计算机组成原理

存储系统概述、SRAM、DRAM

王浩宇,教授

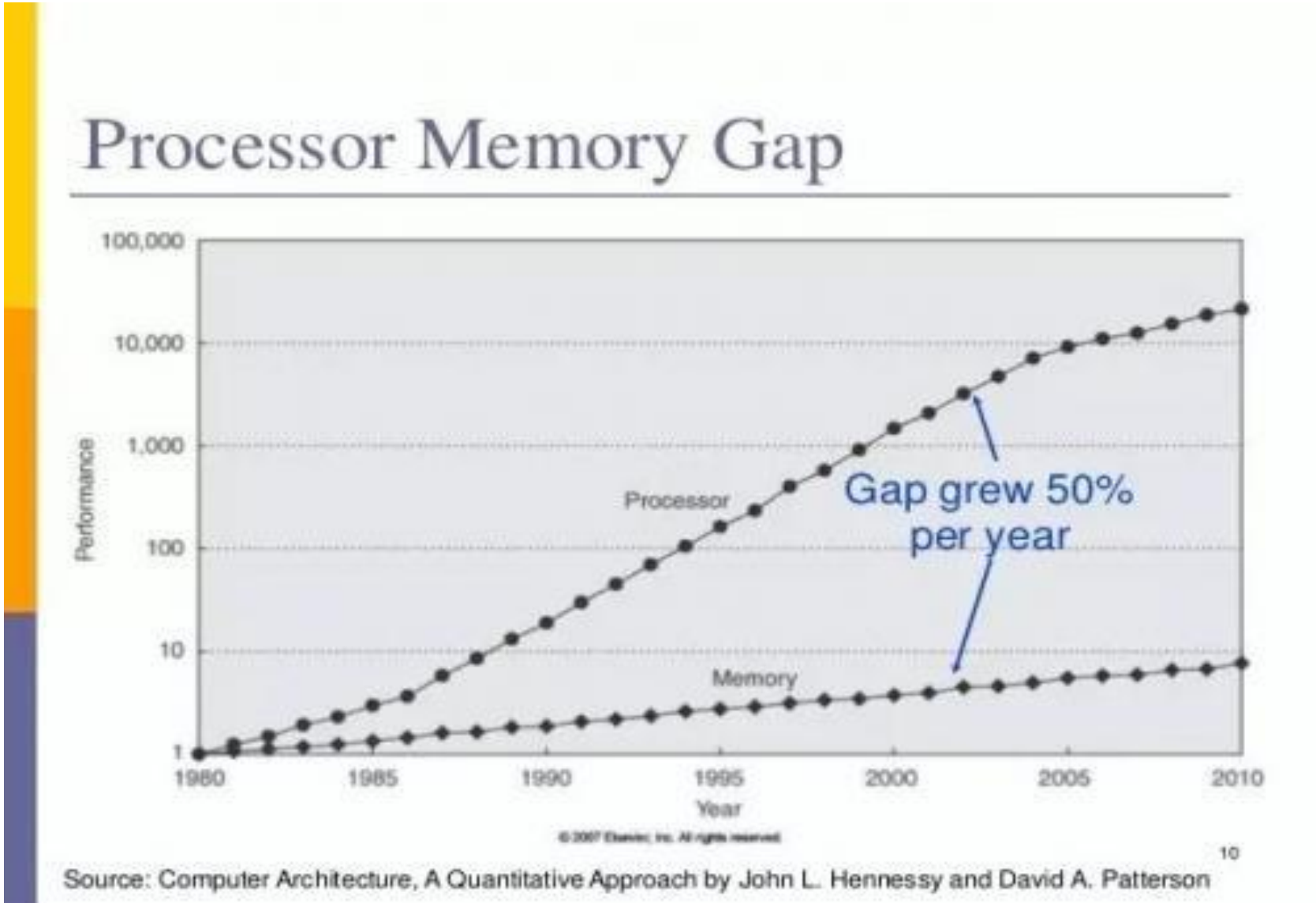
haoyuwang@hust.edu.cn

https://howiepku.github.io/

本节内容

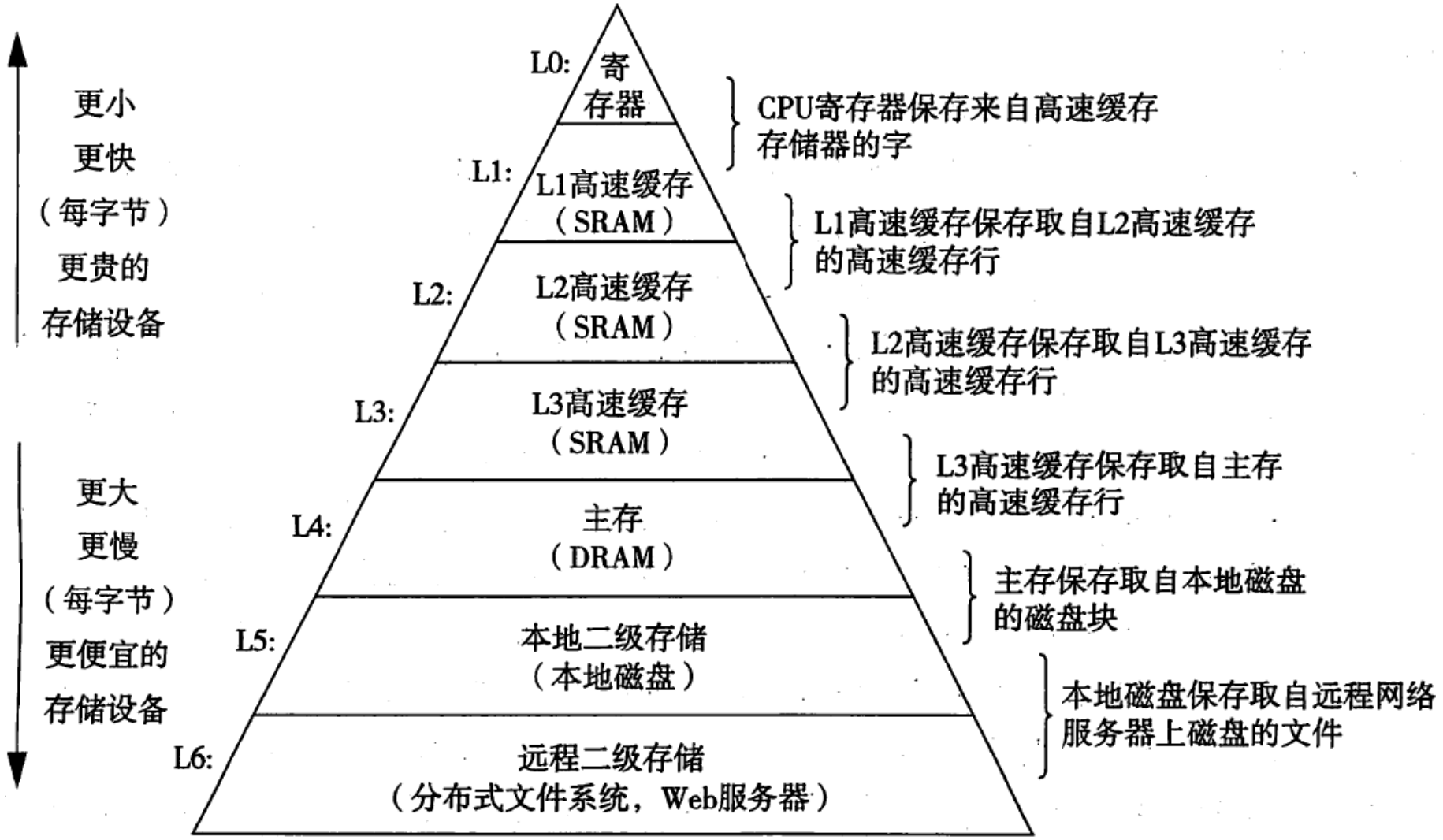
- 存储器概述
- SRAM存储器
- DRAM存储器

Processor Memory Gap



Source: Computer Architecture, A Quantitative Approach by John L. Hennessy and David A. Patterson

存储器层次结构

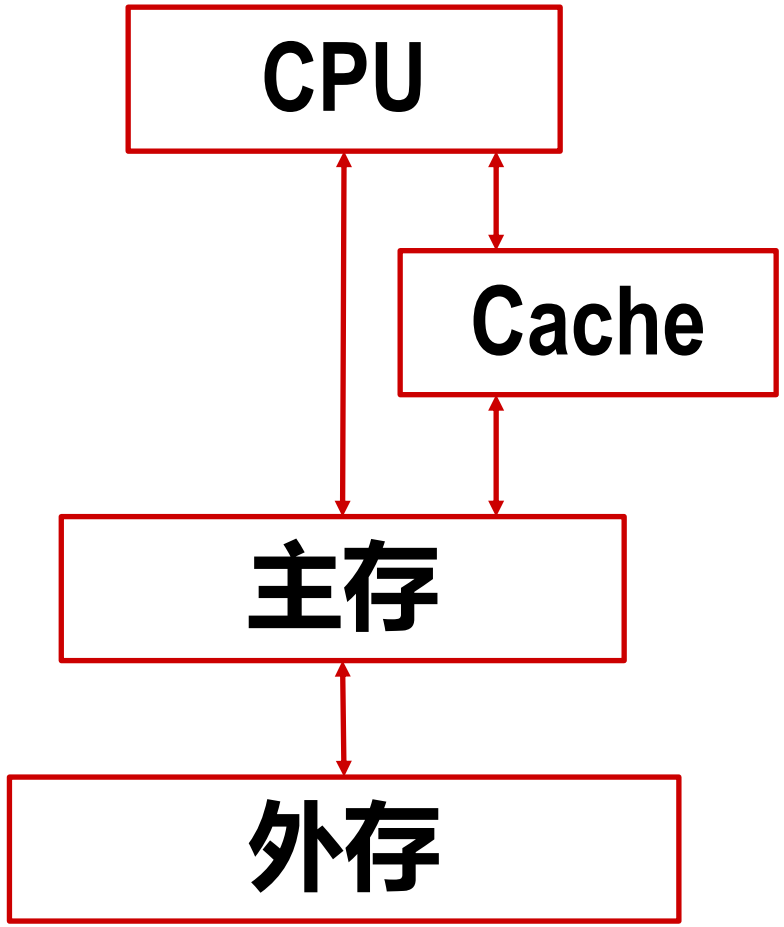

更小
更快
（每字节）
更贵的
存储设备
更大
更慢
（每字节）
更便宜的
存储设备
L0:
寄存器
CPU寄存器保存来自高速缓存存储器的字
L1:
L1高速缓存（SRAM）
L1高速缓存保存取自L2高速缓存的高速缓存行
L2:
L2高速缓存（SRAM）
L2高速缓存保存取自L3高速缓存的高速缓存行
L3:
L3高速缓存（SRAM）
L3高速缓存保存取自主存的高速缓存行
L4:
主存（DRAM）
主存保存取自本地磁盘的磁盘块
L5:
本地二级存储（本地磁盘）
本地磁盘保存取自远程网络服务器上磁盘的文件
L6:
远程二级存储（分布式文件系统，Web服务器）

多级存储体系

- 存储系统目标：容量大、速度快、成本低

对某类存储器而言，这些要求往往是相互矛盾的，如容量大，速度往往不能很快；速度快，成本不可能低。

因此，在一个存储系统中常采用几种不同的存储器，构成多级存储体系，满足系统的要求。



存储系统层次结构

- 主存储器（内存）

  主要存放CPU当前使用的程序和数据 { 速度快；容量有限

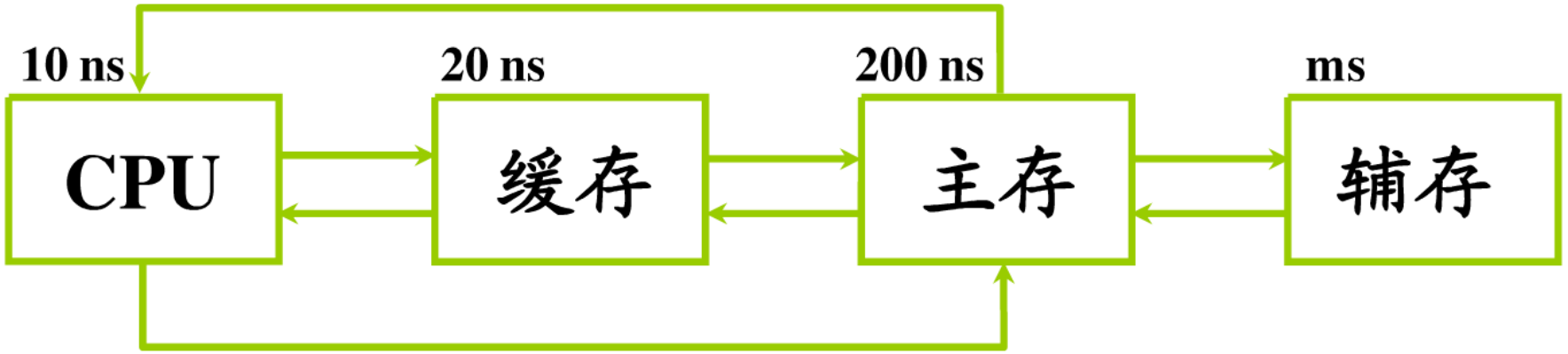
- 辅助存储器（外存）

  存放大量的后备程序和数据 { 速度较慢；容量大

- 高速缓冲存储器Cache

  存放CPU在当前一小段时间内多次使用的程序和数据 { 速度很快；容量小

存储系统层次结构



注意：主存与缓存之间的数据调动是由*硬件自动完成*的。

主存与辅存之间数据调动是*由硬件和OS共同完成*。

（解决速度）　　（解决容量）

缓存 — 主存　　主存 — 辅存

主存储器	虚拟存储器
实地址	虚地址
物理地址	逻辑地址

虚拟存储器

- **仅把应用程序的一部分装入内存便可以运行应用程序的存储器系统**
  - 使得应用程序认为它拥有连续的可用的内存（一个连续完整的地址空间），而实际上，它通常是被分隔成多个物理内存碎片，还有部分暂时存储在外部磁盘存储器上，在需要时进行数据交换

虚拟存储器

- **具有请求调入功能和置换功能，能从逻辑上对内存容量进行扩充**
- **逻辑容量是由内存和外存容量之和所决定，其运行速度接近于内存速度，而每位的成本又接近于外存**

存储器分类

- **按存储介质分类**
  - 半导体存储器
    - 静态存储器：利用双稳态触发器存储信息
    - 动态存储器：依靠电容存储电荷存储信息
  - 磁表面存储器：利用磁层上不同方向的磁化区域表示信息，容量大，非破坏性读出，长期保存信息，速度慢。
  - 光盘存储器
    - 利用光斑的有无表示信息

存储器分类

- 按存取方式分类

（1）存取时间与物理地址无关（随机访问）

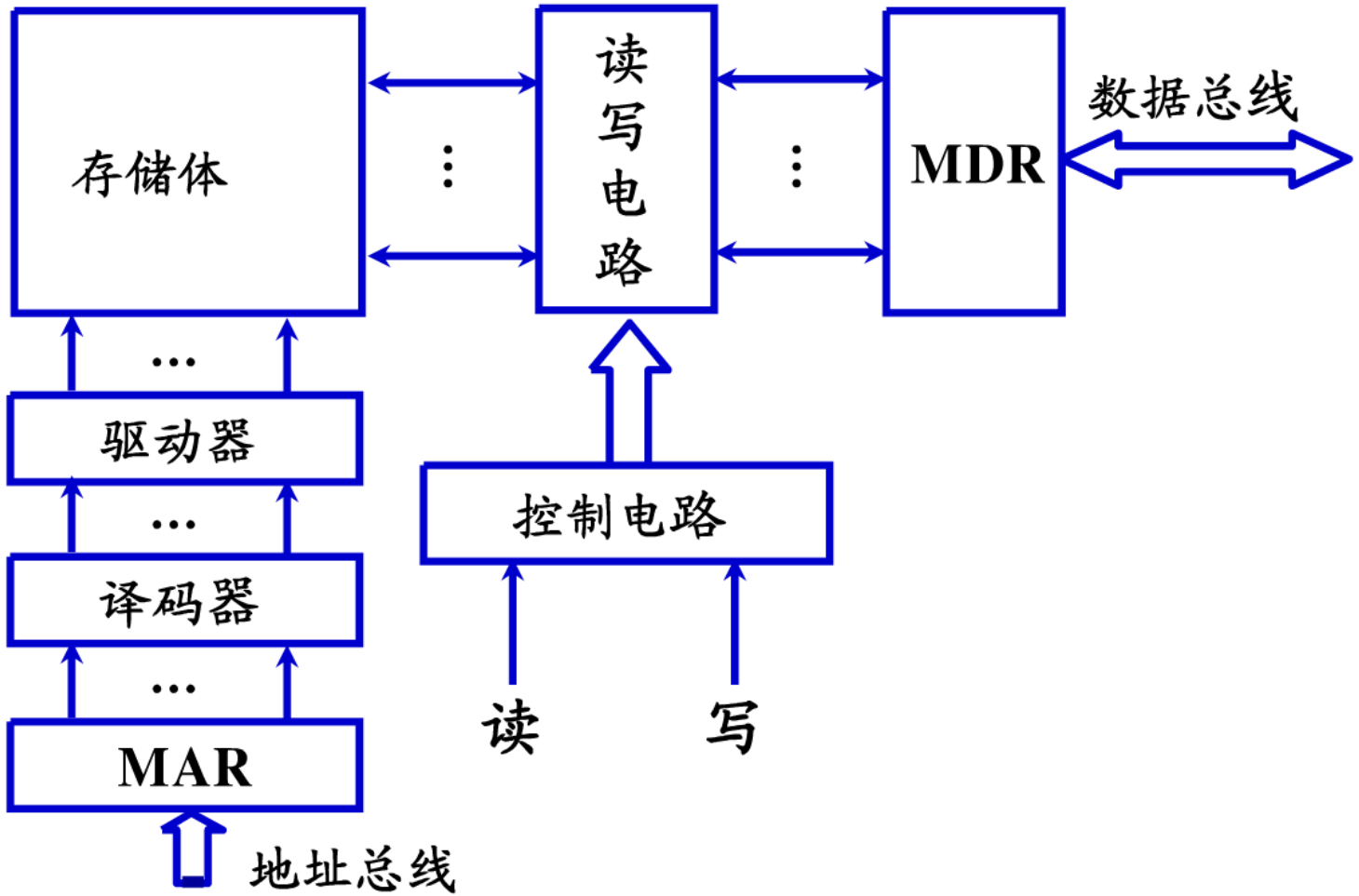
- 随机存储器（RAM）：可读可写　　内存
- 只读存储器（ROM）：只读　　存放BIOS的芯片

（2）存取时间与物理地址有关（串行访问）

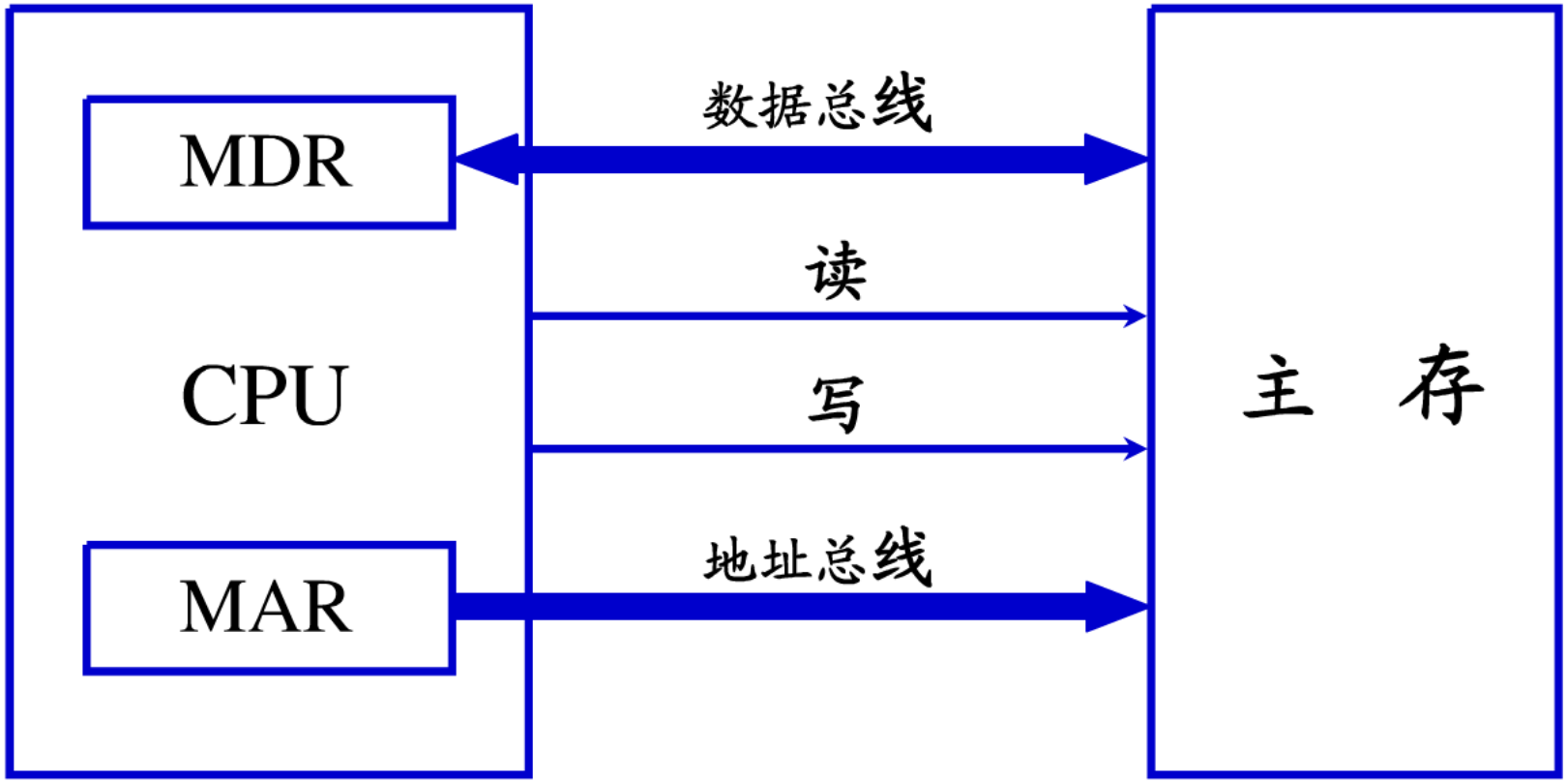
- 顺序存取存储器：磁带
- 直接存取存储器：磁盘

主存储器概述

- 主存的基本组成



主存和CPU的联系

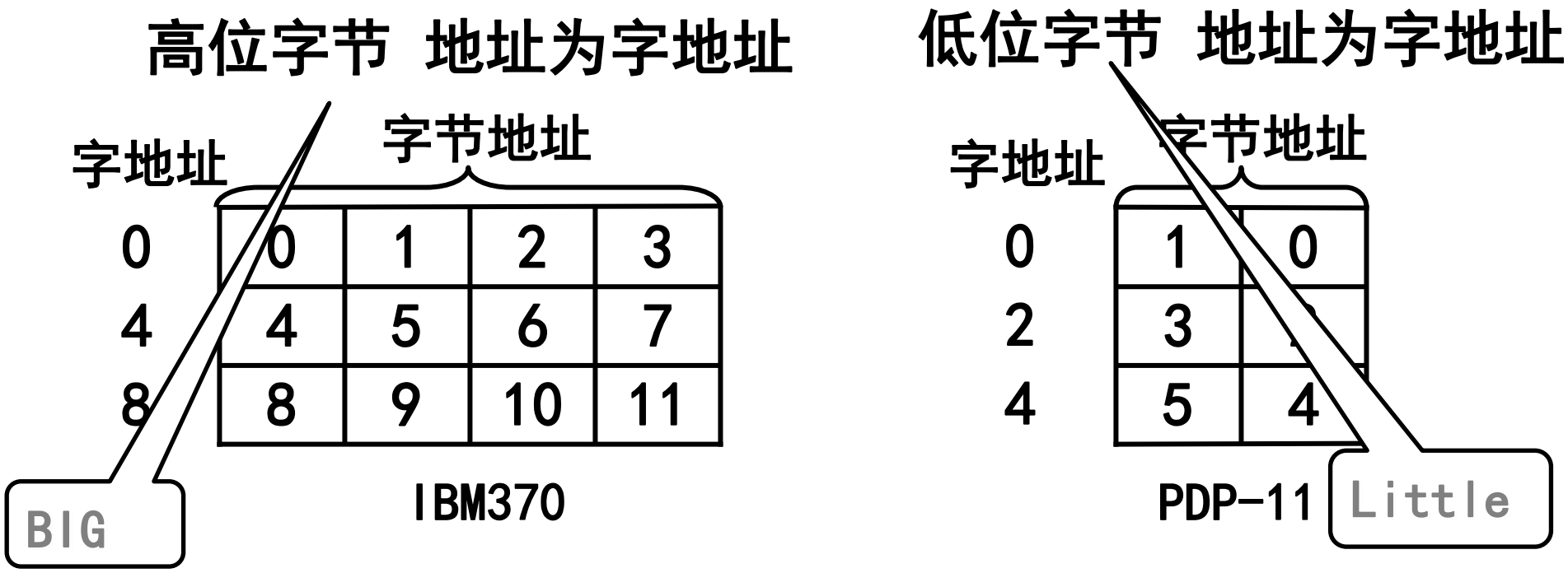


主存的存储单元地址分配

- 字节：8b=1B
- 字：存储字长
  - 双字：两个字
  - 四字：四个字

- 8*8的芯片，字节？字？
- 8*16 字节？字？
- 8*32 字节？字？

主存的存储单元地址分配



- 小端存储：较低的有效字节存放在较低的存储器地址，较高的字节存放在较高的存储器地址；
- 大端存储：较低的有效字节存放在较高的存储器地址，较高的字节存放在较低的存储器地址

如果将一个32位的整数0x12345678 存放到一个整型变量（int）中，这个整型变量采用大端或者小端模式在内存中的存储由下表所示。用OP_0表示一个32位数据的最高字节MSB（Most Significant Byte），使用OP_3表示一个32位数据最低字节LSB（Least Significant Byte）。

地址偏移	大端模式	小端模式
0x00	12（OP_0）	78（OP_3）
0x01	34（OP_1）	56（OP_2）
0x02	56（OP_2）	34（OP_1）
0x03	78（OP_3）	12（OP_0）

小端：较高的有效字节存放在较高的的存储器地址，较低的有效字节存放在较低的存储器地址。

大端：较高的有效字节存放在较低的存储器地址，较低的有效字节存放在较高的存储器地址。

课堂练习

- 假定在一个程序中定义了变量x、y和i，其中，x和y是float型变量（用IEEE754单精度浮点数表示），i是16位short型变量（用补码表示）。程序执行到某一时刻，x = －0.125、y=7.5、i=100，它们都被写到了主存（按字节编址），其地址分别是100，108和112。请分别画出在大端机器和小端机器上变量x、y和i在内存的存放位置。

提问

- 如何写程序判断你的计算机是大端还是小端模式?

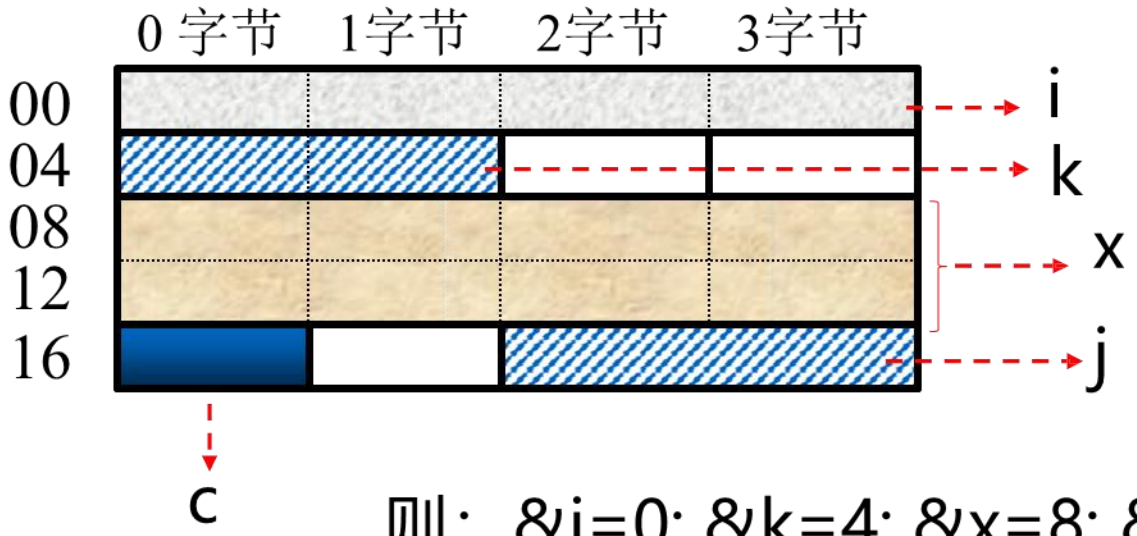
数据的边界对齐

- **现代计算机中主存空间按照字节编址**
- **高级语言中不同数据类型的变量对应不同的字节长度**
  - 不同数据类型的变量包含1个或者多个字节单元
  - 变量在进行主存地址空间分配时，理论上可以从任何字节地址开始
    - 当一个多字节变量分布在不同的字存储单元中时，访问该变量需要多个存储周期
- **为提高数据访问效率，通常要考虑数据变量、数据结构在主存空间中的边界对齐问题**

数据存储与边界的关系

1)按边界对齐的数据存储

int i, short k, double x, char c, short j,……　(32位系统中)

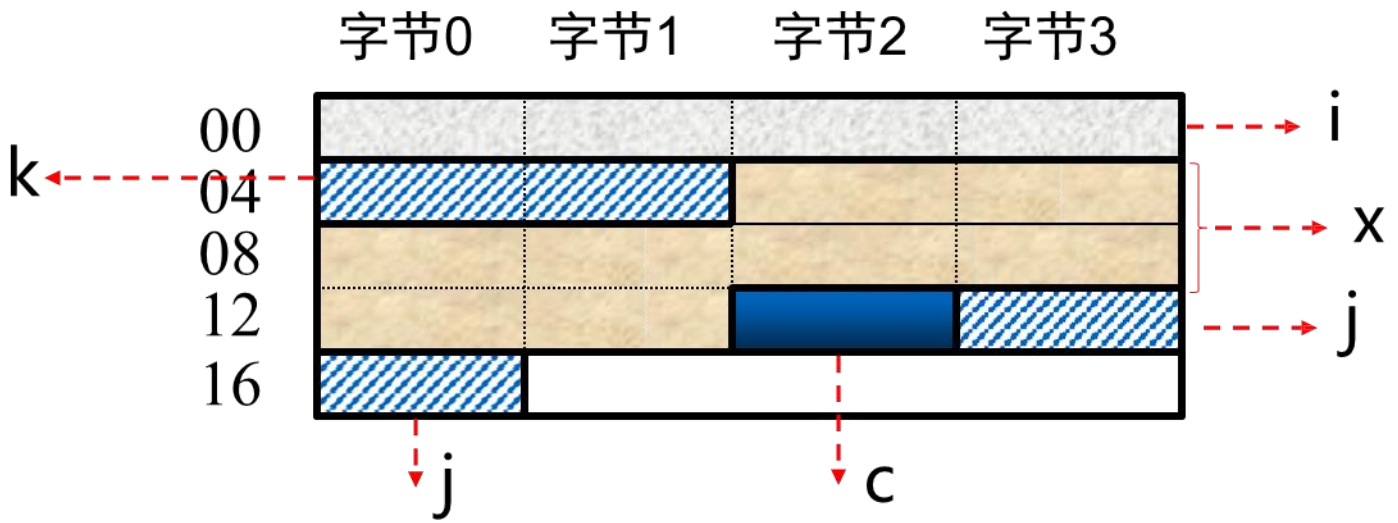


则：&i=0; &k=4; &x=8; &c=16; &j=18;……

数据存储与边界的关系

2)未按边界对齐的数据存储

int i, short k, double x, char c, short j,……　(32位系统中)



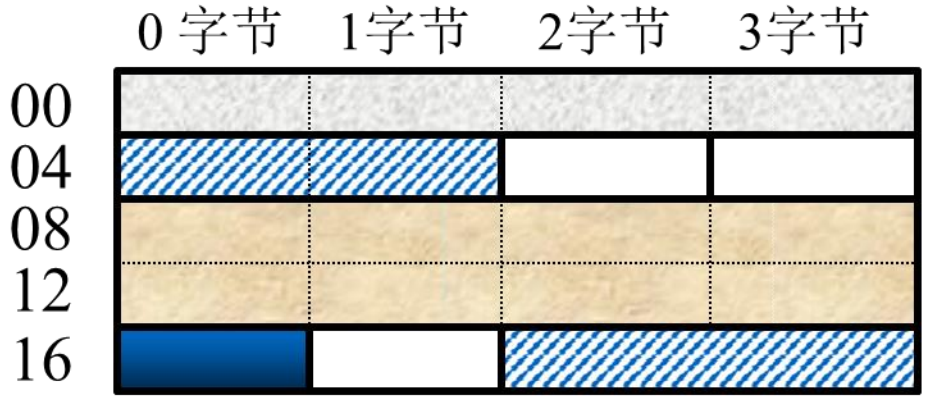
则：　&i=0; &k=4; &x=6; &c=14; &j=15;……

虽节省了空间，但增加了访存次数！

需要在性能与容量间权衡！

数据存储与边界的关系

3)边界对齐与存储地址的关系（以32位为例）



- 双字长数据边界对齐的起始地址的最末三位为000(8字节整数倍;
- 单字长边界对齐的起始地址的末二位为00(4字节整数倍);
- 半字长边界对齐的起始地址的最末一位为0(2 字节整数倍)。

寻址方式

- **按字节寻址　一个字节一个地址**

8*16 需要几位地址线?

- **按字寻址　一个字一个地址**

8*16 需要几位地址线?

- 1MB 按照字节寻址　需要多少位地址线?
- 1MB 按16位字寻址 需要多少位地址线?

主存储器的技术指标

- **字存储单元：**存放一个机器字的存储单元，相应的单元地址叫字地址。
- **字节存储单元：**存放一个字节的单元，相应的地址称为字节地址。
- **存储容量：**指一个存储器中可以容纳的存储单元总数。存储容量越大，能存储的信息就越多
  - 可以用字数或者字节数来表示

按字节或者按字寻址，容量为多少字节

单位：KB, MB, GB. .

地址线数决定最大直接寻址空间大小（n位地址：2^n）

主存储器的技术指标

- **存取时间又称存储器访问时间：**指一次读操作命令发出到该操作完成，将数据读出到数据总线上所经历的时间。通常取写操作时间等于读操作时间，故称为存储器存取时间。
- **存储周期：**指连续启动两次读操作所需间隔的最小时间。通常，存储周期略大于存取时间，其时间单位为ns。
- **存储器带宽：**单位时间里存储器所存取的信息量，通常以位/秒或字节/秒做度量单位。

与存储周期有关。如：存储周期为500ns，每个存储周期访问16位，则存储器的带宽为？

16*1/500ns = 32M位/S

提高存储器带宽，可采用以下措施：

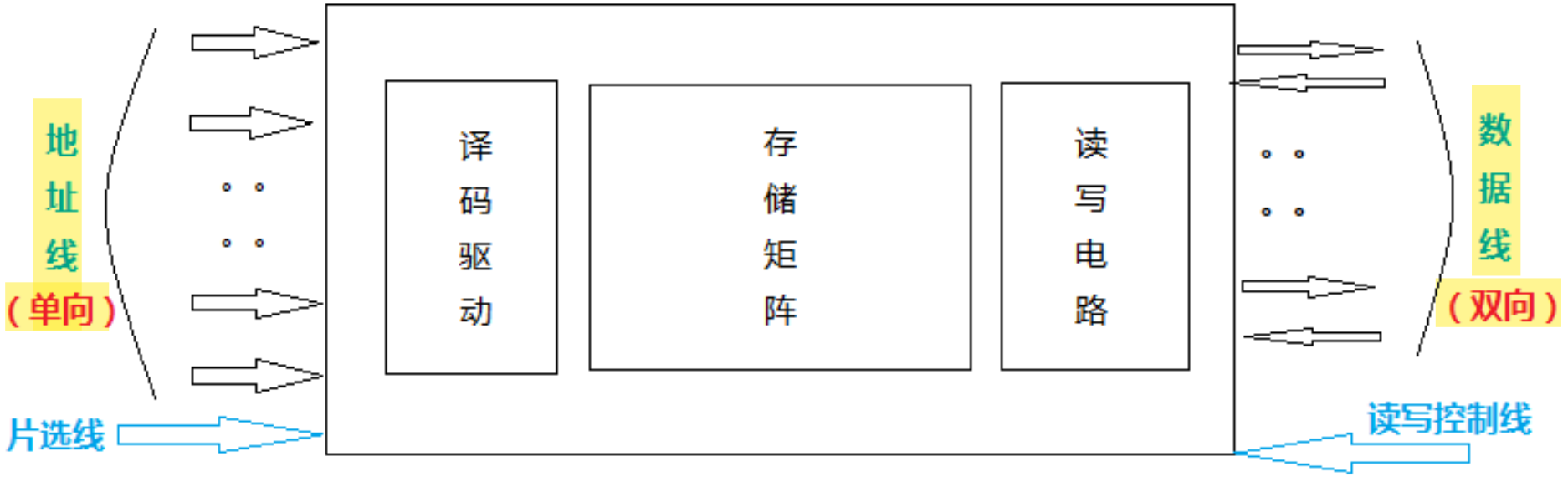
(1) 缩短存取周期

(2) 增加存储字长

(3) 增加存储体

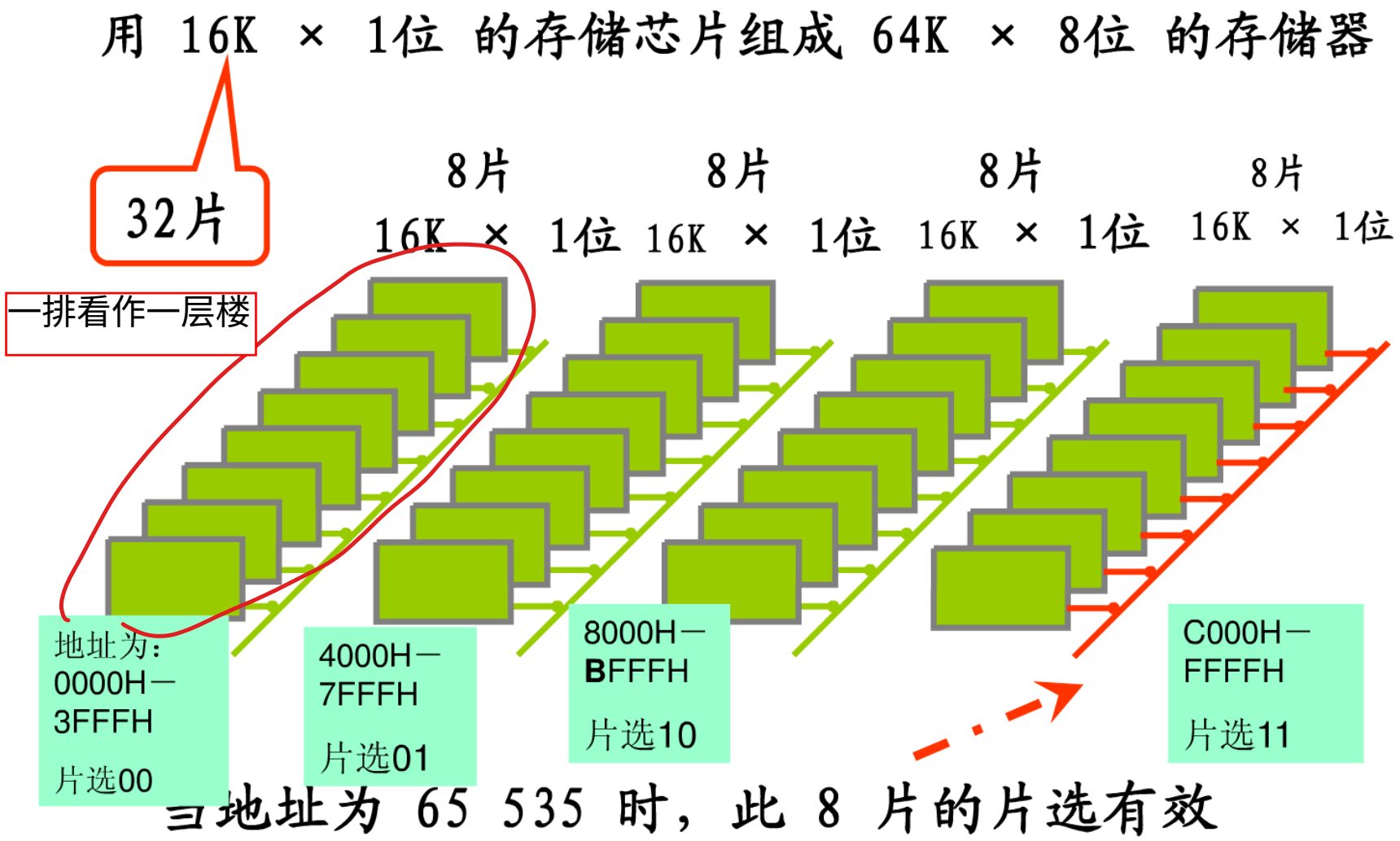
半导体存储芯片

- **半导体存储芯片的基本结构**



- 地址线单向，数据线双向。
- 半导体芯片的容量由地址线和数据线一起决定。
  地址线和数据线的根数表示了内存实际的大小，而cpu理论上的最大寻址范围，由MAR和MDR的大小决定，为$2^{MAR}*MDR$

存储芯片片选线的作用


用 16K × 1位 的存储芯片组成 64K × 8位 的存储器
32片
8片
16K × 1位
8片
16K × 1位
8片
16K × 1位
8片
16K × 1位
一排看作一层楼
地址为:
0000H—
3FFFH
片选00
4000H—
7FFFH
片选01
8000H—
BFFFH
片选10
C000H—
FFFFH
片选11
当地址为 65 535 时，此 8 片的片选有效

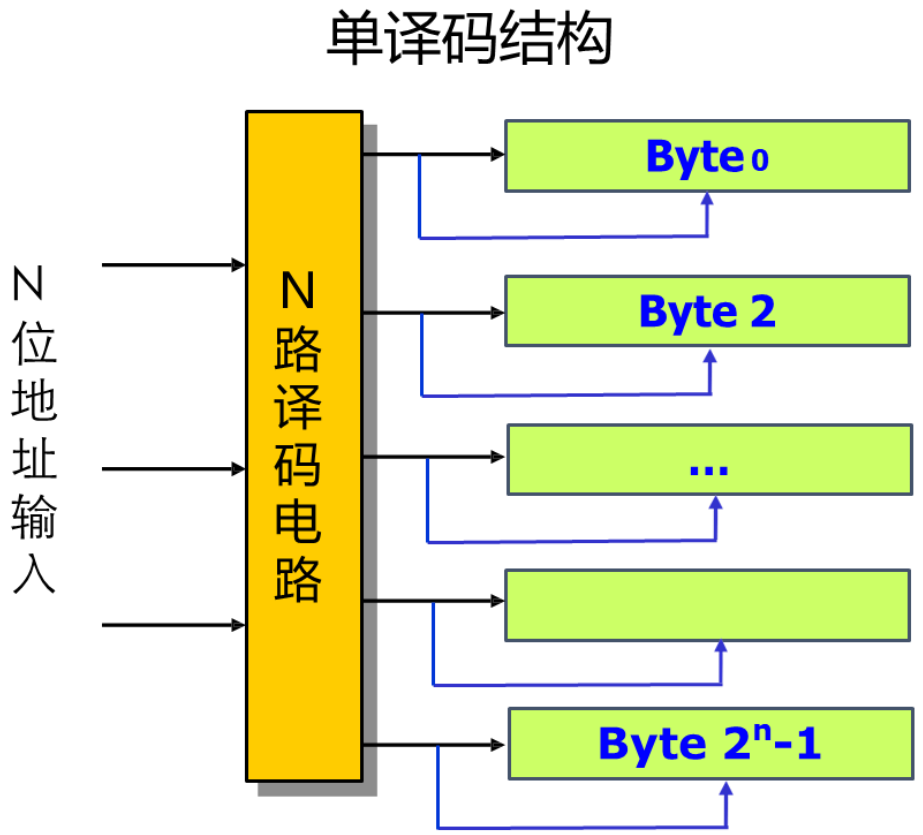
存储芯片片选线的作用

- 用16K*1位的存储芯片组成64K*8位的存储器
  - 因为半导体芯片是用存储矩阵设计的，存储矩阵的一行可以看成一层楼，这层楼有多个小房间。

    因此，先把8个16*1K的存储器放在一行，构成一层楼。再安排4层这样的楼层，构成64K*8的大楼
  - 因为，一个地址线的地址过来后，先通过片选线（地址为的最高几位），选择楼层，然后用剩下的地址线低位地址，选择是楼层的那个房间
  - 例如，地址线过来的数据是65535，转换成二进制是1111 1111。上面楼层有4层，所以地址线的前两位进行片选，也就是11，因此选择最高层。

地址译码方式

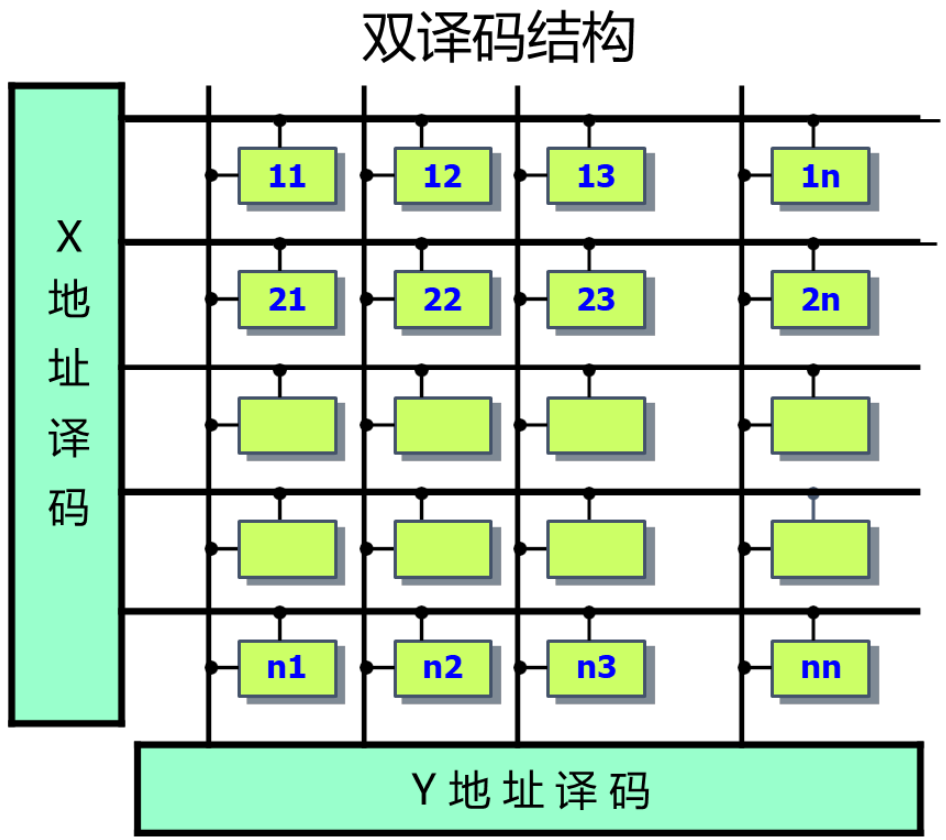
- 单译码方式（线选法）：每个小存储单元占用一行，构成多行的线性结构



N位地址，寻址2^n个存储单元，2^n根译码线

地址译码方式

- 双译码方式（重合法）：组合多个存储单元为一行



N位地址，寻址2^n个存储单元，$2^{n/2+1}$根译码线

半导体存储器

- 主存（内部存储器）是半导体存储器。
- 根据信息存储的机理不同可以分为两类：
  - 静态读写存储器(SRAM)：

存取速度快，存储容量不如DRAM大，利用双稳态触发器存储信息，读写速度快，生产成本高，多用于容量较小的高速缓冲存储器（Cache）

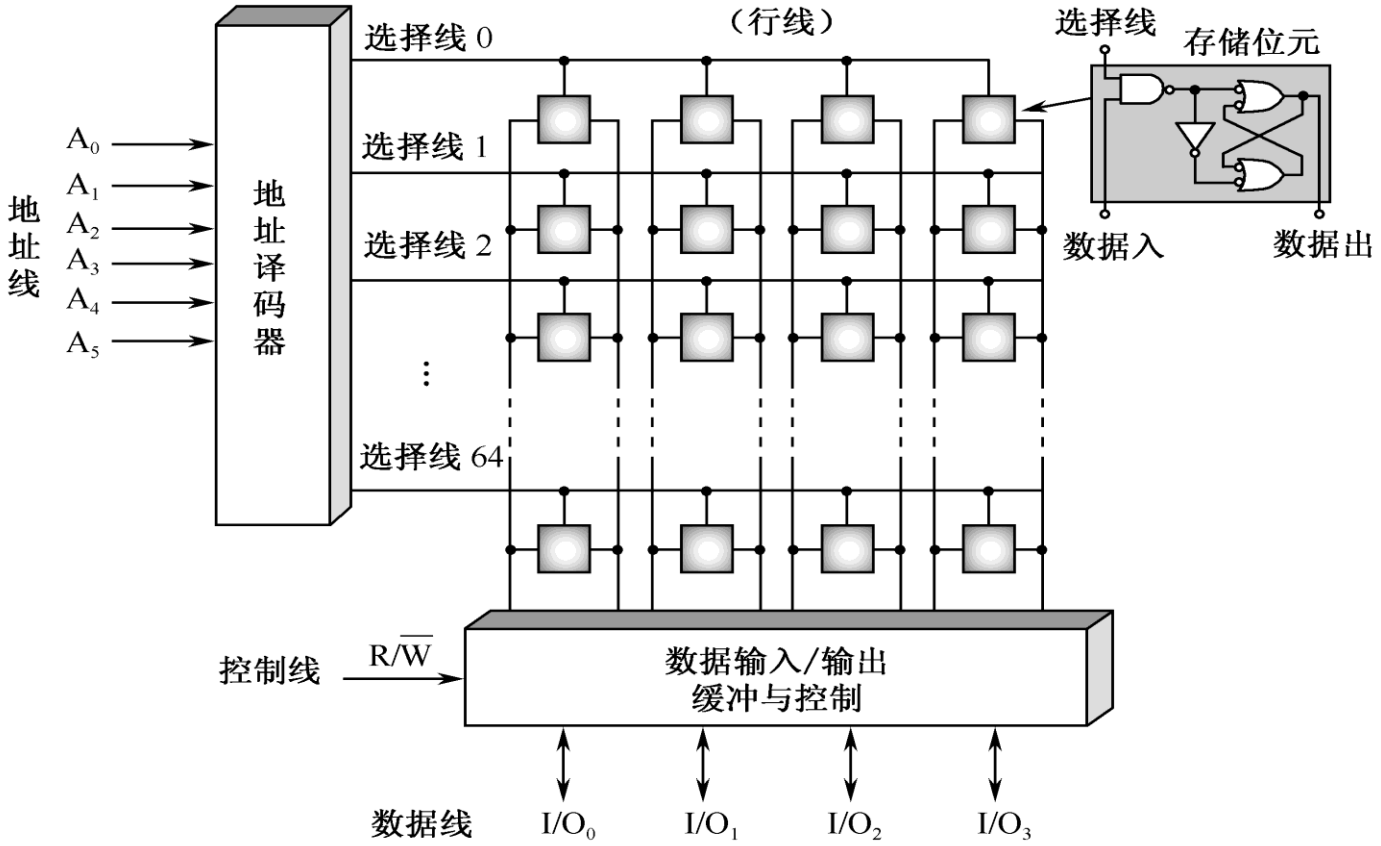
  - 动态读写存储器(DRAM)：

依靠电容存储电荷存储信息，在不进行读写操作时，DRAM 存储器的各单元处于断电状态，由于漏电的存在，保存在电容CS上的电荷会慢慢地漏掉，需要定期刷新。

读写速度较慢，集成度高，生产成本低，多用于容量较大的主存储器

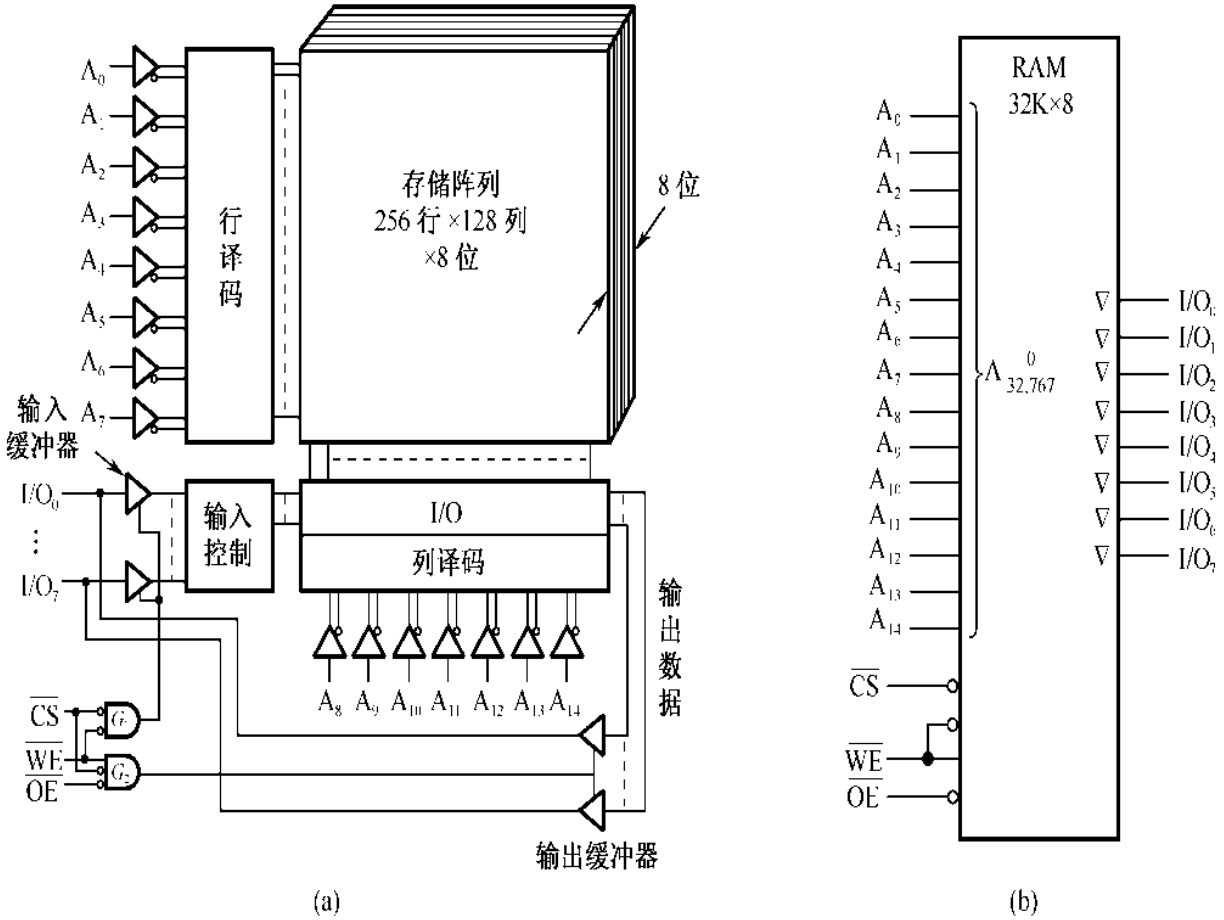
SRAM存储器

- 所有的SRAM是用一个锁存器(触发器)作为存储位元。
- SRAM包含三组信号：
  - 地址线：选择单元，确定容量(单元数)
  - 数据线：单元的位数
  - 控制线：读写控制
- SRAM的结构
  - 地址译码器
  - 存储矩阵
  - 输入/输出电路



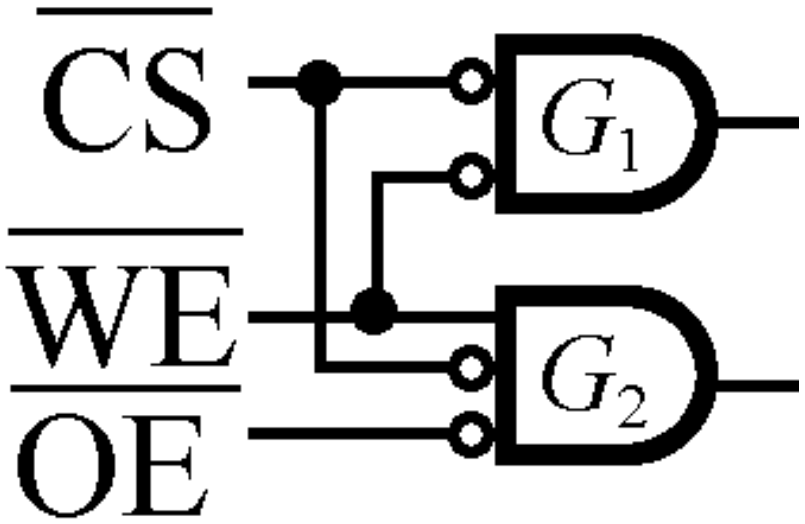
基本的SRAM逻辑结构

- SRAM芯大多采用双译码方式，以便组织更大的存储容量。采用了二级译码：将地址分成x向、y向两部分

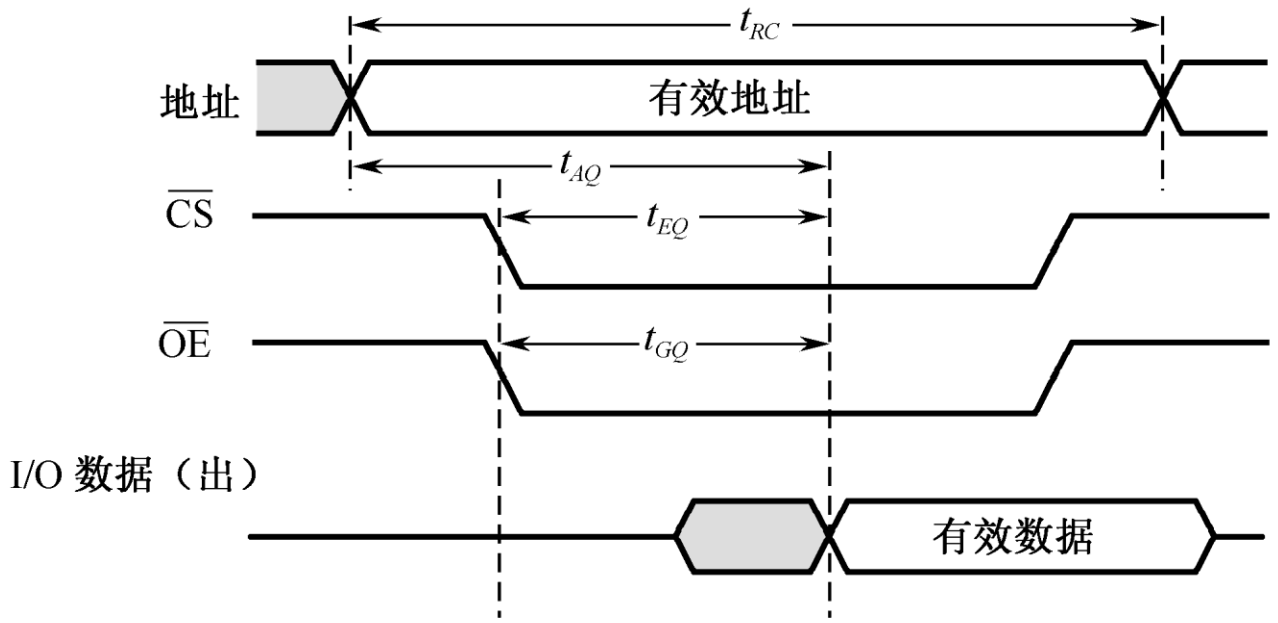


SRAM存储器

- 读与写的互锁逻辑
  - 控制信号中CS是片选信号，CS有效时（低电平），门G1、G2均被打开
  - OE为读出使能信号，OE有效时（低电平），门G2开启，当写命令WE=1时（高电平），门G1关闭，存储器进行读操作。
  - 写操作时，WE=0，门G1开启，门G2关闭。
  - 注意，门G1和G2是互锁的，一个开启时另一个必定关闭，这样保证了读时不写，写时不读



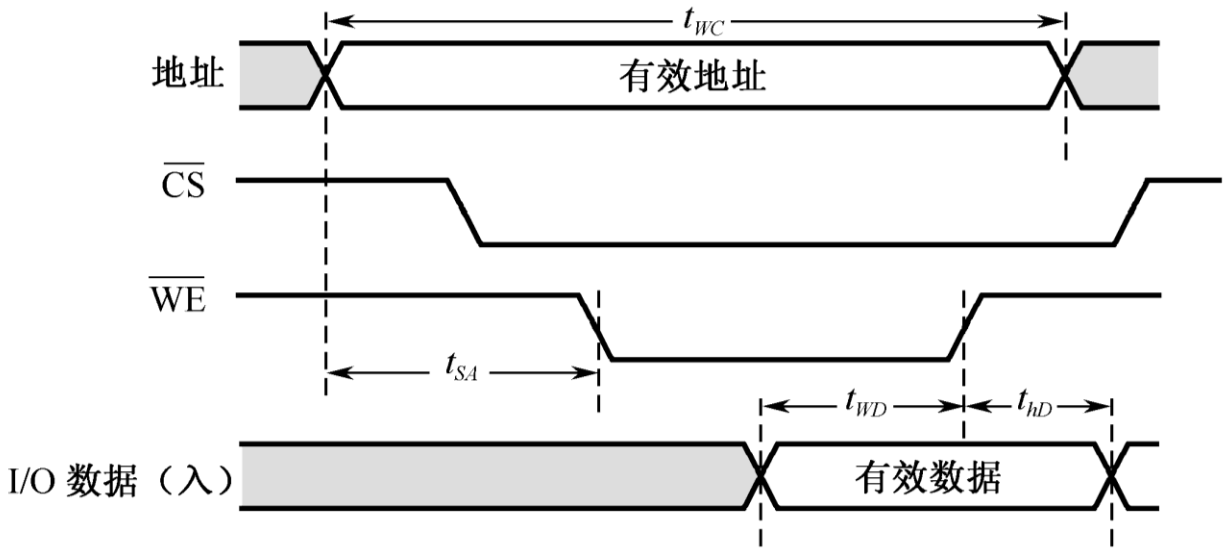
SRAM存储器的读写周期



(a) 读周期 ($\overline{WE}$ 高)

- **读周期**
  - 地址线先有效，以便进行地址译码，选中存储单元。
  - 为了读出数据，片选信号CS和读出使能信号OE也必须有效
  - 从地址有效开始经过t_{AQ}时间（读出时间），数据总线IO出现有效的读出数据，之后CS、OE信号恢复高电平，t_{RC}以后才允许地址总线发生改变。t_{RC}时间为读周期时间。
  - 读周期与读出时间是两个概念，读周期时间表示存储片进行两次连续读操作时所必须间隔的时间，总是大于等于读出时间

SRAM存储器的读写周期



(b) 写周期 ($\overline{WE}$ 低)

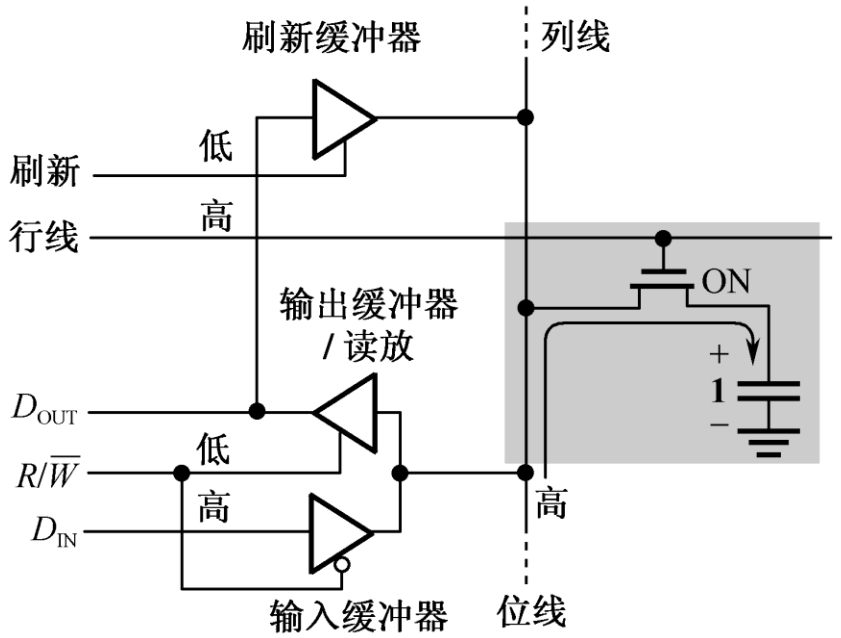
- **写周期**
  - 地址线先有效，然后片选信号CS有效，写命令WE有效
  - 为保证地址变化期间不会发生错误写入而破坏存储器内容，只有在地址有效后再经过一段时间Ts后，写命令才有效
  - IO总线置写入数据，在t_{WD}时间段将数据写入存储器
  - 之后撤销WE和CS
  - IO线的写入数据要有维持时间，CS的维持时间也比读周期长
  - t_{WC}为写周期时间

DRAM存储器

- **DRAM存储位元的记忆原理**
  - SRAM存储器的存储位元是一个触发器，它具有两个稳定的状态。
  - **DRAM存储器的存储位元是由一个MOS晶体管和电容器组成的记忆电路**
    - MOS管作为开关使用
    - 存储的信息1或0则是由电容器上面的电荷量体现
      - 当电容器充满电荷时，代表存储了1，当电容器放电没有电荷时，代表存储了0。

DRAM存储器

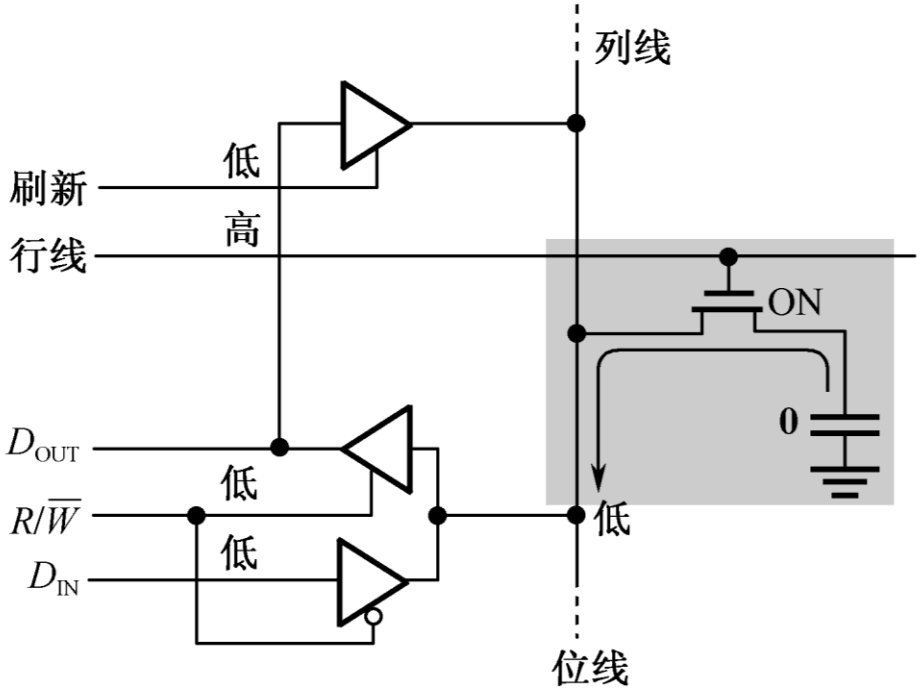
- 写1到存储位元
  - 输出缓冲器关闭、刷新缓冲器关闭，输入缓冲器打开（R/W为低）
  - 输入数据DIN=1送到存储元位线上，而行选线为高，打开MOS管，于是位线上的高平给电容器充电，表示存储了1



(a) 写 1 到存储位元

DRAM存储器

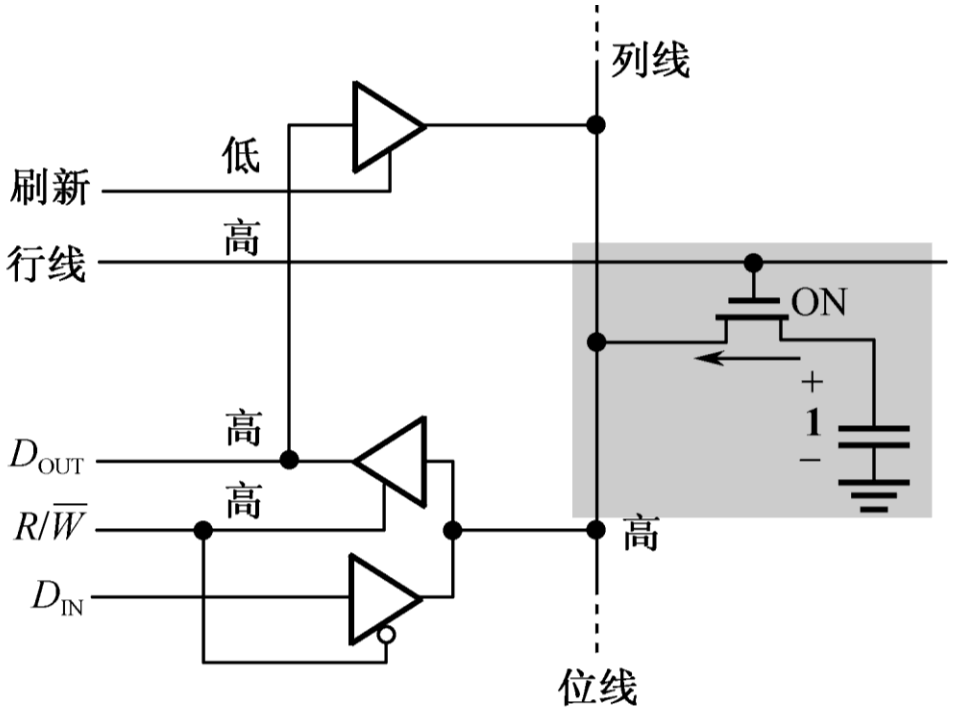
- 写0到存储位元
  - 输出缓冲器和刷新缓冲器关闭，输入缓冲器打开
  - 输入数据DIN=0送到存储元位线上；行选线为高，打开MOS管，于是电容上的电荷通过MOS管和位线放电，表示存储了0



(b) 写 0 到存储位元

DRAM存储器

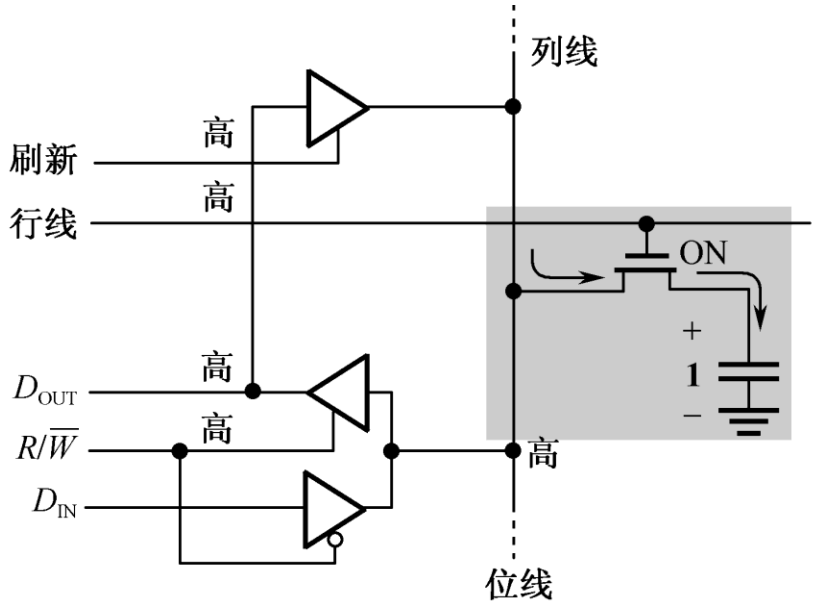
- 从存储位元读出1
  - 输入缓冲器和刷新缓冲器关闭，输出缓冲器/读放打开（R/W为高）
  - 行选线为高，打开MOS管，电容上所存储的1送到位线上，通过输出缓冲器/读出放大器发送到D_{OUT}，即D_{OUT}=1。



(c) 从存储位元读出 1

DRAM存储器

- **读出1后存储位元重写1**
  - 由于(c)中读出1是破坏性读出，必须恢复存储位元中原存的1
  - 输入缓冲器关闭，刷新缓冲器打开，输出缓冲器/读放打开
  - DOUT=1经刷新缓冲器送到位线上，再经MOS管写到电容上
  - 输入缓冲器与输出缓冲器总是互锁的。这是因为读操作和写操作是互斥的，不会同时发生。

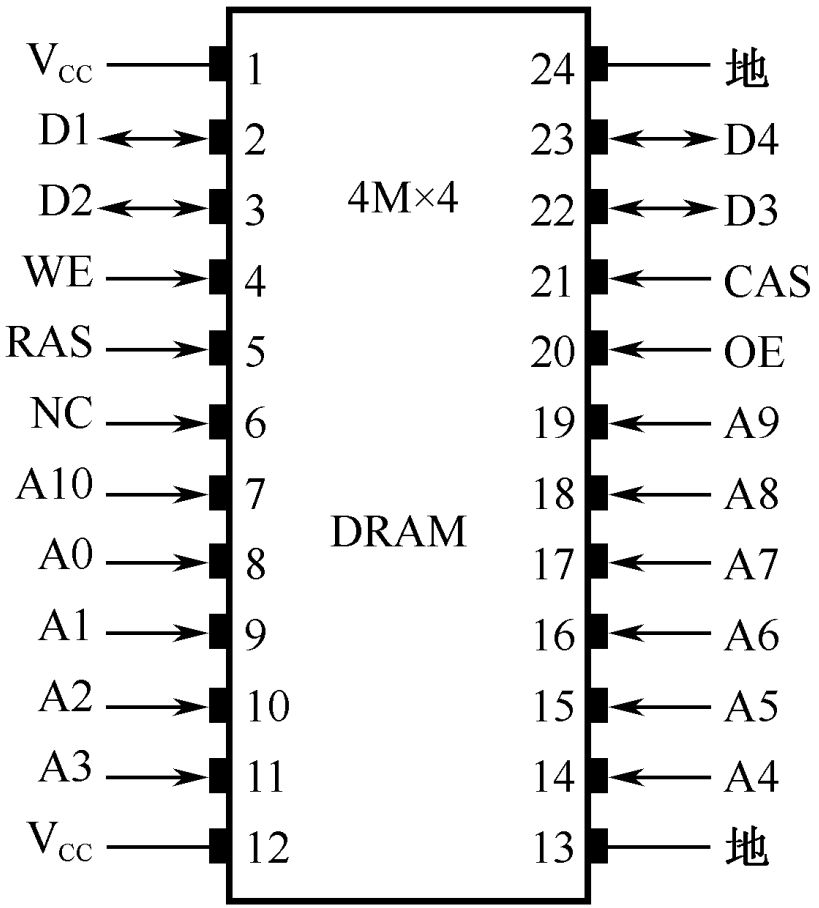


(d) 刷新存储位元的 1

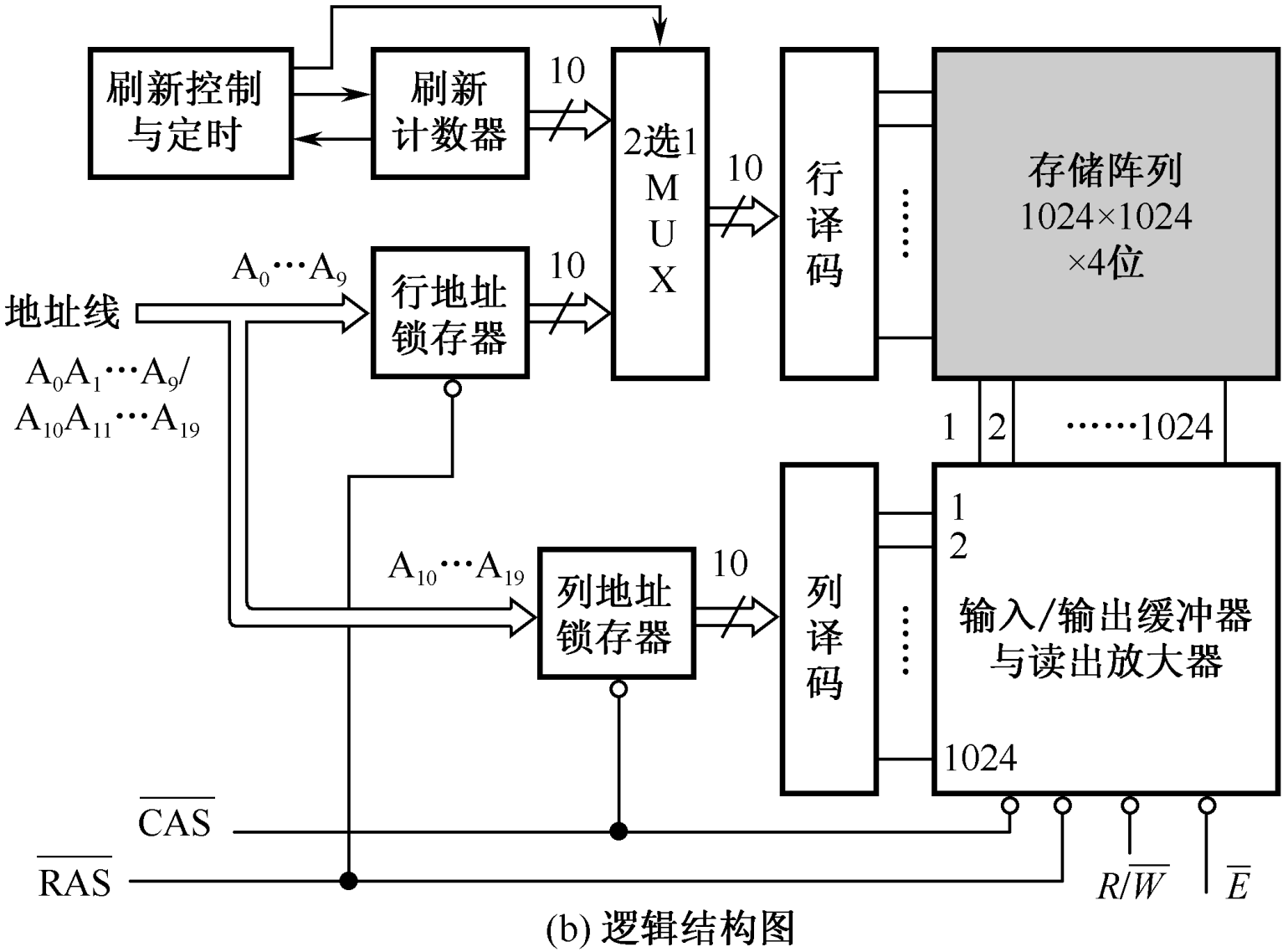
DRAM存储器

- 特点
  - DRAM：动态随机存储器，定时刷新电路，使用电容存储数据，不掉电情况下也需要定时对电容充电。
  - DRAM的存储矩阵是二维的，有行有列
    - 所以要对行和列进行片选。行列片选的片选片进行复用，即同一个针脚，先选择行，后选择列。
  - 动态RAM刷新
    - 刷新与行地址有关：默认数据能保持2ms

DRAM存储器逻辑结构



(a) 管脚图



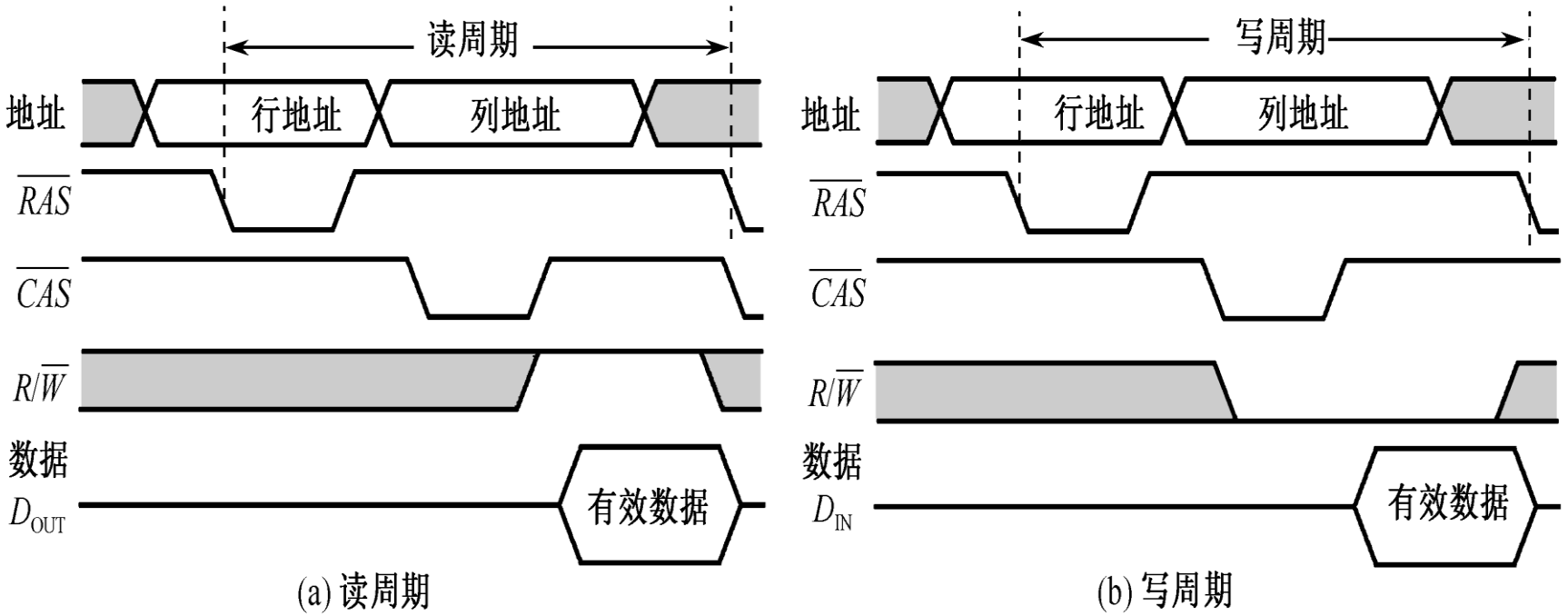
(b) 逻辑结构图

DRAM存储器逻辑结构

- **增加了行地址锁存器和列地址锁存器**
  - 由于DRAM存储器容量很大，地址线宽度相应要增加，这势必增加芯片地址线的管脚数目。为避免这种情况，采取的办法是分时传送地址码
    - 若地址总线宽度为10位，先传送地址码A0～A9，由行选通信号RAS打入到行地址锁存器；然后传送地址码A10～A19，由列选通信号CAS打入到列地址锁存器。
    - 芯片内部两部分合起来，地址线宽度达20位，存储容量为1M×4位。
- **增加了刷新计数器和相应的控制电路**
  - DRAM读出后必须刷新，而未读写的存储元也要定期刷新，且按行刷新，所以刷新计数器的长度等于行地址锁存器。
  - 刷新操作与读/写操作是交替进行的，所以通过2选1多路开关来提供刷新行地址或正常读/写的行地址

DRAM存储器读/写周期

- 读周期、写周期的定义是从行选通信号RAS下降沿开始，到下一个RAS信号的下降沿为止的时间，也就是连续两个读周期的时间间隔
- 通常为控制方便，读周期和写周期时间相等



DRAM存储器

- 刷新周期
  - DRAM存储位元是基于电容器上的电荷量存储，这个电荷量随着时间和温度而减少，因此必须定期地刷新，以保持它们原来记忆的正确信息

DRAM存储器刷新周期

- 设DRAM中电容的电荷每2ms就会丢失，所以2ms内必须对其补充
  - 补充电荷是按行来进行的
  - 为了**全部内存**都能保住电荷，必须对**所有的行**都得补充
- 假设刷新1行的时间为0.5μs
  - 刷新时间是等于存取周期的
  - 因为刷新的过程与一次存取相同，只是没有在总线上输入输出。
  - 存取周期＞真正用于存取的时间，因为存取周期内、存取操作结束后仍然需要一些时间来更改状态
- 并假设按存储单元(1B/单元)分为64行64列

（64×64个单元×1B/单元 = 2^12个单元×1B/单元 = 4KB内存）

DRAM存储器三种刷新方式

- **集中刷新**
  - 快到2ms的时候，停止一切对内存的读取操作，使用0.5μs×64对64行依次刷新，这将占用32μs
  - 在这32μs中，内存只用来刷新，阻塞一切存取操作
- **为什么刷新与存取不能并行？**
  - 因为内存就一套地址译码和片选装置，刷新与存取有相似的过程，它要选中一行——这期间片选线、地址线、地址译码器全被占用着
  - 同理，刷新操作之间也不能并行——意味着一次只能刷一行

DRAM存储器三种刷新方式

- **分散刷新**
  - 在每个存取操作后绑定一个刷新操作。这样存取周期就成了0.5μs + 0.5μs = 1μs。
- **分散刷新延长了存取周期**
  - 但是由于与存取操作绑定，就不需要专门给出一段时间来刷新
  - 这样，每有64个读取操作，就会把0-63行（共计64行）全部刷新一遍。又因为刷新是不间地断循环着的——循环对64行依次刷新，所以对于同一行，每64次读取就会轮到其被刷新——它的刷新周期是1μs × 64 = 64μs ＜2ms ，在2ms丢失电荷前就会及时补充。

DRAM存储器三种刷新方式

- **异步刷新**
  - 分散刷新的刷新周期64μs，不需要这么频繁，异步刷新就是恰好卡在2ms这个时间点上
- **对于每行以2ms为刷新周期足够了**
  - 刷新循环到它需要64刷新次操作， 2ms ÷ 64 作为“每次刷新的周期”，过64次刚好保证每行的刷新周期为2ms。
  - 刷新操作周期为2ms ÷ 64 。但是这个时间并不是绑定在存取周期内，所以仍然是拒绝存取的死时间。但是它已经很小了，所以这种刷新策略非常可行。
- **注意每次刷新周期与特定行的刷新周期的不同**
  - 每次刷新间隔指对于内存来说它隔多长时间就进行一次刷新操作，轮着刷新时，刷新的行是上一次刷新的行的下一行——是不同的两行，但对于全局内存来说确实是两次刷新操作间隔。**特定行的刷新周期：**下一次对它进行刷新的间隔，期间要经过64次内存刷新周期才又轮得到它。

练习

- **一个1Kx4位的DRAM芯片，若其内部结构排列成64x64形式，且存取周期为0.1us**
  - 若采用分散刷新和集中刷新相结合的方式，即用异步刷新的方法，刷新的信号周期应取多少？
  - 若采用集中刷新，则对该存储芯片刷新一遍需要多少时间？死时间率是多少？

动态 RAM 和静态 RAM 的比较

主存

缓存

	DRAM	SRAM
存储原理	电容	触发器
集成度	高	低
芯片引脚	少	多
功耗	小	大
价格	低	高
速度	慢	快
刷新	有	无