第5章 存 储 器

视频讲解

5.1 存储器概述

存储器(memory)是用来存储程序和数据的部件,是计算机系统中必不可少的组成部分。从与 CPU 的关系来看,可分为内存储器和外存储器。内存储器(内存,也称主存)通常由半导体存储器组成,它直接与 CPU 的外部总线相连,是计算机主机的组成部分,用来存放当前正在执行的数据和程序,是本章主要讨论的内容。外存储器(外存,也称辅存)位于主机外面,它通过接口逻辑电路与主机相连接,是作为计算机的外部设备来配置的。外存用来存放暂时不用的那些程序和数据,使用时必须先调入内存才能执行。常用的外存有硬盘、光盘、U 盘等。

5.1.1 半导体存储器的分类

从存储器的工作特点、作用等角度来看,半导体存储器分类如图 5.1 所示。

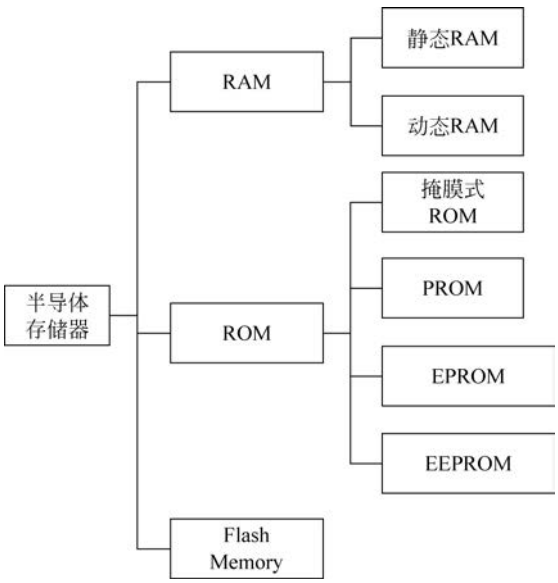

图 5.1 半导体存储器的分类

1. RAM

RAM(随机存取存储器)的特点是存储器中的信息既可以读又可以写,且对存储器中的任一存储单元进行读写操作的时间大致相同。RAM 中的信息在掉电后立即消失,是一种

易失性存储器。RAM 分为静态 RAM(SRAM)和动态 RAM(DRAM)两种。

1) SRAM

SRAM 是利用触发器的两个稳定状态表示“1”和“0”,最简单的 TTL 电路组成的 SRAM 存储单元由两个双发射极晶体管和两个电阻构成的触发器电路组成;而 MOS 管组成的 SRAM 存储单元由 6 个 MOS 管构成的双稳态触发电路组成。SRAM 的特点是只要电源不掉电,写入 SRAM 的信息就不会丢失,同时对 SRAM 进行读操作时不会破坏原有信息。SRAM 的功耗较大,容量较小,存取速度快,不需要刷新,常用于容量较小的单板机、工业控制等小系统中。

2) DRAM

DRAM 是利用 MOS 管的栅极对其衬底间的分布电容来保存信息的。DRAM 的每个存储单元所需要的 MOS 管较少,典型的由单管 MOS 组成,因此 DRAM 的集成度较高、功耗低;缺点是 MOS 管栅极对其衬底间的分布电容上的电荷会随着电容器的放电过程而逐渐消失。一般地,信息在 DRAM 中保存的时间为 2ms 左右。为了保存 DRAM 中的信息,每隔 1～2ms 要对其刷新一次。刷新的过程就是“读出”的过程,由读出再生电路完成。因此采用 DRAM 的计算机必须配置刷新电路。另外,DRAM 的存储器速度较慢,容量较大,且功耗低。DRAM 广泛应用于内存容量较大的微型机系统,如 PC 的内存。

2. ROM

ROM(Read Only Memory,只读存储器)的特点是用户在使用时只能读出信息,不能写入新的信息,存储信息断电后不会丢失。ROM 用来存放固定的应用程序、系统软件、监控程序、常数表格等。ROM 按写入信息方式的不同可分为以下几种。

1) 掩膜式 ROM

掩膜式 ROM(Masked ROM)存储单元中的信息由 ROM 制造厂家在生产时使用掩膜式工艺将信息一次性写入,其内部信息不再能更改,所以也称固定存储器。它适用于大批量生产。

2) PROM

PROM(Programmable ROM)中的程序和数据是由用户使用专门的编程器自行一次性写入的,一旦写入就无法更改。

3) EPROM

EPROM(Erasable Programmable ROM)也是由用户使用专门的编程器自行写入程序和数据,但写入后的信息可用紫外线照射芯片的石英窗口来擦除,芯片中的信息全部擦除后可再重新写入新的内容。EPROM 可以多次擦除、多次写入。

4) EEPROM

EEPROM(Electrically Erasable Programmable ROM)是可用电信号进行擦除和写入信息的存储器,芯片不离开插件板便可擦除部分或全部信息和写入其中信息。EEPROM 为经常需要修改程序和参数的应用领域提供了极大的方便,但存取速度较慢,价格较贵。

3. Flash Memory

Flash Memory(闪速存储器)又称为快闪存储器,简称闪存,是一种新型的可读/写、非易失性半导体存储器,近年来技术发展极快。Flash Memory 芯片具有功耗低、集成度高、体积小、可靠性高等特点。目前,Flash Memory 广泛应用于便携式计算机的 PC 卡存储器(固

态硬盘),以及用来存放主板上的BIOS以代替原来的EPROM的BIOS。现在的移动设备普遍用Flash Memory来存储信息。对于需要实时代码或数据更新的嵌入式系统应用,Flash Memory也是一种理想的存储器。

近年来采用Flash Memory制作的"U盘"已广泛应用在台式机和便携机中替代软盘,成为大容量、高速度的移动式存储器。

5.1.2 半导体存储器的主要性能指标

半导体存储器的主要性能指标有存储容量和存取速度。

1. 存储容量

一个半导体存储器芯片的存储容量是指存储器可以容纳的二进制信息量。一般存储器都采用一维线性编址,存储器中每个存储单元都被赋予一个地址。因此,存储器的存储容量与存储器中存储地址数有关,与每个存储单元的位数有关,即存储器的存储容量$=2^M \times N$,其中M是存储器的地址数,N是存储器的每个存储单元的位数。表示存储器容量的单位通常用字节(B)表示,也可用位(b)表示。这里有1B=8b。

例如,某存储器芯片的地址数为16位,存储字长为8位,则该存储器的存储容量为$2^{16} \times 8$b=64KB=512Kb。

2. 存储速度

存储器的存储速度可以用两个时间参数表示:一个是"存取时间"T_A(Access time),定义为从启动一次存储操作,到完成该操作所经历的时间;另一个是"存储周期"T_M(Memory cycle),定义为启动两次独立的存储器操作之间所需的最小时间间隔。通常存储周期T_M略大于存取时间T_A。

5.1.3 典型的半导体存储器芯片

1. 静态RAM芯片HM6116

HM6116是一种2048×8位的高速静态CMOS随机存取存储器,其引脚排列如图5.2所示。

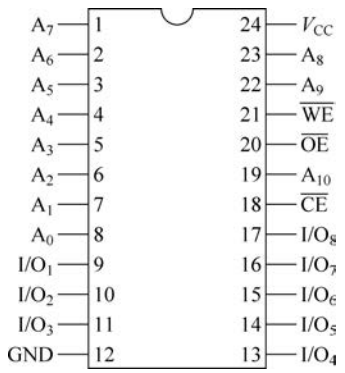

图5.2 HM6116的引脚排列

HM6116片内有16 384(即16K)个存储单元,排列成128×128的矩阵,构成2K个字,字长为8位,可构成2KB的内存。该芯片有11条地址线$A_0 \sim A_{10}$、8条数据线$I/O_1 \sim I/O_8$,还有3条控制线:片选信号$\overline{CE}$,用来选择芯片;写允许信号$\overline{WE}$控制读/写操作,当$\overline{WE}$为高电平时,为读出,当$\overline{WE}$为低电平时,为写入;输出允许信号$\overline{OE}$,用来把输出数据输出到数据线。

常用的静态RAM芯片有2114(1K×4b)、6116(2K×8b)、6264(8K×8b)、62128(16K×8b)、62256(32K×8b)、62512(64K×8b)及更大容量的HM628128(128K×8b)和HM628512(512K×8b)等。

2. 动态 RAM 芯片 Intel 2164A

Intel 2164A 是 64K×1b 的动态随机存取存储器，其引脚排列如图 5.3 所示。

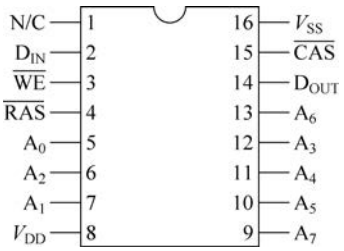

图 5.3　Intel 2164A 的引脚排列

Intel 2164A 的片内有 64K（65 536）个内存单元，有 64K 个存储地址，每个存储单元存储一位数据，片内要寻址 64K 个单元，需要 16 条地址线，为了减少封装引脚，地址线分为行地址和列地址两部分，芯片的地址引脚只有 8 条，片内有地址锁存器，可利用外接多路开关，由行地址选通信号 $\overline{RAS}$ 将先送入的 8 位行地址送到片内行地址锁存器，然后由列地址选通信号 $\overline{CAS}$ 将后送入的 8 位列地址送到片内列地址锁存器。16 位地址信号选中 64K 存储单元中的一个单元。

Intel 2164A 的数据线是输入和输出分开的，则 $\overline{WE}$ 信号控制读/写。当 $\overline{WE}$ 为高电平时，为读出，所选中单元的内容经过输出三态缓冲器，从 D_{OUT} 引脚读出；当 $\overline{WE}$ 为低电平时，为写入，D_{IN} 引脚上的内容经过输入三态缓冲器，对选中单元进行写入。

Intel 2164A 芯片无专门的片选信号，一般行选通信号和列地址选通信号也起到了片选的作用。

常用的动态 RAM 典型芯片有 2116（16K×1b）、2164（64K×1b）、21256（256K×1b）、21010（1M×1b）、21014（256K×4b）、44100（1M×4b）等。

3. 只读存储器芯片 Intel 2732A

Intel 2732A 是一种 4K×8b 的 EPROM，其引脚排列如图 5.4 所示。

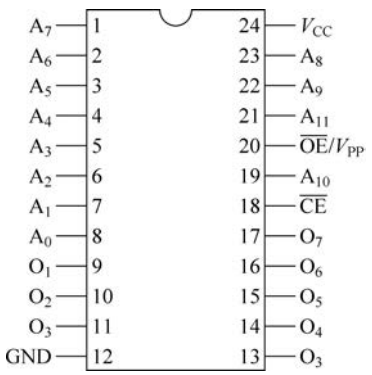

图 5.4　Intel 2732A 的引脚排列

Intel 2732A 的存储容量为 4K×8b，有 12 条地址线 A_{11}～A_0，8 条数据线 O_7～O_0。2 个控制信号中 $\overline{CE}$ 为芯片允许信号，用来选择芯片；$\overline{OE}/V_{PP}$ 为输出允许信号及编程电源输入线。当 $\overline{CE}$ 为低电平时，若 $\overline{OE}/V_{PP}$ 也为低电平，对存储器进行读操作，把输出数据送上数据线；若 $\overline{OE}/V_{PP}$ 加上 21V 编程电压，则对存储器重新编程。当 $\overline{CE}$ 为高电平时，Intel 2732A 处于待用状态（又称为静止等待方式）。

常用的 EPROM 芯片有 2716（2K×8b）、2732（4K×8b）、2764（8K×8b）、27128（16K×8b）、27256（32K×8b）、27512（64K×8b）等。

5.1.4　存储器的分级结构

现在存储器的衡量指标主要从以下三方面：速度、容量、价格。计算机对存储器的要求是价格低、速度快、容量大，需要尽可能满足这三个标准。但是，一般都是速度越快，价格越高，因此容量一般也比较小。为了尽可能满足计算机的运算速度和高容量的要求，现有的存储系统中都设置了高速缓冲存储器（Cache），并采用虚拟存储技术构成三级存储系统。微机系统中存储器的结构如图 5.5 所示。

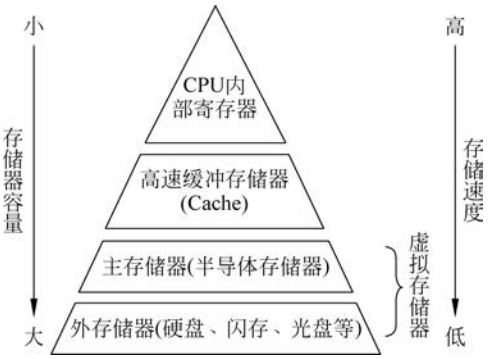

图 5.5　微机系统中存储器的结构

Cache 采用静态半导体存储器,具有速度快、容量小的特点。Cache 位于 CPU 和主存储器(以下简称主存)之间,用于匹配两者的速度,达到高速存取指令和数据的目的。现有的 Cache 系统一般都设置了 3 级以上,其中一级 Cache 都集成到了 CPU 内部。

Cache 和主存构成了微机系统的 Cache-主存系统。CPU 读取数据都是先从 Cache 中读取,如果 Cache 不存在,才去访问主存。如果能够尽可能地保证 CPU 每次都能从 Cache 中读取到数据,从 CPU 的角度看,CPU 的主存的速度接近 Cache,容量则接近于主存。

主存和外存构成了虚拟存储器系统,主要目的是从逻辑上实现对内存容量的扩充。从整体上看,其速度接近于主存的速度,其容量则接近于外存的容量。

计算机存储系统的这种多层次结构,很好地解决了容量、速度、成本三者之间的矛盾。这些不同速度、不同价格、不同容量的存储器,利用了硬件、软件或软硬件结合的技术连接起来,组成了一个完整的存储系统。

视频讲解

5.2　半导体存储芯片结构及使用

5.2.1　半导体存储器的基本结构

计算机系统中内存储器的基本结构如图 5.6 所示。图 5.6 中虚线框内为半导体存储器,其中存储体是存储单元的集合体,容量为 $2^M \times N$; MAR 为存储器地址寄存器,用来存放 CPU 所访问的存储单元的地址; MDR 为存储器数据寄存器,用来存放 CPU 对 MAR 地址所指存储单元的读/写操作的数据。内存储器通过 M 位地址线、N 位数据线和相关的控制线与 CPU 交换信息。

CPU 对内存储器的访问只有读操作、写操作。当 CPU 启动一次存储器读操作时,CPU 先将地址通过地址总线送入 MAR,然后使读相关的控制信号有效,MAR 中的地址码经过地址译码器译码后选中该地址对应的存储单元,并通过读/写驱动器将选中单元的数据送入 MDR,最后通过数据总线读入 CPU。

当 CPU 启动一次存储器写操作时,CPU 先将地址通过地址总线送入 MAR,再将数据通过数据总线送入 MDR,然后使写相关的控制信号有效,MAR 中的地址码经过地址译码器译码后选中该地址对应的存储单元,并通过读/写驱动器将 MDR 的数据送入选中的存储单元。

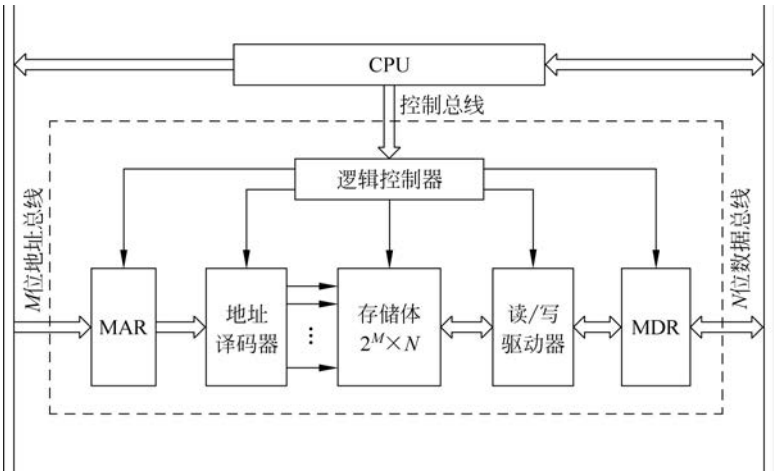

图 5.6　存储器的基本结构

5.2.2　半导体存储芯片的使用

1. 半导体存储芯片的使用概述

CPU 对存储器芯片的使用，是通过软件和硬件协调工作完成的。在软件方面，CPU 执行相应的指令实现对存储器的访问。例如，8086/8088 CPU 读存储器操作可用指令 MOV AL,[1000H]；写存储器操作可用指令 MOV [1000H],AL。在硬件方面，可以将存储器芯片的地址线、数据线、控制线与 CPU 的地址总线、数据总线、控制总线直接相连接，并采用译码电路产生存储器芯片的片选信号，实现 CPU 与半导体存储芯片的正确连接。

由 8088 CPU 组成的 8 位存储器系统如图 5.7 所示。存储器芯片同 CPU 的连接是构筑 8 位存储器子系统的主要工作，有如下三部分内容。

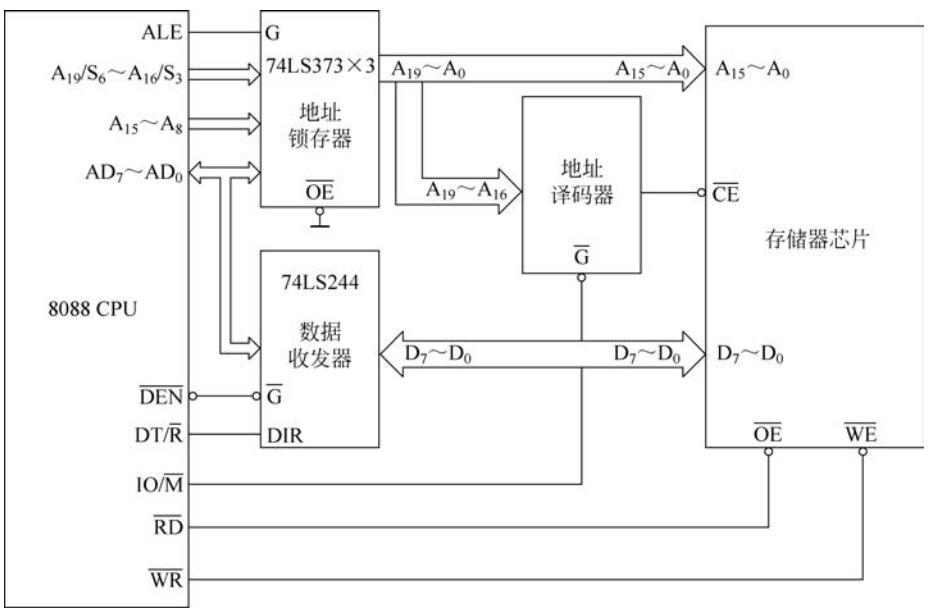

图 5.7　8088 CPU 组成的 8 位存储器系统

1) 地址线的连接

可以根据所选用的半导体存储器芯片地址线的多少,把CPU的地址线两部分:一部分通常是低位的地址线直接和存储器芯片的地址线相连,用于芯片内的地址译码,选中该存储器芯片的一个存储单元;另一部分通常是剩余的高位地址线经另加的地址译码器,产生存储器芯片的片选信号,用来选中CPU所要访问的存储器芯片。图5.7中,对存储器芯片而言,片内地址线为$A_{15}\sim A_0$,直接与CPU的地址线$A_{15}\sim A_0$相连接。CPU未连接的4位高位地址$A_{19}\sim A_{16}$即剩余地址线,经地址译码器译码后输出作为存储器芯片的片选信号。

2) 数据线的连接

图5.7中,存储器芯片有8条数据线,可直接同8088 CPU的8位数据线相连。

3) 控制线的连接

从图5.7中可见,CPU对存储器读/写操作的控制信号主要有$\overline{WR}$、$\overline{RD}$和IO/$\overline{M}$,前两个可直接同存储器芯片的控制信号线$\overline{WE}$和$\overline{OE}$连接,IO/$\overline{M}$可作为地址译码器的使能端实现对存储器的读/写操作。

2. 存储器芯片与CPU连接时应考虑的问题

存储器芯片与CPU连接时,原则上可以将存储器芯片的地址线、数据线、控制线与CPU的地址总线、数据总线、控制总线直接相连接,但在实际操作时,必须考虑以下几个问题。

1) CPU的负载能力

在微机系统中,由于CPU通过总线与多片存储器芯片及I/O接口相连接,这些芯片有的是TTL器件,有的是MOS器件,因此在构成系统时,CPU能否支持其负载是必须考虑的问题。当总线上连接的器件超出CPU的负载能力时,应设法提高总线的驱动能力。采取的措施通常有增加缓冲器或总线驱动器等。

2) 各种信号线的连接

由于CPU的各种信号要求与存储器的各种信号要求有所不同,往往需要增加一些必要的辅助电路。

(1) 数据线。存储器芯片的数据线可与CPU的数据线直接相连。CPU的数据总线是双向的,而存储器的数据线有输入输出共用和分开两种结构。对于输入输出共用的数据线,由于其芯片内部有三态驱动器,故可直接与CPU的数据总线相连。而对于输入和输出分开的数据线,则要外加三态门,才能与CPU的数据总线相连。

(2) 地址线。存储器的地址线一般可直接与CPU的地址总线相连。对于大容量的动态RAM,为了减少封装引线的数目,往往采用分时输入的方式。这时,需要在CPU与存储器芯片之间增加多路转换开关,用$\overline{CAS}$和$\overline{RAS}$分别将高位地址及低位地址送入存储器。

(3) 控制线。存储器的控制信号主要有读写信号$\overline{OE}$、$\overline{WE}$及片选信号$\overline{CS}$,对于动态存储器芯片,还需要地址分时控制信号$\overline{CAS}$和$\overline{RAS}$等。

3) CPU的时序与存储器存取速度的配合

CPU在取指令和存储器读写操作时,其时序是固定的。因此,可根据CPU的时序来选择存储器。对存取速度较慢的存储器,还需增加等待周期T_W,以满足快速CPU的要求。

4) 存储器的地址分配及片选信号的产生

内存通常分RAM和ROM两大部分,RAM又分系统区(监控程序及操作系统占用的

区域)和用户区。所以,合理地对内存进行分配是非常重要的。此外,目前生产的存储器芯片的容量仍然是有限的,所以常常需要许多片存储器芯片才能组成一个存储器系统,这就要求正确选择片选信号。

通常,片选信号由 CPU 剩余的高位地址线译码产生,译码电路既可采取集成电路地址译码器,也可采用基本门电路实现。

(1) 采用集成电路地址译码器。

在微机系统中,常采用中规模集成电路芯片 74LS138 作为地址译码器,其引脚如图 5.8 所示。

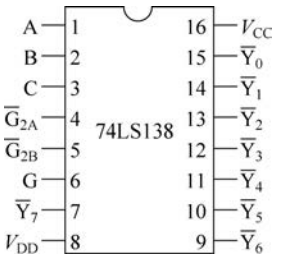

图 5.8　74LS138 译码器引脚

74LS138 是 3 线—8 线译码器/分配器,有 3 个选择输入端 C、B、A,3 个使能输入端 G_1、$\overline{G}_{2A}$、$\overline{G}_{2B}$,以及 8 个输出端 $\overline{Y}_0 \sim \overline{Y}_7$。其逻辑功能表如表 5.1 所示。当 3 个使能输入端 $G_1=1$、$\overline{G}_{2A}=0$、$\overline{G}_{2B}=0$ 同时成立时 74LS138 才进行译码工作,此时 8 个输出端 $\overline{Y}_0 \sim \overline{Y}_7$ 仅有一个有效,由 3 个选择输入端 C、B、A 对应的二进制编码确定。

表 5.1　74LS138 译码器逻辑功能

输　入						输　出							
$\overline{G}_{2A}$	$\overline{G}_{2B}$	G	C	B	A	$\overline{Y}_0$	$\overline{Y}_1$	$\overline{Y}_2$	$\overline{Y}_3$	$\overline{Y}_4$	$\overline{Y}_5$	$\overline{Y}_6$	$\overline{Y}_7$
1	×	×	×	×	×	1	1	1	1	1	1	1	1
×	1	×	×	×	×	1	1	1	1	1	1	1	1
×	×	0	×	×	×	1	1	1	1	1	1	1	1
0	0	1	0	0	0	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	0	1	1	1	1	1
0	0	1	0	1	1	1	1	1	0	1	1	1	1
0	0	1	1	0	0	0	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	0	1	1
0	0	1	1	1	0	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	0

(2) 采用基本门电路实现地址译码。

对于存储芯片较少的存储器,可以采用基本门电路组成的逻辑电路来实现片选控制。典型的或门和与非门如图 5.9 所示。图中两逻辑电路的逻辑关系完全等价,读者可自行验证。

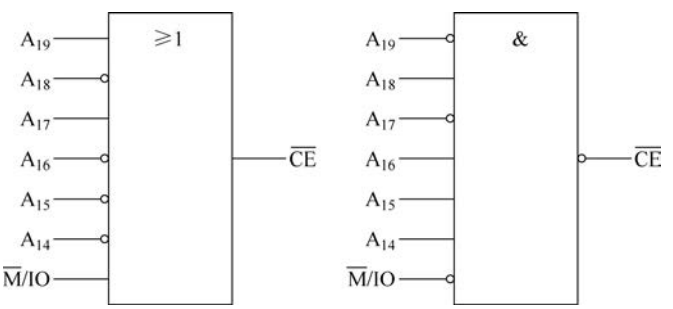

图 5.9　典型的基本门电路实现地址译码

5.3 16位/32位系统的存储器接口

16位及以上CPU,为了向下兼容8位、16位CPU指令,通常一个存储器单元仍采用8位。这样,在其存储器系统设计时,既要保证字节的读/写功能,又要保证字、双字读/写时序的效率,就要考虑存储单元地址对齐问题。地址对齐(adjustment)是指16位CPU(8086)数据的存放地址为2的倍数,即偶数($A_0=0$),32位CPU(80386)数据的存放地址为4的倍数($A_1=A_0=0$),以此类推,以保证数据读/写时序的效率。这种思想体现在以下8086/80386存储器系统的设计应用中。

1. 8086 CPU存储器系统的奇偶分体

8086存储器系统中1MB的存储器地址空间分成两个512KB的存储体——"偶存储体"和"奇存储体",图5.10为8086系统中奇偶分体存储器和CPU的连接图。

图5.10中用3片74LS373地址锁存器和2片74LS245数据收发器分离8086的地址数据复用线$AD_{15}\sim AD_0$和地址状态复用线$A_{19}/S_6\sim A_{16}/S_3$形成20位地址总线AB和16位数据总线DB。偶存储体同8086的低8位数据总线$D_7\sim D_0$相连接,奇存储体同8086的高8位数据总线$D_{15}\sim D_8$相连接,地址总线的$A_{19}\sim A_1$同两个存储体中的地址线$A_{18}\sim A_0$相连接,A_0不用在对存储器的存储单元寻址上,A_0和$\overline{BHE}$被用作对存储器奇偶字库的选择。A_0接偶存储体的$\overline{CE}$,$A_0=0$时选中偶存储体。$\overline{BHE}$信号与奇存储体$\overline{CE}$端相连,$\overline{BHE}=0$时选中奇存储体。当8086对地址对齐的字(对准字)进行读/写操作时,$A_0=0$且$\overline{BHE}=0$,可在一个总线周期内完成读/写操作,而8086对非对准字(奇地址开始的字)进行读/写操作时,需要两个总线周期才可完成。显然,软件应该使用对准字才能显现8086 CPU存储器系统的奇偶分体硬件结构的效率。

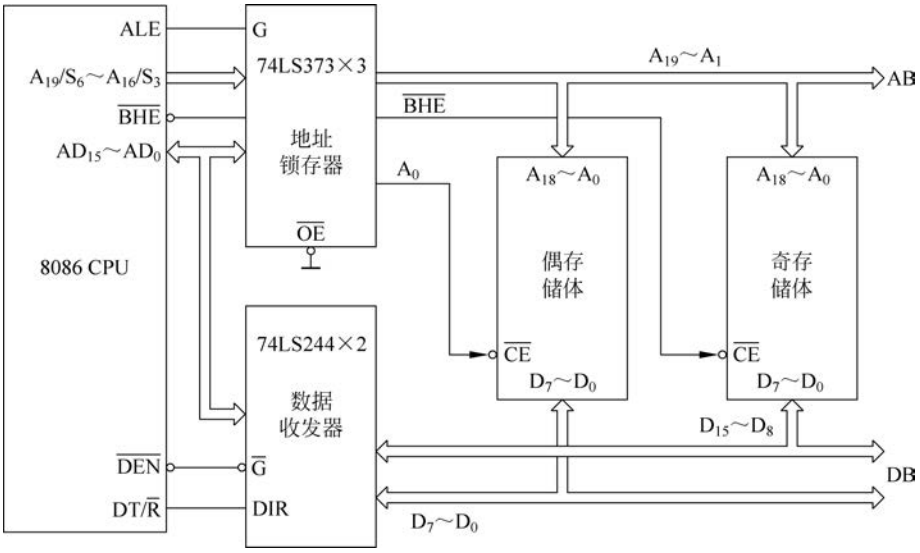

图5.10 8086 CPU存储器系统的奇偶分体

2. 80386 CPU 存储器系统的分体

80386 存储器系统中 4GB 的存储器地址空间分成 4 个 1GB 的存储体，图 5.11 为 80386 CPU 存储器系统分体连接。

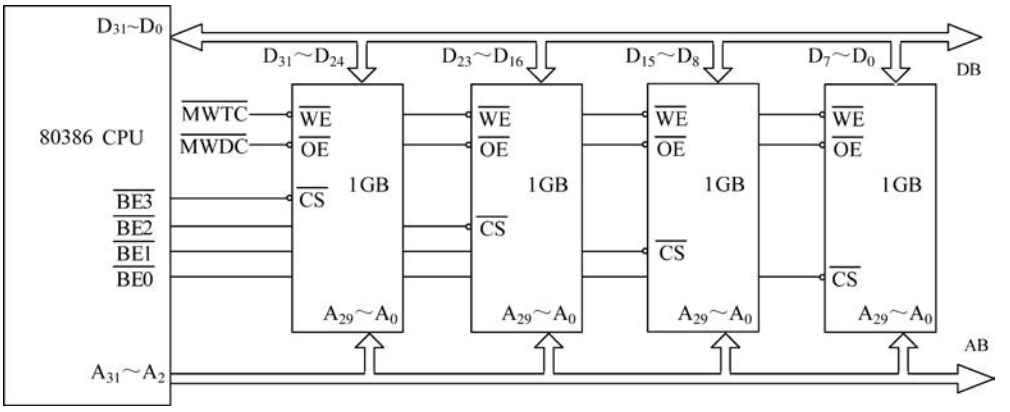

图 5.11　80386 CPU 存储器系统的分体连接

利用高 30 位地址线 $A_{31}\sim A_2$ 和 4 字节使能信号 $\overline{BE3}\sim\overline{BE0}$ 来实现字节、字、双字的数据传送。$\overline{BE3}\sim\overline{BE0}$ 信号由 CPU 内部根据地址线 $A_1\sim A_0$ 产生。4 个 1GB 的存储体的地址单元分别是 $4n+0$、$4n+1$、$4n+2$、$4n+3$ 对应连接在 80386 数据总线 $D_7\sim D_0$、$D_{15}\sim D_8$、$D_{23}\sim D_{16}$、$D_{31}\sim D_{24}$ 上，当 80386 进行双字对准字（地址为 4 的倍数）数据传送时，$\overline{BE3}\sim\overline{BE0}$ 全部有效，可一次完成双字 32 位数据的读/写操作。

视频讲解

5.4　存储器容量的扩展

存储芯片的容量是有限的，实际系统需要更大存储容量时，就必须采用多片现有的存储器芯片构成较大容量的存储器模块，这就是存储器的扩展。扩展存储器的方法有三种：位扩展、字扩展、字位扩展。

5.4.1　位扩展

当存储芯片的存储字的数量满足需要，而存储字长（存储单元的位数）不满足需要时，就需要增加存储字长，即进行位扩展。

如果采用 Intel 2164 芯片，因该芯片 64K×1b 芯片，必须由 8 片才能构成 64KB 的内存，如图 5.12 所示。

位扩展时，各存储芯片的信号线的连接方法如下。

（1）芯片的地址线全部并联，并与 CPU 地址总线中相应的地址线相连。

（2）芯片的数据线分高低位分别与 CPU 数据总线的相应位连接。

（3）芯片的读/写控制信号线并联，接 CPU 控制总线中的读/写控制线；芯片的片选信号并联，可接 CPU 控制总线中的存储器选择信号，也可接地址线高位或地址译码器输出端。

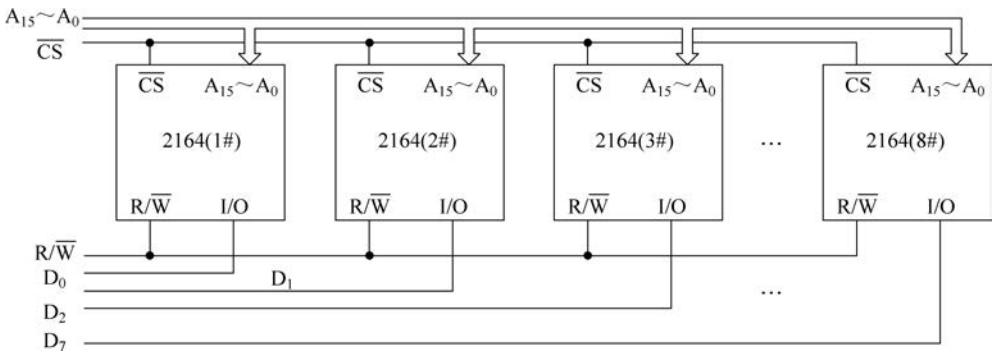

图 5.12　62K×1b 芯片位扩展组成 64K×8b 的内存

5.4.2　字扩展

当存储芯片的存储字长(存储单元的位数)满足需要,而存储字的数量不满足需要时,就需要增加存储字的数量,即进行字扩展。

字扩展时,各存储芯片的信号线的连接方法如下。

(1) 芯片的地址线全部并联,并与 CPU 地址总线中相应的地址线相连。

(2) 芯片的数据线全部并联,并与 CPU 数据总线中相应的数据线连接。

(3) 芯片的读/写控制信号线并联,接 CPU 控制总线中的读/写控制线;芯片的片选信号分别接地址线高位或地址译码器输出端。

其中,关键是如何产生存储器芯片的片选信号。在存储器系统中,实现片选控制的方法有三种,即线选法、部分译码法和全译码法。

1. 线选法

线选法是指利用地址总线的高位地址线直接作为存储器芯片的片选信号,低位地址线和存储器地址相连。采用线选法需保证每次寻址时只能有一个片选信号有效。

线选法的优点是结构简单,缺点是地址空间浪费大。由于部分地址线未参与译码,会出现大量地址重叠。此外,当通过线选的芯片增多时,还可能出现地址空间不连续的情况。

图 5.13 为 16 位系统线选法字扩展的例子,图中 4 片 6264 芯片(8K×8b,SRAM),分为两组:奇片和偶片。

由图 5.13 可知,奇片用 $\overline{\text{BHE}}$ 作为片选,8 位数据线 D_7～D_0 连接到 8086 数据总线 D_{15}～D_8 上;偶片用 A_0 作为片选,8 位数据线 D_7～D_0 连接到 8086 数据总线 D_7～D_0 上。6264 芯片地址线 13 条 A_{12}～A_0 连接到 8086 地址总线 A_{13}～A_1 上。所以图 5.13 中 1#芯片为偶片、2#芯片为奇片构成第 1 组,3#芯片为偶片、4#芯片为奇片构成第 2 组。根据线选法产生片选原则选择高位地址中的 A_{14} 和 A_{15} 作为 2 组存储器芯片的片选。故第 1 组中 1#芯片(偶片)的片选是 $A_{14}=0$ 和 $A_0=0$ 同时成立,2#芯片(奇片)的片选是 $A_{14}=0$ 和 $\overline{\text{BHE}}=0$ 同时成立,图 5.12 中用或门实现上述逻辑。

表 5.2 给出图 5.13 中 4 片 6264 芯片的地址范围。A_{19}～A_{16} 因未参与对 2 组芯片的片选控制,故其值可以是 0 或 1(用×表示任取),这里假定取为全 0,显然 2 组芯片有大量重叠区。考虑 2 组芯片不能被同时选中,所以地址中不允许出现 $A_{15}A_{14}=00$ 的情况。

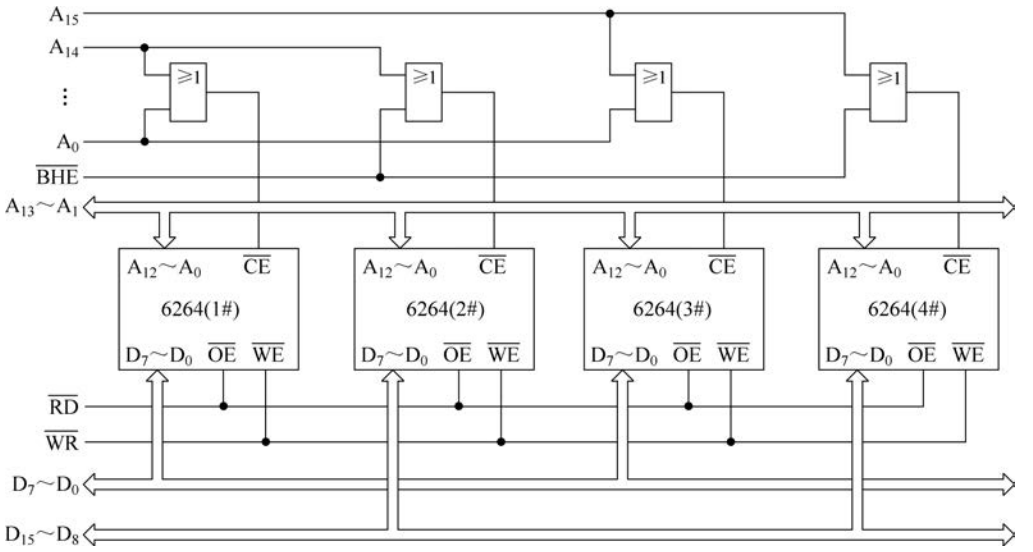

图 5.13　16 位系统线选法字扩展接线

表 5.2　图 5.13 存储器芯片的地址范围

组　别		高位地址线						低位地址线		地址范围
		A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}～A_1	A_0	
第 1 组	1#(偶片)	×	×	×	×	1	0	0000000000000 ～ 1111111111111	0 ～ 0	08000H ～ 0BFFEH
	2#(奇片)	×	×	×	×	1	0	0000000000000 ～ 1111111111111	1 ～ 1	08001H ～ 0BFFFH
第 2 组	3#(偶片) 4#(奇片)	×	×	×	×	0	1	0000000000000 ～ 1111111111111	× ～ ×	04000H ～ 07FFFH

2. 部分译码法

部分译码法是将高位地址线中部分地址进行译码，产生存储器的片选信号。对被选中的芯片而言，未参与译码的高位地址线可以为 0，也可以为 1，即每个存储单元将对应多个地址。使用时一般将未用地址设为 0。采用部分译码法，可简化译码电路，但由于地址重叠，会造成系统地址空间资源的部分浪费。

图 5.14 所示的电路，是采用部分译码法用 4 片 2732 芯片(4K×8b，EPROM)组成的 16 位存储器系统。

图 5.14 中 4 片 2732 芯片分为两组，每组 2 片(奇片和偶片)，其中 1# 和 3# 为偶片，2# 和 4# 为奇片，1# 和 2# 构成第一组，3# 和 4# 构成第二组。2732 芯片地址线 12 条 A_{11}～A_0 连接到 8086 地址总线 A_{12}～A_1 上。部分译码法采用高位地址的部分 A_{16}、A_{15}、A_{14}、A_{13} 经 2 片 74LS138 进行译码，74LS138 的 $\overline{G}_{2B}$ 使能端一个接 A_0，另一个接 $\overline{BHE}$。

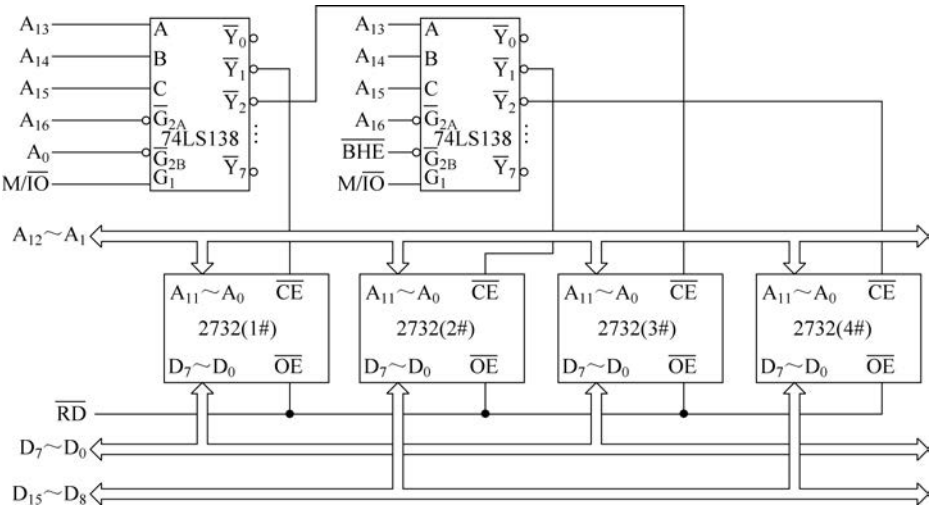

图 5.14　16 位系统部分译码法字扩展接线

表 5.3 给出图 5.14 中 4 片 2732 芯片的地址范围。部分译码时,未使用高位地址线 A_{19}、A_{18} 和 A_{17}。也就是说,这 3 位无论是什么,对芯片寻址都没有影响。所以,每个芯片将同时具有 $2^3=8$ 个可用且不同的地址范围(即重叠区)。

表 5.3　图 5.14 存储器芯片的地址范围

组　别		高位地址线							低位地址线		地址范围
		A_{19}	A_{18}	A_{17}	A_{16} $\overline{G}_{2A}$	A_{15} C	A_{14} B	A_{13} A	$A_{12}\sim A_1$	A_0	
第 1 组	1#(偶片) 2#(奇片)	×	×	×	0	0	0	1	000000000000 ～ 111111111111	× ～ ×	02000H ～ 03FFFH
第 2 组	3#(偶片) 4#(奇片)	×	×	×	0	0	1	0	000000000000 ～ 111111111111	× ～ ×	04000H ～ 05FFFH

3. 全译码法

全译码法是指将地址总线中除片内地址以外的全部剩余高位地址参加译码,产生各存储芯片的片选信号。采用全译码法,每个存储单元的地址都是唯一的,不存在地址重叠,但译码电路较复杂,连线也较多。

图 5.15 所示的电路是采用全译码法用 6 片 62512 芯片(64K×8b,SRAM)组成的 16 位存储器系统。

图 5.15 中 6 片 62512 芯片分为 3 组,每组 2 片(奇片和偶片)。62512 芯片地址线 16 条 $A_{15}\sim A_0$ 连接到 8086 地址总线 $A_{16}\sim A_1$ 上。8086 的高位地址是 $A_{19}\sim A_{17}$,全部经 2 片 74LS138 进行译码,A_0、$\overline{BHE}$ 分别接 2 片 74LS138 的使能端。表 5.4 给出图 5.15 中 6 片 62512 芯片的地址范围。

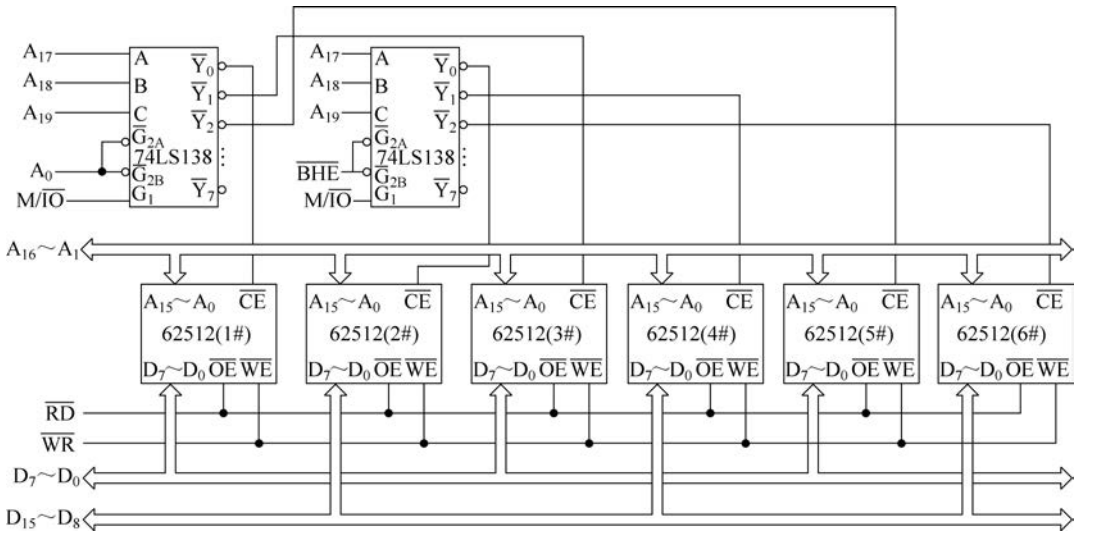

图 5.15　16 位系统全译码法字扩展接线

表 5.4　图 5.15 中 62512 芯片的地址范围

组　别		高位地址线			低位地址线		地址范围
		A_{19} C	A_{18} B	A_{17} A	A_{16}～A_1	A_0	
第 1 组	1#(偶片) 2#(奇片)	0	0	0	0000000000000000 ～ 1111111111111111	× ～ ×	00000H ～ 1FFFFH
第 2 组	3#(偶片) 4#(奇片)	0	0	1	0000000000000000 ～ 1111111111111111	× ～ ×	20000H ～ 3FFFFH
第 3 组	5#(偶片) 6#(奇片)	0	1	0	0000000000000000 ～ 1111111111111111	× ～ ×	40000H ～ 5FFFFH

5.4.3　字位扩展

当存储芯片的存储字长(存储单元的位数)和存储字的数量都不满足需要时,字和位都需要进行扩展,即进行字位扩展。字位扩展是字扩展和位扩展的组合,字位全扩展时,先把存储芯片分组,然后可以采取组内位扩展、组间字扩展或者采取组内字扩展、组间位扩展两种方案。

某 8088 微机系统采取组内位扩展、组间字扩展方案,要求使用 2114 芯片(1K×4b)若干,扩展 4K×8b 存储器模块,且扩展后的存储空间为从 8F000H 开始的连续存储区。扩展后的存储器模块如图 5.16 所示。

图 5.16 中 2114 所需芯片数=(4K×8b)/(1K×4b)=8 片。组内位扩展:2114 芯片每两片一组,组成 4 组按字节存取的 1KB 容量的存储器,4 位数据线分别与 CPU 数据总线的 D_3～D_0 及 D_7～D_4 分别相连接,地址线、控制线并联。组间字扩展:4 组存储器的地址线 A_9～A_0、数据线 D_7～D_0 并联后分别连接到 8088 总线的低位地址 A_9～A_0 和数据线 D_7～D_0

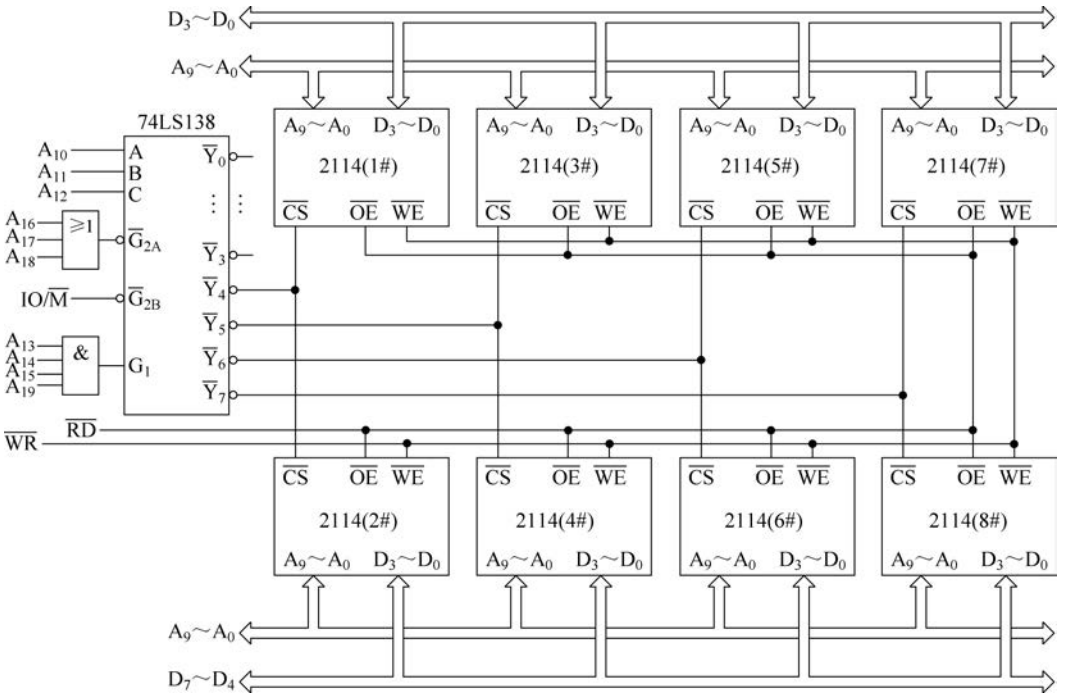

图 5.16　8088 系统字位扩展的接线

上；高位地址线 A_{19}～A_{10} 全译码后为 4 组存储器组提供片选信号，全译码用一片 74LS138 和逻辑门实现，A_{12}、A_{11}、A_{10} 分别连接在 74LS138 的 C、B、A 上，8088 的 IO/$\overline{M}$ 连接到 74LS138 的 $\overline{G}_{2B}$ 上，A_{18}、A_{17}、A_{16} 通过或门连接在 74LS138 的 $\overline{G}_{2A}$ 上，A_{19}、A_{15}、A_{14}、A_{13} 通过与门连接在 74LS138 的 G_1 上。$\overline{Y}_4$、$\overline{Y}_5$、$\overline{Y}_6$、$\overline{Y}_7$ 分别作为 4 组存储器的片选 $\overline{CS}$，可满足起始地址为 8F000H。4 组存储器的地址分配情况如表 5.5 所示。

表 5.5　图 5.16 中 4 组存储器的地址分配情况

组　别		高位地址线				低位地址线	地址范围
		A_{19}～A_{13}	A_{12} C	A_{11} B	A_{10} A	A_9～A_0	
第 1 组	2114(1#) 2114(2#)	1000111	1	0	0	0000000000 ～ 1111111111	8F000H ～ 8F3FFH
第 2 组	2114(3#) 2114(4#)	1000111	1	0	1	0000000000 ～ 1111111111	8F400H ～ 8F7FFH
第 3 组	2114(5#) 2114(6#)	1000111	1	1	0	0000000000 ～ 1111111111	8F800H ～ 8FBFFH
第 4 组	2114(7#) 2114(8#)	1000111	1	1	1	0000000000 ～ 1111111111	8FC00H ～ 8FFFFH

5.4.4 存储器芯片与8086 CPU的连接举例

8086 CPU与半导体存储器芯片的接口如图5.17所示，其中存储器芯片1#～8#为SRAM芯片62512(64K×8b)；9#～16#为EPROM芯片2732(4K×8b)。试分析该接口电路的特性，计算RAM区和ROM区的地址范围(内存为字节编址)。

1. 电路分析

1) 8086 CPU是16位微处理器，存储区须奇偶分体

图5.17中1#、3#、5#和7#这4个RAM芯片的8位数据线接数据总线D_7～D_0，2#、4#、6#和8#这4个RAM芯片的数据线接D_{15}～D_8。所以，前者4片构成偶存储体，后者4片构成奇存储体。同理，ROM区中9#、11#、13#和15#构成偶存储体，10#、12#、14#和16#构成奇存储体。RAM区中8片分为4组，ROM区中8片分为4组。

2) 8086 CPU的双重复用总线经过锁存器和数据收发器分离出地址总线、数据总线

图5.17中采用了3片74LS373和2片74LS245。74LS373有两个控制端G和$\overline{OE}$，G为锁存允许信号，接8086 CPU的ALE(地址锁存允许)。ALE为高电平，将双重复用总线中的$\overline{BHE}/S_7$和AD_{15}～AD_0以及A_{19}/S_6～A_{16}/S_3地址信息存入74LS373，$\overline{OE}$接地，保证74LS373输出有效的地址总线A_{19}～A_0和$\overline{BHE}$。74LS245有两个控制端$\overline{G}$和DIR，$\overline{G}$为使能端，接8086 CPU的$\overline{DEN}$(数据允许)。$\overline{DEN}$为低电平，实现双重复用总线AD_{15}～AD_0中的数据信息与74LS245的连接；DIR为方向端，接8086 CPU的DT/$\overline{R}$(数据发送、接收)，当DT/$\overline{R}$=DIR=1时，8086经74LS245发送数据；当DT/$\overline{R}$=DIR=0，8086经74LS245接收数据。

3) RAM区

图5.17中RAM的译码电路由2片74LS138(17#和18#)构成，62512芯片地址线16条A_{15}～A_0，考虑8086是16位存储器系统，连接到8086地址总线A_{16}～A_1上。高位地址A_{19}～A_{17}同17#和18#的C、B、A连接，采用全译码法。74LS138(17#和18#)的3个使能端G_1、$\overline{G}_{2A}$、$\overline{G}_{2B}$中，17#和18#的G_1与8086的M/$\overline{IO}$相连，8086存储器读/写时M/$\overline{IO}$=1，G_1为有效电平。17#的$\overline{G}_{2A}$、$\overline{G}_{2B}$与地址总线A_0相连，所以17#芯片负责产生RAM的偶存储体1#、3#、5#和7#芯片的片选信号。18#的$\overline{G}_{2A}$、$\overline{G}_{2B}$与地址总线$\overline{BHE}$相连，所以18#芯片负责产生RAM的奇存储体2#、4#、6#和8#芯片的片选信号。17#芯片和18#芯片的输出端$\overline{Y}_0$分别接1#芯片和2#芯片的片选线，故1#芯片和2#芯片为一组。同理，3#芯片和4#芯片为第2组，4#芯片和5#芯片为第3组，6#芯片和7#芯片为第4组，同组的两个芯片前者是偶片，后者为奇片。62512芯片的写控制端$\overline{WE}$和读控制端$\overline{OE}$分别与8086的写控制信号$\overline{WR}$和读控制信号$\overline{RD}$相连。

4) ROM区

图5.17中ROM的译码电路由1片74LS138(19#)构成，2732芯片地址线12条A_{11}～A_0，考虑8086是16位存储器系统，连接到8086地址总线A_{12}～A_1上。高位地址为A_{19}～A_{13}，其中A_{15}、A_{14}、A_{13}同19#芯片的C、B、A连接。19#芯片的3个使能端G_1、$\overline{G}_{2A}$、$\overline{G}_{2B}$中，G_1与8086的M/$\overline{IO}$相连；$\overline{G}_{2A}$与8086的读控制信号$\overline{RD}$相连，当8086运行存储器读指令时使能；$\overline{G}_{2B}$与与非门的输出相连，与非门的4个输入是地址总线A_{19}～A_{16}，显然A_{19}=

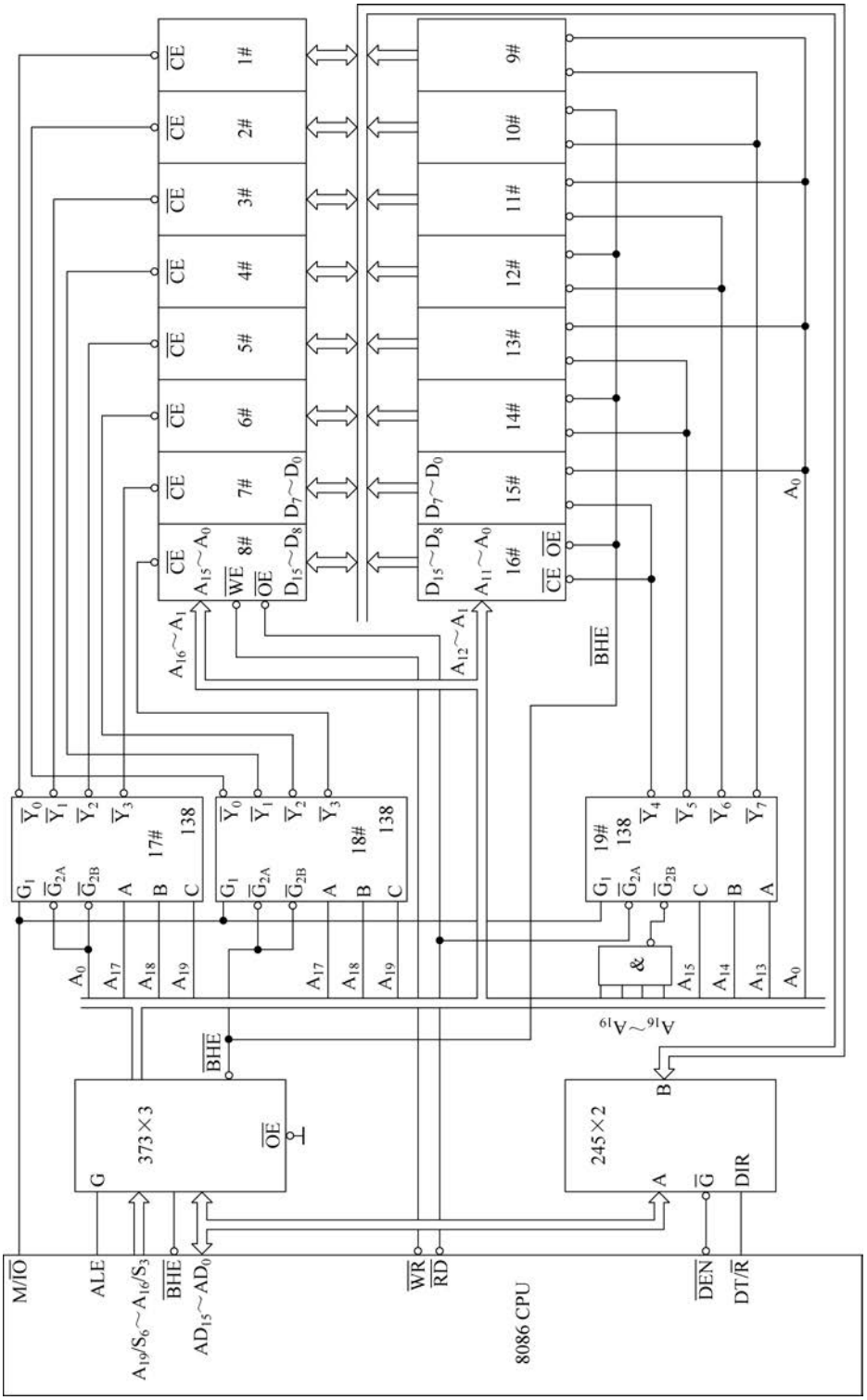

图 5.17　8086 CPU 与半导体存储器芯片的接口实例

$A_{18}=A_{17}=A_{16}=0$ 时，$\overline{G}_{2B}$ 使能。A_0 负责 ROM 的偶存储体 9＃、11＃、13＃和 15＃芯片的片选。$\overline{BHE}$ 负责 ROM 的奇存储体 10＃、12＃、14＃和 16＃芯片的片选。综上所述，高位地址 A_{19}～A_{13} 全部唯一确定为全译码法。

2. RAM 区的地址范围

图 5.17 中 RAM 区各个芯片的地址范围如表 5.6 所示，整个 RAM 区的地址范围为 00000H～7FFFFH，共占 512KB。

表 5.6　RAM 区各个芯片的地址范围

组　别		高位地址线			低位地址线		地址范围
		A_{19} C	A_{18} B	A_{17} A	A_{16}～A_1	A_0	
第 1 组	1＃(偶片) 2＃(奇片)	0	0	0	0000000000000000 ～ 1111111111111111	× ～ ×	00000H ～ 1FFFFH
第 2 组	3＃(偶片) 4＃(奇片)	0	0	1	0000000000000000 ～ 1111111111111111	× ～ ×	20000H ～ 3FFFFH
第 3 组	5＃(偶片) 6＃(奇片)	0	1	0	0000000000000000 ～ 1111111111111111	× ～ ×	40000H ～ 5FFFFH
第 4 组	7＃(偶片) 8＃(奇片)	0	1	1	0000000000000000 ～ 1111111111111111	× ～ ×	60000H ～ 7FFFFH

3. ROM 区的地址范围

图 5.17 中 ROM 区各个芯片的地址范围如表 5.7 所示，整个 ROM 区的地址范围为 F8000H～FFFFFH，共占 32KB。

表 5.7　ROM 区各个芯片的地址范围

组　别		高位地址线							低位地址线		地址范围
		A_{19}	A_{18}	A_{17}	A_{16}	A_{15} C	A_{14} B	A_{13} A	A_{12}～A_1	A_0	
第 1 组	9＃(偶片) 10＃(奇片)	1	1	1	1	1	1	1	000000000000 ～ 111111111111	× ～ ×	FE000H ～ FFFFFH
第 2 组	11＃(偶片) 12＃(奇片)	1	1	1	1	1	1	0	000000000000 ～ 111111111111	× ～ ×	FC000H ～ FDFFFH
第 3 组	13＃(偶片) 14＃(奇片)	1	1	1	1	1	0	1	000000000000 ～ 111111111111	× ～ ×	FA000H ～ FBFFFH
第 4 组	15＃(偶片) 16＃(奇片)	1	1	1	1	1	0	0	000000000000 ～ 111111111111	× ～ ×	F8000H ～ F9FFFH