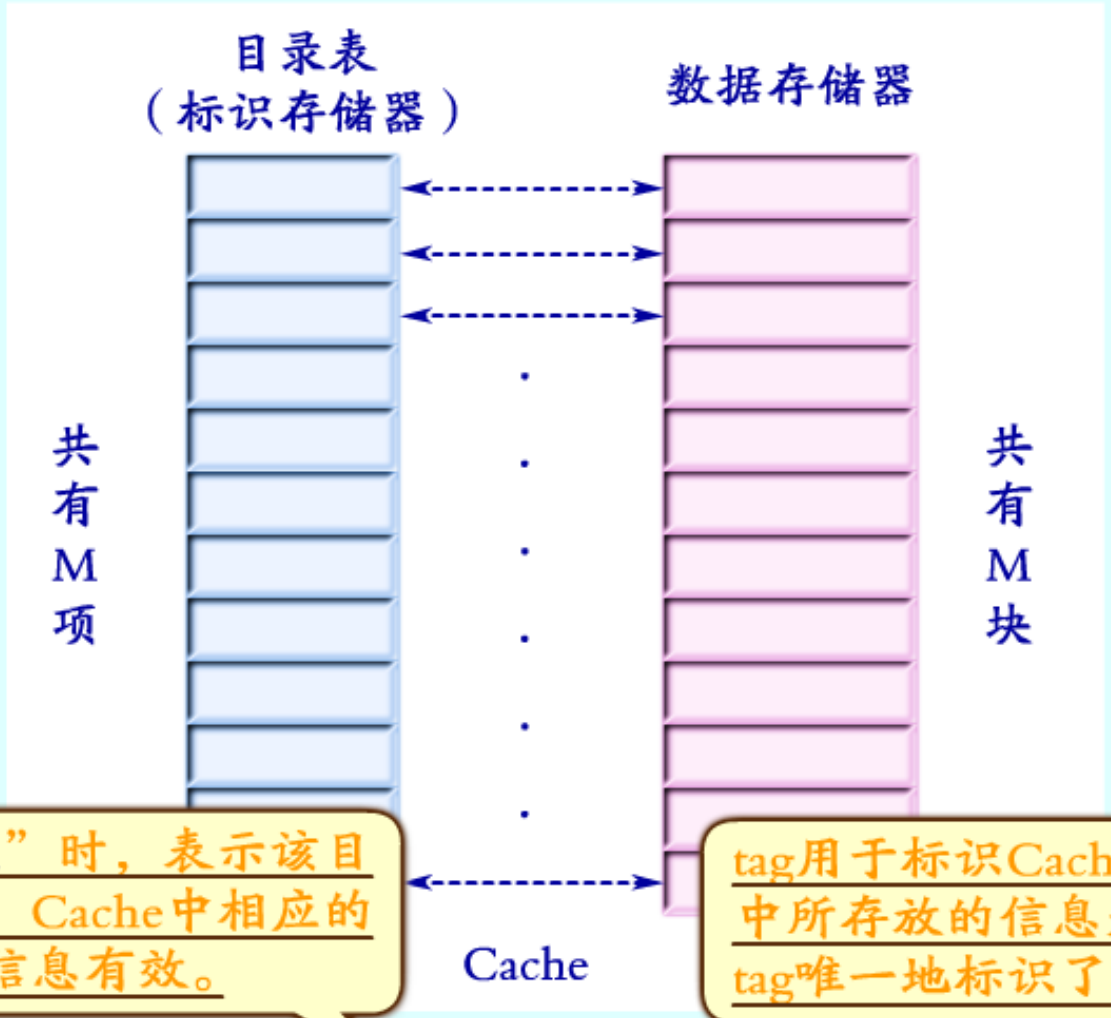
（后面的讨论中，只考虑写回法Cache）

- 有的监听协议还增设了一条Invalidate（作废）消息，用来通知其他各处理器作废其Cache中相应的副本。
  - 与WtMiss的区别：Invalidate不引起调块
- Cache的标识（tag）可直接用来实现监听。
- 作废一个块只需将其有效位置为0，即为无效。
- 给每个Cache块增设一个共享位
  - 为“1”：该块是被多个处理器所共享
  - 为“0”：仅被某个处理器所独占

块的拥有者：拥有该数据块的唯一副本的处理器。

有效位和共享位的组合可以是：00,01,10,11。其中：00和01 代表 该块无效。10代表该块为独占（已修改），11代表该块为共享。所以说，一个块有3中不同的状态：无效，已修改，共享

Cache目录表的结构



当该位为“1”时，表示该目录表项有效，Cache中相应的块所包含的信息有效。

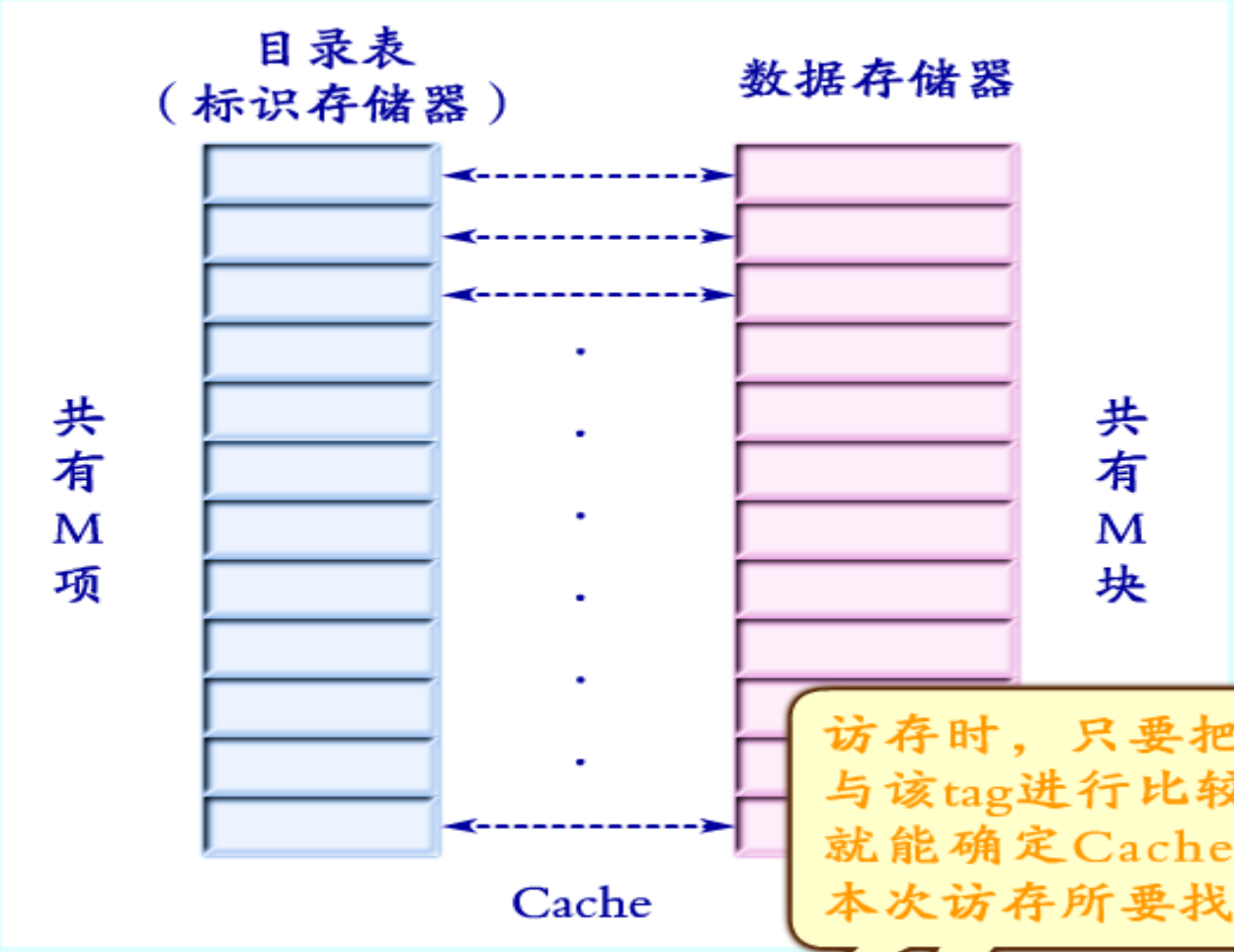
tag用于标识Cache中相应的块位置中所存放的信息是哪个主存块的。tag唯一地标识了一个主存块。

目录表项：

有效位	标识tag

共享位

Cache目录表的结构



访存时，只要把访存地址中的高位与该tag进行比较，判断是否相等，就能确定Cache中相应的块是否是本次访存所要找的块。

目录表项：

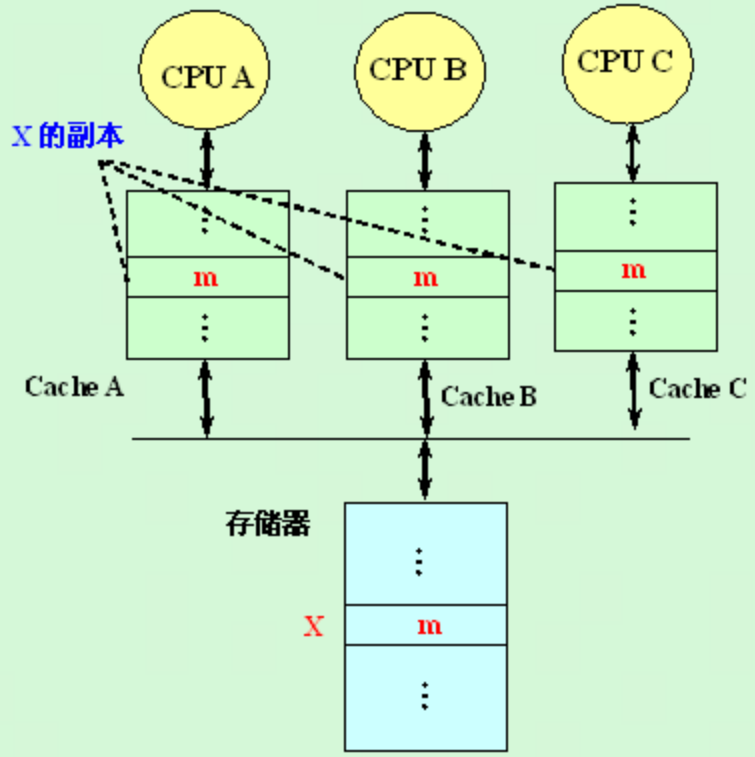
共享位

访存地址：

3. 监听协议举例

- 在每个结点内嵌入一个有限状态控制器。
  - 该控制器根据来自处理器或总线的请求以及Cache块的状态，做出相应的响应。
- 每个数据块的状态取以下3种状态中的一种：
  - 无效（简称I）：Cache中该块的内容为无效。
  - 共享（简称S）：该块可能处于共享状态。
    - 在多个处理器中都有副本。这些副本都相同，且与存储器中相应的块相同。
  - 已修改（简称M）：该块已经被修改过，并且还没写入存储器。（块中的内容是最新的，系统中唯一的最新副本）

有效位和共享位的组合可以是：00,01,10,11。其中：00和01代表该块无效。10代表该块为独占（已修改），11代表该块为共享。所以说，一个块有3中不同的状态：无效，已修改，共享



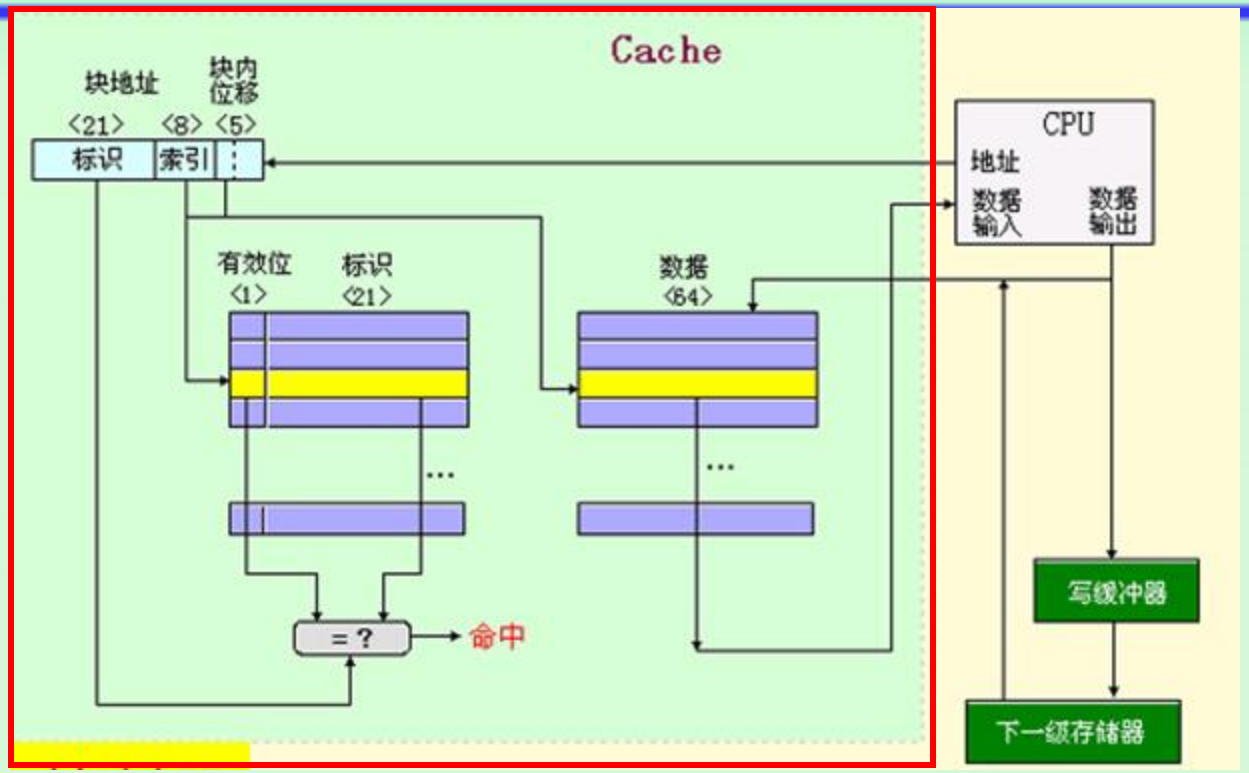
有效位和共享位的组合可以是：00,01,10,11。其中：00和01 代表该块无效。10代表该块为独占（已修改），11代表该块为共享。所以说，一个块有3中不同的状态：无效，已修改，共享

状态	有效位和共享位
无效，	00 或者 01
已修改，	10
共享	11

有限状态控制器

目录表项： 有效位 | 标识tag | 共享位

监听协议



有限状态控制器

目录表项：

共享位

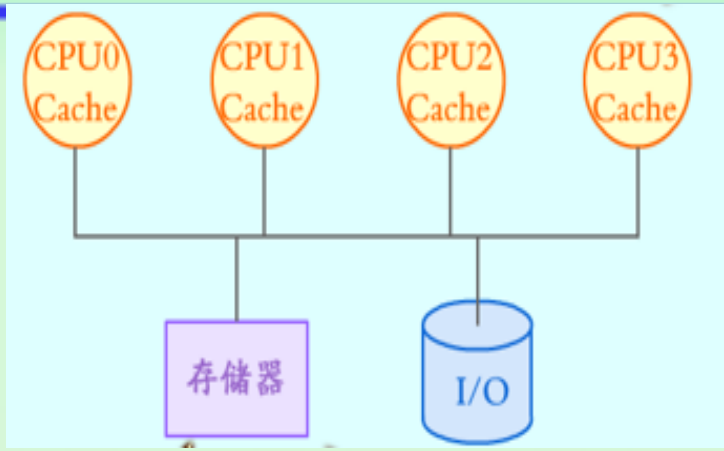
状态

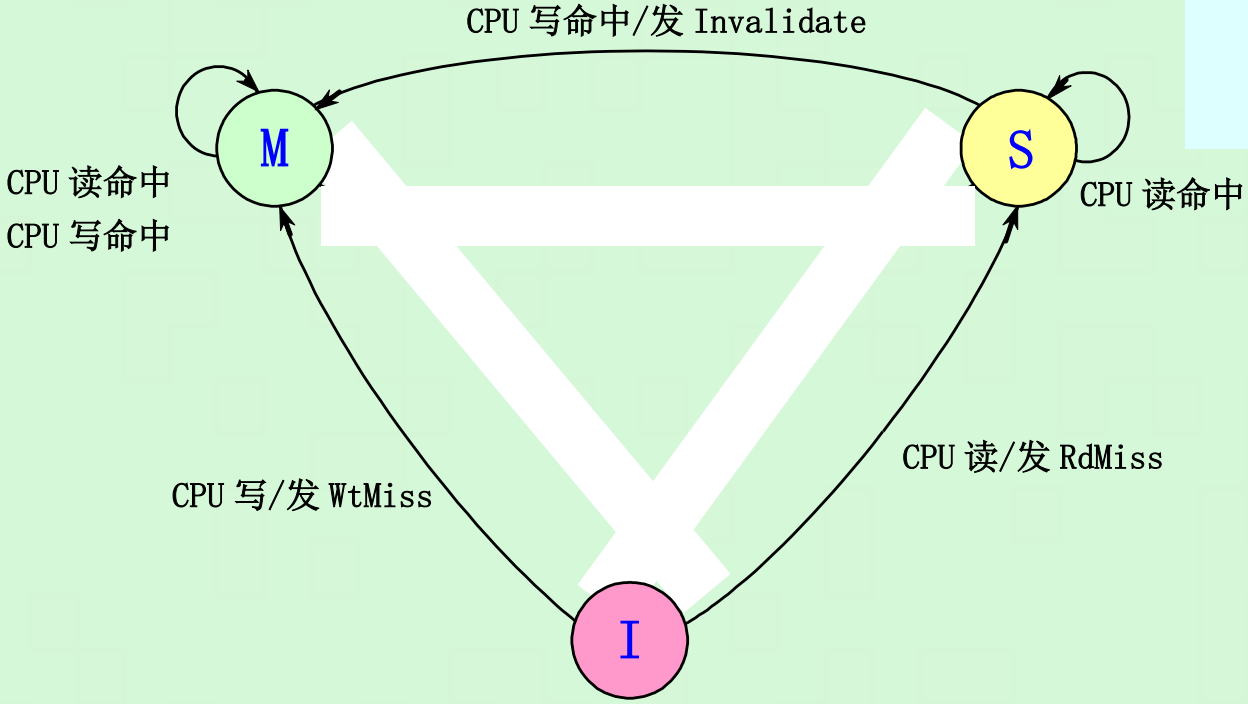
有效位和共享位

无效，	00 或者 01
已修改，	10
共享	11

下面来讨论在各种情况下监听协议所进行的操作。

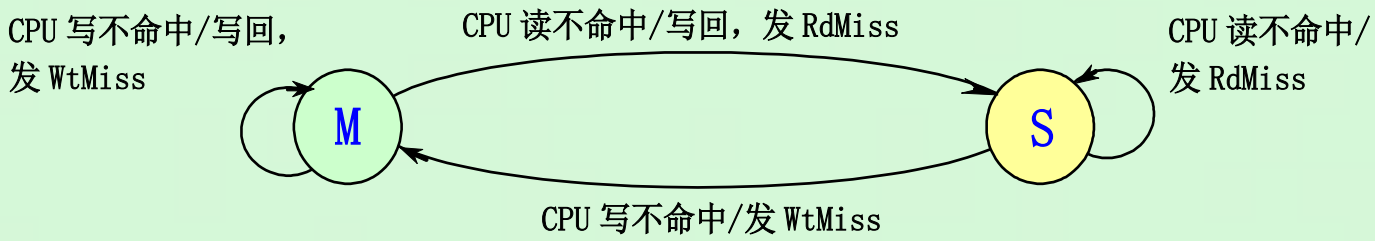
- 响应来自处理器的请求
  - 不发生替换的情况



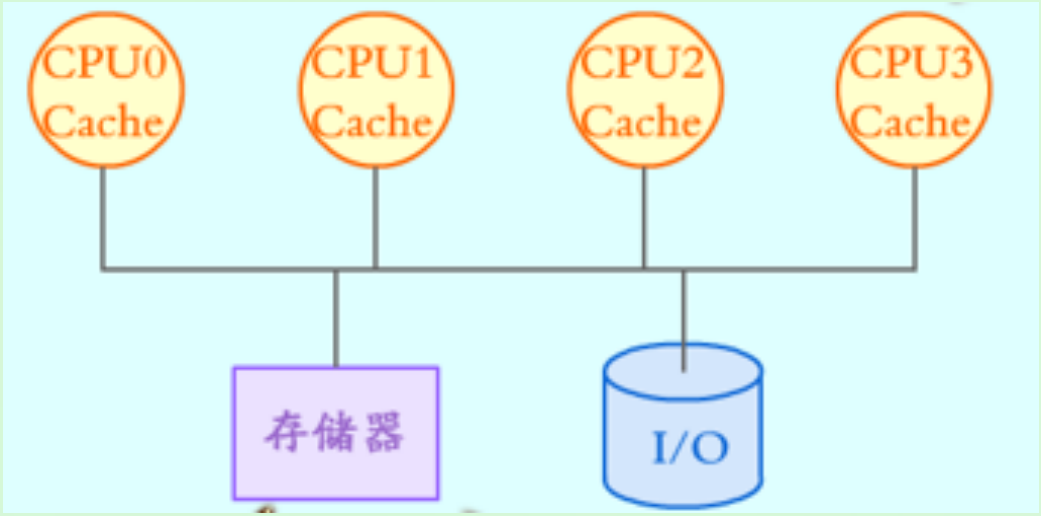


写作废协议中（采用写回法），Cache块的状态转换图

- 发生替换的情况

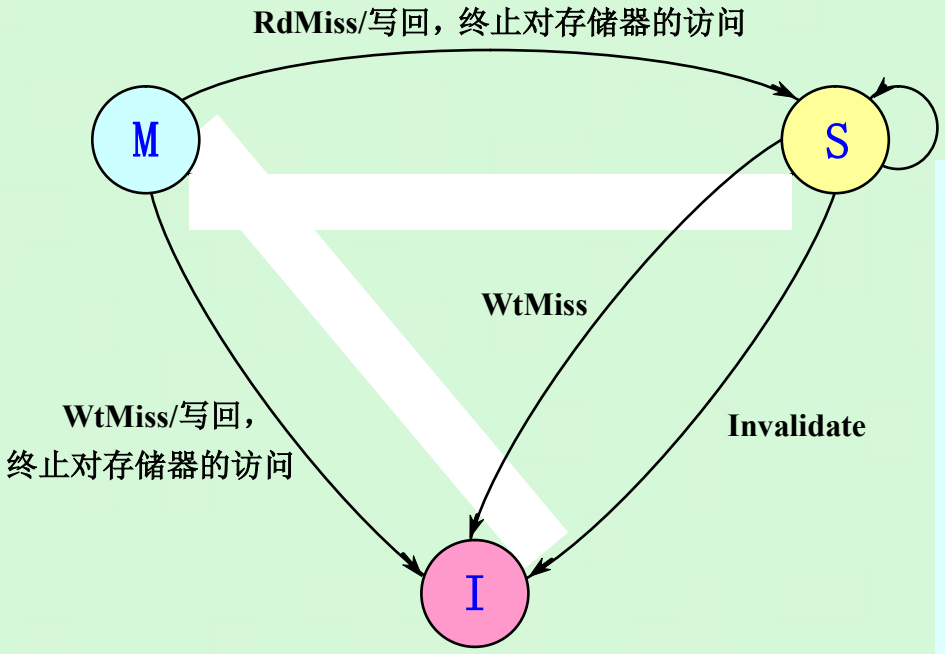


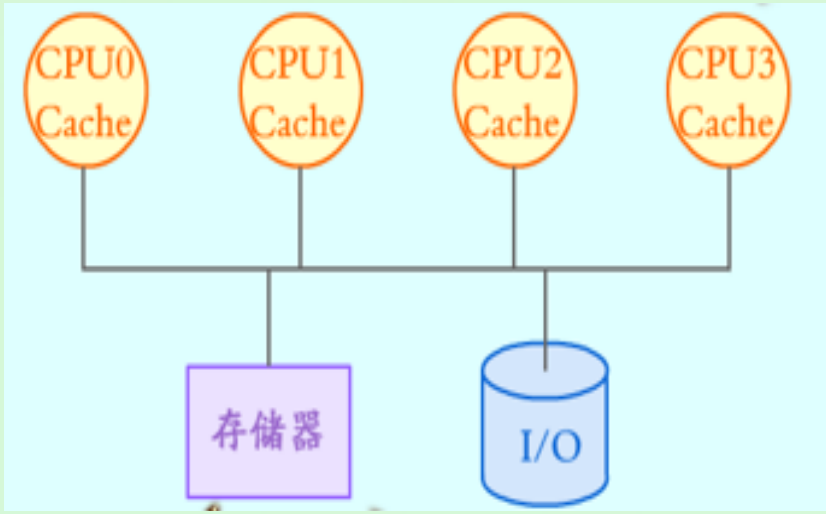
写作废协议中（采用写回法），Cache块的状态转换图



➢ 响应来自总线的请求

- 每个处理器都在监视总线上的消息和地址，当发现有与总线上的地址相匹配的Cache块时，就要根据该块的状态以及总线上的消息，进行相应的处理。





写作废协议中（采用写回法），Cache块的状态转换图

8.3 分布式共享存储器系统结构

8.3.1 目录协议的基本思想

- 广播和监听的机制使得监听一致性协议的可扩放性很差。
- 寻找替代监听协议的一致性协议。

（采用目录协议）

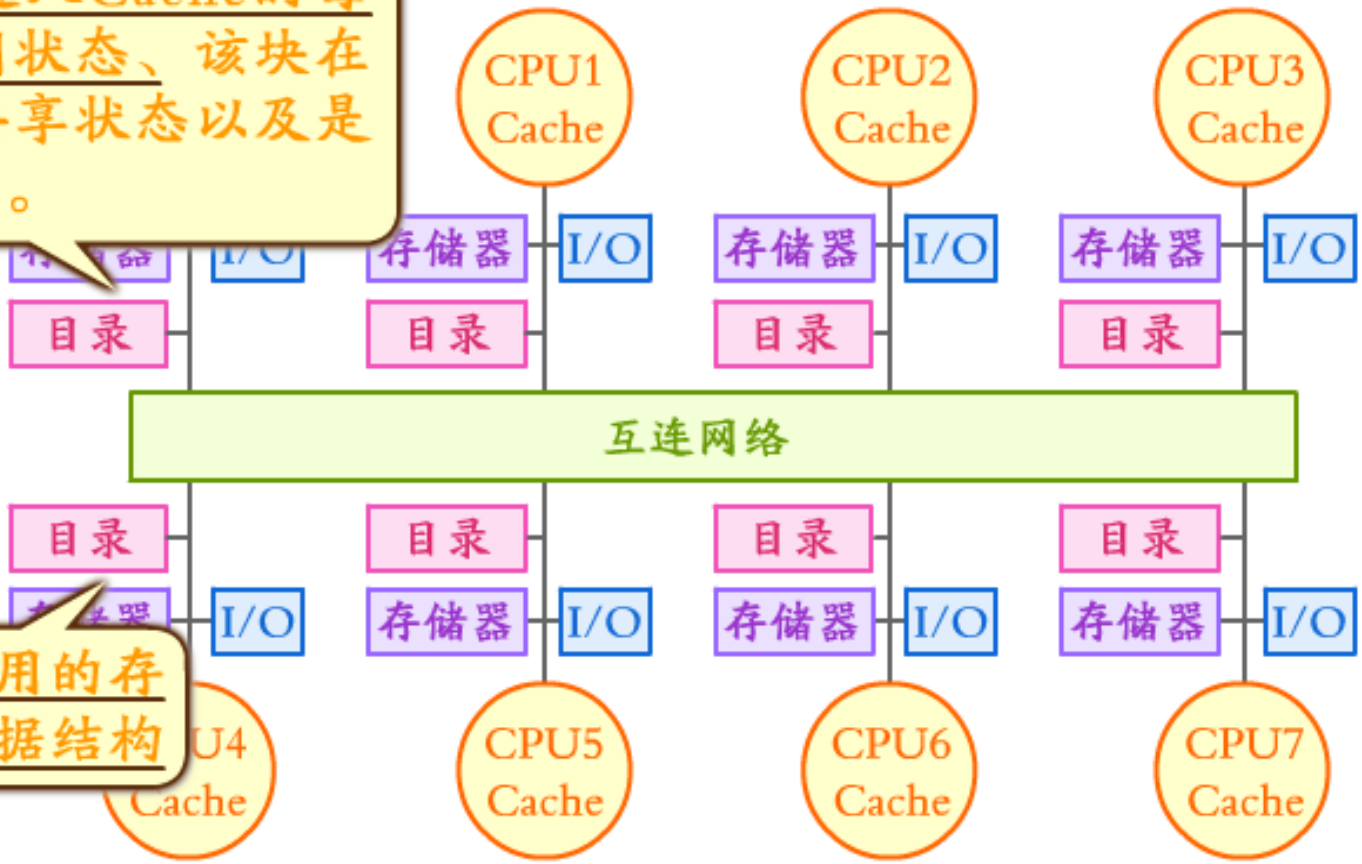
1. 目录协议

- 目录：一种集中的数据结构。对于存储器中的每一个可以调入Cache的数据块，在目录中设置一条目录项，用于记录该块的状态以及哪些Cache中有副本等相关信息。

目录协议

它记录着可以进入Cache的每个数据块的访问状态、该块在各个处理器的共享状态以及是否修改过等信息。

目录是用一种专用的存储器所记录的数据结构

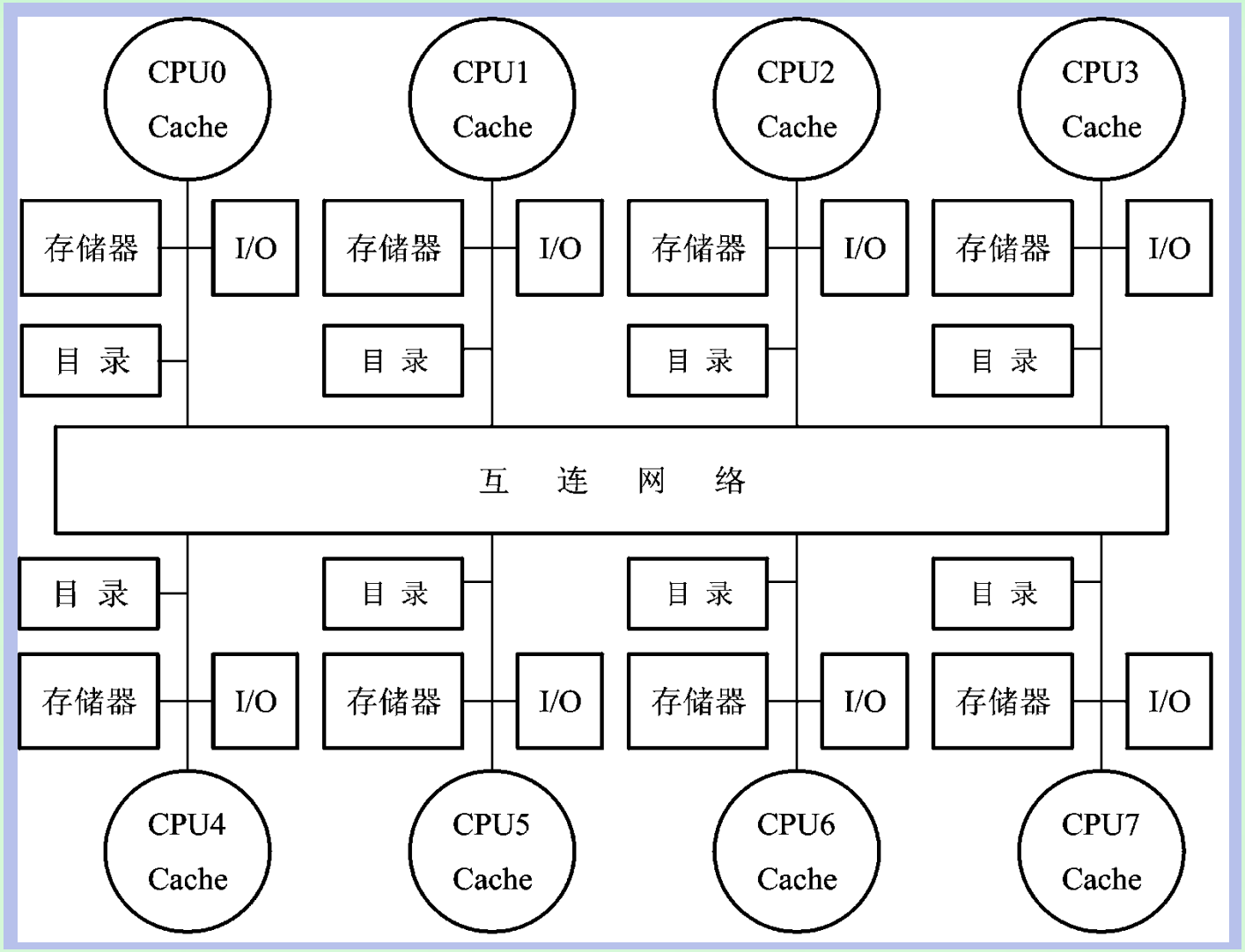


- 特点：

  对于任何一个数据块，都可以快速地在唯一的一个位置中找到相关的信息。这使一致性协议避免了广播操作。

- 位向量：记录哪些Cache中有副本。
  - 每一位对应于一个处理器。
  - 长度与处理器的个数成正比。
  - 由位向量指定的处理机的集合称为共享集S。
- 分布式目录
  - 目录与存储器一起分布到各结点中，从而对于不同目录内容的访问可以在不同的结点进行。

- 对每个结点增加目录后的分布式存储器多处理机



- 目录法最简单的实现方案：对于存储器中每一块都在目录中设置一项。目录中的信息量与M×N成正比。

  其中：

  - M：存储器中存储块的总数量
  - N：处理器的个数
  - 由于M=K×N，K是每个处理机中存储块的数量，所以如果K保持不变，则目录中的信息量就与N^2成正比。

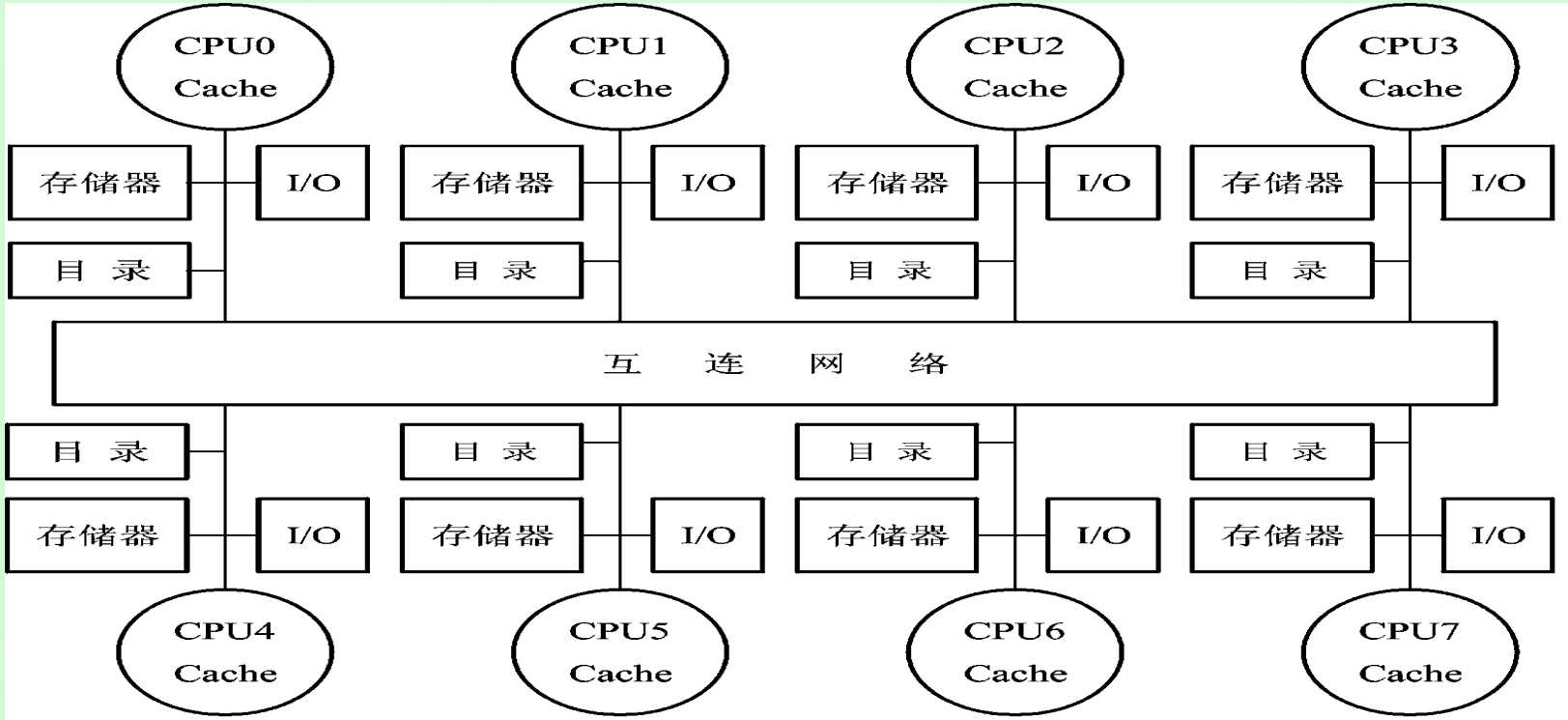
2. 在目录协议中，存储块的状态有3种：

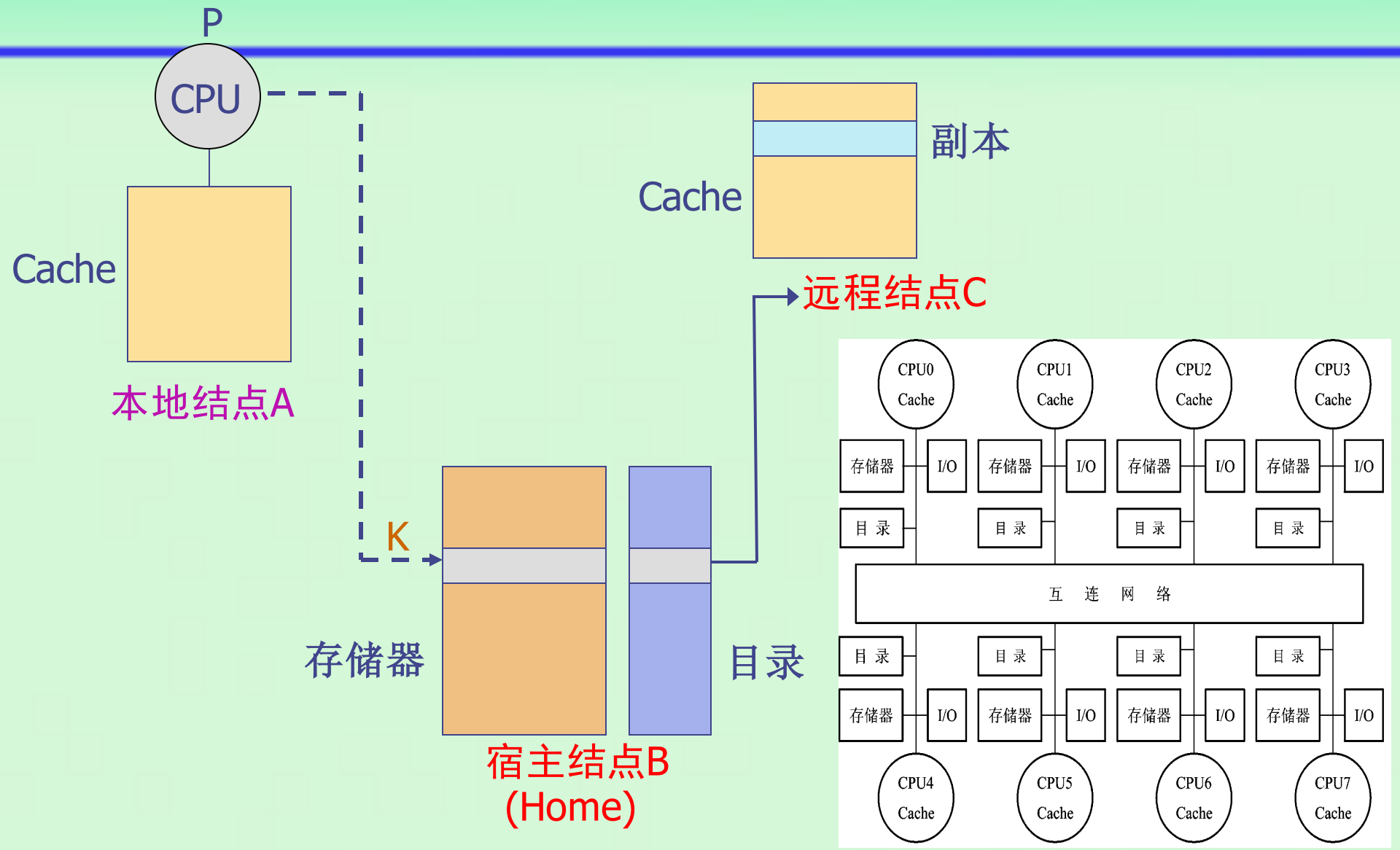
- 未缓冲：该块尚未被调入Cache。所有处理器的Cache中都没有这个块的副本。
- 共享：该块在一个或多个处理机上有这个块的副本，且这些副本与存储器中的该块相同。
- 独占：仅有一个处理机有这个块的副本，且该处理机已经对其进行了写操作，所以其内容是最新的，而存储器中该块的数据已过时。

  这个处理机称为该块的拥有者。

3. 本地结点、宿主结点以及远程结点的关系

- 本地结点：发出访问请求的结点
- 宿主结点：包含所访问的存储单元及其目录项的结点
- 远程结点可以和宿主结点是同一个结点，也可以不是同一个结点。





宿主结点：存放有对应地址的存储器块和目录项的结点

4. 在结点之间发送的消息

- 本地结点发给宿主结点（目录）的消息

说明：括号中的内容表示所带参数。

P：发出请求的处理机编号

K：所要访问的地址

- **RdMiss（P，K）**

处理机P读取地址为A的数据时不命中，请求宿主结点提供数据（块），并要求把P加入共享集。

- **WtMiss（P，K）**

处理机P对地址A进行写入时不命中，请求宿主结点提供数据，并使P成为所访问数据块的独占者。

- **Invalidate（K）**

  请求向所有拥有相应数据块副本（包含地址*K*）的远程Cache发Invalidate消息，作废这些副本。

➢ 宿主结点（目录）发送给远程结点的消息

- **Invalidate（K）**

  作废远程Cache中包含地址K的数据块。

- **Fetch（K）**

  从远程Cache中取出包含地址*K*的数据块，并将之送到宿主结点。把远程Cache中那个块的状态改为“共享”。

- **Fetch&Inv（K）**

- 从远程Cache中取出包含地址*K*的数据块，并将之送到宿主结点。然后作废远程Cache中的那个块。

➢ 宿主结点发送给本地结点的消息

DReply（D）

- D表示数据内容。
- 把从宿主存储器获得的数据返回给本地Cache。

➢ 远程结点发送给宿主结点的消息

WtBack（K，D）

- 把远程Cache中包含地址*K*的数据块写回到宿主结点中， 该消息是远程结点对宿主结点发来的“取数据”或“取/作废”消息的响应。

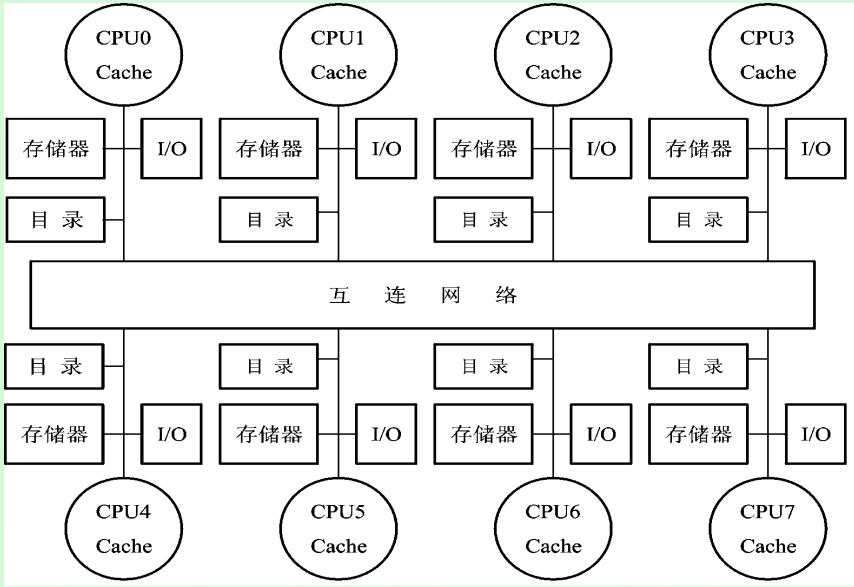
- 本地结点发送给被替换块的宿主结点的消息
  - **MdSharer（P，K）**

    用于当本地Cache中需要替换一个包含地址K的块、且该块未被修改过的情况。这个消息发给该块的宿主结点，请求它将P从共享集中删除。如果删除后共享集变为空集，则宿主结点还要将该块的状态改变为“未缓存”（U）。
  - **WtBack2（P，K，D）**

    用于当本地Cache中需要替换一个包含地址K的块、且该块已被修改过的情况。这个消息发给该块的宿主结点，完成两步操作：①把该块写回；②进行与MdSharer相同的操作。

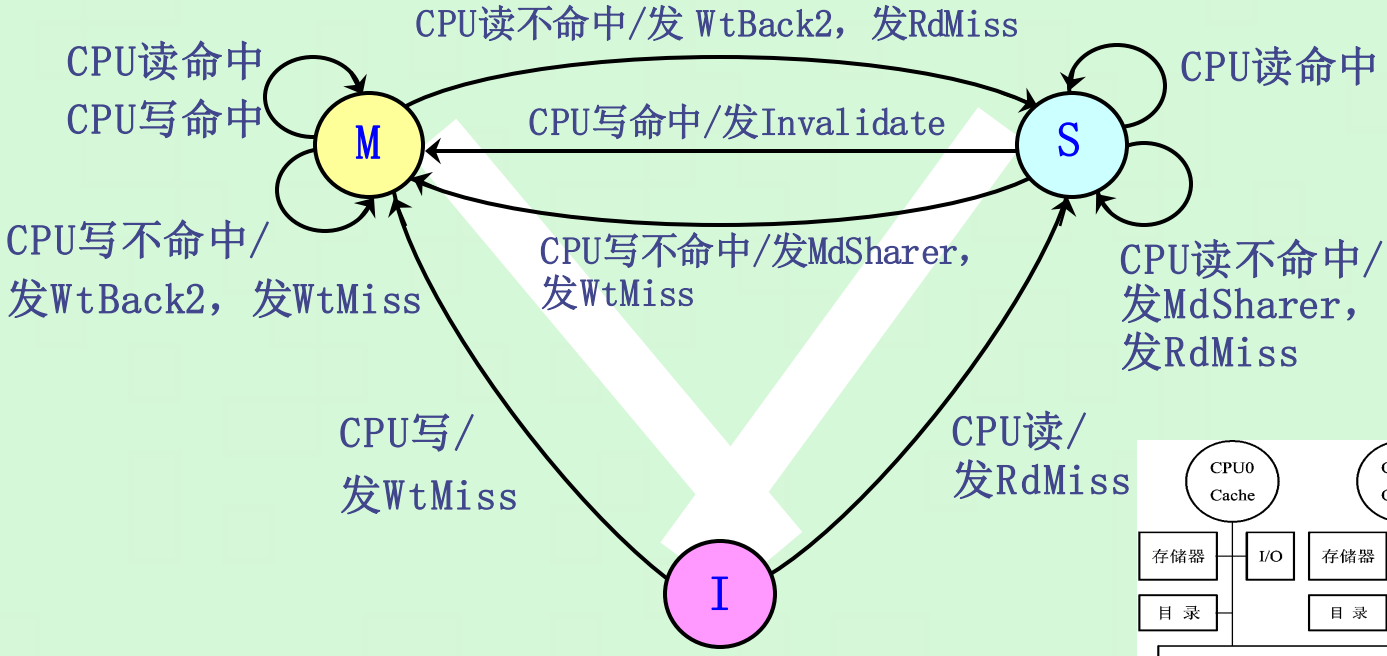
8.3.2 目录协议实例

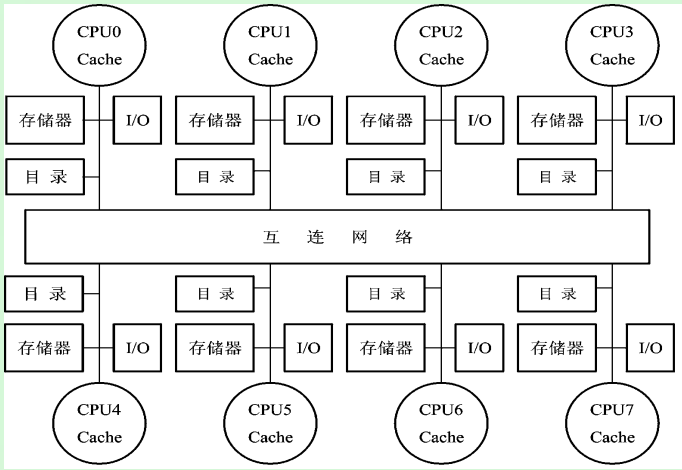
- 在基于目录的协议中，目录承担了一致性协议操作的主要功能。
  - 本地结点把请求发给宿主结点中的目录，再由目录控制器有选择地向远程结点发出相应的消息。
  - 发出的消息会产生两种不同类型的动作：
    - 更新目录状态
    - 使远程结点完成相应的操作



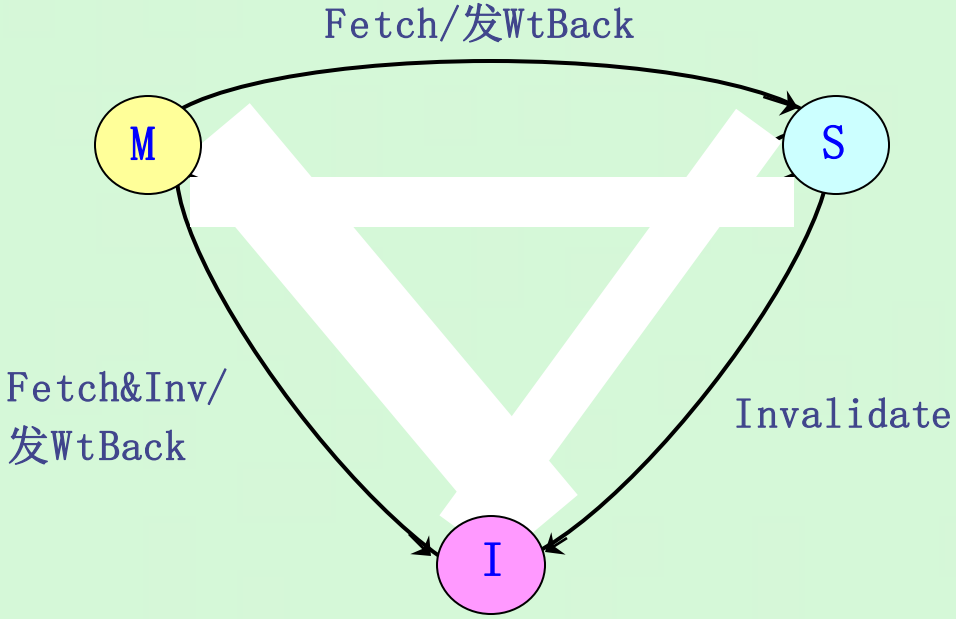
1. 在基于目录协议的系统中， Cache块的状态转换图。

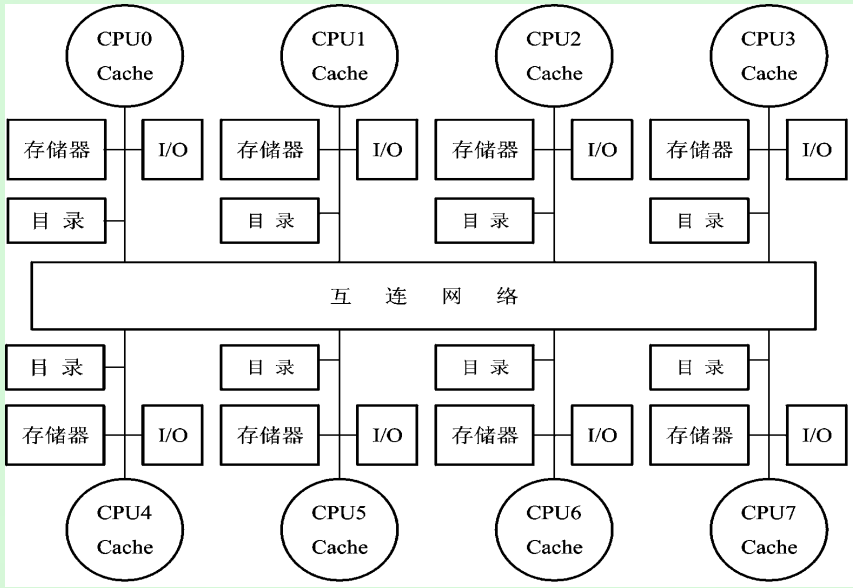
- 响应本地Cache CPU请求





- 远程结点中Cache块响应来自宿主结点的请求的状态转换图





2. 目录的状态转换及相应的操作

如前所述：

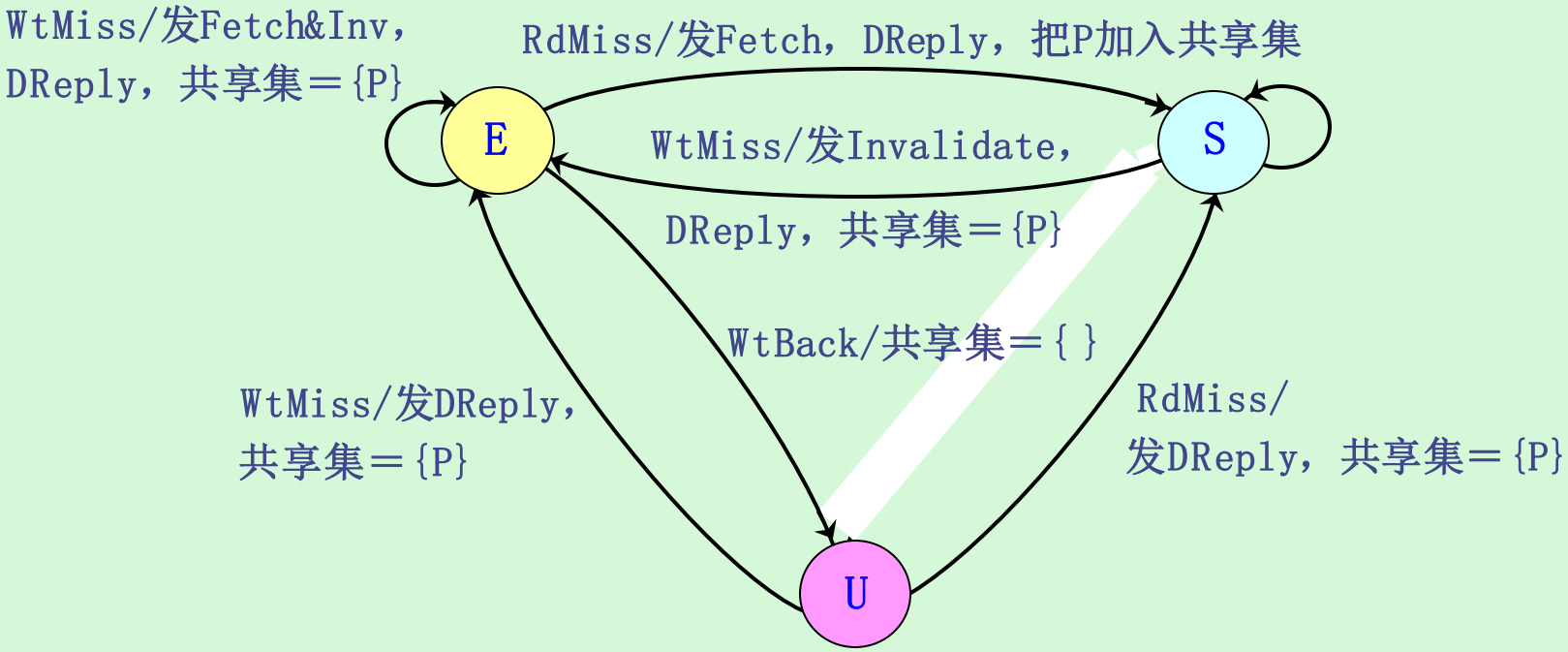
- 目录中存储器块的状态有3种
  - 未缓存
  - 共享
  - 独占
- 位向量记录拥有其副本的处理器的集合。这个集合称为共享集合。
- 对于从本地结点发来的请求，目录所进行的操作包括：

- 向远程结点发送消息以完成相应的操作。这些远程结点由共享集合指出；
- 修改目录中该块的状态；
- 更新共享集合。

➢ 目录可能接收到3种不同的请求

- 读不命中
- 写不命中
- 数据写回

（假设这些操作是原子的）



U：未缓存（Uncached）　　S：共享（Shared）：只读

E：独占（Exclusive）：可读写　　P：本地处理器

目录的状态转换及相应的操作

- 当一个块处于未缓存状态时，对该块发出的请求及处理操作为：
  - RdMiss（读不命中）
    - 将所要访问的存储器数据送往请求方处理机，且该处理机成为该块的唯一共享结点，本块的状态变成共享。
  - WtMiss（写不命中）
    - 将所要访问的存储器数据送往请求方处理机，该块的状态变成独占，表示该块仅存在唯一的副本。其共享集合仅包含该处理机，指出该处理机是其拥有者。

- 当一个块处于共享状态时，其在存储器中的数据是当前最新的，对该块发出的请求及其处理操作为：
  - **RdMiss**
    - 将存储器数据送往请求方处理机，并将其加入共享集合。
  - **WtMiss**
    - 将数据送往请求方处理机，对共享集合中所有的处理机发送作废消息，且将共享集合改为仅含有该处理机，该块的状态变为独占。

- 当某块处于独占状态时，该块的最新值保存在共享集合所指出的唯一处理机（拥有者）中。

  有三种可能的请求：

  - **RdMiss**
    - 将“取数据”的消息发往拥有者处理机，将它所返回给宿主结点的数据写入存储器，并进而把该数据送回请求方处理机，将请求方处理机加入共享集合。
    - 此时共享集合中仍保留原拥有者处理机（因为它仍有一个可读的副本）。
    - 将该块的状态变为共享。

- **WtMiss**
  - 该块将有一个新的拥有者。
  - 给旧的拥有者处理机发送消息，要求它将数据块送回宿主结点写入存储器，然后再从该结点送给请求方处理机。
  - 同时还要把旧拥有者处理机中的该块作废。把请求处理机加入共享者集合，使之成为新的拥有者。
  - 该块的状态仍旧是独占。
- **WtBack**（写回）
  - 当一个块的拥有者处理机要从其Cache中把该块替换出去时，必须将该块写回其宿主结点的

存储器中，从而使存储器中相应的块中存放的数据是最新的（宿主结点实际上成为拥有者）；

- 该块的状态变成未缓冲，其共享集合为空。

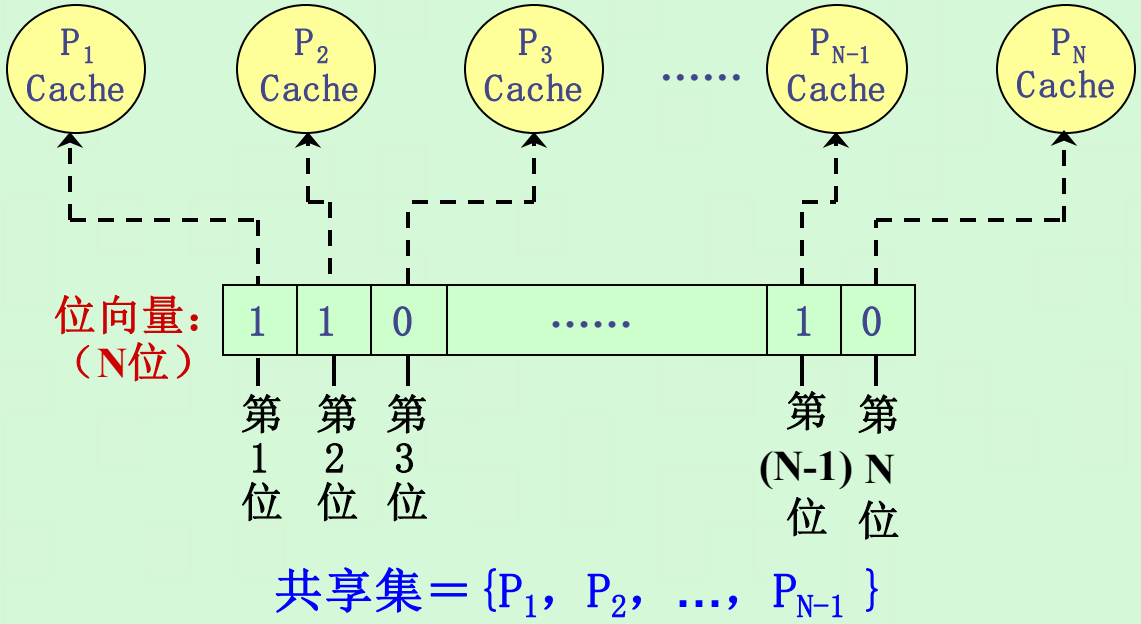
8.3.3 目录的三种结构

- 不同目录协议的主要区别主要有两个
  - 所设置的存储器块的状态及其个数不同
  - 目录的结构
- 目录协议分为3类

全映像目录、有限映像目录、链式目录

1. 全映像目录

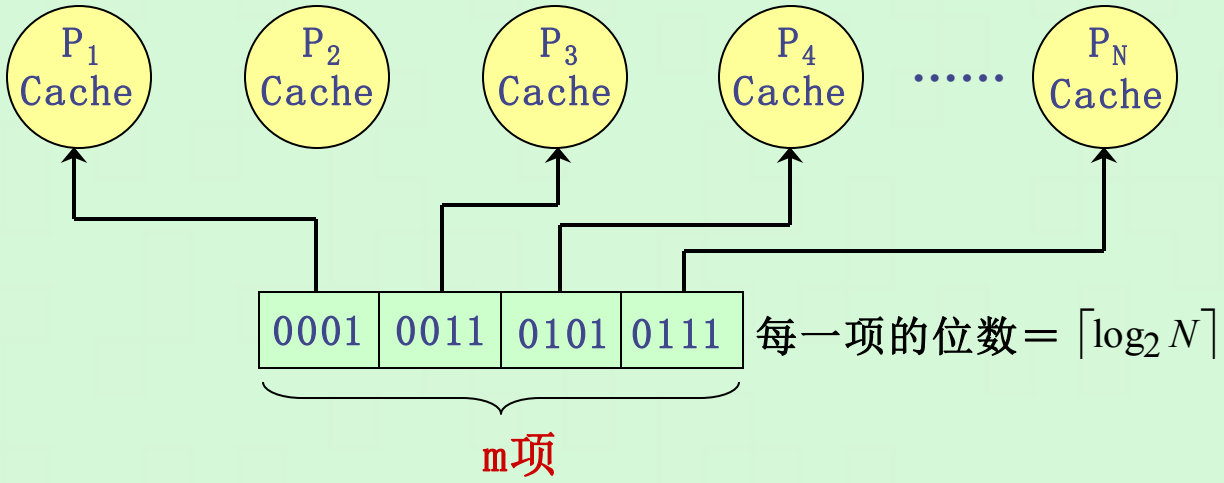
- 每一个目录项都包含一个N位（N为处理机的个数）的位向量，其每一位对应于一个处理机。
  - 举例
- 优点：处理比较简单，速度也比较快。
- 缺点：
  - 存储空间的开销很大。
  - 目录项的数目与处理机的个数N成正比，而目录项的大小（位数）也与N成正比，因此目录所占用的空间与N^2成正比。
  - 可扩放性很差。



- 当位向量中的值为“1”时，就表示它所对应的处理机有该数据块的副本；否则就表示没有。
- 在这种情况下，共享集合由位向量中值为“1”的位所对应的处理机构成。

2. 有限映像目录

- 提高其可扩放性和减少目录所占用的空间。
- 核心思想：采用位数固定的目录项目
  - 限制同一数据块在所有Cache中的副本总数。
  - 例如，限定为常数m。则目录项中用于表示共享集合所需的二进制位数为：$m \times \log_2 N$。
  - 目录所占用的空间与$N \times \lceil \log_2 N \rceil$成正比。
- 举例



共享集＝{P_1，P_3，P_5，P_7 }

有限映像目录（m＝4，N≥8的情况）

- 缺点
  - 当同一数据的副本个数大于m时，必须做特殊处理。当目录项中的m个指针都已经全被占满，而某处理机又需要新调入该块时，就需要在其m个指针中选择一个，将之驱逐，以便腾出位置，存放指向新调入块的处理机的指针。

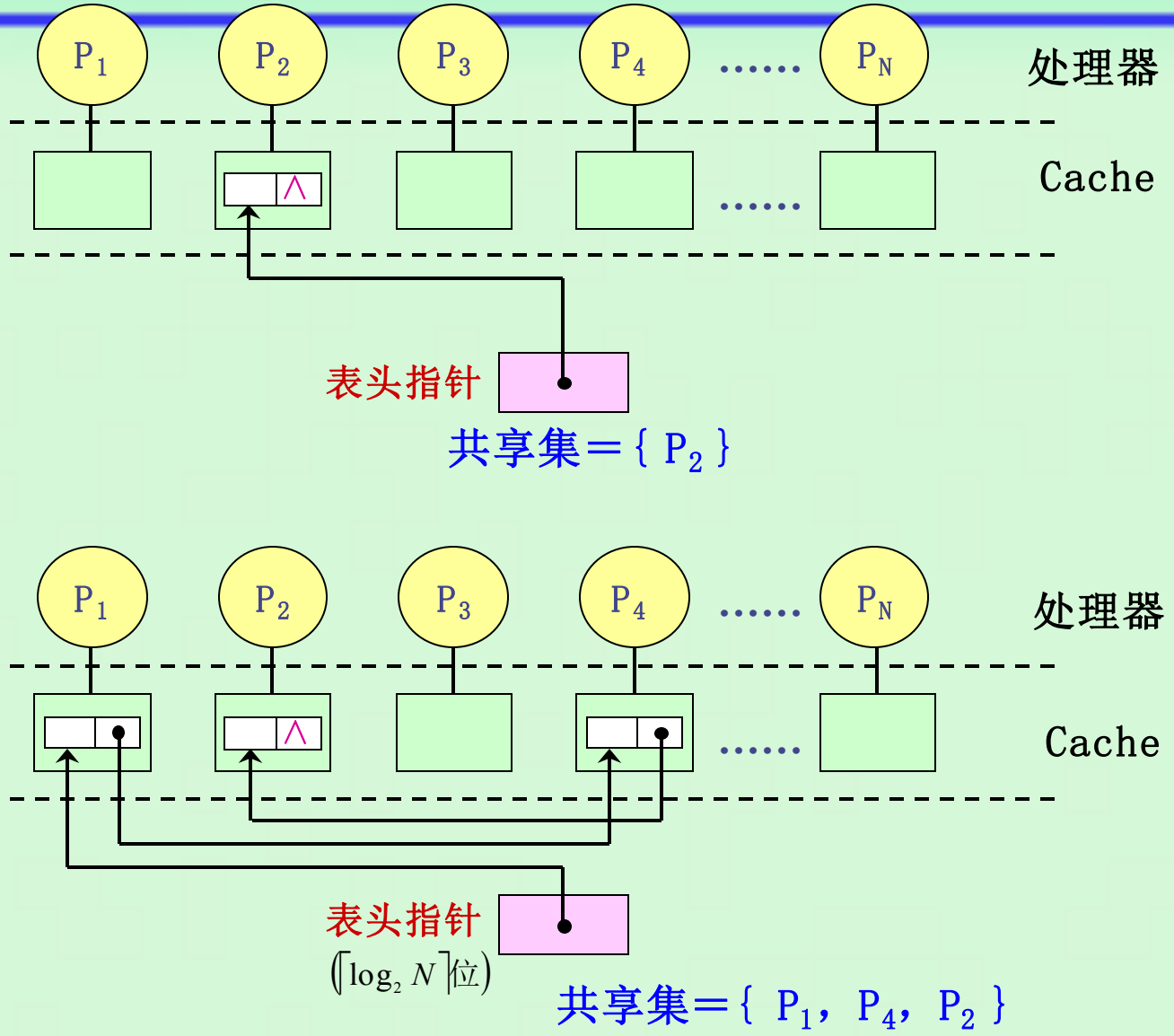
3. 链式目录

- 用一个目录指针链表来表示共享集合。当一个数据块的副本数增加（或减少）时，其指针链表就跟着变长（或变短）。
- 由于链表的长度不受限制，因而带来了以下优点：既不限制副本的个数，又保持了可扩展性。

- 链式目录有两种实现方法
  - 单链法

    当Cache中的块被替换出去时，需要对相应的链表进行操作——把相应的链表元素（假设是链表中的第i个）删除。实现方法有以下两种：
    - 沿着链表往下寻找第i个元素，找到后，修改其前后的链接指针，跳过该元素。
    - 找到第i个元素后，作废它及其后的所有元素所对应的Cache副本。
  - 双链法
    - 在替换时不需要遍历整个链表。
    - 节省了处理时间，但其指针增加了一倍，而且一致性协议也更复杂了。

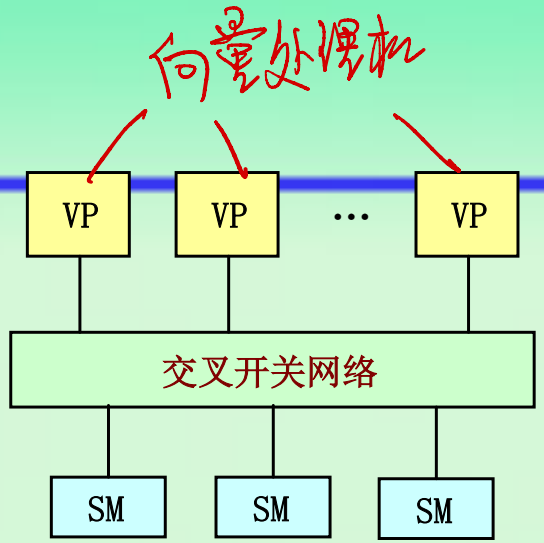
采用单向链法的示意图


P1
P2
P3
P4
……
PN
处理器
Cache
表头指针
共享集＝{ P2 }
P1
P2
P3
P4
……
PN
处理器
Cache
表头指针
(⌈log2 N⌉位)
共享集＝{ P1，P4，P2 }

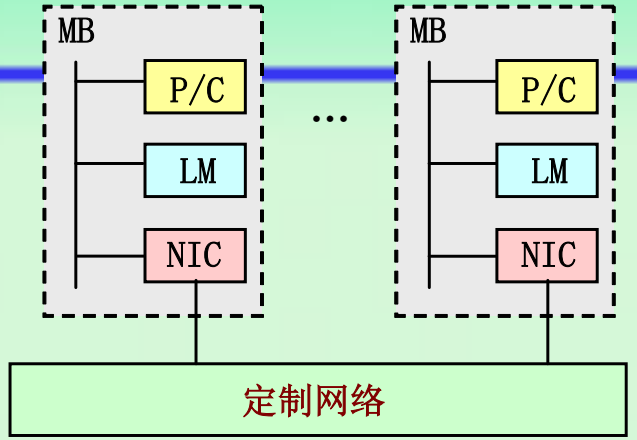
8.6 大规模并行处理机

8.6.1 并行计算机系统结构

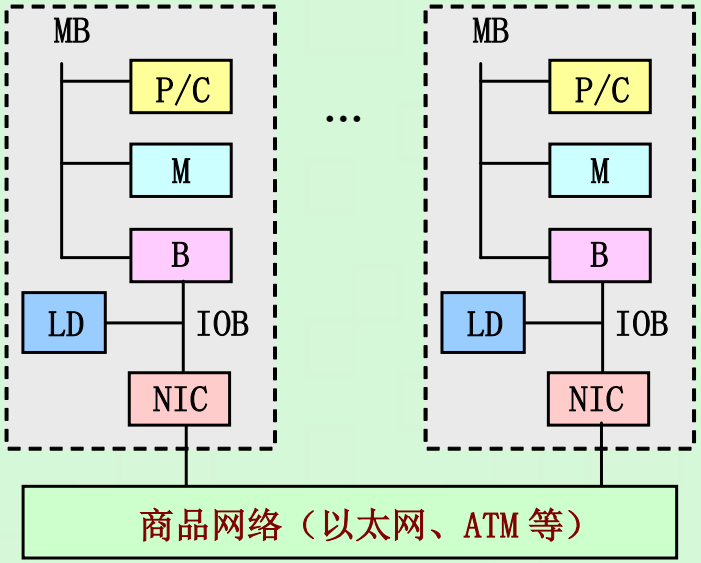
- 目前流行的高性能并行计算机系统结构通常可以分成以下5类：
  - 并行向量处理机（**PVP**）
  - 对称式共享存储器多处理机（**SMP**）
  - 分布式共享存储器多处理机（**DSM**）
  - 大规模并行处理机（**MPP**）
  - 机群计算机（**Cluster**）



（a）PVP



（b）MPP



（c）机群

VP：向量处理器

SM：共享存储器模块

P/C：商品微处理器/Cache

LM、M：本地存储器

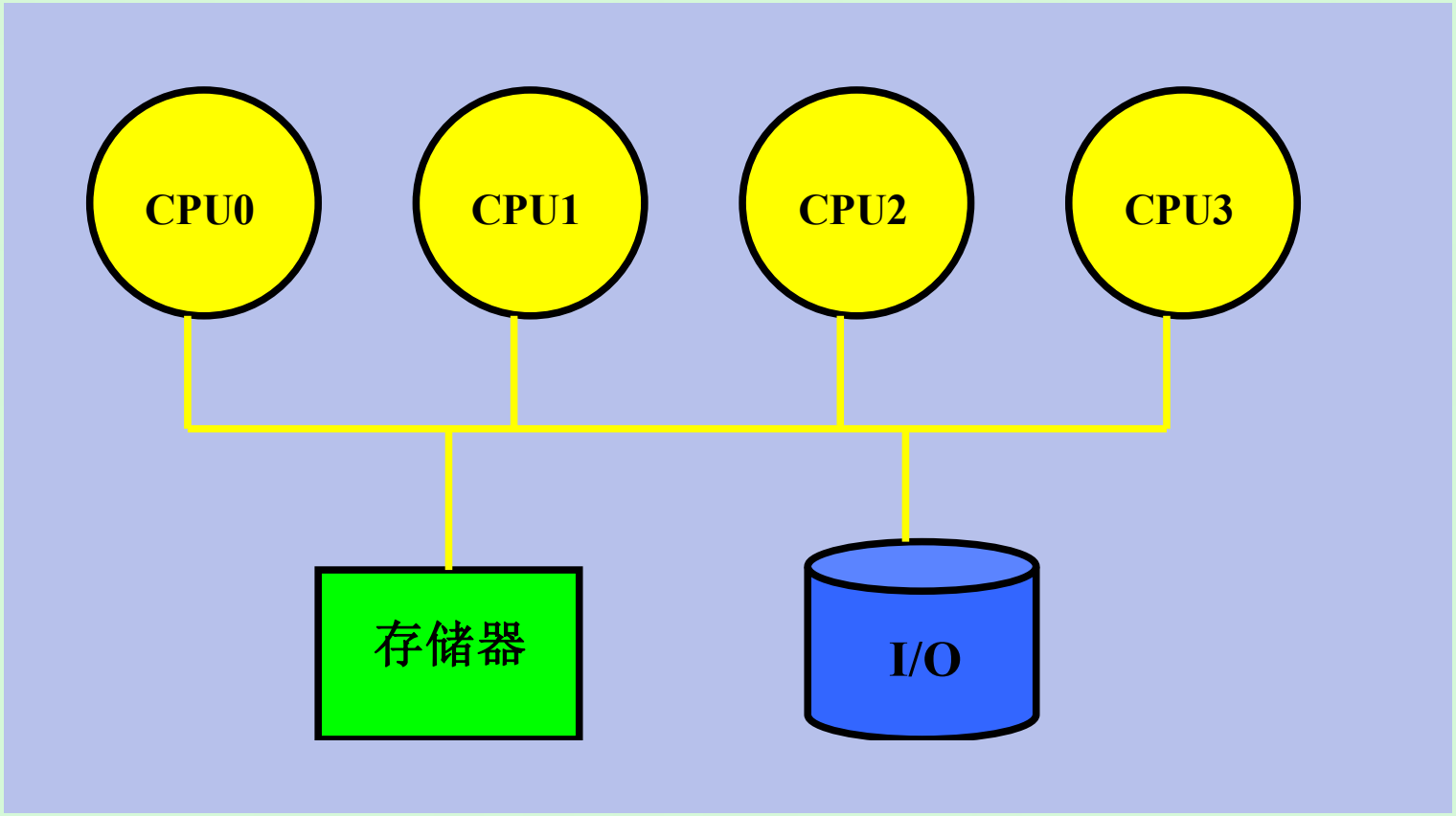
NIC：网络接口电路

MB：存储器总线

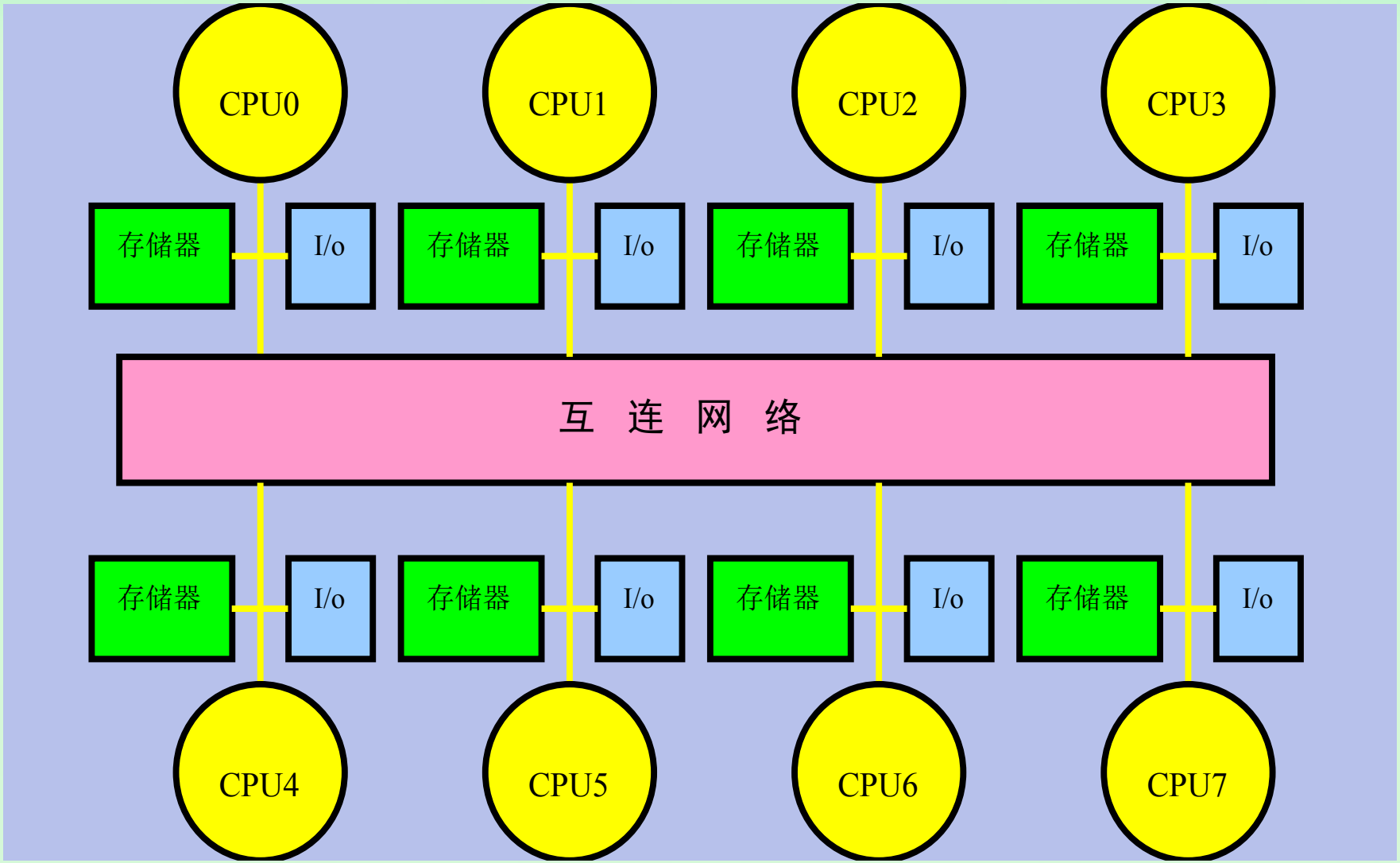
LD：本地磁盘

IOB：I/O 总线

B：存储总线与 I/O 总线之间



对称式共享存储器多处理机系统SMP



分布式共享存储器多处理机系统 DSM

1. 并行向量处理机
   - Cray C-90和 Cray T-90是这类机器的代表。
   - **PVP**系统一般由若干台高性能向量处理机（VP）构成。这些向量处理机是专门设计和定制的，拥有很高的向量处理性能。
   - PVP中经常采用专门设计的高带宽的交叉开关网络，把各VP与共享存储器模块SM连接起来。
   - 这样的机器通常不使用Cache，而是使用大量的向量寄存器和指令缓冲器。
2. 对称式共享存储器多处理机和分布式共享存储器多处理机

3. 大规模并行处理机

- Intel Paragon和IBM SP2是这类机器的代表。
- MPP往往是超大规模的计算机系统。
- 具有以下特点：
  - 处理结点使用商用微处理器，而且每个结点可以有多个微处理器；
  - 具有较好的可扩放性，能扩展成具有成百上千个处理器；
  - 系统中采用分布非共享的存储器，各结点有自己的地址空间；
  - 采用专门设计和定制的高性能互连网络；
  - 采用消息传递的通讯机制。

4. 机群计算机

- 一种价格低廉、易于构建、可扩放性极强的并行计算机系统。它由多台同构或异构的独立计算机通过高性能网络或局域网互连在一起，协同完成特定的并行计算任务。
  - Berkeley NOW和SP2是这类机器的代表。
- 机群的主要特点
  - 每个结点都是一台完整的计算机，拥有本地磁盘和操作系统，可以作为一个单独的计算资源供用户使用。
  - 机群的各个结点一般通过商品化网络连接在一起；
  - 网络接口以松散耦合的方式连接到结点的I/O总线。

5类机器特征比较

属性	PVP	SMP	MPP	DSM	机群
结构类型	MIMD	MIMD	MIMD	MIMD	MIMD
处理器类型	专用定制	商用	商用	商用	商用
互连网络	定制 交叉开关	总线、 交叉开关	定制网络	定制网络	商用网络 （以太网、ATM）
通信机制	共享变量	共享变量	消息传递	共享变量	消息传递
地址空间	单地址空间	单地址空间	多地址空间	单地址空间	多地址空间
系统存储器	集中共享	集中共享	分布非共享	分布共享	分布非共享
访存模型	UMA	UMA	NORMA	NUMA	NORMA
代表机器	Cray C-90， Cray T-90， NEC SX4， 银河1号	IBM R50， SGI Power Challenge， DEC Alpha 服务器8400， 曙光1号	Intel Paragon， IBM SP2， Intel TFLOPS， 曙光-1000/2000	Stanford DASH， Cray T 3D， SGI/Cray Origin 2000	Berkeley NOW， Alpha Farm， Digital Trucluster

8.6.2 大规模并行处理机

1. MPP的出现和发展

- 从20世纪80年代末开始，MPP系统逐渐地显示出代替和超越向量计算多处理机系统的趋势。
  - 早期的MPP：Intel Paragon(1992年)、KSR1.Cray T3D(1993年)、IBM SP2(1994年)等。

    （分布存储的MIMD计算机）
  - MPP的高端机器：1996年Intel公司的ASCI Red

    1997年SGI Cray公司的T3E900

    （万亿次浮点运算的高性能并行计算机）

- 在这个时期，消息传递的大规模并行处理系统得到了迅速发展。

- 从90年代后期开始，基于消息传递的MPP系统慢慢地从主流的并行处理市场退出。
- 随着网络技术的发展，机群系统和MPP系统的界限越来越模糊。
  - 90年代后期以来高性能计算机系统结构发展的一个趋势，新涌现的高性能计算机系统大多数都是由可扩放的高速互连网络连接的基于RISC微处理器的对称多处理机机群。
  - 机群系统已经成了构建超大规模并行计算机系统的主要模式，MPP则是在慢慢地退居二三线了。

2. MPP系统概述

- MPP结构的一个重要特性：可扩放性
  - 处理器的数量可扩放至数千个处理器。
  - 主存、I/O能力和带宽也能随处理器数量的增长而成比例地增长。
- MPP主要采用了以下技术来提高系统的可扩放性。
  - 使用物理上分布的主存体系结构，使分布式主存的总容量和总带宽能随处理结点数量的增加而增加。
  - 处理能力、主存与I/O能力平衡发展。
  - 计算能力与并行性平衡发展。

3. 3种大型MPP的特点（分别代表构造大型系统的不同方法）

MPP模型	Intel/Sandia ASCI Option Red	IBM SP2	SGI/Cray Origin 2000
典型配置	9072个处理器 1.8 Tflop/s（NSL）	400个处理器 100 Gflop/s（MHPCC）	128个处理器 51 Gflop/s（NCSA）
推出日期	1996年12月	1994年9月	1996年10月
CPU类型	200 MHz，200 Mflop/s Pentium Pro	67 MHz，267 Mflop/s POWER2	200 MHz，400Mflop/s MIPS R10000
节点结构 数据存储	2个处理器， 32～256MB主存， 共享磁盘	1个处理器， 64MB～2GB本地主存， 1GB～14.5G本地磁盘	2个处理器， 64MB～256MB分布共享主存和共享磁盘
互连网络	分离二维网孔	多级网络	胖超立方体网格
访存模型	NORMA	NORMA	CC-NUMA
节点OS	轻量级内核（LWK）	完全AIX（IBM UNIX）	微内核Cellular IRIX
编程语言	基于PUMA Portals的MPI	MPI和PVM	Power C, Power Fortran
其他编程模型	NX, PVM, HPF	HPF, Linda	MPI, PVM

4. 一些典型的MPP系统的特性

结构特性	IBM SP2	Cray T3D	Cray T3E	Intel Paragon	Intel/Sandia Option Red
典型配置	400个节点 100 Gflop/s	512个节点 153 Gflop/s	512个节点 1.2 Tflop/s	400个节点 40 Gflop/s	4536个节点 1.8 Tflop/s
推出日期	1994年	1993年	1996年	1992年	1996年
CPU类型	67 MHz 267 Mflop/s POWER2	150 MHz 150 Mflop/s Alpha 21064	300 MHz 600 Mflop/s Alpha 21164	50 MHz 100 Mflop/s Intel i860	200 MHz 200 Mflop/s Pentium Pro
节点结构 数据存储	1CPU 64MB～2GB 本地存储器， 1～4.5GB 本地磁盘	2CPU 64MB主存， 50GB共享磁盘	4～8CPU 256MB～16GB DSM 主存， 共享磁盘	1～2CPU 16～128MB 本地存储器， 40GB共享磁盘	2CPU 32～256MB 本地存储器， 共享磁盘
互连网络	多级网络	三维环绕	三维环绕	二维网孔	分离二维网孔
访存模型	NORMA	NUMA	NCC—NUMA	NORMA	NORMA

结构特性	IBM SP2	Cray T3D	Cray T3E	Intel Paragon	Intel/Sandia Option Red
结构特性	IBM SP2	Cray T3D	Cray T3E	Intel Paragon	Intel/Sandia Option Red
节点OS	完全AIX（IBM UNIX）	微内核	基于Chorus的微内核	微内核	轻量级内核（LWK）
编程模型	消息传递	共享变量、消息传递、PVM	共享变量、消息传递、PVM	消息传递	基于PUMA Portals消息传递
编程语言	MPI、PVM、HPF、Linda	MPI、HPF	MPI、HPF	NX、MPI、PVM	NX、PVM、HPF
点到点通信延迟	40μs	2μs	N/A	30μs	10μs
点到点带宽	35 MB/s	150 MB/s	480 MB/s	175 MB/s	380 MB/s

8.7 多处理机实例1： T1

1. T1的结构

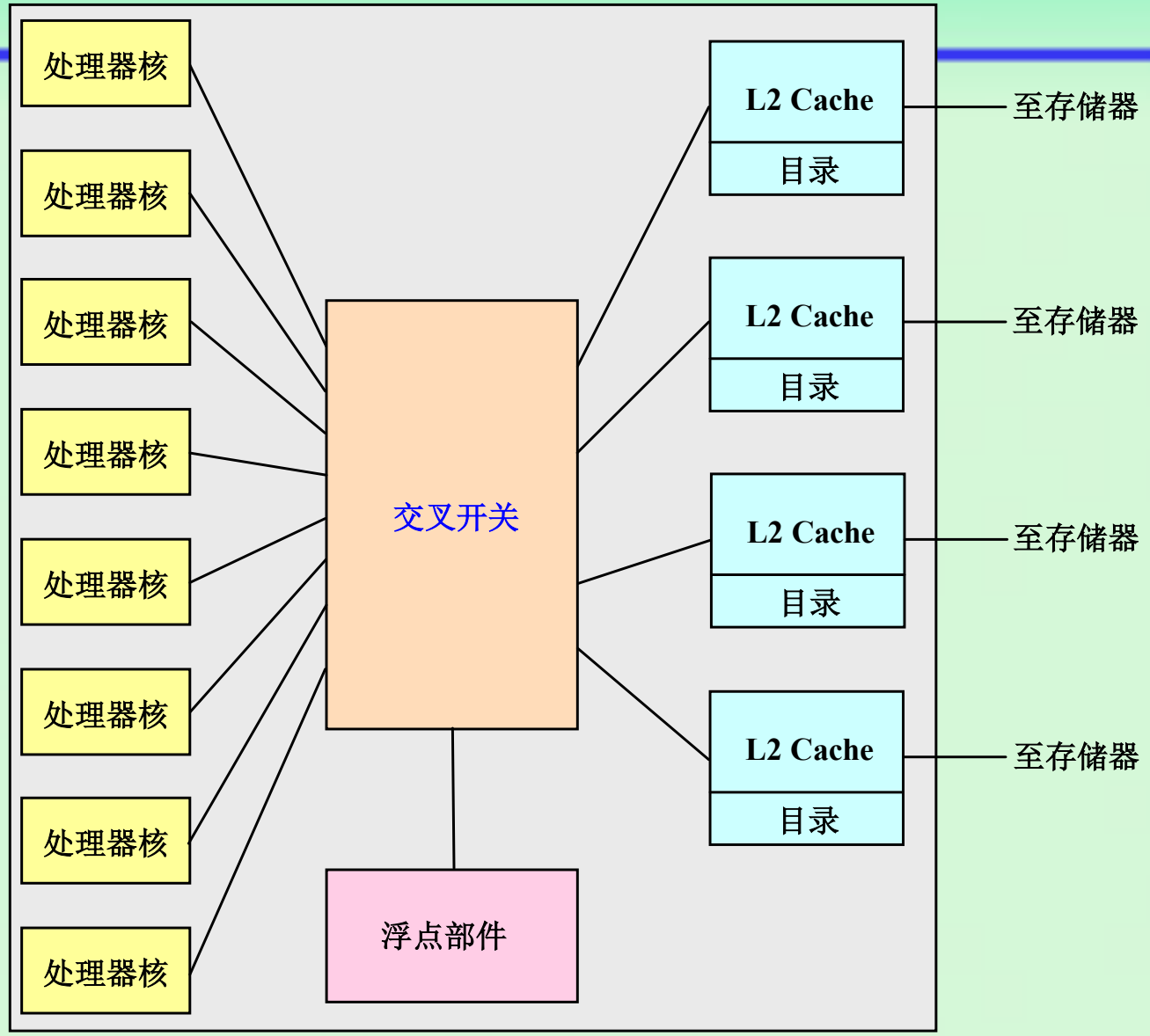
- SUN公司于2005年作为服务器处理器发布的多核多处理器。
  - 同时采用了多线程和多核技术，全面提高吞吐率。
  - 每个T1处理器中有8个处理器核，每个核最多支持4个线程。
  - 每个处理器核中都有一条6段的单流出流水线。

（类似于前面介绍的5段流水线，只是增加了一个段，用于进行线程切换。）

- 采用细粒度多线程，在每个时钟周期都可以切换到新的线程，而且在调度时可以跳过那些因流水线延迟或Cache不命中而处于等待状态的线程。
- T1处理器只有在所有其4个线程都处于等待或停顿时才会出现空闲状态。
- load和分支指令会导致3个时钟周期的延迟，不过它们都可以通过执行其他线程而被隐藏。

T1的组成结构

- 有8个核，每个核中都带有一个第一级Cache。
- 8个核通过一个交叉开关与4个第二级（L2）Cache相连。
- 用目录表法来实现Cache内容的一致性 。


处理器核
处理器核
处理器核
处理器核
处理器核
处理器核
处理器核
处理器核
交叉开关
浮点部件
L2 Cache
目录
L2 Cache
目录
L2 Cache
目录
L2 Cache
目录
至存储器
至存储器
至存储器
至存储器

T1的结构

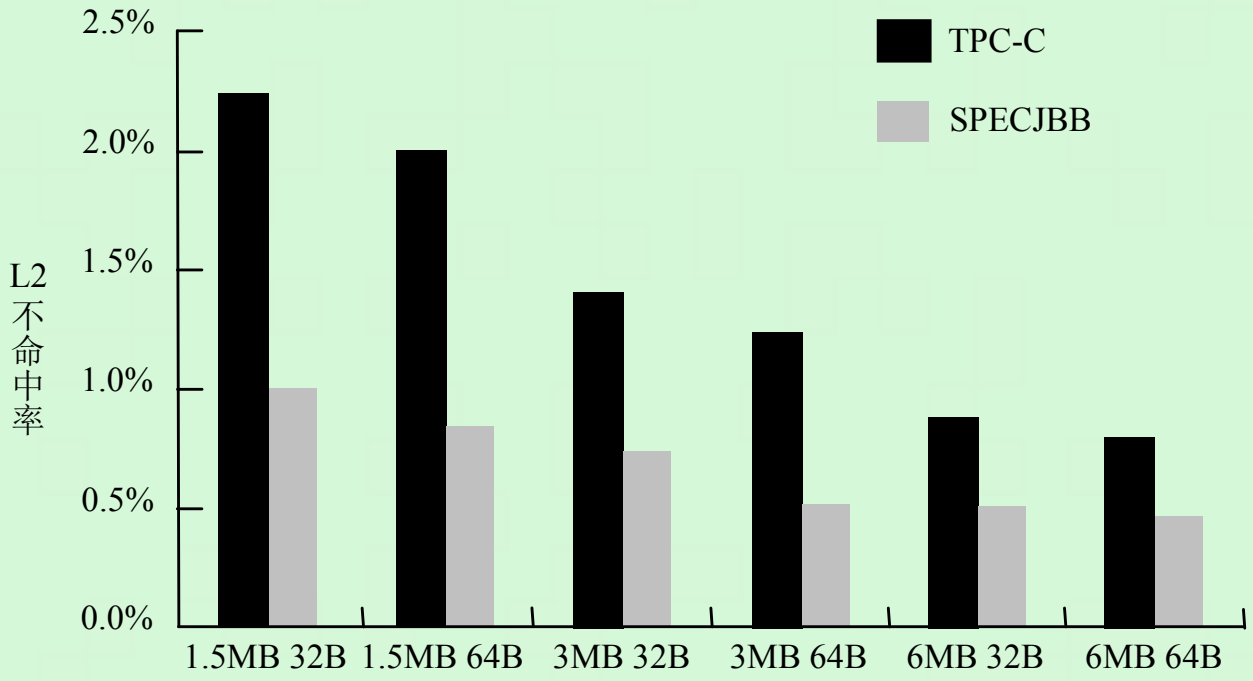
➢ T1的主要特性

特征	Sun T1
多处理器和多线程支持	每芯片8个核，每核4个线程。细粒度线程调度。 8个核共享一个浮点运算部件。支持片内多处理器。
流水线结构	简单的按序6段流水线，load和分支的延迟为3个时钟周期。
一级Cache	16KB指令Cache，8KB数据Cache。64字节块大小。 在无竞争的情况下，L1不命中的开销是23个时钟周期。
二级Cache	4个独立的二级Cache，每个750KB且和存储体相连。64字节块大小。 在无竞争的情况下，L2不命中的开销是110个时钟周期。
初始版本	90nm工艺，最高时钟频率1.2GHz，电源功率79W， 300M个晶体管，圆片面积大小379mm^2。

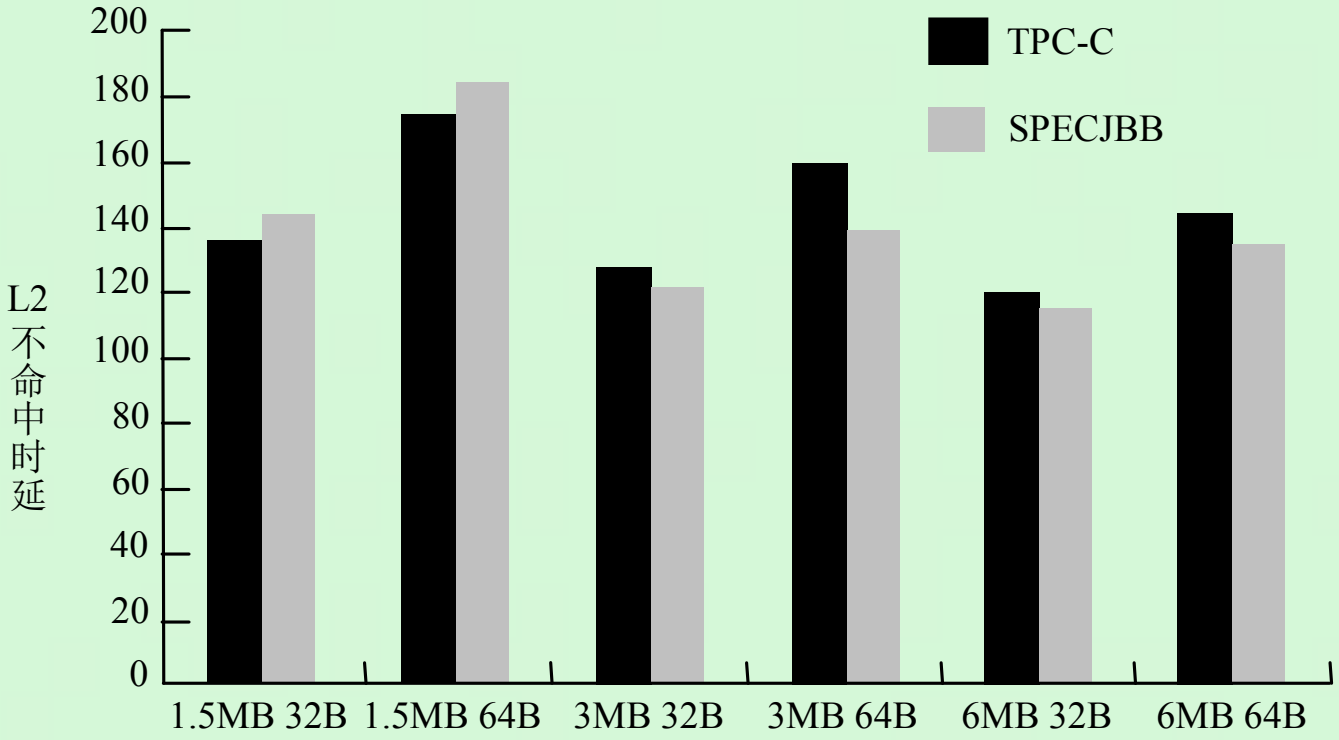
2. T1的性能

基准测试程序：TPC-C、SPECJBB、SPECWeb99

- 在以下不同情况下L2 Cache的不命中率
  - 容量分别为：1.5MB、3MB和6MB
  - 块大小分别：32B和64B



- 在不同容量和块大小的情况下（与上图同），L2 Cache的不命中延迟



- T1的每线程CPI、每核CPI以及有效的IPC（每个时钟周期完成的指令数）

基准测试程序	每线程CPI	每核CPI	8个核的有效CPI	8个核的有效IPC
TPC-C	7.2	1.8	0.225	4.4
SPECJBB	5.6	1.40	0.175	5.7
SPECWeb99	6.6	1.65	0.206	4.8

有效IPC=8÷每核CPI

3. 4种多核处理器的性能对比

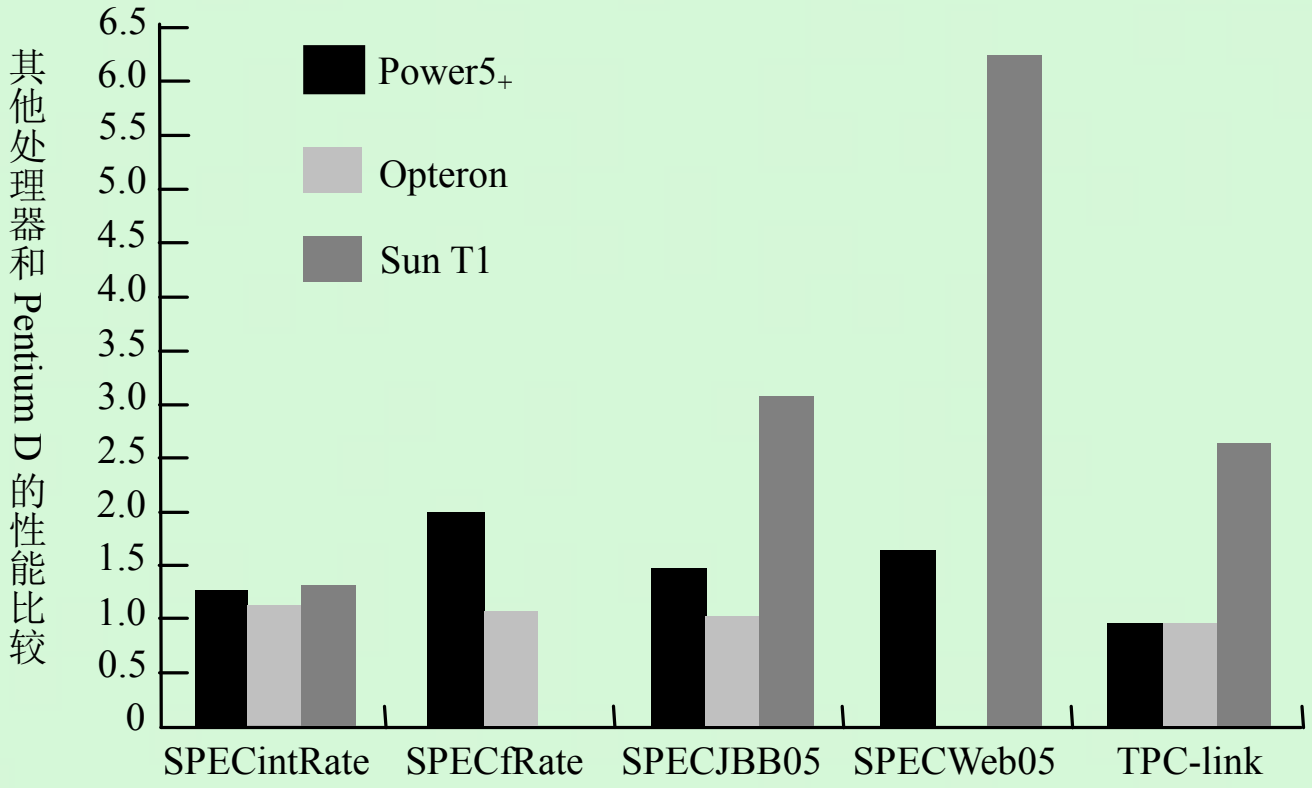
特征	SUN T1	AMD Opteron	Intel Pentium D	IBM Power 5
核	8	2	2	2
每个核每时钟 周期发射的指令	1	3	3	4
多线程	Fine-grained	No	SMT	SMT
Cache	16/8	64/64	12k uops/16	64/32
一级 I/D in KB per core	3MB shared	1MB/core	1MB/core	二级： 1.9MB shared
二级 Per core/shared				三级：36MB

特征	SUN T1	AMD Opteron	Intel Pentium D	IBM Power 5
三级（off-chip）				
存储器带宽峰值（DDR2 DRAMS）	34.4GB/s	8.6GB/s	4.3GB/s	17.2GB/s
MIPS峰值	9600	7200	9600	7600
FLOPS	1200	4800(w.SSE)	6400(w.SSE)	7600
时钟频率（GHz）	1.2	2.4	3.2	1.9
晶体管数量（百万）	300	233	230	276
晶片面积（mm^2）	379	199	206	389
电源功率（W）	79	110	130	125

- 除了是重点开发ILP还是TLP的区别外，这些多核处理器还有一些根本的不同。
  - 它们在对浮点运算提供的支持以及浮点运算的性能上有很大的不同。
  - 它们的多处理器扩展能力不同，这对存储器的设计以及外部接口的使用有很大的影响。
    - Power5的可扩展性是最好的
  - 所用的实现技术差别很大，难以对它们的晶片大小和功耗进行比较。
  - 对存储器系统及其带宽的要求不同。

4种多核处理器的性能

- 以SPECRate、SPECJBB2005、SPECWeb05以及类TPC-C测试基准程序为负载
- 图中所有的数据都对Pentium D的数据进行了归一化处理，即Pentium D的值都是1。



8.8 多处理机实例2：Origin 2000

- Origin 2000系列可扩展服务器产品
  - 该系列包括：Origin 200、Origin 2000 Deskside、Origin 2000 Rack和Cray Origin 2000 4种机器。
  - Origin 2000 Deskside桌面服务器系统支持的处理器数目最多为8个
  - Origin 2000 Rack机柜服务器系统支持的处理器数目最多为16个
  - Cray Origin 2000服务器系统具有大规模扩充能力，支持的处理器数目最多可达到128个。

1. Origin 2000系列服务器产品优点
   - 不仅具有SMP的易编程和平稳扩充特性，而且还具有MPP的高可扩放性，应用非常广泛。
   - 该系列服务器综合平衡了高性能、可扩放性、可用性和兼容性，能满足许多应用的需求。
   - Origin 2000 服务器系列的I/O带宽可达102CB/s，系统传输速率比同类SMP服务器快几十倍。

     （处理、存储和传输各种多媒体信息的理想系统）
2. Origin 2000的关键技术

- CrayLink开关网络技术
  - 多重交叉开关互连技术，用于连接处理器、存储器、I/O设备等。
  - 替代总线成为处理器结点之间的互连网络。
    - 使Origin 2000 系统成为模块化系统，系统规模可以是一个基本的模块，也可以是若干模块的互连，而且还可以方便地通过增加模块数量来扩充。
    - Origin 2000 系统的可扩放性体系结构最多可以扩展至1024个处理器，而且规模增加可使系统性能也呈线性增长，包括计算能力、主存容量和带宽、系统互连带宽、I/O带宽和网络连接能力。

- 通过CrayLink，分布在所有处理器结点上的存储器在逻辑上形成单一寻址空间的共享存储器系统，但对本地和远程存储器访问的时间是不同的，是一个NUMA结构。

Cellular IRIX操作系统

- 工业界最早投入使用的蜂窝式操作系统。
- 将操作系统功能分布到各个处理器结点上，可以实现从小系统到大系统的无缝扩展。
  - 把多个相同的操作系统核心功能分别放到多个“蜂窝”（操作系统单元）中，每个蜂窝分别管理服务器中所有处理器的一个子集。每个操作系统单元都可以非常有效地扩展，单元之间互相通信，为用户提供一种单一的操作系统接口。

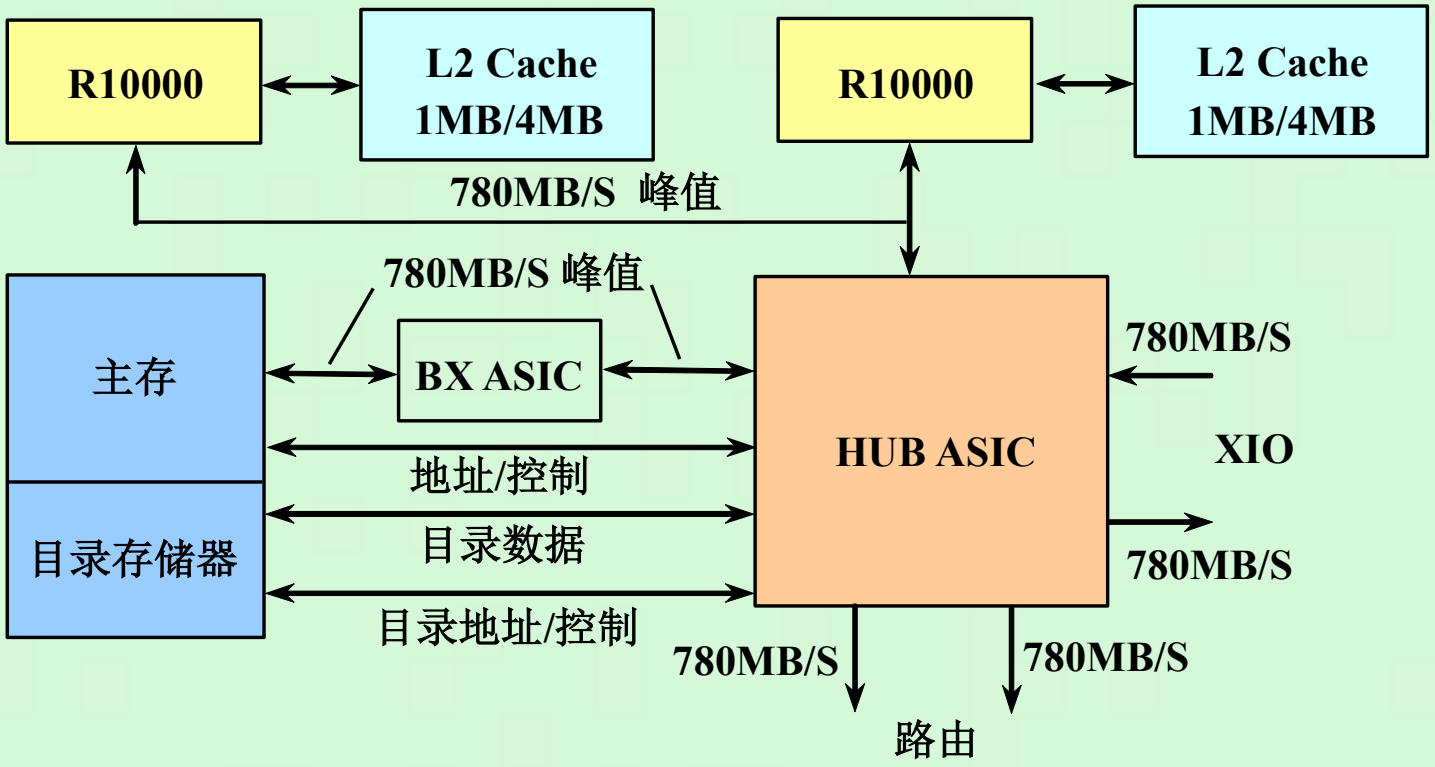
- 操作系统的这种蜂窝结构与积木式的硬件结构相结合，能够把故障隔离起来，可使故障局限于个别操作系统单元中，提高了服务器的可用性和可靠性。

- 从SGI的IRIX演变而来的，是以UNIX为基础的64位蜂窝式操作系统。

Origin 2000系列服务器的硬件结构：

1. 结点板 （Origin200的主板）

- 组成部分
  - 一个或两个MIPS R10000微处理器（内含第一级Cache）。其主频是180MHz或195MHz。



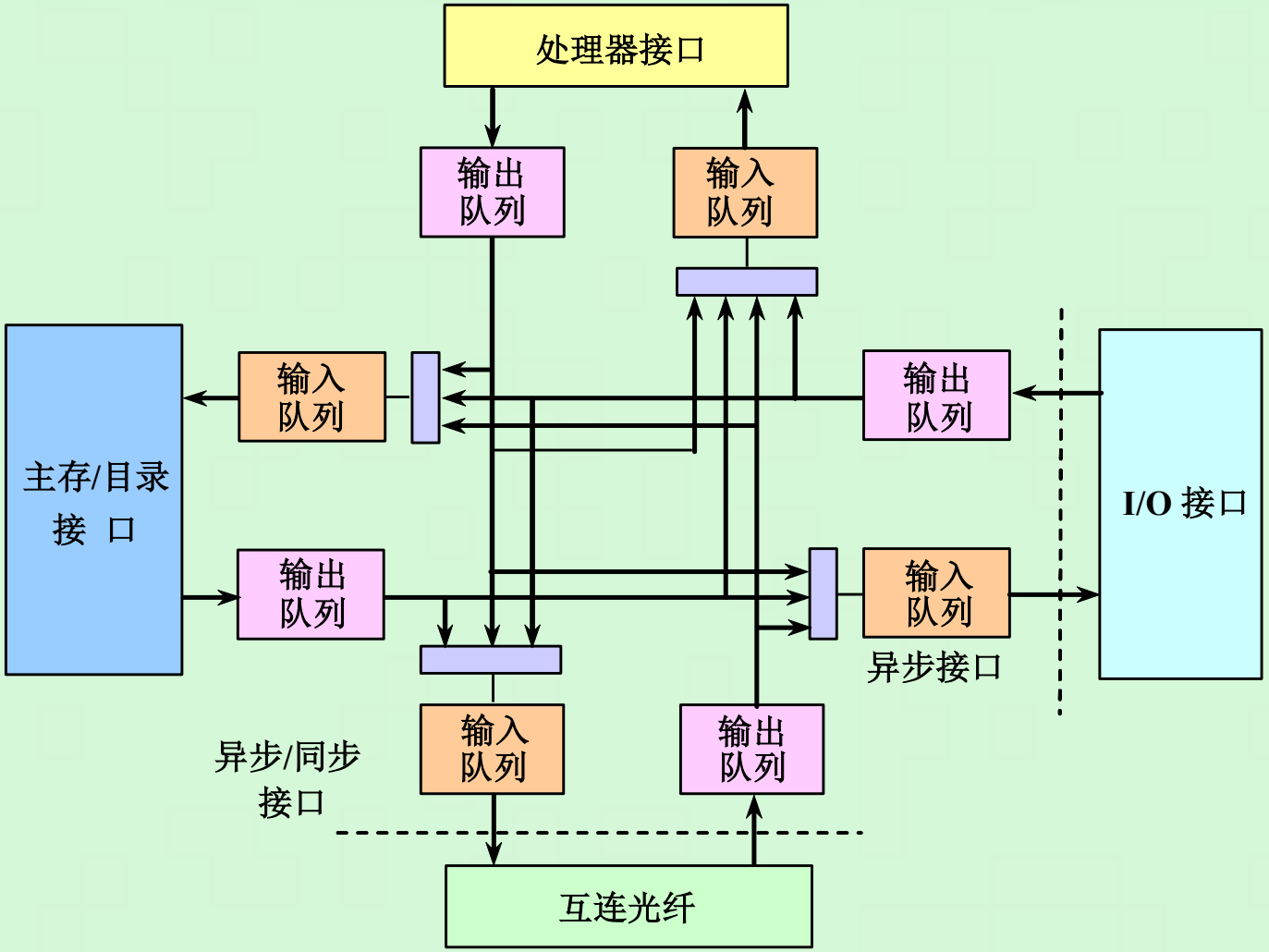
Origin 2000结点板结构

- 与处理器相配的第二级Cache，其容量为1MB或4MB；
- 主存储器（本地）以及用于实现Cache一致性的目录存储器；
- 用于实现互连的ASIC芯片，称为HUB。

  提供了4个接口：
  - 与处理器的接口
  - 与存储器的接口
  - I/O接口
  - 路由接口（接CrayLink互连网络）

HUB的结构

- 4个端口在内部以交叉开关互连，通过发送消息进行通信。
- 存储器接口能双向传送数据，最大传输率为780MB/s，


处理器接口
输出队列
输入队列
输入队列
输出队列
主存/目录接 口
输出队列
输入队列
I/O 接口
异步接口
异步/同步接口
输入队列
输出队列
互连光纤

- I/O和路由器接口各有两个半双工传送端口，最大传输率为2×780MB/s，即1.56GB/s。
- 每个Hub接口连接2个先进先出（FIFO）缓冲器，分别用于输入和输出的缓冲。

2. I/O子系统

- 由一组高速链路构成。称为Crosstalk（XTALK）。
- Crosstalk I/O系统是分布的，在每个结点板上有一个I/O端口，可以被每个处理器访问。
- I/O操作通过结点板上的单端口Crosstalk协议的链路进行控制，或者通过在Crossbow（XBOW）

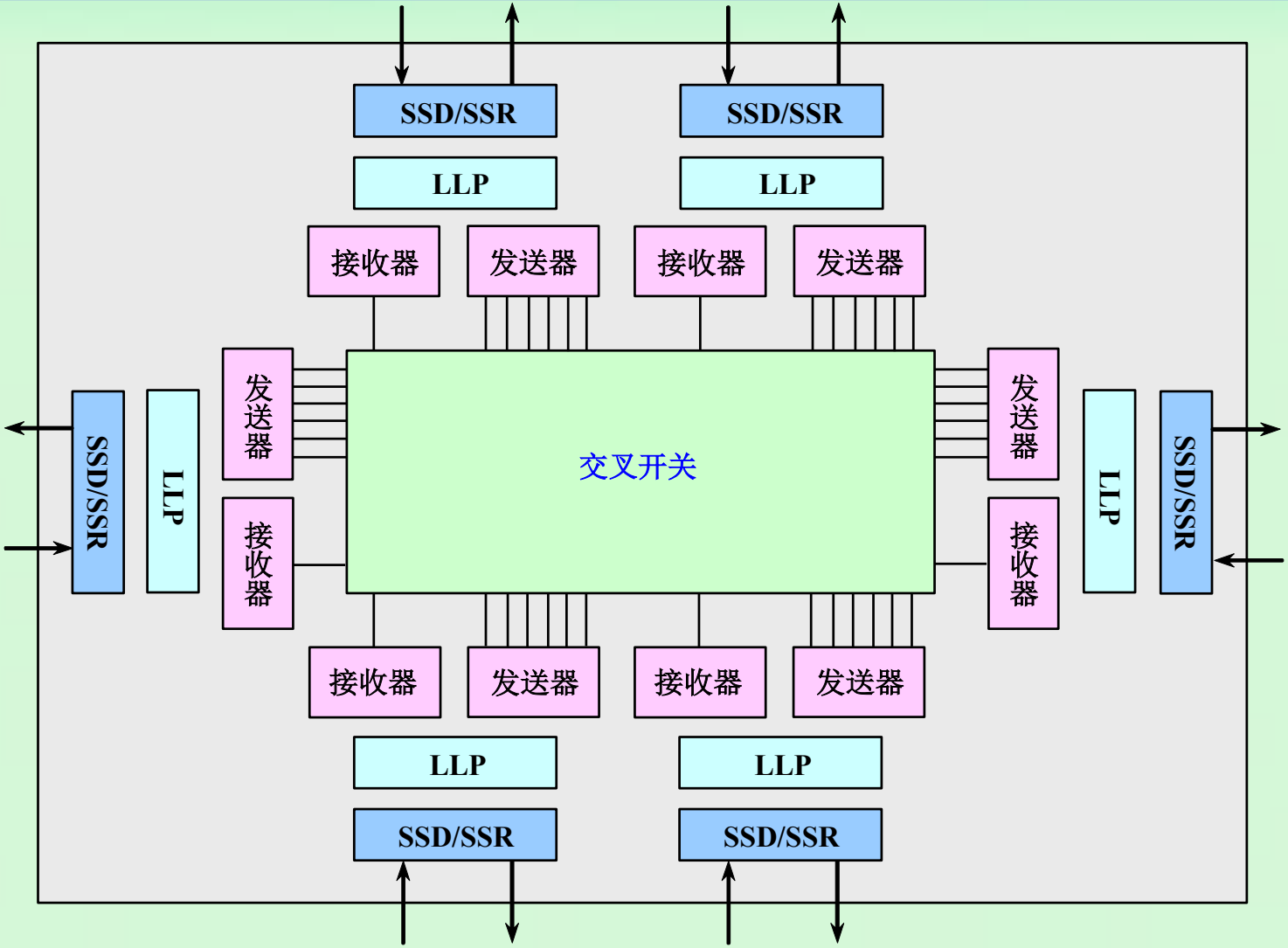
ASIC芯片上的智能交叉开关进行互连。

- XBOW ASIC芯片将Crosstalk I/O端口扩充到8个端口。
  - 6个端口用于I/O
  - 2个端口用于连接到结点板

3. 互连网络子系统

- 互连网络子系统是由路由器和链路构成的。
  - 每个路由器由一组交叉开关组成，能实现多路无阻塞连接。
  - 每条双向链路带宽峰值达到1.6GB/s。

- 互连网络CrayLink Interconnect为每对结点提供至少两条独立链路进行通信。
  - 这种结构使得结点之间的通信可以绕过不能运行的路由器和断开了的链路。
- 路由器将结点板上的HUB物理地连接到CrayLink Interconnect上。
  - 路由器的核心：实现6路无阻塞交叉开关的路由ASIC芯片
  - 路由器的交叉开关允许6个路由端口全双工同时操作，每个端口有2条单向的数据通路。
- 路由ASIC芯片的结构



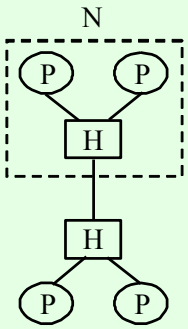
Origin 2000结构

- ASIC芯片的主要功能
  - 选择发送端口和接收端口的最高效连接，动态地切换6个端口的连接。
  - 在CrayLink Interconnect的链路层协议（LLP）控制下与其他路由器和HUB进行可靠通信；
  - 消息的包以虫蚀寻径方式通过路由器以减少通信时延；
  - 对CrayLink信息提供缓存。
- 路由器提供的峰值通信带宽达到9.36GB/s。

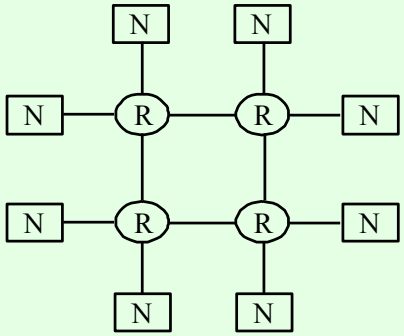
4. 不同的配置和互连

- Origin 2000在不同处理器个数配置情况下的互连拓扑结构
  - 处理器数目：4、16、32、64和128个
  - P：处理器
  - N：结点板
  - H：HUB
  - R：路由器
- 128处理器系统

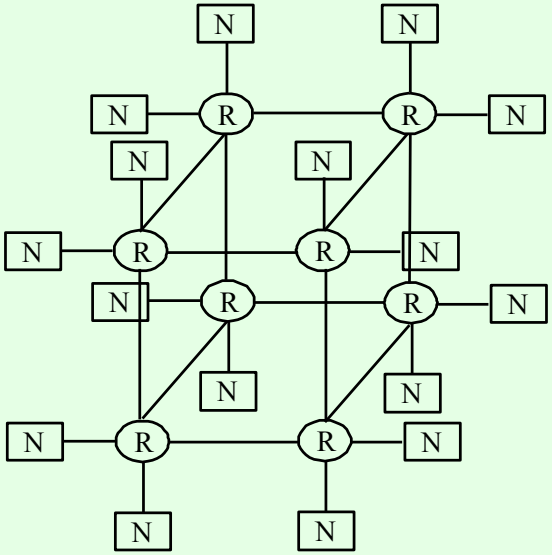
  由4个立方体组成，在立方体之间传送数据多经过了一级路由器。



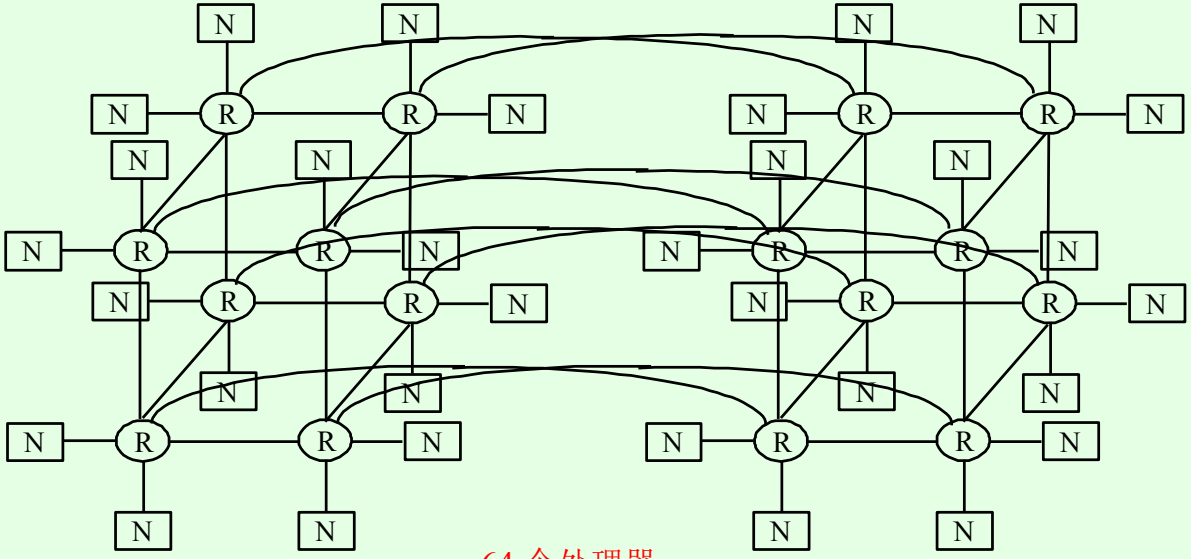
4 个处理器



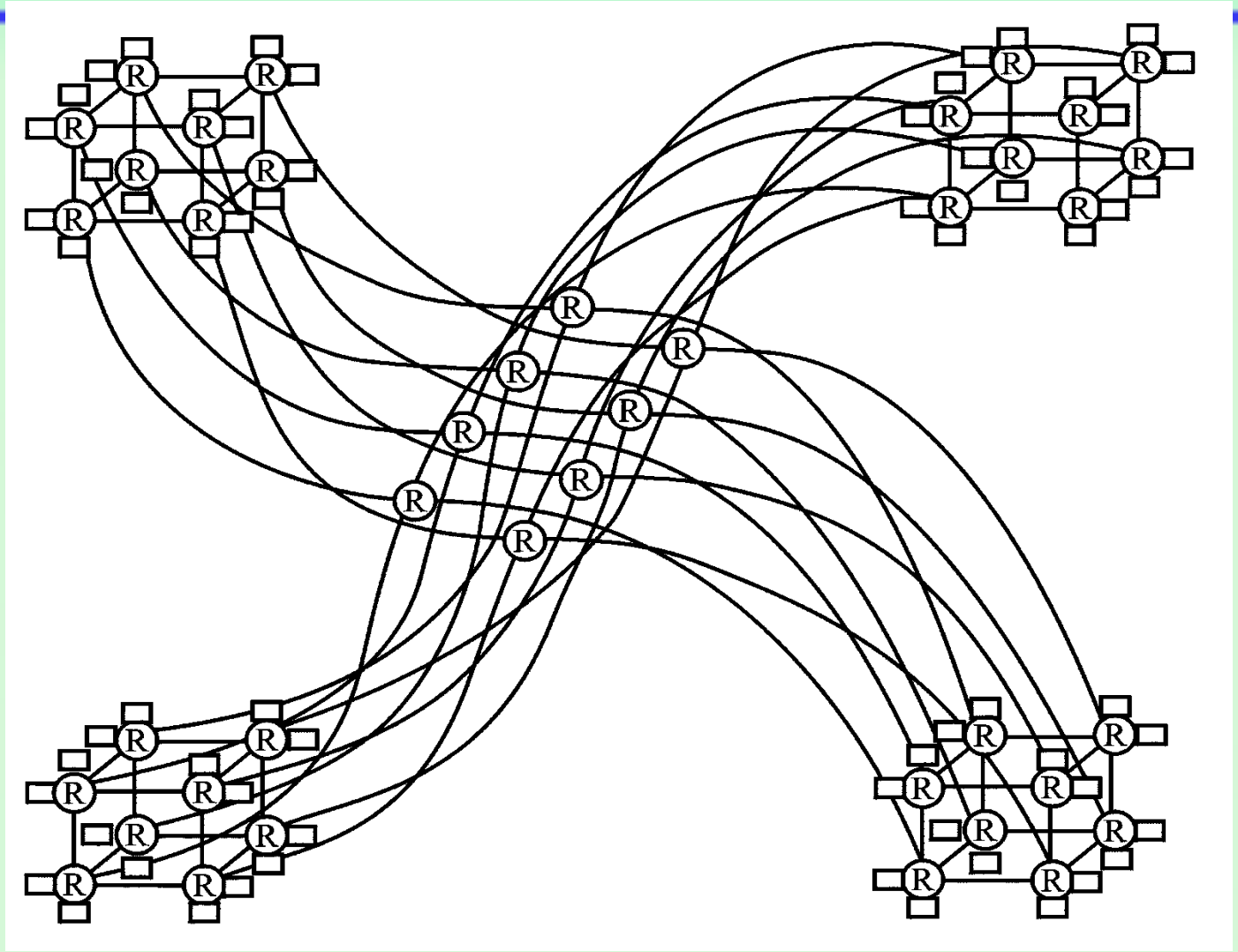
16 个处理器



32 个处理器



64 个处理器



128个处理器

- 在结点内部实现的是SMP（对称多处理器）结构，由于只有两个处理器，所以不存在SMP结构的总线瓶颈问题。
- 在结点之间实现的是大规模并行处理结构，但又解决了共享存储器问题。因此在Origin系统中，无论是访问存储器的时间还是结点间传送数据的带宽都很理想。

5. Origin系统中CPU访问存储器的延迟时间

- 假设：
  - CPU的主频为195MHz
  - Cache不命中
  - 最小延迟时间：CPU访问本结点存储器的时间
  - 最大延迟时间：CPU访问距离最远的存储器的时间

Origin系统中CPU访问存储器的延迟时间

系统CPU数	最小延迟时间	最大延迟时间	平均延迟时间
2	318ns	343ns	343ns
4	318ns	554ns	441ns
8	318ns	759ns	623ns
16	318ns	759ns	691ns
32	318ns	836ns	764ns
64	318ns	1067ns	851ns
128	318ns	1169ns	959ns

6. Origin系统的带宽

每个Hub连到路由器和互连网络的最大频宽为：

1.56Gb/s（全双工，2×780Mb/s）

系统处理器数	带宽 （无快速传送连线）	带宽 （无快速传送连线）
8	1.56Gb/s	3.12Gb/s
16	3.12Gb/s	6.24Gb/s
32	6.24Gb/s	12.5Gb/s
64	12.5Gb/s	--
128	25Gb/s	--

7. 存储层次

寄存器、L1 Cache、L2 Cache和主存储器

- 寄存器和L1 Cache在R10000微处理器中
- 寄存器的存取时间最短
- L1 Cache又分成指令Cache和数据Cache两部分

  （避免取指令和存/取数据发生冲突）
- L2 Cache安装在结点卡中，统一存放指令和数据，由SRAM组成。
- 主存储器地址是统一编址的，每个处理器通过互连网络可访问系统中任一存储单元。

8. 实现Cache的一致性

- 基于目录协议与写作废协议
- 每个结点中，有一个存储器和一个目录存储器。
- 每块对应于一个目录项，每个目录项包含其对应存储器块的状态信息和系统中各Cache共享该存储块情况的位向量，根据位向量可以知道哪些Cache中有其副本。