

第六章 存储器层次结构

- 教 师： 郑贵滨
- 听觉智能研究中心
- 哈尔滨工业大学，计算机科学与技术学院

主要内容

- 存储技术及其趋势
- 局部性
- 存储器层次结构中的高速缓存

随机访问存储器(RAM)

■ 关键特征

- RAM封装在芯片上
- 基本存储单位是一个单元(cell)，每单元存储 1 比特
- 多个RAM芯片一起组成一个存储器

■ RAM 分为两类：

- SRAM (静态RAM)
- DRAM (动态 RAM)

SRAM vs DRAM一览

	每位晶体管数	相对访问时间	需要刷新?	需要差错检测电路(EDC)?	相对花费	应用
SRAM	4 或 6	1X	否	可能	100x	高速缓存存储器
DRAM	1	10X	是	是	1X	主存、帧缓冲区

增强的DRAMs

- DRAM自1966年问世以来，其基本单元就没有变化。
 - Intel 于1970年将其推向市场
- DRAM 集成了更好接口逻辑、更快的I/O传输接口：
 - 同步 DRAM (SDRAM)
 - 使用与内存控制器相同的时钟信号，取代异步控制信号
 - 允许行地址复用(比如, RAS, CAS, CAS, CAS)

增强的DRAMs

- DRAM 集成了更好接口逻辑、更快的I/O传输接口(...)
 - 双倍数据速率同步DRAM (Double Data-Rate SDRAM, DDR SDRAM)
 - 每个时钟周期每个引脚使用两个时钟沿传送两比特控制信号
 - 以预取缓冲区的大小来划分不同类型:
DDR (2 bits), DDR2 (4 bits), DDR3 (8 bits)
 - 到2010年, 多数服务器和桌面系统均支持该标准
 - Intel Core i7 支持DDR3 和 DDR4 SDRAM

非易失性存储器

■ DRAM 和 SRAM 是易失性存储器

- 断电数据丢失

■ 非易失性存储器断电后，依然保持数据

- 只读存储器(**ROM**): 生产时写入程序，只能写一次
- 可编程 ROM (**PROM**): 可以重新编程一次
- 可擦除 PROM (**EPROM**): 可用紫外线整块擦除
- 电可擦除PROM (**EEPROM**): 可用电子信号整块擦除
- 闪存: 基于EEPROM, 以块为单位进行擦除
 - 100,000 次擦除后即磨损坏

非易失性存储器

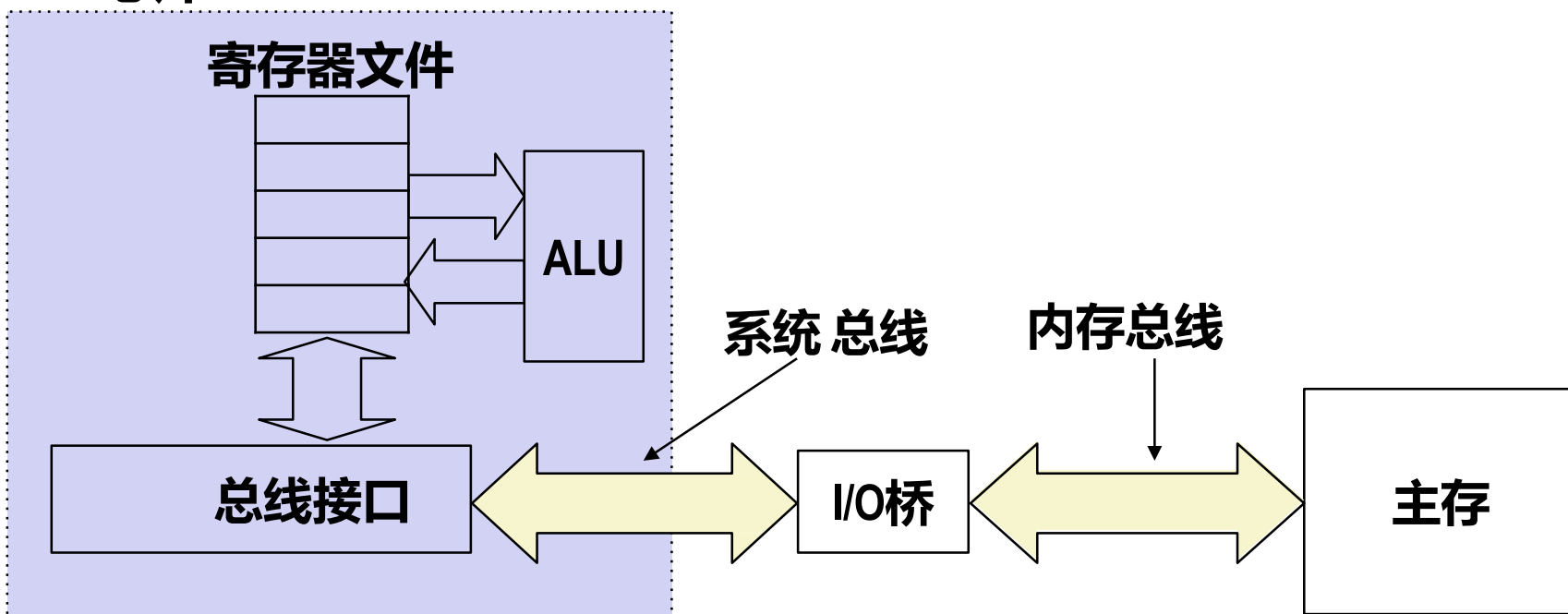
■ 非易失性存储器的应用

- 存储固件程序的ROM(BIOS,磁盘控制器, 网卡,图形加速器, 安全子系统,...)
- 固态硬盘(U盘, 智能手机, mp3播放器, 平板电脑, 笔记本电脑...)
- 磁盘高速缓存

连接CPU和存储器的典型总线结构

- 一条总线(**bus**)是由多条并排的电线组成的一束线，其传输地址、数据和控制信号
- 多个设备共享多条总线

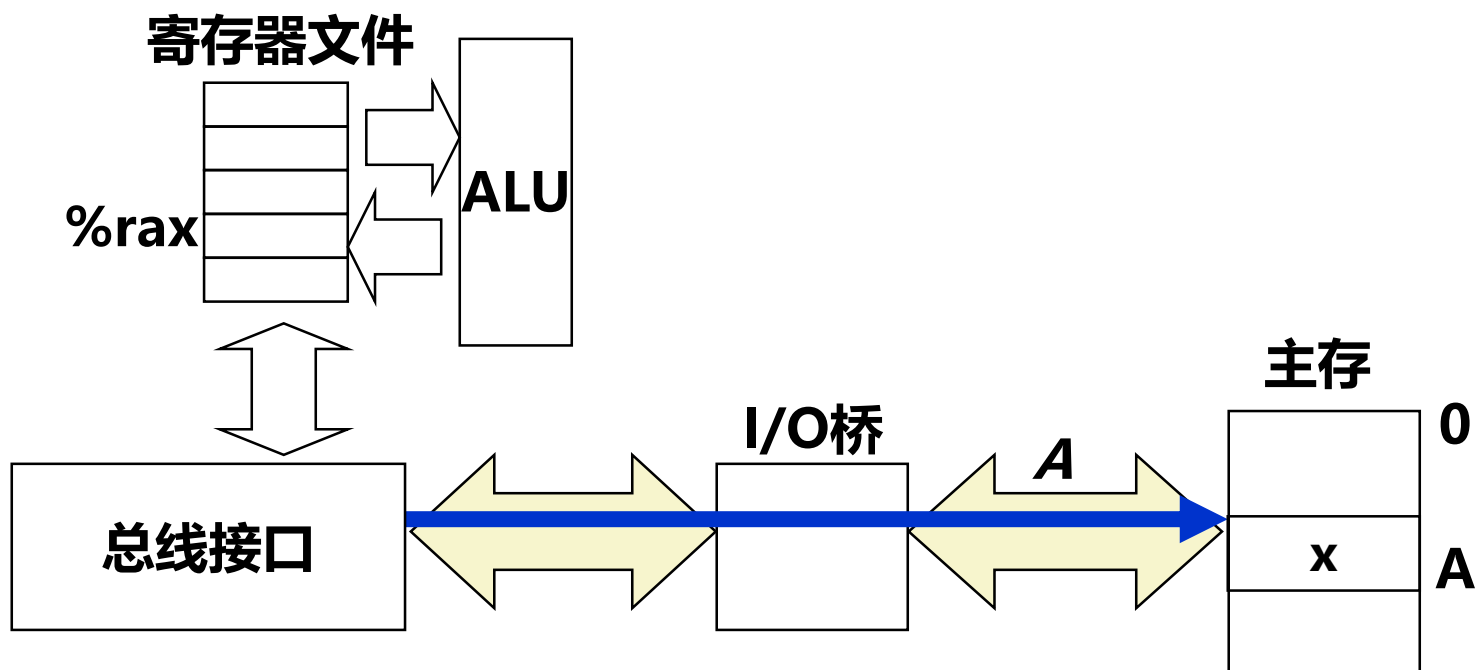
CPU 芯片



存储器读事务(1)

加载操作: `movq A, %rax`

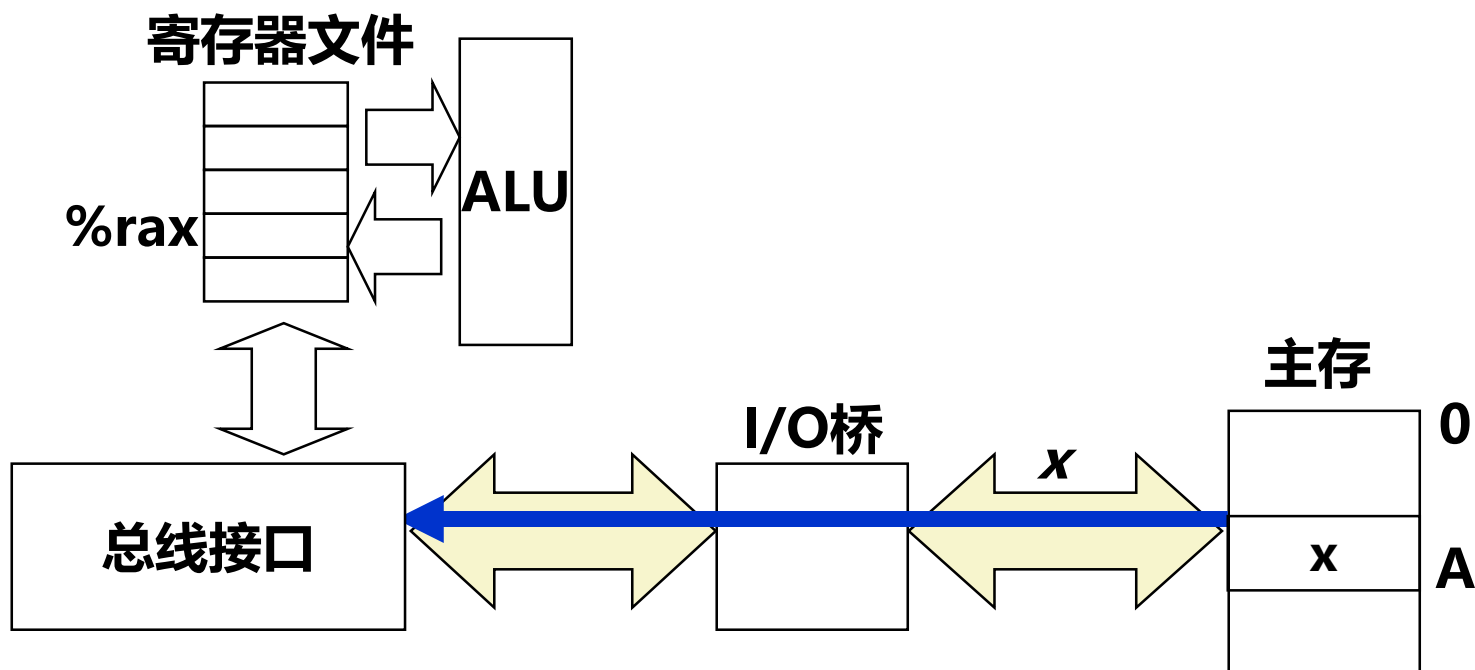
■ CPU将地址A放到总线上



存储器读事务(2)

加载操作: `movq A, %rax`

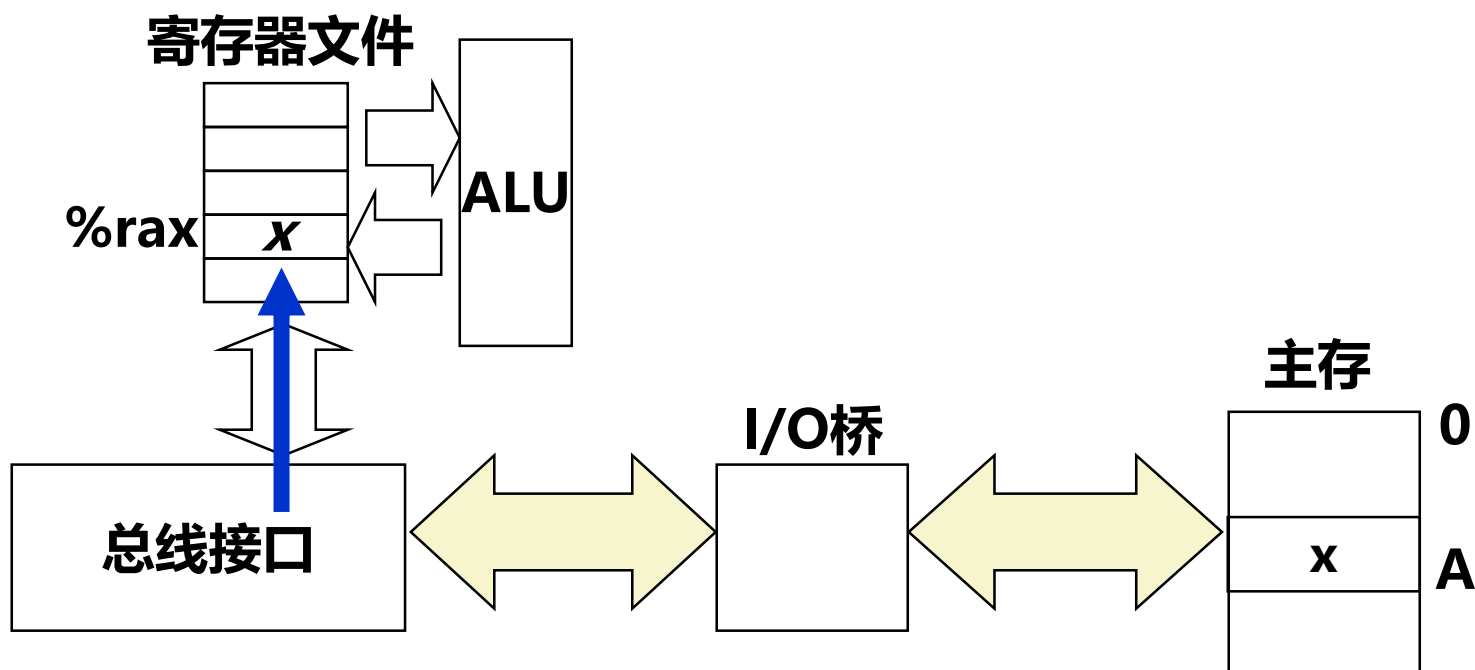
- 主存储器从总线上读地址A，取出字x，然后将x放到总线上



存储器读事务(3)

加载操作: `movq A, %rax`

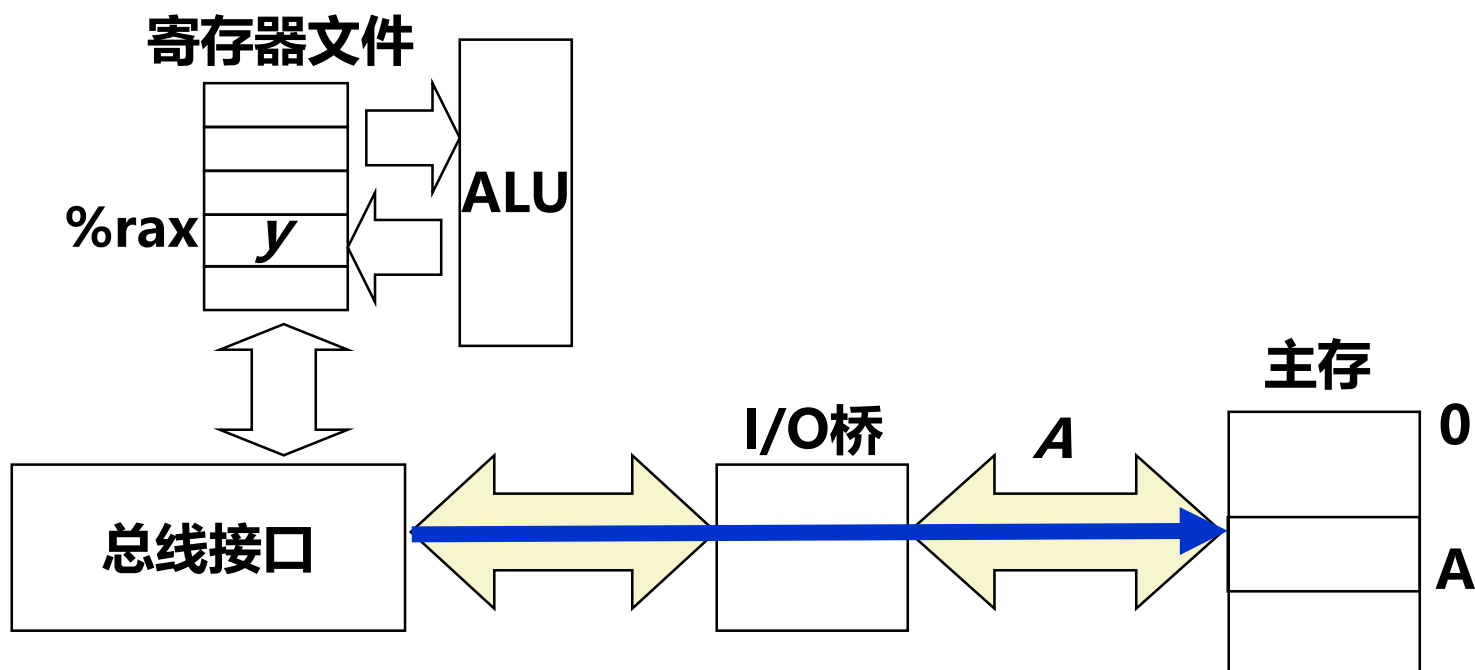
- CPU 从总线上读入字 x , 并将其放入寄存器 `%rax`



存储器写事务(1)

- CPU 将地址A放到总线上，主存储器读地址A并等待数据到来

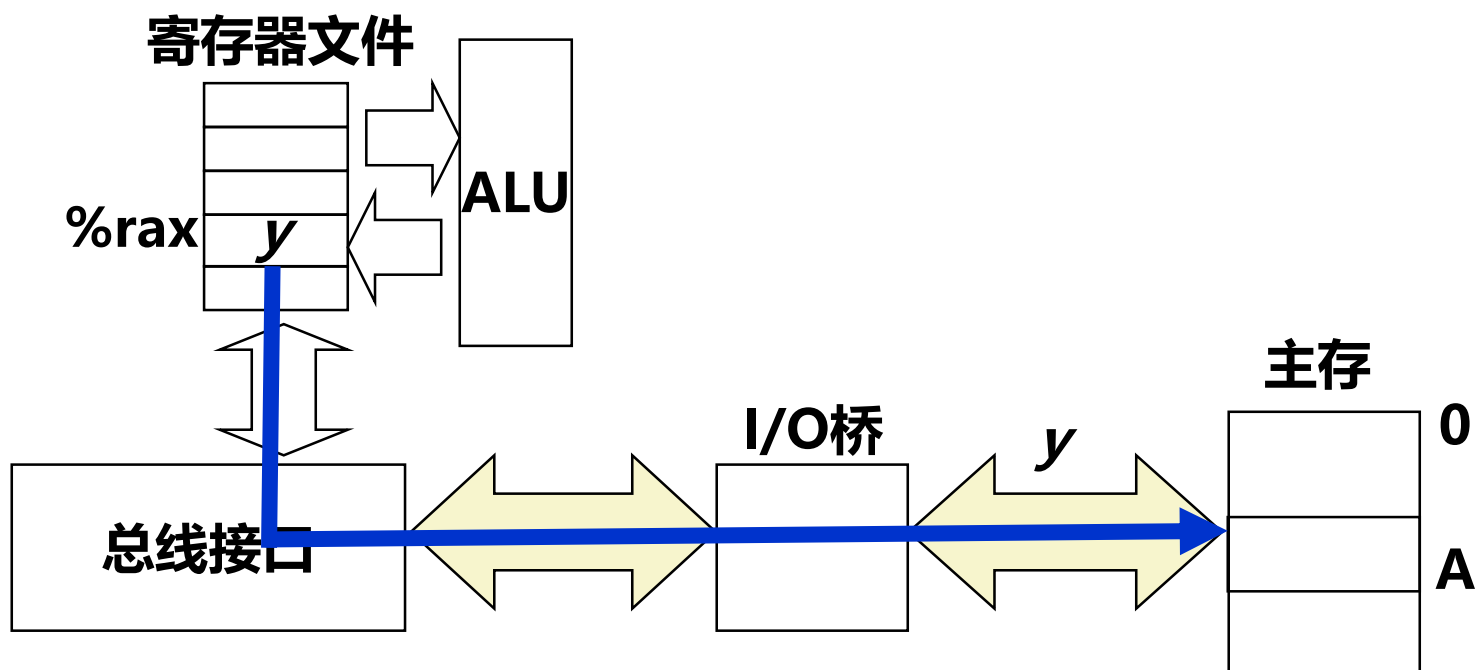
存储操作: `movq %rax, A`



存储器写事务(2)

■ CPU 将字y放到总线上

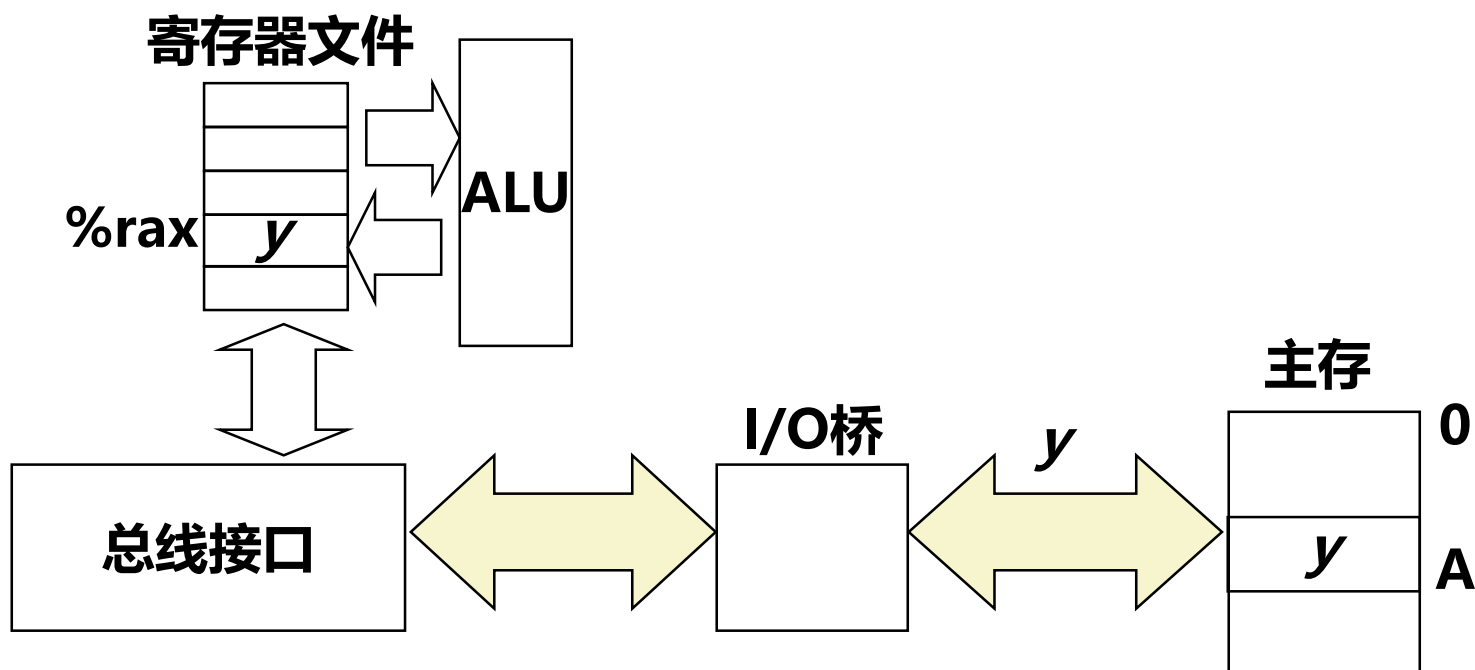
存储操作: `movq %rax, A`



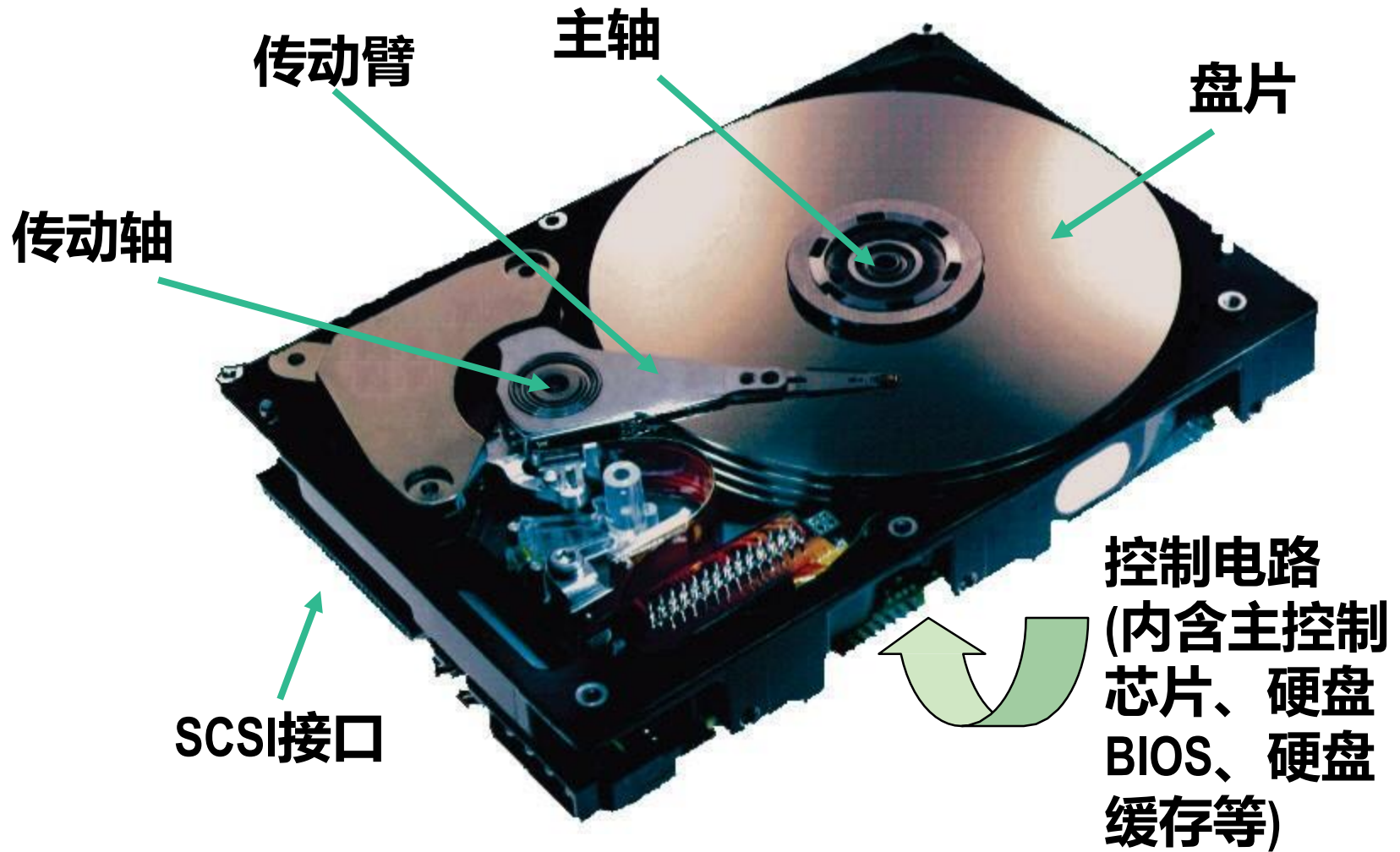
存储器写事务(3)

■ CPU 将字y放到总线上

存储操作: `movq %rax, A`

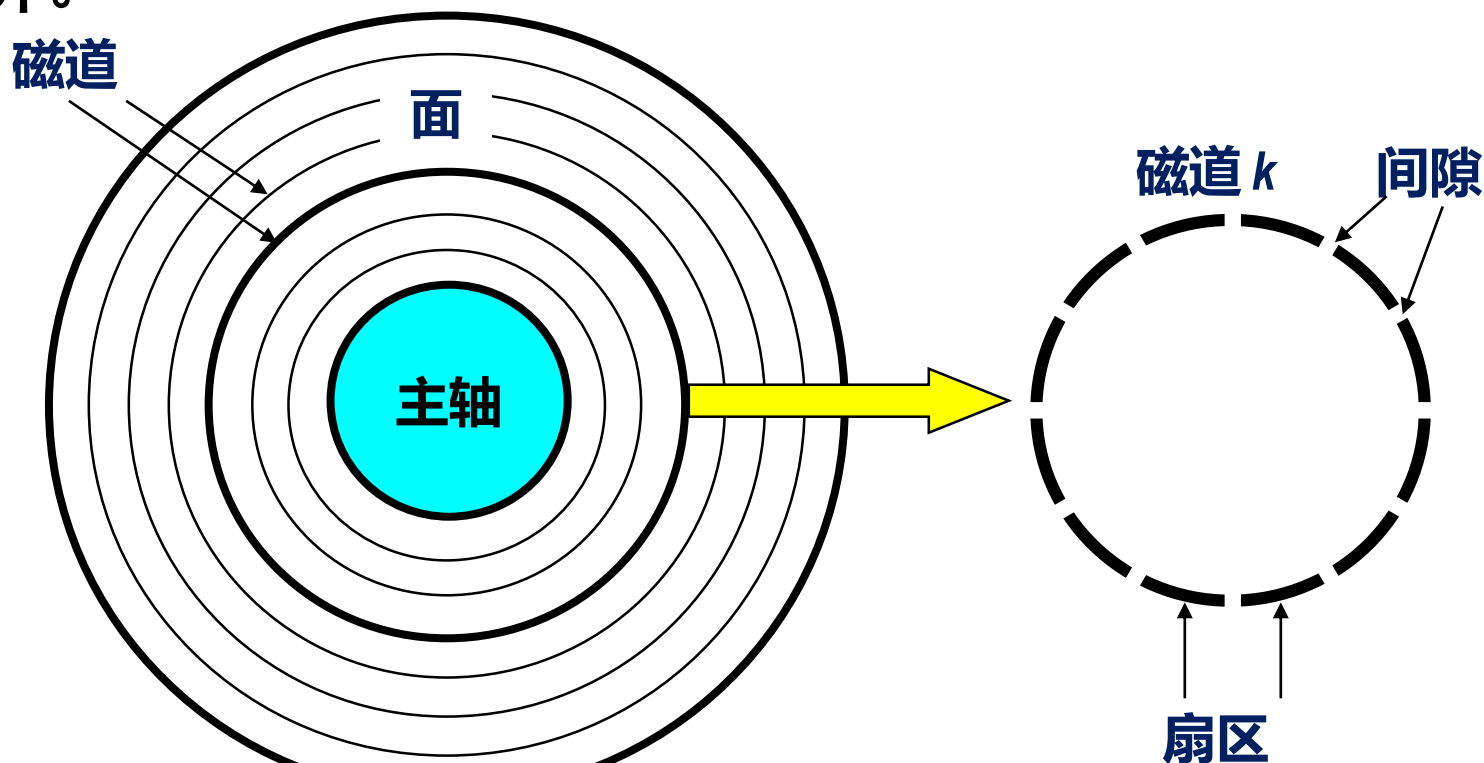


磁盘内部结构



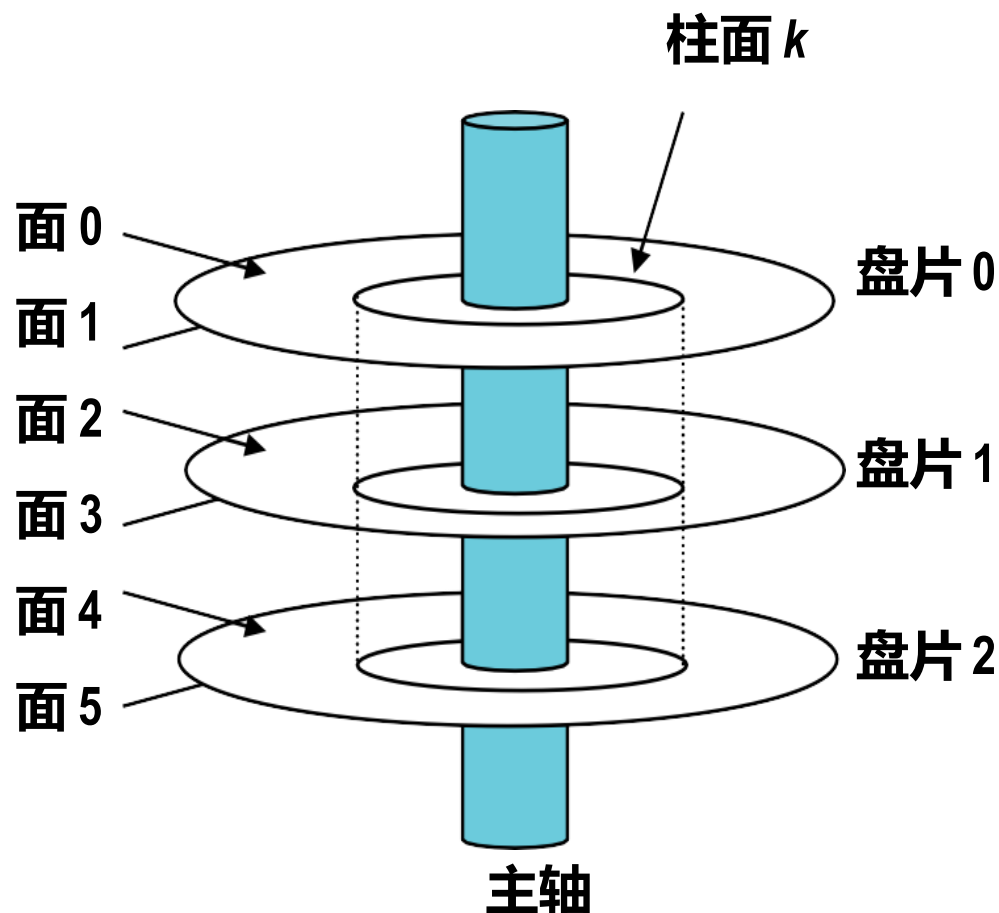
磁盘结构

- 磁盘由盘片(platter)构成，每个盘片包含两面(surface)。
- 每面由一组称为磁道(track)的同心圆组成。
- 每个磁道划分为一组扇区(sector)，扇区之间由间隙(gap)隔开。



磁盘结构(多个盘片)

- 同一半径上的所有磁道组成一个柱面。

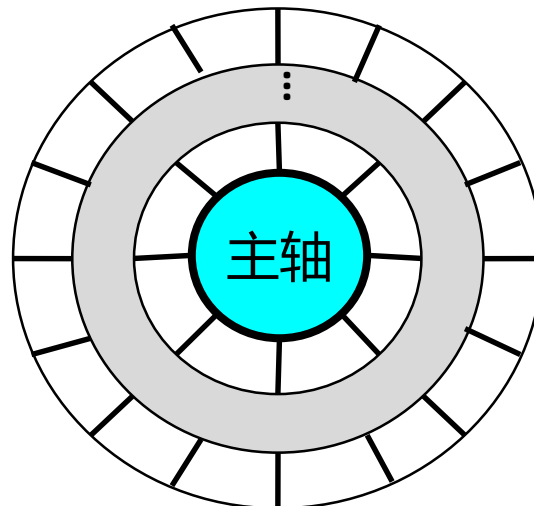


磁盘容量

- **容量(Capacity):** 磁盘上可以存储的最大位(bits)数。
 - 制造商以千兆字节(GB)为单位来表达磁盘容量
 - $1 \text{ GB} = 10^9$ 字节。
- **磁盘容量由以下技术因素决定:**
 - **记录密度(Recording density)** (位/英寸): 磁道一英寸的段中可放入的位数。
 - **磁道密度(Track density)** (道/英寸): 从盘片中心出发半径上一英寸的段内可以有的磁道数。
 - **面密度(Areal density)** (位/平方英寸): 记录密度与磁道密度的乘积。

分区记录

- 现代磁盘将所有磁道划分为若干分组(**recording zone**)，组内各磁道相邻
 - 区域内各磁道的扇区数目相同，扇区数取决于区域内最内侧磁道的圆周长
 - 各区域的每磁道扇区数都不同，外圈区域的每磁道扇区数比内圈区域多
 - 所以我们使用每磁道平均扇区数来计算磁盘容量。

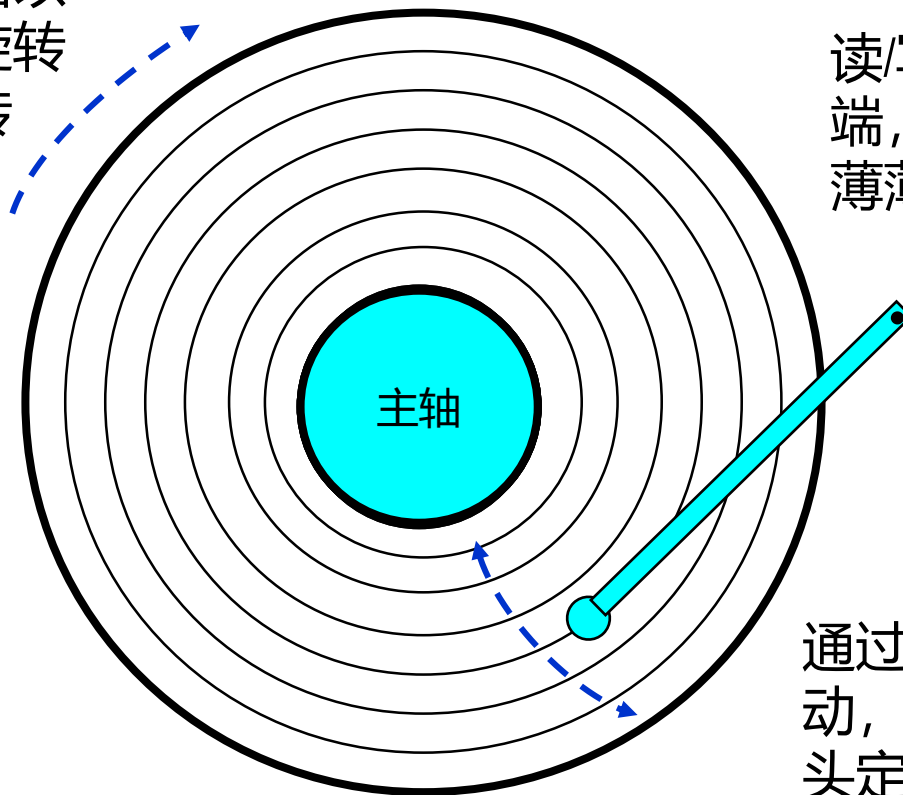


计算磁盘容量

- 容量 = (字节数/扇区) × (平均扇区数/磁道) × (磁道数/面) × (面数/盘片) × (盘片数/磁盘)
- 例:
 - 512 字节/扇区
 - 300 扇区/磁道 (平均值)
 - 20,000 磁道/面
 - 2 面/盘片
 - 5 盘片/磁道
- 容量 = $512 \times 300 \times 20,000 \times 2 \times 5$
= 30,720,000,000
= 30.72 GB

磁盘操作 (单盘片视图)

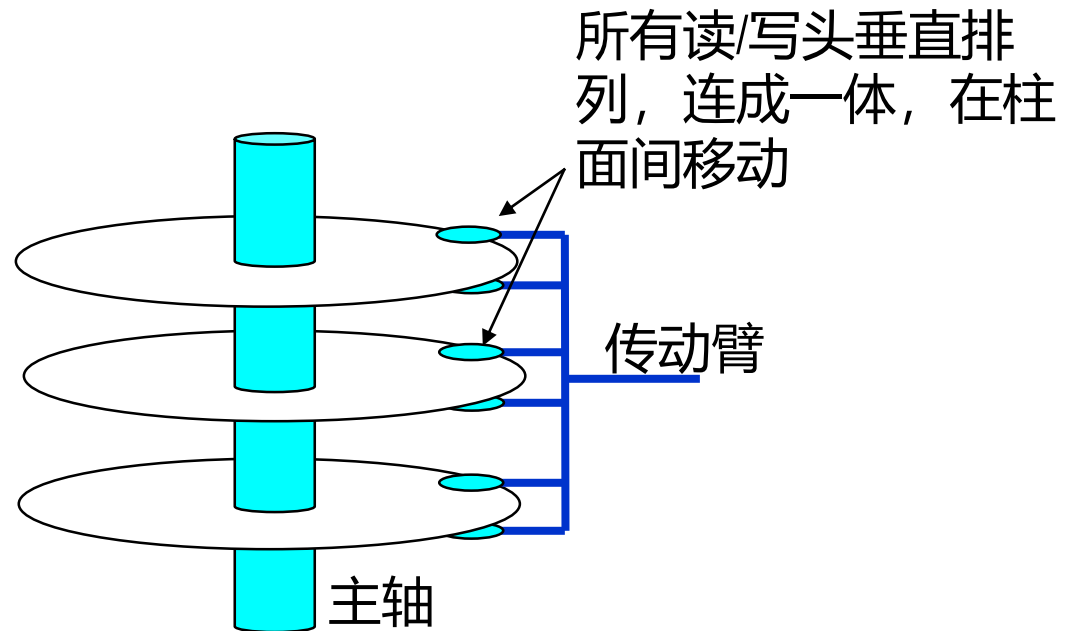
磁盘表面以
固定的旋转
速率旋转



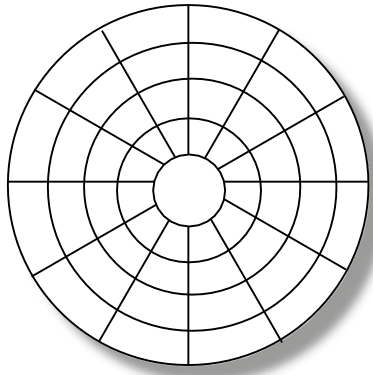
读/写头连到传动臂的末端，在磁盘表面上一层薄薄的气垫上飞翔

通过在半径方向上移动，传动臂可以将读/写头定位在任何磁道上

磁盘操作(多盘片)



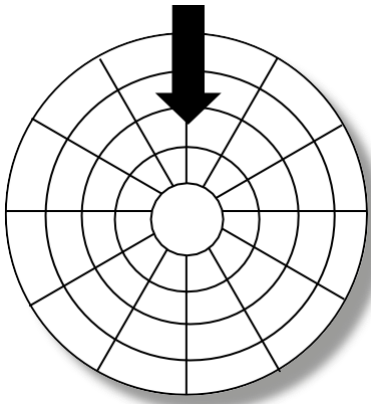
磁盘结构 – 单盘片俯视图



面由若干磁道构成

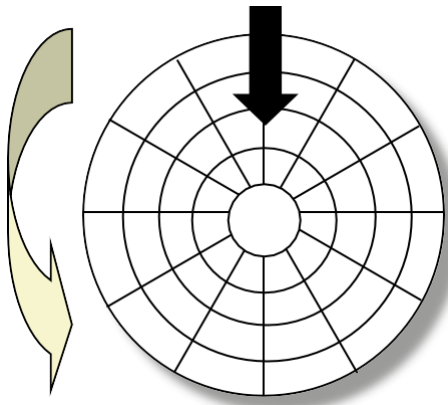
磁道被划分为若干扇区

磁盘访问



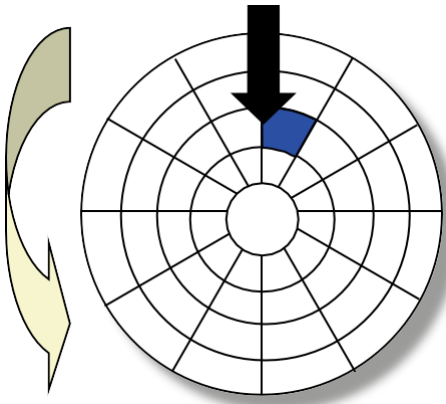
读/写头在磁道上方

磁盘访问



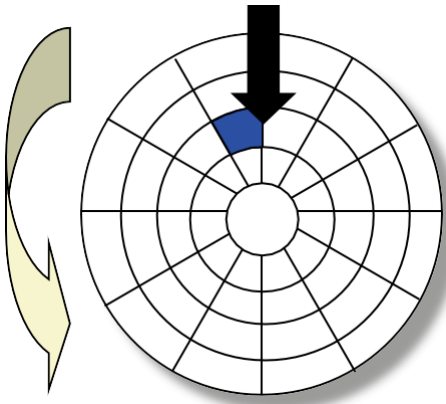
盘面逆时针旋转

磁盘访问 - 读



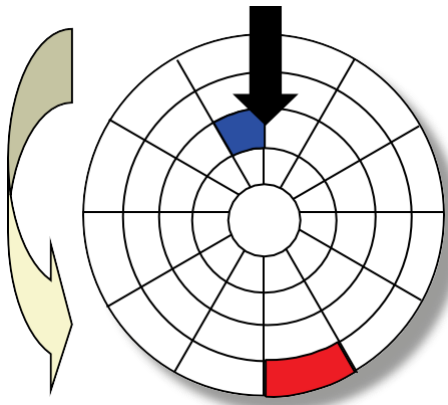
读取蓝色扇区

磁盘访问 - 读



读完蓝色扇区

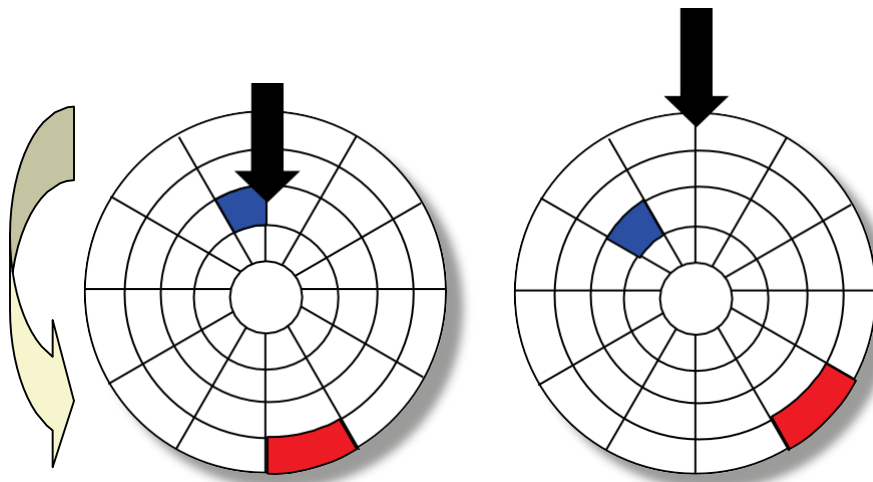
磁盘访问 - 读



读完蓝色扇区

请求读取红色扇区

磁盘访问 - 寻道

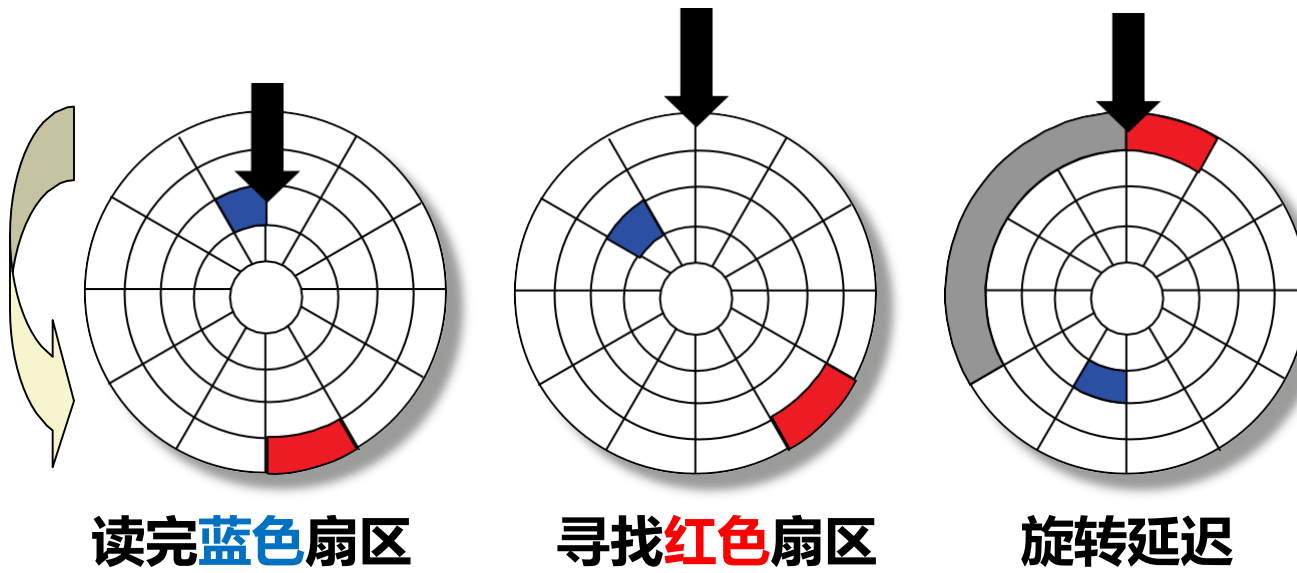


读完蓝色扇区

寻找红色扇区

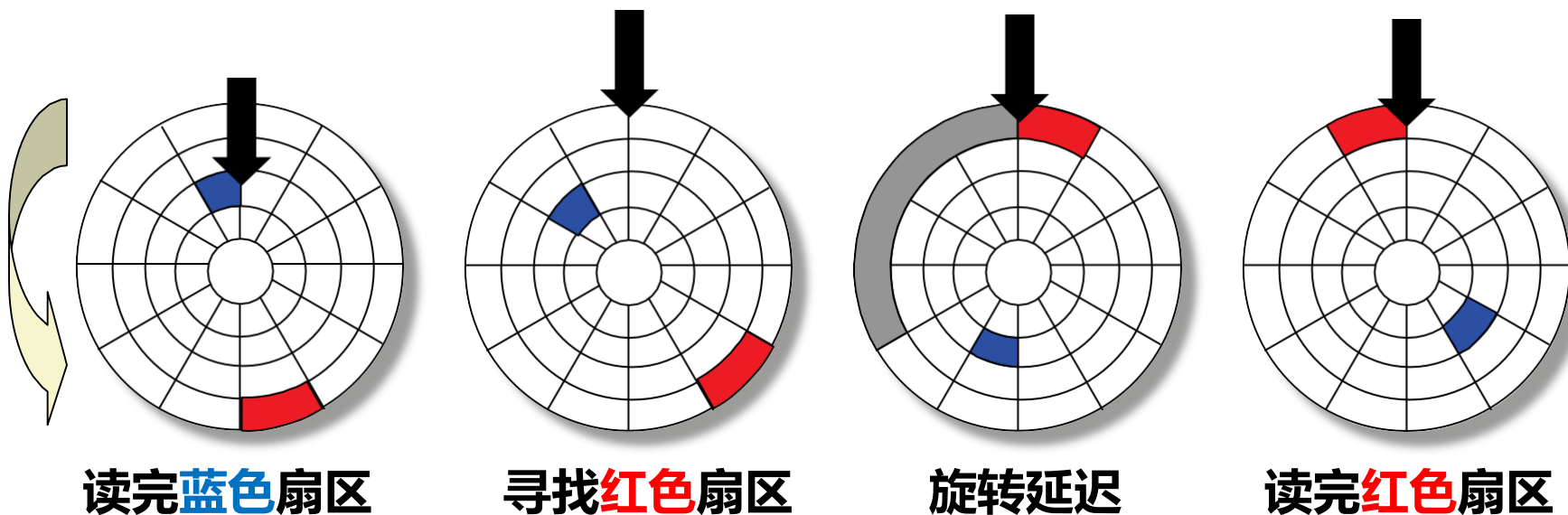
寻找红色扇区所在磁道

磁盘访问- 旋转延迟



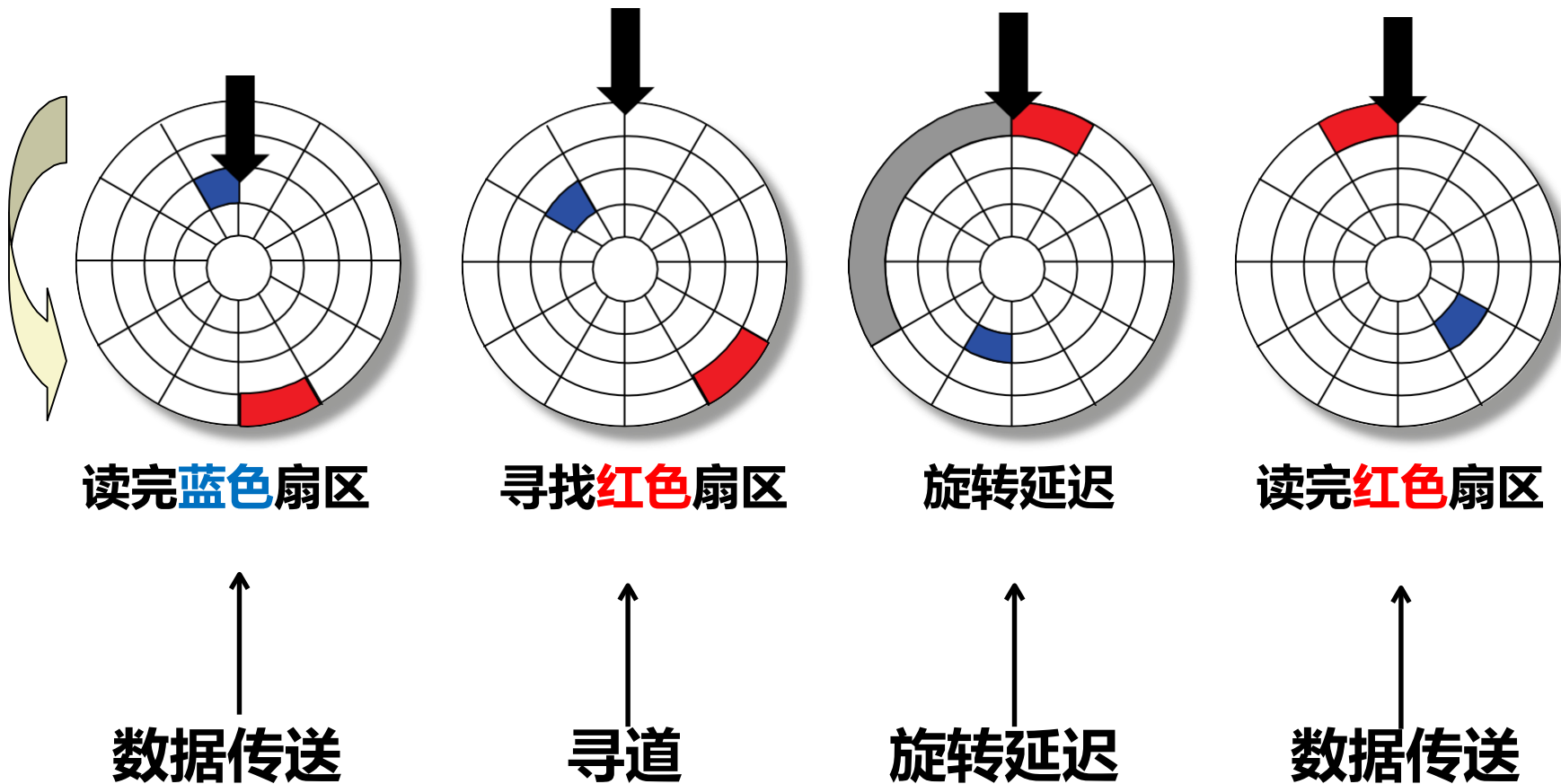
旋转盘面，使读/写头在红色扇区上方

磁盘访问- 读



读取红色扇区

磁盘访问- 访问时间构成



磁盘访问时间

■ 访问目标扇区的平均时间大致为

- 访问时间 = 寻道时间 + 平均旋转延迟 + 数据传输时间

■ 寻道时间(Seek time)

- 读/写头移动到目标柱面所用时间
- 通常寻道时间为：3—9 ms

■ 旋转延迟(Rotational latency)

- 旋转盘面使读/写头到达目标扇区上方所用时间
- 平均旋转延迟 = $1/2 \times 1/\text{RPMs} \times 60 \text{ sec}/1 \text{ min}$ (RPM: 转/分钟)
- 通常 RPMs = 7,200

■ 数据传输时间(Transfer time)

- 读目标扇区所用时间
- 数据传输时间 = $1/\text{RPM} \times 1/(\text{平均扇区数/磁道}) \times 60 \text{ secs}/1 \text{ min}$

磁盘访问时间 - 举例

■ 给定条件:

- 旋转频率 = 7,200 转/分钟
- 平均寻道时间 = 9 ms
- 平均扇区数/磁道 = 400

■ 计算:

- 平均旋转延迟 = ?
- 数据传输时间 = ?
- 访问时间 = ?

■ 访问目标扇区的平均时间大致为

- 访问时间 = 寻道时间 + 平均旋转延迟 + 数据传输时间

■ 寻道时间(Seek time)

- 读/写头移动到目标柱面所用时间
- 通常寻道时间为: 3—9 ms

■ 旋转延迟(Rotational latency)

- 旋转盘面使读/写头到达目标扇区上方所用时间
- 平均旋转延迟 = $1/2 \times 1/\text{RPMs} \times 60 \text{ sec}/1 \text{ min}$ (RPM: 转/分钟)
- 通常 RPMs = 7,200 RPMs

■ 数据传输时间(Transfer time)

- 读目标扇区所用时间
- 数据传输时间 = $1/\text{RPM} \times 1/(\text{平均扇区数/磁道}) \times 60 \text{ secs}/1 \text{ min}$

磁盘访问时间 - 举例

■ 给定条件:

- 旋转频率 = 7,200 转/分钟
- 平均寻道时间 = 9 ms
- 平均扇区数/磁道 = 400.

■ 计算结果:

- 平均旋转延迟 = $1/2 \times (60 \text{ secs}/7200 \text{ RPM}) \times 1000 \text{ ms/sec} = 4 \text{ ms}$
- 数据传输时间 = $60/7200 \text{ RPM} \times 1/400 \text{ 扇区数/磁道} \times 1000 \text{ ms/sec} = 0.02 \text{ ms}$
- 访问时间 = $9 \text{ ms} + 4 \text{ ms} + 0.02 \text{ ms}$

■ 重 点:

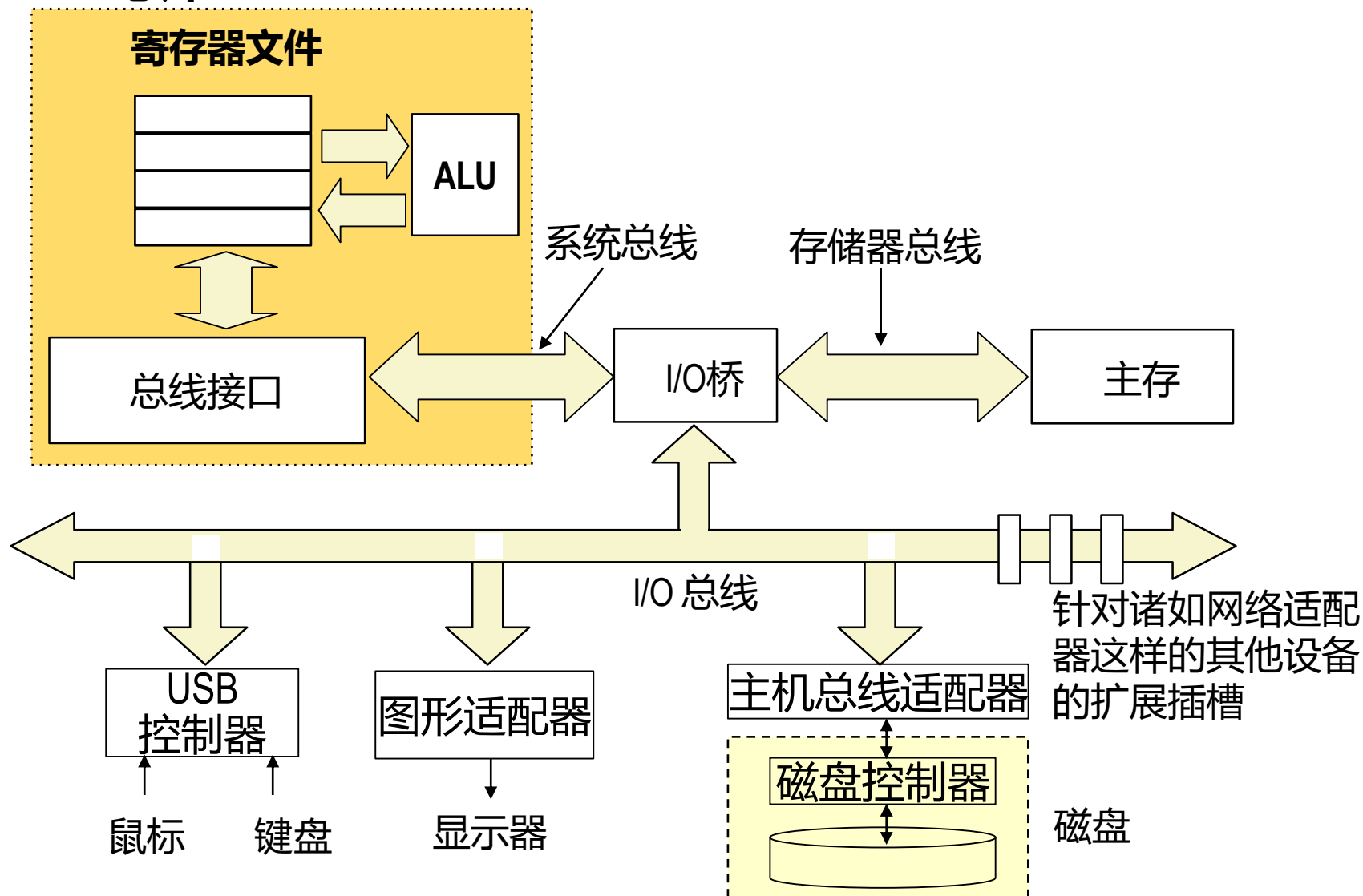
- 访问时间主要由寻道时间和旋转延迟时间组成。
- 访问扇区首位花费时间较长, 其他位较快。
- *SRAM 访问时间大约为 4 ns/双字, DRAM 大约为 60 ns/双字。*
 - *磁盘比SRAM慢大约 40,000 倍, 比DRAM慢大约 2,500 倍。*

逻辑磁盘块

- 现代磁盘以简单的抽象视图来表示复杂的磁盘构造:
 - 磁盘被抽象成 b 个扇区大小的逻辑块(logical block)序列 (编号为0, 1, 2, ...)
- 逻辑块与物理扇区之间的映射关系
 - 由磁盘控制器维护
 - 磁盘控制器将逻辑块号转换为一个三元组(盘面,磁道,扇区)
- 允许磁盘控制器为每个分区预留一组柱面作为备份
 - 区分“格式化容量”与“最大容量”

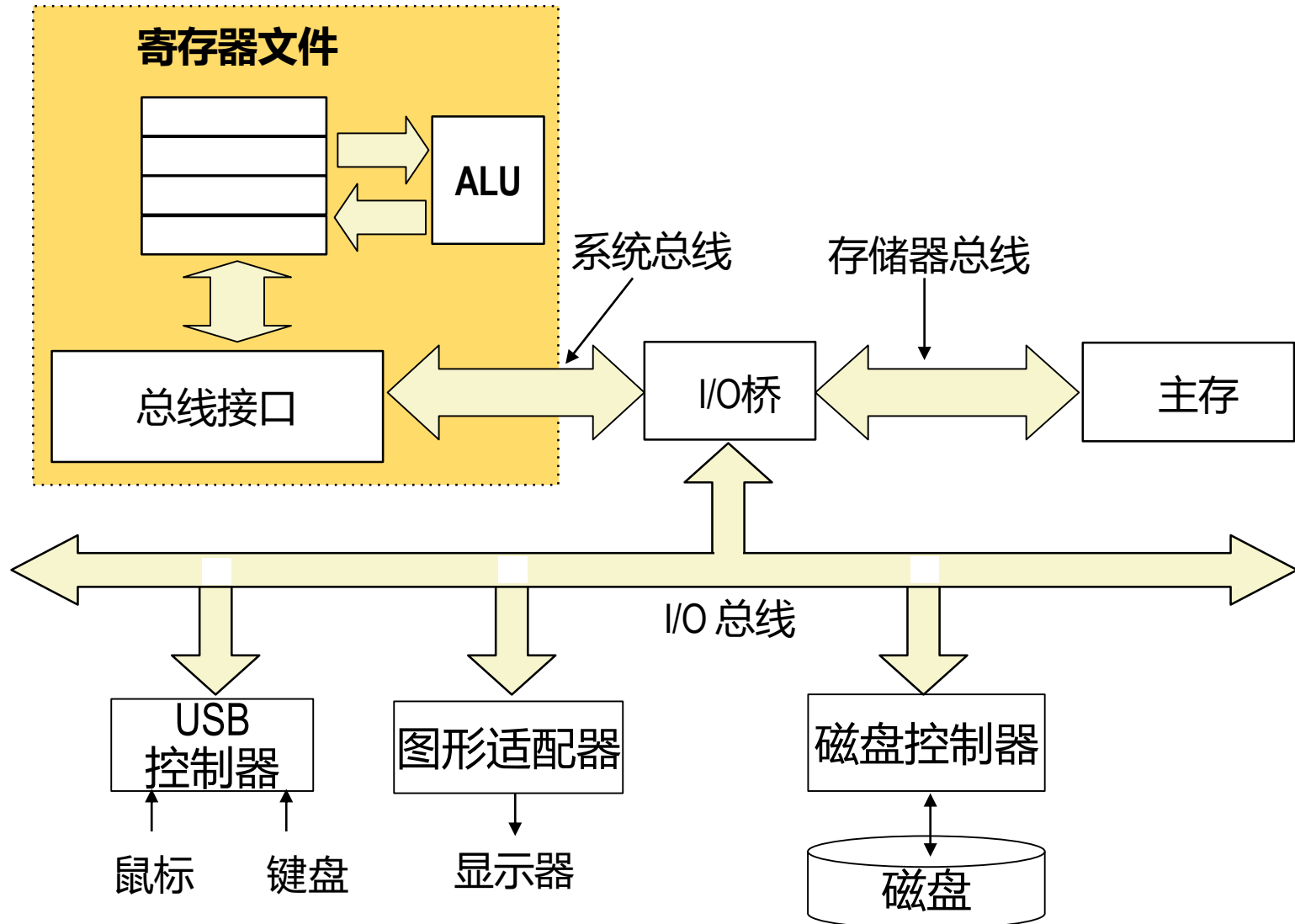
I/O 总线

CPU 芯片



I/O 总线

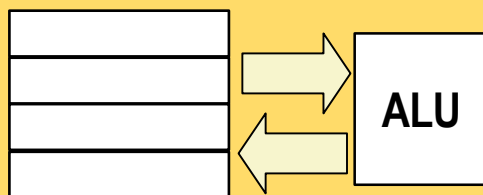
CPU 芯片



读磁盘扇区 (1)

CPU 芯片

寄存器文件



总线接口

CPU 通过将命令、逻辑块号和目的内存地址写到与磁盘相关联的内存映射地址(port), 发起一个磁盘读操作。

I/O 桥

主存

I/O 总线

USB
控制器

图形适配器

磁盘控制器

鼠标

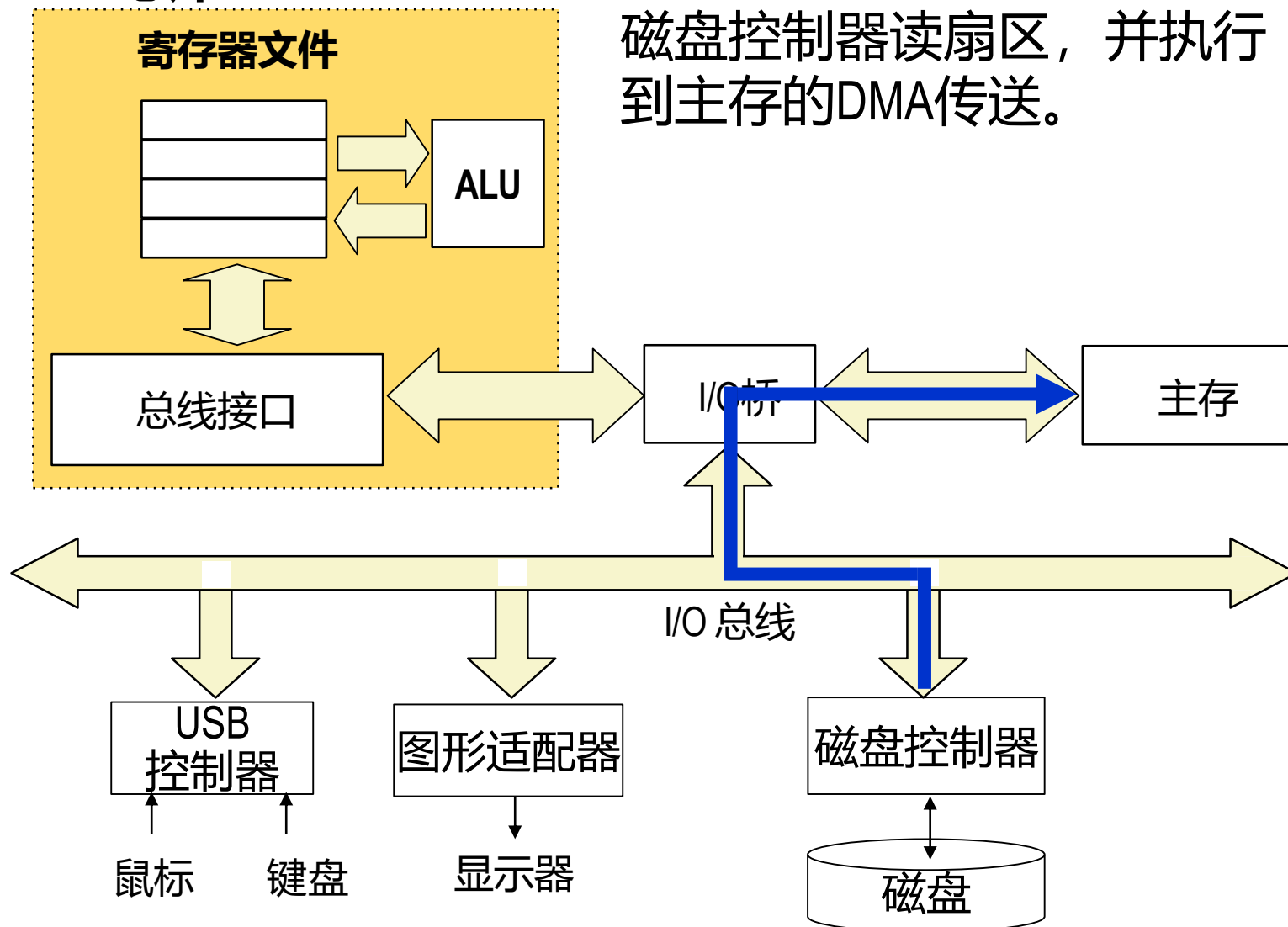
键盘

显示器

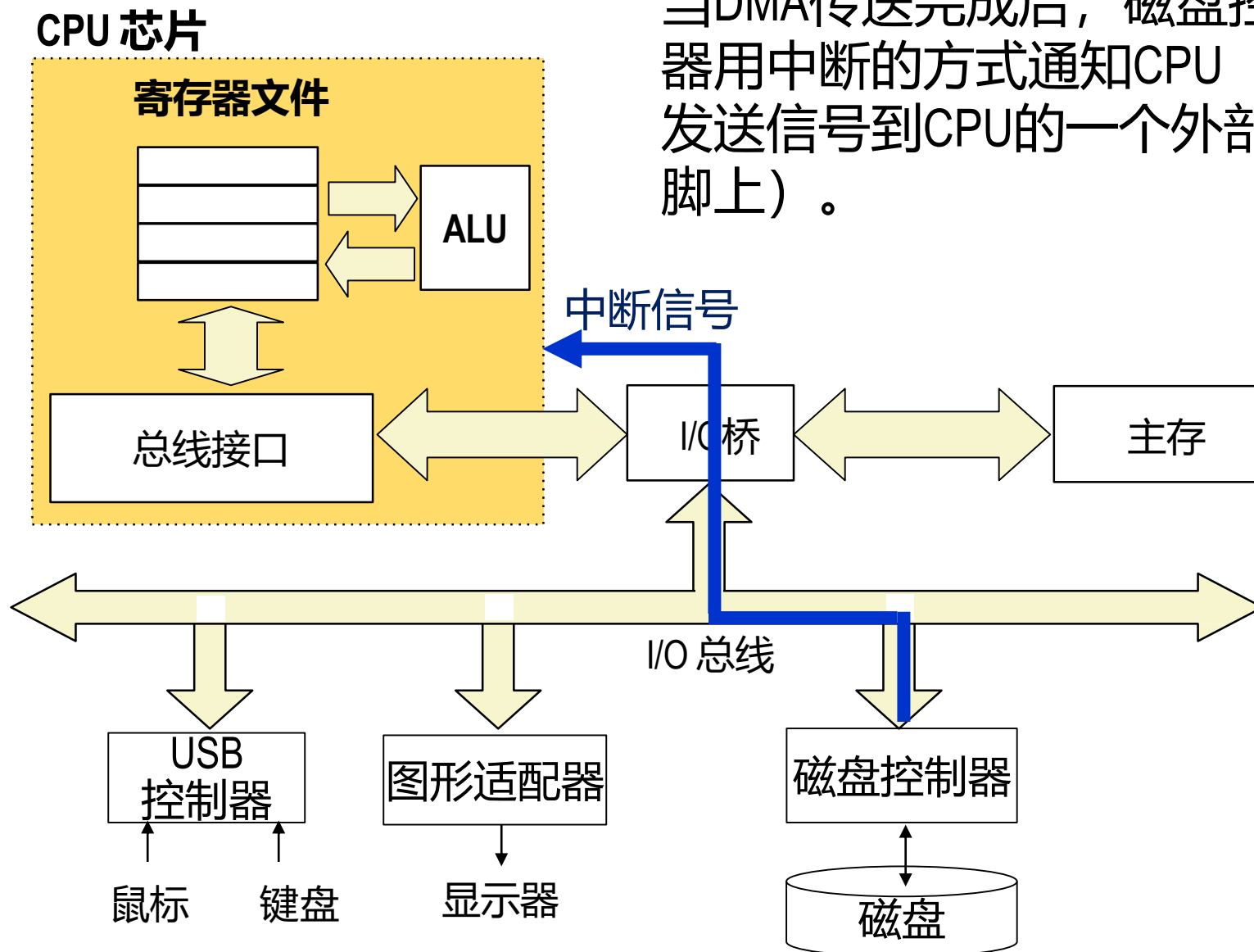
磁盘

读磁盘扇区 (1)

CPU 芯片

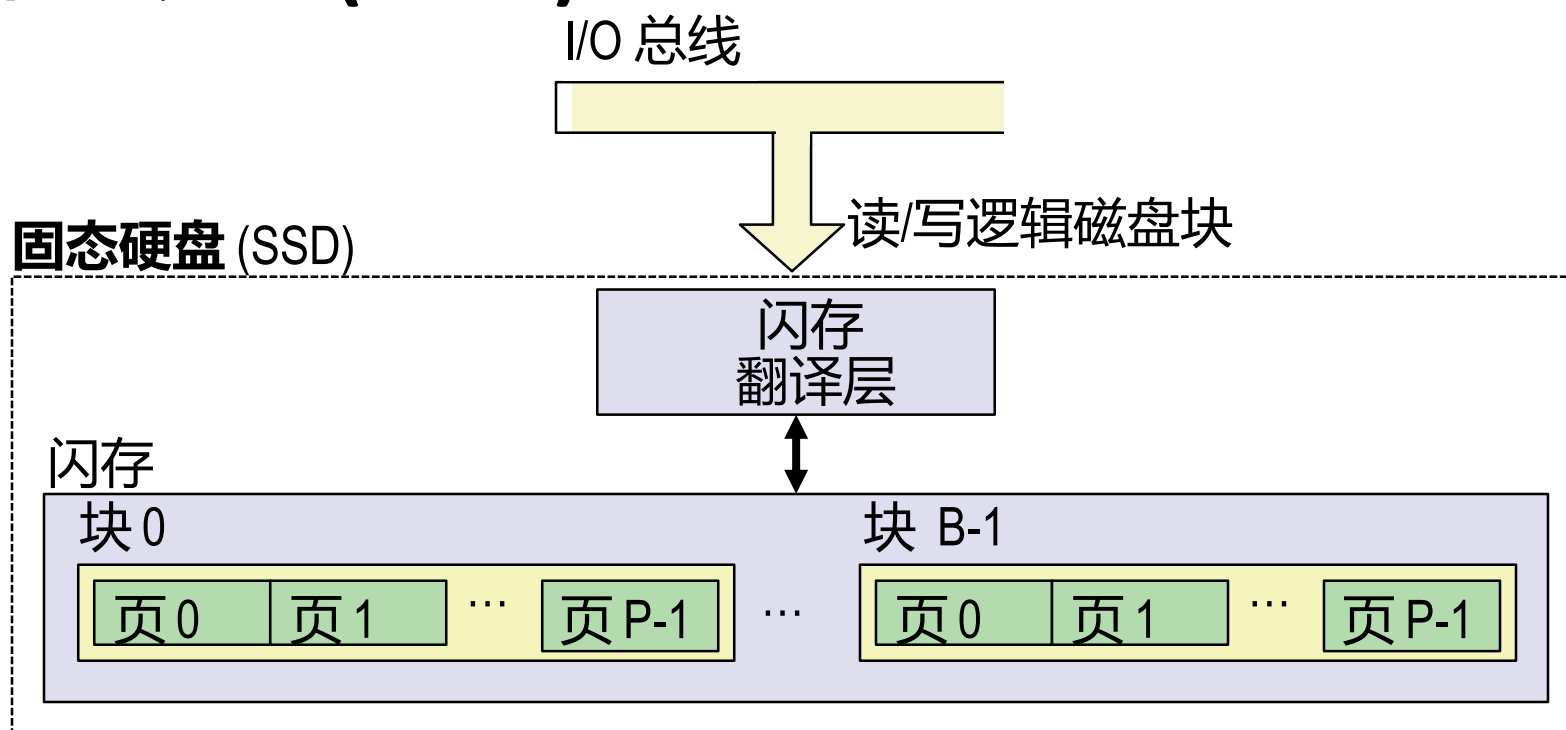


读磁盘扇区 (3)



当DMA传送完成后，磁盘控制器用中断的方式通知CPU（即发送信号到CPU的一个外部引脚上）。

固态硬盘 (SSDs)



- 页大小：512B ~ 4KB, 块大小：32 ~ 128页
- 数据以页为单位进行读写
- 只有某页所属块整个被擦除后，才能写该页
- 大约 100,000 次重复写之后，块就会磨损坏

SSD 性能特性

顺序读吞吐量	550 MB/s	顺序写吞吐量	470 MB/s
随机读吞吐量	365 MB/s	随机写吞吐量	303 MB/s
平均顺序读访问时间	50 μ s	平均顺序写访问时间	60 μ s

- **顺序访问比随机访问快**
 - 典型存储器层次结构问题
- **随机写较慢**
 - 擦除块需要较长的时间(~ 1 ms)
 - 修改一页需要将块中所有页复制到新的块中
 - 早期SSD 读/写速度之间的差距更大

资料来源: Intel SSD 730 产品详细说明书

SSD vs 机械磁盘

■ 优点

- 没有移动部件 → 更快、能耗更低、更结实（抗震）

■ 缺点

- 会磨损

通过flash翻译层中的“磨损均衡逻辑”缓解：

将擦除平均分布在所有块上来最大化每个块的寿命

Intel SSD 730 保证能经得起 128 PB (128×10^{15} 字节) 的写

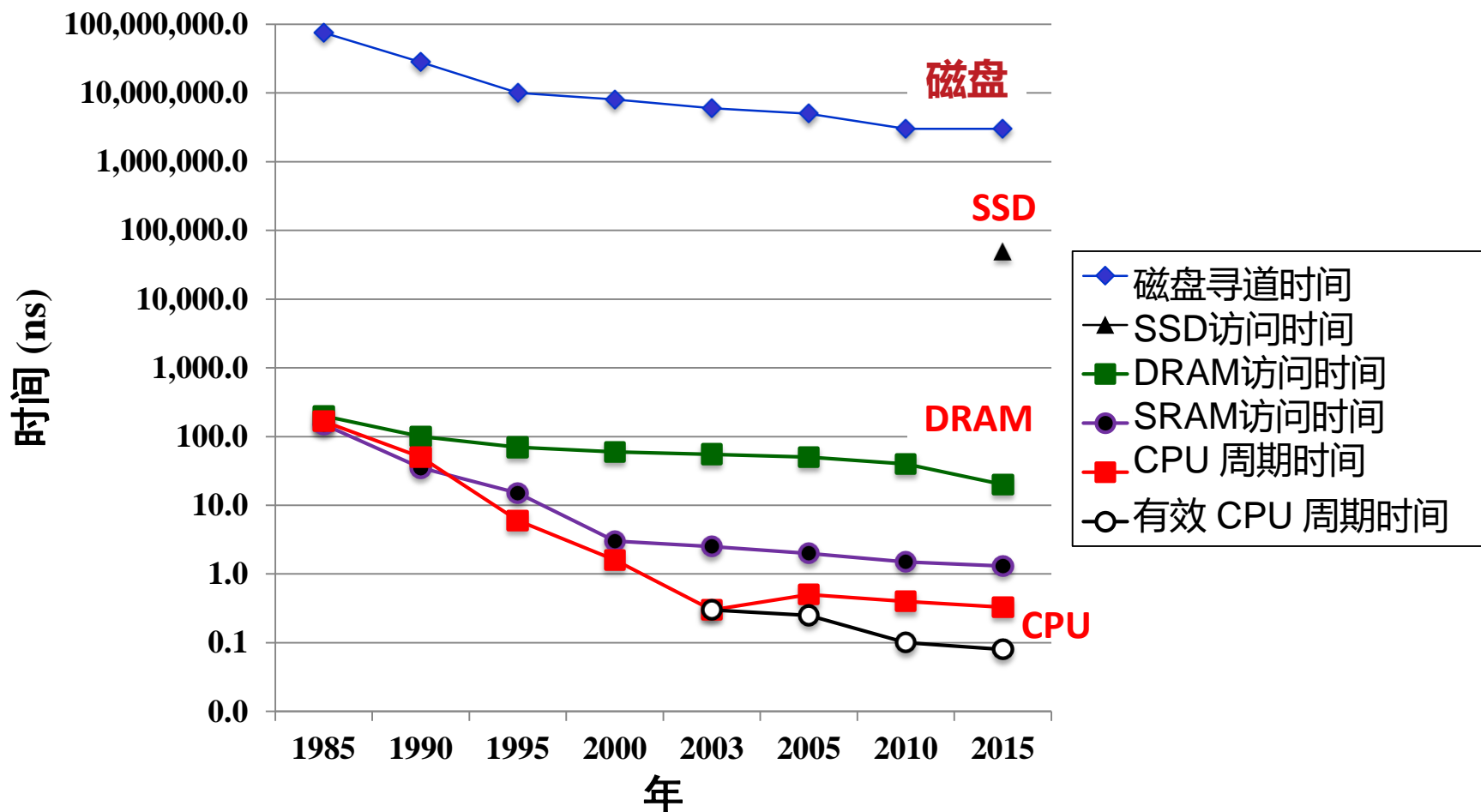
- 2015年，SSD每字节比机械磁盘贵大约30倍,目前约10倍

■ 应用

- MP3播放器、智能手机、笔记本电脑
- 开始在台式机和服务器中应用

CPU-储存器的速度差距

DRAM、磁盘和CPU之间的速度差距在变大.....



主要内容

- 存储技术与趋势
- 局部性
- 存储器层次结构中的高速缓存

用局部性原理(locality)来解决

- 解决CPU-存储器之间速度差距的关键是程序中特有的**局部性**特点。

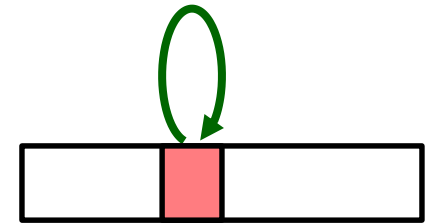
局部性

■ 局部性原理(Principle of Locality)

程序倾向于使用距离最近用过的指令/数据地址相近或相等的指令/数据。

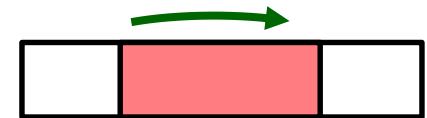
■ 时间局部性(Temporal locality)

- 最近访问过的信息，很可能在近期还会被再次访问



■ 空间局部性(Spatial locality)

地址接近的数据项，被使用的时间也倾向于接近



局部性举例

```
sum = 0;  
for (i = 0; i < n; i++)  
    sum += a[i];  
return sum;
```

■ 对数据的引用

- 顺序访问数组元素
(步长为1的引用模式)
- 变量`sum`在每次循环迭代中被引用一次

空间局部性

时间局部性

■ 对指令的引用

- 顺序读取指令
- 重复循环执行for循环体

空间局部性

时间局部性

对局部性的定性评价

- 通过查看程序代码，能对程序局部性有定性的认识
 - 专业程序员的一项关键技能。
- **问题：**针对数组 `a`，函数 `sum_array_rows` 具有良好的局部性吗？

```
int sum_array_rows(int a[M][N])
{
    int i, j, sum = 0;

    for (i = 0; i < M; i++)
        for (j = 0; j < N; j++)
            sum += a[i][j];
    return sum;
}
```

局部性举例

- **问题：**针对数组 `a`，函数 `sum_array_rows` 具有良好的局部性吗？

```
int sum_array_cols(int a[M][N])
{
    int i, j, sum = 0;

    for (j = 0; j < N; j++)
        for (i = 0; i < M; i++)
            sum += a[i][j];
    return sum;
}
```

局部性举例

- **问题:** 改变下面函数中循环的顺序，使它以步长为 1 的引用模式扫描三维数组 a，从而函数具有良好的局部性

```
int sum_array_3d(int a[M][N][N])
{
    int i, j, k, sum = 0;

    for (i = 0; i < N; i++)
        for (j = 0; j < N; j++)
            for (k = 0; k < M; k++)
                sum += a[k][i][j];

    return sum;
}
```

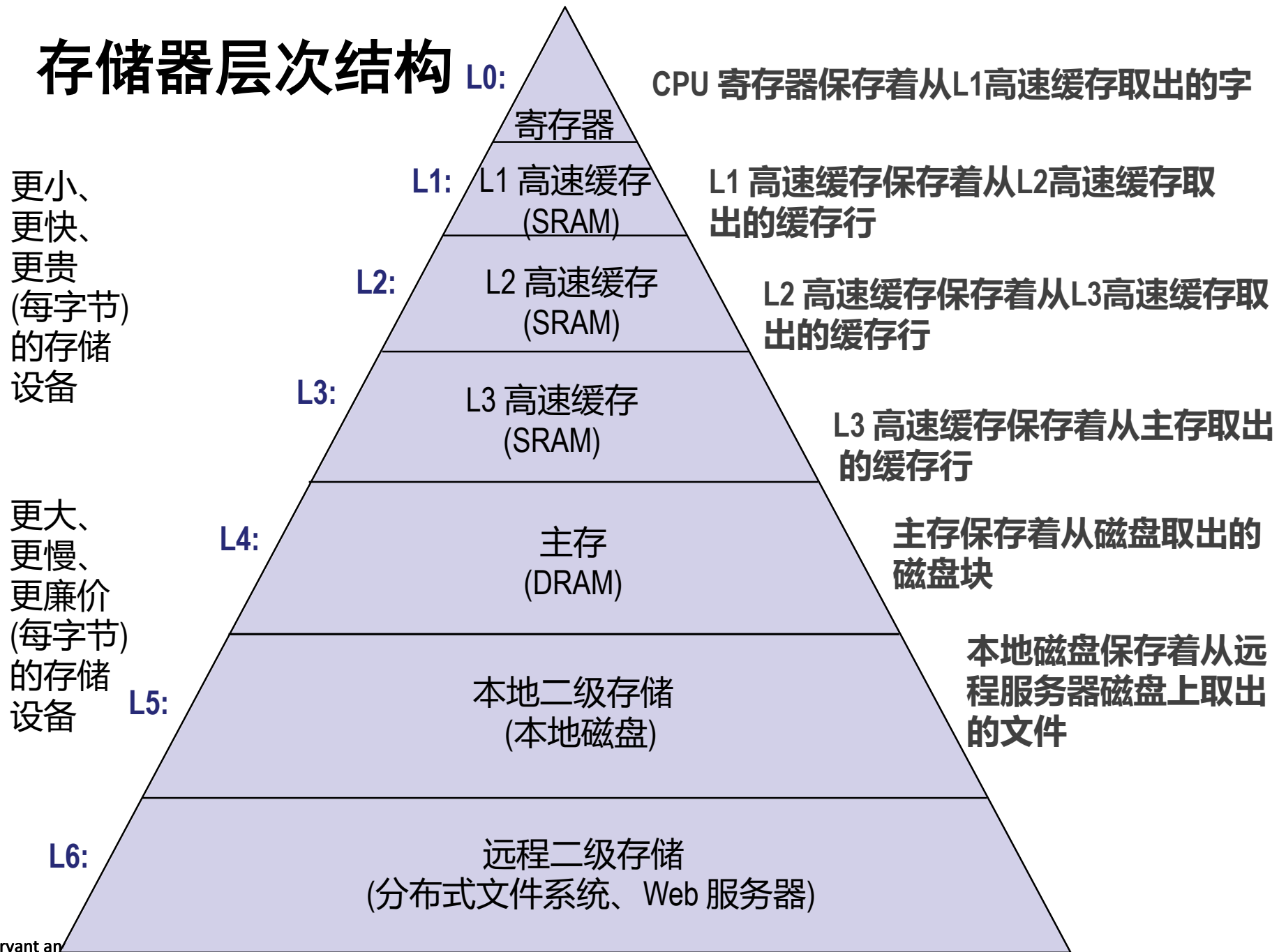
存储器层次结构

- 软硬件的基本特性：
 - **高速**存储器：成本高、容量小、耗电大易发热
 - **CPU**与存储器之间的速度差距越来越大
 - 程序编写的好，往往表现出良好的**局部性**
- 这些基本特性相互补充
- 基于以上特性，构造好的存储器系统——存储器层次结构(memory hierarchy).

主要内容

- 存储技术及其趋势
- 局部性
- 存储器层次结构、高速缓存

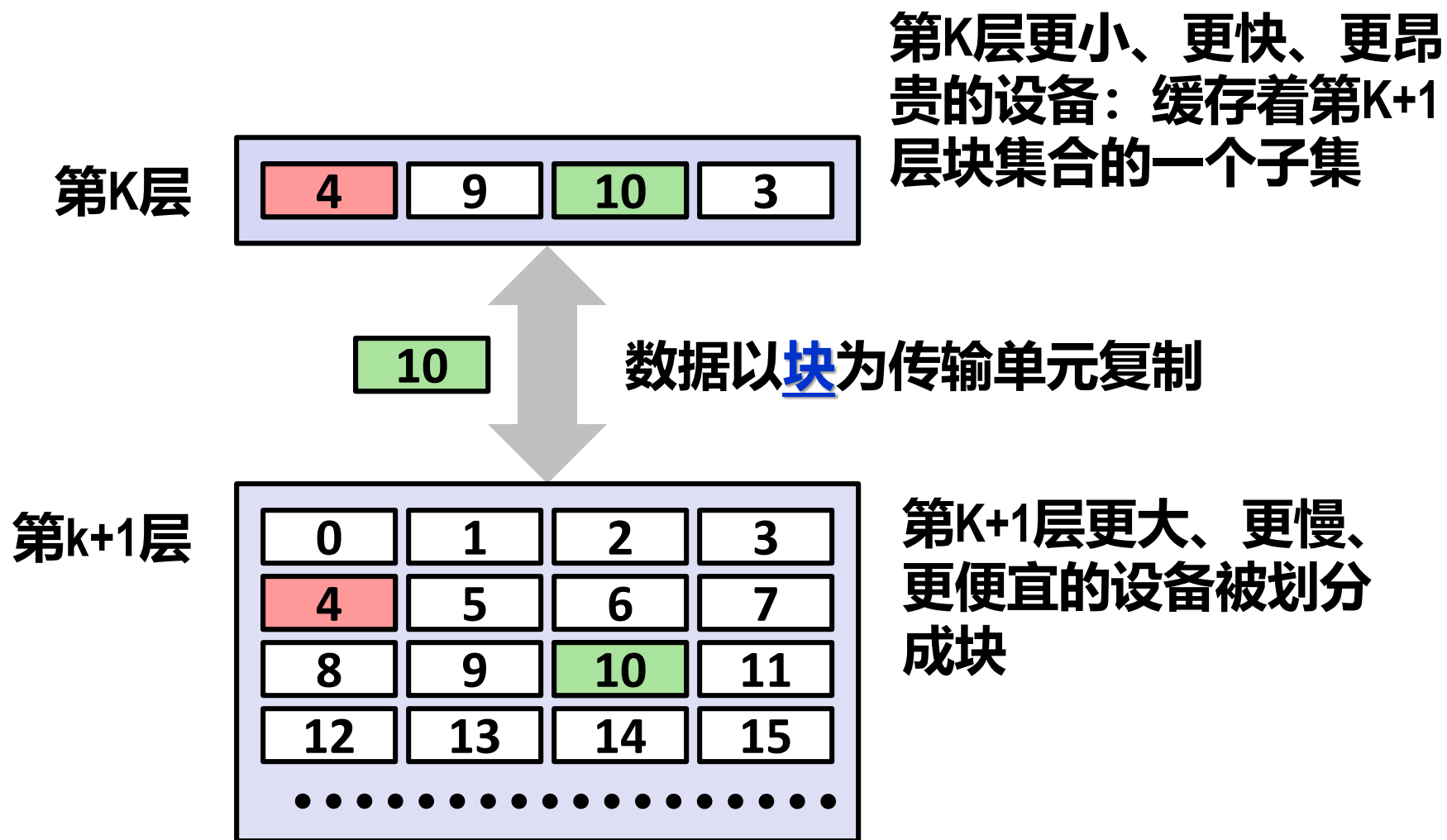
存储器层次结构



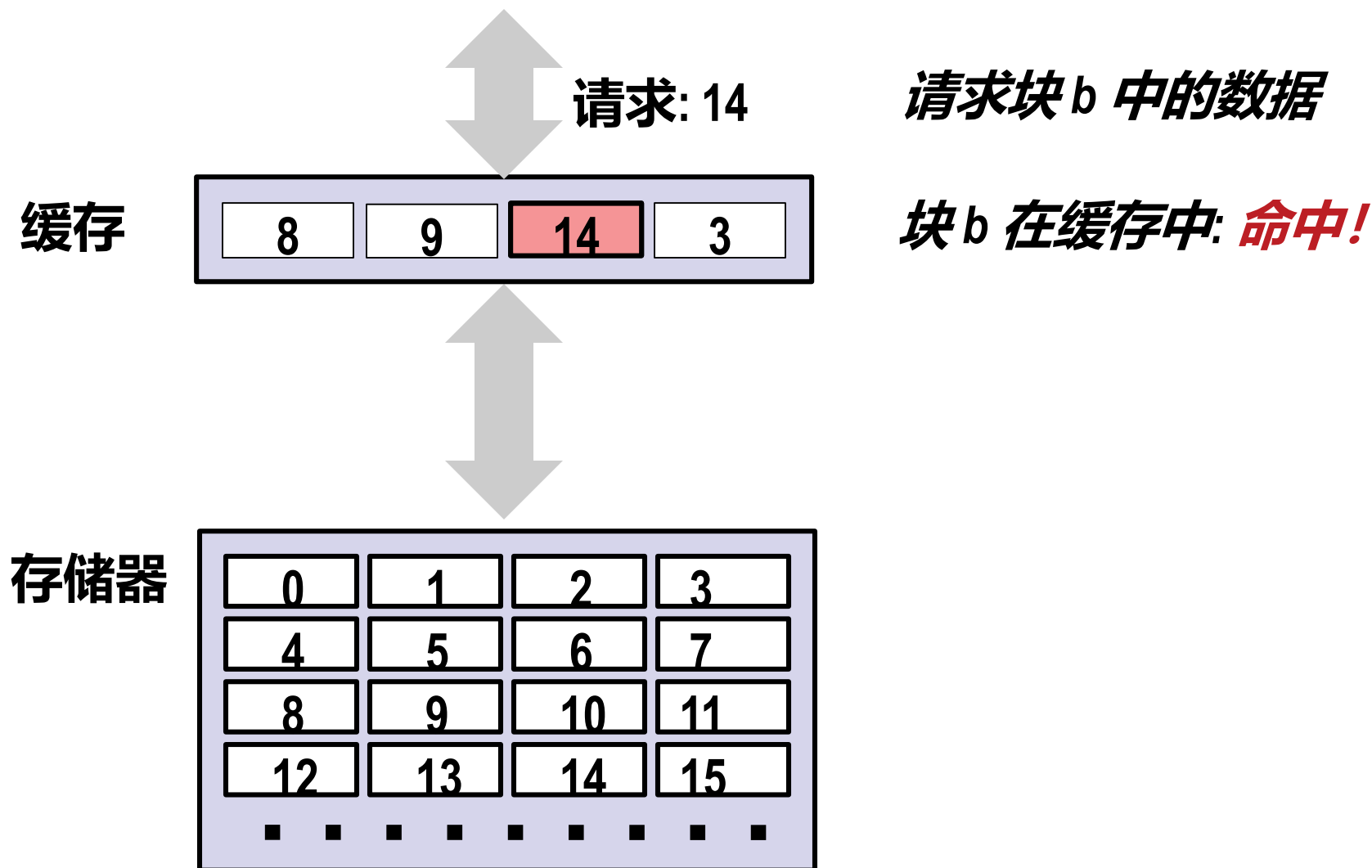
高速缓存

- **高速缓存(Cache):** 将一种更小、速度更快的存储设备，作为更大、更慢存储设备的缓存区。
- **存储器层次结构的基本思想**
 - 位于 k 层的更快更小存储设备作为位于 $k+1$ 层的更大更慢存储设备的缓存。
- **存储器层次结构为何行的通？**
 - 因为局部性原理，程序访问第 k 层的数据比访问第 $k+1$ 层的数据要频繁
 - 第 $k+1$ 层的存储设备更慢、容量更大、价格更便宜
- **妙策:** 存储器层次结构，构建了一个大容量的存储池，像底层存储器一样廉价，而又可以达到顶层存储器的速度。

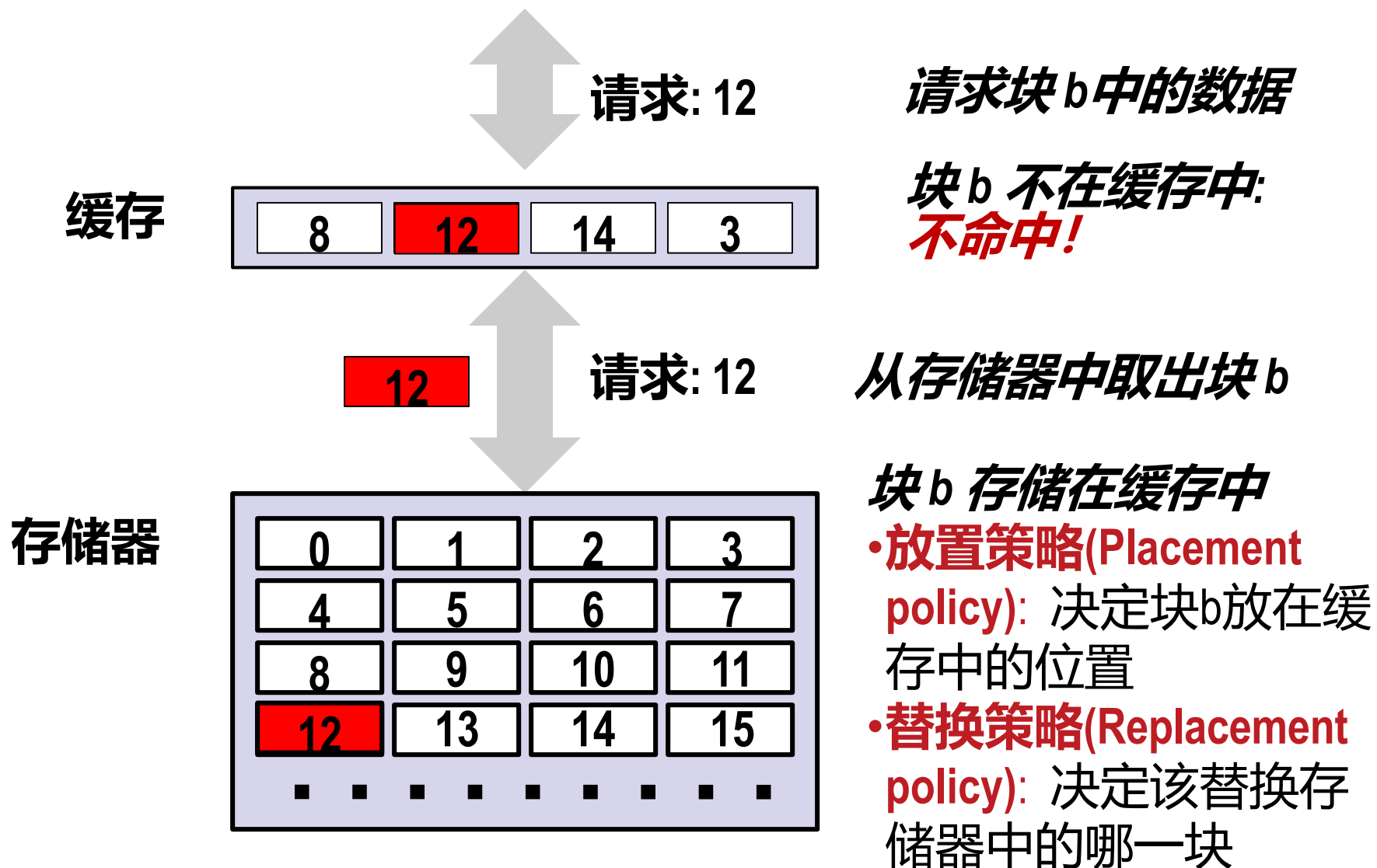
高速缓存的基本概念



高速缓存的基本概念: 命中



高速缓存的基本概念: 不命中



高速缓存基本概念:缓存不命中的种类

■ 冷不命中（或强制性不命中）

- 当缓存为空时，对任何数据的请求都会不命中

■ 冲突不命中

- 大部分缓存将第 $k+1$ 层的某个块限制在第 k 层块的一个子集里(有时只是一个块)
 - 例如，第 $k+1$ 层的块 i 必须放置在第 k 层的块 $(i \bmod 4)$ 中
- 当缓存足够大，但是被引用的对象都映射到同一缓存块中，此种不命中称为冲突不命中

例：程序请求块 0, 8, 0, 8, 0, 8, 每次请求都不命中

■ 容量不命中

- 当工作集(working set)的大小超过缓存的大小时，会发生容量不命中

存储器层次结构中的缓存

缓存类型	缓存什么	存在何处	延迟(周期数)	谁管理
寄存器	4-8 字节字	CPU 核心	0	编译器
TLB	地址译码	片上 TLB	0	硬件MMU
L1 高速缓存	64字节块	片上 L1	4	硬件
L2 高速缓存	64字节块	片上 L2	10	硬件
虚拟内存	4KB 页	主存	100	硬件 + OS
缓冲区缓存	部分文件	主存	100	OS
磁盘缓存	磁盘扇区	磁盘控制器	100,000	磁盘固件
网络缓冲区缓存	部分文件	本地磁盘	10,000,000	NFS 客户
浏览器缓存	Web页	本地磁盘	10,000,000	Web浏览器
Web缓存	Web 页	远程服务器 磁盘	1,000,000,000	Web代理 服务器

总结

- CPU、主存、大容量存储设备之间的速度差距持续扩大
- 编写良好的程序表现出良好的局部性
- 利用局部性特点，基于高速缓存的存储器层次结构有利于缩小速度差距

补充:

DRAM内存组织结构

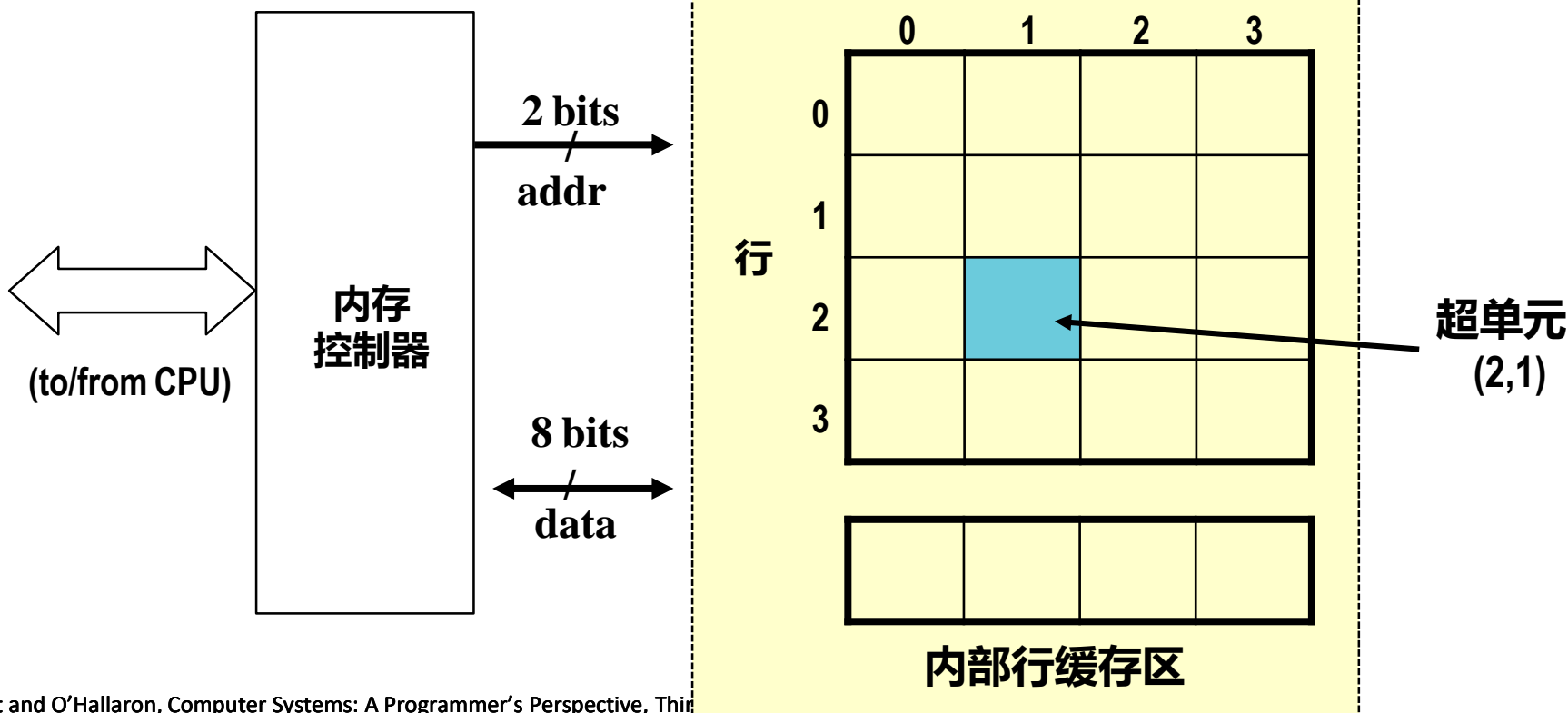
读/写操作

传统 DRAM 组织结构

■ $d \times w$ DRAM:

- DRAM芯片中的单元被分成 d 个超单元，每个超单元都由 w 个DRAM单元组成。一个 $d \times w$ 的DRAM总共存储 dw 位信息。

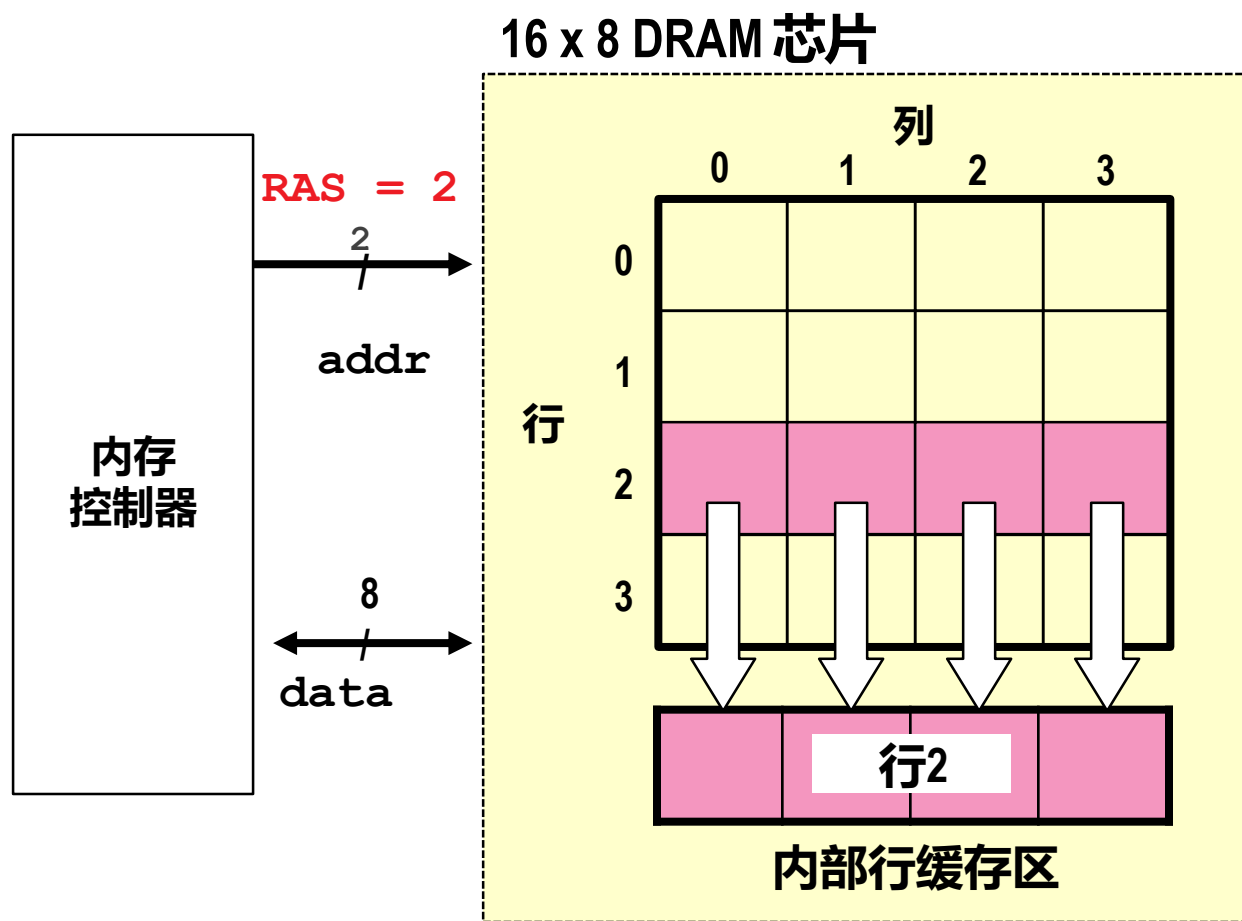
16 x 8 DRAM 芯片



读 DRAM 超单元 (2,1)

Step 1(a): 行访问选通脉冲(**RAS**) 选中行 2。

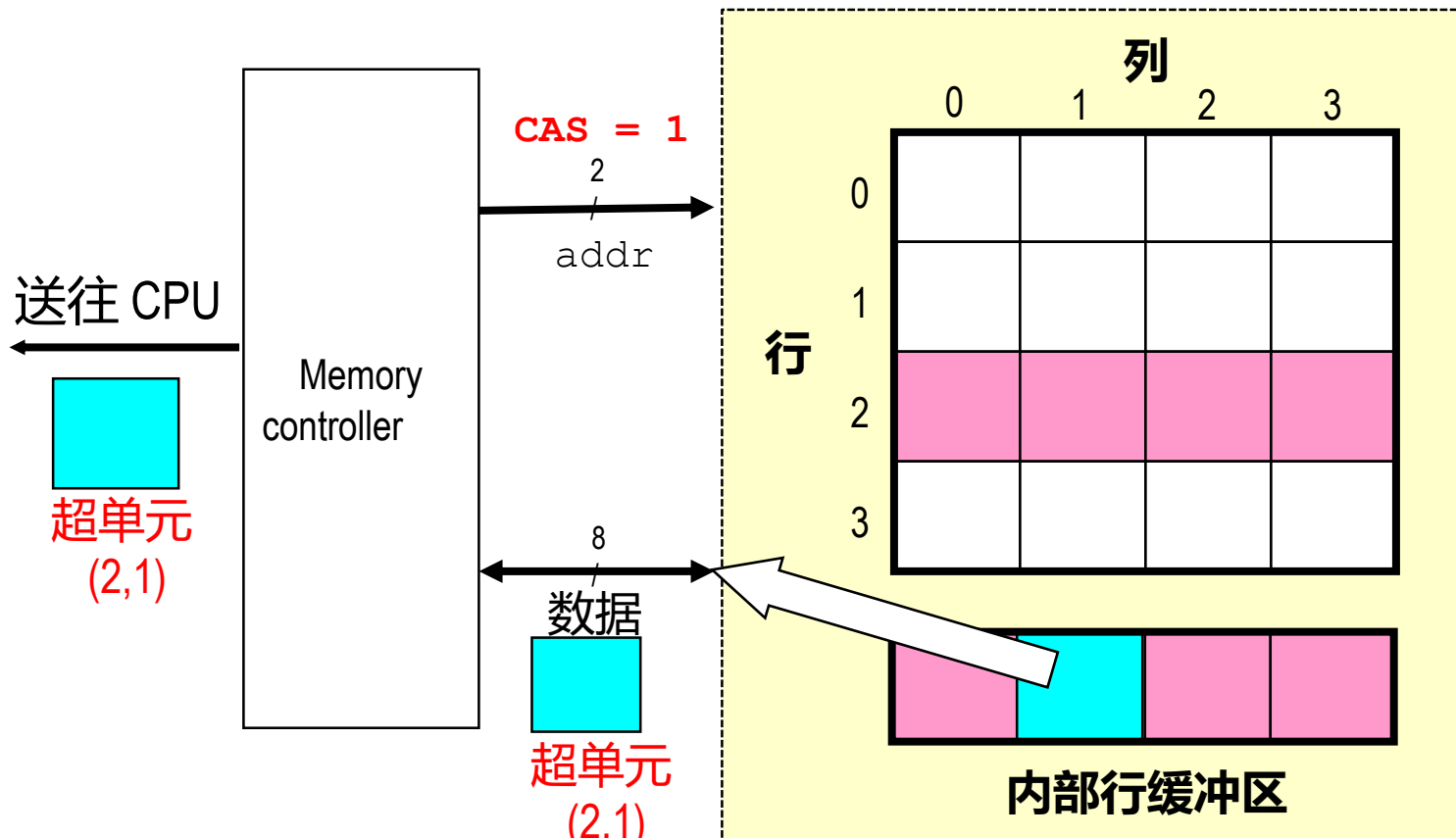
Step 1(b): 行 2 的整个内容复制到内部行缓存区。



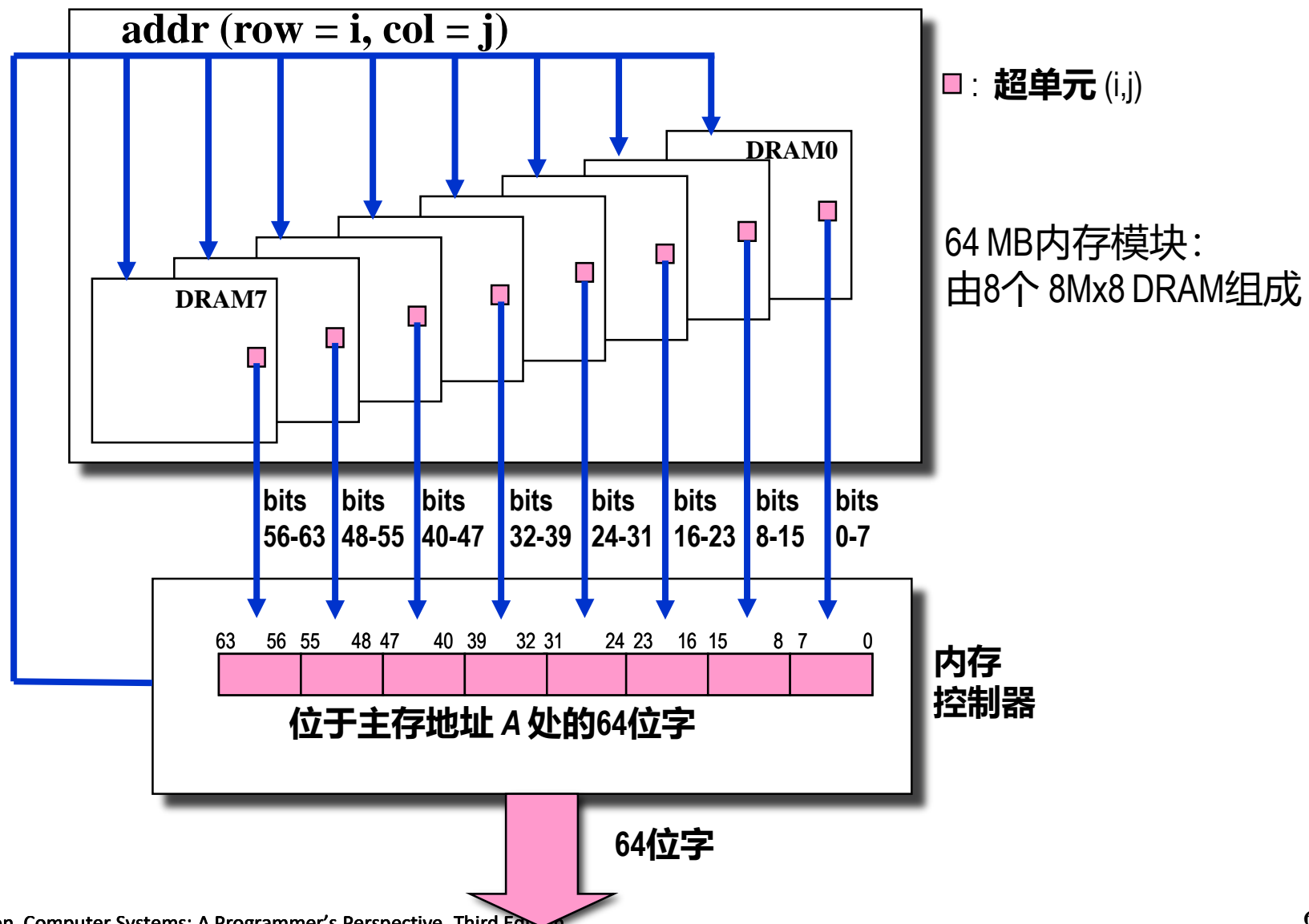
读 DRAM 超单元 (2,1)

- Step 2(a): 列访问选通脉冲 (**CAS**) 选中列 1。
- Step 2(b): 超单元 (2,1) 从内部行缓存区复制到data线上，最终发送给 CPU。

16 x 8 DRAM 芯片



内存模块



存储技术趋势

SRAM

度量标准	1985	1990	1995	2000	2005	2010	2015	2015:1985
美元/MB	2,900	320	256	100	75	60	25	116
访问时间 (ns)	150	35	15	3	2	1.5	1.3	115

DRAM

度量标准	1985	1990	1995	2000	2005	2010	2015	2015:1985
美元/MB	880	100	30	1	0.1	0.06	0.02	44,000
访问时间 (ns)	200	100	70	60	50	40	20	10
典型的大小 (MB)	0.256	4	16	64	2,000	8,000	16,000	62,500

磁盘

度量标准	1985	1990	1995	2000	2005	2010	2015	2015:1985
美元/GB	100,000	8,000	300	10	5	0.3	0.03	3,333,333
访问时间 (ms)	75	28	10	8	5	3	3	25
典型的大小 (GB)	0.01	0.16	1	20	160	1,500	3,000	300,000

CPU时钟频率

计算机历史的转折点——设计师撞上“电源墙”

	1985	1990	1995	2003	2005	2010	2015	2015:1985
CPU	80286	80386	Pentium	P-4	Core 2	Core i7(n)	Core i7(h)	
时钟 频率(MHz)	6	20	150	3,300	2,000	2,500	3,000	500
时钟 周期 (ns)	166	50	6	0.30	0.50	0.4	0.33	500
核数	1	1	1	1	2	4	4	4
有效 时钟 周期(ns)	166	50	6	0.30	0.25	0.10	0.08	2,075

(n) Nehalem 处理器
(h) Haswell 处理器