第3章 存储器及接口设计

3.1 概　　述

存储器(Memory)是计算机系统中用来存放程序和数据的硬件设备。现代计算机系统中的全部信息,比如输入的原始数据、计算机程序、中间运行结果和最终运行结果都保存在存储器中,是计算机系统不可或缺的功能部件。

计算机系统的存储器可分为两大类,即主存储器(内存)和辅助存储器(外存),主存储器的速度远快于辅助存储器。主存通过总线与微处理器连接,用来存放正在执行的程序和正在处理的数据;辅存通过接口电路和主机连接,用来存放暂时没有被执行的程序和还没有被处理的数据。CPU 可以直接访问内存,但访问外存时必须先把要访问的信息从外存读到内存。

内存的速度虽然比外存快,但远远跟不上 CPU 的速度。为弥补主存和 CPU 之间的速度差异,可以在 CPU 和内存之间增加高速缓冲存储器(Cache)。

另外,为了提高运算速度,在 CPU 内部还设置了少量容量相对较小的高速存储部件,即寄存器(Register)。这些寄存器在程序运行过程中可用来暂存指令、数据和地址等信息。这样,在现代计算机系统中就形成了辅存、主存、高速缓存、寄存器这样的 4 层存储结构。存储器分类层次结构如图 3.1 所示。

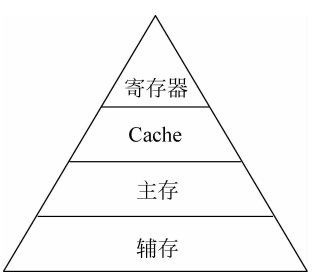


图 3.1　存储器分类层次结构

各级存储器参数比较如表 3.1 所示。

表 3.1　各级存储器参数比较

名　　称		位　　置	速　　度	容　　量	价　格
寄存器		CPU 内部	与 CPU 同速	几十字节或字	高
Cache	L1	CPU 内部	与 CPU 同速	1～64KB	高
	L2	CPU 内部或外部	与 CPU 同速或稍慢	≤4MB	高
主存		主板内	比 Cache 慢,比辅存快	比 L2 大,比辅存小	较高
磁盘		主板外,微机系统内	慢	理论上无限大	低
移动存储器		微机系统外	慢	理论上无限大	低

需要注意的是,一个存储系统的容量往往比较大,单一存储芯片的存储单元的个数及数据宽度往往不能满足存储系统的要求,这时还要采用存储器接口技术来对所有存储器芯片

的连接和控制进行管理。

本章主要介绍主存储器技术、高速缓存技术和存储器扩展技术。

3.2 半导体存储器分类及性能指标

3.2.1 半导体存储器的分类

目前，辅助存储器大多采用磁性材料作为存储介质，虽然价格较低、容量很大，但速度相对较慢，因此对于主存来说，一般都会采用存取速度更快的半导体器件作为存储介质。

主存储器种类很多，分类也多种多样。按设计电路可分为双极型和 MOS 型存储器；根据信息的存储原理来分有静态和动态存储器；从信息的传送方式来分有并行和串行存储器。而最常见的分类方法是按信息的存取方式进行划分，即随机存取存储器 RAM 和只读存储器 ROM。

1. RAM 的分类

RAM 是指当计算机工作时，可随机地对存储器进行读出和写入操作。

RAM 又可分为双极型和 MOS 型两大类。双极型存取速度快，但功耗大、集成度低、单片容量小、成本高，因此一般都作为容量较小但速度要求很高的高速缓冲存储器；与双极型 RAM 相比，MOS 型 RAM 具有功耗低、集成度高、单片容量大、成本低等特点，但存取速度相对较慢。MOS 型 RAM 又可以分为静态 RAM(SRAM)和动态 RAM(DRAM)两种，其存储原理将在后面小节中进行详细讲解。

2. ROM 的分类

ROM 是指存储器内容是事先写入的，计算机正常工作时只能读出不能写入。一旦写入信息，就不能轻易改变，掉电后信息也不会丢失。

按制作工艺和使用特性，ROM 可分为 4 种：掩膜式只读存储器(Mask ROM)、可编程只读存储器(Programmable ROM)、紫外线可擦除可编程只读存储器(Erasable Programmable ROM)和电可擦除可编程只读存储器(Electrically Erasable Programmable ROM)。

半导体存储器的分类如图 3.2 所示。

3.2.2 半导体存储器的性能指标

随着现代计算机系统的发展，存储系统的性能从某种意义上来说决定着整个计算机系统的性能。主要原因是建立在存储程序概念基础之上的现代计算机系统在运行过程中的访存操作约占中央处理器(CPU)时间的 70%左右。因此，存储器组织与管理方法的好坏会直接影响到整机效率。存储器的容量、速度、带宽等性能越来越成为影响计算机系统性能的瓶颈。

1. 存储容量

在微型计算机系统中，存储器以一个字节作为一个存储单元。存储器可以容纳的存储单元总数称为该存储器的存储容量。一般情况下，存储容量的单位为 B(BYTE，字节)，1B=8b。但由于计算机的存储容量都很大，因此常以 KB(2^{10}字节)、MB(2^{20}字节)、GB(2^{30}字

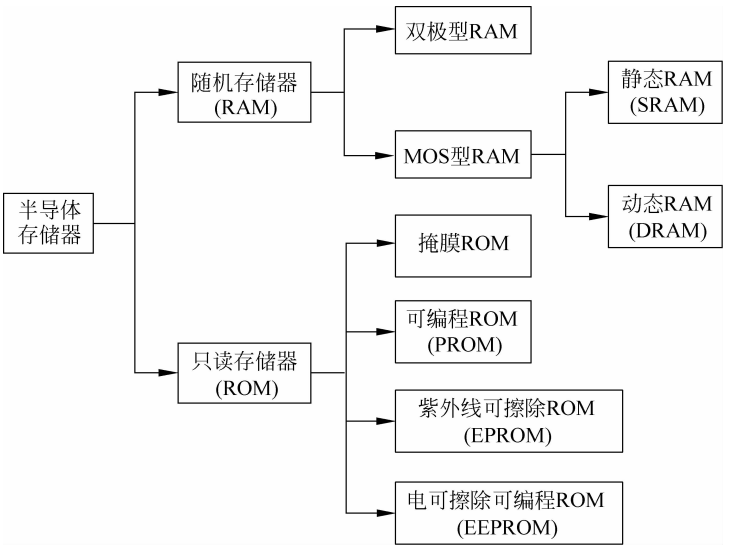


图 3.2　半导体存储器分类

节)、TB(2^{40}字节)、PB(2^{50}字节)、EB(2^{60}字节)、YB(2^{70}字节)、NB(2^{80}字节)、DB(2^{90}字节)等为单位。主存的容量越大,存储的信息量也就越大,计算机运行的速度一般也就越快。

存储一位二进制信息的单元称为一个基本存储元。对于 1MB 的存储器,其内部有 2^{20} 个存储单元,$2^{20}\times 8$ 个基本存储元。在标定某存储器芯片的容量时,经常同时标出存储单元的个数和每个存储单元的位数,即

存储器芯片容量 = 存储单元个数 × 存储单元位数

例如,Intel 62256 芯片的容量为 $2^{15}\times 8$b,表示该种芯片的存储单元个数为 2^{15},每个存储单元 8 位,总容量为 32KB。

2. 速度

计算机运行过程中要不断与主存进行信息交换,因此主存的速度是衡量整个计算机系统运行速度的一项重要指标。存储器芯片的速度通常用访问时间或存取周期来衡量。

存取周期(memory cycle time)又称读/写周期,是指存储器从接收到地址到完成读出或写入操作的时间。在一般情况下存取周期越短,计算机运行的速度就越快。半导体存储器的存取周期一般为几至几百纳秒。

3. 带宽

存储器带宽(memory bandwidth)是指单位时间内存储器所存取的信息量。存储器的带宽决定了单位时间内数据传输的多少,单位为位/秒(bps)或字节/秒(Bps)。带宽可用以下公式计算,即

带宽 = 每个存取周期访问位数 / 存取周期

如存取周期为 500ns,每个存取周期可访问 16 位,则它的带宽为 32Mb/s(或 4MB/s),即 32Mb/s。

4. 可靠性

存储器的可靠性是指对电磁场、温度变化等因素造成干扰的抵抗能力,以及在高速运转时也能正确地存取。半导体存储器内部连线少、体积小,易于采取保护措施,所以与相同容

量的其他类型存储器相比，其抗干扰能力较强。存储器的可靠性可以用平均故障间隔时间来衡量。

5. 集成度

集成度是指在一块数平方毫米的芯片上所制作的基本存储元个数，单位为“位/片”或“字节/片”。很显然，存储芯片的集成度越高，构成相同容量的存储器所需要的存储芯片的数目就越少。MOS 型存储器的集成度高于双极型的存储器，动态存储器的集成度高于静态存储器。

目前，30nm 等级制造工艺的应用使得存储芯片单片容量已经达到了 4GB，单根 PC 内存条的容量达到了 32GB。

6. 功耗

半导体存储器属于大规模集成电路，集成度高，体积小，不容易散热，因此在保证存储容量和速度的前提下应尽量减小功耗。与双极型存储器相比，MOS 型存储器功耗更小。

7. 其他

在存储器性能上还应考虑输入输出电平是否与外接电路兼容、使用是否方便灵活及性价比等其他因素。

3.3 随机存储器 RAM

3.3.1 SRAM 存储器

1. SRAM 基本存储元电路

图 3.3 所示为一个主要由 6 个 MOS 管组成的 SRAM 的基本存储元及其读、写电路。

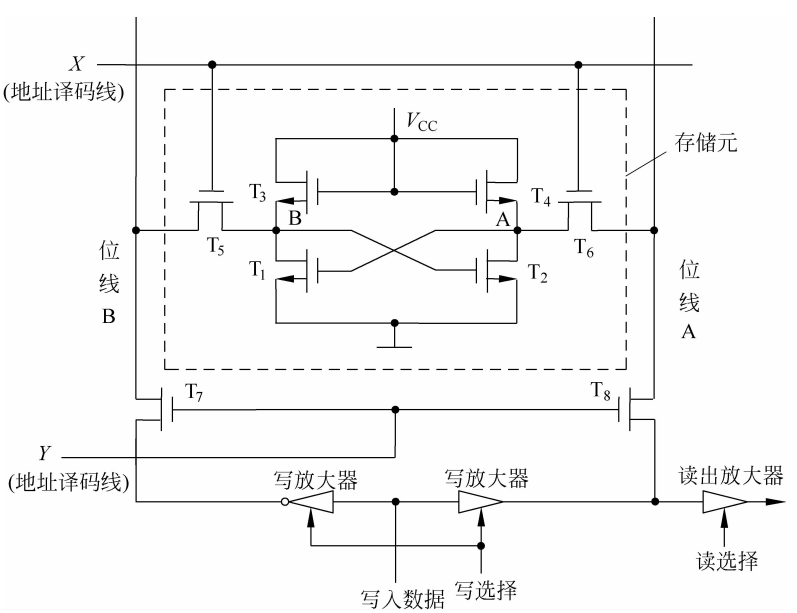


图 3.3 六管静态存储元电路

图 3.3 中，T_1、T_2、T_3、T_4、T_5 和 T_6 这 6 个 MOS 管组成 6 管静态存储元。T_1、T_2、T_3、T_4 组成了一个触发器基本电路，其中 T_1 和 T_2 交叉耦合组成双稳态电路，T_3 和 T_4 为负载管（相当于 T_1 和 T_2 的负载电阻）。T_5、T_6、T_7、T_8 起到开关作用，是存储元读出和写入数据时的控制管，其中 T_5、T_6 受地址译码线 X 控制，T_7、T_8 受地址译码线 Y 控制。需要注意的是，T_5、T_6、T_7、T_8 并不包含在某个基本存储元电路内，而是由存储芯片内同一列的所有存储元电路所共有。

下面举例说明该 6 管静态存储元电路的读、写工作过程。

读数据时，X、Y 地址译码线上的信号同时为高电平，使 T_5、T_6、T_7、T_8 均导通，这时 A 点电平通过 T_6 和位线 A 及 T_8 后作为读出放大器输入信号，在读选择有效时将数据读出。如果原来 A 点为高电平，即存储的是 1，则读出的是高电平；如果原来 A 点为低电平，即存储的是 0，则读出的是低电平。

写数据时，不管触发器原来状态如何，只要将数据送到“写入数据”端，在写选择有效时，经两个写放大器，使两端输出为相反电平。当行、列地址选择有效时，T_5、T_6、T_7、T_8 均导通，会使 A 与 B 两点置成完全相反的电平。例如，当写入数据“1”时，在两个写放大器和 T_7、T_8 两个 MOS 管的作用下，位线 A 和位线 B 上分别出现高电平和低电平，再通过 T_6、T_7 两个 MOS 管分别送到 A 点和 B 点。这时 A 点为高电平，B 点为低电平，表示写入了 1。之后，虽然外部数据线上的数据信号撤销，写控制信号和地址译码线上的信号也都变为无效的低电平，但 A、B 两点上的信号仍然可通过交叉耦合达到稳定状态。其原理为：A 点高电平送到 T_1 管的栅极，能让 T_1 管饱和导通，使得 B 点接地；B 点低电平送到 T_2 管的栅极，能让 T_2 管截止，使得 A 点通过负载管 T_4 与电源 V_{CC} 相连。只要不掉电，这一过程不断重复，最终达到稳定状态。

通过上面的分析可以知道，静态 RAM 是触发器存储信息，因此即使信息读出后，它仍然会保持原来状态，信息也只需要写入一次就能达到稳定状态，不需要刷新，因此速度较快，主要应用于高速缓冲存储器。

2. SRAM 存储器的组成

SRAM 由存储元阵列、地址译码器和读/写控制以及数据驱动/缓冲几个部分组成。一个典型的 SRAM 逻辑结构如图 3.4 所示。

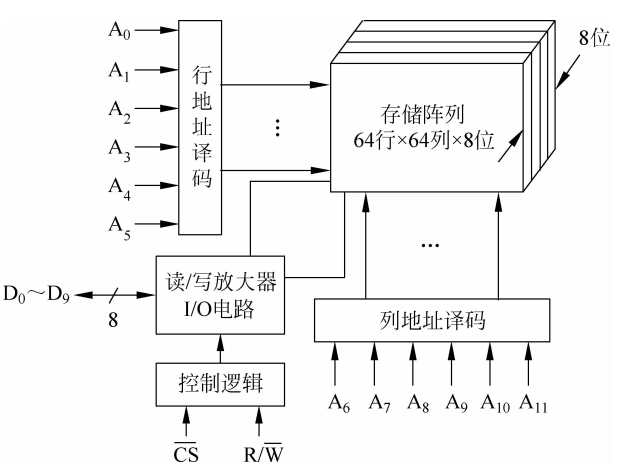


图 3.4　典型 SRAM 逻辑结构

图 3.4 是一个 4K×8 位的 SRAM 组成框图。它有 12 条地址线，8 条双向数据线，可寻址的存储单元有 4K 个，每个存储单元存储 8 位数据。存储芯片中的$\overline{CS}$为片选控制信号，R/$\overline{W}$ 为读、写控制信号。

对于较大容量的存储器通常像图 3.4 这样把多个相同容量的存储器芯片组成一组，各个字节(字)的同一位组织在同一片中。图 3.4 中 8 个存储容量都为 4K×1 位的芯片组成了一个 4K×8 位的存储器，这 8 个芯片的片选输入端连接在一起，使用时总是同时被选中或同时未被选中，从而实现 8 位数据同时读出和写入。

另外，由于地址线数目有限，选择存储单元时必须经过地址译码器生成各个芯片的片选信号。地址线的译码方式主要有两种，即双译码和单译码，如图 3.5 和图 3.6 所示。

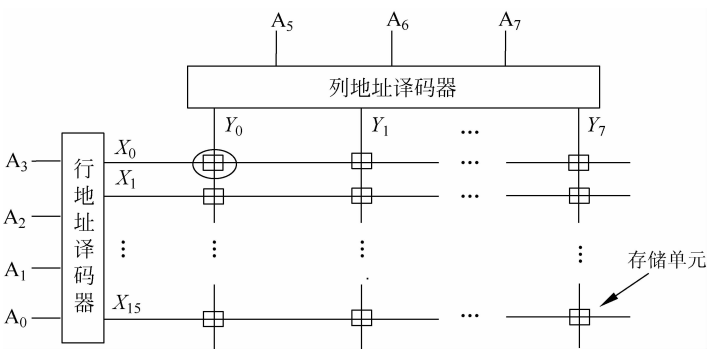


图 3.5　双译码方式

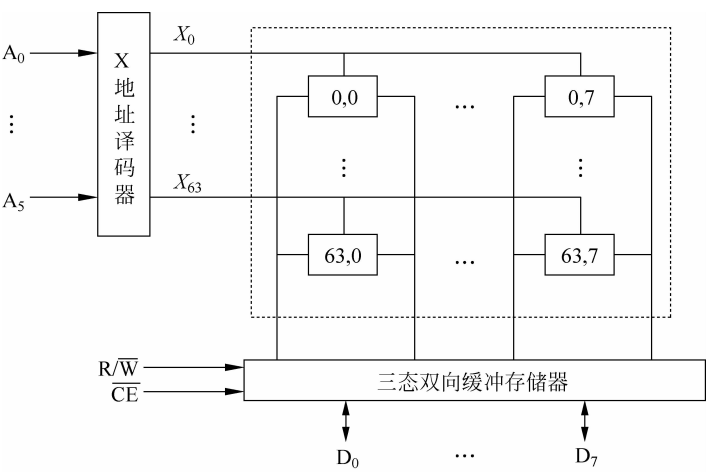


图 3.6　单译码方式

双译码也称为复合译码，即将地址线分为两组，分别送行地址译码器和列地址译码器进行译码，由行列译码输出同时选中的单元就是所访问的存储元或存储单元。例如，图 3.5 中行译码器译码输出的 X_0 和列译码器译码输出的 Y_0 同时选中的左上角的存储元(存储单元)就是要访问的存储元(存储单元)。

单译码也称为线性译码，即所有地址线通过一个地址译码器译码输出，每个译码输出线

直接选中一个存储元或存储单元。例如，10 条地址线译码输出后就会有 1024 条输出线选择存储元或存储单元。这种方式适用于容量不大的存储芯片中。

典型的 SRAM 芯片有 Intel 2114(1K×4b)、Intel 6116(2K×8b)、Intel 6264(8K×8b)、Intel 62256(32K×8b)等。

3.3.2 DRAM 存储器

DRAM 芯片是利用电容是否充有电荷来存储信息的，其基本存储元电路一般由 4 管、3 管或单管组成，以 3 管和单管比较常用。由于 DRAM 需要的 MOS 管较少，所以功耗低，容易扩大每个存储芯片的容量，在微机系统中应用广泛。

1. DRAM 基本存储元电路

图 3.7 所示为一个主要由 MOS 管和电容器组成的动态 RAM 基本存储元电路。

读数据时，使行列选择线为高电平，两个 MOS 管导通，若原来存储的是“1”，则 C 上有电荷，这时 C 放电，会通过 T 管在位线(数据线)上产生电流，完成读“1”操作；若读“0”，则 C 不会放电，位线(数据线)上不会产生电流。

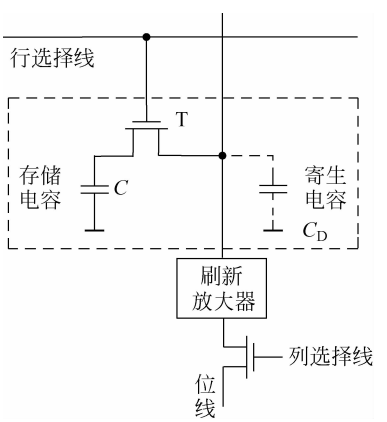


图 3.7 单管 MOS 动态 RAM 基本存储元电路

写数据时，同样使行列选择线为高电平，两个 MOS 管导通，若写入 1，位线(数据线)上加高电平，这时会对电容 C 进行充电；若写入“0”，位线上加低电平，如果原来 C 上有电荷(存储的是“1”)，就会通过位线进行释放，到达无电荷状态(写入 0)。

需要注意的是，这种单管 MOS 动态 RAM 的读数据过程是电容 C 的放电过程，是一种破坏性读出，因此要想保持原有的信息，数据读出后必须重写。图 3.7 中的刷新放大器在数据读出后，会将原信息重新写回到电容 C 中。

DRAM 存储元以电容为基础，电路简单，集成度高，功耗低，但需要额外的定时刷新操作，因此速度与 SRAM 相比相对较慢，适用于大容量存储器，如内存。

2. DRAM 存储器的组成

SRAM 由存储元阵列、地址译码器和读/写控制、数据驱动/缓冲、地址锁存器和刷新控制电路几个部分组成。一个典型的 DRAM 逻辑结构如图 3.8 所示。

与 SRAM 相比，DRAM 中增加了地址锁存器和刷新控制电路。

DRAM 芯片的容量相对较大，也就需要更多的地址线进行访问，但增多地址线无疑会导致芯片引脚数目增加。为解决这一问题，在 DRAM 中采用了分时传送地址码的技术。图 3.8 中，要访问 1M×4b 容量的 DRAM 存储器需要 20 条地址线，但芯片引脚只提供了 10 条地址线，这 10 条地址线分时把低 10 位地址和高 10 位地址传送到 DRAM 的行地址锁存器和列地址锁存器中进行锁存，之后在 CPU 发出的读/写控制命令作用下完成对某一存储单元的读/写操作。这里地址信号要分两次传送，因此影响了读、写速度。

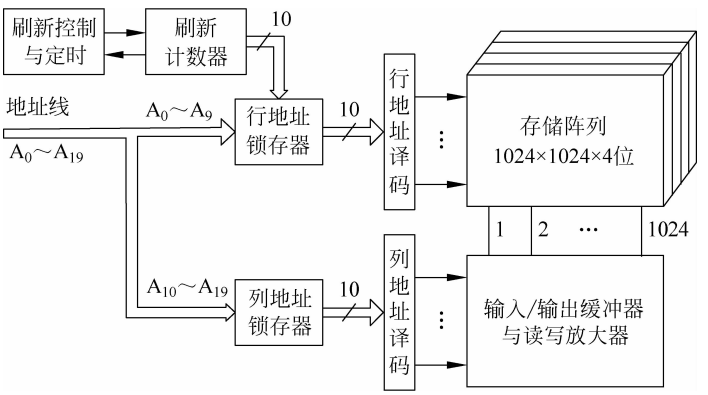


图 3.8　DRAM 存储器的逻辑结构

另外，由于 DRAM 的读操作是破坏性的，再加上电容本身的漏电问题，必须对存储芯片进行定时刷新操作。例如，对于早期的 128×128 芯片，要求在 2ms 内完成 128 行的刷新操作，这段时间叫做刷新周期或再生周期。随着半导体芯片技术的发展，现在刷新周期最长可达 8ms 以上。

3. DRAM 的刷新方式

刷新操作是逐行进行的，相当于一系列的读和写操作。当某一行选择信号为“1”时，选中了该行，电容上的信息送到刷新放大器，刷新放大器又对这些电容立即进行重写。由于刷新时列选择信号总为“0”，因此电容上的信息不可能被送到数据总线上，也就是说，刷新操作和正常的读/写操作不能同时进行。

按照再生操作的时机，常见的刷新方式有集中式、分散式和异步式 3 种。

1）集中式刷新

集中式刷新是将刷新周期划分为两个部分，前一部分由 CPU 对存储器进行正常的读/写或维持操作，后一部分集中完成所有行的刷新操作，如图 3.9 所示.

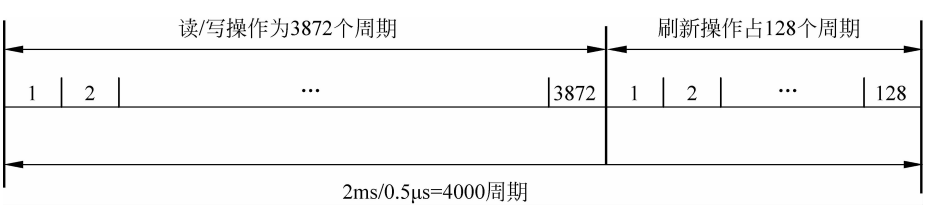


图 3.9　集中式刷新

假设某 DRAM 刷新周期为 2ms，存取周期为 0.5μs（即刷新一行需要 0.5μs），则刷新 128 行需要 64μs。这段时间内 CPU 不能对存储器进行正常的读/写操作，这段时间称为 CPU 的“死区”，或称“死时间”。

2）分散式刷新

集中式刷新中存在“死区”，尽管时间很短，但有些情况下仍然会影响系统效率。分散式刷新可以避免 CPU 较长时间不能正常读/写存储器的问题。这种刷新方式是把对每行的刷新分散到每个读/写周期内完成。把存储周期分成两部分，前半部分用于读/写或维持，后半部分用于刷新操作，每次刷新一行，如图 3.10 所示。

图 3.10　分散式刷新

这种刷新方式克服了“死区”的缺点，但它有可能导致整机的工作效率下降。因为尽管把刷新操作分散在读/写操作后，但每次刷新操作同样需要一个读/写周期时间，结果使系统的存取周期增加了一倍。另外，这种方式也不利于充分利用所允许的最大刷新时间间隔，导致可能会产生频繁刷新现象，这也降低了存储器的存取速度。

3）异步式刷新

为真正提高整机的工作效率，可以采用集中式与分散式相结合的方式，把刷新操作平均分配到刷新周期内进行，既能克服“死区”，又能充分利用最大刷新间隔，如图 3.11 所示。

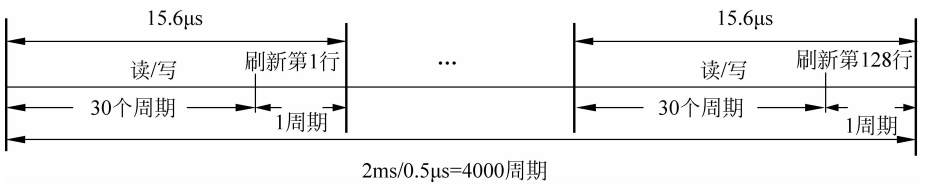


图 3.11　异步刷新

这种刷新方式每隔一定时间（刷新周期/行数）刷新一行。例如，对于 128×128 的存储芯片，刷新周期为 2ms 时，只需要每隔 2ms/128＝15.6μs 刷新一行，而每行刷新的时间仍为读/写周期 0.5μs，即刷新一行只停止一个读/写周期，“死区”大大缩短。

3.3.3　现代 DRAM

到目前为止，主存的基本核心部件仍然采用 DRAM，这与高速发展的 CPU 技术形成了鲜明的对比，已经成为整机性能发展的瓶颈。因此，人们相继从提高时钟频率、带宽和缩短存储周期等方面开发了基于基本 DRAM 结构的增强型的存储芯片。

1. FPM DRAM（Fast Page Mode DRAM）

FPM DRAM 为快速页面切换模式动态随机存取存储器，是一种改良版的 DRAM，在 486 时期曾被普遍应用。传统的 DRAM 在存取一位数据时，必须送出行地址和列地址各一次才能读/写数据，而 FPM DRAM 在触发了行地址后，如果 CPU 需要的地址在同一行内，则可以连续输出列地址而不必再输出行地址了。FPM DRAM 每 3 个时钟周期传输一次数据。

2. EDO RAM（Extended Data Output RAM）

EDO RAM 为扩展数据输出存储器，是一种在 RAM 中加入一块静态 RAM 而生成的动态存储器。因为 SRAM 的访问速度要快于 DRAM，所以这会加快访问内存的速度。EDO RAM 有时作为二级缓存和 ESDRAM（带缓存的 RAM）一起使用。EDO RAM 每 2 个时钟周期传输一次数据。

3. SDRAM(Synchronous DRAM)

SDRAM 为同步动态随机存储器,它与系统时钟以相同的速度同步工作,内部命令的发送与数据的传输都以它为基准,可以实现数据随机存取,整块传输。SDRAM 在每个时钟周期的上升沿可传输一次数据。

4. DDR SDRAM(Double Data Rate SDRAM)

DDR SDRAM 即双倍速率同步动态随机存储器,习惯简称 DDR,就是市场上的 DDR 内存。DDR 内存是在 SDRAM 内存基础上发展而来的,仍然沿用 SDRAM 生产体系,因此对于内存厂商而言,只需将制造普通 SDRAM 的设备稍加改进,即可实现 DDR 内存的生产,可有效降低成本。DDR 内存每个时钟周期内可以传输两次数据,即能够在时钟的上升期和下降期各传输一次数据,称为双倍速率同步动态随机存储器,因此与 SDRAM 相比可以达到更高的数据传输率。

DDR2(Double Data Rate 2)是 DDR SDRAM 的第二代产品,采用了诸多的新技术,改善了 DDR 的诸多不足。DDR2 与上一代 DDR 内存技术标准最大的不同就是,虽然同是采用了在时钟的上升/下降沿各传送一次数据的基本方式,但 DDR2 内存拥有两倍于上一代 DDR 内存预读取能力(即 4 位数据预读取)。换句话说,DDR2 内存每个时钟能够以 4 倍外部总线的速度读/写数据,并且能够以内部控制总线 4 倍的速度运行。此外,DDR2 的耗电量更低,散热性能更优良。

当前市场上比较流行的是 DDR3 内存。面向 64 位构架的 DDR3 在频率和速度(8 位数据预读取)上拥有更多的优势,同时还采用了根据温度自动刷新、局部自刷新等其他一些功能,在功耗方面 DDR3 也要出色得多。

现在,基于差分信号技术的 DDR4 内存也即将浮出水面。DDR4 内存的传输速率可以达到 1.6～3.2Gb/s,频率最高可达 4266MHz,1.2V 的低电压,拥有更好的对等保护和错误恢复等技术。最新消息显示,DDR4 内存或于 2014 年首先用于服务器领域,然后再过一年半左右进入桌面。

除上述几种芯片外,还先后出现了 CDRAM、SLDRAM、RDRAM、并行型 RDRAM、Direct RDRAM、PC100 SDRAM、DRDRAM 和 VCM 等增强型的 DRAM 芯片。

3.4 只读存储器 ROM

RAM 存储器具有易失性,掉电后数据将会丢失。然而,在实际应用中常常还需要掉电后内容不丢失的存储器,半导体只读存储器 ROM 就具有这一功能。早期的 ROM 也可以分为双极型和 MOS 型,但现在双极型的早已被淘汰,所以这里只对 MOS 型 ROM 加以介绍。

1. 掩膜式只读存储器(MROM)

MROM 中的信息是芯片在生产时由生产厂家固化(掩膜)的,生产时需要专用的掩膜模具,制成以后不能擦除和修改。这种 ROM 往往会存储一些特殊功能的程序或数据,如早期的 BIOS 芯片就是用 MROM 实现的。这种存储器可靠性高、成本低,一般适用于在大批量应用的场合。MROM 的存储结构如图 3.12 所示。

掩膜 ROM 基本存储电路由单管组成,集成度高。图 3.12 是一个 4×4 位 MOS 管 MROM,采用单译码方式,两位地址线 A_1 和 A_0 译码后可译出 4 种状态,输出 4 条选择线,

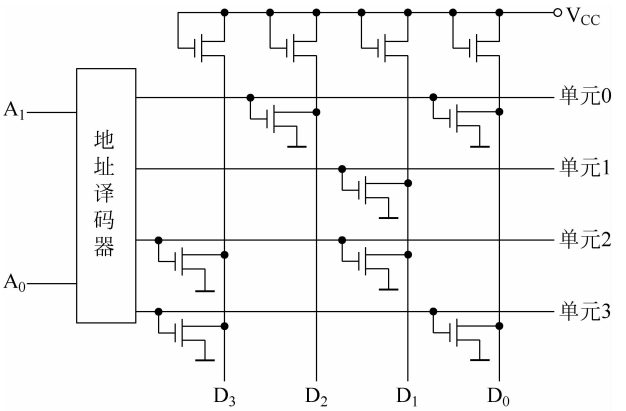


图 3.12　MROM 存储结构

分别选中 4 个单元，每个单元有 4 个(位)输出。存储矩阵中，行列选择线交叉有的连有 MOS 管，有的没有，有 MOS 管表示存储的信息为 0，否则为 1。

2. 可编程只读存储器(PROM)

PROM 通常是指用户能够进行一次性编程写入信息的 ROM 芯片，即芯片出厂时里面还没有写入任何信息，用户使用时可以将自己的程序和数据通过专用的编程器写入芯片中。但需要注意的是，这种写入是“破坏性”的，也就是说，信息一旦写入就不能再修改或删除。

PROM 的成本比 MROM 高，信息写入的速度比 MROM 要慢，一般只适用于少量需求的场合或是 MROM 批量生产前的试制。

图 3.13 是一个 PROM 的基本存储元结构。

图 3.13 中的熔丝一般采用镍铬熔丝，连接到存储元件的发射极，出厂时所有存储元的熔丝都是连着的，表示存储的数据全是“1”。编程写入时用足够大的电流把熔丝烧断表示写入了数据“0”。当读取信息时，行线加高电平，三极管导通，如果熔丝未断，列线会与 V_{CC} 接通，即输出高电平，表示读出了 1；否则输出低电平，表示读出了 0。

3. 紫外线可擦除可编程只读存储器(EPROM)

EPROM 是紫外线可擦除可编程的只读存储器，解决了 PROM 芯片只能写入一次的弊端。图 3.14 是一个 EPROM 基本存储元电路。

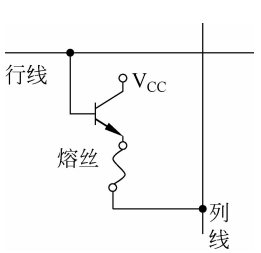


图 3.13　PROM 的基本存储元结构

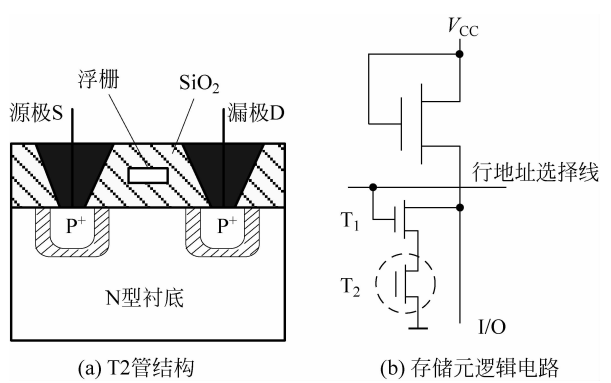


图 3.14　EPROM 基本存储元电路

图 3.14(a)是一个 P 沟道浮栅 MOS 管结构。它与普通 P 沟道 MOS 电路的区别就是栅极没有引出线，称为“浮栅”。EPROM 芯片制造好后，栅极上没有电荷，源极 S 和漏极 D 之间不导通，存储的信息为 1。

在编程写入时，S 和 D 之间加＋12V 或＋25V 的正电压(不同芯片，该电压要求不同)和编程脉冲，使得在漏极与衬底之间反向偏置的 PN 结被迅速击穿，于是有高能电子通过极薄的 SiO_2 绝缘层注入到浮栅上。当高电压去掉后，注入到浮栅中的电子被绝缘层包围，无法复合掉靠近栅极的多余空穴，于是在 S 和 D 之间就形成了一个导电沟道，S 和 D 能够导通，这时相当于写入了信息“0”。

在读出数据时，行地址选择线置高电平，T_1 管导通，如果 T_2 中存储的是 1，即 T_2 不能导通，会在 I/O 线上输出高电平；如果 T_2 中存储的是 0，即 T_2 导通，则会在 I/O 线上输出低电平。

EPROM 芯片要想重新写入数据必须先进行擦除。EPROM 出厂时会在陶瓷封装表面开一个石英窗口，透过该窗口，可以看到其内部的集成电路，用紫外线透过该孔照射内部芯片若干分钟，浮栅中的电子获得能量会穿过绝缘层跑掉，就可以擦除存储的数据，使所有单元的内容均初始化为 1，之后就可以重新编程写入数据了。

4. 电可擦除可编程只读存储器(EEPROM)

EPROM 虽然满足了可改写的需求，但只能擦除全部信息，而且擦除时间过长，使用紫外线也不太方便。EEPROM 是一种可用电气方法在线擦除和再编程的只读存储器，写入时不需要专门的编程电压，可直接用系统＋5V 的电源。EEPROM 既具有 ROM 非易失性的特点，又像 RAM 一样可以随机地进行读/写操作，写入的数据保留的时间可长达 20 年，并且每个单元可重复进行一万次以上的改写，最近推出的 EEPROM 芯片的可改写次数已经达到了 10 万次。图 3.15 是一个 EEPROM 基本存储元电路。

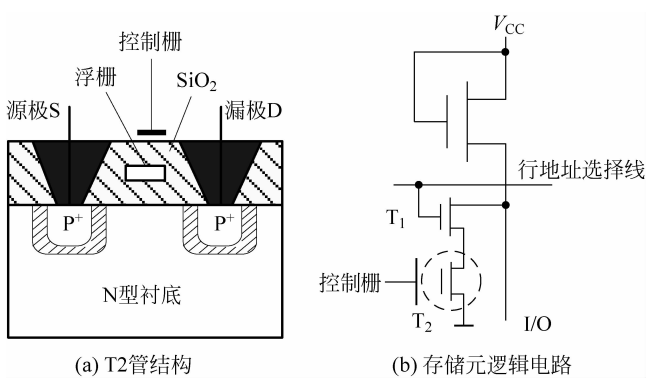


图 3.15　EEPROM 基本存储元结构

EEPROM 与 EPROM 相比，存储元结构大体相同，也采用浮栅技术，以浮栅上是否带有电荷决定存储的信息是 1 还是 0，只不过额外在浮栅上方的 SiO_2 上增加了一层金属层，构成了一个控制栅。

EEPROM 信息的写入方法与 EPROM 相同，而在擦除信息时要把控制栅接地，并在源极 S 加较高的正电压，浮栅中的电子在较强的电场力作用下被拉出吸到源极，从而实现了信息的擦除。需要注意的是，随着技术和工艺的进步，许多 EEPROM 芯片在写入数据时并不

需要先擦除原来存储的信息，也不再需要额外提高擦除和写入电压，而是直接使用＋5V 的工作电压就可以实现在线写入了。例如，采用 EEPROM 的 BIOS 芯片大多都有双电压特性，通过跳线开关就可很方便地实现编程电压和工作电压的切换。

5. 闪速存储器(Flash)

闪速存储器即通常所说的闪存，按照内部结构和数据访问方式，它属于 ROM 类存储器，是在 EPROM 和 EEPROM 基础上发展而成的，因此两者有很多相似之处。闪存对 RAM 和 ROM 各取所长，具有集成度高、功耗低、抗干扰能力强、速度快、容量大、非易失性和价格便宜等优点。

现在 Flash 用途广泛，主要用于 U 盘、显卡、网卡、声卡等计算机接口设备中，以及数码相机、数码摄像机、手机等智能设备和终端设备中。

总之，存储器的发展具有容量更大、体积更小、速度更快、成本更低、耗电更少、可靠性更高、易用性更强等趋势。尤其闪存技术的发展，替代磁性存储介质的趋势已经越来越明显，虽然想要替代各种 RAM 存储器还有很长的路要走，但终究值得期待。

3.5 存储器接口设计

计算机存储系统的容量往往很大，单个存储芯片的容量和数据宽度都很难满足系统要求，这就需要选择使用多个芯片组合成大的存储系统。然而，不同型号和种类的存储芯片所拥有的存储单元个数和每个单元的数据位数都可能不同，如何选用不同的存储芯片、如何进行地址译码和片选、如何与 CPU 进行连接来构成系统所需的存储器，是本节要讨论的问题。

3.5.1 存储芯片的选择

1. 芯片类型的选择

用于存放需修改的程序、现场采集的数据、运算的中间结果或其他临时性的信息，应选用既能读又能写的 RAM 芯片。对于存储容量要求较小的应用系统、专用设备和速度要求较高的场合应尽量选用 SRAM，因为 SRAM 硬件电路简单，速度也快；对于存储容量要求较大的场合，如计算机内存，则应选择 DRAM。

用于存放监控程序、用户的固定程序、固定的表格和常数，则选用 ROM 存储器，如智能仪表、家用电气等设备的存储器一般就采用 ROM 芯片。

用于存放永久性的程序和数据的情况可选用 ROM 或 PROM；其他需要频繁读写的场合则应优先考虑选用 EPROM、EEPROM 或 Flash。

此外，还要综合考虑芯片的功耗、容量、速度、价格、封装引脚等因素。

2. 芯片数量的确定及存储器扩展方式

芯片数量的确定要从存储器的存储单元个数和存储单元位数两个角度考虑，其原则是在满足系统要求的前提下需要的芯片数量最少。

1) 位扩展

当存储芯片的每个单元的数据位数不能满足存储器所需要的位数时，就要进行位扩展。存储芯片根据单元位数划分有 1 位片、4 位片、8 位片、16 位片和 32 位片等，如 Intel 6116 芯

片就是 8 位片。如果采用 1 位片构成 16 位存储系统则需要 16 片，而如果采用 8 位片构成 16 位存储系统则只需要两片。这些组合芯片可以采用位并联的方法进行连接，即各芯片并联在一起，每个芯片占据存储单元不同的数据位，从而构成所需要的存储位数。

例如，用 1K×2 位的存储芯片构成 1K×8 位的存储器，所需芯片数为 4 片。扩展连线方法如图 3.16 所示。

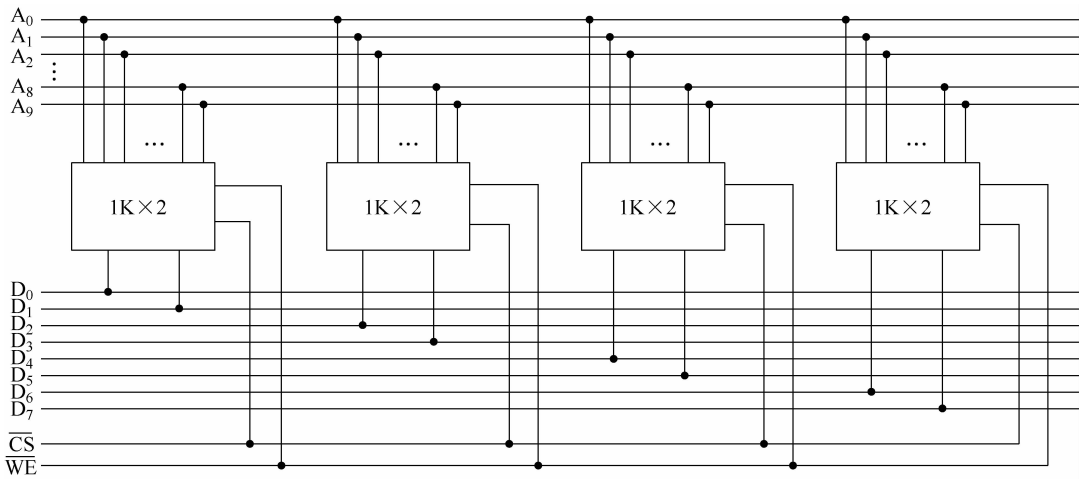


图 3.16　位扩展电路示意图

图 3.16 中 4 个芯片地址线都为 $A_0 \sim A_9$。读取数据时在$\overline{CS}$信号的作用下会同时选中 4 个芯片，每个芯片输出两位数据信息，组合成 8 位数据一起在数据总线上输出，写入数据过程同理。

2）字扩展

字扩展就是增加存储单元的数目，也称为地址扩充。扩展时用一些容量较小的存储芯片采用地址串联的方法组成容量较大的存储器。每个芯片的片选线通过高位地址译码后分别生成，以免地址重叠。图 3.17 是用 4 片 1K×8 位存储芯片组成 4K×8 位的存储器原理。

图 3.17 中 4 个芯片数据线都为 $D_0 \sim D_7$，且它们共用地址线 $A_0 \sim A_9$ 用于各芯片内部寻址。地址总线最高两位 A_{10} 和 A_{11} 输入到 2-4 译码器编码产生 4 个芯片的片选信号$\overline{CS}$，这样各芯片内部地址虽然相同，但由于高两位地址各不相同，所以每个芯片的物理地址并不重叠。

3）字位扩展

在实际应用中，在构成存储器时往往需要同时进行字扩展和位扩展，这种扩展方式称为字位扩展。图 3.18 是用 8 片 1K×4 位的存储芯片构成 4K × 8 位的存储器原理。

图 3.18 中共用到了 8 片 1K×4 位的存储芯片，这些芯片两两一组进行位扩展，共组成了 4 组 1K×8 位的存储器。然后 4 组 1K×8 位的存储器再进行字扩展，最终组成 4K×8 位的存储器。这里地址线最高两位作为片选地址译码生成 4 组 1K×8 位的存储器的片选信号，地址线 $A_0 \sim A_9$ 用于片内寻址。

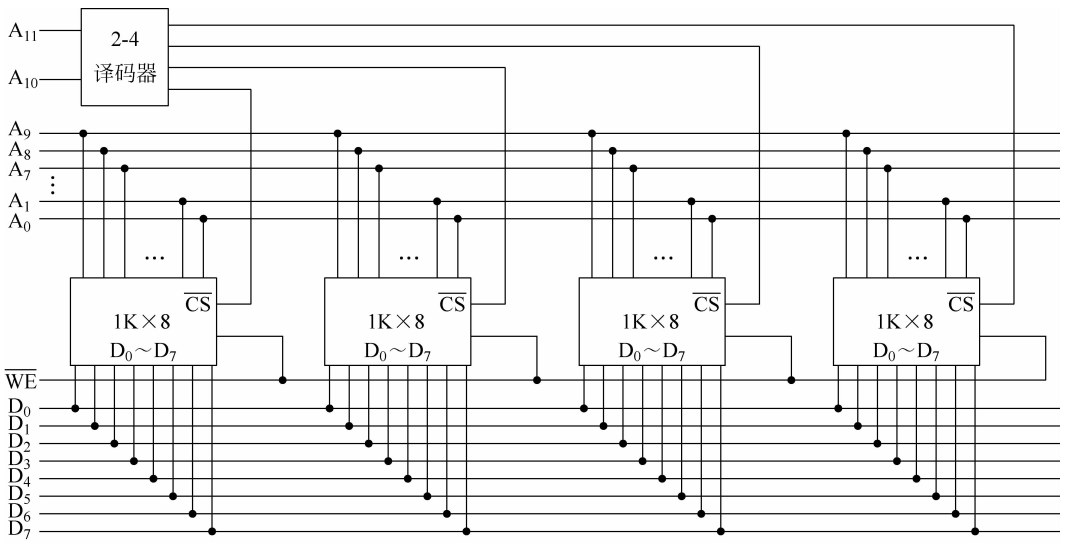


图 3.17　字扩展电路示意图

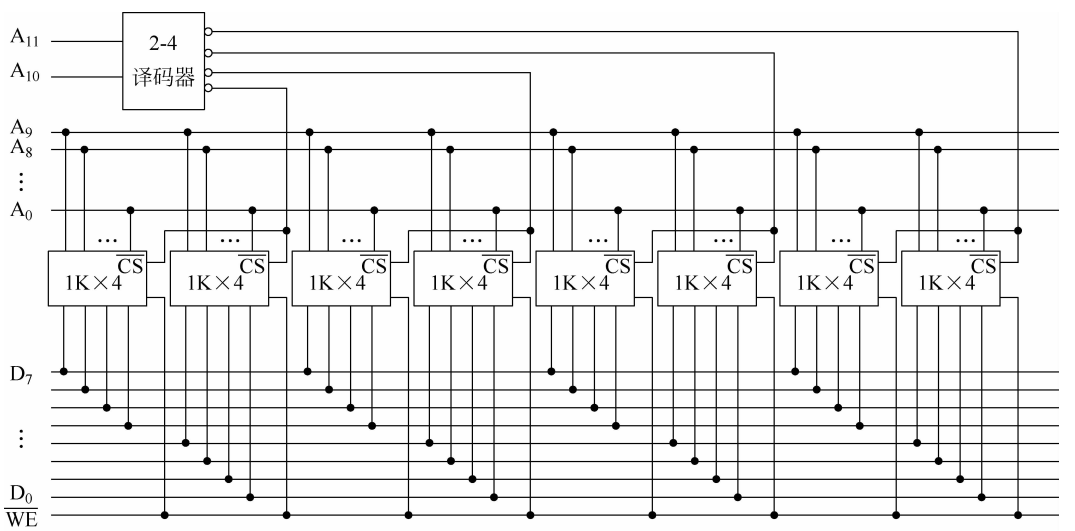


图 3.18　字位扩展电路示意图

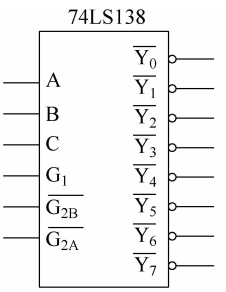
3.5.2　存储器的地址译码及地址分配

前面已经提到，一个容量较大的存储器往往需要用多片容量较小的芯片组合扩展来实现，为了能准确无误地访问到存储器中的任意一个单元，必须先确定要访问的单元在哪个存储芯片中，这个工作需要由逻辑译码电路或译码器来完成。

1. 译码器

常见的译码器有 Intel 8025、74LS138、74LS139、74LS155、74LS156 等，其中 74LS138 极为典型。图 3.19 是 74LS138 引脚定义及输入输出信号关系。

当一个选通端(G_1)为高电平，另两个选通端($\overline{G_{2A}}$和$\overline{G_{2B}}$)为低电平时，可将地址端(A、



输入				输出							
G_1	$\overline{G_{2A}}$	$\overline{G_{2B}}$	$C\ B\ A$	Y_7	Y_6	Y_5	Y_4	Y_3	Y_2	Y_1	Y_0
1	0	0	0 0 0	1	1	1	1	1	1	1	0
1	0	0	0 0 1	1	1	1	1	1	1	0	1
1	0	0	0 1 0	1	1	1	1	1	0	1	1
1	0	0	0 1 1	1	1	1	1	0	1	1	1
1	0	0	1 0 0	1	1	1	0	1	1	1	1
1	0	0	1 0 1	1	1	0	1	1	1	1	1
1	0	0	1 1 0	1	0	1	1	1	1	1	1
1	0	0	1 1 1	0	1	1	1	1	1	1	1
0	×	×	×××	1	1	1	1	1	1	1	1
×	1	×	×××	1	1	1	1	1	1	1	1
×	×	1	×××	1	1	1	1	1	1	1	1

图 3.19　74LS138 引脚及真值表

B、C)的二进制编码在一个对应的输出端以低电平译出。

2. 地址译码方案及地址分配

译码器可以产生各存储芯片的片选信号，从而能够决定各芯片的地址范围。常用的译码方案有 3 种，即全译码方式、部分译码方式和线性译码方式。采用译码方式进行存储器扩展时如果不需要全部存储空间，可以将多余的地址线悬空，便于系统扩展。

1) 全译码方式

全译码方式是指全部的片选线(片选地址)都参加片选译码，这种方式不浪费存储器地址空间，每个存储单元地址唯一，各存储芯片的地址连续且不重叠。

【例 3.1】 假设某系统有 16 条地址线，用 4 片 16K×8 位存储芯片组成存储系统，则系统结构图及各芯片地址分配如图 3.20 所示。

可以看出，存储单元的物理地址由片选地址和片内地址两部分组成，各芯片的片内地址范围都是相同的，所不同的就是地址线的最高两位即片选信号不同，因此存储器中各存储单元的物理地址都是不同的。

2) 部分译码方式

部分译码方式是指片选线(片选地址)中的部分线参与译码，产生存储芯片片选信号。这种译码方式每个存储单元的地址不唯一，因为没有参加译码的片选线可以为 0 也可以为 1。

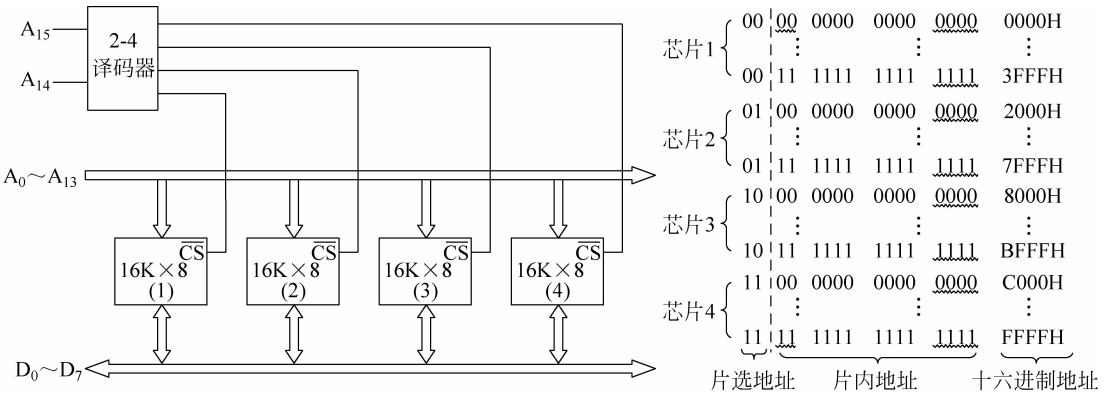


图 3.20　全译码电路及地址分配

【例 3.2】 设一个有 16 根地址线的微机系统，有 2KB 的 RAM 和 1KB 的 ROM，分别选用 1K×8 位的存储芯片，且 RAM 和 ROM 统一编址。试采用部分译码方式设计该微机的存储系统。

根据题意，每个存储芯片大小都为 1K×8 位，其片内寻址只需要使用低 10 位地址线（A_0～A_9）。另外，3 个存储芯片共需要 3 个片选信号线，剩余 6 条高位地址线中任取两根连接 2-4 译码器就可满足要求。根据上述分析，可设计出如图 3.21 所示的译码电路。

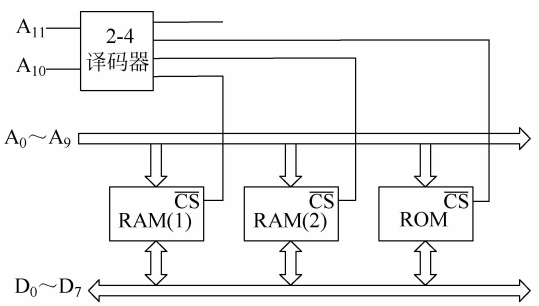


图 3.21　部分译码方式

3 个存储芯片的地址范围如表 3.2 所示。需要注意的是，A_{12}～A_{15}没有参加译码，所以都可以是 0 或 1，导致每个存储单元都有 16 个地址。如果不用 A_{10}和 A_{11}译码，而选用其他高位地址线译码，结果会使存储芯片地址不连续。另外，$A_{11}A_{10}=11$ 时，由于没有选中任何芯片，导致了地址空间的浪费。

表 3.2　芯片地址范围

译码输入	选中芯片	芯片片内地址	芯片高 4 位地址	芯片物理地址（X=0～F）
$A_{11}A_{10}=00$	RAM(1)	0000H～03FFH	0000B～1111B	X000H～X3FFH
$A_{11}A_{10}=01$	RAM(2)	0000H～03FFH	0000B～1111B	X400H～X7FFH
$A_{11}A_{10}=10$	RAM(3)	0000H～03FFH	0000B～1111B	X800H～XBFFH
$A_{11}A_{10}=11$	无	无	无	无

3）线性译码方式

线性译码方式是指直接用某根地址线作为存储芯片的片选线。这种译码方式的优点是

不需要译码电路，所以线路简单；缺点是地址可能不连续，而且会造成存储空间的浪费，只适用于存储芯片较少的情况。

【例 3.3】 设一个具有 16 根地址线的微机系统有 2KB 的 RAM，选用 1K×8 位的存储芯片实现。试采用线性译码方式设计该微机的存储系统。

题目中只有两个芯片需要连接，所以可以只利用一条片选线结合一个非门来生成两个芯片的片选信号。具体电路连接及芯片地址范围如图 3.22 所示。

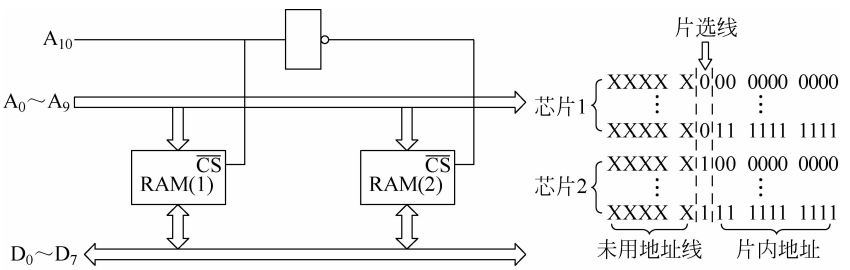


图 3.22 线性译码电路及地址分配

注意，这里是把 A_{10} 作为片选线，如果采用 A_{15} 作为片选线，两个芯片的地址范围又是多少呢？请读者自己分析。

3.5.3 存储器接口设计举例

【例 3.4】 设 CPU 有 16 根地址线，8 根数据线，用 $\overline{MREQ}$ 作访存控制信号（低电平有效），用 $\overline{WR}$ 作读/写控制信号（高电平为读，低电平为写）。其地址空间分配以下：最低 8K 地址为系统程序区，与其相邻的 8K 地址为用户程序区，最高 4K 地址空间为系统程序工作区。现有以下存储器芯片可供选择：4K×8 位的 ROM、8K×4 位的 RAM 和 4K×8 位的 RAM。另有一个 3-8 译码器可供使用。要求：

（1）选择适当的芯片类型及片数。

（2）写出各芯片的地址编码范围。

（3）确定芯片的片选逻辑，并画出各器件之间的逻辑连接图。

解：

（1）根据题目要求，最低 8K 地址空间为系统程序区，一般只能进行读操作，应该采用两片 4K×8 位的 ROM 芯片；与其相邻的 8K 地址为用户程序区，需要经常被改写，所以采用两片 8K×4 位的或两片 4K×8 位的 RAM 芯片，这里选择 8K×4 位的芯片；最高 4K 地址空间为系统程序工作区，也应选择且只能选择 4K×8 位的 RAM 芯片。

（2）各芯片的地址范围如图 3.23 所示。

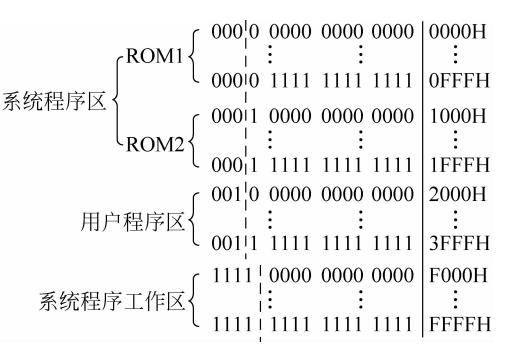


图 3.23 各芯片地址范围

可以看出，两片 4K×8 位的 ROM 芯片组合进行字扩展，构成 8K×8 位的 ROM，所以系统程序区共需要 A_0～A_{12} 共 13 条地址

线进行内部寻址；用户程序区由两个 8K×4 位的 RAM 芯片通过位扩展构成一个 8K×8 位 ROM，也需 A_0～A_{12} 共 13 条地址线进行内部寻址；系统程序工作区由 4K×8 位的 RAM 组成，共需要 A_0～A_{11} 共 12 条地址线进行内部寻址。

(3) 根据题意及(2)中的分析，地址线最高 3 位 A_{13}、A_{14} 和 A_{15} 应该作为 3-8 译码器的译码输入信号，输出的 $\overline{Y_0}$ 和 A_{12} 进行逻辑或后作为 ROM1 的片选信号；A_{12} 取反后再与 $\overline{Y_0}$ 进行逻辑或，其结果作为 ROM2 的片选信号；$\overline{Y_1}$ 同时作为 RAM1 和 RAM2 的片选信号；A_{12} 取反后再与 $\overline{Y_7}$ 进行逻辑或，其结果作为 RAM3 的片选信号。

经过上述分析，得出芯片的片选逻辑及各器件之间的逻辑连接如图 3.24 所示。

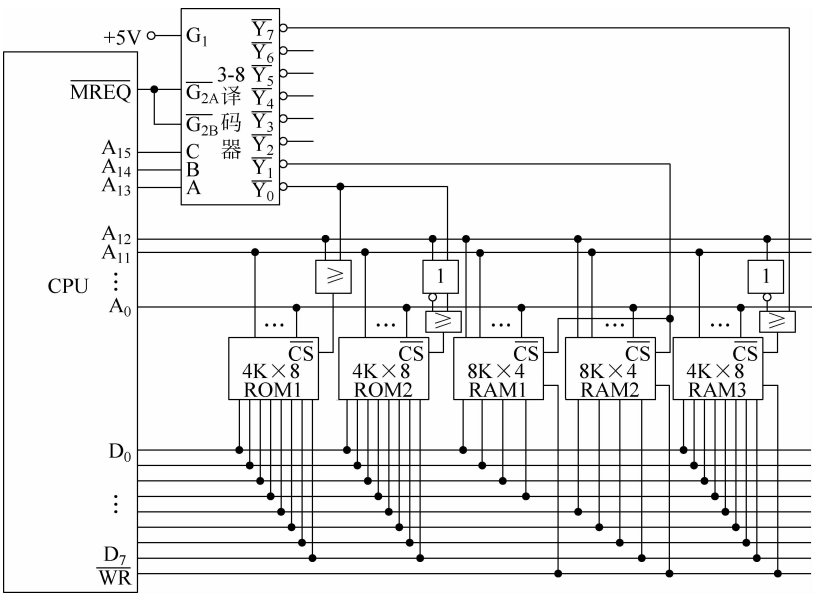


图 3.24　CPU 与存储芯片连接

前面提到 8K 用户程序区还可以采用两片 4K×8 位的 RAM 芯片组成，其实这样做逻辑连线更简单，为什么？读者可以自己画出图形分析解决。

3.6　高速缓冲存储器

随着计算机技术的发展及应用的普及，计算机处理的信息量越来越大、越来越复杂，因此对计算机的整体性能要求越来越高。与运算速度越来越快的 CPU 相比，主存的存取速度显然已经成为计算机系统整体性能提升的主要瓶颈。在这样的背景下，出现了高速缓冲存储器(Cache)。

为了加快运算速度，从 80486 开始的微处理器中增设一级或两级与 CPU 处理速度相当的 Cache。目前微机系统中大多有两级 Cache；第 1 级 Cache(简记为 L1)位于处理器内部，分为数据缓存和指令缓存两部分；第 2 级 Cache(简记为 L2)有处于处理器内部和外部两种，容量比 L1 大一些，速度比 L1 慢一些，但要比内存快。Cache 与各种存储器的性能比较见表 3.1。

3.6.1 Cache 概述

Cache 是位于 CPU 与内存之间的临时存储器，是为了解决 CPU 运算速度与内存读写速度不匹配的矛盾而采用的一项技术。Cache 的容量比内存小得多，但是速度比内存要快得多，与 CPU 相匹配。前面已经介绍，主存一般采用容量较大但速度相对较慢的 DRAM 芯片，而 Cache 则一般采用容量较小但速度很快的 SRAM 芯片。

1. Cache 的基本原理

对大量典型程序运行情况分析结果表明，在某一个较短的时间间隔内，CPU 对一个程序指令和数据的访问往往集中在主存的某个较小的空间范围内。这种在某段时间内对存储器局部范围的频繁访问，而对范围以外的存储空间很少访问的现象，叫做程序访问的局部性原理。正是由于这种局部性原理，才使得将 Cache 技术应用到计算机系统成为可能。

2. Cache 的工作过程

在拥有 Cache 的计算机系统中，主存中保存所有的程序和数据，而 Cache 中保存主存的部分副本。当 CPU 访问存储器时，首先通过一个主存—Cache 的地址映射机构将主存地址转换成一个 Cache 地址。如果 CPU 要访问的内容已经在 Cache 中，则从 Cache 中将要访问的内容读出并送 CPU，这种情况称为命中 Cache；若 CPU 要访问的内容不在 Cache 中，即没有命中 Cache，那么 CPU 必须直接访问主存，从主存中将内容读出并送入 CPU，从而完成了一次 CPU 的访存操作。在正常情况下，CPU 对 Cache 的存取命中率可达 95％以上。Cache 的工作原理框图如图 3.25 所示。

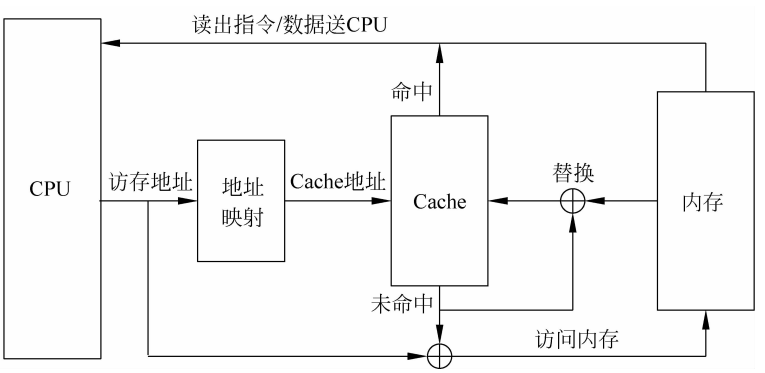


图 3.25　Cache 工作原理框图

在 Cache 存储器组织中，Cache 和主存都按块进行组织，每一个块由若干个字或字节组成，且主存的块和 Cache 的块大小相同，它们之间的数据交换都以块为单位进行。在没有命中 Cache 的情况下，CPU 访存结束后，还需将本次所访问的内容所在主存块的全部单元内容调入 Cache 中。若当前 Cache 已满，则需要采用一定的规则将 Cache 中的一个旧块替换出来。

3.6.2 Cache 的映射方式

前面已经介绍，在没有命中 Cache 的情况下，CPU 访存结束后，还需将本次所访问的内容所在的整个区块从主存复制到 Cache 中。为此，需要在主存和 Cache 之间建立一种映像

关系，即地址映射。地址映射即是应用某种策略把主存地址定位到 Cache 中，在需要时将主存块按照事先建立的地址映射关系装入对应的 Cache 块中。

常用的 Cache 地址映射方式有 3 种，即直接映射方式、全相联映射方式和组相联映射方式。

1. 直接映射方式

直接映射方式是指每个主存块只能映射到 Cache 中的一个特定块，是一种多对一的映射关系，如图 3.26 所示。

直接映射方式实现简单，硬件成本低。缺点是每个主存块在 Cache 中只有一个固定的位置可存放，容易产生冲突，导致频繁地发生替换，降低了命中率。另外，这种方式也使得 Cache 中的空块不能得到充分利用。

2. 全相联映射方式

全相联映射是指每个主存块可以映射到 Cache 中的任意一块，也是一种多对一的映射关系，如图 3.27 所示。

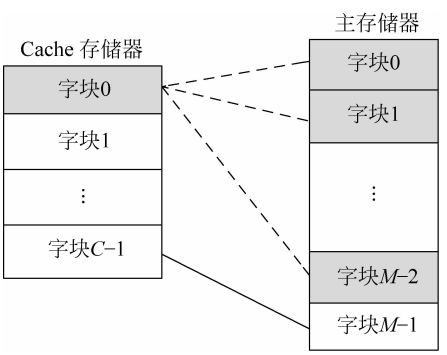


图 3.26　直接映射

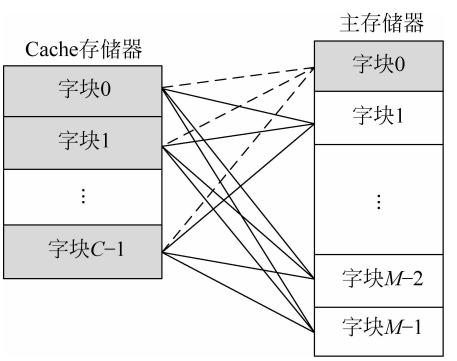


图 3.27　全相联映射

这种映射方式可以从已被占满的 Cache 中替换出任一旧块。显然这种方式更灵活，命中率也更高，减少了冲突。但访问 Cache 时需要和 Cache 所有块逐一比较才能判断出所访问的内容是否在 Cache 中，所需逻辑电路较多，成本较高。

3. 组相联映射方式

组相联映射是指主存的一个块映射到 Cache 的有限的地方，这种映射方式是对直接映射和全相联映射的一种折中。在这种方式下，一个 Cache 分为许多组，在一个组里有两个或多个区块，主存的区块映射到 Cache 的某个对应的组中，但这个区块可能出现在这个组内的任何地方，如图 3.28 所示。

3.6.3　Cache 的替换策略

CPU 在访存时如果没有命中 Cache，就需要选择旧的区块替换出去，保证 Cache 中的内容是当前 CPU 所需的，从而获得较高的 Cache 命中率，这要靠 Cache 替换策略来管理。常用的替换策略有先进先出(FIFO)策略、最不经常使用(LFU)策略、最近最少使用(LRU)策略和随机替换策略。

1. 先进先出策略

先进先出策略总是将最先调入 Cache 的字块替换出来。这种方法开销小，容易实现，但

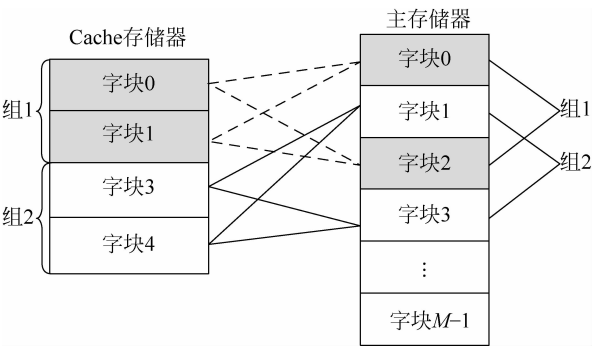


图 3.28　组相联映射

有可能使一些经常访问的块也被替换出去。

2. 最不经常使用策略

最不经常使用策略是将一段时间内被访问次数最少的 Cache 块替换出去。为实现这种策略，需要为 Cache 的每个字块都设置一个计数器，从 0 开始计数，Cache 每命中一次，被命中块的计数器值加 1。当需要替换时，把计数值最小的 Cache 块替换出去，同时将对应的计数器清零。但由于计数值最小的字块有可能是刚换进来的，所以这种方法有可能把刚换进来的块又马上替换出去。

3. 最近最少使用策略

最近最少使用策略是将近期最少访问的块替换出去。为实现这种策略，也需要为 Cache 的每个字块都设置一个计数器，Cache 每命中一次，就把所命中块的计数器清零，其他各块的计数器增 1。当需要替换时，把计数值最大的那块替换出去。这种策略保护了刚复制到 Cache 中的新数据块，保证 Cache 具有较高命中率。

4. 随机替换策略

随机替换策略是随机地选取 Cache 中的一块替换出去。这种策略易于实现，速度较快，但被替换出去的块可能马上又要访问，所以会降低 Cache 的命中率。

3.6.4　Cache 的数据更新方法

前面介绍的 Cache 替换策略主要是针对 Cache 的读操作，但实际应用中 CPU 还需要向 Cache 写入内容。而 Cache 的内容是主存部分内容的副本，写入操作会导致 Cache 与主存的内容不一致，为解决这一问题，需要采用适当的 Cache 数据更新方法来管理。

常用的 Cache 更新方法有写回法、全写法和写一次法。

1. 写回法

当 CPU 对 Cache 写命中时，只修改 Cache 的内容而不立即写入主存，只有此块被换出时才写回主存。对于 Cache 写未命中，由于此后对此块的多次读/写访问的可能性很大，所以写回法的处理是将内存块复制到 Cache 后再对其进行修改，同样也只有该块被换出时才写回主存。

为支持这种策略，每个 Cache 块必须配置一个修改位，以反映此块是否被 CPU 修改过。当某块被换出时，根据此块修改位为 1 还是为 0，决定是否将该块内容写回主存。

这种策略使 Cache 在 CPU—主存之间，不仅在读方向而且在写方向上都起到高速缓存作用。对一 Cache 块的多次写命中都在 Cache 中快速完成修改，只是需被替换时才写回速度较慢的主存，减少了访问主存的次数从而提高了效率。但写回法也存在 Cache/主存不一致性的隐患。

2. 全写法

全写法又称写直达法。是当 Cache 写命中时，Cache 与主存同时发生写操作。这种策略显然较好地维护了 Cache 与主存的内容一致性，当然这并不等于说全部解决了一致性问题；当 Cache 写未命中时，直接向主存写入数据，但此时是否将修改过的主存块取到 Cache，写直达法却有两种选择：一种是将主存块读取到 Cache 并且为它分配一个字块位置，称为 WTWA 法；另一种是不读取主存块，称为 WTNWA 法。前一种方法保持了 Cache/主存的一致性，但操作复杂，而后一种方法操作简化，但命中率降低。

全写法是写 Cache 与写主存同步进行，其优点是 Cache 每块无须设置修改位以及相应的判断逻辑，缺点是 Cache 对 CPU 向主存的写操作无高速缓冲功能，降低了 Cache 的功效。

3. 写一次法

写一次法是基于写回法并结合了全写法的写策略，即写命中和写未命中的处理与写回法基本相同，只是第一次写命中时要同时写入主存。这种策略主要用于某些处理器的片内 Cache，如 Pentium 处理器的片内数据 Cache 就采用的是写一次法。

习　　题

1. 半导体存储器主要有哪几类？它们的技术指标是什么？

2. 简述 DRAM 和 SRAM 的工作原理。

3. 什么叫刷新？刷新有几种方法？什么是“死时间”？为什么说分散式刷新方式可能会产生存储器的频繁刷新现象？

4. 若刷新周期为 8ms，说明 1M×1 位 DRAM 芯片的刷新方法。

5. 有一个 16K×16 位的存储器，由 1K×4 的内部结构为 64×64 的 DRAM 芯片构成，则：(1)采用异步刷新方式时，如果刷新周期为 2ms，则相邻两行之间的刷新间隔是多少？(2)采用集中刷新方式时，如果存储周期为 0.5μs，则存储器刷新一遍最少需要多少个存储周期？死区占多少时间？

6. 内存地址从 60000H～CFFFFH 共有多少个地址？

7. 下列存储器芯片各需要多少条地址线进行寻址？每个芯片有多少条数据线？

(1) 64K×4 位；(2) 1M×8 位；(3) 128K×16 位

8. 假定系统需要 8KB 容量的静态 RAM 和 8KB 容量的 EPROM，选用 1K×4 位的 2114 静态 RAM 芯片和 4K×8 位的 2732 EPROM 芯片，各需要多少片？

9. 存储器扩展中，每次寻址时能够同时有多位片选信号有效？为什么？

10. 在图 3.21 中，如果改用 A_{11} 和 A_{12} 作为译码输入，则各存储芯片的地址范围是多少？

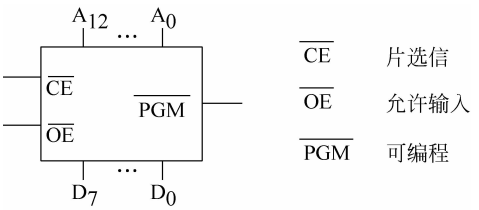
11. 某 8 位机系统地址线有 16 条，若使用 16K×4 位的 RAM 芯片组成存储器，试问：

(1) 该机所允许的最大主存空间是多少？

(2) 共需要多少片 16K×4 位的 RAM 芯片?

(3) 画出此存储器的组成框图。

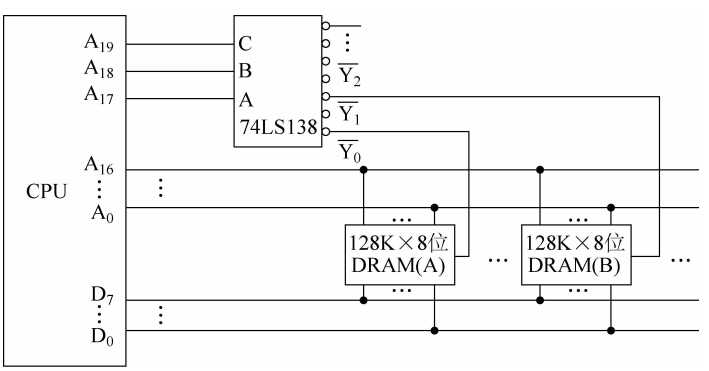
12. 设 CPU 有 20 根地址线,8 根数据线。现有 2764 EPROM (8K × 8 位),外特性如下:



试用 74LS138 译码器及其他门电路(门电路自定)画出 CPU 和 2764 的连接图。要求地址为 F0000H～FFFFFH,并写出每片 2764 的地址范围。

13. 试为某 8 位机系统设计一个具有 8KB 的 ROM 和 40KB 的 RAM 的存储器。要求 ROM 用 2732 EPROM 芯片组成,从 0 地址开始; RAM 用 6264 SRAM 芯片组成,从 4000H 地址开始。

14. 已知某 1M×8 位的存储器由若干 128K×8 位的 DRAM 芯片组成,如下图所示:



请回答以下问题:

(1) 每个 128K×8 位 DRAM 芯片的片内地址线和数据线各有多少条?

(2) 每个 128K×8 位 DRAM 芯片的片内地址范围是多少?

(3) 该 1M×8 位的存储器需要多少个 128K×8 位 DRAM 芯片组成?

(4) 图中 A 和 B 两个 DRAM 芯片各自的物理地址范围是多少?

15. 简述全译码方式、部分译码方式和线性译码方式的原理和优、缺点。

16. 用全译码方式将 2K×8 位的存储芯片连接成具有 8K×8 位容量的存储器。要求用 74LS138 译码器译码,并使存储空间从 48000H 开始且连续。试画出 CPU 与存储器连接接口电路图。

17. 什么是 Cache? Cache 的地址映射方式有哪些? 替换算法有哪些? 数据更新方法有哪些?