# 第8章 输入输出系统

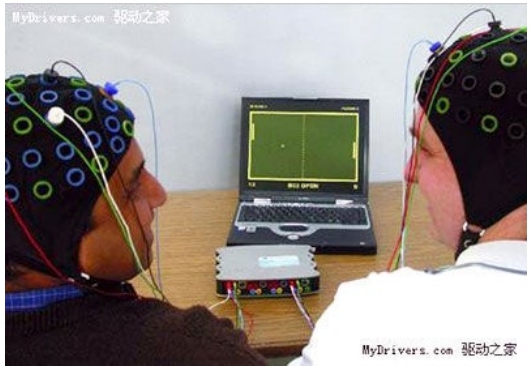
# 输入/输出(I/O，Input/Output）

没有了I/O计算机将毫无用处

- Over time, literally thousands of forms of computer I/O: punch cards to **brain interfaces**

I/O包含的范围非常广泛:

- Secondary/Tertiary storage (flash/disk/tape)
- Network (Ethernet, WiFi, Bluetooth, LTE)
- Human-machine interfaces (keyboard, mouse, touchscreen, graphics, audio, video, neural,…)
- Printers (line, laser, inkjet, photo, 3D, …)
- Sensors (process control, GPS, heartrate, …)
- Actuators (valves, robots, car brakes, …)




I/O设备高度依赖于应用程序

# 8.1 I/O系统的性能

1. 输入/输出系统简称I/O系统

   它包括：

   - 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. 按照主要完成的工作进行分类：
   - 存储I/O系统（本章内容）
   - 通信I/O系统

4. I/O系统的性能对CPU的性能有很大的影响，若两者的性能不匹配，I/O系统就有可能成为整个系统的瓶颈
5. 评价I/O系统性能的参数主要有：
   - 连接特性

     **（哪些I/O设备可以和计算机系统相连接）**
   - I/O系统的容量

     **（I/O系统可以容纳的I/O设备数）**
   - 响应时间和吞吐率等

6. 系统的响应时间(衡量计算机系统的一个更好的指标)
   - 从用户输入命令开始，到得到结果所花费的时间
   - 由两部分构成：
     - **I/O系统的响应时间**
     - **CPU的处理时间**

   多进程技术只能够提高系统吞吐率，并不能够减少系统响应时间

7. 另一种衡量I/O系统性能的方法：考虑I/O操作对CPU执行的打扰(interference)情况
   - 即考查由于其他进程的I/O操作，使得某个进程的执行时间增加了多少

# 8.2 I/O系统的可靠性、可用性和可信性

1. 反映外设可靠性能的参数有：
   - 可靠性（Reliability）
   - 可用性（Availability）
   - 可信性（Dependability）
2. **系统的可靠性**：系统从某个初始参考点开始一直连续提供服务的能力。
   - 用平均无故障时间MTTF来衡量
   - 系统中断服务的时间用平均修复时间MTTR来衡量
   - MTTF的倒数就是系统的失效率
   - 如果系统中每个模块的生存期服从指数分布，则系统整体的失效率是各部件的失效率之和

3. **系统的可用性**：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

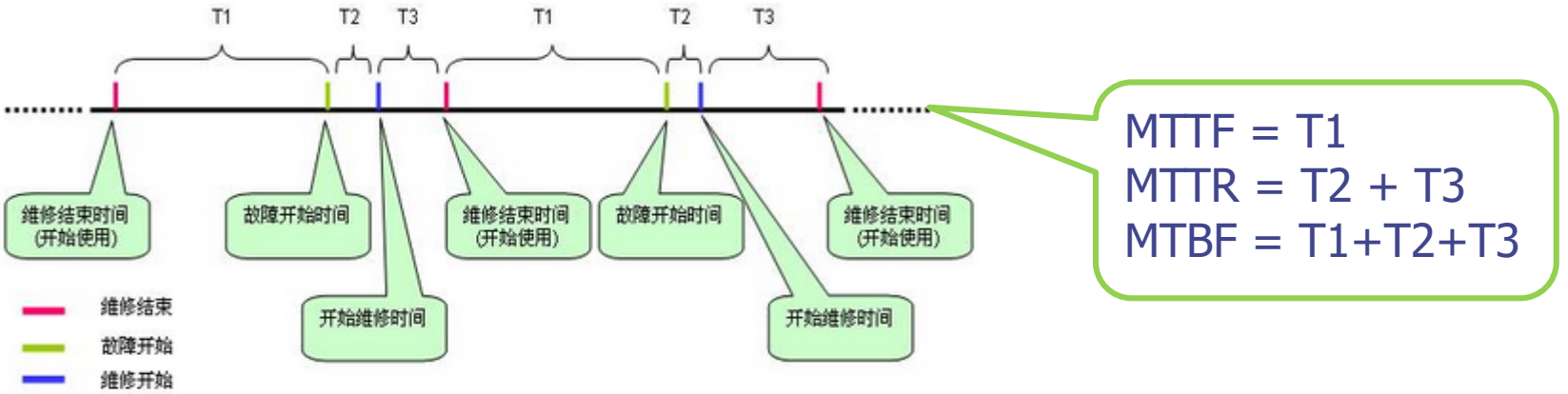
$$可用性=\frac{MTTF}{MTTF+MTTR}$$

N个9

- 平均失效间隔时间MTBF：MTTF+MTTR

（Mean Time Between Failure）

4. **系统的可信性**：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。**（不可以度量）**



# 数据信息的可用性(Availability)

| % Uptime | % Downtime | Downtime per Year | Downtime per Week |
|---|---|---|---|
| 98% | 2% | 7.3 days | 3hrs 22 min |
| 99% | 1% | 3.65 days | 1 hr 41 min |
| 99.8% | 0.2% | 17 hrs 31 min | 20 min 10 sec |
| 99.9% | 0.1% | 8 hrs 45 min | 10 min 5 sec |
| 99.99% | 0.01% | 52.5 min | 1 min |
| **99.999%** | 0.001% | 5.25 min | 6 sec |
| **99.9999%** | 0.0001% | 31.5 sec | 0.6 sec |

## 可信性（Dependability）

Users crave dependable storage, but how do you define it? In the computer industry, it is harder than looking it up in the dictionary. After considerable debate, the following is considered the standard definition (Laprie 1985):

> *Computer system dependability is the quality of delivered service such that reliance can justifiably be placed on this service. The service delivered by a system is its observed actual behavior as perceived by other system(s) interacting with this system's users. Each module also has an ideal specified behavior, where a service specification is an agreed description of the expected behavior. A system failure occurs when the actual behavior deviates from the specified behavior.*

服务等级协议SLA或服务等级目标SLO是否实现

# 可靠性模型

可靠度：R (t)，不可靠度：F (t)，
关系式：R (t) + F (t) = 1。

1. 串联系统


- 系统由n个部件串联而成，其中任何一个部件失效就引起系统的失效，
- 故串联系统寿命等于其中最先失效部件的寿命。

若每个单元的可靠度分别为$R_1, R_2..., R_n$,且每个单元相互独立，则系统可靠度为：

$$R_s = \prod_{i=1}^{n} R_i$$

提高串联系统可靠性：
提高单元可靠度
减少串联单元数目

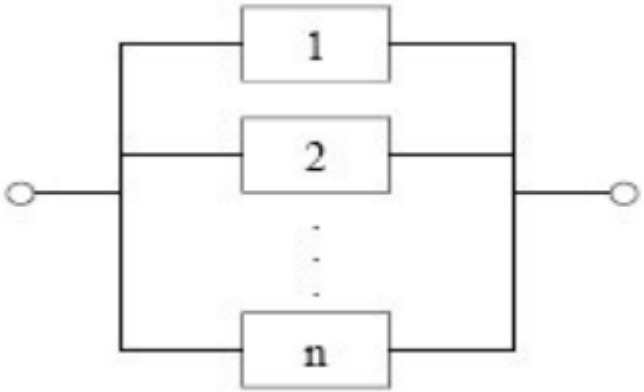
# 可靠性模型

## 2. 并联系统

- 系统由n个部件并联而成，当这n个部件都失效时才引起系统失效。
- 故并联系统寿命等于其中最后失效部件的寿命。

若每个单元的可靠度分别为$R_1, R_2 ..., R_n$,且每个单元相互独立，则系统可靠度为：

$$R_s = 1 - \prod_{i=1}^{n} (1 - R_i)$$




提高并联系统可靠性：

提高单元可靠度

增加并联单元数目

# 可靠性模型

Return

可靠度：R (t)，不可靠度：F (t)，关系式：R (t) + F (t) = 1。

在各单元相同情况下，用n代表串联倍数，m代表并联倍数。

①串并联系统：先串联、后并联，可靠度

$$R(t) = 1-[1-R_i^n(t)]^m$$

②并串联系统：先并联、后串联，可靠度

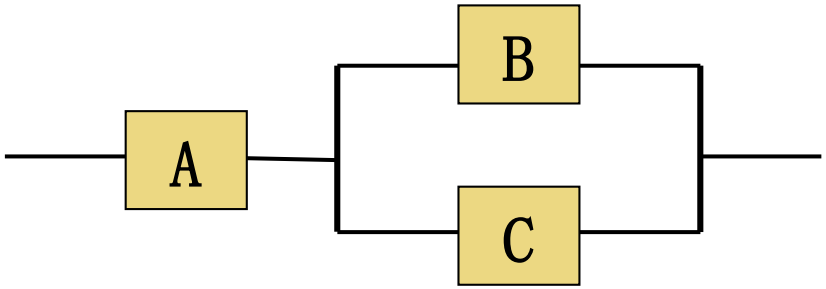
$$R(t) = [1-(1-R_i(t))^m]^n$$

公式用途：RAID 0+1是串并联系统，RAID 1+0是并串联系统

# 可靠性模型

## 3. 举例

◆设一个设备由A、B、C3个相互独立的部件组成，其可靠性框图见下图，这3个部件的可靠度分别为$R_A$=0.9 $R_B$=0.8 $R_C$=0.6，现不考虑其他因素，求系统的可靠度R




$R_{BC}=1-(1-R_B)(1-R_C)$

$=1-0.2*0.4=0.92$

$R=R_AR_{BC}=0.9*0.92=0.828$

例8.1 假设磁盘子系统的组成部件和它们的MTTF如下：

（1）磁盘子系统由10个磁盘构成，每个磁盘的MTTF为1,000,000小时；

（2）1个SCSI控制器，其MTTF为500,000小时；

（3）1个不间断电源，其MTTF为200,000小时；

（4）1个风扇，其MTTF为200,000小时；

（5）1根SCSI连线，其MTTF为1,000,000小时。

假定每个部件的生存期服从指数分布，同时假定各部件的故障是相互独立的，求整个系统的MTTF。

**解：整个系统的失效率为：**

$$\text{系统失效率}=10\times\frac{1}{1000000}+\frac{1}{500000}+\frac{1}{200000}+\frac{1}{200000}+\frac{1}{1000000}=\frac{23}{1000000}$$

**系统的MTTF为系统失效率的倒数，即：**

$$\text{MTTF}=\frac{1000000}{23}=43500\text{小时}$$

**即将近5年。**

**磁盘1,000,000小时(约合114年)无故障，现实中情况如何？**

*Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you?*

------ In Proc. of FAST'07. CMU

- MTTF specified in their datasheets, ranges from 1,000,000 to 1,500,000 hours, suggesting a nominal annual failure rate of at most 0.88%
- annual disk replacement rates typically exceed 1%, with 2-4% common and up to 13% observed on some systems.

## 5. 提高系统组成部件可靠性的方法

- 有效构建方法（valid construction）

  **在构建系统的过程中消除故障隐患，这样建立起来的系统就不会出现故障**

- 纠错方法（error correction）

  **在系统构建中采用容错的方法。这样即使出现故障，也可以通过容错信息保证系统正常工作**

# 8.3 廉价磁盘冗余阵列RAID

1. 磁盘阵列DA（Disk Array）：使用多个磁盘（包括驱动器）的组合来代替一个大容量的磁盘
   - 多个磁盘并行工作
   - 以条带为单位把数据均匀地分布到多个磁盘上

     **（交叉存放）**
   - 条带存放可以使多个数据读/写请求并行地被处理，从而提高总的I/O性能
2. 这里并行性有两方面的含义：

- 多个独立的请求可以由多个盘来并行地处理

  **减少了I/O请求的排队等待时间**

- 如果一个请求访问多个块，就可以由多个盘合作来并行处理

  **提高了单个请求的数据传输率**

3. 问题：阵列中磁盘数量的增加会导致磁盘阵列可靠性的下降

   **如果使用了N个磁盘构成磁盘阵列，那么整个阵列的可靠性将降低为单个磁盘的1/N**

- 解决方法：在磁盘阵列中设置冗余信息盘

  **当单个磁盘失效时，丢失的信息可以通过冗余盘中的信息重新构建**

## 4. 廉价磁盘冗余阵列RAID

Redundant Arrays of Inexpensive Disks

[图书] A case for redundant arrays of inexpensive disks (RAID)
DA Patterson, G Gibson, RH Katz - 1988 - dl.acm.org
Abstract Increasing performance of CPUs and memories will be squandered if not matched by a similar performance increase in I/O. While the capacity of Single Large Expensive Disks (SLED) has grown rapidly, the performance improvement of SLED has been modest. ...
被引用次数：3378 相关文章 所有 176 个版本 引用 保存

- 独立磁盘冗余阵列

Redundant Arrays of Independent Disks

## 5. 大多数磁盘阵列的组成可以由以下两个特征来区分：

- 数据交叉存放的粒度

**（可以是细粒度的，也可以是粗粒度的）**

- **细粒度磁盘阵列是在概念上把数据分割成相对较小的单位交叉存放**

- **优点**：所有I/O请求都能够获得很高的数据传输率
- **缺点**：在任何时间，都只有一个逻辑上的I/O在处理当中，而且所有的磁盘都会因为为每个请求进行定位而浪费时间

- **粗粒度磁盘阵列**是把数据以相对较大的单位交叉存放
  - 多个较小规模的请求可以同时得到处理
  - 对于较大规模的请求又能获得较高的传输率

➢ 冗余数据的计算方法以及在磁盘阵列中的存放方式

6. 在磁盘阵列中设置冗余需要解决以下两个问题：

- 如何计算冗余信息?
  - **大多都是采用奇偶校验码；**
  - **也有采用汉明码（Hamming code）或Reed-Solomon码的**
- 如何把冗余信息分布到磁盘阵列中的各个盘?

  **有两种方法：**
  - **把冗余信息集中存放在少数的几个盘中**
  - **把冗余信息均匀地存放到所有的盘中**

    **（能避免出现热点问题）**

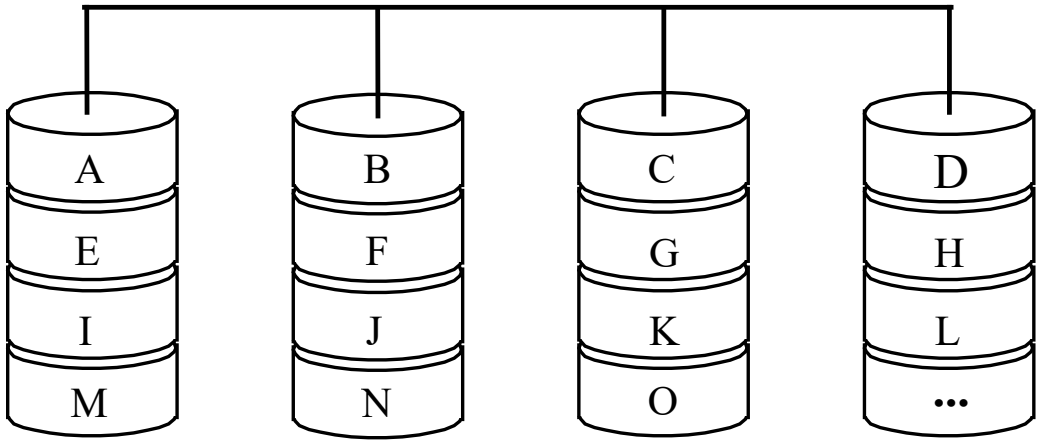
# 7. RAID的分级及其特性

| RAID级别 | 可以容忍的故障个数以及当数据盘为8个时，所需要的检测盘的个数 | 优 点 | 缺 点 | 公司产品 |
|---|---|---|---|---|
| 0<br>非冗余，条带存放 | 0个故障；<br>0个检测盘 | 没有空间开销 | 没有纠错能力 | 广泛应用 |
| 1<br>镜像 | 1个故障；<br>8个检测盘 | 不需要计算奇偶校验，数据恢复快，读数据快。而且其小规模写操作比更高级别的RAID快 | 检测空间开销最大（即需要的检测盘最多） | EMC，<br>HP（Tandem），<br>IBM |
| 2<br>存储器式ECC | 1个故障；<br>4个检测盘 | 不依靠故障盘进行自诊断 | 检测空间开销的级别是$\log_2 m$级（m为数据盘的个数） | 没有 |
| 3<br>位交叉奇偶校验 | 1个故障；<br>1个检测盘 | 检测空间开销小（即需要的检测盘少），大规模读写操作的带宽高 | 对小规模、随机的读写操作没有提供特别的支持 | 外存概念 |

| RAID级别 | 可以容忍的故障个数以及当数据盘为8个时，所需要的检测盘的个数 | 优 点 | 缺 点 | 公司产品 |
| --- | --- | --- | --- | --- |
| 4<br>块交叉奇偶校验 | 1个故障；<br>1个检测盘 | 检测空间开销小，小规模的读操作带宽更高 | 校验盘是小规模写的瓶颈 | 网络设备 |
| 5<br>块交叉分布奇偶校验 | 1个故障；<br>1个检测盘 | 检测空间开销小，小规模的读写操作带宽更高 | 小规模写操作需要访问磁盘4次 | 广泛应用 |
| 6<br>P＋Q双奇偶校验 | 2个故障；<br>2个检测盘 | 具有容忍2个故障的能力 | 小规模写操作需要访问磁盘6次，检测空间开销加倍（与RAID3、4、5比较） | 网络设备 |

## 8.3.1 RAID0

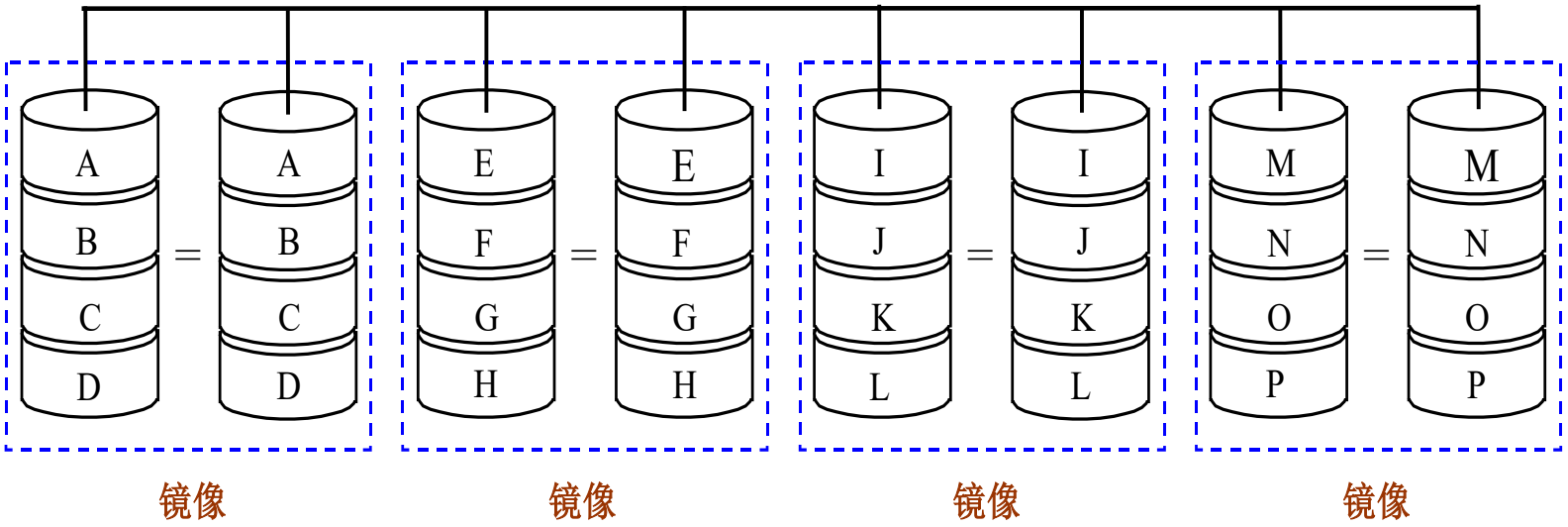
1. 非冗余磁盘阵列，无冗余信息
2. 严格地说，它不属于RAID系列
3. 把数据切分成条带，以条带为单位交叉地分布存放到多个磁盘中



## 8.3.2 RAID1

1. 镜像磁盘，对所有的磁盘数据提供一份冗余的备份。

   - 每当把数据写入磁盘时，将该数据也写入其镜像盘。在系统中所有的数据都有两份。



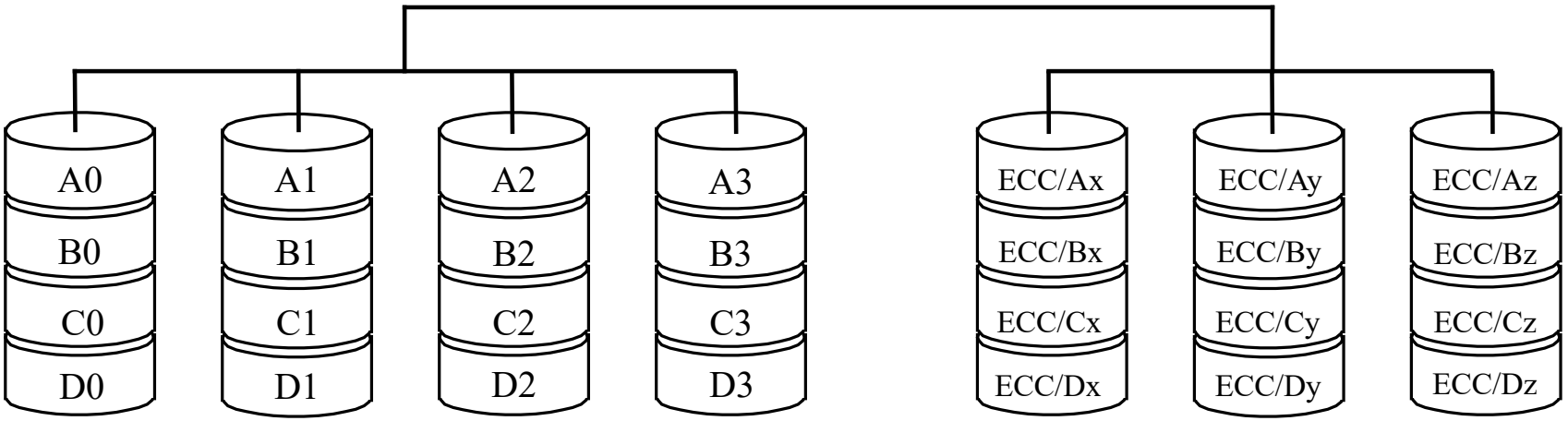
## 2. RAID1的特点

- 能实现快速的读取操作。
- 对于写入操作，镜像的两个磁盘都要写入。但可并行进行，而且不需要计算校验信息，所以其速度比某些级别的RAID快。
- 可靠性很高，数据的恢复很简单。
- 实现成本最高。

## 8.3.3 RAID2

1. 存储器式的磁盘阵列（按汉明纠错码的思路构建）

- 含4个数据盘的RAID2

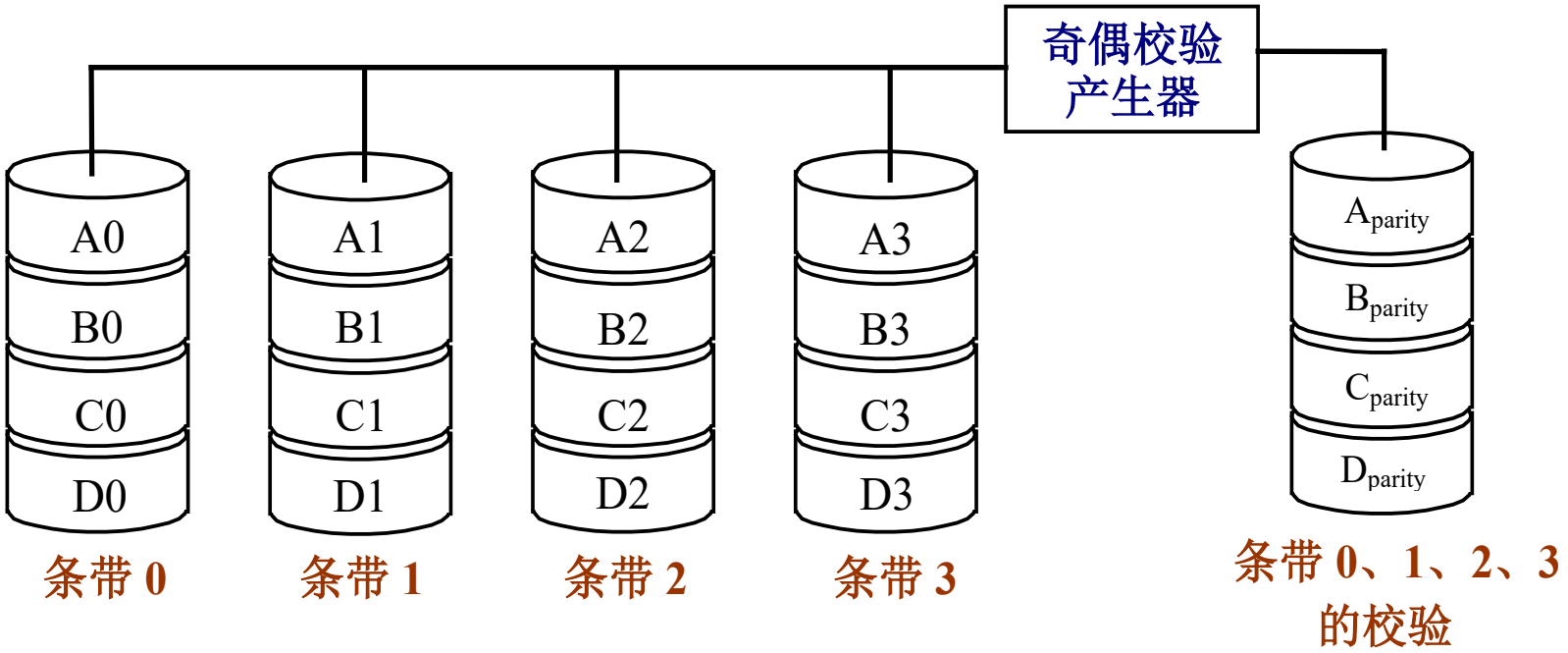



字 A＝A0～A3　字 B＝B0～B3
字 C＝C0～C3　字 D＝D0～D3

字 A 的 ECC＝Ax～Az　字 B 的 ECC＝Bx～Bz
字 C 的 ECC＝Cx～Cz　字 D 的 ECC＝Dx～Dz

## 8.3.4 RAID3

### 1. 位交叉奇偶校验磁盘阵列



## 2. RAID3的特点

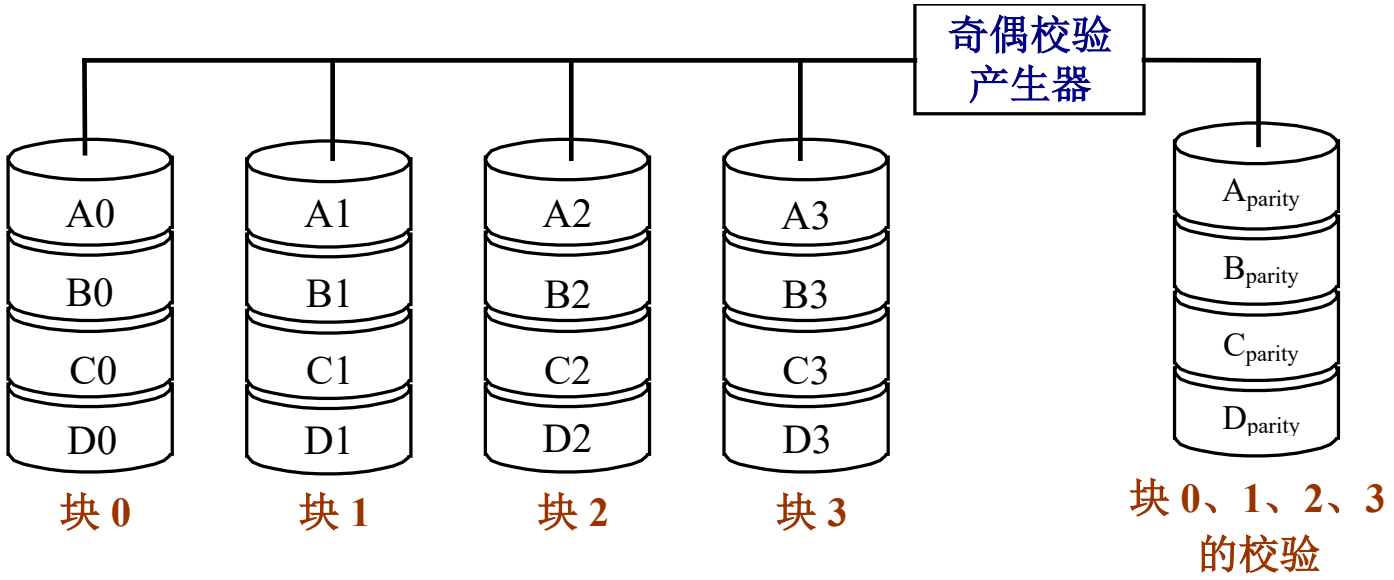
- 采用奇偶校验
  - **写数据时：为每行数据形成奇偶校验位并写入校验盘**
  - **读出数据时：如果控制器发现某个磁盘出故障，就可以根据故障盘以外的所有其他盘中的正确信息恢复故障盘中的数据。（通过异或运算实现）**
- 细粒度的磁盘阵列，即采用的条带宽度较小。

  **（可以是1个字节或1位）**
  - **多个磁盘的并行访问**
  - **不能同时进行多个I/O请求的处理 。**
- 只需要一个校验盘，校验空间开销比较小。

## 8.3.5 RAID4

1. 块交叉奇偶校验磁盘阵列
2. 采用比较大的条带，以块为单位进行交叉存放和计算奇偶校验。

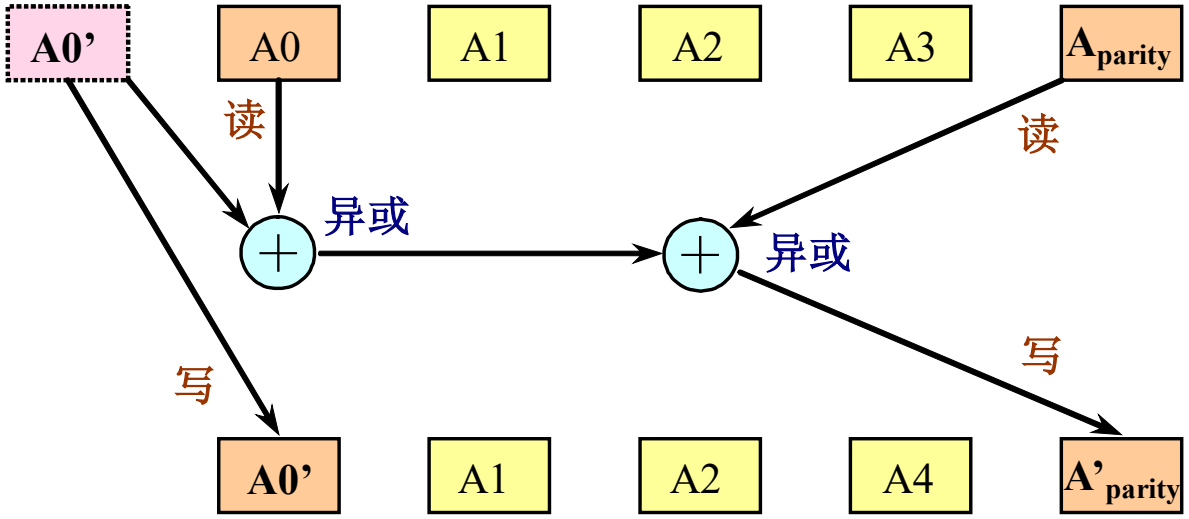
- 实现目标：能同时处理多个小规模访问请求



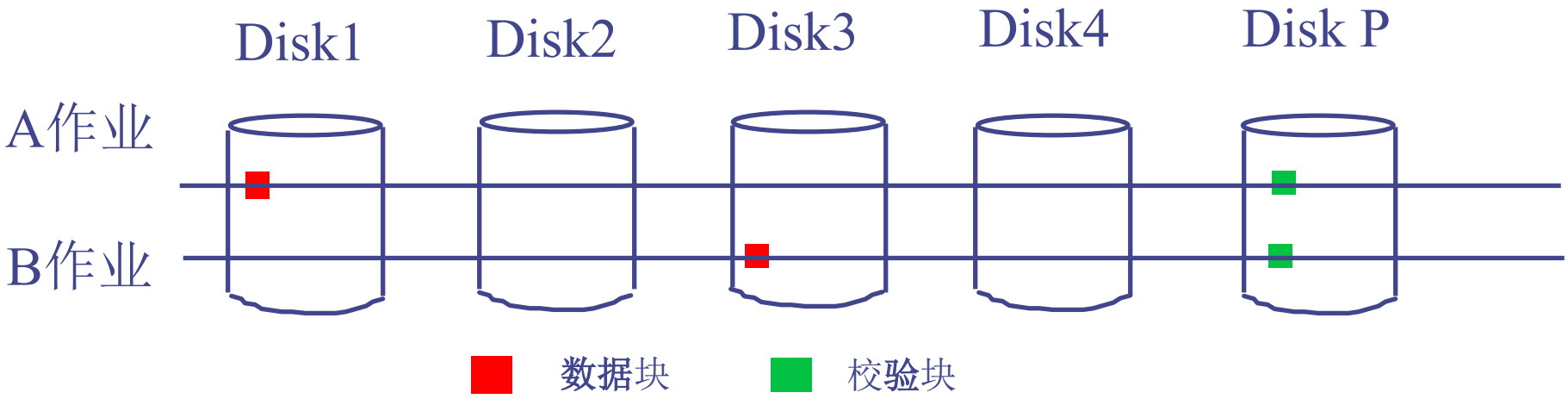
## 3. RAID4读写特点

- 读取操作
  - 每次只需访问数据所在的磁盘。
  - 仅在该磁盘出现故障时，才会去读校验盘，并进行数据的重建。
- 写入操作
  - 假定:有4个数据盘和一个冗余盘。
  - 写数据需要2次磁盘读和2次磁盘写操作。

## 8.3 廉价磁盘冗余阵列RAID



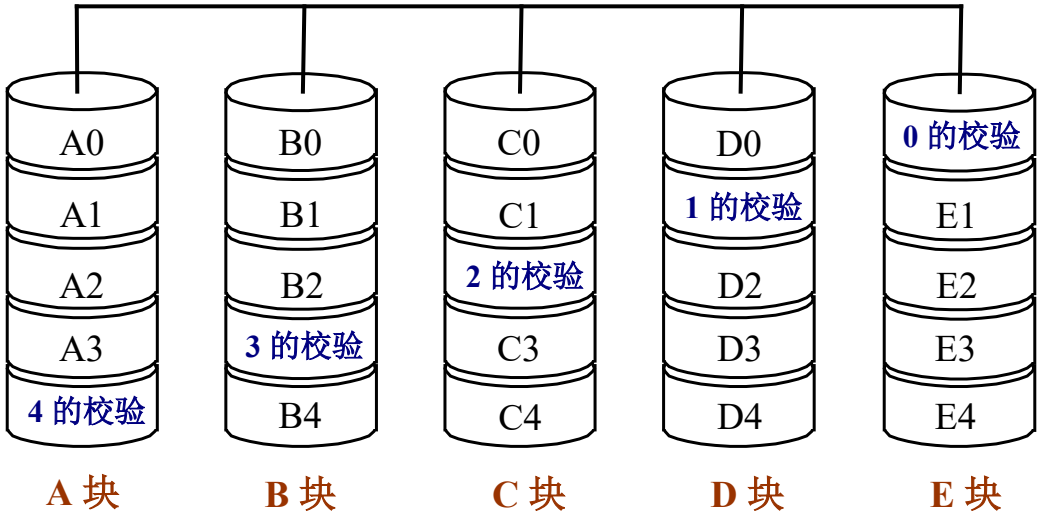
# RAID 3、4瓶颈问题




在同一时刻DISK P只能有一个I/O操作，因此作业 B 只能在作业A完成后才能开始

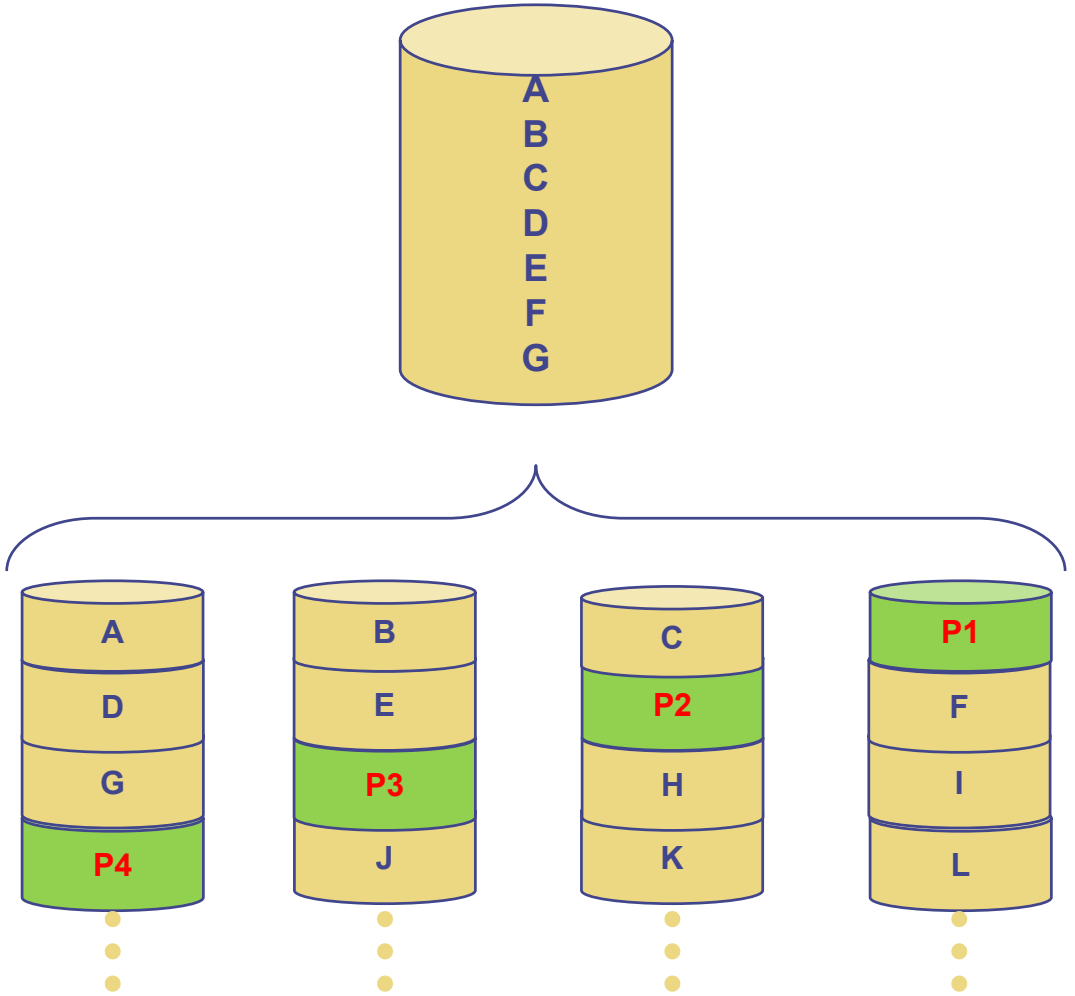
## 8.3.6 RAID5

1. 块交叉分布奇偶校验磁盘阵列
2. 数据以块交叉的方式存于各盘，无专用冗余盘，奇偶校验信息均匀分布在所有磁盘上。



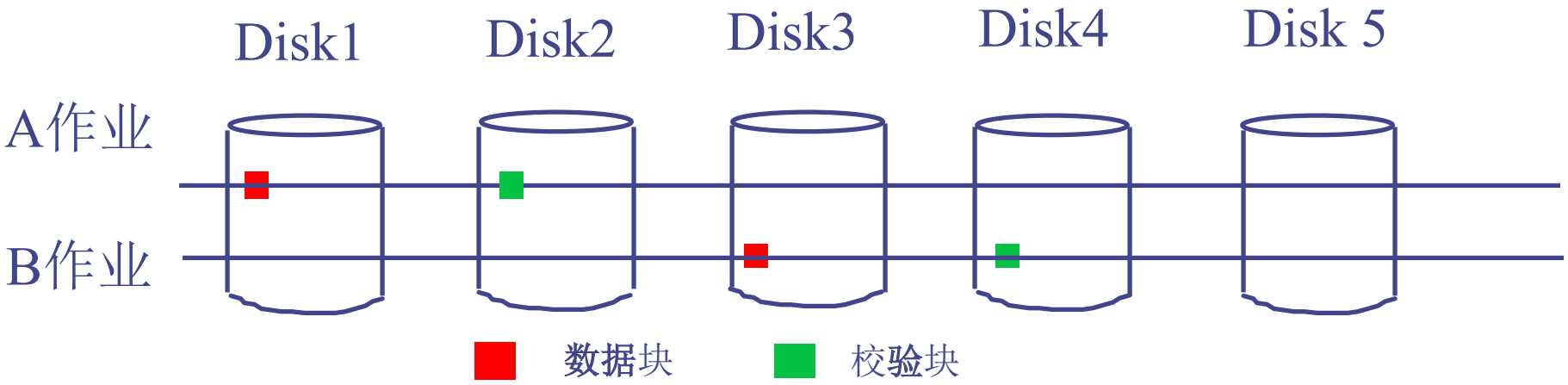
# RAID 5: Rotate Parity

RAID array




Data striped across disks, with parity rotating

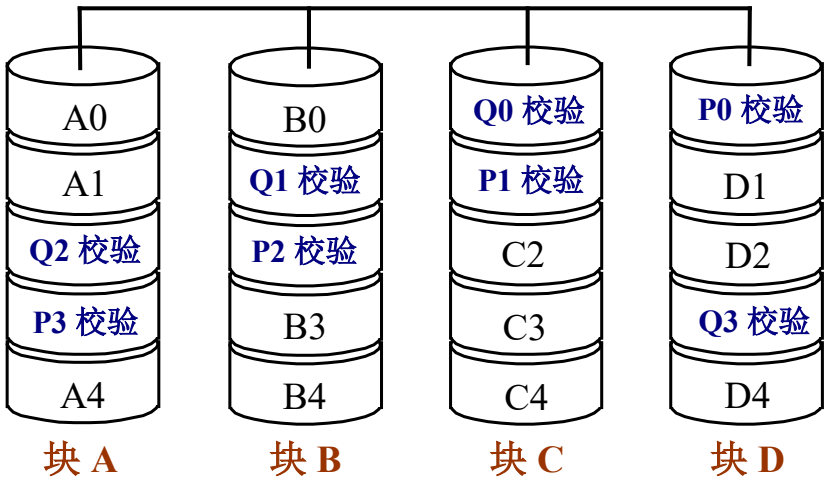
# RAID 5




**在同一时刻可以同时完成作业A及作业B**

## 8.3.7 RAID6

1. P+Q双校验磁盘阵列
2. 特点
   - 校验空间开销是RAID5的两倍
   - 容忍两个磁盘出错
   - 应用越来越广



## RAID5的重建与初始化

1. 重建(Rebuild)：读出非故障盘上的数据，按照校验信息的计算方法计算得到故障盘上的数据，然后写入到新替换的盘上
2. 重构（Reconstruct）：重建的微观操作过程

# 校验信息的正确完整性问题

1. 磁盘上初始信息是随机的
2. 满条块写和大写可以保证校验信息的正确性
3. 若原始校验信息不正确，则小写不能保证校验信息的正确，例如：

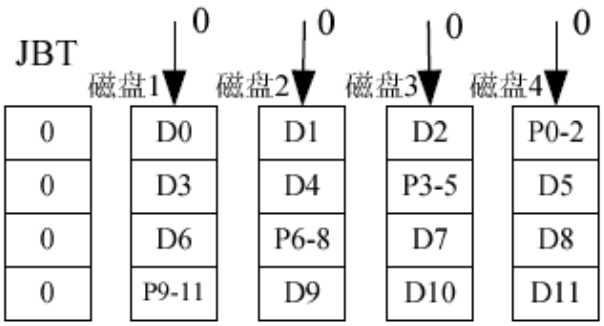
| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | （正确值1） |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | ? | （应该是0） |

根据小写计算得： $0 \oplus 1 \oplus 0 = 1$

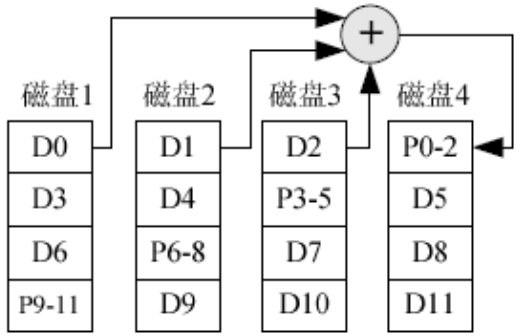
依据错误的校验信息进行重建得出的数据也是不正确的

# RAID5 初始化过程

1. 原则：保证校验信息一致正确
2. 方法：
   - 全部写“0”
   - 计算校验




(b) 改进的数据同步过程

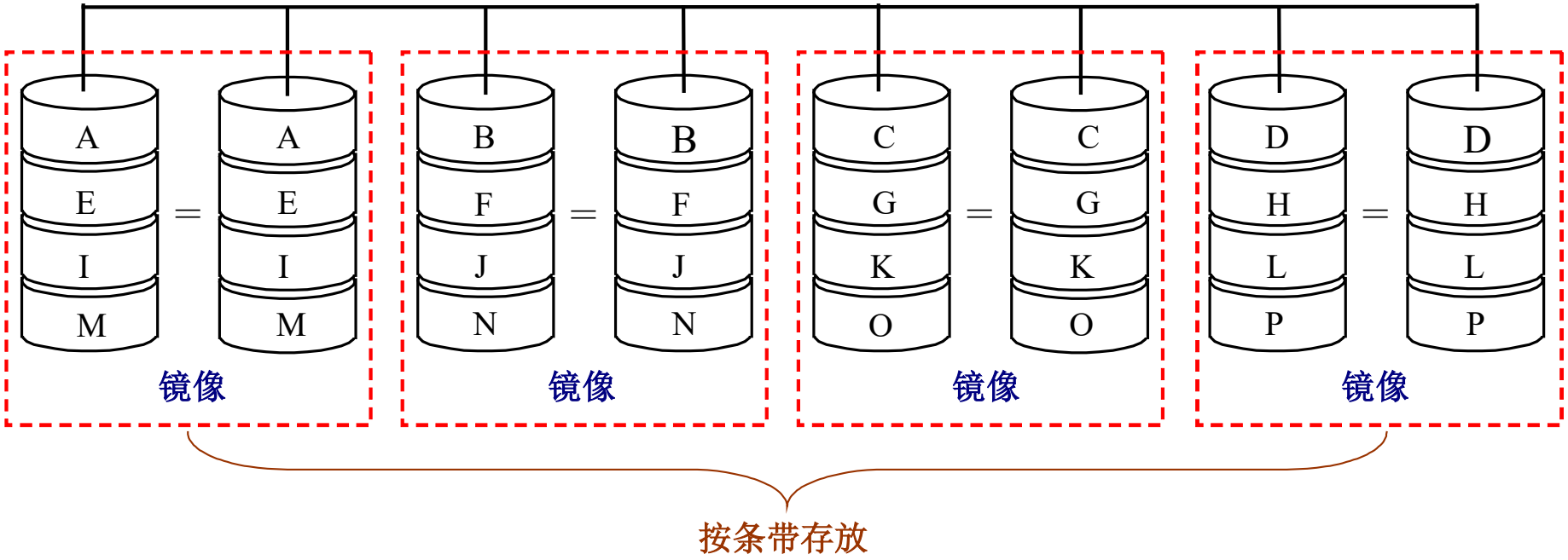



(a) 传统的数据同步过程

## 8.3.8 RAID10与RAID01

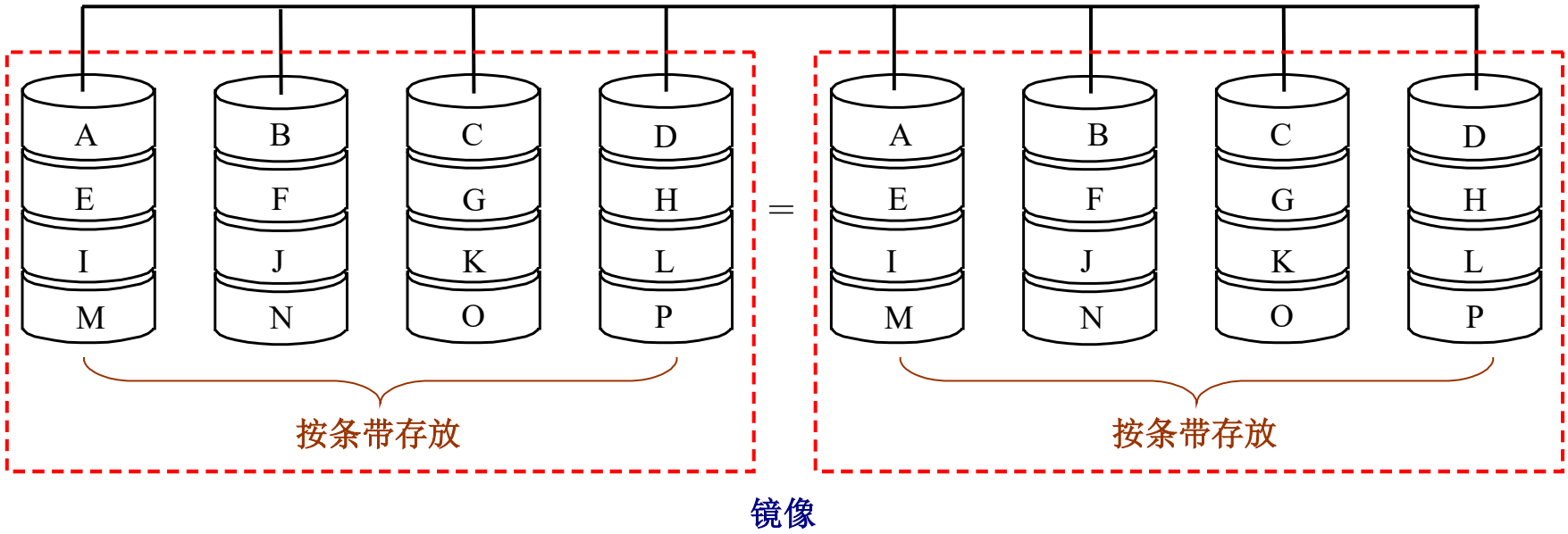
### 1. RAID10又称为RAID1+0

先进行镜像（RAID1），再进行条带存放（RAID0）



## 2. RAID01又称为RAID0+1

先进行条带存放（RAID0），再进行镜像（RAID1）




课堂练习：假设磁盘的可靠度为90%，试比较由8个盘组成的RAID10与RAID01的可靠性并画出各自可靠性框图？

串并可靠性模型

## 8. 有关RAID的几个问题

- 关键问题：如何发现磁盘的故障
  - 在磁盘扇区中除了保存数据信息外，还保存有用于发现该扇区错误的检测信息
- 设计的另一个问题
  - 如何减少平均修复时间MTTR
  - 典型的做法：在系统中增加热备份盘
- 热切换技术
  - 与热备份盘相关的一种技术

## 8.3.9 RAID的实现与发展

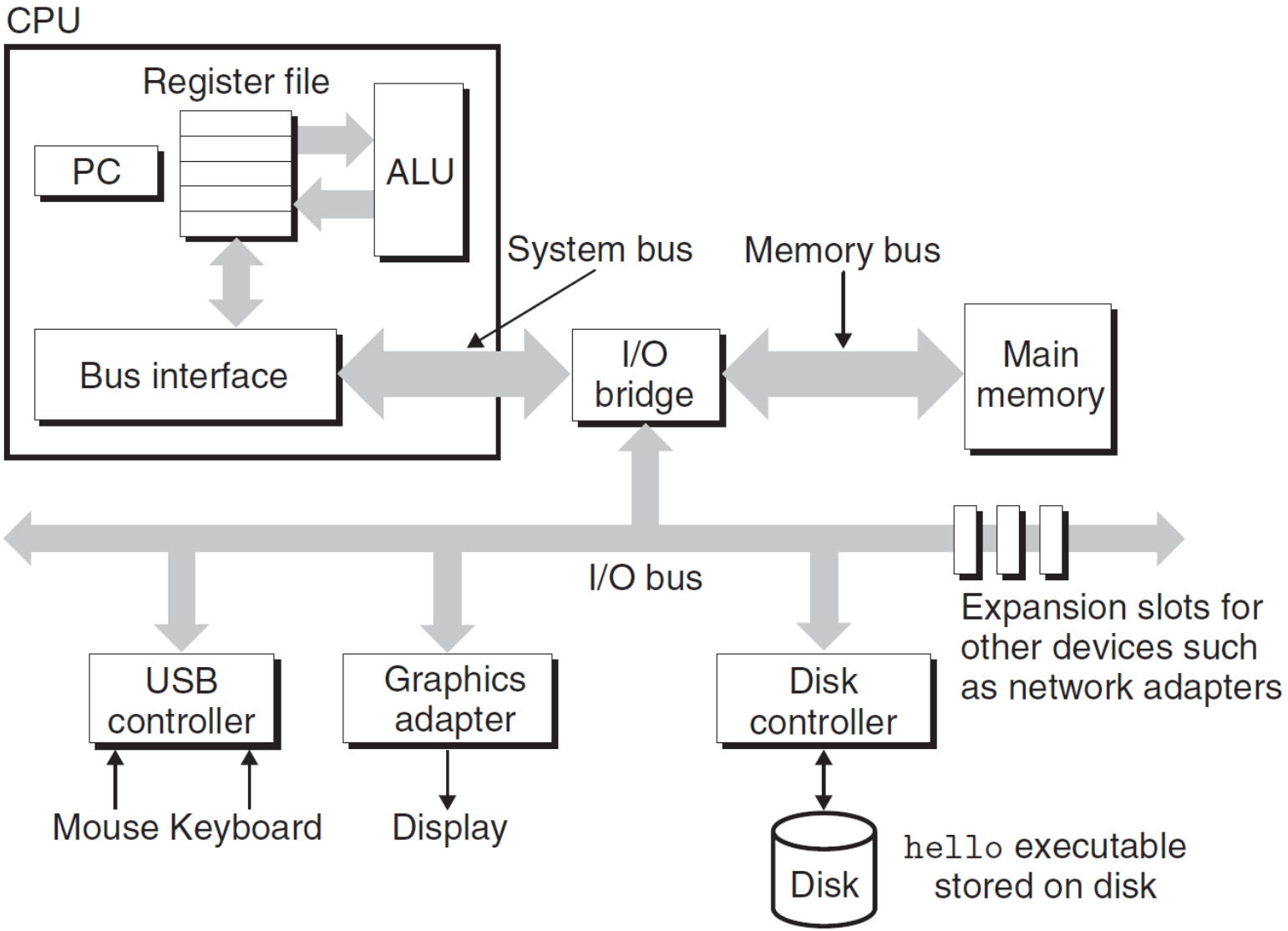
1. 实现盘阵列的方式主要有三种：

- 软件方式：阵列管理软件由主机来实现。
  - 优点：**成本低**
  - 缺点：**过多地占用主机时间，且带宽指标上不去。**
- 阵列卡方式：把RAID管理软件固化在I/O控制卡上，从而可不占用主机时间，一般用于工作站和PC机。
- 子系统方式：一种基于通用接口总线的开放式平台，可用于各种主机平台和网络系统。

# 8.4 总 线

- 在计算机系统中，各子系统之间可以通过总线互相连接。
- 优点：成本低、简单
- 主要缺点：它是由不同的外设分时共享的，形成了信息交换的瓶颈，从而限制了系统中总的I/O吞吐量。

# 典型的计算机系统的硬件组成



## 8.4.1 总线的设计

1. 总线设计存在很多技术难点

- 一个重要原因：总线上信息传送的速度极大地受限于各种物理因素

  **如总线的长度、设备的数目、信号的强度等，这些物理因素限制了总线性能的提高**

另外，我们一方面要求I/O操作响应快，另一方面又要求高吞吐量，这可能造成设计需求上的冲突

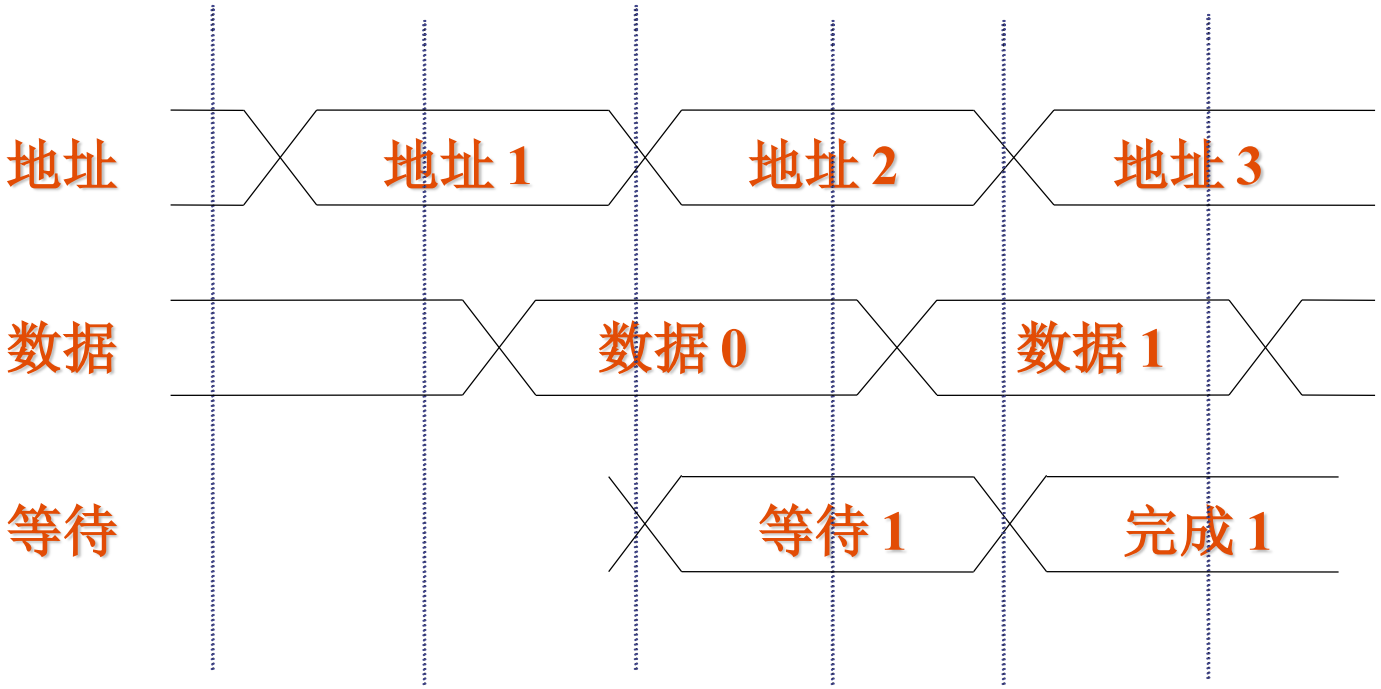
## 2. 设计总线时需要考虑的一些问题

| 特性 | 高性能 | 低价格 |
| --- | --- | --- |
| 总线宽度 | 独立的地址和数据总线 | 数据和地址分时<br>共用同一套总线 |
| 数据总线宽度 | 越宽越快（例如：64位） | 越窄越便宜（例如：8位） |
| 传输块大小 | 块越大总线开销越小 | 单字传送更简单 |
| 总线主设备 | 多个（需要仲裁） | 单个（无需仲裁） |
| 分离事务 | 采用——分离的请求包和<br>回答包能提高总线带宽 | 不采用——持续连接成本<br>更低，而且延迟更小 |
| 定时方式 | 同步 | 异步 |

## 3. 分离事务总线

（又称：流水总线、悬挂总线、包交换总线）

- 在有多个主设备时，可以通过包交换来提高总线带宽
- 基本思想
  - 将总线事务分成请求和应答两部分
  - 在请求和应答之间的空闲时间内，总线可以供其它的I/O使用，这样就不必在整个I/O过程中都独占总线
- 工作过程的示意图




- 分离事务总线有较高的带宽，但是它的数据传送延迟通常比独占总线方法大

## 4. 同步总线

- 同步总线的控制线中包含一个时钟，总线上所有设备的所有的通信操作都以该时钟为基准
- 优点：速度快、成本低
- 缺点
  - **由于时钟通过长距离传输后会扭曲，因而同步总线不能用于长距离的连接。特别是对于高速同步总线来说，更是如此**
  - **总线上的所有设备都必须以同样的时钟频率工作**
- CPU-存储器总线通常是采用同步总线

## 5. 异步总线

- 没有统一的参考时钟，每个设备都有各自的定时方法
- 采用握手协议
- 不存在时钟扭曲和同步的问题，传输距离可以比较长
- 很多I/O总线都采用异步总线
- 同步总线通常比异步总线快

## 8.4.2 总线标准和实例

1. I/O总线标准：定义如何将设备与计算机进行连接的文档
2. 常见I/O总线的一些典型特征

# 几种常用并行I/O总线

| | IDE / Ultra ATA | SCSI | PCI | PCI-X |
|---|---|---|---|---|
| 数据宽度（b） | 16 | 8/16 | 32/64 | 32/64 |
| 时钟频率（MHz） | 最高100 | 10（Fast）<br>20（Ultra）<br>40（Ultra2）<br>80（Ultra3）<br>160（Ultra4） | 33/66 | 66/100/133 |
| 总线主设备数量 | 1个 | 多个 | 多个 | 多个 |
| 峰值带宽（MBps） | 200 | 320 | 533 | 1066 |
| 同步方式 | 异步 | 异步/ | 同步 | 同步 |
| 标准 | 无 | ANSI X3.131 | 无 | 无 |

在嵌入式系统中使用较多的4种串行I/O总线的一些典型特征

| | $I^2C$ | 1-wire | RS-232 | SPI |
|---|---|---|---|---|
| 数据宽度（b） | 1 | 1 | 2 | 1 |
| 信号线数量 | 2 | 1 | 9/25 | 3 |
| 时钟频率（MHz） | 0.4～10 | 异步 | 0.04或异步 | 异步 |
| 总线主设备数量 | 多个 | 多个 | 多个 | 多个 |
| 峰值带宽（Mbps） | 0.4～3.4 | 0.014 | 0.192 | 1 |
| 同步方式 | 异步 | 异步 | 异步 | 异步 |
| 标准 | 无 | 无 | EIA, ITU-T V.21 | 无 |

在服务器系统中使用的CPU-存储器互连系统

| | HP HyperPlane Crossbar | IBM SP | SUN Gigaplane-XB |
|---|---|---|---|
| 数据宽度（b） | 64 | 128 | 128 |
| 时钟频率（MHz） | 120 | 111 | 83.3 |
| 总线的主设备数 | 多个 | 多个 | 多个 |
| 每端口峰值带宽（MBps） | 960 | 1700 | 1300 |
| 总峰值带宽（MBps） | 7680 | 14200 | 10667 |
| 同步方式 | 同步 | 同步 | 同步 |
| 标准 | 无 | 无 | 无 |

| Characteristic | Firewire (1394) | USB 2.0 | PCI Express | Serial ATA | Serial Attached SCSI |
|---|---|---|---|---|---|
| Intended use | External | External | Internal | Internal | External |
| Devices per channel | 63 | 127 | 1 | 1 | 4 |
| Basic data width (signals) | 4 | 2 | 2 per lane | 4 | 4 |
| Theoretical peak bandwidth | 50 MB/sec (Firewire 400) or 100 MB/sec (Firewire 800) | 0.2 MB/sec (low speed), 1.5 MB/sec (full speed), or 60 MB/sec (high speed) | 250 MB/sec per lane (1x); PCIe cards come as 1x, 2x, 4x, 8x, 16x, or 32x | 300 MB/sec | 300 MB/sec |
| Hot pluggable | Yes | Yes | Depends on form factor | Yes | Yes |
| Maximum bus length (copper wire) | 4.5 meters | 5 meters | 0.5 meters | 1 meter | 8 meters |
| Standard name | IEEE 1394, 1394b | USB Implementors Forum | PCI-SIG | SATA-IO | T10 committee |

**FIGURE 6.8 Key characteristics of five dominant I/O standards.** The intended use column indicates whether it is designed to be used with cables external to the computer or just inside the computer with short cables or wire on printed circuit boards. PCIe can support simultaneous reads and writes, so some publications double the bandwidth per lane assuming a 50/50 split of read versus write bandwidth.

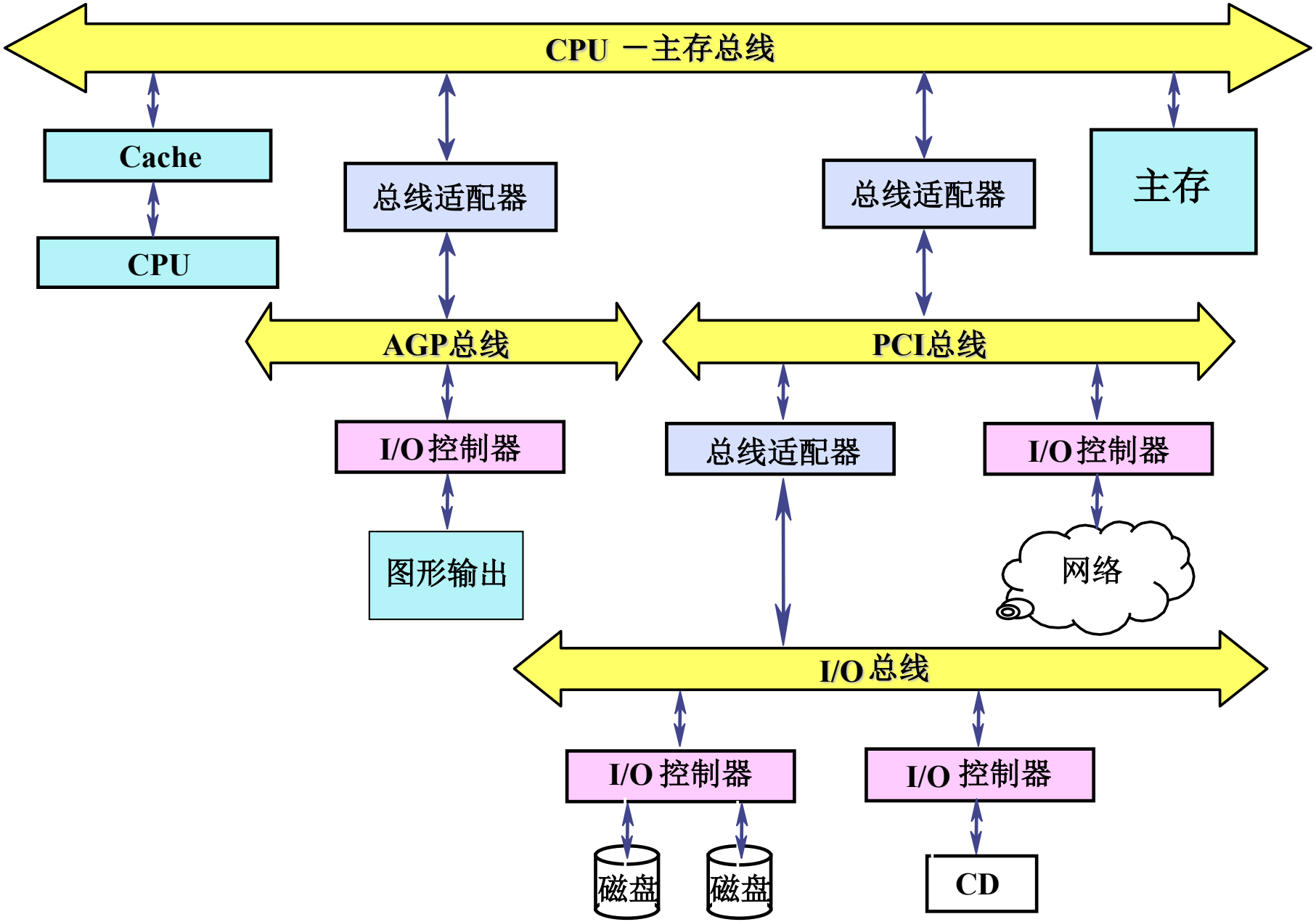
# PCI-Express 接口标准

| 版本 | 数据传输带宽 | 单向单通道带宽 | 双向16通道带宽 | 原始传输率 | 发表日期 |
|---|---|---|---|---|---|
| 1.0 | 2Gb/s | 250MB/s | 8GB/s | 2.5GT/s | 2002年7月22日 |
| 1.0a | 2Gb/s | 250MB/s | 8GB/s | 2.5GT/s | 2003年4月15日 |
| 1.1 | 2Gb/s | 250MB/s | 8GB/s | 2.5GT/s | 2005年3月28日 |
| 2.0 | 4Gb/s | 500MB/s | 16GB/s | 5.0GT/s | 2006年12月20日 |
| 2.1 | 4Gb/s | 500MB/s | 16GB/s | 5.0GT/s | 2009年3月4日 |
| 3.0 | 8Gb/s | 1GB/s | 32GB/s | 8.0GT/s | 2010年11月10日 |
| 4.0 | 16Gb/s | 2GB/s | 64GB/s | 16.0GT/s | 2014年-2015年[3] |

## 8.4.3 与CPU的连接

1. I/O总线的物理连接方式有两种选择
   - 连接到存储器上

     **更常见**
   - 连接到Cache上
2. I/O总线连接到存储器总线上的方式
   - 一种典型的组织结构


CPU —主存总线
Cache
CPU
总线适配器
总线适配器
主存
AGP总线
PCI总线
I/O控制器
总线适配器
I/O控制器
图形输出
网络
I/O总线
I/O 控制器
I/O 控制器
磁盘
磁盘
CD

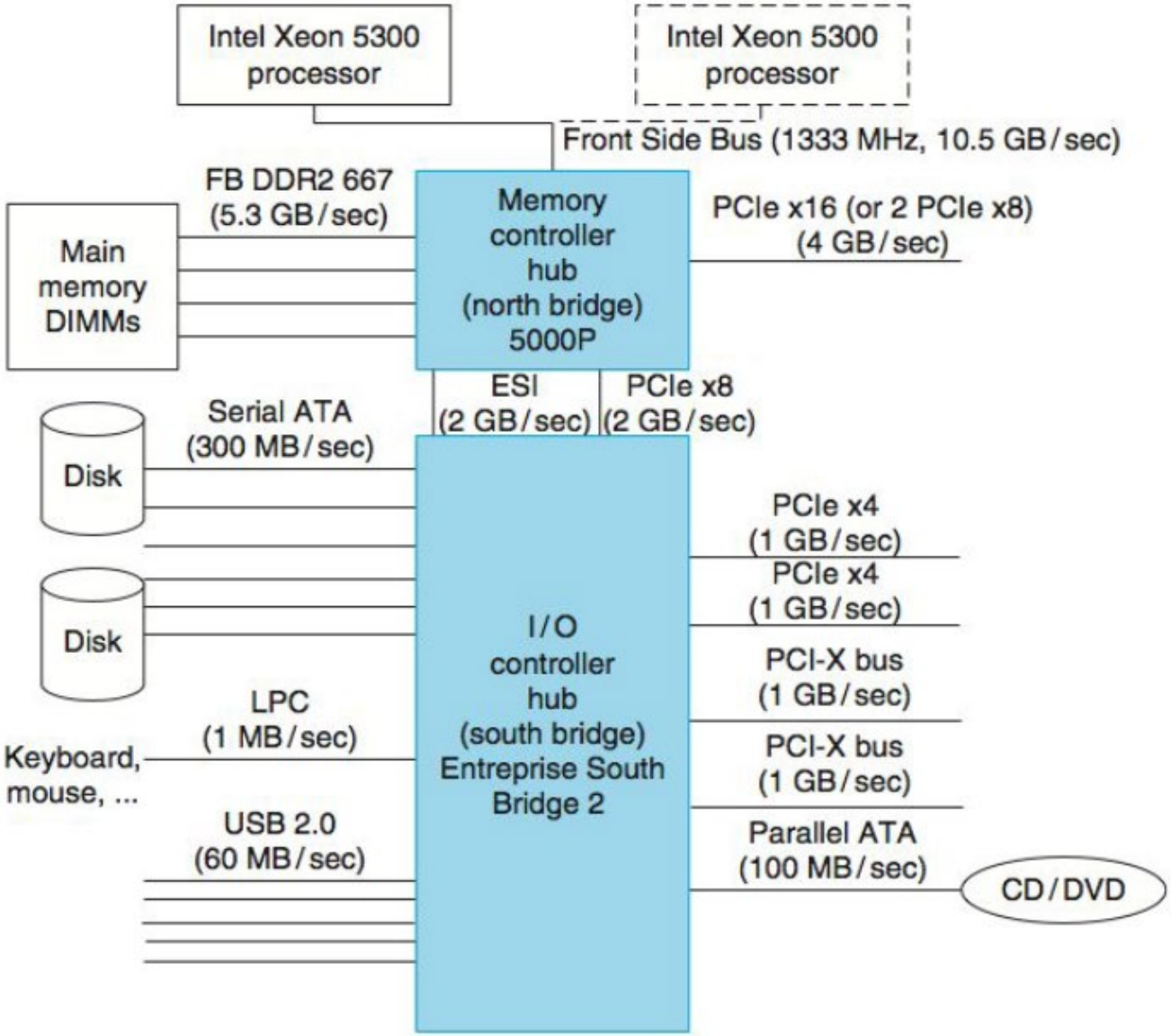
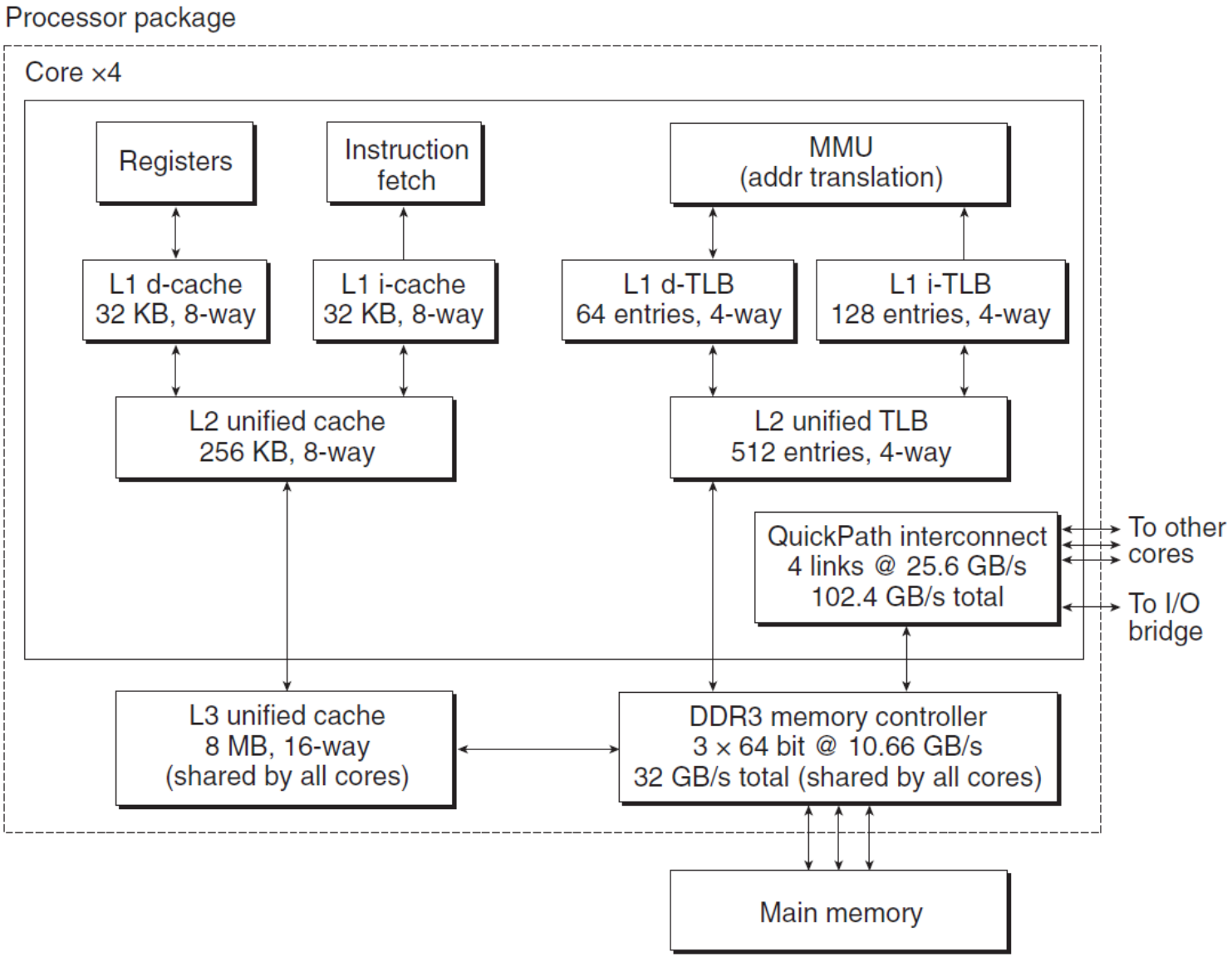



**FIGURE 6.9 Organization of the I/O system on an Intel server using the Intel 5000P chip set.** If you assume reads and writes are each half the traffic, you can double the bandwidth per link for PCIe.

# The Core i7 memory system



## 3. CPU对I/O设备的编址有两种方式

- 存储器映射I/O（也称为I/O设备统一编址）
  - 将一部分存储器地址空间分配给I/O设备，用load指令和store指令对这些地址进行读写将引起I/O设备的数据传输
  - 将一部分存储空间留出用于设备控制，对这一部分地址空间进行读写就是向设备发出控制命令
- 给I/O设备独立编址
  - 需要在CPU中设置专用的I/O指令来访问I/O设备
  - CPU需要发出一个标志信号来表示所访问的地址是I/O设备的地址

4. CPU与外部设备进行输入/输出的方式可分为4种

- 程序查询
- 中断
- DMA
- 通道

## 8.6.1 DMA和虚拟存储器

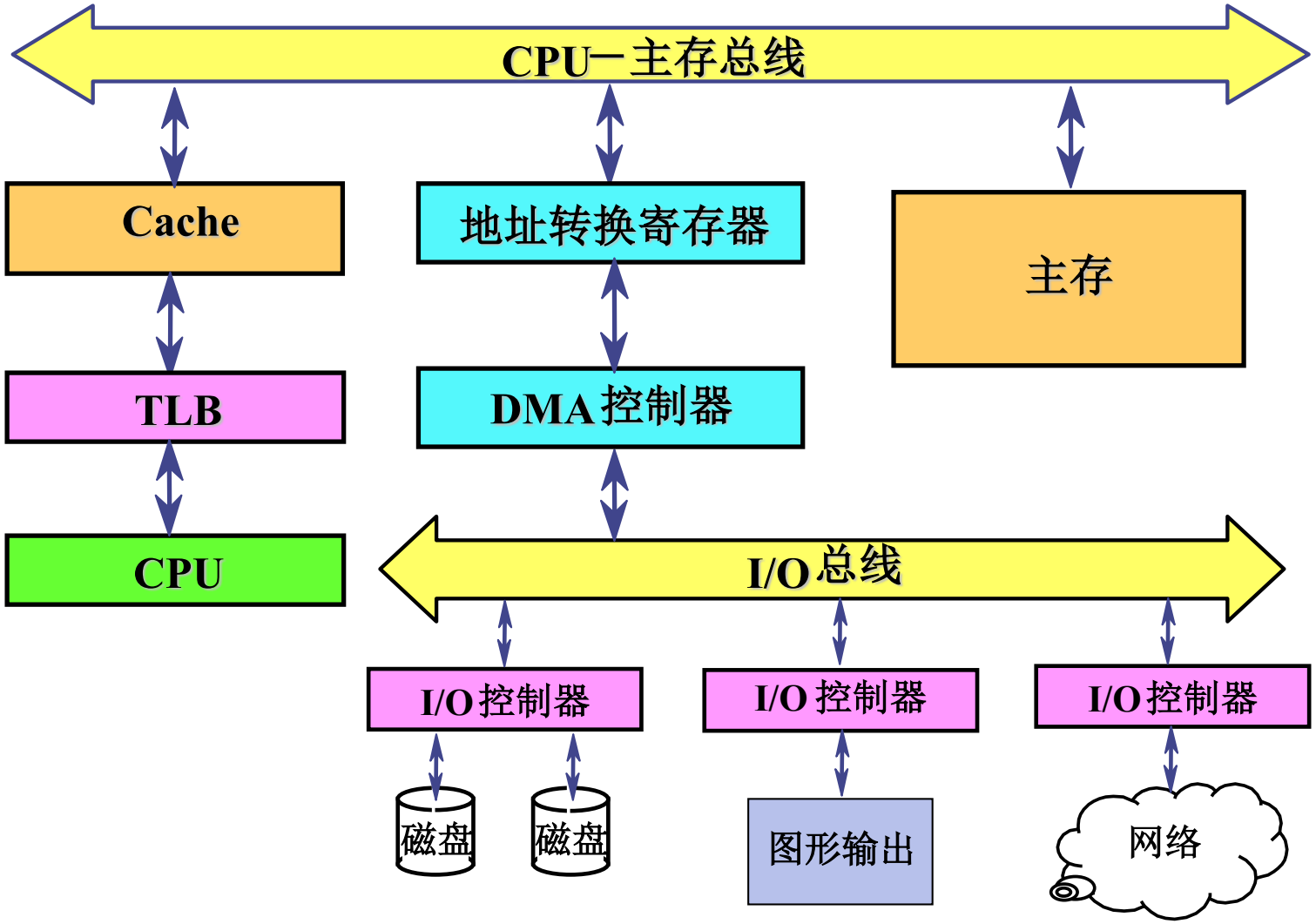
DMA是使用虚拟地址还是物理地址？

1. 使用物理地址进行DMA传输，存在以下两个问题：
   - 对于超过一页的数据缓冲区，由于缓冲区使用的页面在物理存储器中不一定是连续的，所以传输可能会发生问题
   - 如果DMA正在存储器和缓冲区之间传输数据时，操作系统从存储器中移出（或重定位）一些页面，那么，DMA将会在存储器中错误的物理页面上进行数据传输

2. 解决这些问题的方法

- 使操作系统在I/O的传输过程中确保DMA设备所访问的页面都位于物理存储器中，这些页面被称为是钉在了主存中
- “虚拟DMA”技术
  - 允许DMA设备直接使用虚拟地址，并在DMA期间由硬件将虚拟地址转换为物理地址
  - 在采用虚拟DMA的情况下，如果进程在内存中被移动，操作系统应该能够及时地修改相应的DMA地址表

# 虚拟DMA的I/O连接


CPU一主存总线
Cache
地址转换寄存器
主存
TLB
DMA控制器
CPU
I/O总线
I/O控制器
I/O控制器
I/O控制器
磁盘
磁盘
图形输出
网络

## 8.6.2 I/O和Cache数据一致性

1. Cache会使一个数据出现两个副本：

   一个在Cache中，另一个在主存中

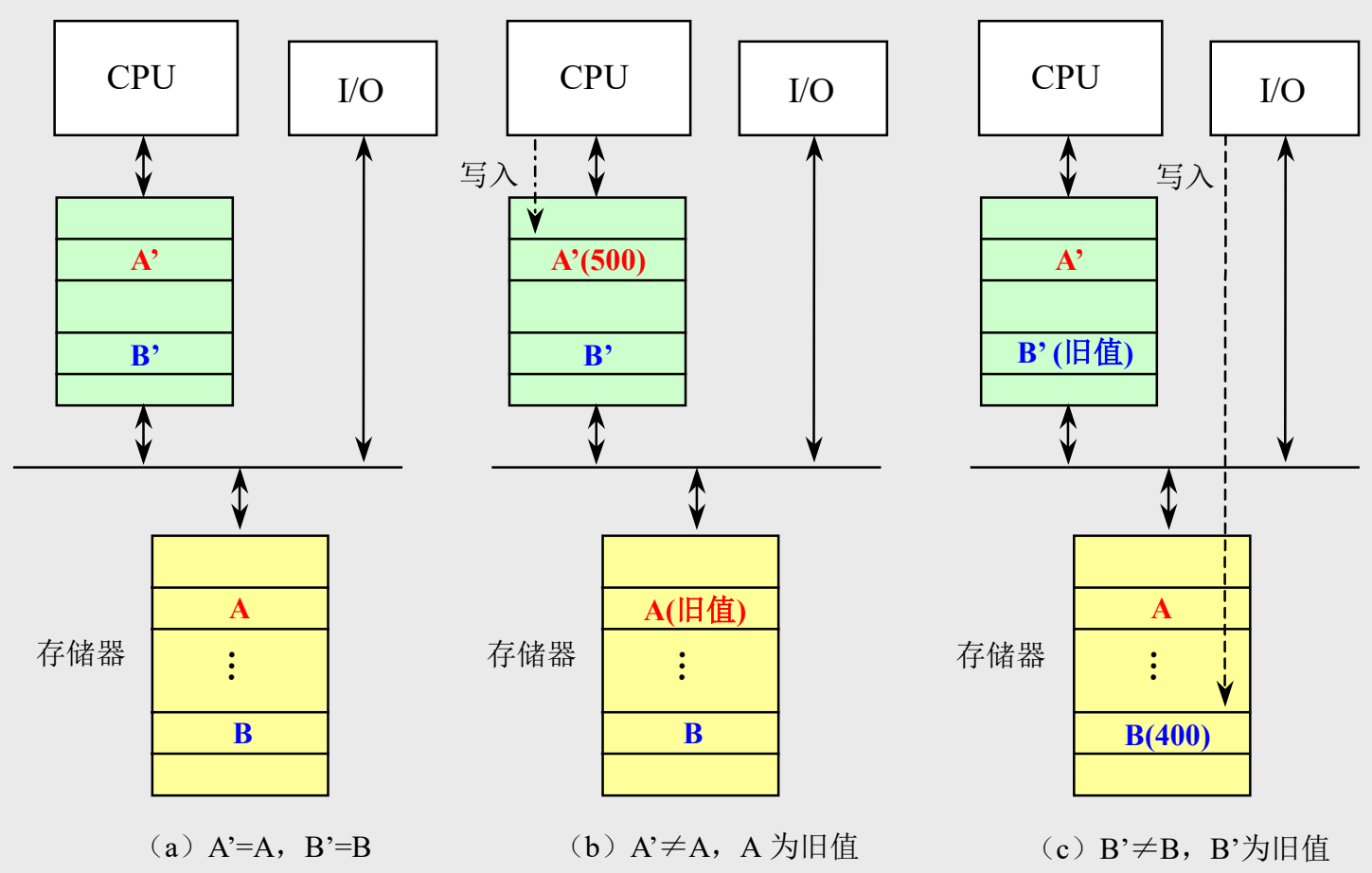
2. I/O设备可以修改存储器中的内容

   - 把I/O连接到存储器上

     **会出现以下情况：**

     - **CPU修改了Cache的内容后，由于存储器的内容跟不上Cache内容的变化，I/O系统进行输出操作时所看到的数据是旧值。（写直达Cache没有这样的问题）**
     - **I/O系统进行输入操作后，存储器的内容发生了变化，但CPU在Cache中所看到的内容依然是旧值**

举例：假设Cache采用写回法，并且A’是A的副本，B’是B的副本。



## 解决内容一致性问题的方法

**（不管Cache是采用写直达法还是写回法）**

- 软件的方法
  - **设法保证I/O缓冲器中的所有各块都不在Cache中**
  - **具体做法有两种**
    - **把I/O缓冲器的页面设置为不可进入Cache的，在进行输入操作时，操作系统总是把输入的数据放到该页面上**
    - **在进行输入操作之前，操作系统先把Cache中与I/O缓冲器相关的数据“赶出”Cache，即把相应的数据块设置为“无效”状态**

- 硬件的方法
  - 在进行输入操作时，检查相应的I/O地址（I/O缓冲器中的单元）是否在Cache中（即是否有数据副本）
  - 如果发现I/O地址在Cache中有匹配的项，就把相应的Cache块设置为“无效”

- 把I/O直接连接到Cache上
  - 不会产生由I/O导致的数据不一致的问题
    - 所有I/O设备和CPU都能在Cache中看到最新的数据
  - I/O会跟CPU竞争访问Cache，在进行I/O时，会造成CPU的停顿
  - I/O还可能会破坏Cache中CPU访问的内容，因为I/O操作可能导致一些新数据被加入Cache，而这些新数据可能在近期内并不会被CPU访问

作业：

8. 11

补充8. 12

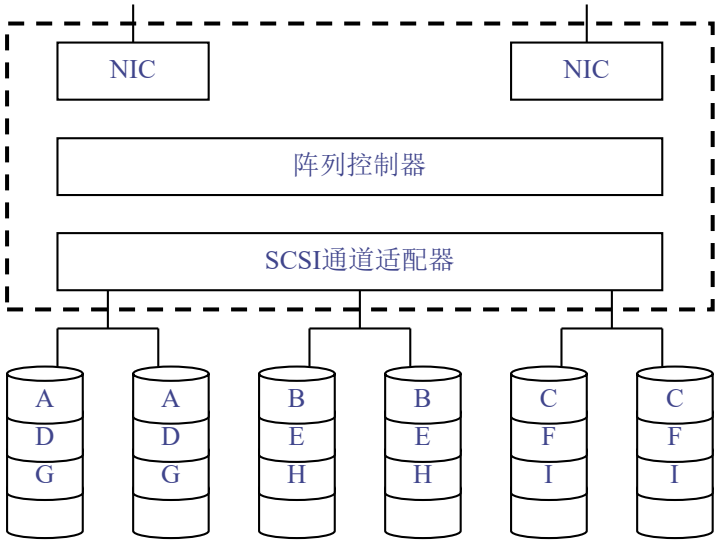
## Homework:习题8.12（补充）

假定某网络型RAID系统包含6个SCSI磁盘，采用RAID 1+0结构，对给定时间t，各部分可靠度为：网络接口通道NIC的R1=0.9，阵列控制器R2=0.95，SCSI通道适配器R3=0.95，磁盘R4=0.8。

（1）画出系统可靠性框图；

（2）写出系统可靠性R的表达式，计算R的数值；

（3）提出进一步增强系统可靠性的若干建议。



有如下假设，分析直接从磁盘上读入一个页到cache中对CPU性能的影响:

1. 页大小为 16 KB, cache块大小 64 bytes.
2. 对应一个新页的地址不在cache中，CPU 不访问新页中的任何数据.
3. 95% 被移除的cache块会被重新访问,这时每次访问都会造成一次cache缺失.
4. cache采用直接映像、写回法, 且50% 的块是脏的.
5. 在写入cache之前I/O系统缓存整个cache块 (this is called a *speed-matching* buffer, matching transfer bandwidth of the I/O system and memory).
6. cache块的访问和访问缺失均服从均匀分布.
7. 在CPU和 I/O系统之间，没有其他访问cache的干扰.
8. 当没有I/O的时候，平均每1,000,000个时钟周期发生15,000 次cache缺失.
9. 不命中的开销是 30个时钟周期, 如果遇到脏块，需要外加 30 个周期把它写回.

现假设每1,000,000个时钟周期处理一个页，试分析I/O对性能的影响。

1. (1). 可靠度
2. 可靠度参数(R(t))主要从一个系统能够正常工作的时间长短来描述系统可靠性。
3. ——定义为：系统在 $t_0$时刻正常工作的条件下,在[$t_0$,t]时间区间内正常工作的概率。

$$R(t) = p\{X > t\}$$

4. 不可靠度为: $F(t) = 1 - R(t)$

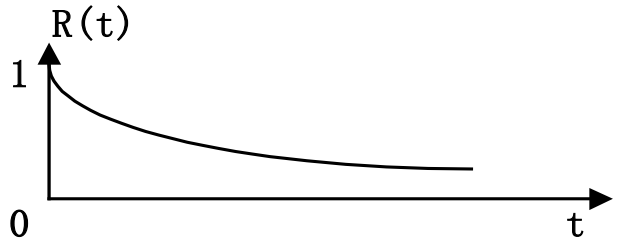
5. 它主要应用于不可修复或极难修复的系统等。在可靠性发展的早期阶段，人们主要使用可靠度参数。

## 可靠性分析的基本理论

1. 可靠度R(t)
   - T大于t的概率（读成“…小时以上存活概率”）。R(t) = P(T＞t)。
   - 事后统计：到t时刻未失效产品与最初产品总数之比。
   - R(t)是单调下降的[0，1]间函数，R(0) = 1，R(∞) = 0

□ 不可靠度F(t)（故障概率）

- T小于t的概率（读成“……小时以内故障概率” 。F(t) = P(T≤t)。
- 事后统计：到t时刻已失效产品与最初产品总数之比。
- F(t)是单调上升的[0，1]间函数，F(0) = 0，F(∞) = 1。





T —— 产品实际寿命长度。损坏前是一未知随机变量。
t —— 希望产品能正常工作到的时间长度。
时间单位：T和t的单位在具体问题除指定外，默认“小时”。

## 可靠性分析的基本理论

1. 故障密度函数f (t)（失效密度）

   - **事后统计：到t时刻在单位时间内发生故障的产品数与最初产品总数之比。**
   - **f(t) =** $\lim_{\Delta t \to 0} \frac{\Delta F(t)}{\Delta t}$ **= F'(t) = - R'(t)，逆式F(t) =** $\int_0^t f(t)dt$

2. 故障率λ(t)（失效率）

   - **事后统计：到t时刻在单位时间内发生故障的产品数与在时刻t时仍正常产品数之比。**
   - λ(t) = f (t) / R (t) = F'(t) / R (t) = -R'(t) / R (t)

3. 平均寿命MTTF，又记为E(T)

   - **产品寿命T的数学期望，是f(t)对时间的积分平均值。**
   - **E(T) =** $\int_0^\infty t \cdot f(t)dt$ **=** $\int_0^\infty R(t)dt$

## 可靠性基本理论——指数分布

1. 在一个系统整个寿命周期中，系统失效率随时间的变化规律可用浴盆曲线来描述。

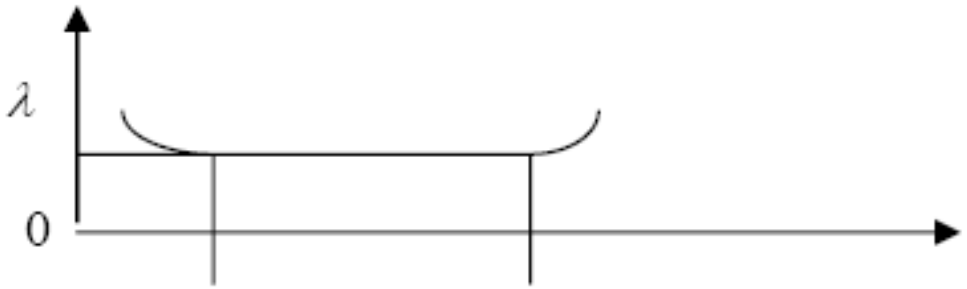



图 2.3　浴盆曲线

系统寿命周期分三阶段：第一阶段早期故障期，常又称调试期。随着调试进行，早期故障不断排除，进入第二阶段随机故障期。这一时期是正常工作时期，其失效率不随时间变化而变化。随着系统运行时间越来越长，失效率不断增大，系统进入损耗故障期。

## 可靠性基本理论——指数分布

1. 随机故障期是系统实际使用期，也是系统可靠性建模和分析最关心的时期。这期间系统失效率基本稳定.

2. 随机故障期的系统可靠度函数服从指数分布规律，

3. $R(t) = e^{-\lambda t}$

4. 若产品寿命T的故障密度函数为$f(t)= \lambda e^{-\lambda t}$ ($\lambda > 0$，$t \geq 0$)，则称T服从参数为$\lambda$的指数分布。

5. 这是系统可靠性建模和分析中很重要的一个特性和假设前提。

## 可靠性基本理论——指数分布

1. ① 可靠度$R(t) = 1-F(t) = e^{-\lambda t}$ $(t\geq 0)$
2. ② 不可靠度$F(t) = \int_0^t f(t)dt = 1-e^{-\lambda t}$ $(t\geq 0)$
3. ③ 故障密度函数为$f(t) = \lambda e^{-\lambda t}$ $(\lambda > 0,\ t\geq 0)$
4. ④ 故障率$\lambda(t) = f(t)/R(t) = \lambda$
5. ⑤ 平均寿命$MTTF = \int_0^\infty R(t)dt = 1/\lambda$

1. （1）. 串联系统

◆假设这n个部件的工作是相互独立的，第i个部件寿命为Ti，可靠度为Ri（t）=P{Ti>t},失效率λi(t).

◆T =min(T1，T2，…，Tn）

◆可靠度R（t）=P{T>t}=∏P{Ti>t}=∏Ri（t） i=1..n

$$\text{可靠度}R(t) = \prod_{i=1}^{k} R_i(t)$$

1. (1) 串联系统

若各单元的寿命服从指数分布，则串联系统的寿命也服从指数分布。

① 可靠度$R(t) = e^{-\lambda t}$

② 不可靠度$F(t) = 1-e^{-\lambda t}$

③ 故障密度函数为$f(t) = \lambda e^{-\lambda t}$

④ 故障率$\lambda(t) = \sum\lambda i(t) = \lambda$，$i=1..n$

⑤ 平均寿命$MTTF=E[T]=1/\sum\lambda i = 1/\lambda$，$i=1..n$

- （2）.并联系统

◆假设这m个部件的工作是相互独立的，第i个部件寿命为Ti，可靠度为Ri（t）=P{Ti>t},失效率λi(t).

◆T =max(T1, T2, …, Tm)

◆可靠度R（t）=P{T>t}=1-∏[1-Ri（t）]= $\prod_{i=1}^{m}[1-R_i(t)]$ i=1..m.

◆不可靠度F(t)= $\prod_{i=1}^{m}F_i(t)$

# 可靠性基本理论——串联、并联、混联系统的可靠性

1. (2) 并联系统
2. 各单元寿命服从指数分布时，并联系统寿命并不服从指数分布。如果各单元相同，则
3. ① 可靠度$R(t) = 1-(1-e^{-\lambda it})^m$
4. ② 不可靠度$F(t) = (1-e^{-\lambda it})^m$
5. ③ 故障密度函数$f(t) = F'(t)$
6. ④ 故障率$\lambda(t)= f(t)/R(t) = F'(t)/R(t) = -R'(t)/R(t)$
7. ⑤ 平均寿命MTTF = $\int_0^{\infty} 1-\left(1-e^{-\lambda_i t}\right)^m dt = \frac{1}{\lambda}\sum_{i=1}^{m}\frac{1}{i}$