第3章 微处理器及其系统

在微处理器领域，Intel 系列 CPU 产品一直占着主导地位。尽管 8086/8088 后续的 80286、80386、80486 以及 Pentium 系列 CPU 结构与功能已经发生很大的变化，但从基本概念与基本结构以及指令格式上来讲，它们仍然是经典的 8086/8088 CPU 的延续与提升。并且，其他系列流行的 CPU（如 AMD 公司的 6x86 MX/M Ⅱ 等）也可以与 80x86 CPU 兼容。

本章着重介绍 Intel 8086/8088 微处理器及其指令系统；同时也简要介绍 80286、80386、80486 与 Pentium 系列微处理器的结构特点及其技术精髓。

3.1 8086/8088 微处理器

8086 是 Intel 系列的 16 位微处理器。在推出 8086 之后不久，Intel 公司还推出了准 16 位微处理器 8088。8088 的内部寄存器、运算器以及内部数据总线与 8086 一样都是按 16 位设计的，但其外部数据总线只有 8 条。这样设计的目的主要是为了与 Intel 原有的 8 位外围接口芯片直接兼容。

3.1.1 8086/8088 CPU 的内部结构

8086/8088 CPU 的内部结构基本上是相似的，为了简化，在图 3.1 中只绘制了 8086 CPU 的内部功能结构框图。由图 3.1 可知，8086/8088 CPU 内部可分为两个独立的功能单元，即总线接口单元（Bus Interface Unit，BIU）和执行单元（Execution Unit，EU）。

1. 总线接口单元

BIU 是与总线连接的接口部件，其基本功能是根据执行单元的请求负责 CPU 与存储器或 I/O 端口之间的数据传送。在 CPU 取指令时，它从内存中取出指令送到指令队列缓冲器；而在执行指令时，它要与指定的内存单元或者 I/O 端口交换数据。

BIU 内有 4 个 16 位段寄存器，即 CS（代码段寄存器）、DS（数据段寄存器）、SS（堆栈段寄存器）和 ES（附加段寄存器），16 位指令指针 IP，6 字节指令队列缓冲器，20 位地址加法器和总线控制电路。

1）指令队列缓冲器

8086 的指令队列由 6 个字节的寄存器组成，最多可存入 6 个字节的指令代码，而 8088 的指令队列只有 4 个字节。在 8086/8088 执行指令时，将从内存中取出 1 条或几条指令，依次放在指令队列中。它们采用“先进先出”的原则，按顺序存放，并按顺序取到 EU 中去执行。其操作将遵循下列原则。

(1) 取指令时，每当指令队列中存满 1 条指令后，EU 就立即开始执行。

(2) 当指令队列中空出 2 个(对 8086)或 1 个(对 8088)指令字节时，BIU 便自动执行取指操作，直到填满为止。

(3) EU 在执行指令的过程中，若 CPU 需要访问存储器或 I/O 端口，则 EU 自动请求 BIU 去完成访问操作。此时若 BIU 空闲，则会立即完成 EU 的请求；否则 BIU 首先将指令取至指令队列，再响应 EU 的请求。

(4) 当 EU 执行完转移、调用和返回指令时，则要清除指令队列缓冲器，并要求 BIU 从新的地址重新开始取指令，新取的第 1 条指令将直接经指令队列送到 EU 去执行，随后取来的指令将填入指令队列缓冲器。

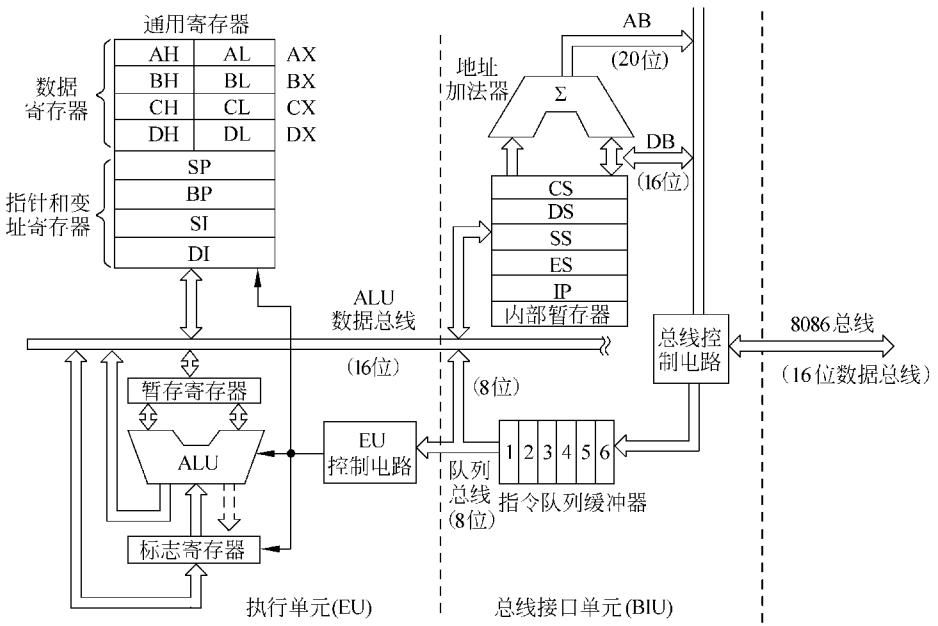


图 3.1　8086/8088 CPU 的内部功能结构框图

2）地址加法器和段寄存器

8086 有 20 根地址线，但内部寄存器只有 16 位，不能直接提供对 20 位地址的寻址信息。如何实现对 20 位地址的寻址呢？这里采用了一种称为“段加偏移”的重要技术，即采用将可移位的 16 位段寄存器与 16 位偏移地址相加的办法，从而巧妙地解决了这一矛盾。具体地说，就是利用各段寄存器分别来存放确定各段的 20 位起始地址的高 16 位段地址信息，而由 IP 提供或由 EU 按寻址方式计算出寻址单元的 16 位偏移地址(又称为逻辑地址或偏移量)，然后，将它与左移 4 位后的段寄存器的内容同时送到地址加法器进行相加，

最后形成一个 20 位的实际地址(又称为物理地址),以对存储单元寻址。图 3.2 所示为实际地址的产生过程。例如,要形成某指令码的实际地址,就要将 IP 的值与代码段寄存器(Code Segment,CS)左移 4 位后的内容相加。

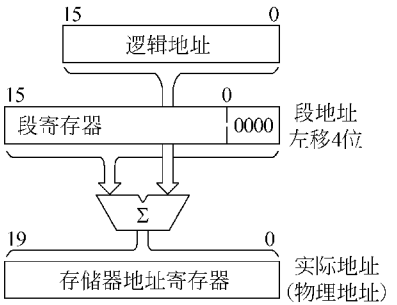


图 3.2 物理地址(实际地址)的产生过程

【例 3.1】 假设 CS=4000H,IP=0300H,则指令的物理地址 PA = 4000H × 16 + 0300H = 40300H。

3) 16 位指令指针

IP(Instruction Pointer)的功能与 8 位 CPU 中的 PC 类似。正常运行时,IP 中含有 BIU 要取的下一条指令(字节)的偏移地址。IP 在程序运行中能自动加 1 修正,使之指向要执行的下一条指令(字节)。有些指令(如转移、调用、中断和返回指令)能使 IP 值改变,或将 IP 值压进堆栈保存,或由堆栈弹出恢复原值。

2. 执行单元

EU 的功能是负责执行指令,执行的指令从 BIU 的指令队列中取得,执行指令的结果或执行指令所需要的数据,都由 EU 向 BIU 发出请求,再由 BIU 经总线控制电路对存储器或 I/O 端口存取。EU 由下列 5 个部分组成。

(1) 16 位算术逻辑单元(ALU):它可以用于进行算术、逻辑运算,也可以按指令的寻址方式计算出寻址单元的 16 位偏移量。

(2) 16 位标志寄存器 F:它用来反映 CPU 运算的状态特征或存放控制标志。

(3) 数据暂存寄存器:它协助 ALU 完成运算,暂存参加运算的数据。

(4) 通用寄存器组:它包括 4 个 16 位数据寄存器,即 AX、BX、CX、DX 和 4 个 16 位指针与变址寄存器,即 SP、BP 与 SI、DI。

(5) EU 控制电路:它是控制、定时与状态逻辑电路,接收从 BIU 中指令队列取来的指令,经过指令译码形成各种定时控制信号,对 EU 的各个部件实现特定的定时操作。

EU 中所有的寄存器和数据通道(除队列总线为 8 位外)都是 16 位的宽度,可实现数据的快速传送。

注意:由于 BIU 与 EU 分开独立设计,因此,在一般情况下,CPU 执行完一条指令后就可以立即执行下一条指令。16 位 CPU 这种并行重叠操作的特点,提高了总线的信息传输效率和整个系统的执行速度。

8088 CPU 的内部结构与 8086 的基本相似,只是 8088 的 BIU 中指令队列长度为 4 字节;8088 的 BIU 通过总线控制电路与外部交换数据的总线宽度是 8 位,总线控制电路与专用寄存器组之间的数据总线宽度也是 8 位。

3.1.2 8086/8088 的寄存器结构

对于微机应用系统的开发者来说,最重要的是掌握 CPU 的编程结构或程序设计模型。8086/8088 的内部寄存器编程结构如图 3.3 所示。它共有 13 个 16 位寄存器和 1 个

只用了 9 位的标志寄存器。其中，阴影部分与 8080/8085 CPU 相同。

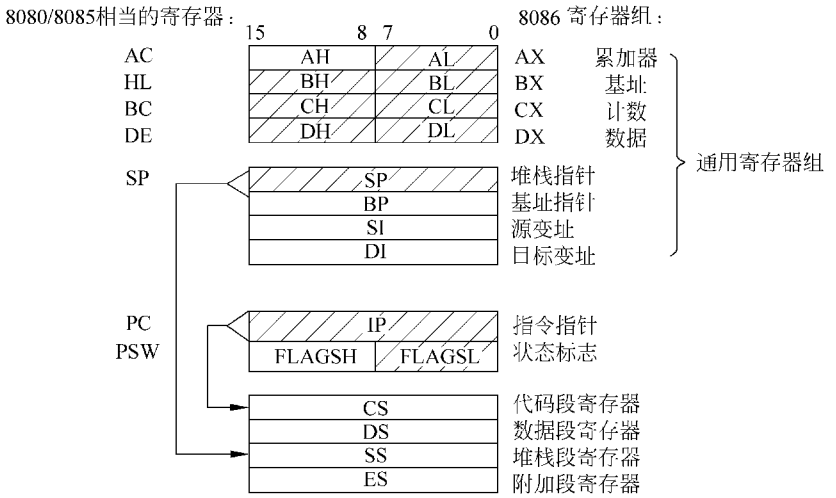


图 3.3　8086/8088 的编程结构

下面将根据寄存器的功能分别加以简要说明。

1. 通用寄存器

通用寄存器分为两组，即数据寄存器；指针寄存器和变址寄存器。

(1) 数据寄存器：执行单元(EU)中有 4 个 16 位数据寄存器，即 AX、BX、CX 和 DX。每个数据寄存器分为高字节 H 和低字节 L，它们均可作为 8 位数据寄存器独立寻址，独立使用。

在多数情况下，这些数据寄存器用在算术运算或逻辑运算指令中，用来进行算术逻辑运算。在有些指令中，它们则有特定的用途。这些寄存器在指令中的特定功能是被系统隐含使用的(如表 3.1 所示)。

表 3.1　数据寄存器的隐含使用

寄存器	操　作	寄存器	操　作
AX	字乘，字除，字 I/O	CL	多位移位和旋转
AL	字节乘，字节除，字节 I/O，转换，十进制运算	DX	字乘，字除，间接 I/O
AH	字节乘，字节除	SP	堆栈操作
BX	转换	SI	数据串操作
CX	数据串操作，循环	DI	数据串操作

(2) 指针寄存器和变址寄存器：指针寄存器是指堆栈指针寄存器(SP)和堆栈基址指针寄存器(BP)，简称为 P 组。变址寄存器是指源变址寄存器(SI)和目的变址寄存器(DI)，简称为 I 组。它们都是 16 位寄存器，一般用来存放偏移地址。

SP 和 BP 都用来指示存取位于当前堆栈段中的数据所在的地址，但 SP 和 BP 在使用上有区别。入栈(PUSH)和出栈(POP)指令是由 SP 给出栈顶的偏移地址，故称为堆栈指针寄存器。而 BP 则是存放位于堆栈段中的一个数据区基地址的偏移地址，故称为堆栈

基址指针寄存器。显然，由 SP 所指定的堆栈存储区的栈顶和由 BP 所指定的堆栈段中某一块数据区的首地址是两个不同的意思，不可混淆。

SI 和 DI 是存放当前数据段的偏移地址的。源操作数的偏移地址放于 SI 中，所以 SI 称为源变址寄存器；目的操作数偏移地址存放于 DI 中，故 DI 称为目的变址寄存器。例如，在数据串操作指令中，被处理的数据串的偏移地址由 SI 给出，处理后的结果数据串的偏移地址则由 DI 给出。

2. 段寄存器

8086/8088 CPU 内部设计了 4 个 16 位的段寄存器，用这些段寄存器的内容作为段地址，再由段寄存器左移 4 位形成 20 位的段起始地址，它们通常被称为段基地址或段基址。再利用“段加偏移”技术，8086/8088 就有可能寻址 1MB 存储空间并将其分成为若干个逻辑段，使每个逻辑段的长度为 64KB(它由 16 位的偏移地址限定)。

注意：这些逻辑段可以通过修改段寄存器的内容被任意设置在整个 1MB 存储空间上下浮动。换句话说，逻辑段在存储器中定位以前，还不是微处理器可以真正寻址的实际内存地址，也正因为这样，通常人们就将未定位之前在程序中存在的地址叫做逻辑地址。这个概念对于后面将要讨论的程序“重定位”十分有用。

4 个 16 位段寄存器都可以被指令直接访问。其中，CS 用来存放程序当前使用的代码段的段地址，CPU 执行的指令将从代码段取得；SS 用来存放程序当前所使用的堆栈段的段地址，堆栈操作的数据就在堆栈段中；DS 用来存放程序当前使用的数据段的段地址，一般来说，程序所用的数据就存放在数据段中；ES 用来存放程序当前使用的附加段的段地址，也用来存放数据，但其典型用法是存放处理后的数据。

3. 标志寄存器

8086/8088 的 16 位标志寄存器 F 只用了其中的 9 位作为标志位，即 6 个状态标志位，3 个控制标志位。

如图 3.4 所示，低 8 位 FL 的 5 个标志与 8080/8085 的标志相同。

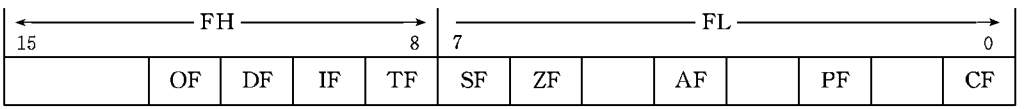


图 3.4　8086/8088 的标志寄存器

状态标志位用来反映算术或逻辑运算后结果的状态，以记录 CPU 的状态特征。下面介绍这 6 个标志位。

(1) CF(Carry Flag)进位标志：当执行一个加法或减法运算使最高位产生进位或借位时，则 CF 为 1；否则为 0。此外，循环指令也会影响它。

(2) PF(Parity Flag)奇偶性标志：当指令执行结果的低 8 位中含有偶数个“1”时，则 PF 为 1；否则为 0。此标志位用于微机中传送信息时，对产生的代码出错情况提供检测条件。此标志在现代程序设计中很少使用；现在，奇偶校验常常由数据通信设备完成，而不是由微处理器完成。

(3) AF(Auxiliary Carry Flag)辅助进位标志：当执行一个加法或减法运算使结果的

低字节的低 4 位向高 4 位(即 D_3 位向 D_4 位)进位或借位时,则 AF 为 1;否则为 0。DAA 和 DAS 指令测试这个特殊标志位,该标志一般用在 BCD 码运算中作为是否需要对 AL 寄存器进行十进制调整的依据。

(4) ZF(Zero Flag)零标志:零标志表示一个算术或逻辑操作的结果是否为 0。若当前的运算结果为 0,则 ZF 为 1;否则为 0。

(5) SF(Sign Flag)符号标志:符号标志保持算术或逻辑运算指令执行后结果的算术符号。它和运算结果的最高位相同。当数据用补码表示时,负数的最高位为 1,正数的最高位为 0。

(6) OF(Overflow Flag)溢出标志:溢出标志用于判断在有符号数进行加法或减法运算时是否可能出现溢出。溢出将指示运算结果已超出机器能够表示的数值范围。当补码运算有溢出时,例如,用 8 位加法将 7FH(+127)加上 01H,结果为 80H(−128)。由于此结果已超出 8 位二进制补码所能表示的最大整数范围(+127),故此时 OF 标志为 1;否则为 0。

注意:对于无符号数的操作,将不会影响溢出标志。

控制标志位有 3 个,用来控制 CPU 的操作,由程序设置或清除。

(1) DF(Direction Flag)方向标志:它用来控制数据串操作指令的步进方向。若用 STD 指令将 DF 置 1,则数据串操作过程中地址会自动递减;若用 CLD 指令将 DF 清零,则数据串操作过程中地址会自动递增。地址的递增或递减由 DI 或 SI 变址寄存器来实现。

(2) IF(Interrupt Enable Flag)中断允许标志:它是控制可屏蔽中断的标志。若用 STI 指令将 IF 置 1,则表示允许 8086/8088 CPU 接收外部从其 INTR 引脚上发来的可屏蔽中断请求信号;若用 CLI 指令将 IF 清零,则禁止 CPU 接收外来的可屏蔽中断请求信号。IF 的状态不影响非屏蔽中断(NMI)请求,也不影响 CPU 响应内部的中断请求。

(3) TF(Trap Flag)跟踪(陷阱)标志:它是为调试程序方便而设置的。若将 TF 标志置为 1,则 CPU 处于单步工作方式;否则,将正常执行程序。

注意:在高型号微处理器中,跟踪(陷阱)标志能够激活芯片上的调试特性(调试程序,以便找到错误或故障),当 TF 标志为 1 时,则微处理器将根据调试寄存器和控制寄存器的指示中断程序流。

最后需要指出的是,8086/8088 所有上述标志位对 Intel 系列后续高型号微处理器的标志寄存器都是兼容的,只不过后者有些增强功能或者新增加了一些标志位而已。

3.1.3 总线周期

总线周期是微处理器操作时所依据的一个基准时间段,通常,它是指微处理器完成一次访问存储器或 I/O 端口操作所需的时间。

对于 8086/8088 CPU 来说,总线周期由 4 个时钟周期组成,这 4 个时钟周期也称为 T_1、T_2、T_3 与 T_4 四个状态;在每一个状态中,CPU 在操作时,总线所处的状态都不同。一般在 T_1 状态,CPU 往多路复用总线上发送寻址的地址信息,以选中某个被寻址的存储器

单元或端口地址；在 T_2 状态，CPU 从总线上撤销地址，为传送数据做准备；在 T_3 状态，多路总线的高 4 位继续提供状态信息，而其低 16 位（对 8086 CPU）或低 8 位（对 8088 CPU）上将出现由 CPU 读入或写出的数据；在 T_4 状态，CPU 采样数据总线，完成本次读或写操作，最后结束总线周期。

注意：不同的 CPU 在一个总线周期内所处的总线状态是不同的；而即使同一个 CPU，其读操作或写操作的具体状态也不相同。一般，在 T_2～T_4，若是写操作，则 CPU 在此期间是先把输出数据送到总线上；若是读操作，则 CPU 在 T_3～T_4 期间将从总线上输入数据。T_2 时复用地址数据总线处于悬空状态，以便使 CPU 有一个缓冲时间把输出地址的写操作转换为输入数据的读操作。

此外，如果存储器或外设的速度较慢，不能及时地跟上 CPU 的速度时，存储器或外设就会通过 READY 信号线在 T_3 状态启动之前向 CPU 发一个“数据未准备好”信号，并且，CPU 会在 T_3 之后自动插入一个或多个等待状态 T_W，以等待存储器或外设准备好传送数据。只有在存储器或外设准备就绪时，它们才又通过 READY 的信号线向 CPU 发出一个有效的“准备好”信号，CPU 接收到这一信号后，才会自动脱离 T_W 状态而进入 T_4 状态。

总线周期只用于 CPU 取指和它同存储器或 I/O 端口交换数据；否则，总线接口单元 BIU 将不和总线“打交道”，即系统总线处于空闲状态，即执行空闲周期，这时，虽然 CPU 对总线进行空操作，但 CPU 内部的执行单元 EU 仍在进行操作，如逻辑运算单元 ALU 仍在进行运算，内部寄存器之间也在传送数据。

图 3.5 所示为一个典型的总线周期序列。

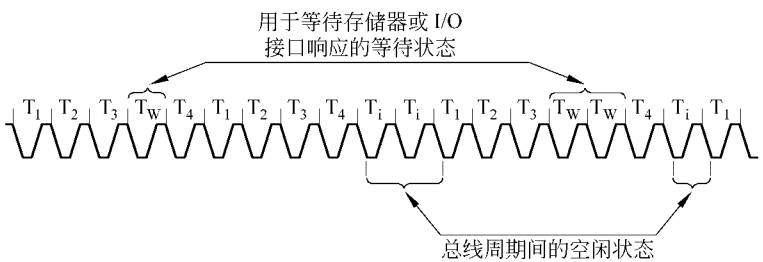


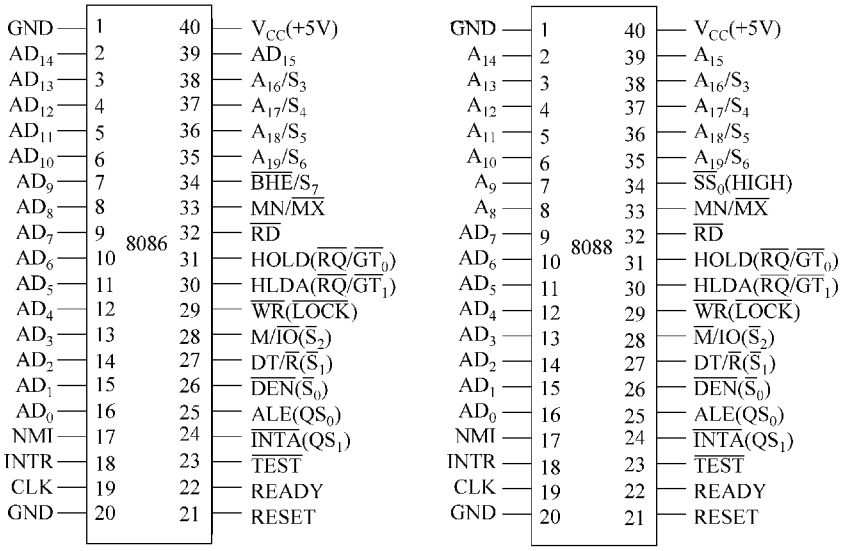
图 3.5　一个典型的总线周期序列

3.1.4　8086/8088 的引脚信号和功能

图 3.6 所示为 8086 和 8088 的引脚信号图。它们的 40 条引线按功能可分为以下 5 类。

1. 地址/数据总线

AD_{15}～AD_0 是分时复用的存储器或端口的地址和数据总线。传送地址时为单向的三态输出，而传送数据时可为双向三态输入/输出。8086/8088 CPU 正是利用分时复用的方法才能使其用 40 条引脚实现 20 位地址、16 位数据及众多控制信号和状态信号的传输。在 8088 中，由于只能传输 8 位数据，因此，只有 AD_7～AD_0 八条地址/数据线，A_{15}～A_8 只用来输出地址。



(a) 8086的引脚信号　　(b) 8088的引脚信号

图 3.6　8086/8088 的引脚信号(括号中为最大方式时的引脚名称)

作为复用引脚,在总线周期的 T_1 状态用来输出要寻址的存储器或 I/O 端口地址;在 T_2 状态浮置成高阻状态,为传输数据作准备;在 T_3 状态,用于传输数据;T_4 状态结束总线周期。当 CPU 响应中断以及与系统总线"保持响应"时,复用线都被浮置为高阻状态。

2. 地址/状态总线

地址/状态总线 $A_{19}/S_6 \sim A_{16}/S_3$ 为输出、三态总线,采用分时输出,即 T_1 状态输出地址的最高 4 位,$T_2 \sim T_4$ 状态输出状态信息。当访问存储器时,T_1 状态时输出的 $A_{19} \sim A_{16}$ 送到锁存器(8282)锁存,与 $AD_{15} \sim AD_0$ 组成 20 位的地址信号;而访问 I/O 端口时,不使用这 4 条引线,即 $A_{19} \sim A_{16}=0$。状态信息中的 S_6 为 0 用来指示 8086/8088 当前与总线相连,所以,在 $T_2 \sim T_4$ 状态,S_6 总等于 0,以表示 8086/8088 当前连在总线上。S_5 表明中断允许标志位 IF 的当前设置。S_4 和 S_3 用来指示当前正在使用哪个段寄存器,如表 3.2 所示。

表 3.2　S_4、S_3 的代码组合和对应的状态

S_4	S_3	状　态
0	0	当前正在使用 ES
0	1	当前正在使用 SS
1	0	当前正在使用 CS 或未用任何段寄存器
1	1	当前正在使用 DS

当系统总线处于"保持响应"状态时,这些引线被浮置为高阻状态。

3. 控制总线

(1) $\overline{BHE}/S_7$:高 8 位数据总线允许/状态复用引脚,三态、输出。$\overline{BHE}$在总线周期的 T_1 状态时输出,S_7 在 $T_2 \sim T_4$ 时输出。在 8086 中,当$\overline{BHE}/S_7$ 引脚上输出$\overline{BHE}$信号时,

表示总线高 8 位 AD_{15}～AD_8 上的数据有效。在 8088 中，第 34 引脚不是$\overline{BHE}/S_7$，而是被赋予另外的信号：在最小方式时，它为$\overline{SS_0}$，和 DT/$\overline{R}$、$\overline{M}$/IO 一起决定了 8088 当前总线周期的读/写动作；在最大方式时，它恒为高电平。S_7 在当前的 8086 芯片设计中未被赋予定义，暂作备用状态信号线。

(2) $\overline{RD}$：读控制信号，三态、输出。当$\overline{RD}$=0 时，表示 CPU 执行存储器或 I/O 端口的读操作。是对内存单元还是对 I/O 端口读取数据，取决于 M/$\overline{IO}$(8086)或 $\overline{M}$/IO(8088)信号。在执行 DMA 操作时，$\overline{RD}$被浮空。

(3) READY："准备好"信号线，输入。该引脚接收被寻址的内存或 I/O 端口发给 CPU 的响应信号，高电平时表示内存或I/O 端口已准备就绪，CPU 可以进行数据传输。CPU 在 T_3 状态开始对 READY 信号采样。若检测到 READY 为低电平，表示内存或 I/O 端口尚未准备就绪，则 CPU 在 T_3 状态之后自动插入等待状态 T_W，直到 READY 变为高电平，内存或 I/O 端口已准备就绪，CPU 才可以进行数据传送。

(4) $\overline{TEST}$：等待测试输入信号，低电平有效。它用于多处理器系统中且只有在执行 WAIT 指令时才使用。当 CPU 执行 WAIT 指令时，它就进入空转的等待状态，并且每隔 5 个时钟周期对该线的输入进行一次测试；若$\overline{TEST}$=1，则 CPU 将停止取下一条指令而继续处于等待状态，重复执行 WAIT 指令，直至$\overline{TEST}$=0 时，CPU 才结束 WAIT 指令的等待状态，继续执行下一条指令。等待期间允许外部中断。

(5) INTR：可屏蔽中断请求输入信号，高电平有效。它为高电平时，表示外设有中断请求，CPU 在每个指令周期的最后一个 T 状态采样此信号。若 IF=1，则 CPU 响应中断，并转去执行中断服务程序。若 IF=0(关中断)，则外设的中断请求被屏蔽，CPU 将不响应中断。

(6) NMI：非屏蔽中断请求输入信号，上升沿触发。此信号不受 IF 状态的影响，只要它一出现，CPU 就会在现行指令结束后引起中断。

(7) RESET：复位输入信号，高电平有效。通常，它与 8284A(时钟发生/驱动器)的复位输出端相连，8086/8088 要求复位脉冲宽度不得小于 4 个时钟周期，而初次接通电源时所引起的复位，则要求维持的高电平不能小于 50μs；复位后，CPU 的主程序流程恢复到启动时的循环待命初始状态，其内部寄存器状态如表 3.3 所示。在程序执行时，RESET 线保持低电平。

表 3.3　复位后内部寄存器的状态

内部寄存器	状　态	内部寄存器	状　态
标志寄存器	清除	SS	0000H
IP	0000H	ES	0000H
CS	FFFFH	指令队列缓冲器	清除
DS	0000H		

(8) CLK：系统时钟，输入。通常与 8284A 时钟发生器的时钟输出端 CLK 相连，该时钟信号的低/高之比常采用 2∶1(占空度为 1/3)。

4. 电源线和地线

电源线 V_{CC}接入的电压为+5V±10%，有两条地线 GND，均应接地。

5. 其他控制线

这些控制线(24～31 引脚)的功能将根据方式控制线 MN/$\overline{MX}$所处的状态而确定。关于这些引脚在最小方式与最大方式下的具体功能差异,将在 3.2 节中给予详细说明。

由上述可知,8086/8088 CPU 引脚的主要特点是:数据总线和地址总线的低 16 位 AD_{15}～AD_0 或低 8 位 AD_7～AD_0 采用分时复用技术。还有一些引脚也具有两种功能,这由引脚 33(MN/$\overline{MX}$)来控制。当 MN/$\overline{MX}$=1 时,8086/8088 工作于最小方式(MN),在此方式下,全部控制信号由 CPU 本身提供。当 MN/$\overline{MX}$=0 时,8086/8088 工作于最大方式。这时,系统的控制信号由 8288 总线控制器提供,而不是由 8086/8088 直接提供。

3.2 8086/8088 系统的最小/最大工作方式

由 8086/8088 CPU 构成的微机系统,有最小方式和最大方式两种系统配置方式。

3.2.1 最小方式

8086 与 8088 构成的最小方式系统区别甚小,现以 8086 最小方式系统为例加以说明。

当 MN/$\overline{MX}$接电源电压时,系统工作于最小方式,即单处理器系统方式,它适合于较小规模的应用。8086 最小方式典型的系统结构如图 3.7 所示。它和 8 位微处理器系统类似,系统芯片可根据用户需要接入。图 3.7 中的 8284A 为时钟发生/驱动器,外接晶体

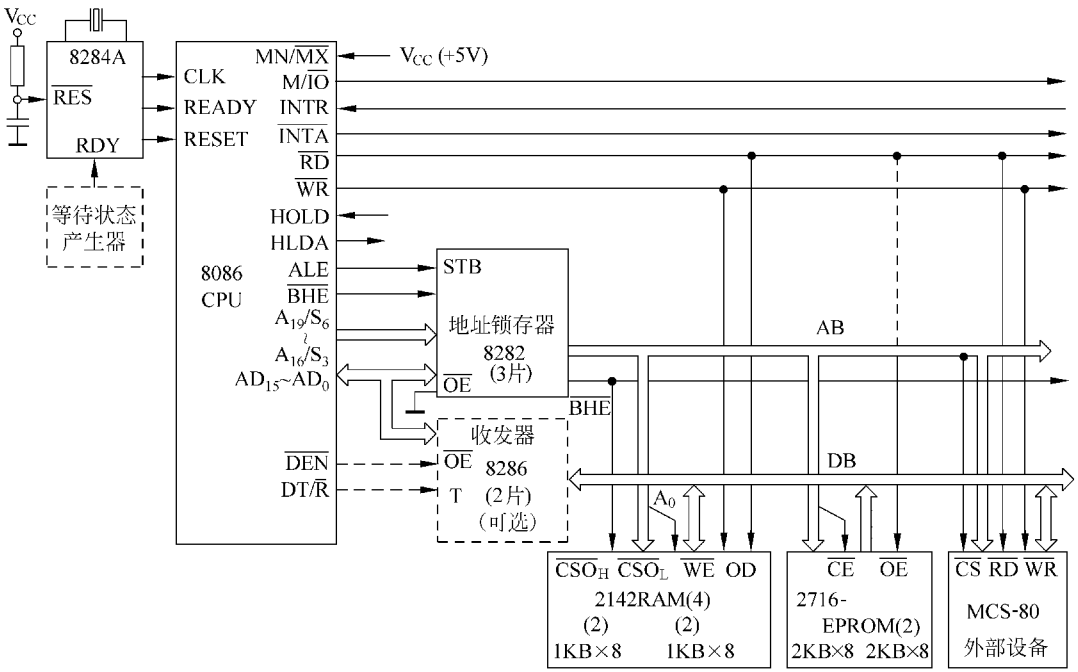


图 3.7 8086 最小方式的典型系统结构示意图