北京郵電大學
BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

Computer Organization and Architecture

知识点梳理（一）

School of Computer Science（National Pilot Software Engineering School）
AO XIONG (熊翱)
xiongao@bupt.edu.cn

第一章

- 基本概念
  - 组织与架构
  - 结构与功能
- 重点知识点
  - 计算机的主要部件
  - 指令执行过程
  - 组织架构在体系中的位置

基本概念

- 体系架构Architecture： 程序员可以看到的属性，包括指令集，用于数据表示的位数、I/O机制、寻址技术等
- 组织Organization： 架构是怎么实现的，例如通过什么控制信号来实现控制，采用什么接口，用什么内存技术
- 结构Structure： 计算机各个组件之间相互关联的方式
- 功能Function： 组成计算机的组件可以进行的操作

知识点

- 计算机的主要部件
  - 处理器CPU
  - 内存
  - 磁盘控制器和磁盘
  - 输入输出设备，如键盘，显示器，还有图形适配器
  - 内部Cache
  - 连接各个部件的总线
  - 其他部件

知识点

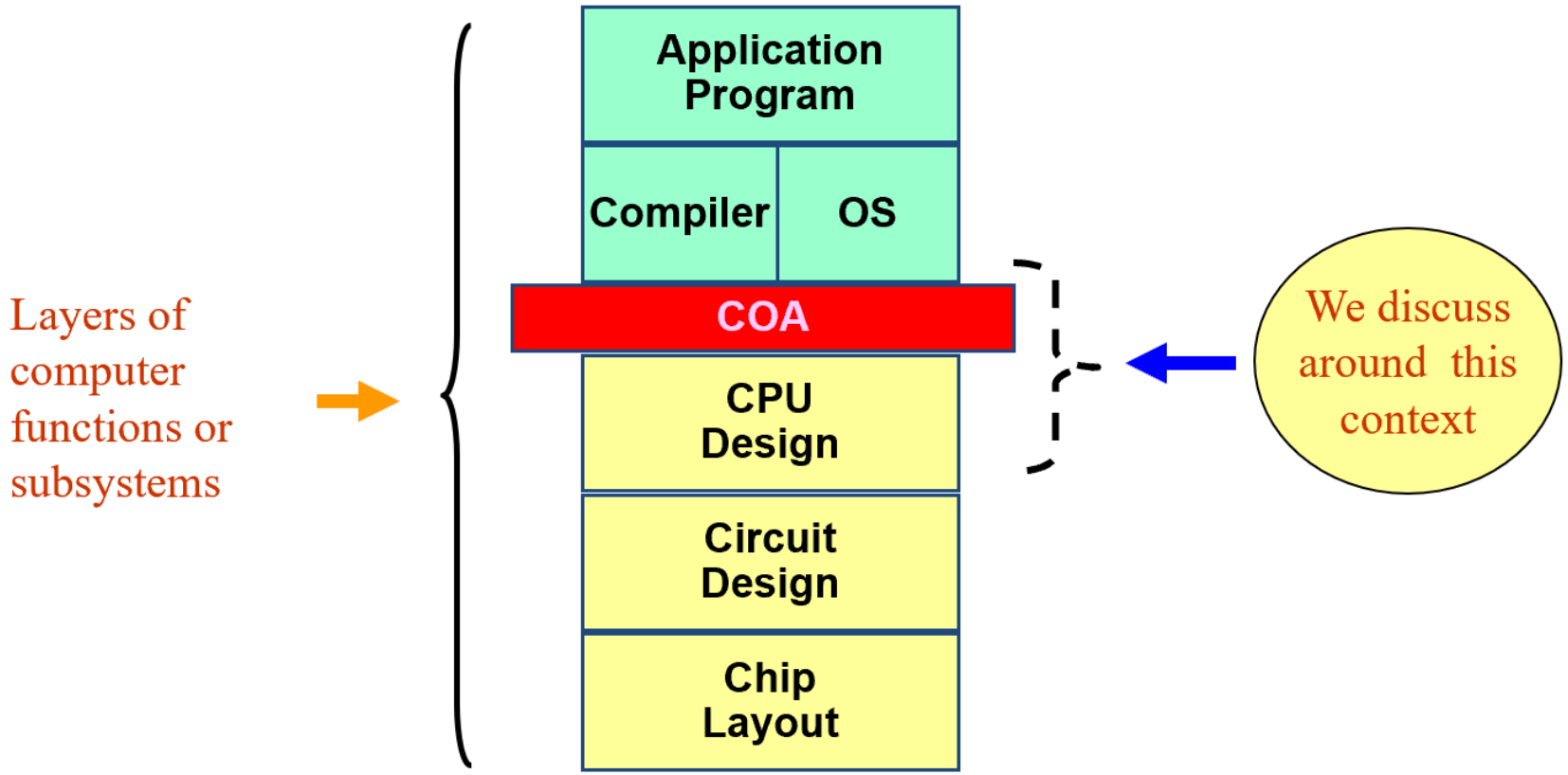
- 指令简单的执行过程

Add R1, R2

a. $ALU_1 \leftarrow [R1]$	{content of R1 is moved to ALU_1}
b. $ALU_2 \leftarrow [R2]$	{content of R2 is moved to ALU_2}
c. ADD	{content of ALU_1 + ALU_2 = ALU_3}
d. $R1 \leftarrow [ALU_3]$	{Result of addition is moved to R1

知识点

- 组织架构在体系中的位置



第二章

- 基本概念
  - 冯诺依曼机
  - 系列机
  - 摩尔定律
  - 平衡思想
- 重点知识点
  - 计算机的发展代次
  - 性能评估方法
  - 阿姆达定律
  - 基准测试的评估计算

基本概念

- 冯诺依曼机
  - 有一个主存储器，主存储器内可以存储数据和指令；
  - 数据和指令都以二进制的方式存储在主存储器中；
  - 数字逻辑单元ALU执行基本的操作；
  - 数据和指令都存储在主存储中，通过地址来进行寻址；
  - 由控制单元CU来确定合适的操作顺序。

基本概念

- 系列机
  - 具有相同或几乎相同的指令集；
  - 相同或相似的操作系统；
  - 随着系列的提高，计算机的速度逐步提升；
  - 计算机的I/O端口的数量逐步提升；
  - 计算机的主存容量的逐步增加；
  - 计算机的成本也随之增加。

基本概念

- 摩尔定律
  - 1965年发现芯片上的晶体管数量每年翻一倍，他断言芯片上晶体管的数量增长会持续维持这个趋势。事实证明了这个断言。到1970年之后，这个趋势下降为每18个月翻一倍，之后又长期维持了这个趋势
  - 摩尔定律的意义
    - 芯片的成本几乎没有变化：计算机的成本下降了
    - 更高的封装密度意味着更短的电子路径，提供更高的速度
    - 体积更小，便于放置在各种环境中
    - 降低了电源和冷却要求
    - 更少的互连增加了可靠性

基本概念

- 平衡思想
  - 计算机由多个部件组成，并且每个组件的性能存在较大的差异。因此解决计算机性能问题的关键是平衡，也就是要在各个组成部件之间找到一个最佳的组合方案，在处理器、主存、IO设备以及互连结构的吞吐量和处理要求之间进行平衡。
  - 计算机架构设计中，不断进行设计的更新，以达到处理器、内存、I/O设备以及互连结构之间的平衡，以实现最大的整体性能

重点知识点

- 计算机的发展代次
  - 第一代：真空管
  - 第二代：晶体管
  - 第三代：集成电路

重点知识点

- 性能评估方法CPI
  - CPU constant frequency(固定频率) f
  - constant cycle time τ, where $\tau = 1/f$.
  - Instruction count I_c
  - CPI

$$CPI = \frac{\sum_{i=1}^{n}(CPI_i \times I_i)}{I_c}$$

  - MIPS

$$\text{MIPS rate} = \frac{I_c}{T \times 10^6} = \frac{f}{CPI \times 10^6}$$

  - MFLOPS

$$\text{MFLOPS rate} = \frac{\textit{Number of executed floating-point operations in a program}}{\textit{Execution time} \times 10^6}$$

重点知识点

- 阿姆达定律
  - 假定一段代码中，有f部分是可以无限制地进行并行，并且没有调度产生的代价；1-f部分是必须串行运行的。如果这个程序在单处理器上的执行时间是T，那么，如果我们使用N个处理器来运行这个程序，运行速度的比例，也就是加速比可以用下面的公式来计算：

$$Speedup = \frac{\text{time to execute program on a single processor}}{\text{time to execute program on } N \text{ parallel processors}} = \frac{T(1-f)+Tf}{T(1-f)+\frac{Tf}{N}} = \frac{1}{(1-f)+\frac{f}{N}}$$

  - 计算机的处理程序适应并行执行环境，这样采用多核处理器才有意义

重点知识点

- 基准测试的评估计算
  - 算术平均：计算平均执行速度

$$R_A=\frac{1}{m}\sum_{i=1}^{m} Ri$$

  - 调和平均：指令的平均执行时间

$$R_B=m/\sum_{i=1}^{m} 1/Ri$$

- 基本概念
  - PC、IR、MAR、MBR、ALU、CU
  - 指令周期
  - 中断
  - 互联结构
  - 总线
  - 总线仲裁
  - 总线带宽
  - 总线速度

重点知识点

- 指令的执行过程，包括带中断的和不带中断的执行过程
- 中断的目的，中断处理方式，多重中断的处理
- 总线的类型，总线的使用方式，总线结构
- 总线设计中的要素：类型，仲裁方式，时序，带宽等，每个设计要素的特点
- 同步时序和异步时序
- 数据总线、地址总线、控制总线宽度的含义

基本概念

- PC、IR、MAR、MBR、ALU、CU
  - PC：Program counter　程序计数器
  - IR ：Instruction register　指令寄存器
  - MAR ：Memory address register　存储器地址寄存器
  - MBR ：Memory buffer register　存储器缓冲寄存器
  - I/O AR ：　I/O address register　IO地址寄存器
  - I/O BR ：　I/O buffer register　IO缓冲寄存器
  - AC：Accumulator　累加器
  - ALU：Arithmetic and Logic Unit 算术逻辑单元
  - CU：　Control Unit　控制单元

基本概念

- 指令周期
  - 一条指令要求的处理过程称为一个指令周期
  - 指令周期包含两个步骤，读取指令和执行指令，这两个步骤称为取指周期和执行周期
  - 取指周期中，计算机进行指令的读取
  - 执行周期中，计算机执行这个指令。然后继续读取下一条指令
  - 中断周期：进行中断的处理
  - 间接周期：间接寻址

基本概念

- 中断
  - 中断实际上是一种处理机制，这种机制允许其他模块，比如说I/O模块打断当前的指令执行顺序，而执行另外的一段指令。执行完成之后，再返回到中断前的状态，继续执行之前的指令
  - 目的：减少CPU的等待时间，提高CPU的效率

基本概念

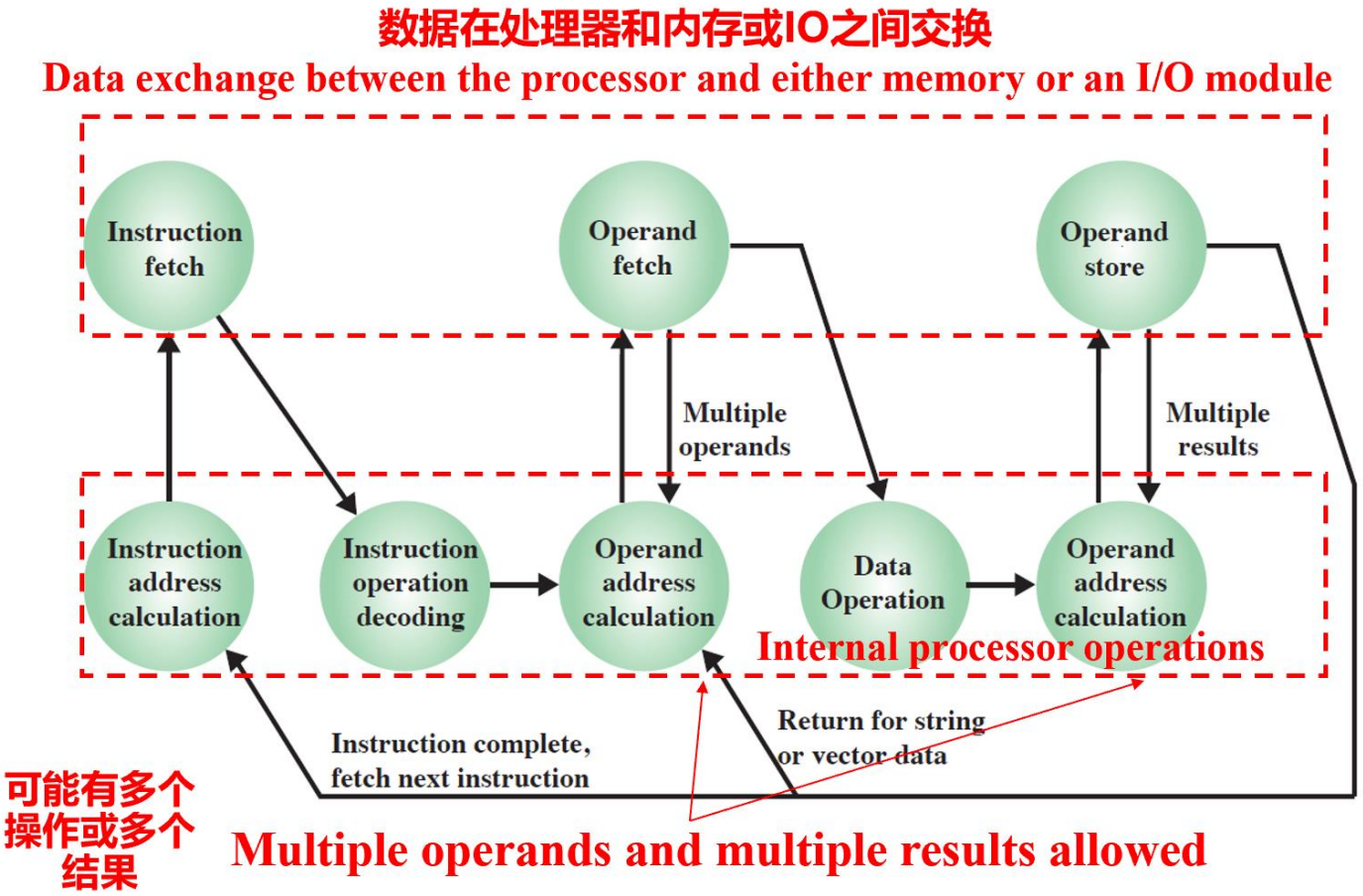
- 互联结构 Interconnection structure
  - 互联结构是连接各个模块之间的通道的集合。这些通道将CPU、内存、I/O模块等连接在一起，进行数据交换。
  - 互联结构的具体要求需要根据模块之间需要进行交换的内容来确定
  - 总线结构是互联结构的一种形式，并且得到了广泛的使用

基本概念

- 总线
  - 连接2个或多个设备的通信通道
  - 一般采用广播方式进行数据传输。多个设备都连接到总线上，任何一个设备发送的数据，在总线上的所有设备都能收到。如果两个设备同时发送信号，将无法正常进行数据接收。因此，必须采用某种仲裁方式，每个时间段只能有一个设备进行数据的发送
  - 一般采用组的方式，将多条线放在一起，形成一组总线，同时可以传送多位。一组总线中传输线的数量，称为总线宽度
  - 一般分为数据总线，地址总线和控制总线
  - 总线速度： 1秒钟通过每根导线能传多少个bit
  - 总线带宽：也叫吞吐量，总线传输速率，指的是通过总线1秒钟能传输的总数据量

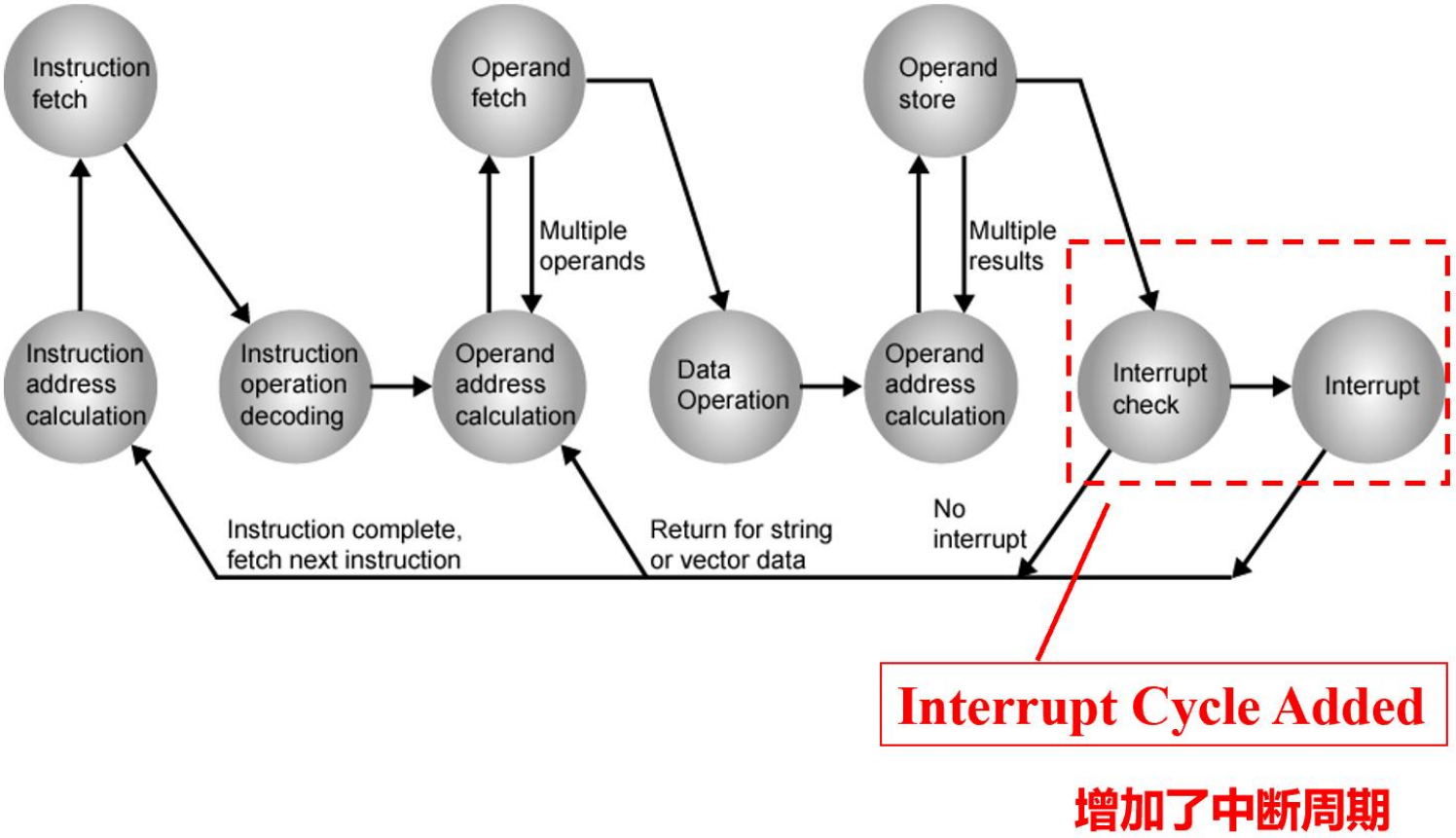
重点知识点

- 指令的执行过程，不带中断的执行过程



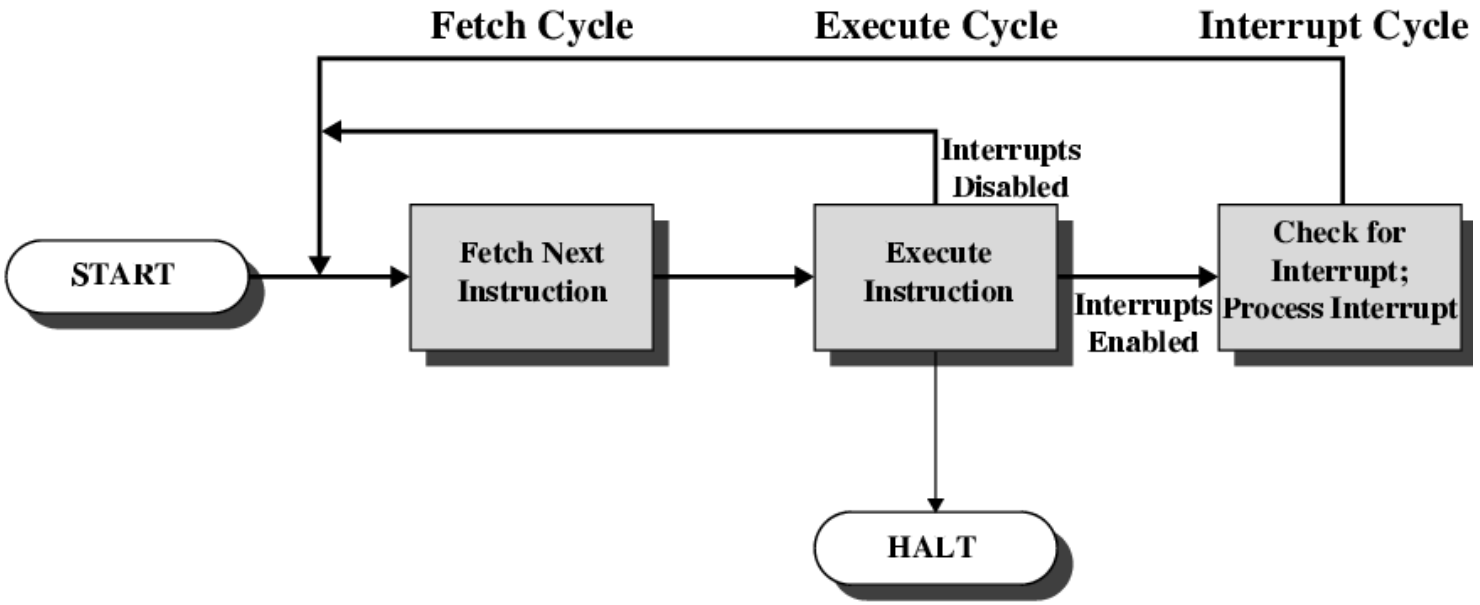
重点知识点

- 指令的执行过程，带中断的执行过程



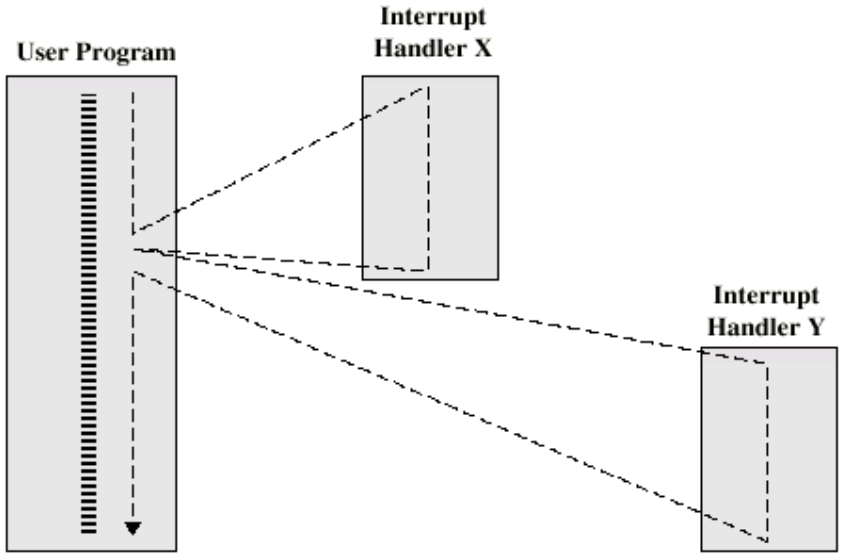
重点知识点

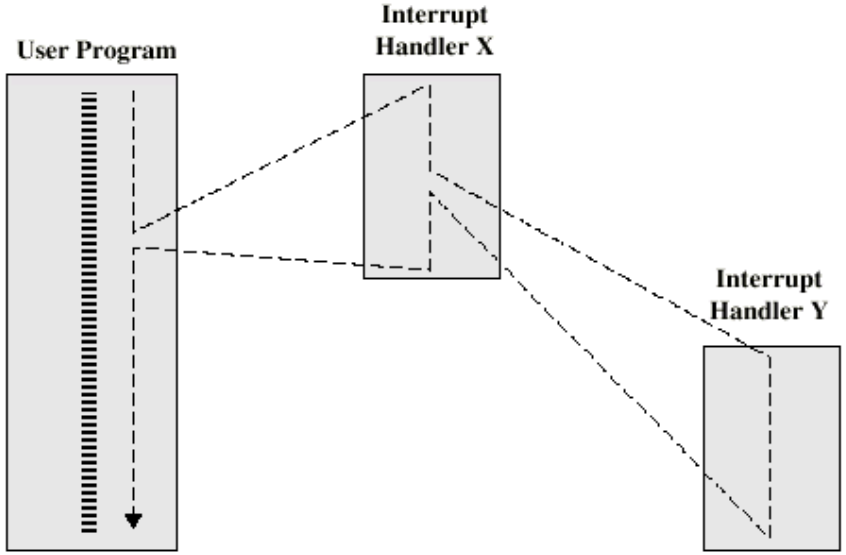
- 中断处理方式



重点知识点

- 多重中断的处理
  - 顺序处理中断
  - 中断嵌套





重点知识点

- 总线的类型
  - 系统总线：连接计算机主要部件的总线称为系统总线
  - 数据总线：提供模块之间传输数据的通道。总线中的数据线越多，那么同时传送的数据就越多
  - 地址总线：用于确定数据在存储器中的位置。地址线越多，能够寻址的存储器的空间也越大
  - 控制总线：用于发送控制信号。控制总线中控制线越多，能发出的控制信号也能越复杂

重点知识点

- 总线的使用方式
  - 假定一个模块需要发送数据给另一个模块，它需要这样去做：

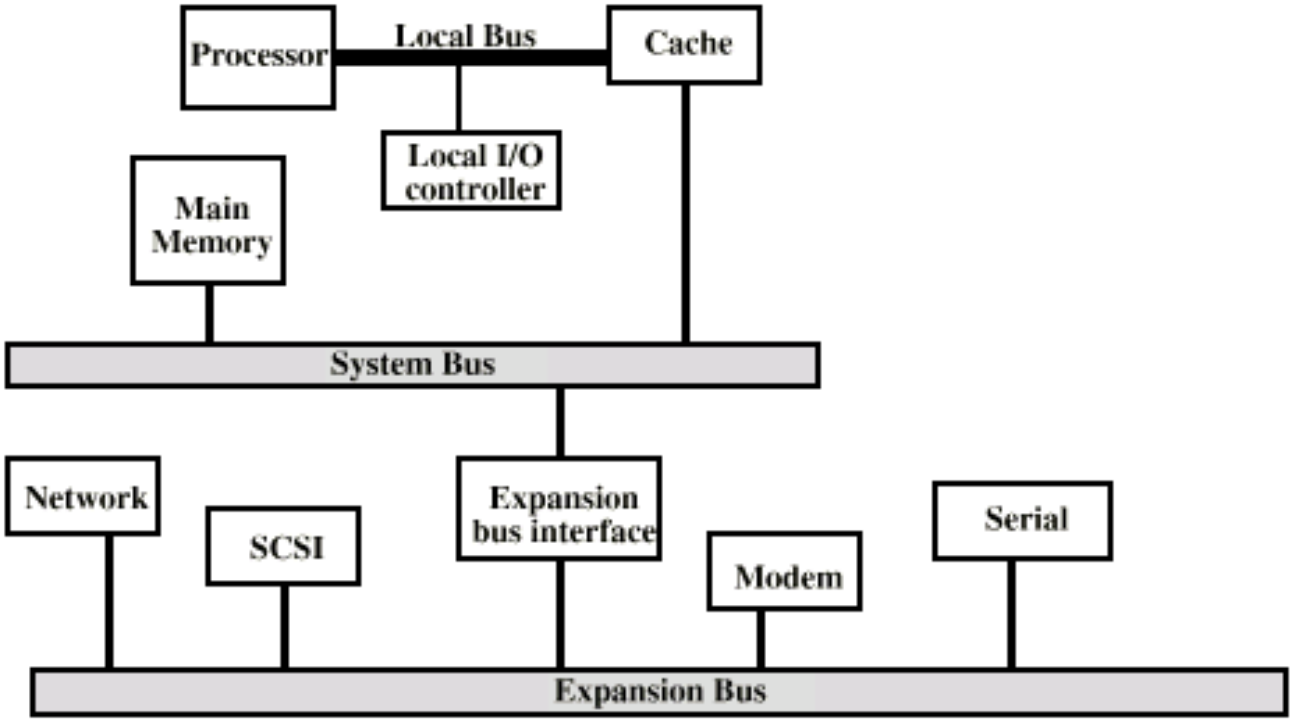
1. 获取总线的使用权；
2. 获得总线的使用权之后，再通过总线发送数据。

  - 假定一个模块需要向另一个模块请求数据，它需要这样去做：

1. 获取总线的使用权；
2. 通过控制和地址总线向那个模块发送数据请求；
3. 等待那个模块发送数据过来。

重点知识点

- 总线结构
  - 单总线结构
  - 多总线结构



重点知识点

- 总线设计要素：类型
  - 专用总线：只负责某一方面的工作，和其他工作不冲突
  - 共享总线
    - 多种功能都用同一个总线
    - 线少
    - 控制复杂，性能受限

重点知识点

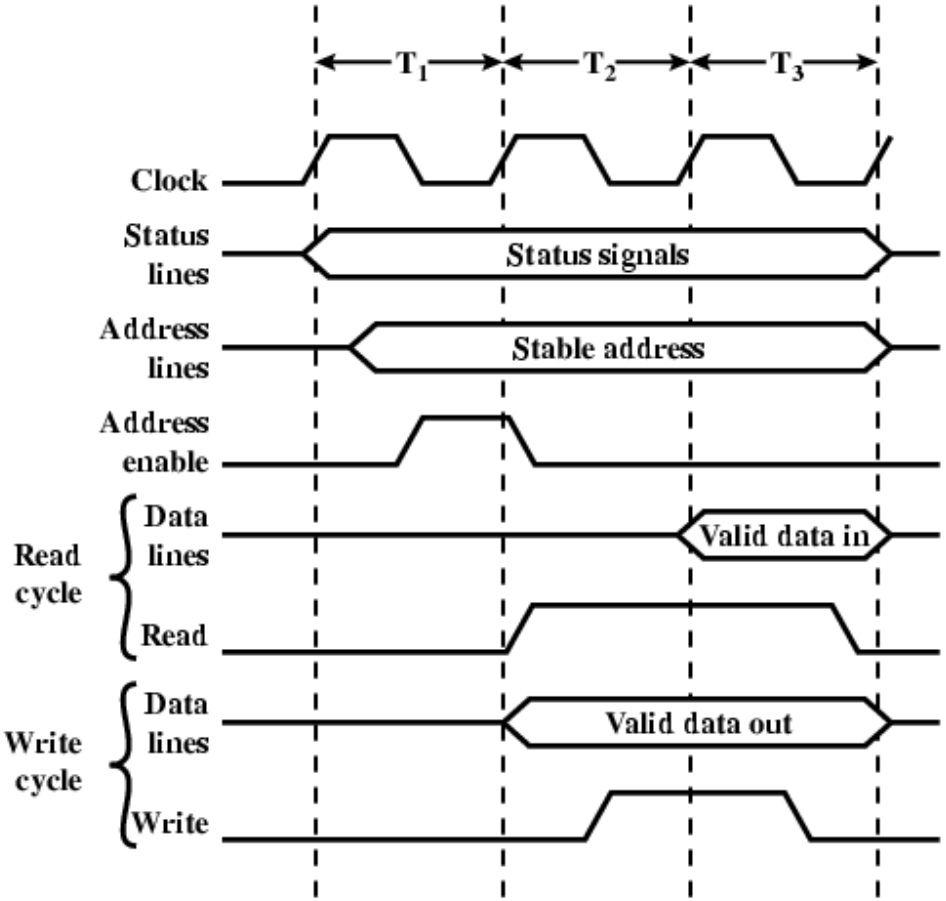
- 总线设计要素：仲裁方式
  - 多个模块需要控制总线的时候，需要有一个仲裁机构来判断，谁来使用这个总线
  - 集中式仲裁
    - 链式询问方式
    - 计数器定时询问
    - 独立请求
  - 分布式仲裁

重点知识点

- 总线设计要素：时序
  - 作用：确定总线上协调时间的方式
  - 同步时序：总线上事件的发生由一个同步时钟来决定
  - 异步时序：总线上事件的发生由该事件的上一个事件来决定

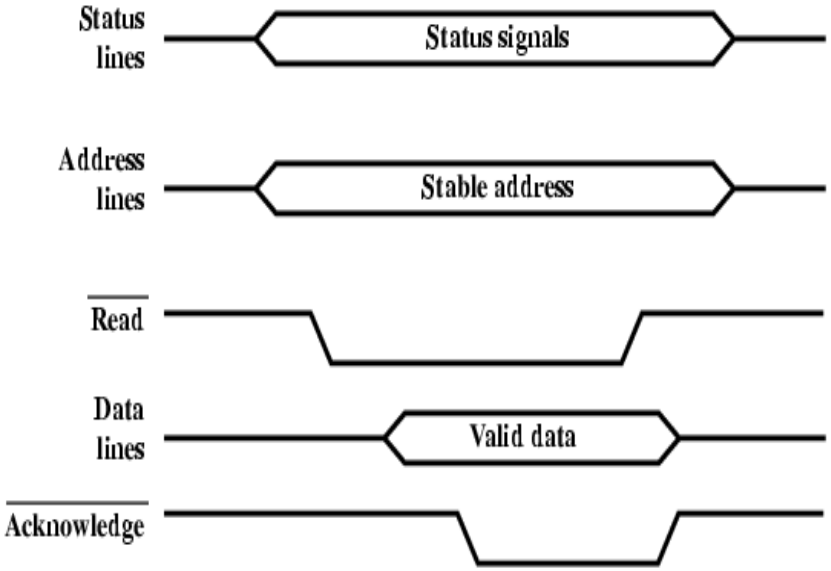
重点知识点

- 总线设计要素：同步时序

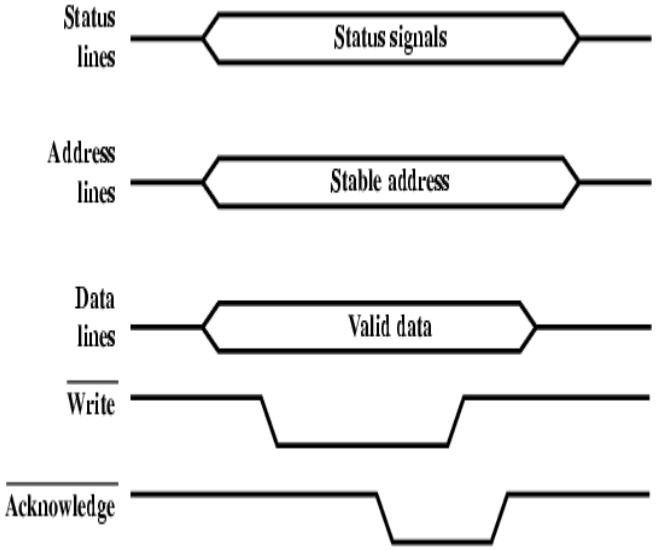


重点知识点

- 总线设计要素：异步时序



异步读



异步写

重点知识点

- 总线设计要素：宽度
  - 数据数据总线的宽度决定系统性能：一次传输的数据位数越多，系统性能越高
  - 地址总线的宽度决定了系统的容量：寻址的空间越大，这也意味着信息存储的越多
  - 控制总线的宽度决定了操作的灵活性：控制总线越宽，控制信号类型越多，操作越灵活

重点知识点

- 总线设计要素：带宽
  - 总线速度：每根线上每秒传输的比特数
  - 总线带宽：也称为吞吐量，总线传输速度，指的是通过这个总线每秒传输的总数据量。带宽的单位可以用每秒bit数或每秒字节数。
  - 带宽=总线宽度*总线速度

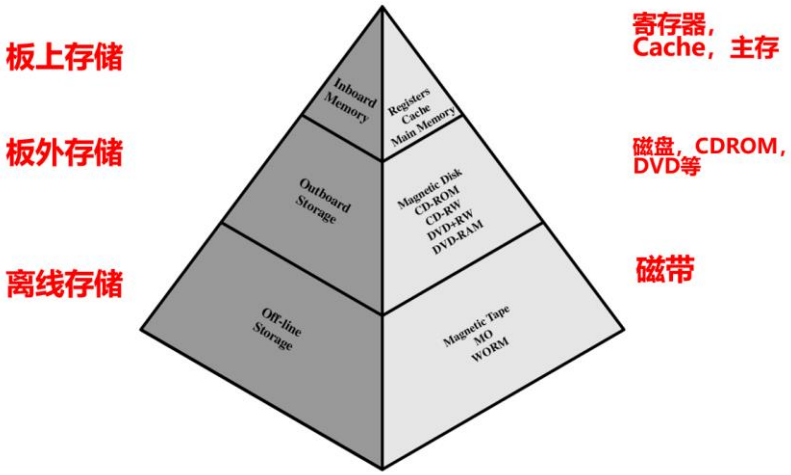
- 基本概念
  - 存储系统层次结构
  - Cache
  - Cache的行、存储器的块
  - 命中率与失效率
  - 映射算法
  - 写策略
  - 多级cache的概念

第四章

- 重点知识点
  - 存储系统的特征
  - 命中率、访问时间的计算
  - 直接映射算法
  - 全相联映射算法
  - 组相联映射算法
  - 替换算法

基本概念

- 存储系统层次结构
  - 目的：通过多层存储的方式，来解决存储器的容量、性能和价格的问题
  - 包括不同的层次的内存单元
  - 每一层的存储单元的容量、访问时间和价格都不一样

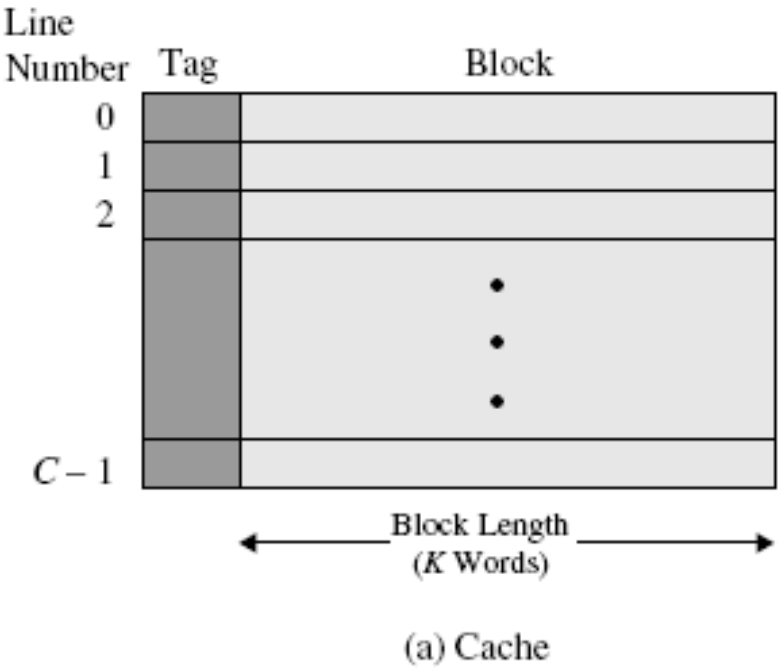


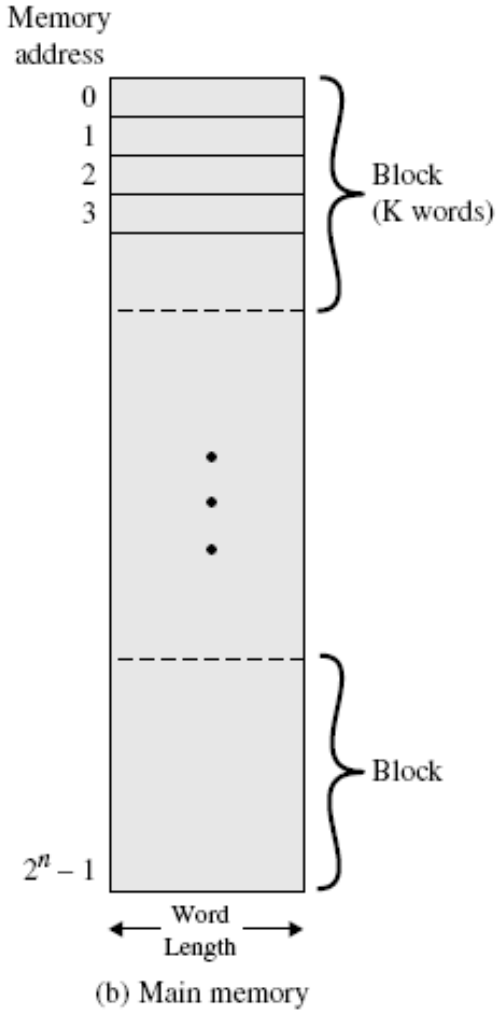
基本概念

- Cache
  - 容量小；
  - 速度快：基本上能跟上CPU的节奏；
  - 位于主存和CPU之间；
  - 可能是一个单独的模块，也可能就直接放在CPU芯片上；
  - 价格昂贵

基本概念

- Cache的行、存储器的块





基本概念

- 命中率与失效率
  - 命中：CPU请求数据在cache中，cache将内容返回给CPU，这称为一次命中
  - 失效：CPU请求的数据不在cache中，数据需要从内存中读取
  - 命中率：指的是上一层的数据请求在本层中能够完成的比例
  - 失效率：在本层中无法完成的比例。失效率=1-命中率
  - 失效惩罚：替换cache块的时间+传送数据给处理器的时间

基本概念

- 映射算法
  - 映射算法：内存中的块数M远大于cache的行数，内存中的块映射到cache中的特定行的算法
  - 直接映射
  - 相联映射
  - 组相联映射

基本概念

- 写策略
  - cache只是内存数据的映射，最终数据还是需要保存在内存中
  - cache的数据发生了变化，如果不采用适当的写策略，替换后cache的数据就丢了
  - 写直达
  - 写回法

基本概念

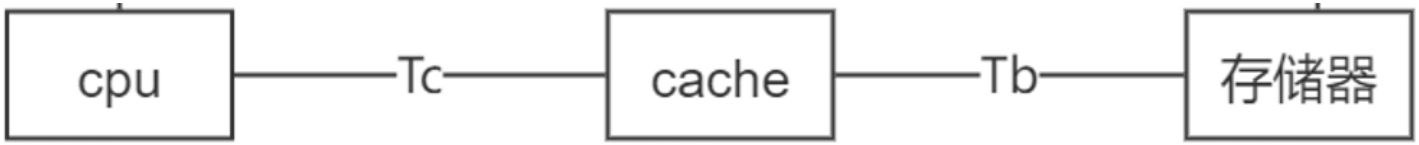
- 多级cache
  - 集成度的提高使得cache可以放在CPU芯片上
  - 常用的是既使用片上cache，也使用片外cache

重点知识点

- 存储系统特征
  - Location位置：内部，外部
  - Capacity容量：字节数，可寻址单元数
  - Unit of transfer传送单位
  - Access method访问方式：顺序存取，直接存取，随机存取，关联存取
  - Performance性能：访问时间，存储周期，传输速率
  - Physical type物理类型：半导体，磁介质，光学材料，其他
  - Physical characteristics物理特征：衰减，易失性，非易失性，可擦除性
  - Organisation组织形式

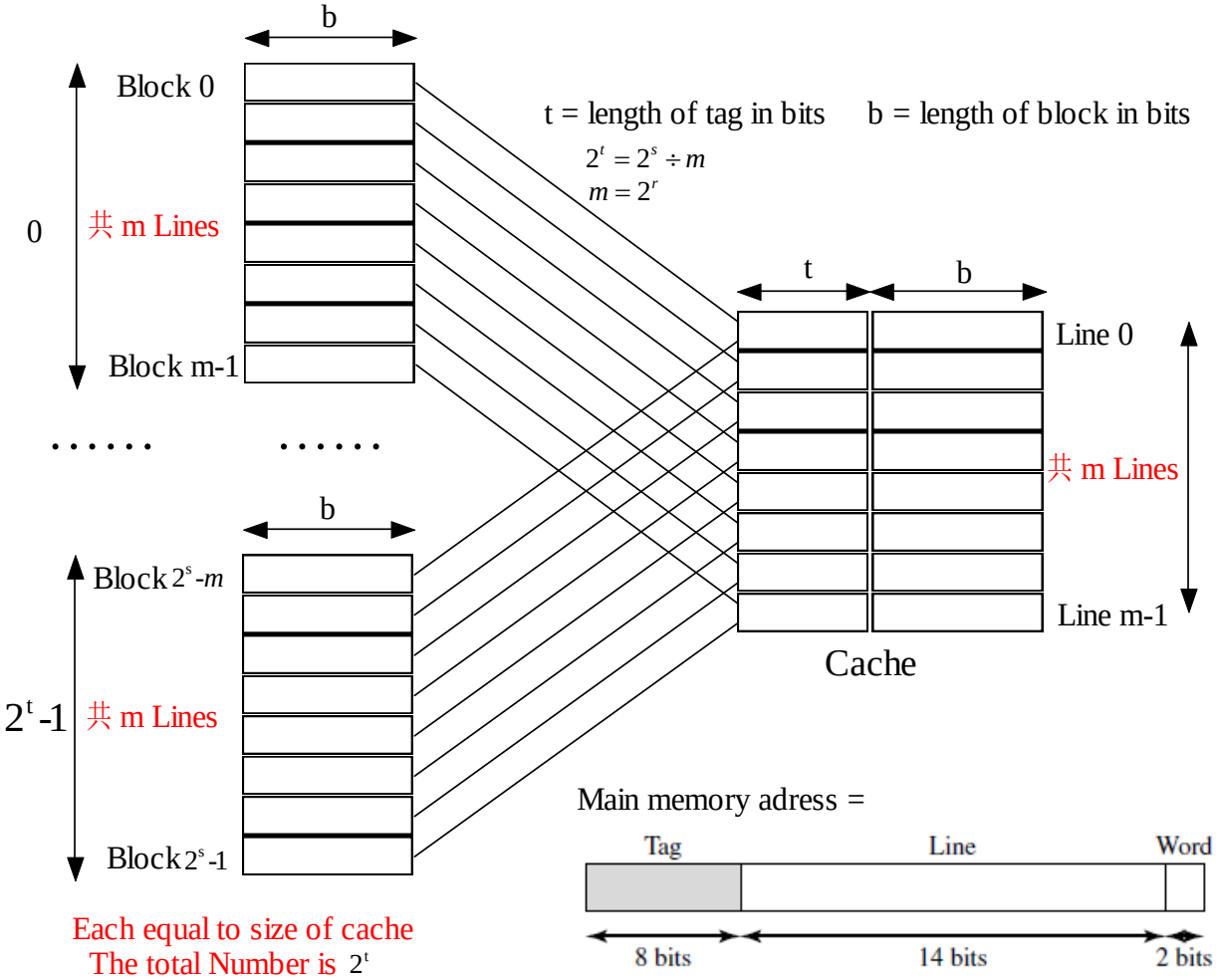
重点知识点

- 命中率、访问时间等计算
  - 根据命中率计算访问时间
  - 根据访问时间的预期，计算命中率
  - 读写性能综合计算
  - 考虑写回策略的计算



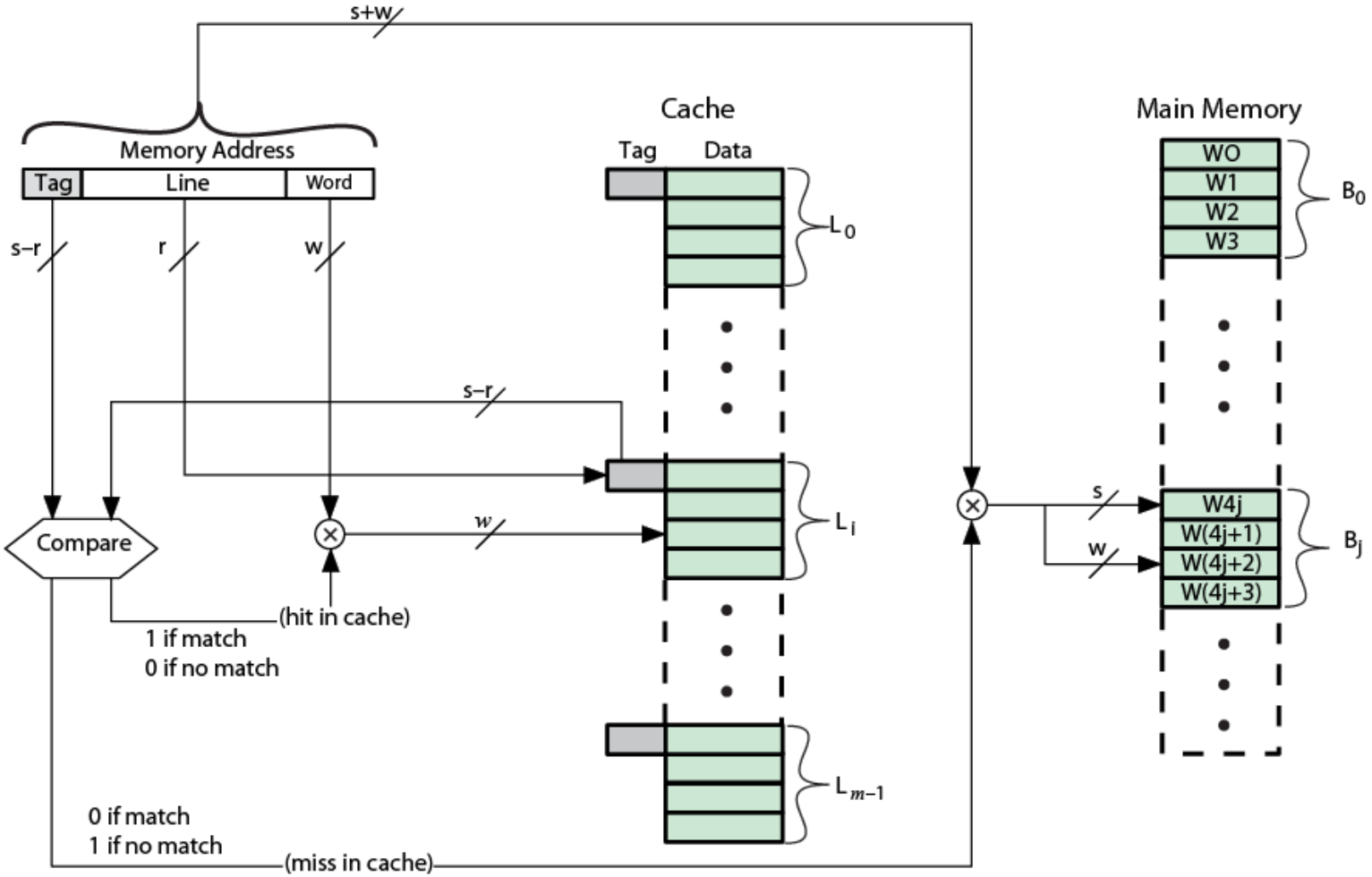
重点知识点

- 直接映射算法



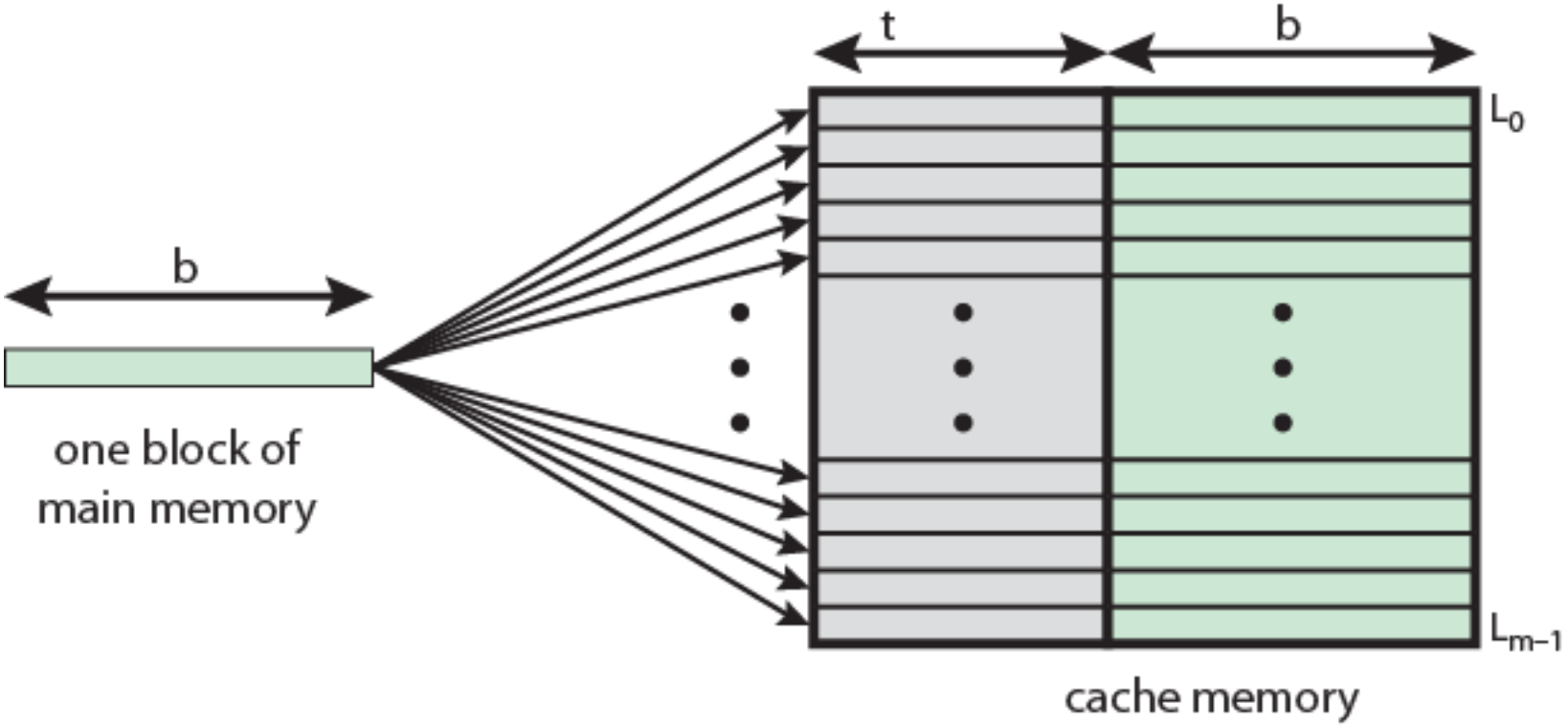
重点知识点

- 直接映射算法



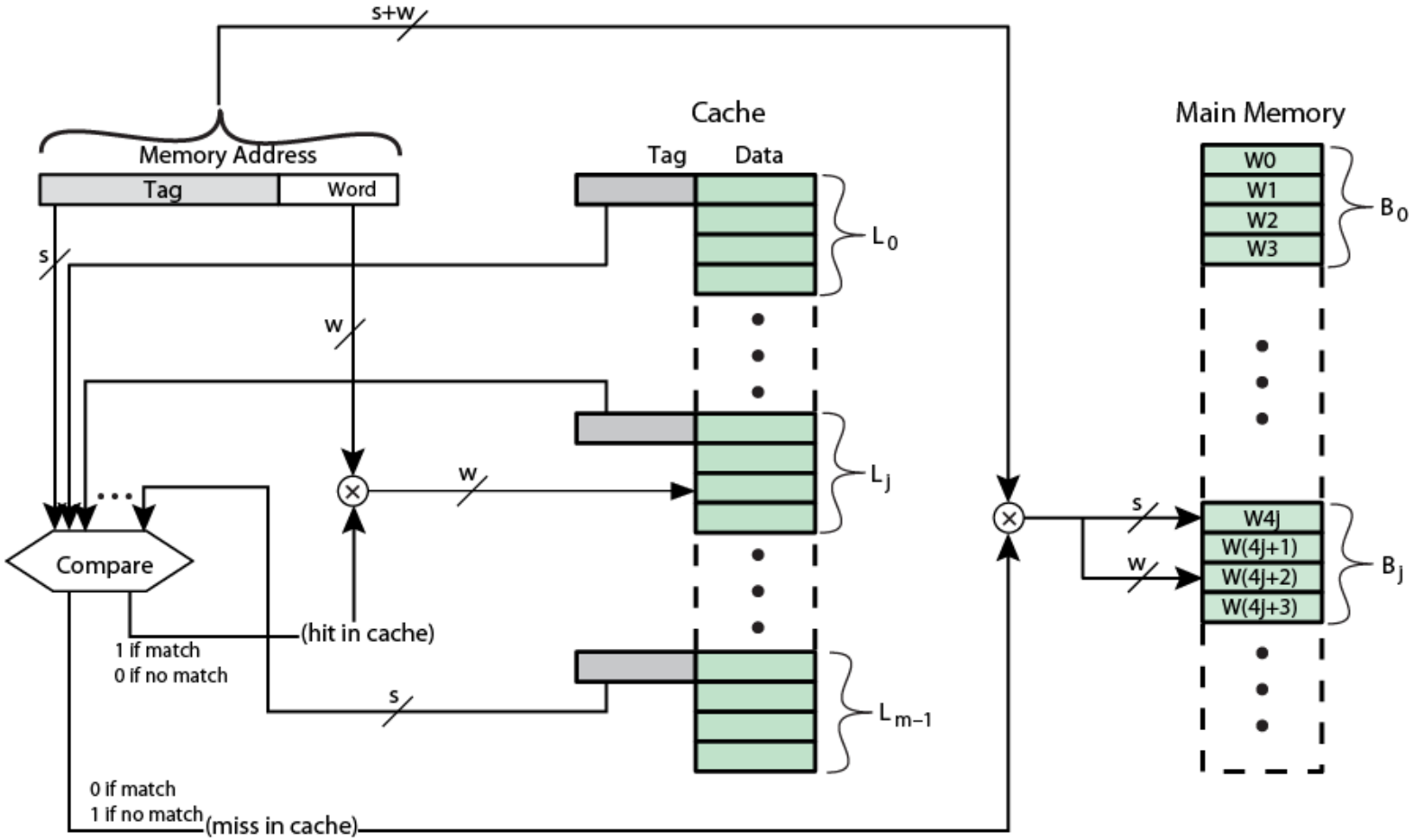
重点知识点

- 全相联映射算法



重点知识点

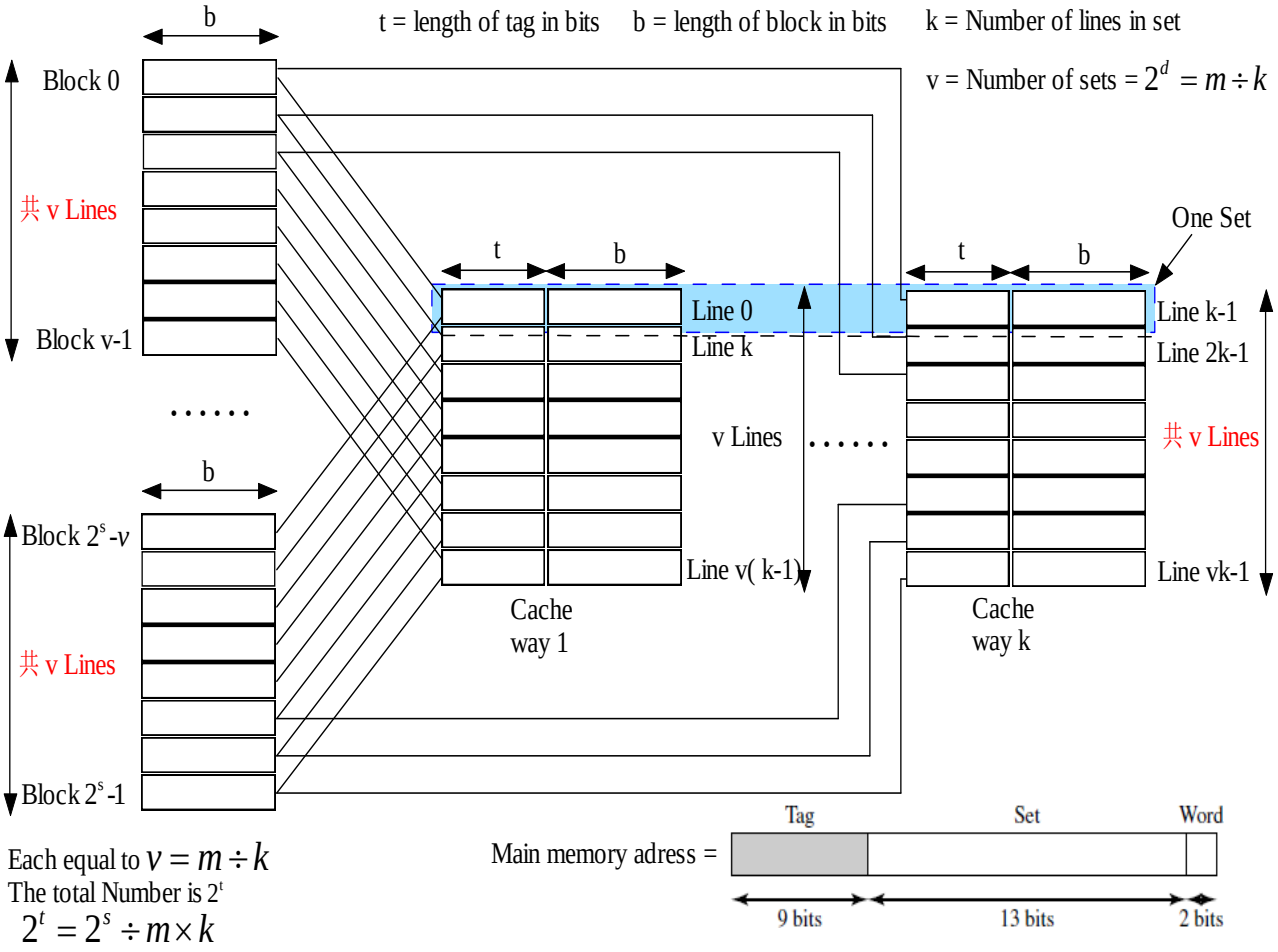
- 全相联映射算法



重点知识点

- K路组相联映射算法

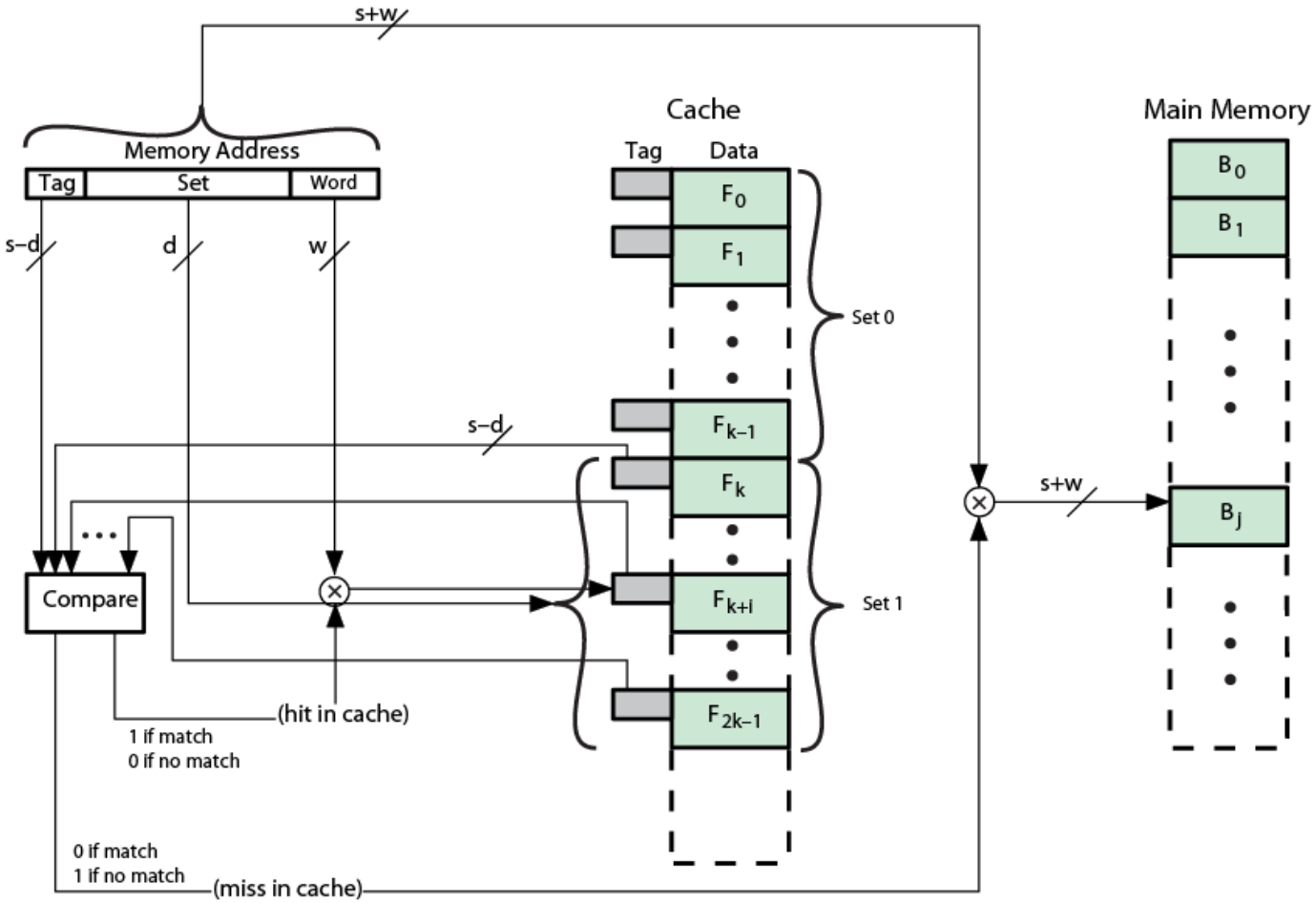
内存分为若干个组，每个组的块数等于v



t是标志位长度，b是cache行的长度，v是组数，k是组中的行数

重点知识点

- K路组相联映射算法



重点知识点

- 替换算法
  - 直接映射：只能替换到它对应的cache行
  - 相联映射：可以采用多种替换方法
    - Least Recently used (LRU)　最近使用原则
    - First in first out (FIFO)　先进先出原则
    - Least frequently used　最少访问频率原则
    - Random　随机原则

- 基本概念
  - SRAM和DRAM
  - ROM
  - 内存刷新
  - 纠错
  - 纠错码

重点知识点

- 内存的组织和访问
- 内存的扩展
- 地址解码
- 内存刷新
- 汉明纠错码

基本概念

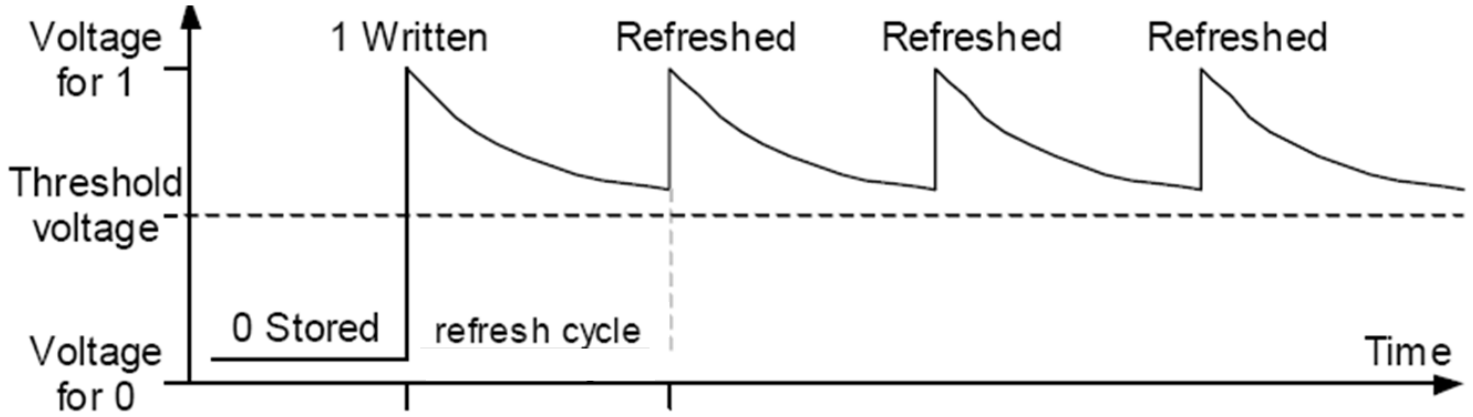
- SRAM和DRAM
  - 随机读取存储器
  - 易失性的存储器，需要持续供电
  - DRAM
    - 构造简单，一个晶体管和一个电容，模拟信号，集成度高，价格便宜。需要定时刷新。主要用作内存
  - SRAM
    - 构造复杂，6个晶体管组成的阵列，数字信号，每一位的体积大，价格昂贵。不需要定时刷新。主要用作cache

基本概念

- ROM
  - 只读存储器
  - 非易失性，永久存储
  - 重要的应用程序，如微程序，子程序库，BIOS等
  - 主要用于读，也能写的存储器
    - PROM，EPROM，EEPROM，Flash

基本概念

- 内存刷新
  - DRAM必须刷新
  - 每2~4ms刷新一次
  - 按行刷新



基本概念

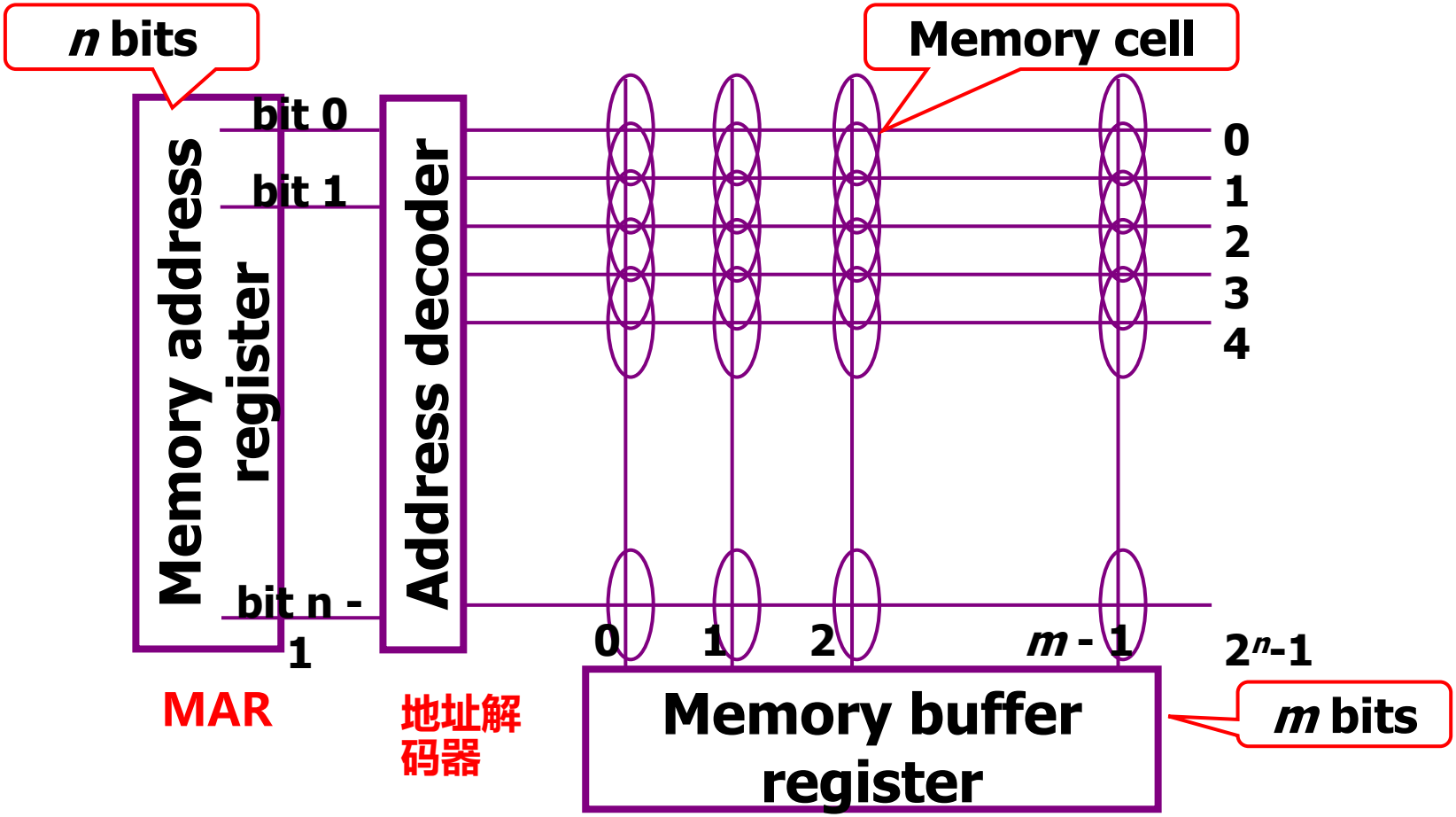
- 纠错和纠错码
  - 半导体器件容易因为多种原因导致错误
  - 大多数现在存储系统包含检错和纠错的逻辑
  - 通过附加位来实现检错和纠错，附加的位数称为纠错码
  - 纠错码会导致需要保存的字长增加，占用更多的存储空间
  - 常用的纠错码是汉明码，贝尔实验室的理查德.汉明发明。

重点知识点

- 内存组织
  - 半导体存储器一般都是封装成芯片，每个芯片里包含存储位元组成的阵列，可以存储N*M的位元数据
  - 将多个存储芯片组织一个阵列
    - 扩展内存使内存变大，可寻址空间更大
    - 扩展内存使内存变宽，每次读写的位数等多
  - 一维阵列
  - 二维阵列

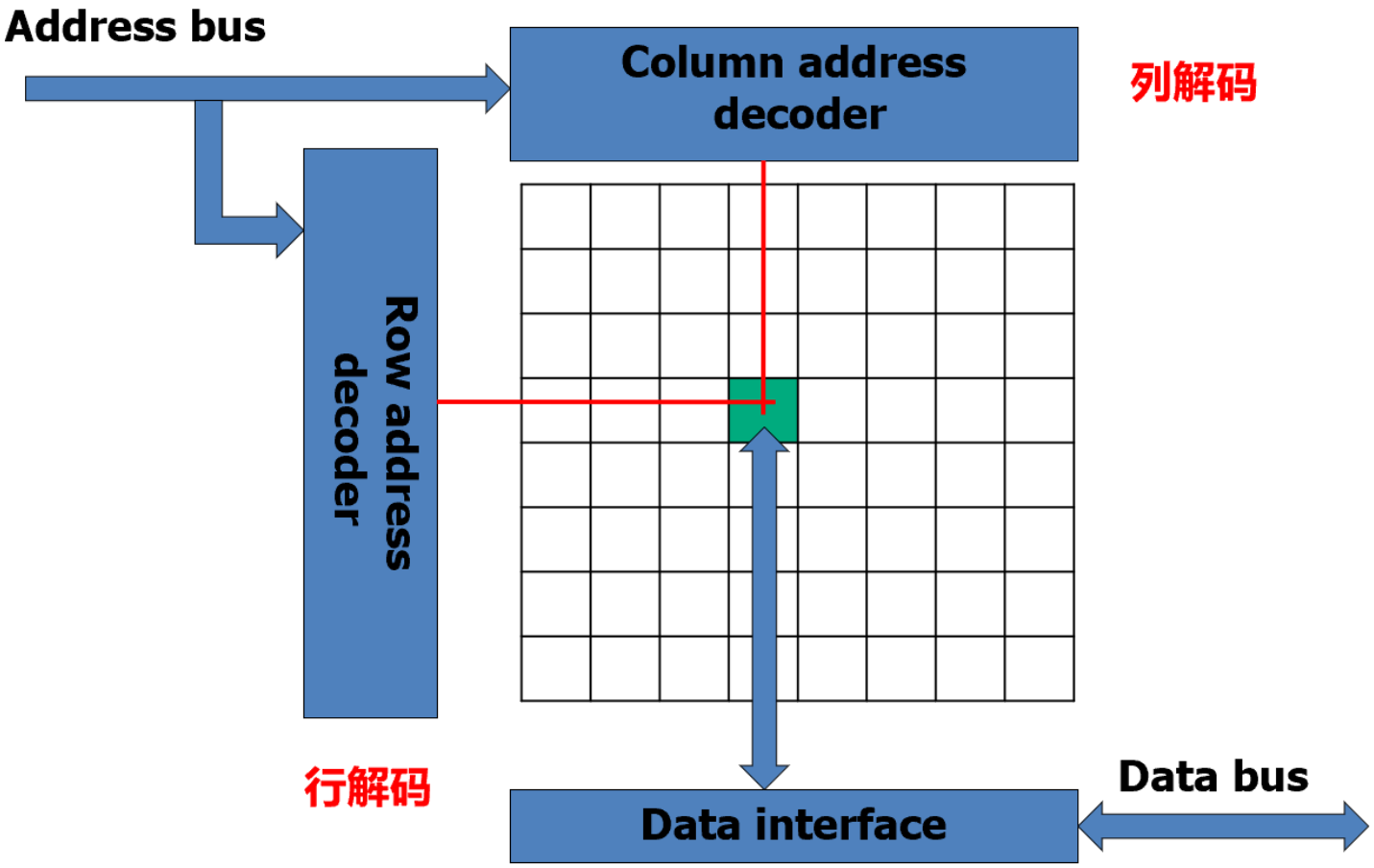
重点知识点

- 内存访问—按行访问



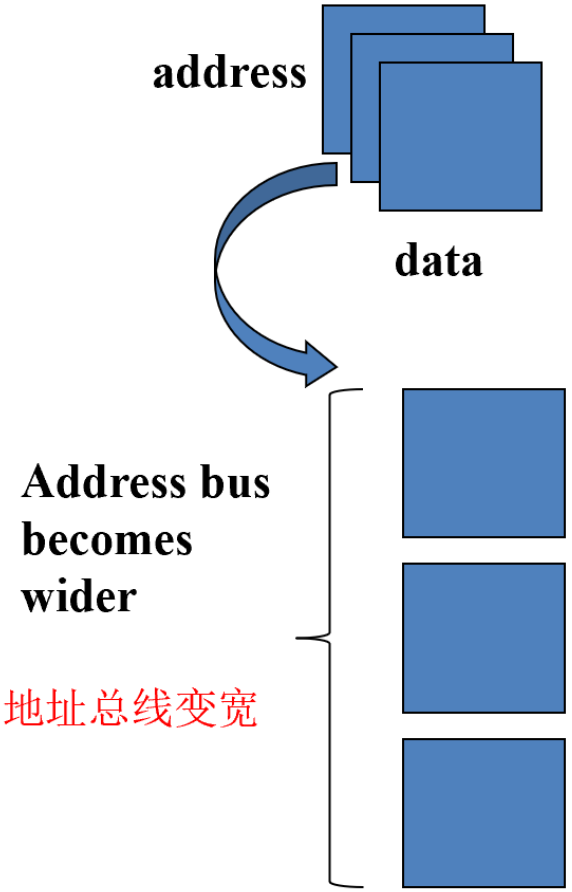
重点知识点

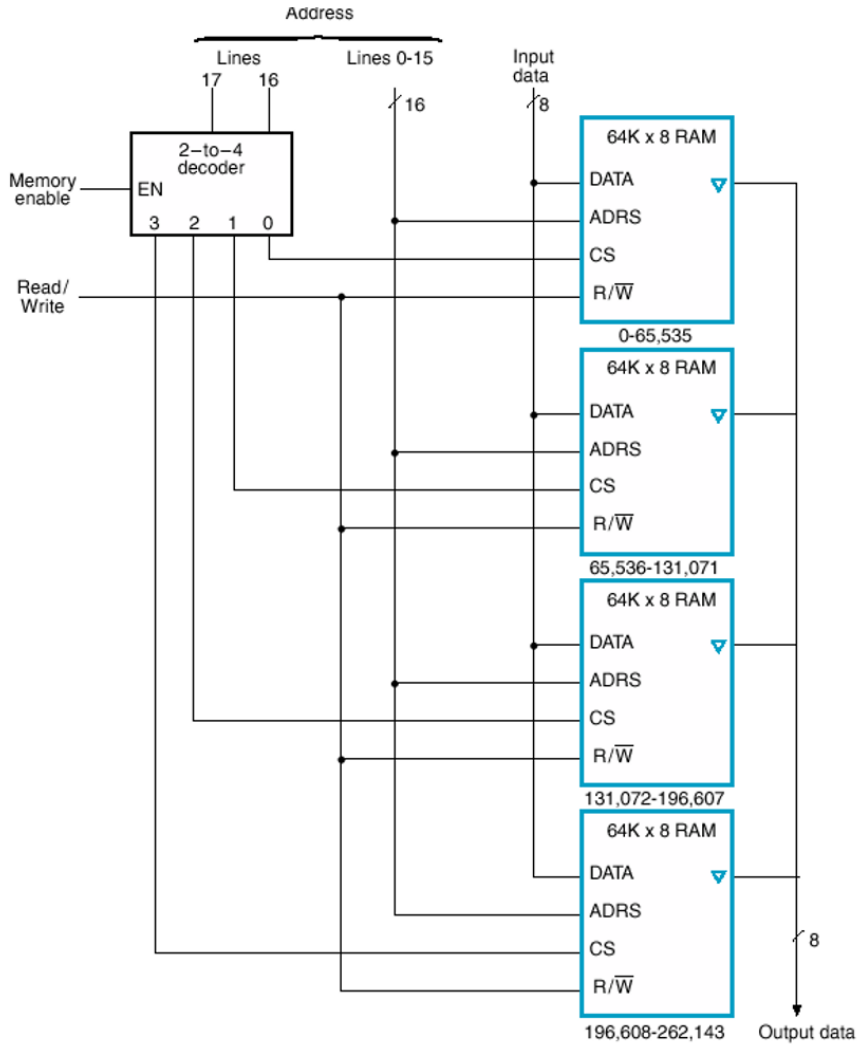
- 内存访问—网格访问



重点知识点

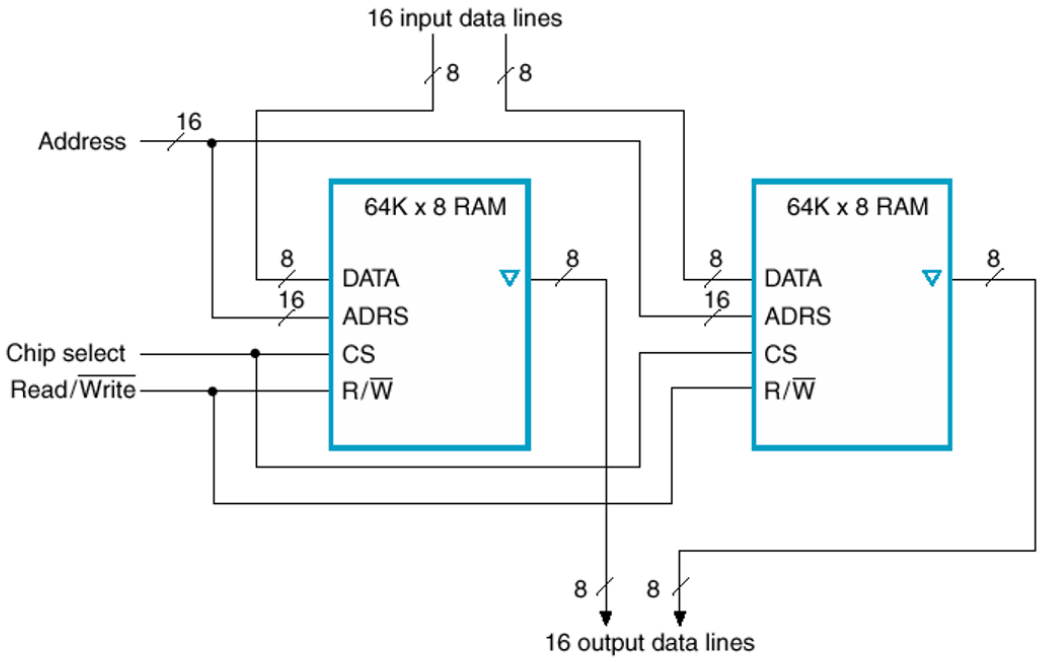
- 扩展内存—地址空间加大

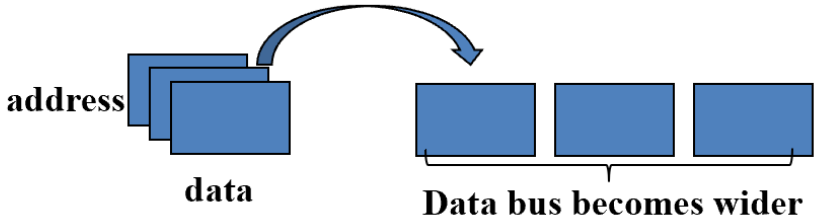




重点知识点

- 扩展内存—内存变宽

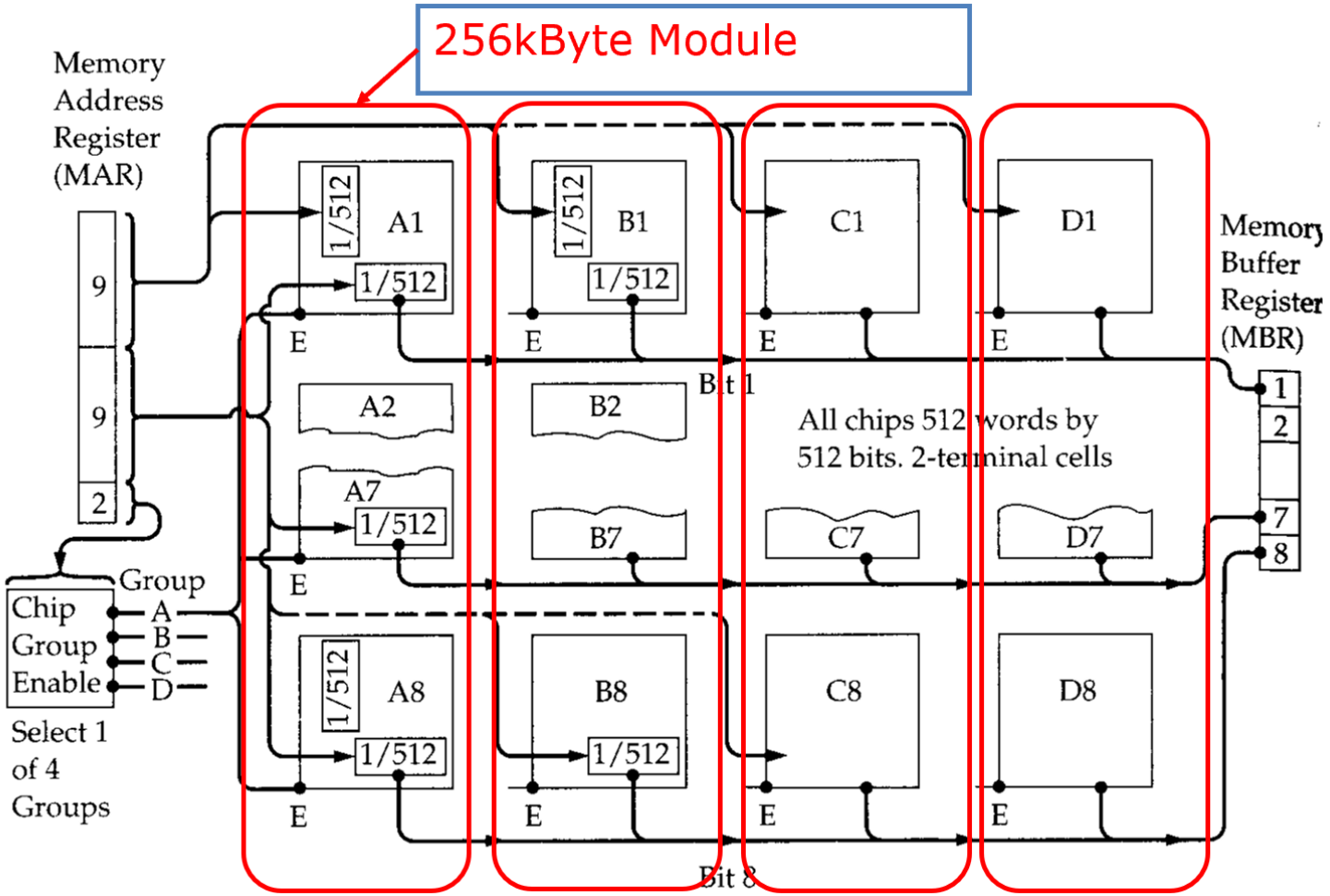




数据总线变宽

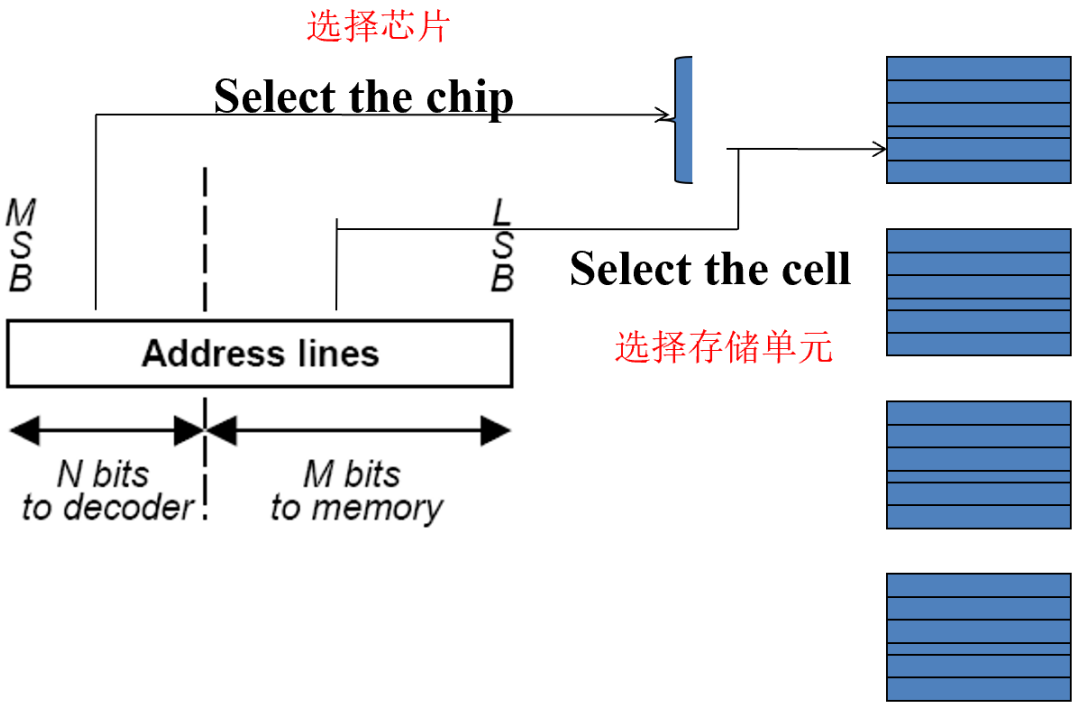
重点知识点

- 扩展内存—混合扩展



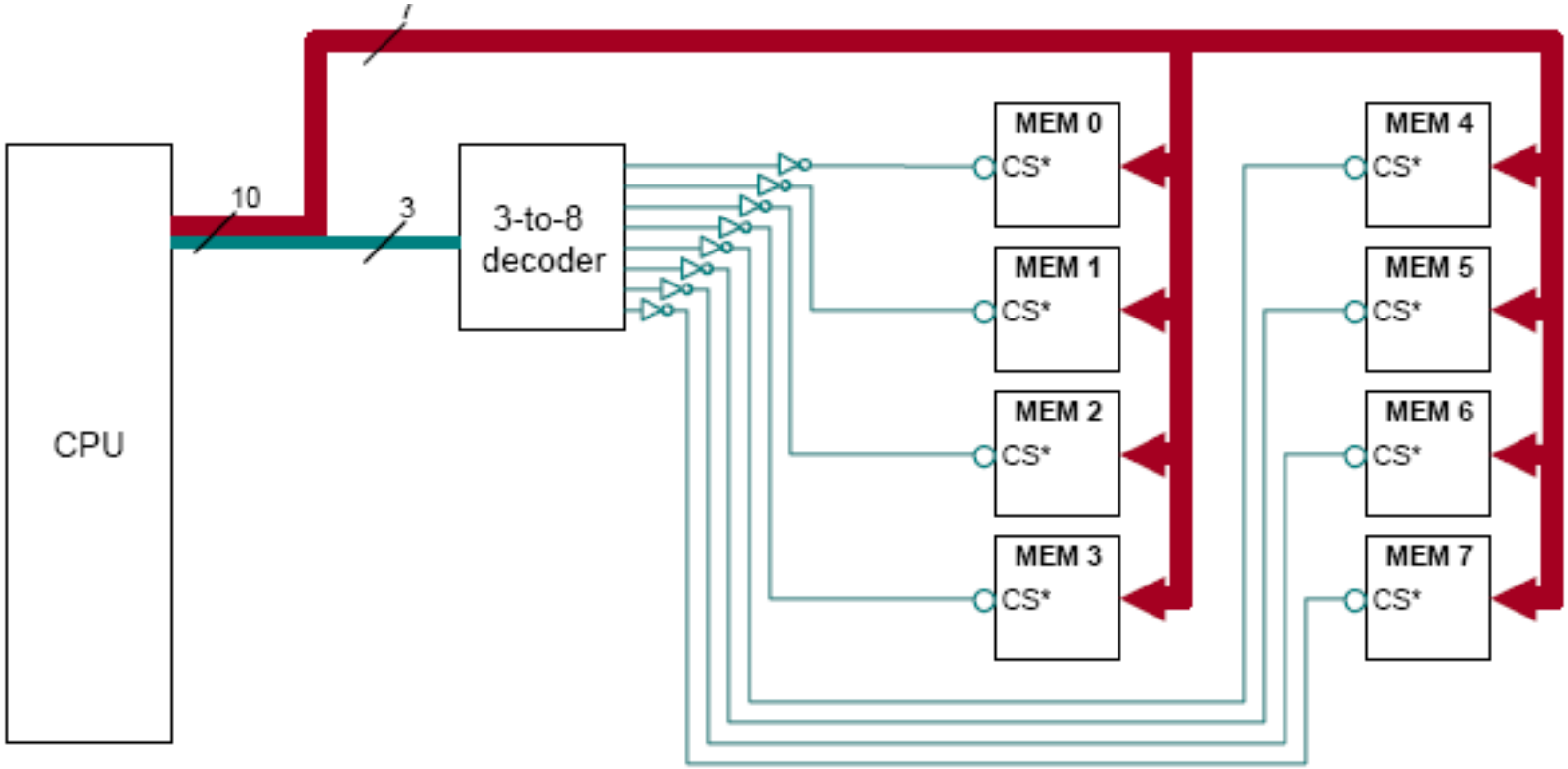
重点知识点

- 内存地址解码
  - 存储系统一般都会包含多个存储芯片
  - 地址解码就是为每个芯片生成片选信号的过程。由解码器来完成。



重点知识点

- 内存地址解码

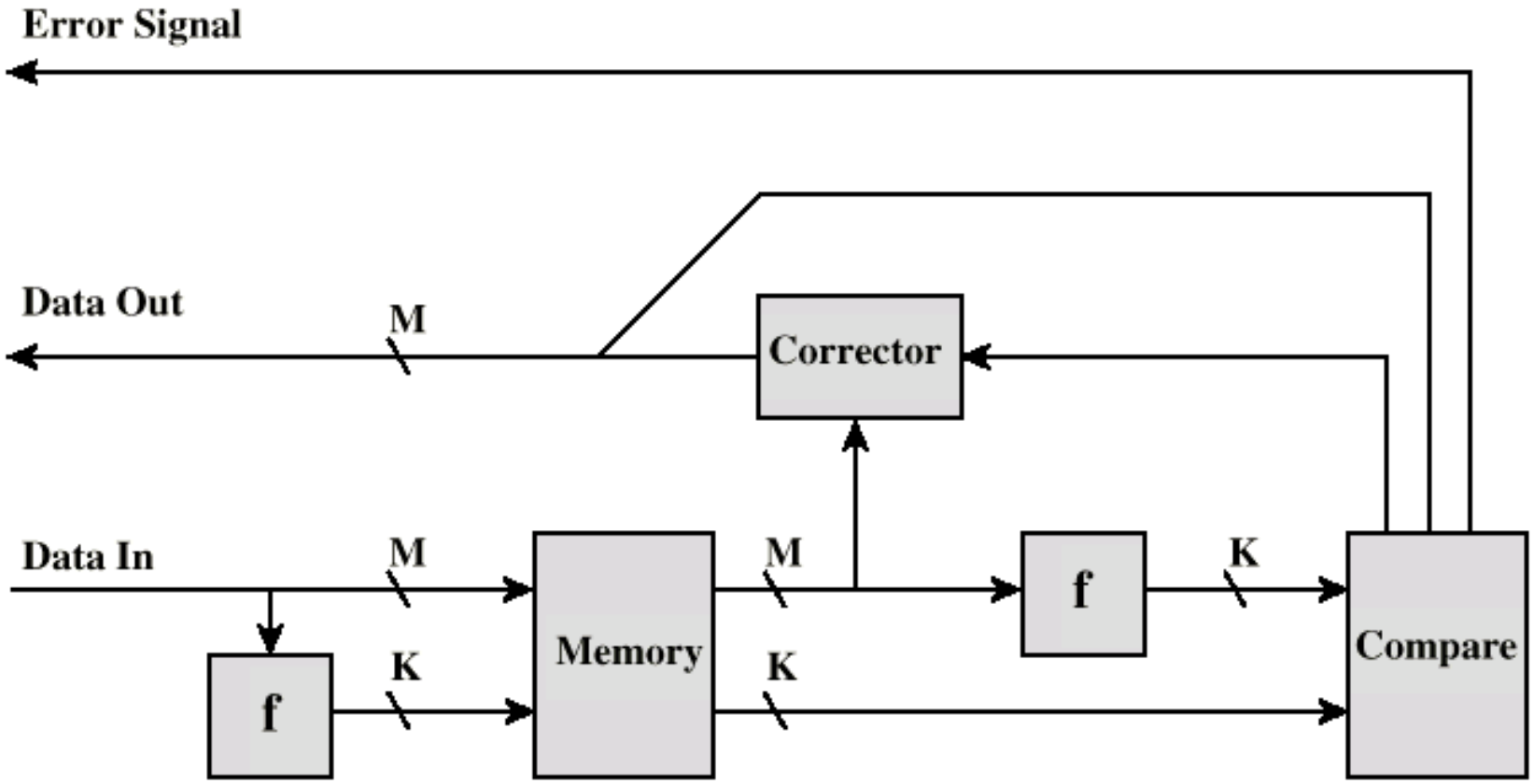


重点知识点

- 内存刷新的计算
  - 一般内存会组织成N*N*m bit的形式，其中m为一次读写的位数，比如256MB的内存，组织成16k*16k*8bit的形式
  - 按行刷新，刷新总时间=行数*单行的刷新时间
  - 刷新会导致内存带宽的损失
  - 内存行数越多，刷新带来的内存带宽的损失越大

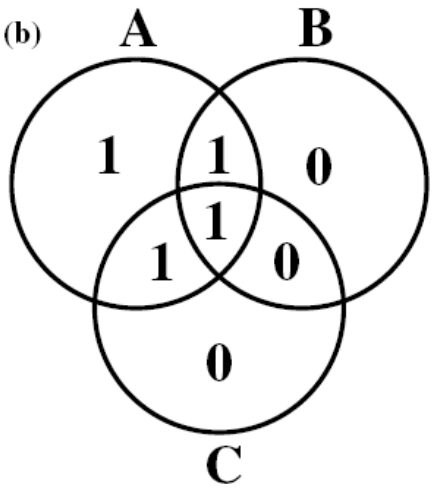
重点知识点

- 检错过程


Error Signal
Data Out
M
Corrector
Data In
M
M
f
K
Compare
f
K
Memory
K

重点知识点

- 汉明纠错码的原理
  - 4个数据位，3个纠错位
  - 数据位填到相交的位置
  - 不相交的部分填入数字，使得每个圆中的1的个数为偶数
  - 可以发现并纠正1个错误
  - $2^K - 1 >= M + K$



重点知识点

- 汉明纠错码生成过程

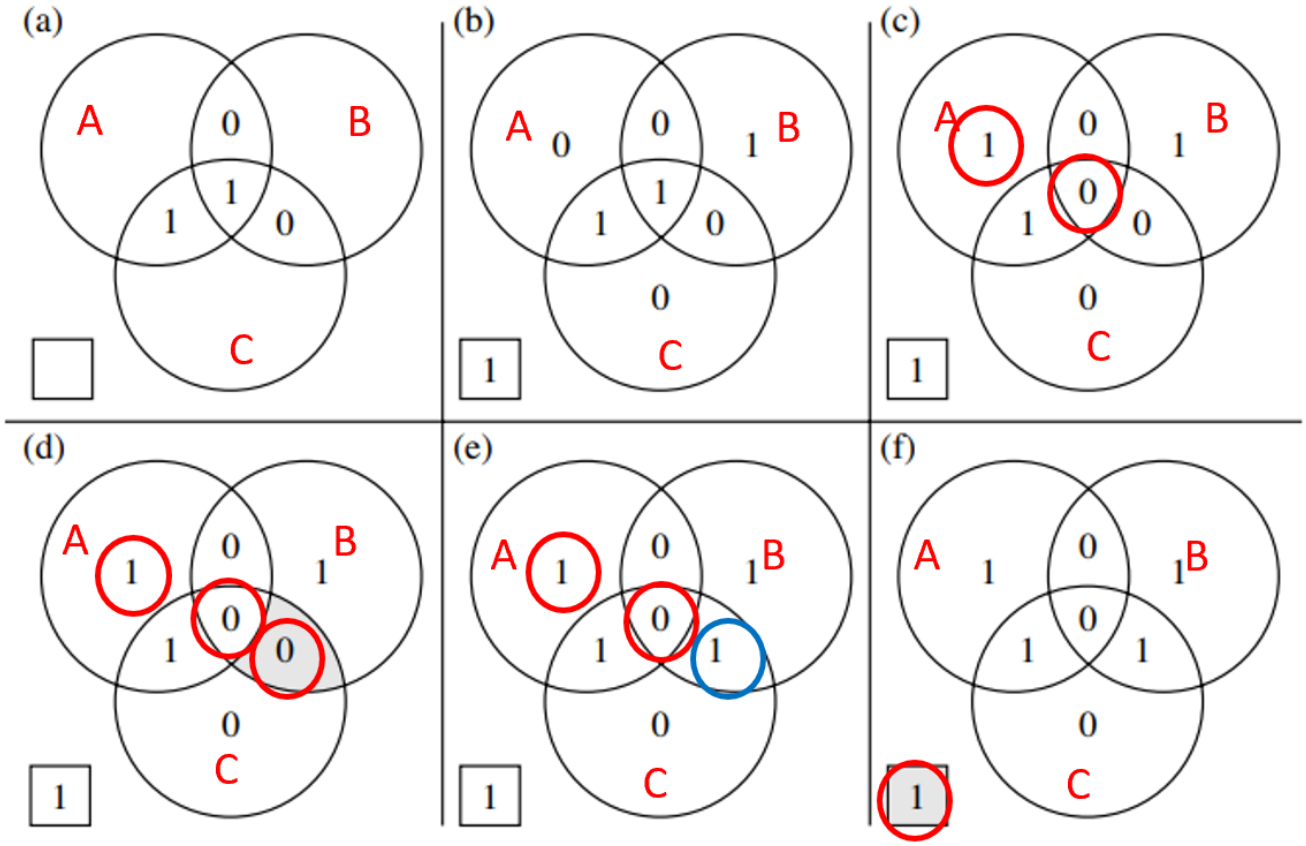
Bit position	12	11	10	9	8	7	6	5	4	3	2	1
Position number	1100	1011	1010	1001	1000	0111	0110	0101	0100	0011	0010	0001
Data bit	D8	D7	D6	D5		D4	D3	D2		D1		
Check bit					C8				C4		C2	C1

Figure 5.9 Layout of Data Bits and Check Bits

$$
\begin{aligned}
C1 &= D1 \oplus D2 \oplus D4 \oplus D5 \oplus D7 \\
C2 &= D1 \oplus D3 \oplus D4 \oplus D6 \oplus D7 \\
C4 &= D2 \oplus D3 \oplus D4 \oplus D8 \\
C8 &= D5 \oplus D6 \oplus D7 \oplus D8
\end{aligned}
$$

重点知识点

- 汉明单纠错双检错



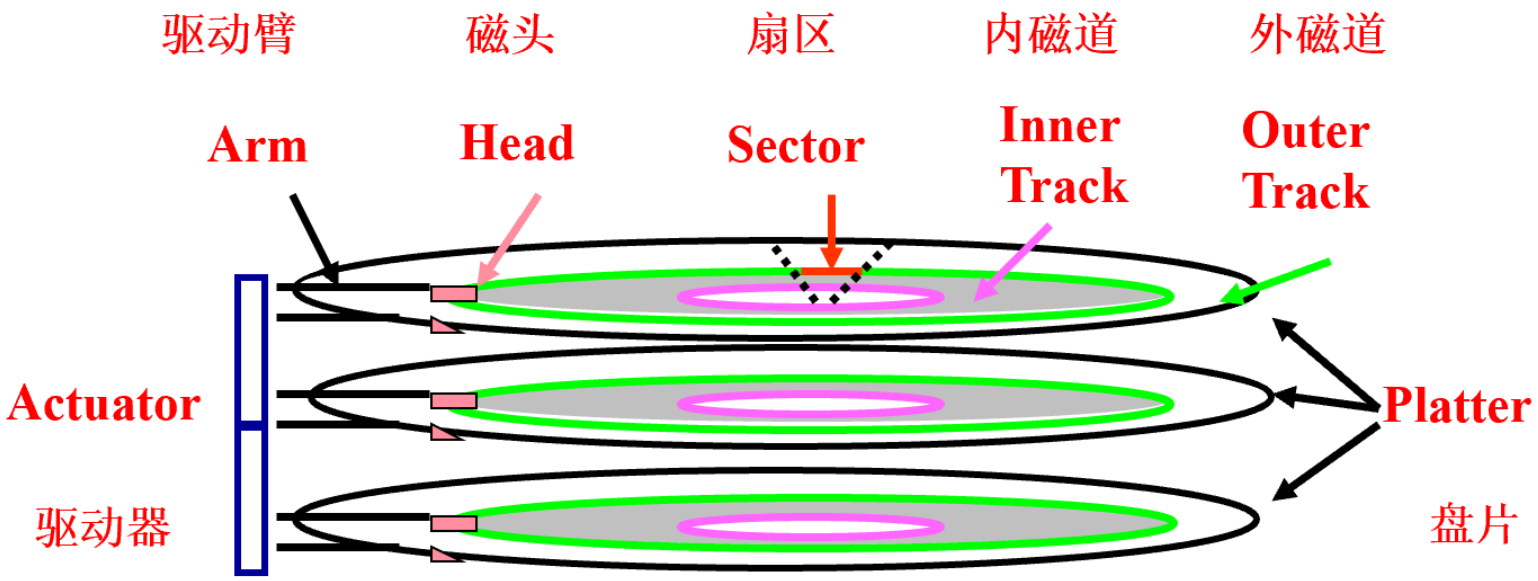
- 基本概念
  - 磁盘中的各个术语，磁盘，磁头，磁道，扇区，柱面
  - 寻道时间
  - 旋转延时
  - 传送时间
  - RAID0~RAID6
  - 光盘
  - 磁带

第六章

- 重点知识点
  - 磁盘容量的计算
  - 磁盘读写速度的计算
  - RAID中各种RAID的物理容量和可用容量的计算
  - 光盘的容量计算，包括CDROM、DVD、HD-DVD

基本概念

- 磁盘中的各个术语，磁盘，磁头，磁道，扇区，柱面

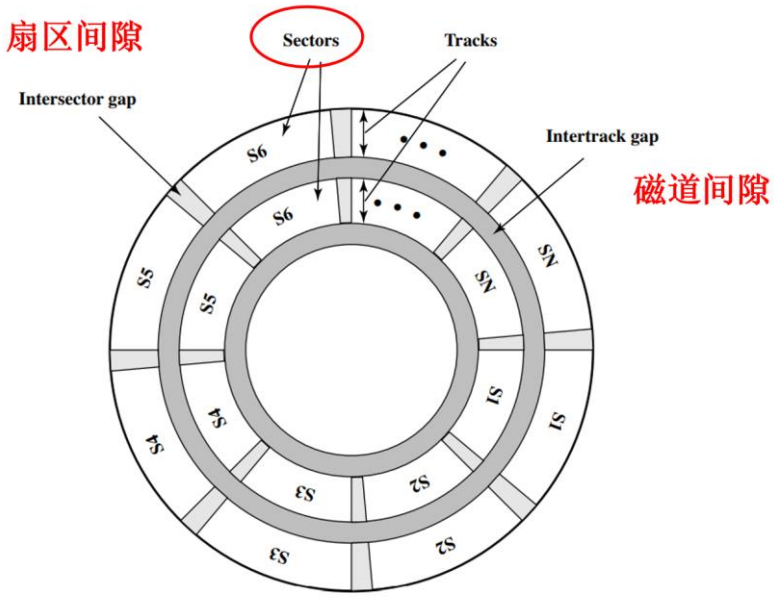


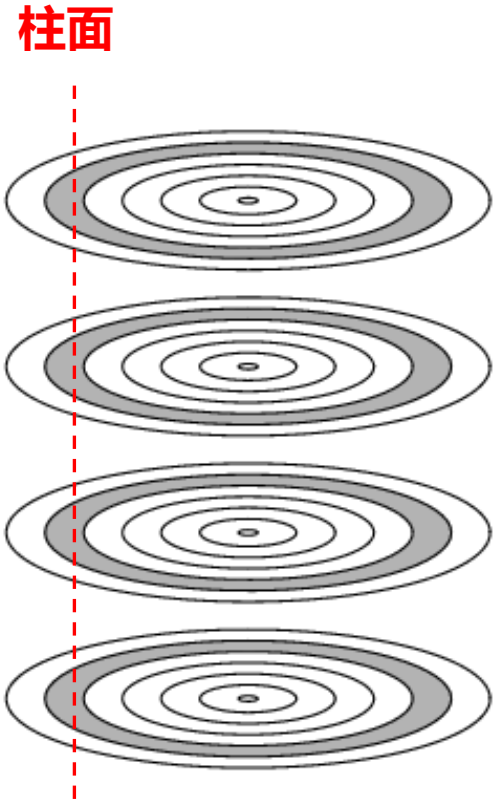
基本概念

- 磁道，扇区，柱面
  - 磁道：磁盘划分为若干个同心的环，这样的圆环称为磁道
  - 扇区：每个磁道划分为若干个扇区。每个磁道一般包含几百个扇区。扇区是数据存储的最小单位，通用的扇区大小是512个字
  - 柱面：每个盘面上相对应的位置的一组磁道设置为一个柱面，数据按照柱面进行存储。可以减少磁头的移动，提高数据传输的速度。

基本概念

- 磁道，扇区，柱面


扇区间隙
Sectors
Tracks
Intersector gap
Intertrack gap
磁道间隙
S6
S5
S4
S3
S2
S1
N5


柱面

基本概念

- 磁盘访问的时间
  - 寻道时间seek time：磁头移动到数据所在磁道需要的时间。由磁头移动时间决定。一般是10ms。
  - 旋转延时rotational delay：磁头到达磁道后，到数据所在扇区从当前位置移动到磁头下面所需要的时间。由磁盘转速决定。
  - 传送时间transfer time，指的是数据定位后开始进行实际传送的时间
  - 存取时间Access time =寻道时间 Seek +旋转延迟 rotational delay

基本概念

- RAID
  - Redundant Array of Independent Disks　冗余独立磁盘阵列
  - 常用的有7个级别
  - 一组物理磁盘，由操作系统驱动形成单一的逻辑硬盘
  - 数据在不同的物理盘上分布
  - 使用冗余容量来保存校验信息
  - 作用：增加容量，提高可靠性，提高读写性能

基本概念

- 光盘
  - Compact Disk——CD：压缩光盘
  - 数据就是用这个膜的表面上的微小凹坑来表示
  - 采用的是一条螺旋线，从中心开始往外旋转，一直到最外边。最内和最外的扇区的长度就是一样的
  - 盘片旋转的线速度恒定
  - CD，CD-R，CD-RW，DVD，DVD-W，HD-DVD，Blue-ray
- 磁带

基本概念

- 磁带
  - 顺序访问
  - 速度慢
  - 价格便宜
  - 主要用于备份和恢复

重点知识点

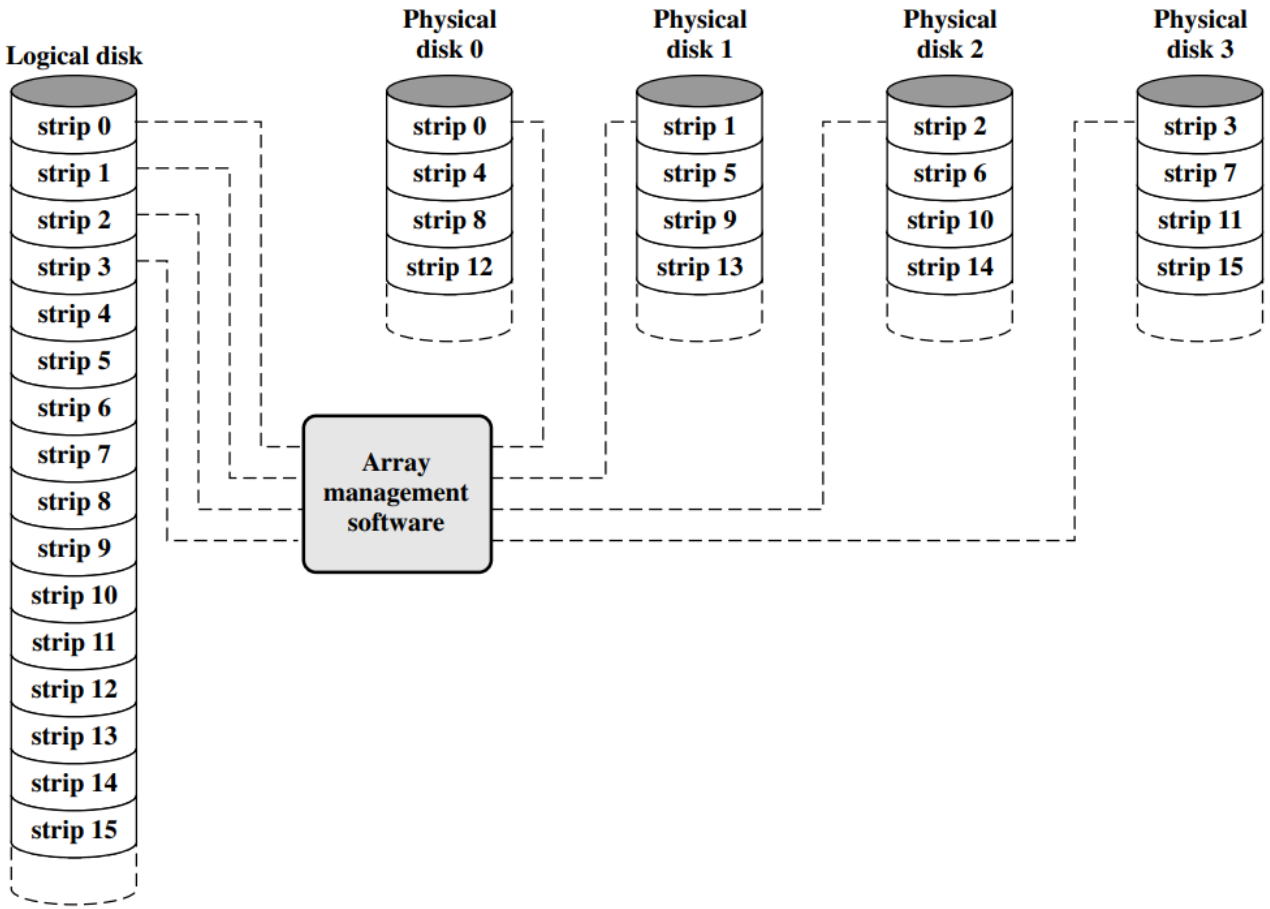
- 磁盘容量的计算
  - 磁盘以相同的角速度旋转
  - 每个磁道上的扇区数相同
  - 内圈的存储密度决定了磁盘的容量
  - 磁盘容量=每扇区字节数*磁道的扇区数*磁道数*盘面数

重点知识点

- 磁盘读写速度的计算
  - Transfer time - T 传送时间T
    - T = b/rN
    - b = number of bytes to be transferred 需要传送的字节数
    - N = number of bytes on a track 一个磁道上的字节总数
    - r = rotation speed, in revolutions per second 旋转速度
  - The total average access time (including transfer time) 总的平均访问时间，包括传输时间
    - Ta = Ts + 1/2r + b/rN 总的时间=寻道时间+平均旋转延时+传送时间
    - Ts = seek time 寻道时间
    - 1/2r is the average Rotational latency 平均旋转延时

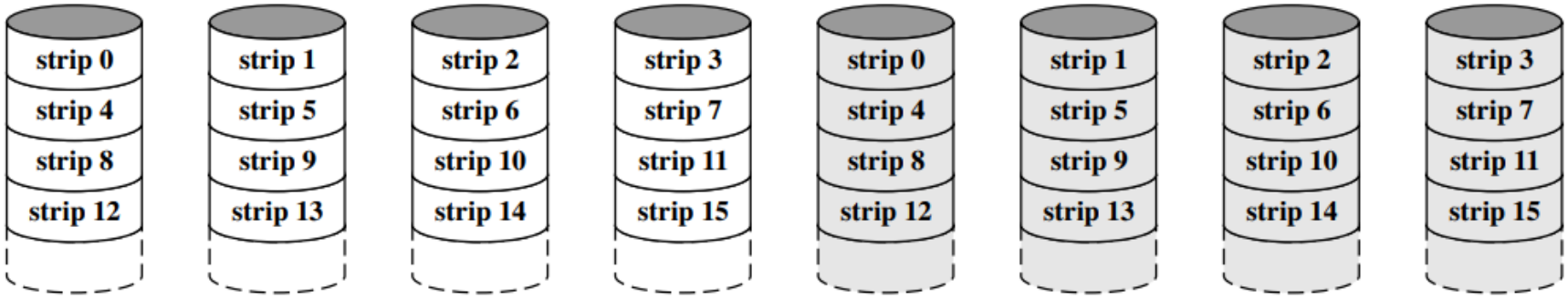
- RAID0的特点
  - 没有冗余
  - 数据条带跨磁盘存储
  - 能够提升访问速度
  - 设计简单，容易实现
  - 没有容错机制
  - 可用空间为所有磁盘空间之和
  - 适用于成本比可靠性的要求更高的场景，比如视频生产和编辑

- RAID0的映射图


Logical disk
strip 0
strip 1
strip 2
strip 3
strip 4
strip 5
strip 6
strip 7
strip 8
strip 9
strip 10
strip 11
strip 12
strip 13
strip 14
strip 15
Array management software
Physical disk 0
strip 0
strip 4
strip 8
strip 12
Physical disk 1
strip 1
strip 5
strip 9
strip 13
Physical disk 2
strip 2
strip 6
strip 10
strip 14
Physical disk 3
strip 3
strip 7
strip 11
strip 15

- RAID1的特点
  - 全镜像
  - 单个磁盘读，两个磁盘写
  - 故障恢复简单
  - 价格昂贵
  - 可用空间为全部空间的1/2
  - 适用于可靠性要求高的场合

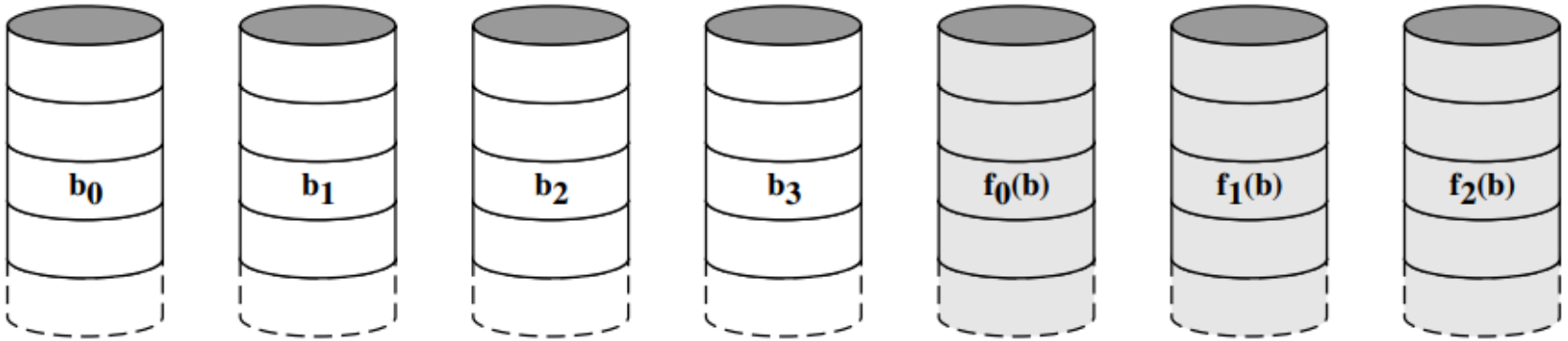
- RAID1的映射图



(b) RAID 1 (Mirrored)

- RAID2的特点
  - 磁盘分为数据盘和纠错盘
  - 采用汉明纠错码生成纠错码，保存在纠错盘中
  - 冗余度很高，价格昂贵
  - 读写比较慢，需要全部磁盘参与计算
  - 根据数据盘的数量决定冗余盘的数量。例如，4个数据盘，需要3个纠错盘；8个数据盘，需要4个纠错盘
  - 使用的不多

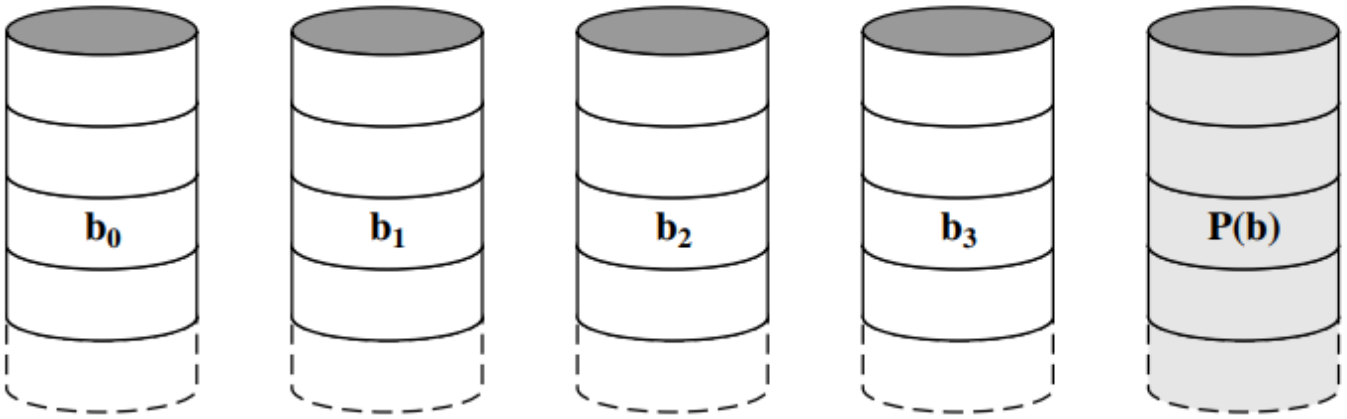
- RAID2的映射图



(c) RAID 2 (Redundancy through Hamming code)

- RAID3的特点
  - 只设置了一个冗余盘
  - 奇偶校验
  - 损坏一个盘，可以通过其他盘重构数据
  - 写比较慢，需要进行奇偶校验位计算
  - 读写都是全部盘参与
  - 可用空间为（N-1）/N

- RAID3的映射图



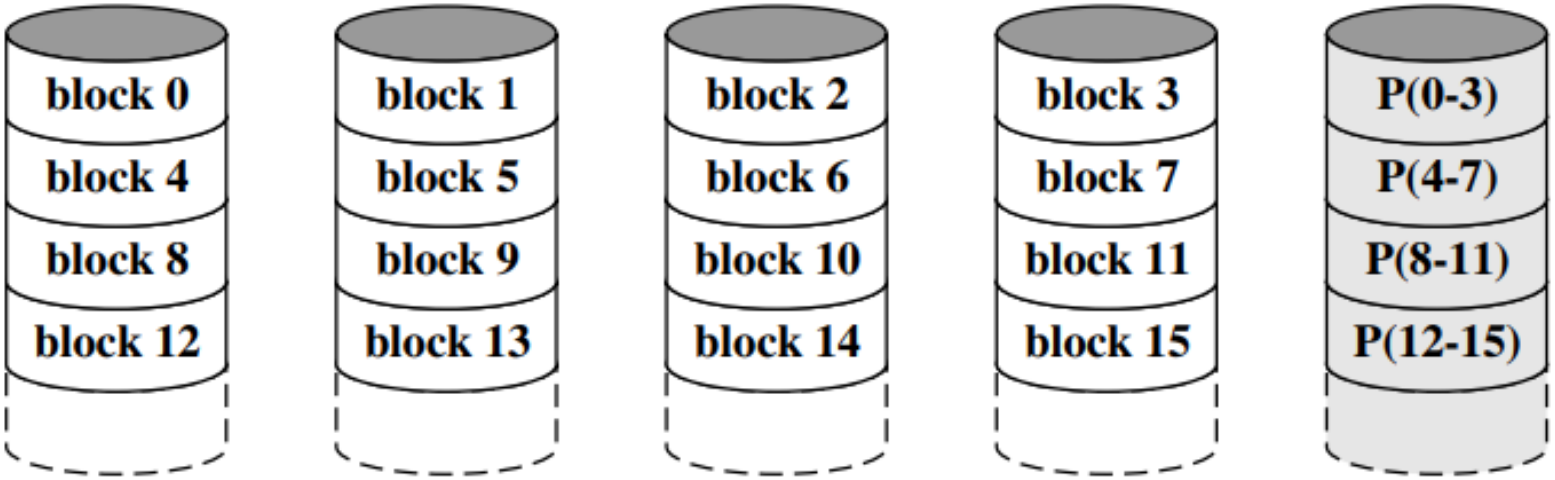
(d) RAID 3 (Bit-interleaved parity)

位交错奇偶校验

- Reconstruct is easy: 重构容易
  - X4(i) = X3(i)⊕ X2(i)⊕ X1(i)⊕ X0(i)
  - X1(i) = X4(i)⊕ X3(i)⊕ X2(i)⊕ X0(i)

- RAID4的特点
  - 只设置了一个校验盘
  - 各个数据盘独立
  - 数据盘采用大条带
  - 数据盘写入数据时，同时根据奇偶校验，将校验位写到校验盘
  - 读数据可以并行进行，写入需要计算校验，比较慢
  - 校验盘的压力比较大
  - 可用空间为（N-1）/N

- RAID4的映射图

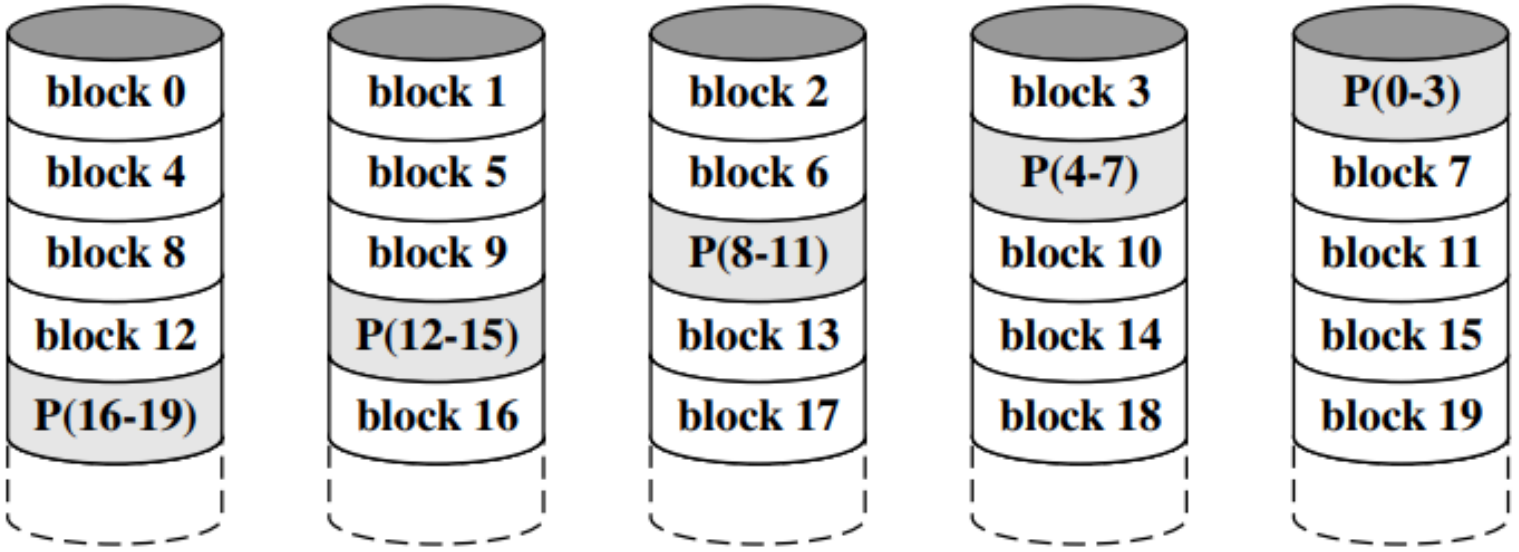


(e) RAID 4 (Block-level parity)

块级别校验

- RAID5的特点
  - 只设置了一个校验盘，和RAID4类似
  - 校验条带轮流分布在各个盘上
  - 避免RAID4的校验盘的瓶颈
  - 可用空间为（N-1）/N
  - 通常用在网络服务器上

- RAID5的映射图

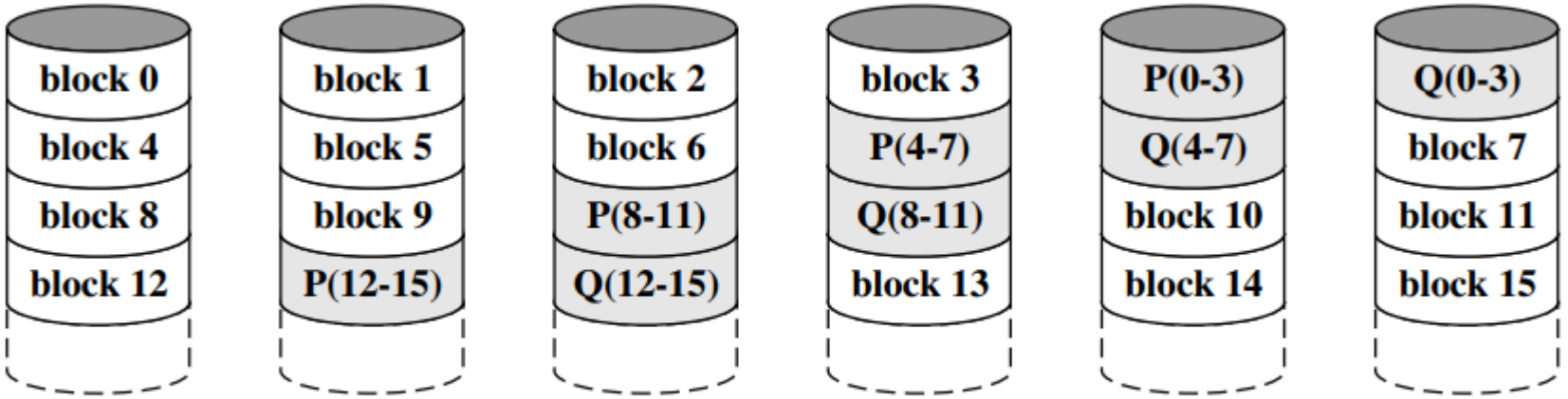


(f) RAID 5 (Block-level distributed parity)

块级别分布式校验

- RAID6的特点
  - 双校验盘
  - 不同的算法，生成2个奇偶校验码，放在不同的校验条带上
  - 校验条带轮流分布在各个盘上
  - 可靠性高，写速度比较慢
  - 可用空间为（N-2）/N
  - 通常用在可靠性要求较高的场合，比如数据库服务器

- RAID6的映射图



(g) RAID 6 (Dual redundancy)

双冗余

- 典型光盘的容量
  - CD：650MB，大概70分钟的音频
  - DVD：4.7GB
  - 双面双层DVD：17GB
  - HD-DVD：单面单层15GB
  - Blue-ray：单层 25GB

第七章

- 基本概念
  - I/O模块
  - 编程式I/O
  - 中断式I/O
  - DMA
  - 周期窃取
  - I/O通道和处理器

第七章

- 重点知识点
  - I/O的功能和框图
  - 编程式I/O的工作流程
  - 中断式I/O的工作原理
  - DMA的工作原理
  - I/O的速度计算

基本概念

- I/O模块
  - I/O模块不仅实现了外设和系统总线的互连，同时还需要完成外设和系统总线之间的通信逻辑
  - I/O模块向上通过系统总线或中央交换器与处理器和存储器进行连接
  - I/O模块向下通过专用的数据线与一个或多个外设连接
  - I/O模块的功能：控制和定时，处理器通信，设备通信，数据缓冲，检错处理

基本概念

- 编程式I/O
  - 处理器能直接控制I/O模块，感知I/O模块的状态，并且下发读、写命令，进行数据的传输
  - CPU下发指令后，需要等待I/O模块完成操作
  - 浪费CPU时间

基本概念

- 中断式I/O
  - CPU不需要等待，也不需要不停地检查I/O设备的状态，而是可以继续处理后续的指令
  - I/O模块在准备好和处理器进行数据交换的时候，它会发起一个中断请求
  - 处理器发现有中断，就进行中断处理。不用一直等待，降低了CPU的无效等待时间，提高了处理效率

基本概念

- DMA
  - Direct Memory Access
  - 能够实现内存和 I/O直接传输数据的模块
  - DMA代替CPU完成内存和 I/O之间的传输
  - 减少了CPU的参与度，进一步提高了CPU的效率

基本概念

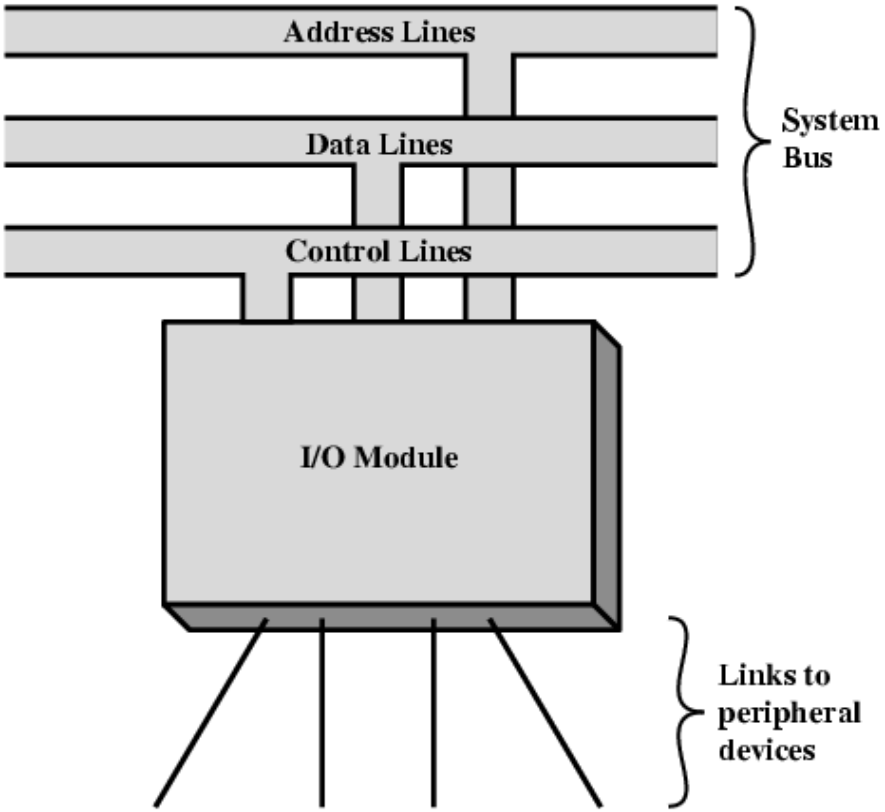
- 周期窃取
  - DMA需要使用系统总线来管理数据的传输
  - 或者是处理器不需要系统总线的时候工作，或者强制处理器临时挂起操作，以便DMA模块使用总线
  - 强制处理器临时挂起的方式称为“周期窃取”，相当于说DMA模块窃取了一个总线周期
  - 和中断不一样，不需要保存上下文信息，CPU仅仅是挂起一个时钟周期

基本概念

- I/O通道和I/O处理器
  - I/O模块经过增强后，成为一个有自主控制权的独立处理器，具有定制的指令集。CPU指示I/O处理器在内存中执行I/O程序，无需CPU干预。
  - I/O处理器负责处理大部分任务，包括控制终端
  - 不对I/O通道和I/O处理器做区分，统称为I/O通道

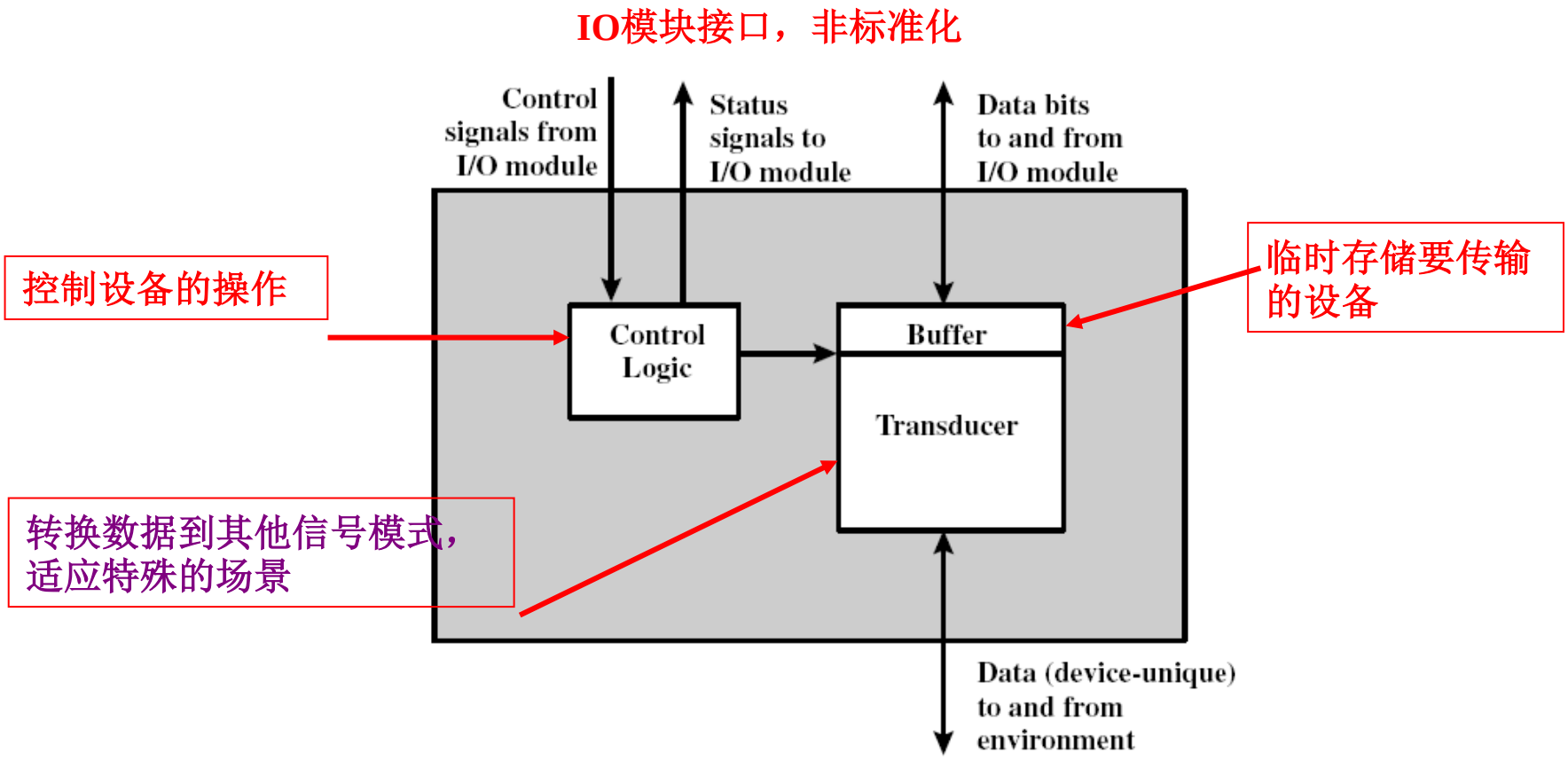
重点知识点

- I/O模块通用模型



重点知识点

- 外设结构框图

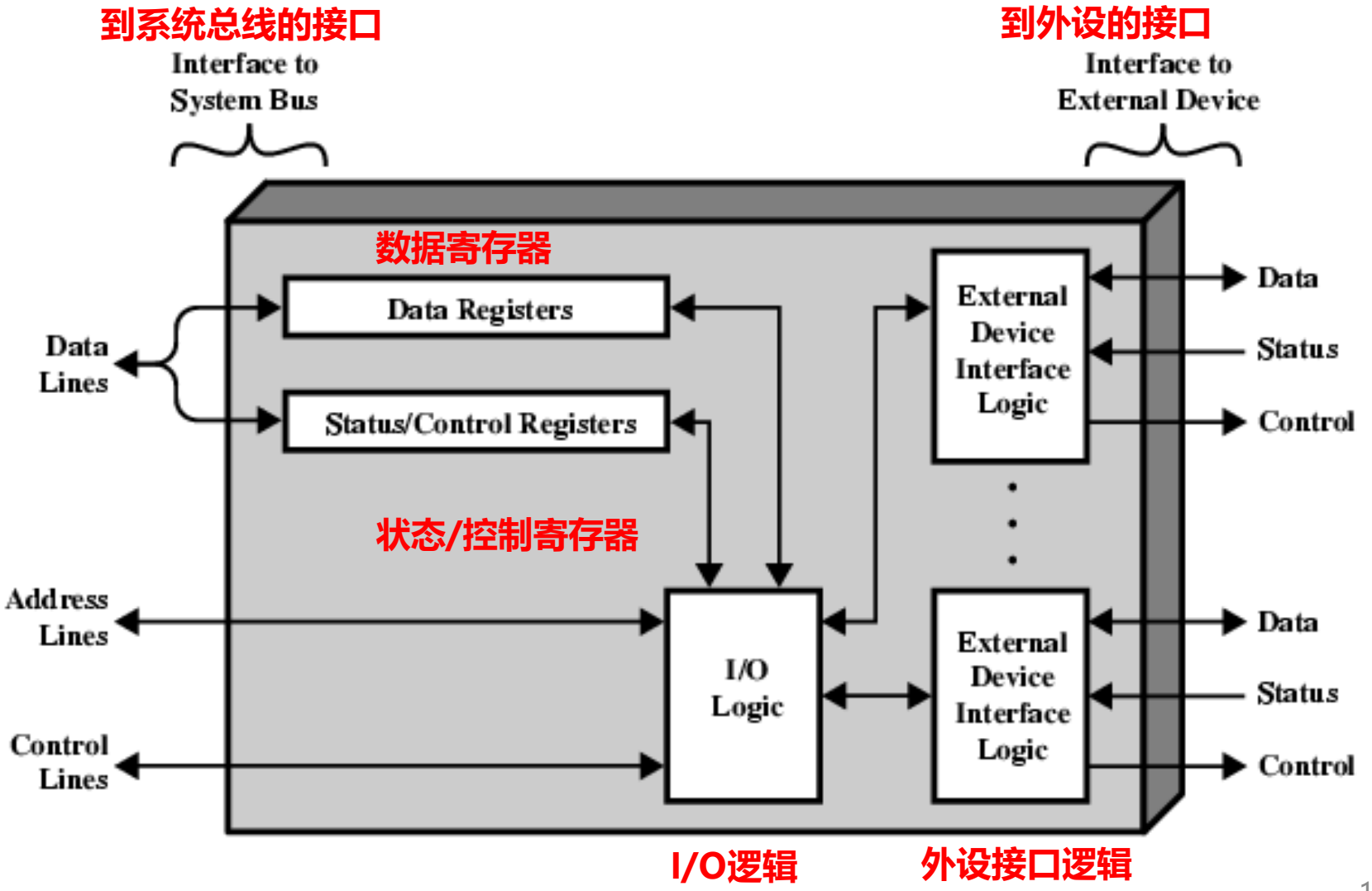


重点知识点

- 设备和CPU之间的数据传输过程
  - CPU检测IO模块的状态
  - IO模块返回状态
  - 如果就绪，CPU请求数据
  - IO模块从设备中获取数据
  - IO模块传输数据给CPU

重点知识点

- I/O模块框图



重点知识点

- I/O模块和处理器的通信功能
  - 命令解码
  - 地址识别
  - 传输传送
  - 状态报告

重点知识点

- 编程式I/O模块工作流程
  - CPU需要执行I/O操作的时候，CPU发送指令，请求I/O操作
  - I/O模块执行对应的操作，执行完成后，设置状态位
  - 在待期间，CPU会周期性地检查状态，看I/O模块是不是执行完了
  - I/O在执行完成之后，不会通知CPU它的完成状态，也不会发送中断信息给CPU
  - 在这个期间，CPU只能等待，或者过一会儿再来检查状态位

重点知识点

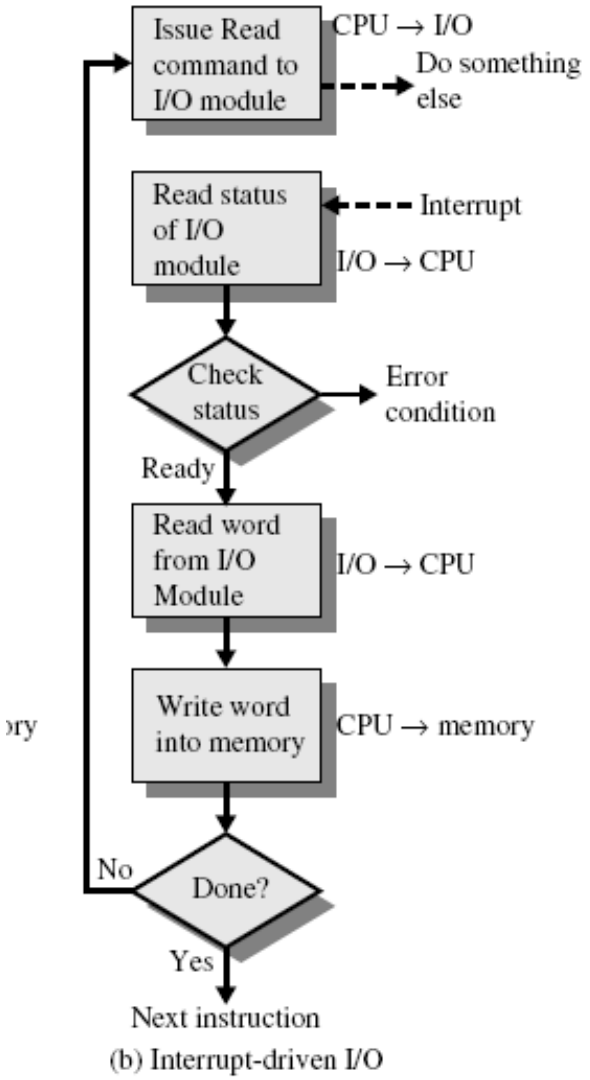
- I/O编址方式
  - 内存映射I/O编址
    - 地址空间分为2个部分：一部分给内存，一部分给I/O
    - 能够使用相同的指令去传输数据
    - 浪费内存空间
  - I/O独立编址
    - 需要I/O或内存的选择线
    - 需要专用的I/O指令
    - 不浪费内存空间

重点知识点

- 中断式I/O模块工作流程

  - CPU发出IO指令
  - CPU继续工作
  - IO模块：接收指令，处理工作。

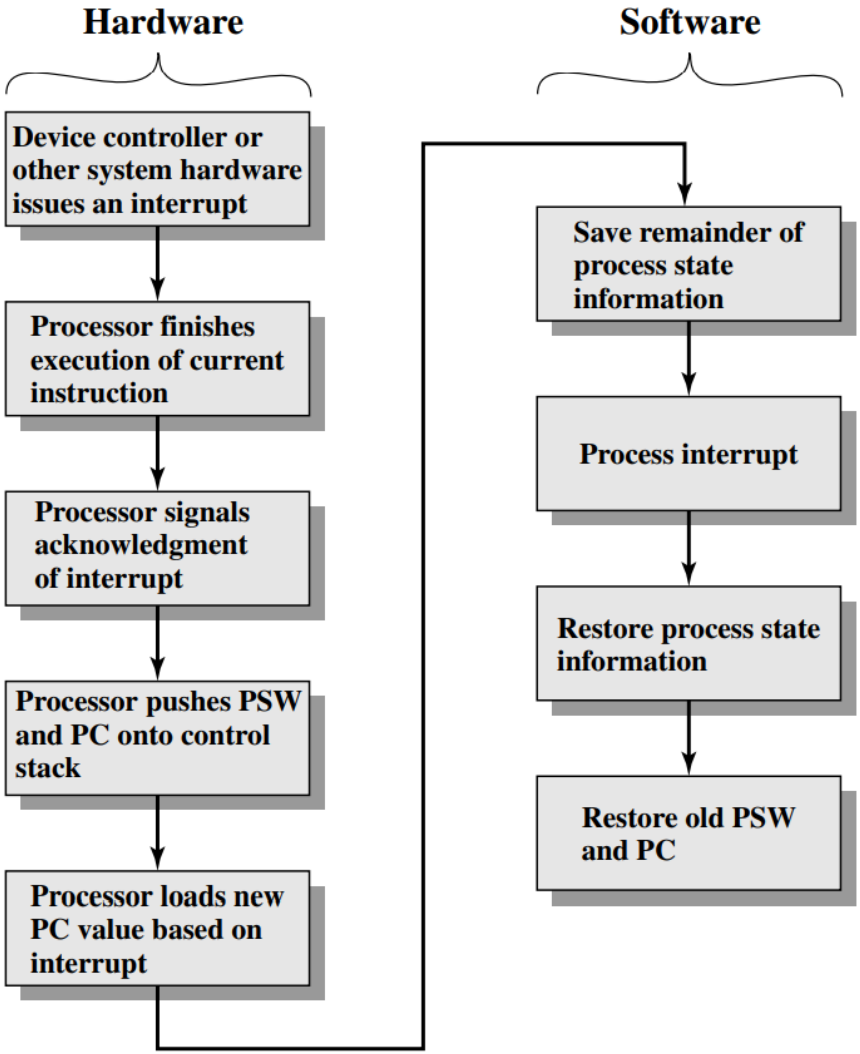
    处理完后，给CPU发一个中断



(b) Interrupt-driven I/O

重点知识点

- 中断式I/O模块工作流程



设备控制器发出中断

处理器完成当前指令的执行

处理器检测到中断

将程序状态字PSW和PC压栈

根据中断获取新的PC值

保存其他处理器状态信息

处理中断

恢复处理器状态信息

恢复PSW和PC

重点知识点

- 识别中断设备的方式
  - 独立中断线
  - 软件轮询
  - 硬件轮询
  - 总线仲裁

重点知识点

- 多重中断处理方式
  - FIFO
  - 多个中断线，每个中断线都有优先级
  - 硬件或软件轮询，轮询顺序决定了优先级
  - 总线仲裁，仲裁方式决定了优先级

重点知识点

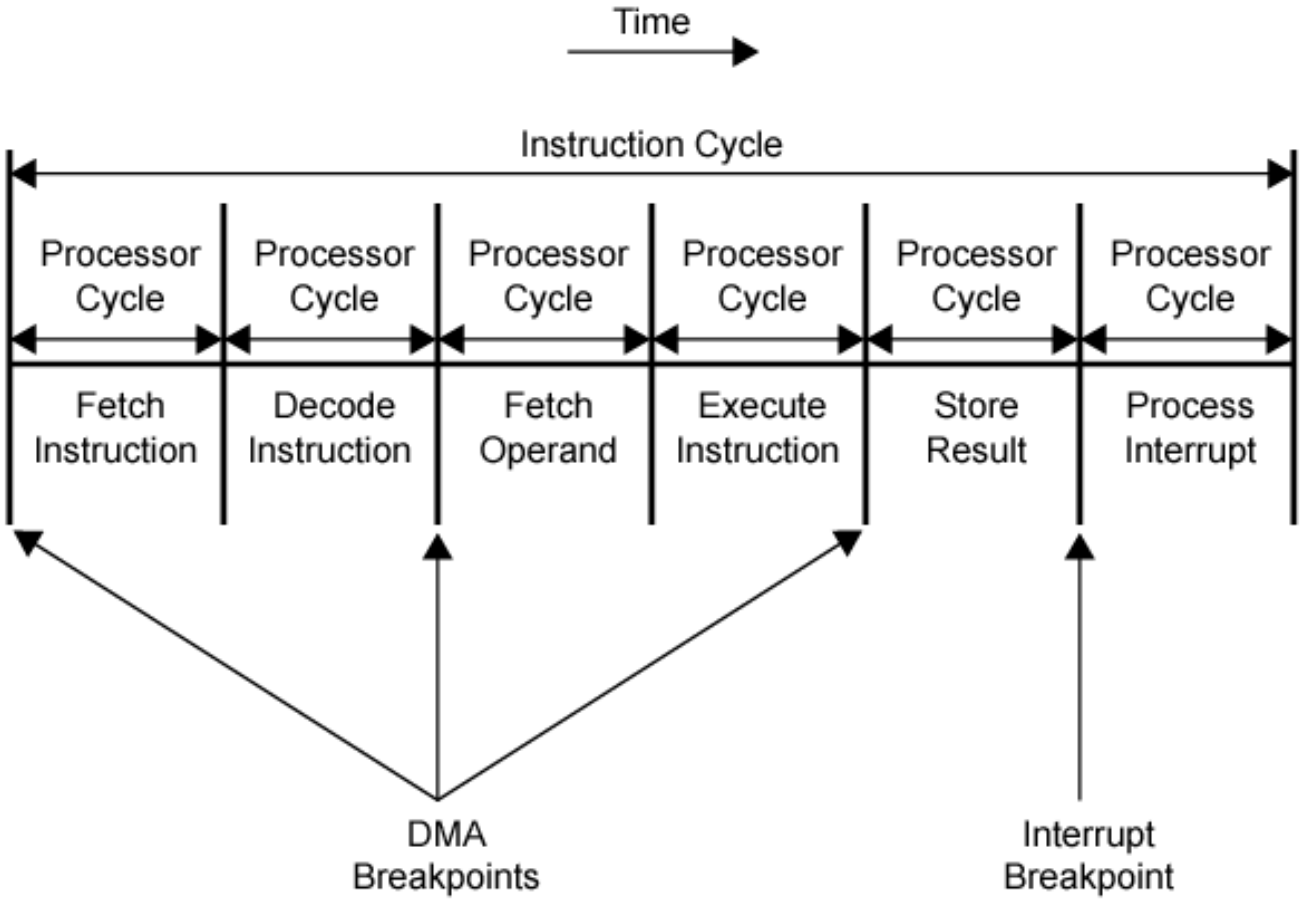
- DMA工作流程
  - CPU告诉DMA控制器：操作类型，设备地址，内存块开始地址，数据数量
  - CPU继续做其他工作
  - DMA控制器处理传输，完成后发出中断
    - I/O设备到内存，或者内存到I/O设备
    - 地址自动增加，计数器更新
  - CPU只在传输的开始和结束时介入

重点知识点

- 周期窃取工作原理
  - DMA控制器获得一个时钟周期的总线控制权，在这个时钟周期内传送一个字的数据，传送完成后，将总线控制权还给处理器。
  - 周期窃取不是一个中断。中断处理时，CPU需要保存上下文后再进行中断处理。而在周期窃取中，CPU不需要保存上下文，仅仅是挂起一个周期，然后可以继续访问总线，进行后续的操作。并且挂起只会发生在CPU需要访问总线前的这个周期。如果当前的操作不需要使用总线，那么CPU可以继续进行
  - 周期窃取会导致CPU挂起一个时钟周期，所以会使得CPU的处理减慢一点点。但是对于数据传输来说，这种方式的效率比编程式I/O或者中断式I/O都要高

重点知识点

- 周期窃取原理图



DMA断点　　**中断断点**

第八章

- 基本概念
  - 进程和进程调度
  - 操作系统
  - 虚拟地址
  - 交换
  - 分区
  - 分页
  - 分段
- 重点知识点
  - 通过虚拟地址访问存储器的过程
  - 分页模式下的页-页帧映射算法

基本概念

- 操作系统
  - 最基本的系统程序，它在用户和计算机之间搭起了一个桥梁，为用户使用计算机提供了接口，可以更方便使用计算机
  - 操作系统的功能
    - 程序创建和执行
    - I/O设备访问
    - 文件访问控制
    - 系统访问
    - 错误检测和处理
    - 统计和账务

基本概念

- 操作系统分类
  - 交互式操作系统，批处理操作系统
  - 单道程序操作系统，多道程序操作系统

基本概念

- 进程
  - 进程是正在执行的程序
  - 进程是在计算机上运行的程序的实例
  - 进程是可以分配给处理器并在处理器上执行的实体
  - 进程是一种活动单元，其特征是执行一系列指令、当前状态和一组相关的系统指令
  - 从上面的描述可以看到，进程是动态运行的，而程序是静态的。

基本概念

- 进程交换
  - 操作系统维护了一个进程的长期队列，队列中的进程放在磁盘上
  - 内存有空间了，长期队列中的一个进程就会调入到内存中。当进程执行完成后，它就从内存中调出
  - 如果内存中所有的进程都处于阻塞状态，把某个阻塞的进程“交换”到一个中间队列中，这个中间队列保存的都是临时从内存中调出的进程。然后，从这个中间队列中调入一个已经就绪的进程，或者从长期队列中调度一个新的进程到内存中。这个过程称为“交换”

基本概念

- 内存分区
  - 将内存分为几个部分，分配给包括操作系统在内的进程
  - 固定分区
    - 不一定都是相同的大小
    - 进程分配给容纳它的最小分区
    - 有内存浪费
  - 变长分区
    - 按进程要求分配
    - 会产生内存碎片
    - 合并或紧缩法处理碎片

基本概念

- 逻辑地址
  - 进程重新加载到内存中，没有机制保证会在同一个位置
  - 采用固定地址的话，那么进程被交换出去后，再交换进来，进程的内存地址发生了变化，就会出现问题
  - 采用逻辑地址和物理地址的方法。物理地址是进程在内存中的实际单元地址，而逻辑地址是相对于进程起始地址的一个相对地址
  - 当前进程的起始单元地址，称为基址，加上逻辑地址，就可以得到在内存中的物理地址

基本概念

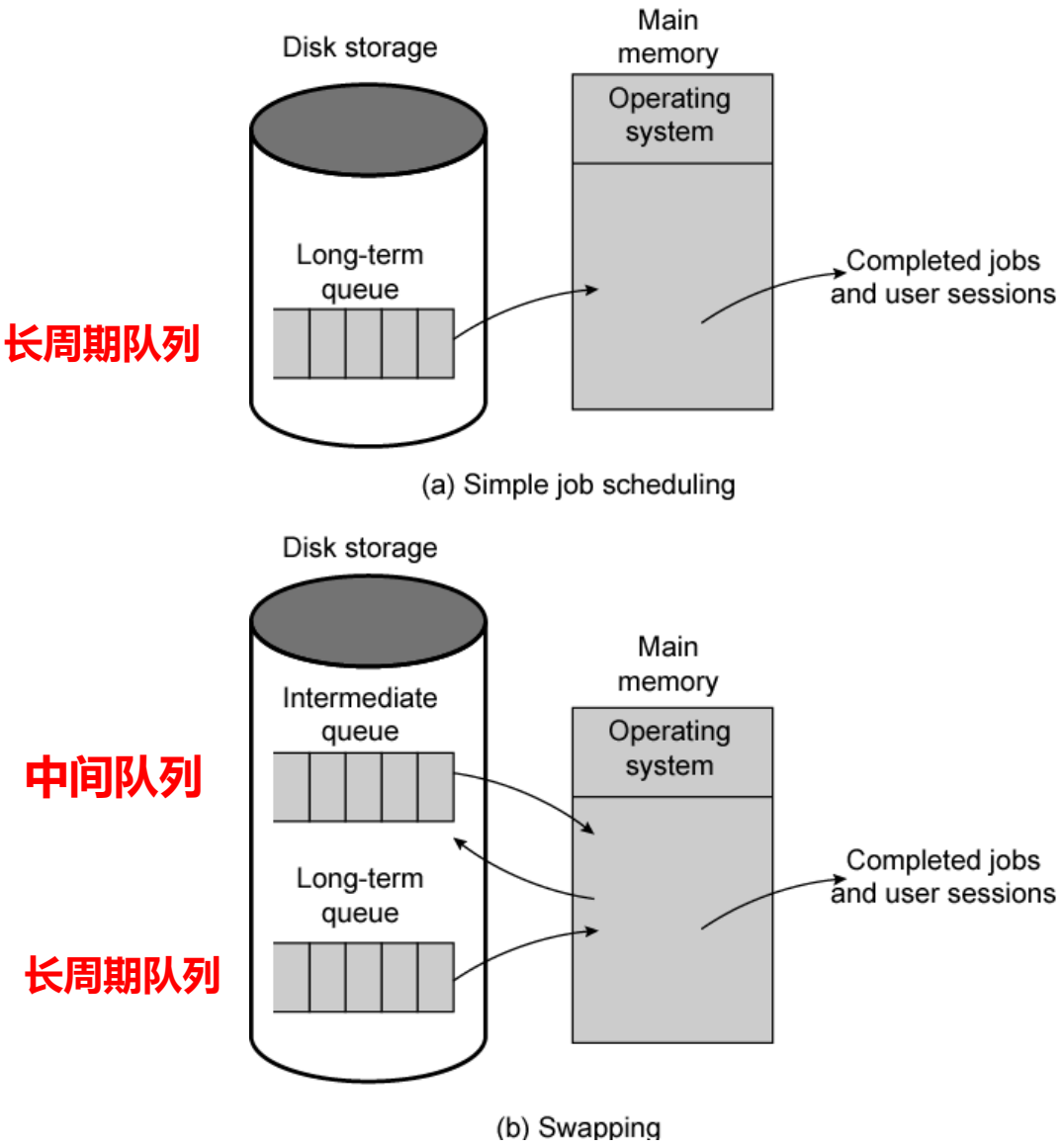
- 分页
  - 将内存分为固定大小的小存储块，这个小存储块我们称为“帧”或“页帧”
  - 进程也按照同样的大小划分为若干个固定块，称为“页”。操作系统负责维护空闲帧的列表
  - 在给进程分配内存的时候，按照帧来进行分配若干个帧给进程。进程占用的帧不一定需要是连续的帧
  - 进程中的页不需要全部全部加载到内存中，可以在需要的时候进行换入
  - 产生了虚拟地址的概念

基本概念

- 分段
  - 分页技术中，页对程序员来说是不可见的，由操作系统来完成，通过分页技术，给程序员提供了比实际内存更大的地址空间
  - 分段技术对程序员来说是可见的。程序员或操作系统可以给程序或数据分配不同的段，一个程序可以包含多个程序段或数据段
  - 好处是：简化管理要求，允许程序对不同段进行独立修改和编译，并且多个进程可以共享段，可以实现段保护
  - 一般采用分段和分页相结合的方式。段表决定段的开始地址

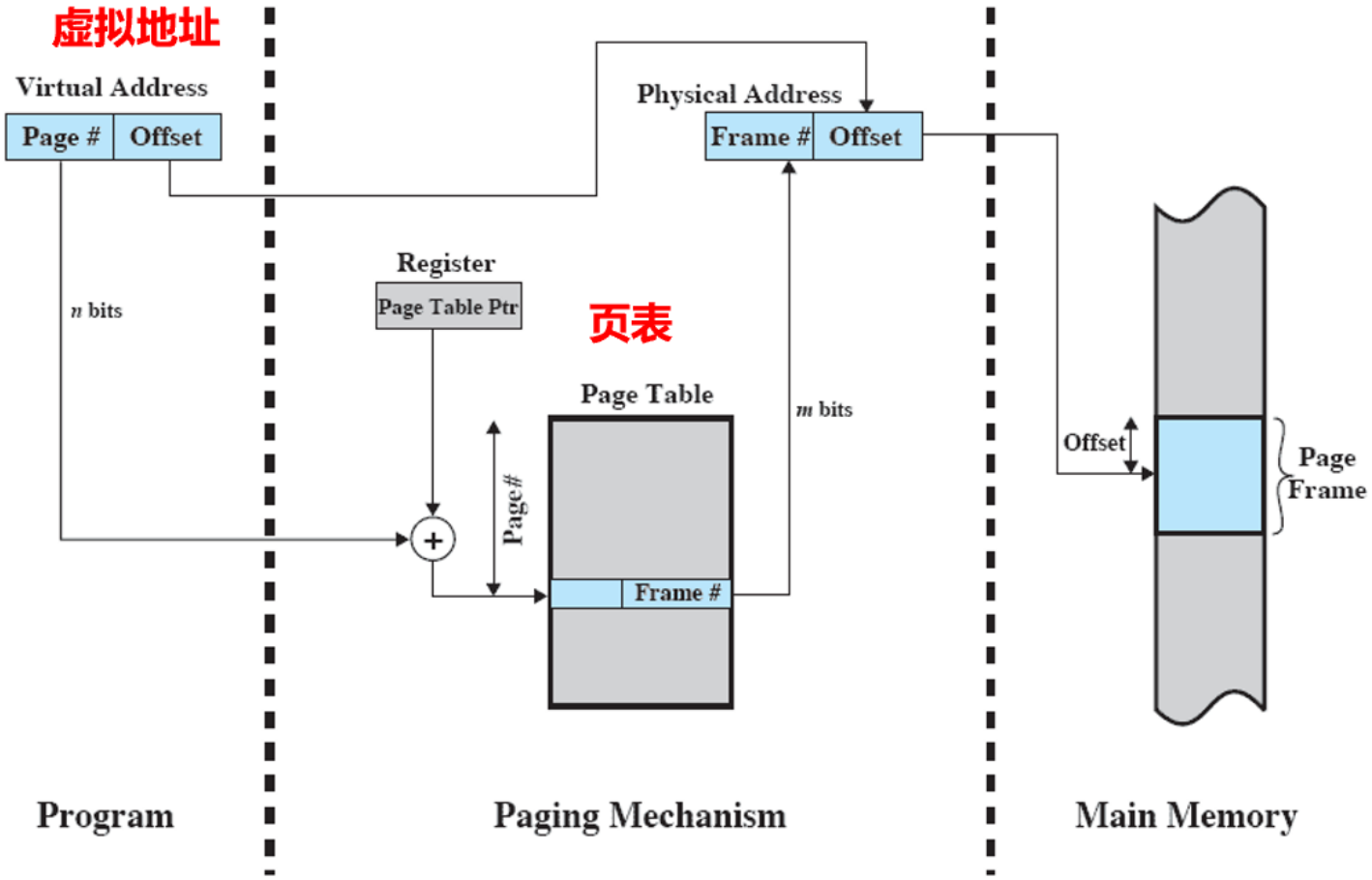
重点知识点

- 交换示意图


Disk storage
Main memory
Operating system
Long-term queue
Completed jobs and user sessions
长周期队列
(a) Simple job scheduling
Disk storage
Main memory
Intermediate queue
Operating system
中间队列
Long-term queue
Completed jobs and user sessions
长周期队列
(b) Swapping

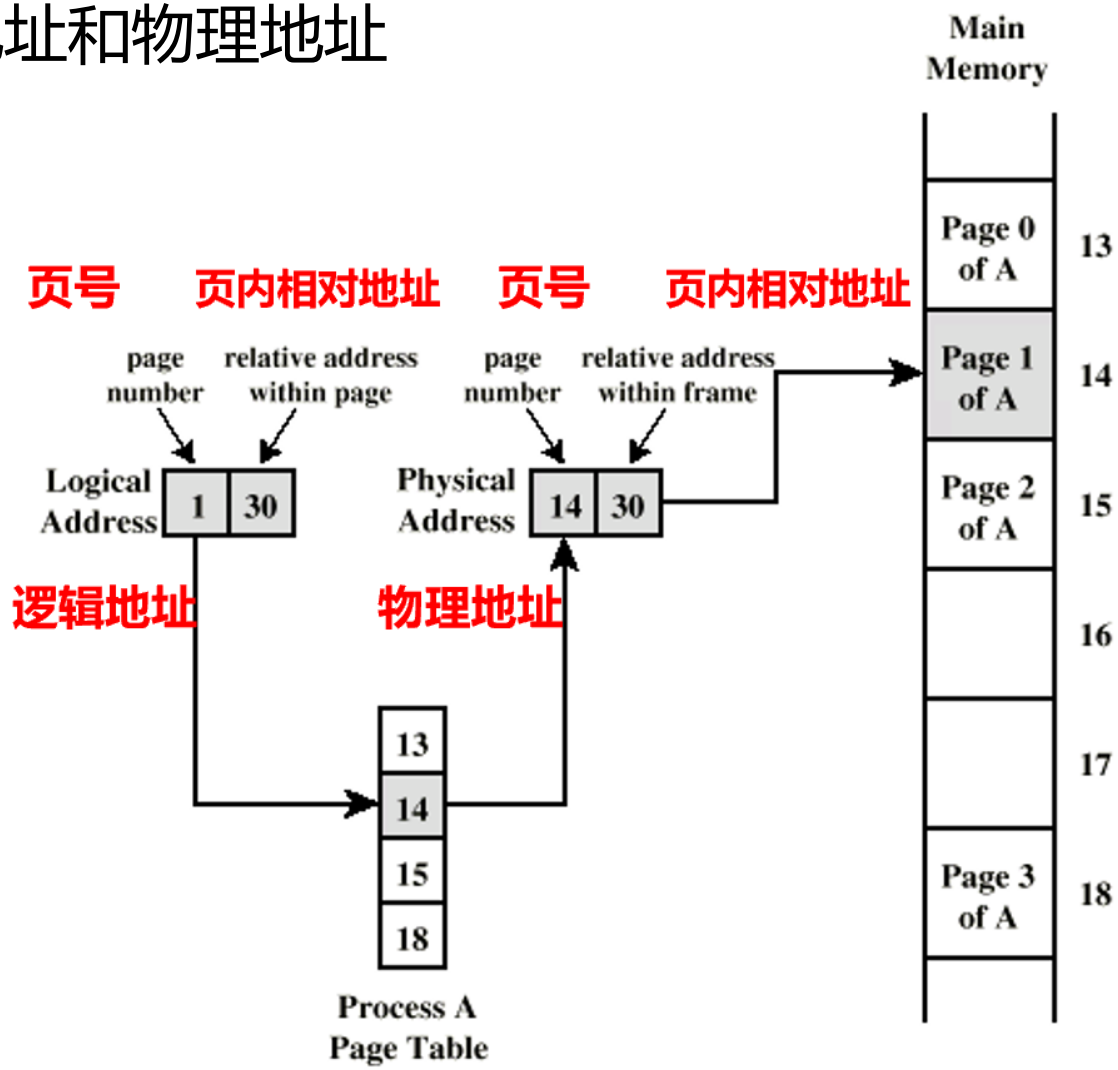
重点知识点

- 分页的地址翻译



重点知识点

- 逻辑地址和物理地址



重点知识点

- 虚拟存储器
  - 请求分页
    - 一个进程的所有页不需要都加载到存储器中
    - 需要的页才会载入内存中
  - 页失效的处理
    - 需要的页不在内存中的时候，就会产生一个“页失效”，page fault。
    - 操作系统将需要的页载入到内存中，同时需要将某页替换出去，以给载入的页提供空间，这称之为“页替换”。
    - 将哪个页替换出去，是需要考虑的一个问题。通常是采用最近最少使用原则

重点知识点

- 虚拟存储器的优点
  - 不需要将所有需要运行的进程放到内存中
  - 根据需要交换页
  - 运行的进程大于实际可用的内存
  - 主存称为实际内存
  - 用户程序能够看到更多的内存-虚拟内存