第 10 章　存储系统

(一) 存储器的分类

1.按存储介质分类

（1）半导体存储器　TTL、MOS　　易失

（2）磁表面存储器　磁头、载磁体

（3）磁芯存储器　硬磁材料、环状元件　　非易失

（4）光盘存储器　激光、磁光材料

2. 按存取方式分类

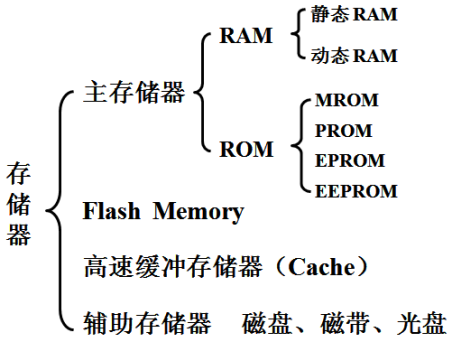
（1）存取时间与物理地址无关（随机访问）

- 随机存储器　在程序的执行过程中可读可写
- 只读存储器　在程序的执行过程中只读

（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

3. 按在计算机中的作用分类



静态 RAM 和动态 RAM 之间的比较。目前，动态 RAM 的应用比静态 RAM 要广泛的多：

①　同样大小的芯片中，动态的 RAM 的集成度远高于静态 RAM，DRAM 的基本单元电路为一个 MOS 管，SRAM 的基本单元电路可为 4~6 个 MOS 管

②　DRAM 行、列按先后顺序输送，减少了芯片引脚，封装尺寸也减少

③ DRAM 的功耗比 SRAM 小

④ DRAM 的价格比 SRAM 的价格便宜

DRAM 也有缺点

① 由于使用动态元件(电容)，因此它的速度比 SRAM 低

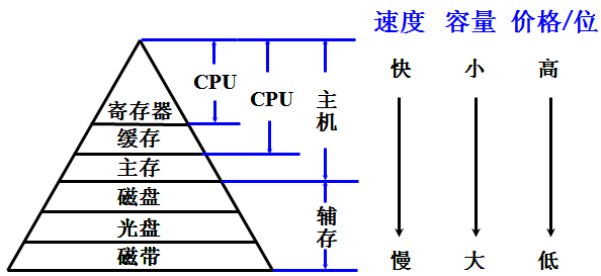
② DRAM 需再生，需配置再生电路，也消耗一部分功率。通常容量不大的 Cache 大多用 SRAM 实现存储器与 CPU 连接

对比项目	SRAM	DRAM
储存信息	触发器	电容
破坏性读出	非	是
需要刷新	非	是
行列地址	同时送	分两次
运行速度	快	慢
集成度	低	高
发热量	大	小
存储成本	高	低

(二) 存储器的层次化结构

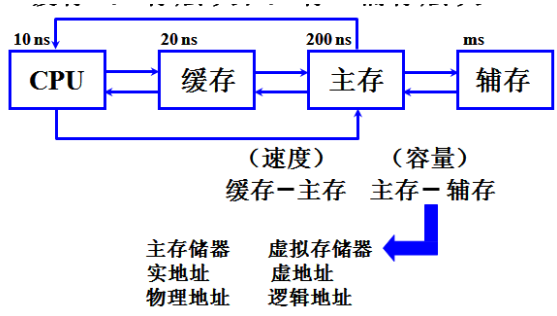
1.存储器三个主要特征的关系

存储器有 3 个重要的指标：速度，容量和每位价格，一般来说，速度越快，位价越高;容量越大，位价越低，容量大，速度就越低。上述三者的关系如图所示。



2.缓存-主存层次和主存-辅存层次

存储系统层次结构主要体现在缓存-主存-辅存这两个存储层次上，如图所示。



（1）缓存-主存层次主要解决 CPU 和主存**速度**不匹配的问题

（2）主存-辅存层次主要解决存储系

统的**容量**问题

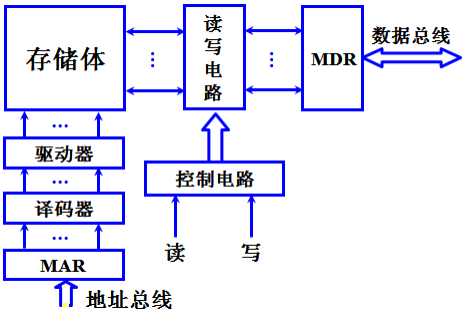
从 CPU 角度来看**缓存-主存层次**的速度接近于缓存，高于主存；其容量和价位却接近于主存，这就从速度和成本的矛盾中获得了理想的解决办法。

主存-辅存层次从整体分析，其速度接近于主存，容量接近于辅存，平均价位也接近于低速的、廉价的存储价位，这又解决了速度、容量、成本这三者之间的矛盾。

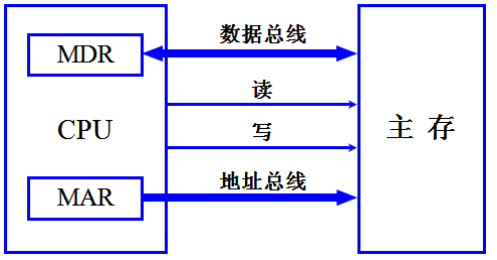
现代计算机系统几乎都具有这两个存储层次，构成了缓存、主存、辅存三级存储系统。

3.主存基本特征

（1）主存的基本组成



（2）主存和 CPU 的关系



（3）主存中存储单元地址的分配

高位字节 地址为字地址　　低位字节 地址为字地址

字地址	字节地址			
0	0	1	2	3
4	4	5	6	7
8	8	9	10	11

字地址	字节地址	
0	1	0
2	3	2
4	5	4

设地址线 24 根　　按 字节 寻址　$2^{24}=16$ M

若字长为 16 位　　按 字 寻址　8 M

若字长为 32 位　　按 字 寻址　4 M

（4）主存的技术指标

① 存储容量　主存 存放二进制代码的总位数

② 存储速度

- 存取时间　存储器的 访问时间

  读出时间 写入时间

- 存取周期　连续两次独立的存储器操作

  （读或写）所需的 最小间隔时间

  读周期 写周期

③ 存储器的带宽　位/秒

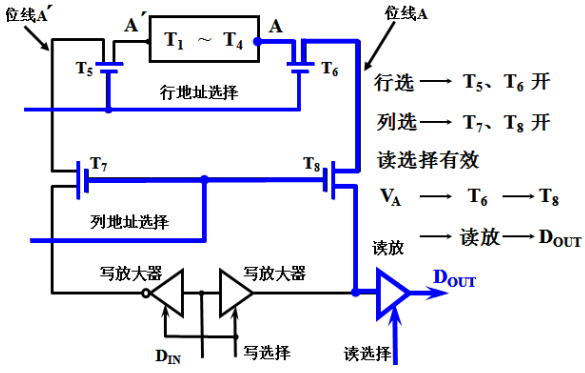
4.半导体存储芯片简介

(三) 半导体随机存取存储器

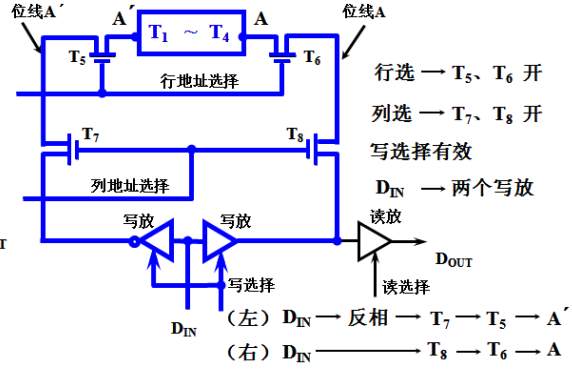
1. SRAM 存储器的工作原理

SRAM 静态存储单元的每个存储位需要四到六个晶体管组成。比较典型的是六管存储单元，即一个存储单元存储一位信息"0"或"1"。静态存储单元保存的信息比较稳定，信息为非破坏性读出，故不需要重写或者刷新操作；另一方面，其结构简单，可靠性高，速度较快，但其占用元件较多，占硅片面积大，且功耗大，所以集成度不高。

① 静态 RAM 基本电路的 读 操作



② 静态 RAM 基本电路的 写 操作

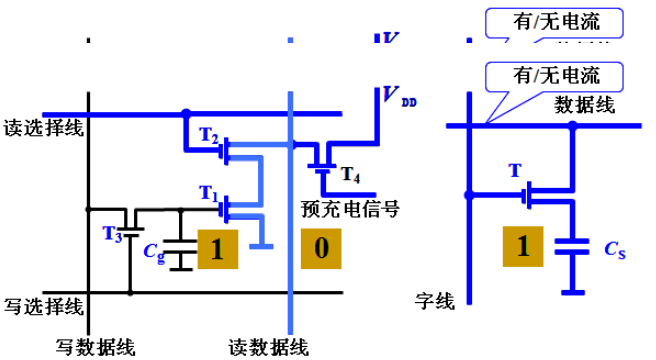


2. DRAM 存储器的工作原理

常见的 DRAM 存储单元有三管式和单管式两种，它们的共特点是靠电容存储电荷的原理来寄存信息。若电容上存有足够的电荷表示“1”，电容上无电荷表示"0"。电容上的电荷一般只能维持 1-2ms，因此即使电源不掉电，电容上的电荷会自动消失。因此，为保证信息的不丢失，必须在 2ms 之内就要对存储单元进行一次恢复操

作，这个过程称为再生或者刷新。与 SRAM 相比，DRAM 具有集成度更高，功耗低等特点，目前被各类计算机广泛使用。

（1）动态 RAM 基本单元电路

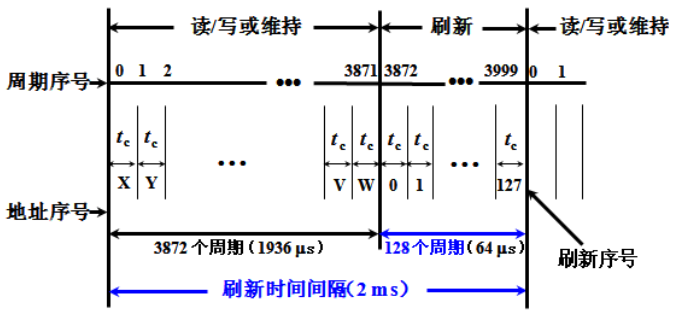


读出与原存信息相反　　读出时数据线有电流为“1”

写入与输入信息相同　　写入时 C_S 充电为“1”放电为“0”

（2）动态 RAM 刷新（刷新与行地址有关）

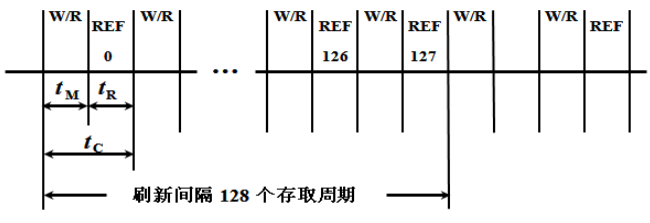
① 集中刷新（存取周期为0.5 μs）以128 × 128 矩阵为例



“死区”为　0.5 μs × 128 = 64 μs

“死时间率”为　128/4 000 × 100% = 3.2%

② 分散刷新（存取周期为1 μs）以 128 × 128 矩阵为例

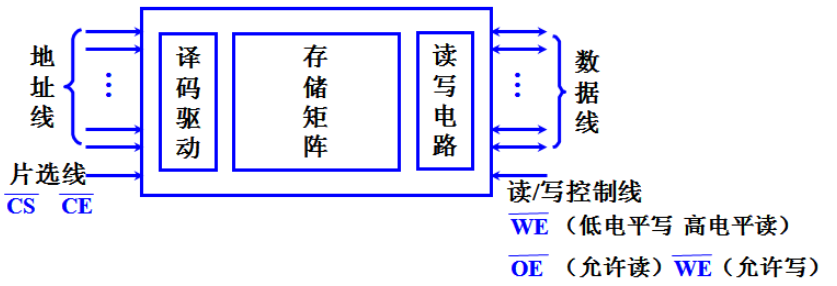


$t_C = t_M + t_R$　　无“死区”

读写　刷新　　(存取周期为 0.5 μs + 0.5 μs)

3. 半导体存储芯片简介

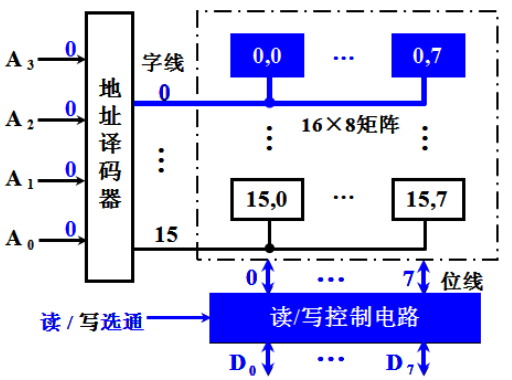
（1）半导体存储芯片的基本结构



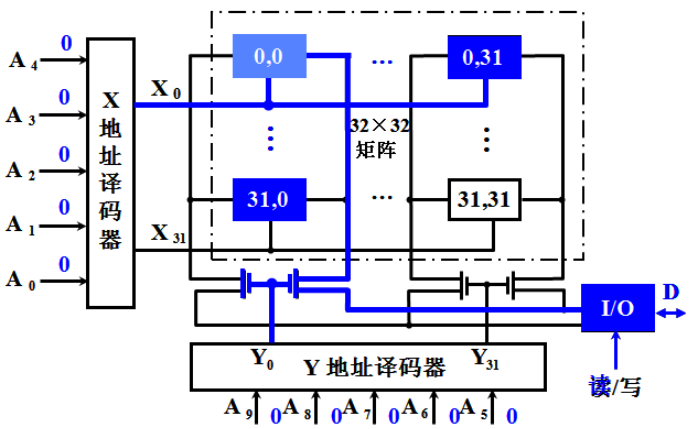
地址线（单向）	数据线（双向）	芯片容量
10	4	1K×4位
14	1	16K×1位
13	8	8K×8位

（2）半导体存储芯片的译码驱动方式

① 一维地址译码（线选法）



② 二维地址译码（重合法）



(四) 半导体随机存取存储器

前面介绍的 DRAM 和 SRAM 均为可任意读 / 写的随机存储器，当掉电时，所存储的内容消失，所以是易失性存储器。只读存储器，即使停电，存储内容也不丢失。根据半导体制造工艺不同，分为 ROM，PROM，EPROM，E2ROM 和 Flash Memory

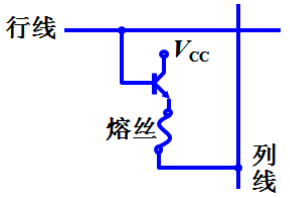
1. 只读存储器(ROM)

掩模式 ROM 由芯片制造商在制造时写入内容，以后只能读而不能再写入。其基本存储原理是以元件的“有 / 无”来表示该存储单元的信息(“1”或“0”)，可以用二极管或晶体管作为元件，显而易见，其存储内容是不会改变的。

- 行列选择线交叉处有 MOS 管为“1”
- 行列选择线交叉处无 MOS 管为“0”

2. 可编程序的只读存储器(PROM)

PROM 可由用户根据自己的需要来确定 ROM 中的内容，常见的熔丝式 PROM 是以熔丝的通和断开来表示所存的信息为“1”或“0”。刚出厂的产品，其熔丝是全部接通的。根据需要断开某些单元的熔丝(写入)。显而易见，断开后的熔丝是不能再接通了，因而一次性写入的存储器。掉电后不会影响其所存储的内容。



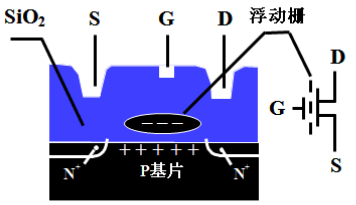
熔丝断　为“0”

熔丝未断　为“1”

3. 可擦可编程序的只读存储器(EPROM)

为了能修改 ROM 中的内容，出现了 EPROM。利用浮动栅 MOS 电路保存信息，信息改写用紫外线照射即可擦除。

(1) N型沟道浮动栅MOS 电路



G 栅极

S 源

D 漏

紫外线全部擦洗

D 端加正电压　形成浮动栅　S 与 D 不导通为“0”

D 端不加正电压　不形成浮动栅　S 与 D 导通为“1”

4. 可电擦可编程序只读存储器(E2PROM)

E2PROM 的编程序原理与 EPROM 相同，擦除原理完全不同，重复改写次数有限制(因氧化层被磨损)，一般 10 万次。

其读写操作可按每个位或每个字节进行，类似 SRAM，但每字节的写入周期要几毫秒，比 SRAM 长得多。E2PROM 每个存储单元采则 2 个晶体管。其栅极氧化层比 EPROM 薄，因此具有电擦除功能。

5. 快除读写存储器(Flash　Memory)

F1ash Memory 是在 EPROM 与 E2PROM 基础上发展起来的，其读写过程和 E2PROM 不同，F1ash Memory 的读写操作一般是以块为单位。

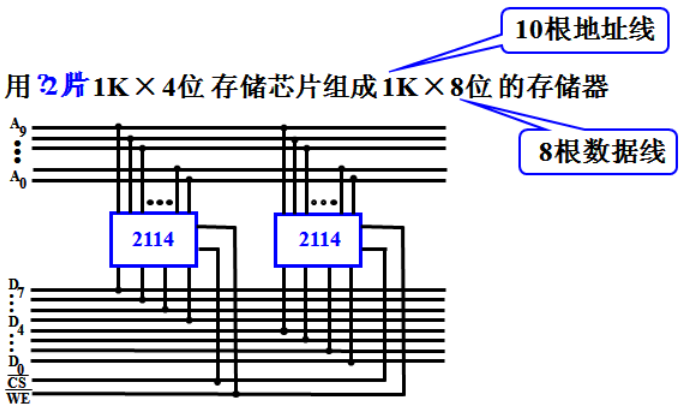
(五) 主存储器与 CPU 的连接

1. 存储器容量的扩展

1 个存储器的芯片的容量是有限的，它在字数或字长方面与实际存储器的要求都有很大差距，所以需要在字向和位向进行扩充才能满足需要。根据存储器所需的存储容量和所提供的芯片的实际容量，可以计算出总的芯片数。一个存储器的容量为 M×N 位，若使用 L×K 位存储器芯片，那么，这个存储器共需要 M/L×N/K 存储器芯片。

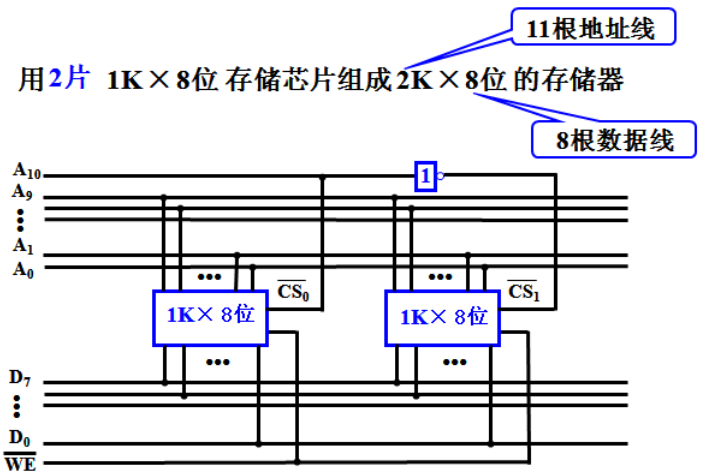
（1）位扩展（增加存储字长）

位扩展指的是用多个存储器器件对字长进行扩充。位扩展的连接方式是将多片存储器的地址，片选已，读写控制端 R/W 可相应并联，数据端分别引出。



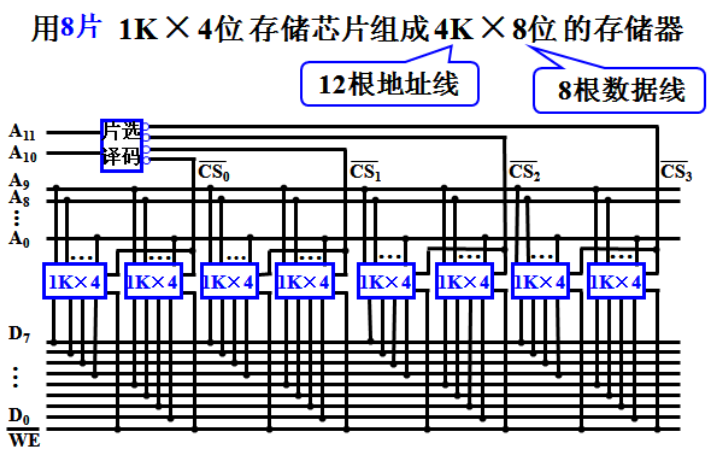
（2）字扩展 （增加存储字的数量）

静态存储器进行字扩展时，将各芯片的地址线，数据线，读写控制线相应并联，而由片选信号来区分各芯片的地址范围。



（3）字位扩展

实际存储器往往需要字向和位向同时扩充。



2. 存储器与 CPU 的连接

（1）合理选择存储芯片

（2）读/写命令线的连接

（3）数据线的连接

（4）地址线的连接

（5）片选线的连接

（6）其他——时序、负载

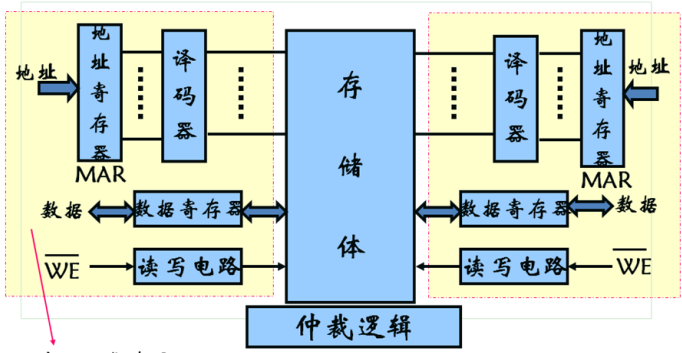
(六) 双口 RAM 和多模块存储器

提高访存速度的措施

- **采用高速器件（双端口存储器）**
- **采用层次存储结构（cache——主存）**
- **调整主存结构**

1.双端口存储器

双端口存储器是一种具有两个单独的读/写端口及控制电路的存储器，通过增加一个读/写端口，双端口存储器扩展了存储器的的信息交换能力。

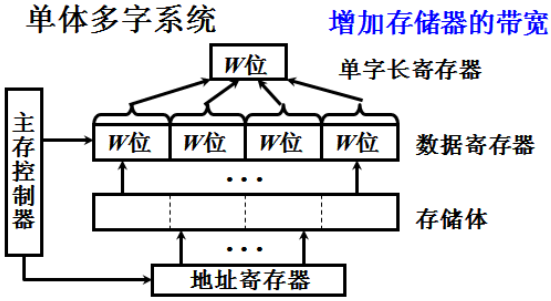


双端口存储器组成框图

2.多模块存储器

为了解决 CPU 与主存储器之间的速度匹配问题，在高速存储器中，普遍采用并行主存系统。即利用类似存储器扩展(位扩展，字扩展，字位扩展)的方法，将 n 个字长为 W 位的存储器并行连接，构建一个更大的存储器。并行主存有单体多字方式，

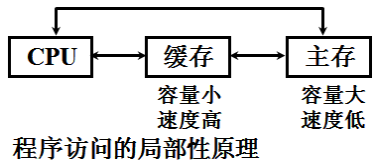
多体并行方式和多体交叉方式。



(七) 高速缓冲存储器 Cache

Cache 实际上，这是来自法文的一个单词，意思是隐蔽之所或藏东西的地方

1. 问题的提出



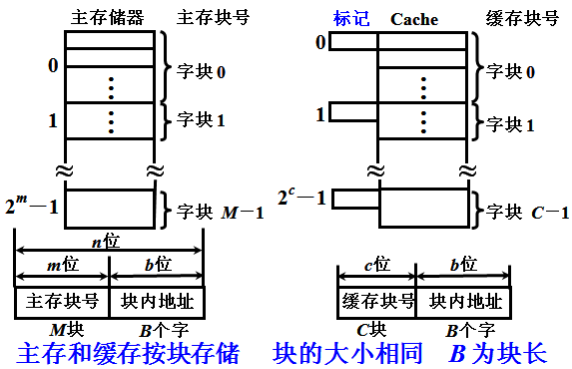
- 避免 CPU“空等”现象
- CPU 和主存(DRAM)的速度差异
- 程序访问的局部性

从大量的统计中得到的一个规律是，程序中对于存储空间 90%的访问局限于存储空间的 10%的区域中，而另外 10%的访问则分布在存储空间的其余 90%的区域中。这就是通常说的局部性原理。访存的局部性规律包括两个方面：

- 时间局部性：如果一个存储项被访问，则可能该项会很快被再次访问。
- 空间局部性：如果一个存储项被访问，则该项及其邻近的项也可能很快被访问。

2. Cache 的基本概念

（1）主存和缓存的编址



（2）命中与未命中

缓存共有 C 块

主存共有 M 块　$M >> C$

命中　主存块调入缓存

主存块与缓存块建立了对应关系

用标记记录与某缓存块建立了对应关系的主存块号

未命中　主存块未调入缓存

主存块与缓存块未建立对应关系

（3）Cache 的命中率

CPU 欲访问的信息在 Cache 中的比率

命中率与 Cache 的容量与块长有关

一般每块可取 4～8 个字

块长取一个存取周期内从主存调出的信息长度

CRAY_1　16体交叉　块长取 16 个存储字

IBM 370/168　4体交叉　块长取 4 个存储字

（64位×4＝256位）

（4）Cache——主存系统的效率

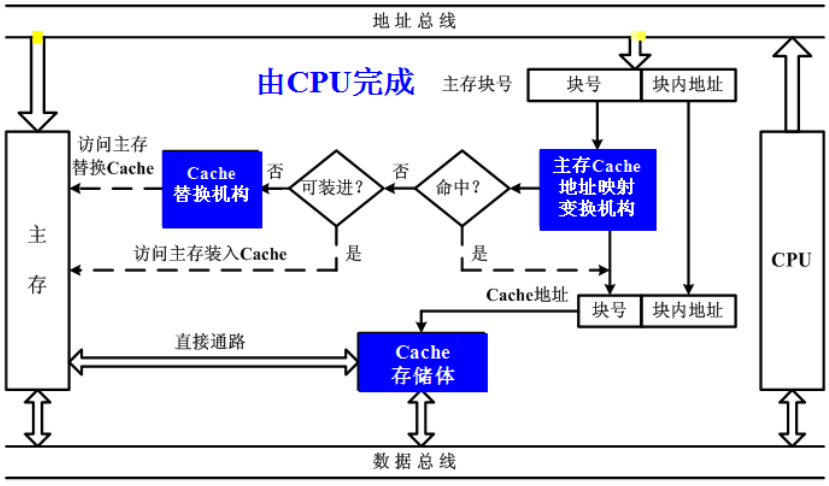
效率 e 与命中率有关

$$e = \frac{\text{访问 Cache 的时间}}{\text{平均访问时间}} \times 100\%$$

设 Cache 命中率为 h，访问 Cache 的时间为 t_c，访问主存的时间为 t_m

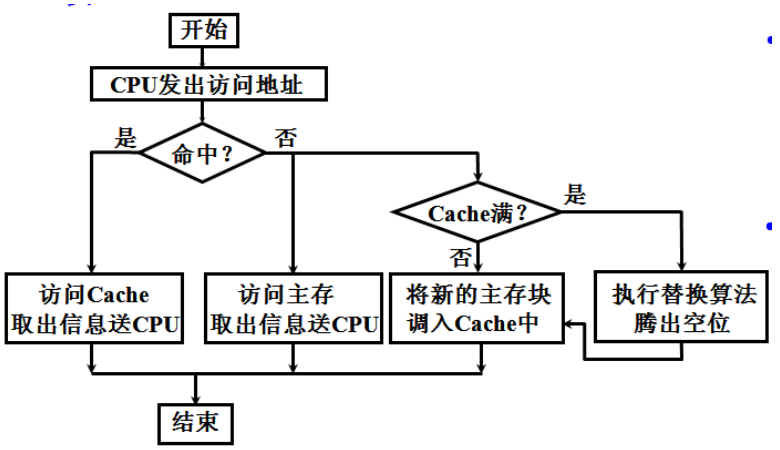
则 $e = \dfrac{t_c}{h \times t_c + (1-h) \times t_m} \times 100\%$

3. Cache 的基本结构



4.Cache 的读写操作

读



写

Cache 和主存的一致性

- **写直达法（Write – through）**

  写操作时数据既写入Cache又写入主存

  写操作时间就是访问主存的时间，读操作时不涉及对主存的写操作，更新策略比较容易实现

- **写回法（Write – back）**

  写操作时只把数据写入 Cache 而不写入主存

  当 Cache 数据被替换出去时才写回主存

  写操作时间就是访问 Cache 的时间，

  读操作 Cache 失效发生数据替换时，

  被替换的块需写回主存，增加了 Cache 的复杂性

5.Cache 的改进

（1）增加 Cache 的级数

- 片载（片内）Cache
- 片外 cache

（2）统一缓存和分立缓存

- 指令 Cache，数据 Cache
- 与主存结构有关
- 与指令执行的控制方式有关——是否流水

例如：Pentium　　8K 指令 Cache　　8K 数据 Cache

PowerPC620　　32K 指令 Caceh　　32K 数据 Cache

6.Cache 的工作原理

Cache 通常由两部分组成，块表和快速存储器。其工作原理是：处理机按主存地址访问存储器，存储器地址的高段通过主存-Cache 地址映象机构借助查表判定该地址的存储单元是否在 Cache 中，如果在，则 Cache 命中，按 Cache 地址访问 Cache。否则，Cache 不命中，则需要访问主存，并从主存中调入相应数据块到 Cache 中，若 Cache 中已写满，则要按某种算法将 Cache 中的某一块替换出去，并修改有关的地址映象关系。

从这个工作原理我们可以看出，它已经涉及到了两个问题。首先是定位，然后是替换的问题。

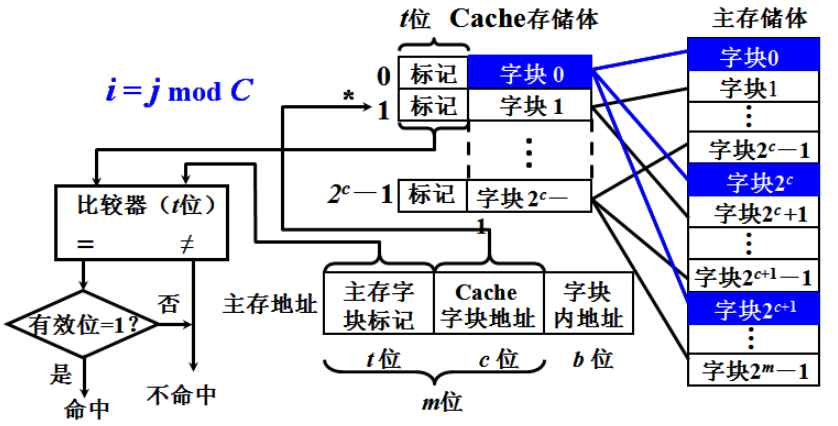
Cache 的存在对程序员是透明的。其地址变换和数据块的替换算法均由硬件实现。通常 Cache 被集成到 CPU 内以提高访问速度。

（1）Cache 和主存之间的映射方式

因为处理机访问都是按主存地址访问的，而 Cache 的空间远小于主存，如何知道这一次的访问内容是不是在 Cache 中，在 Cache 中的哪一个位置呢？这就需要地址映象，即把主存中的地址映射成 Cache 中的地址。让 Cache 中一个存储块(空间)与主存中若干块相对应，如此，访问一个主存地址时，就可以对应地知道在 cache 中哪一个地址了。地址映象的方法有三种：直接映象，全相联映象和组相联映象。

① 直接映射

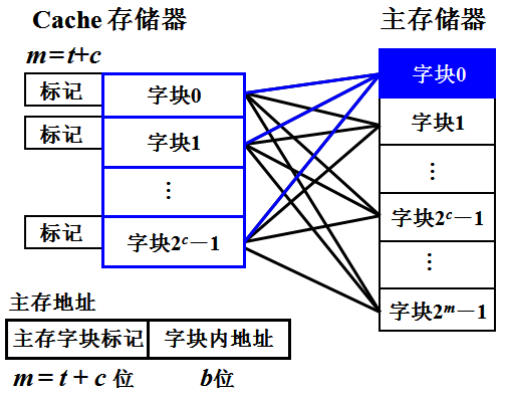
直接映象就是将主存地址映象到 Cache 中的一个指定地址。任何时候，主存中存储单元的数据只能调入到 Cache 中的一个位置，这是固定的，若这个位置已有数据，则产生冲突，原来的块将无条件地被替换出去。



- **每个缓存块 i 可以和若干个主存块对应；**
- **每个主存块 j 只能和一个缓存块对应；**

② 全相联映射

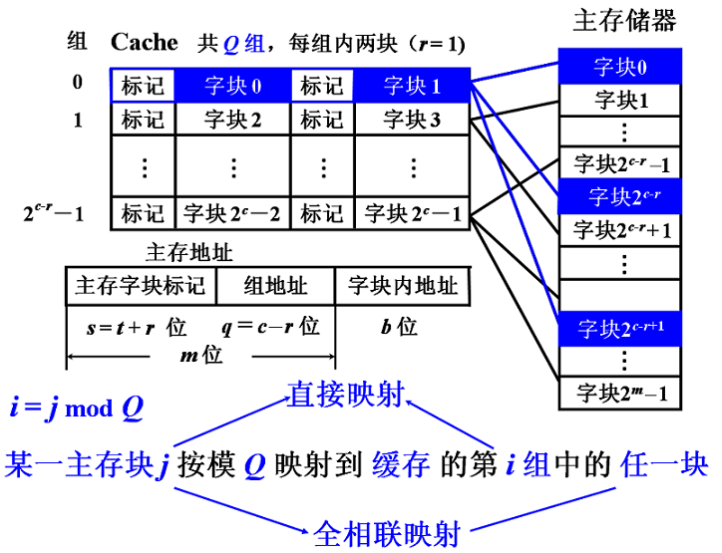
全相联映象就是任何主存地址可映象到任何 Cache 地址的方式。在这种方式下，主存中存储单元的数据可调入到 Cache 中的任意位置。只有在 Cache 中的块全部装满后才会出现块冲突。



- **主存中的任一块可以映射到缓存中的任一块。**

③ 组相联映射

组相联映象指的是将存储空间的页面分成若干组，各组之间的直接映象，而组内各块之间则是全相联映象。



小结：

- **直接映射不灵活，某一主存块只能固定映射到某一缓存块**
- **全相联映射成本高，某一主存块能映射到任一缓存块**
- **组相联映射，某一主存块只能映射到某一缓存组中的任一块**

（2）Cache 中主存块的替换算法

在直接映象方式下，不存在块替换的算法，因为每一块的位置映象是固定的，需要哪一块数据就可直接确定地将该块数据调入上层确定位置。而其他两种映象就存在替换策略的问题，就是要选择替换到哪一个 Cache 块。即替换算法。

	思想	优点	缺点
随机算法 RAND	用软的或硬的随机数产生器产生上层中要被替换的页号	简单，易于实现	没有利用上层存储器使用的"历史信息"，没有反映等程序局部性，命中率低。
先进先出 FIFO	选择最早装入上层的页作为被替换的页	实现方便，利用了主存历史的信息	不能正确反映程序局部性原理，命中率不高，可能出现一种异常现象。
近期最少使用法 LRU	选择近期最少访问的页作为被替换的页	比较正确反映程序局部性，利用访存的历史信息，命中率较高	实现较复杂
优化替换算法 OPT	将未来近期不用的页换出去	命中率最高，可作为衡量其他替换算法的标准	不现实，只是一种理想算法

（3）Cache 写策略

对 Cache 的写操作，情况比读操作要复杂一些。由于写入 Cache 时，并没有写入主存，因此就出现 Cache 和主存数据不一致的情况。如何处理 Cache 和主存不一致的方法就称为更新策略。

更新策略	思想	优点	缺点
写回法	是指在 CPU 执行写操作时，信息只写入 Cache 中，仅当需要替换时，才将改写过的 Cache 块先送回主存(写回)，然后再调块(设置 dirty 位)	有利于省去许多将中间结果写入主存的无谓开销。	需设修改位增加 Cache 的复杂性
全写法(写直达法)	在写操作时，将数据同时写入 Cache 和主存	实现开销小，简单	为了写中间结果浪费了不少时间

另外，当写不命中时(也就是写 Cache 块时，这块早被人替换出去而在 Cache 中找不到时)是不是要把这块再取回 Cache 中，有两个解决方法：

不按写分配法，就是直接写到主存里，不再把该地址对应的块调回 Cache 中。

按写分配法，就是写到主存，而且把这一块从主存中调入到 Cache。

一般写回法用按写分配法，全写法则采用不按写分配。

(八) 虚拟存储器

1. 虚拟存储器的基本概念

虚拟存储器是主存的扩展，虚拟存储器的空间大小取决于计算机的访存能力而不是实际外存的大小，实际存储空间可以小于虚拟地址空间。从程序员的角度看，外存被看作逻辑存储空间，访问的地址是一个逻辑地址(虚地址)，虚拟存储器使存储系统既具有相当于外存的容量又有接近于主存的访问速度。

虚拟存储器的访问也涉及到虚地址与实地址的映象，替换算法等，这与 Cache 中的类似，前面我们讲的地址映象以块为单位，而在虚拟存储器中，地址映象以页为单位。设计虚拟存储系统需考虑的指标是主存空间利用率和主存的命中率。

虚拟存储器与 Cache 存储器的管理方法有许多相同之处，它们都需要地址映象表和地址变换机构。但是二者也是不同的。

虚拟存储器的三种不同管理方式：按存储映象算法，分为段式，页式和段页式等，这些管理方式的基本原理是类似的。

2. 页式虚拟存储器

页式管理：是把虚拟存储空间和实际空间等分成固定大小的页，各虚拟页可装入主存中的不同实际页面位置。页式存储中，处理机逻辑地址由虚页号和页内地址两部分组成，实际地址也分为页号和页内地址两部分，由地址映象机构将虚页号转换成主存的实际页号。

页式管理用一个页表，包括页号，每页在主存中起始位置，装入位等。页表是虚拟页号与物理页号的映射表。页式管理由操作系统进行，对应用程序员的透明的。

3. 段式虚拟存储器

段式管理：把主存按段分配的存储管理方式。它是一种模块化的存储管理方式，每个用户程序模块可分到一个段，该程序模块只能访问分配给该模块的段所对应的主存空间。段长可以任意设定，并可放大和缩小。

系统中通过一个段表指明各段在主存中的位置。段表中包括段名(段号)，段起点，装入位和段长等。段表本身也是一个段。段一般是按程序模块分的。

4. 段页式虚拟存储器

段页式管理：是上述两种方法的结合，它将存储空间按逻辑模块分成段，每段又分成若干个页，访存通过一个段表和若干个页表进行。段的长度必须是页长的整数倍，段的起点必须是某一页的起点。

5. TLB(快表)

在虚拟存储器中进行地址变换时，需要虚页号变换成主存中实页号的内部地址变换，这一般通过查内页表实现。当表中该页对应的装入位为真时，表示该页在主存中，可按主存地址问主存;如果装入位为假时，表示该页不在存储器中，就产生页失效中断，需从外存调入页。

中断处理时先通过外部地址变换，一般通过查外页表，将虚地址变换为外存中的实际地址，到外存中去选页，然后通过 I/0 通道调入内存。当外存页面调入主存中时还存在一个页面替换略的问题。

提高页表的访问速度是提高地址变换速度的关键。因为，每次访存都要读页表，如果页存放在主存中，就意味着访存时间至少是两次访问主存的时间，这样查表的代价大大。只有内部地址变换速度提高到使访问主存的速度接近于不采用虚拟存储器时的访主存速度时，虚拟存储器才能实用。

根据访存的局部性，表内各项的使用的概率不是均匀分布的。在一段时间内，可能只用表中的很少几项，因此应重点提高使用概率高的这部分页表的访问速度，可用快速硬件构成全表小得多的部分表格，而将整个表格放在主存中，这就引出了快表和慢表的概念和技术。这样，虚地址到实地址的变换方法如后图所示。

查表时，根据虚页表同时查找快表和慢表，当在快表中查到该虚页号时，就能很快找到对应的实页号，将其送入主存实地址寄存器，同时使慢表的查找作废，这时主存的访问速度没降低多少。

如果在快表中查不到，则经过一个访主存的时间延迟后，将从慢表中查到的实页送入实地址寄存器，同时将此虚页号和对应的实页号送入快表，这里也涉及到用一个替换算法从快表中替换出一行。

快表的存在对所有的程序员都是透明的。