第5章 chapter 5

存储系统

本章讲解存储器的分类，详细描述存储器的层次结构，包括存储器的3个重要特性的关系，对主存储器、高速缓冲存储器、虚拟存储器和辅助存储器进行重点讲解，阐述程序的局部性原理以及地址转换的方法。


存储系统概述

本章学习目的：

（1）了解主存储器的中心地位、主存储器的分类、主存储器的主要技术指标、主存储器的基本操作。

（2）掌握存储器的组成和读写过程的时序。

（3）掌握半导体存储器的组成与控制。

（4）掌握存储系统的层次结构，分析层次结构的目的和实现方式。

（5）掌握高速缓冲存储器的原理、基本结构和组织。

（6）掌握虚拟存储器的信息传送单位和存储管理以及虚拟存储器工作的全过程。

5.1 存 储 器

存储器是计算机系统的记忆部件，是计算机系统中最重要的部件之一。存储系统由存放程序和数据的各种存储设备、控制部件、管理信息调度的设备（硬件）和算法（软件）组成，执行程序时，计算机需要的指令和数据都来自存储器，程序的执行结果、各种文档、影像资料都保存在存储器中。在计算机开始工作以后，存储器还为其他部件提供信息，同时保存中间结果和最终结果。传统的冯·诺依曼计算机是以运算器为核心的，这也导致运算器成为计算机系统的性能瓶颈。随着计算机的发展，存储器在计算机系统中的地位越来越重要，由于超大规模集成电路的发展制作技术的发展，使CPU的速率高得惊人，而存储器的存取速度与之很难适配，这使计算机系统的运行速度在很大程度上受到存储器速度的制约。

5.1.1 存储器分类


存储器分类

构成存储器的存储介质目前主要是半导体器件和磁性材料。一个双稳态半导体电路、一个CMOS晶体管或磁性材料的存储元件均可以存储一位二进制码。二进制位是存

储器中最小的存储单位，称为存储位元。由若干存储位元组成一个存储单元，然后再由许多存储单元组成一个存储器。

根据存储材料的性能及使用方法不同，存储器有多种分类方法。

（1）按存储介质分类。作为存储介质的基本要求，必须有两个明显区别的物理状态，分别用来表示二进制的 0 和 1。另外，存储器的存取速度又取决于这种物理状态的改变速度。目前使用的存储介质主要是半导体器件、磁性材料和光存储介质。用半导体器件组成的存储器称为半导体存储器。用磁性材料做成的存储器称为磁表面存储器，如磁盘存储器和磁带存储器。光存储器是指只读光盘或者读写光盘。磁盘和光盘的共同特点是存储容量大，存储的信息不易丢失。

（2）按存取方式分类。如果存储器中任何存储单元的内容都能被随机存取，且存取时间和存储单元的物理位置无关，就称为随机存取存储器。如果存储器只能按某种顺序存取，也就是说存取时间和存储单元的物理位置有关，就称为顺序存取存储器。例如，磁带存储器就是顺序存取存储器，它的存取周期较长。磁盘存储器则是半顺序（直接）存取存储器，沿磁道方向顺序存取，沿垂直于半径的方向随机存取。

（3）按读写功能分类。有些半导体存储器在工作过程中只能读出其中存储的内容而不能向其中写入内容，这种半导体存储器称为只读存储器（Read-Only Memory，ROM）。在存储器工作过程中既能读出又能写入的半导体存储器称为读写存储器或随机存取存储器（Random Access Memory，RAM）。RAM 用来存储当前运行的程序和数据，并可以在程序运行过程中反复更改其内容。而 ROM 常用来存储不变或基本不变的程序和数据（如监控程序、引导加载程序及常数表格等）。RAM 可以根据信息存储方法分为静态 RAM（SRAM）和动态 RAM（DRAM）。SRAM 是用半导体管的导通或截止记忆信息的，只要不掉电，SRAM 中存储的信息就不会丢失。而 DRAM 的信息是用电荷存储在电容上的，随着时间的推移，电荷会逐渐漏失，存储信息也会丢失，因此要周期性地对其刷新。根据工艺和特性的不同，ROM 又分为掩膜 ROM、一次可编程 ROM（PROM）和可擦除 PROM（EPROM），后者又分为紫外线擦除 EPROM（UV-EPROM）、电擦除 EPROM（EEPROM 或 E^2PROM）和闪速只读存储器（Flash）。

（4）按信息易失性分类。断电后信息消失的存储器称为易失性存储器。断电后仍能保存信息的存储器称为非易失性存储器。在半导体存储器中，RAM 是易失性存储器，一旦掉电，其中存储信息全部丢失；而 ROM 是非易失性存储器。磁性材料做成的存储器是非易失性存储器。

（5）按与 CPU 的耦合程度分类。根据存储器在计算机系统中所处的位置，可分为内部存储器和外部存储器。内存又可分为主存和高速缓冲存储器。

5.1.2 存储器的编址和端模式

存放一个机器字的存储单元通常称为字存储单元，相应的地址称为字地址；而存放一字节的存储单元，称为字节存储单元，相应的地址称为字节地址。编址方式是存储器地址的组织方式，一般在设计处理器时就已经确定了。如果计算机中编址的最小单位是字存储单元，则该计算机称为按字编址的计算机；如果计算机中编址的最小单位是字节，

则该计算机称为按字节编址的计算机。一个机器字可以包含数字节，所以一个存储单元也可占用数个能够单独编址的字节地址。例如，一个 16 位二进制的字存储单元包含两字节，当采用字节编址方式时，该字占两字节地址。

当一个存储字的字长超过 8 位时，就存在一个存储字内部的多字节排列顺序问题，其排列方式称为端模式。大端(big-endian)模式将一个字的高有效字节放在内存的低地址端，低有效字节放在内存的高地址端；而小端(little-endian)模式则将一个字的低有效字节放在内存的低地址端，高有效字节放在内存的高地址端。如图 5-1 所示，如果一个 32 位数$(0A0B0C0D)_{16}$按照大端模式存放在内存中，则最低地址存放最高有效字节$(0A)_{16}$，最高地址存放最低有效字节$(0D)_{16}$；而按照小端模式存放时，字节顺序刚好相反。

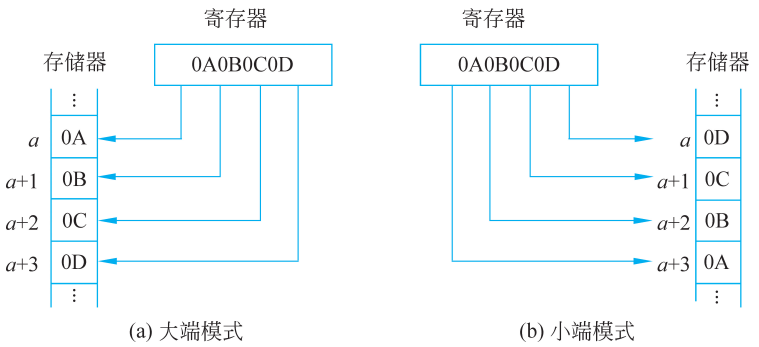


图 5-1 存储器的端模式示例

5.2 存储系统概述


存储系统的层次结构

5.2.1 存储系统的层次结构

在冯·诺依曼体系结构中，存储器是计算机系统的五大组成部件之一。早期的计算机系统只有单一的存储器，用于存放为数不多的数据和指令。但是，随着软件复杂度的提高以及多媒体技术和网络技术的普及，对存储器容量的要求也不断提高。而微电子技术的发展又为大幅度提升存储器的存储密度提供了可能性，这反过来又促使存储器的容量进一步提升。

表 5-1 所示为 1980—2010 年 CPU 和存储器的主要性能参数。表中几种存储器的特点是：SRAM 容量最小，速度最快，价格最高；磁盘容量最大，速度最慢，价格最低；而 DRAM 在 1980 年时能够跟上 CPU 的速度，随着时间的推移，DRAM 的读写速度虽然在提高，但是已逐渐跟不上 CPU 的速度。

在通常情况下，存储器考虑的只是存储容量，但如果存储器的读写速度跟不上 CPU 的速度，计算机的运行效率就会大大降低。图 5-2 展示了有无 Cache 对指令执行时间的影响。

表 5-1　CPU 和存储器的主要性能参数

CPU/存储器	性 能 参 数	1980 年	1990 年	2000 年	2010 年
CPU	名称	8080	80386	Pentium Ⅱ	Core i7
	时钟频率/MHz	1	20	600	2500
	时钟周期/ns	1000	50	1.6	0.4
	核数	1	1	1	4
	有效时钟周期/ns	1000	50	1.6	0.1
SRAM	每兆字节价格/美元	19 200	320	100	60
	存取时间/ns	300	35	3	1.5
DRAM	每兆字节价格/美元	8000	100	1	0.06
	存取时间/ns	375	100	60	40
	典型大小	64KB	4MB	64MB	8000MB
磁盘	每兆字节价格/美元	500	8	0.01	0.0003
	存取时间/ns	87	28	8	3
	典型大小	1MB	160MB	20B	1TB

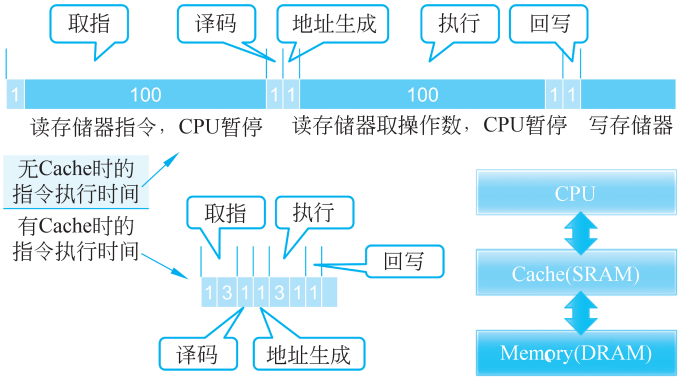


图 5-2　无 Cache 时和有 Cache 时的指令执行时间对比

图 5-1 中的数字表示占用的时间，单位为一个时钟周期。可以看出，当无 Cache 时，因为 DRAM 的读写速度跟不上 CPU 的速度，所以指令的执行时间很长；相反，当存在 Cache 时，因为 Cache 的读写速度远超 DRAM 的读写速度，所以指令的执行时间大为缩短。然而 Cache 高昂的价格导致其存储容量有限，无法满足现代计算机的需求。

由于存储器的价格相对较高，而且在整机成本中占有较大的比例，因而从性能价格比的角度不能通过简单配置更大容量的存储器来满足用户的需求。为此，必须使用某种策略解决成本和性能之间的矛盾。这一策略就是存储器分层，即利用不同容量、成本、功耗和速度的多种存储器构成有机结合的多级存储系统。

存储系统是指把两种或者两种以上不同存储容量、不同存取速度、不同价格的存储

器组成层次结构，并通过管理软件和辅助硬件将不同性能的存储器组合成有机的整体，又称为计算机的存储层次或存储体系。现代计算机采用的典型存储结构有 Cache—主存和主存—辅存两种。

1. Cache—主存存储结构

Cache—主存存储结构如图 5-3 所示。其主要目的是解决 CPU 和主存速度不匹配的问题。虽然 Cache 价格昂贵，导致一般情况下无法使用 Cache 构成大容量的存储空间，但 Cache 的速度比主存的速度高，只要将 CPU 近期要用的信息调入 Cache，CPU 便可以直接从 Cache 中获取信息，从而提高访存速度。但由于 Cache 的容量小，因此需要不断地将主存的内容调入 Cache，使 Cache 中原来的信息被替换。主存和 Cache 之间的数据调动是由硬件自动完成的，对程序员是透明的。

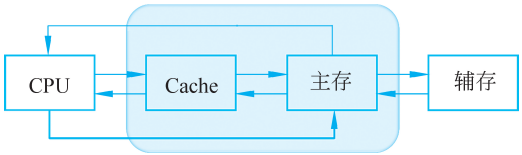


图 5-3 Cache—主存存储结构

2. 主存—辅存存储结构

主存—辅存存储结构如图 5-4 所示。其主要目的是解决存储系统的容量问题。虽然辅存的速度比主存的低，而且不能和 CPU 直接交换信息，但它的容量比主存大得多，所以大量暂时未用到的信息可以保存到辅存中。而当 CPU 需要这些信息时，再将辅存中的内容调入主存，供 CPU 直接访问。主存和辅存之间的数据调动是由硬件和操作系统共同完成的。

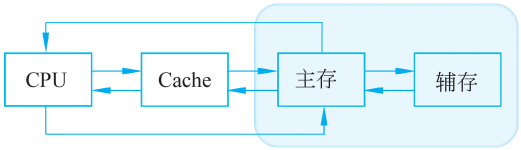


图 5-4 主存—辅存存储结构

从 CPU 角度来看，Cache—主存这一层次的速度接近 Cache，高于主存，其容量和位价却接近主存。这就从速度和成本的矛盾中获得了理想的解决办法；主存—辅存这一层次，从整体分析，其速度接近主存，容量接近辅存，平均位价也接近低速、廉价的辅存，这又解决了速度、容量、成本这三者的矛盾。现代计算机的存储系统几乎都具有这两个存储层次，构成了 Cache、主存、辅存三级存储系统，并且还有很多细分类型，可以用图 5-5 展示。

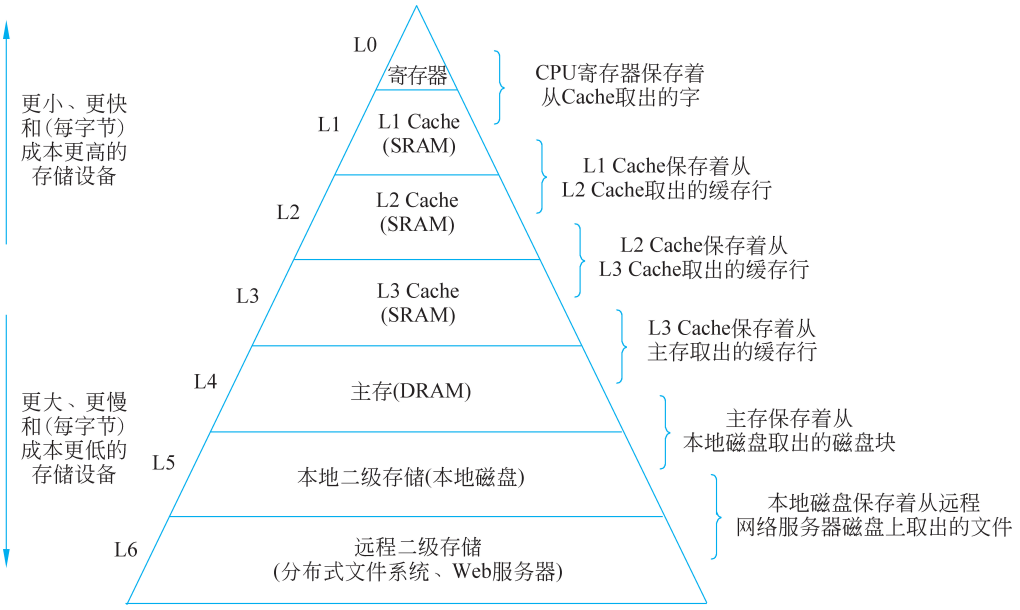


图 5-5　现代存储器层次结构

5.2.2　程序的局部性原理

存储系统的建立依托于程序的局部性原理。统计表明，无论是访问指令还是存取数据，在一个较短的时间间隔内，程序所访问的存储器地址在很大比例上集中在存储器地址空间的很小范围内。这种在某一段时间内频繁访问某一局部的存储器地址空间，而对此范围以外的地址空间则很少访问的现象称为程序的局部性原理。

程序的局部性可以从两个角度分析：

(1) 时间局部性，即最近被访问的信息很可能还要被访问。

(2) 空间局部性，即最近被访问的信息邻近地址的信息也可能被访问。

以下面的程序为例：

```
for (i=0; i<1000; i++)
    for (j=0; j<200; j++)
        sum += a[i][j];
```

变量 sum 在 for 循环中会不停被修改，即意味着存储 sum 这个变量的存储单元会在短时间内被反复访问，这体现了程序的时间局部性；而数组 a 在 for 循环中会以数组的首地址为基础顺序访问后面的地址，这体现了程序的空间局部性。

根据程序的时间局部性，存储系统将近期频繁被访问的主存单元的数据放入 Cache 中，使得 CPU 频繁访问的数据都能在 Cache 中找到；根据程序的空间局部性，存储系统从主存中取回待访问数据时，会同时取回与其位置相邻的主存单元的数据放入 Cache 中，使得 CPU 访问连续的数据时能够在 Cache 中找到后续的数据。

5.3 静态随机存取存储器

静态随机存取存储器

5.3.1 SRAM 的概念

随机存取存储器(RAM)是一种可读写存储器,其特点是存储器的任何一个存储单元的内容都可以随机存取,而且存取时间与存储单元的物理位置无关。计算机系统中的主存都采用随机存储器。基于存储信息原理的不同,随机存取存储器又分为静态随机存取存储器和动态随机存取存储器。

静态随机存储器(Static RAM,SRAM)利用双稳态触发器的两个稳定状态保存信息,只要不断电,SRAM 中的信息就不会丢失,因为其不需要进行动态刷新。图 5-6 为基本的静态存储元阵列。SRAM 用锁存器(触发器)作为存储位元。只要直流供电电源一直加在这个记忆电路上,它就无限期地保持记忆的 1 状态或 0 状态;如果电源断电,则存储的数据(1 或 O)就会丢失。

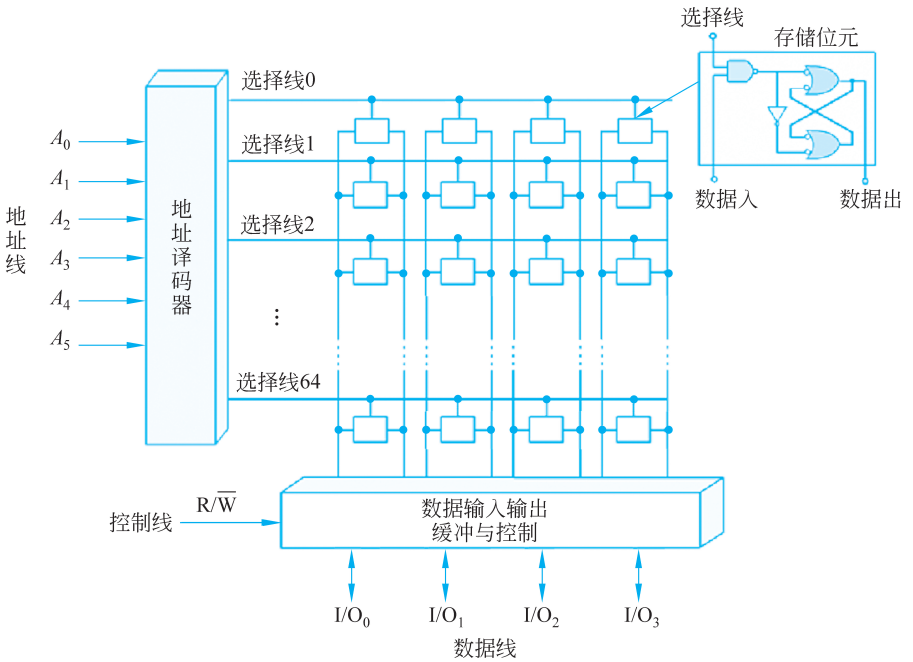


图 5-6 基本的静态存储位元阵列

任何一个 SRAM 都有 3 组信号线与外部打交道:

(1) 地址线。在图 5-6 中有 6 条,即 $A_0 \sim A_5$,它指定了存储器的容量是 $2^6 = 64$ 个存储单元。

(2) 数据线。在图 5-6 中有 4 条,即 $I/O_0 \sim I/O_3$,说明存储器的字长是 4 位,因此存储位位元的总数是 $64 \times 4 = 256$ 个。

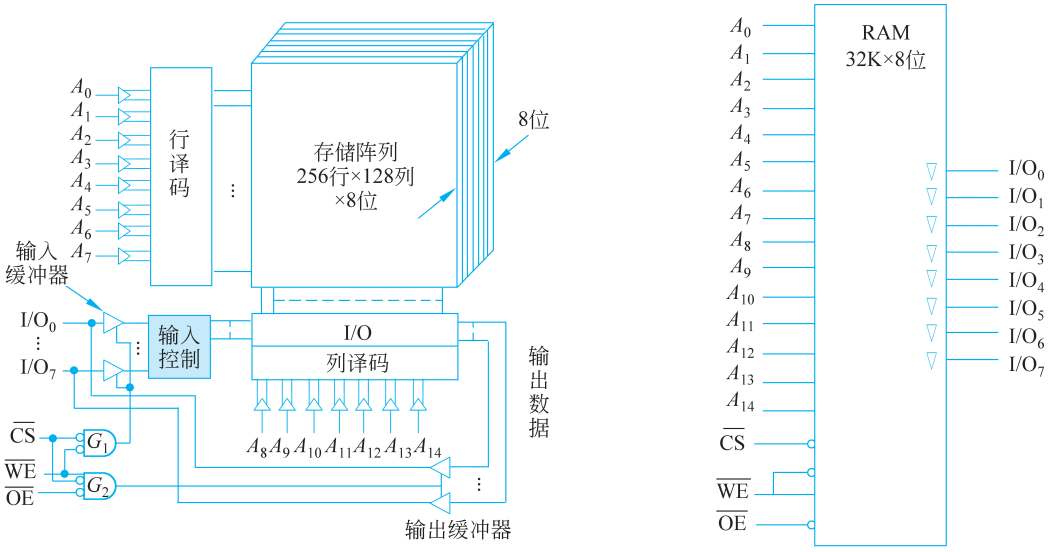
（3）控制线。在图 5-6 中是 R/$\overline{W}$ 控制线，它指定了对存储器进行读操作（R/$\overline{W}$ 高电平）还是进行写操作（R/$\overline{W}$ 低电平）。注意，读写操作不会同时发生。

地址译码器的输出有 64 条选择线，称为行线，其作用是打开每个存储位元的输入与非门。当外部输入数据为 1 时，锁存器便记忆了 1；当外部输入数据为 0 时，锁存器便记忆了 0。

5.3.2 基本的 SRAM 逻辑结构

目前的 SRAM 芯片采用双译码方式，以便组织更大的存储容量。这种译码方式的实质是采用了二级译码：将地址分成 x 方向、y 方向两部分，第一级进行 x 方向（行译码）和 y 方向（列译码）的独立译码，然后在存储位元阵列中完成第二级的交叉译码。而数据宽度为 1 位、4 位、8 位，甚至有更多的字节。

图 5-7(a)表示存储容量为 32K×8 位 SRAM 的结构图。它的地址线共 15 条。其中，x 方向 8 条（A_0～A_7），经行译码输出 256 行；y 方向 7 条（A_8～A_{14}），经列译码输出 128 列。存储位元阵列为三维结构，即 256 行×128 列×8 位。双向数据线有 8 条，即 I/O_0～I/O_7。向 SRAM 写入时，8 个输入缓冲器被打开，而 8 个输出缓冲器被关闭，因而 8 条双向数据线上的数据写入存储位元阵列中；从 SRAM 读出时，8 个输出缓冲器被打开，8 个输入缓冲器被关闭，读出的数据送到 8 条双向数据线上。



(a) 结构图　　(b) 逻辑图

图 5-7　32K×8 位 SRAM 的结构图和逻辑图

控制信号包括$\overline{CS}$、$\overline{OE}$和$\overline{WE}$。$\overline{CS}$是片选信号，$\overline{CS}$有效（低电平）时，门 G_1、G_2 均被打开。$\overline{OE}$为读出使能信号，$\overline{OE}$有效（低电平）时，门 G_2 开启。当写命令信号$\overline{WE}$=1（高电平）时，门 G_1 关闭，存储器进行读操作；写操作时，$\overline{WE}$=0，门 G_1 开启，门 G_2 关闭。注

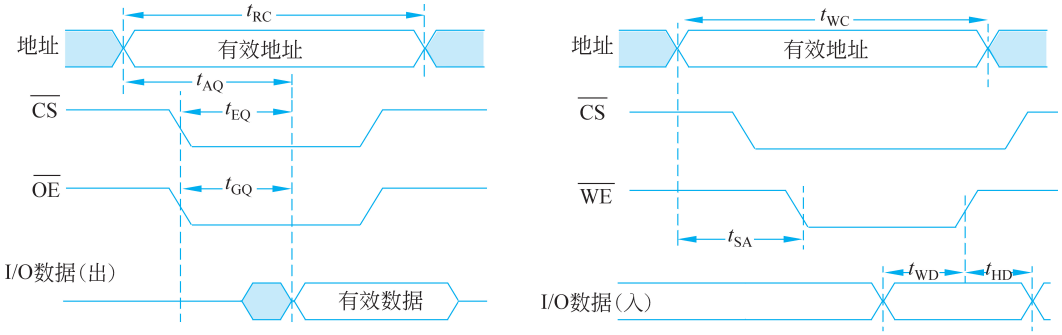
意，门 G_1 和 G_2 是互锁的，一个开启时另一个必定关闭，这样就保证了读时不写、写时不读。图 5-7(b)为 32K×8 位 SRAM 的逻辑图。

5.3.3 SRAM 读写时序

如图 5-8 所示，SRAM 读写时序精确地反映了 SRAM 工作的时间关系。把握住地址线、控制线、数据线 3 组信号线何时有效，就能很容易看懂这个时序。

在读周期中，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$\overline{CS}$和读出使能信号$\overline{OE}$也必须有效(由高电平变为低电平)。从地址有效开始经 t_{AQ}(读出)时间，数据总线上出现了有效的读出数据。此后$\overline{CS}$、$\overline{OE}$信号恢复高电平，经 t_{RC}以后才允许地址总线发生改变。t_{RC}时间即为读周期时间。

在写周期中，也是地址线先有效，接着片选信号$\overline{CS}$有效，写命令$\overline{WE}$有效(低电平)，此时数据总线上必须置写入数据，在 t_{WD}时间段将数据写入存储器。然后撤销写命令$\overline{WE}$和$\overline{CS}$。为了写入可靠，数据总线的写入数据要有维持时间 t_{HD}，$\overline{CS}$的维持时间也比读周期长。t_{WC}时间称为写周期时间。为了控制方便，一般取 $t_{RC}=t_{WC}$，通常称为存取周期。



(a) 读周期($\overline{WE}$高)　　(b) 写周期($\overline{WE}$低)

图 5-8　SRAM 读写时序

5.4 动态随机存取存储器


动态随机存取存储器

5.4.1 DRAM 的概念

动态随机存取存储器(Dynamic RAM，DRAM)简化了每个存储位元的结构，因而 DRAM 的存储密度很高，通常用作计算机的主存储器。

图 5-9 为由一个 MOS 晶体管和电容器组成的单管 DRAM 存储位元。其中 MOS 管作为开关使用，而存储的信息 1 或 0 则是由电容器上的电荷量体现的——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。

写 1 到存储位元时，输出缓冲器关闭，刷新缓冲器关闭，输入缓冲器打开(R/$\overline{W}$ 为低电平)，输入数据 $D_{IN}=1$ 送到存储元位线上，行选线为高电平，打开 MOS 管，于是位线上

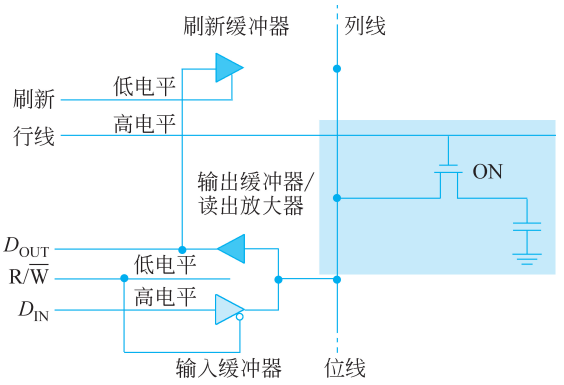


图 5-9　单管 DRAM 存储位元的工作原理

的高电平给电容器充电，表示存储了 1。写 0 到存储位元时，输出缓冲器和刷新缓冲器关闭，输入缓冲器打开，输入数据 $D_{IN}=0$ 送到存储元位线上，行选线为高电平，打开 MOS 管，于是电容器上的电荷通过 MOS 管和位线放电，表示存储了 0。

从存储位元读出时，输入缓冲器和刷新缓冲器关闭，输出缓冲器/读出放大器打开（R/$\overline{W}$ 为高电平）。行选线为高电平，打开 MOS 管，若当前存储的信息为 1，则电容器上存储的 1 送到位线上，通过输出缓冲器/读出放大器发送到 D_{OUT}，即 $D_{OUT}=1$。

读出过程破坏了电容器上存储的信息，所以要把信息重新写入，即刷新。读出的过程中可以完成刷新。读出 1 后，输入缓冲器关闭，刷新缓冲器打开，输出缓冲器/读出放大器打开，读出的数据 $D_{OUT}=1$ 又经刷新缓冲器送到位线上，再经 MOS 管写到电容器上，存储位元重写 1。

注意，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。

与 SRAM 相比，DRAM 的存储位元所需元件更少，所以存储密度更高。但是 DRAM 的附属电路比较复杂，访问时需要额外的电路和操作支持。

5.4.2　DRAM 芯片的逻辑结构

图 5-10(a)给出了 1M×4 位 DRAM 芯片的外部引脚。图 5-10(b)是该芯片的结构图。

与 SRAM 不同的是，图 5-10 中增加了行地址锁存器和列地址锁存器。由于 DRAM 容量很大，地址线的数目相当多，为减少芯片引脚的数量，将地址分为行、列两部分分时传送。存储容量为 2^{10} 字，共需 20 位地址线。此芯片地址引脚的数量为 10 位。先传送行地址码 $A_0 \sim A_9$，由行选通信号$\overline{RAS}$打入行地址锁存器；然后传送列地址码 $A_{10} \sim A_{19}$，由列选通信号$\overline{CAS}$打入列地址锁存器。片选信号的功能也由增加的$\overline{RAS}$和$\overline{CAS}$信号实现。

5.4.3　DRAM 读写时序

图 5-11(a)为 DRAM 的读周期。当地址线上行地址有效后，用行选通信号$\overline{RAS}$打入