第一章　微型计算机概述

§ 1-1　微型计算机的特点和发展

电子计算机通常按体积、性能和价格分为巨型机、大型机、中型机、小型机和微型机五类。从系统结构和基本工作原理上说，微型机和其他几类计算机并没有本质上的区别，所不同的是微型机广泛采用了集成度相当高的器件和部件，因此带来以下一系列特点：

① 体积小、重量轻　由于采用大规模集成电路(LSI)和超大规模集成电路(VLSI)，使微型机所含的器件数目大为减少，体积大为缩小。50年代要由占地上百平方米、耗电上百千瓦的电子计算机实现的功能，在当今，内部只含几十片集成电路的微型机即已具备。

② 价格低廉　当前，一个典型的16位微型计算机系统 IBM XT（包括 512K 内存、2个5英寸软盘驱动器、1个硬盘和 1 个 CRT）只需1000美元左右。

③ 可靠性高、结构灵活　由于内部元器件数目少，所以连线比较少，这样，使微型机的可靠性较高，结构灵活方便。

④ 应用面广　现在，微型机不仅占领了原来使用小型机的各个领域，而且广泛用于过程控制等新的场合。此外，微型机还进入了过去电子计算机无法进入的部门，如测量仪器、仪表、教学部门、医疗设备、家用电器等。

由于微型机具有上面这些特点，所以它的发展速度大大超过了前几代计算机。自从70年代初第一个微处理器诞生以来，微处理器的性能和集成度几乎每两年提高一倍，而价格却降低一个数量级。

第一个微处理器是1971年美国 Intel 公司生产的4004。它本来是为高级袖珍计算器设计的，但生产出来后，取得了意外的成功。于是，Intel 公司对它作了改进，正式生产了通用的 4 位微处理器4040。Intel 4040以它的体积小、价格低廉等特点引起了许多部门和机构的兴趣。1972年，Intel 公司又生产了 8 位的微处理器8008。通常，人们将 Intel 4004、4040、8008称为第一代微处理器。这些微处理器的字长为 4 位或 8 位，集成度大约为2000管/片，时钟频率为 1MHz，平均指令执行时间约为 20μs。

以后，出现了许多生产微处理器的厂家。1973～1977年间，这些厂家生产了多种型号的微处理器，其中设计最成功、应用最广泛的是 Intel 公司的8080/8085，Zilog 公司的 Z80，Motorola 公司的6800/6802，Rockwell 公司的6502。通常，人们把它们称为第二代微处理器。这些微处理器的时钟频率为 2～4MHz，平均指令执行时间为 1～2μs，集成度超过5000管/片，其中，8085、Z80 和6802 的集成度都高达10000管/片。在这个时期，微处理器的设计和生产技术已相当成熟，配套的各类器件也很齐全。后来，微处理器在以下几个方面得到很大发展：提高集成度、提高功能和速度、增加外围电路的功能和种类。

1977年左右，超大规模集成电路工艺已经成熟，一个硅片上可以容纳几万个管子。

1978～1979年，一些厂家推出了性能可与过去中档小型计算机相比的16位微处理器，这中间，有代表性的3种芯片是Intel的8086/8088，Zilog的Z8000，以及Motorola的M68000。这些微处理器的时钟频率为4～8MHz，平均指令执行时间为0.5μs，集成度为20000～60000管/片。人们将这些微处理器称为第一代超大规模集成电路的微处理器。

1980年以后，半导体生产厂家继续在提高电路的集成度、速度和功能方面取得了很大进展，相继出现Intel 80286、Motorola 68010这样一些集成度高达100000管/片、时钟频率为10MHz左右、平均指令执行时间约为0.2μs的16位高性能微处理器。1983年以后，又生产出Intel 80386和Motorola 68020。这两者都是32位的微处理器，时钟频率达16～20MHz，平均指令执行时间约为0.1μs，集成度高达150000～500000管/片。由这类微处理器构成的整机通常称为超级微型机。

国内于1974年开始研制微型计算机。1979年仿制成由4片集成电路构成的仿8080微处理器。此后，又研制出单片8080和8085微处理器，也研制成多片集成电路构成的仿6800微处理器和单片的6800微处理器。80年代，又仿制成功8086微处理器。整机生产方面，国内有DJS-050和DJS-060两个系列，这些系列机都是通过引进成套散件进行组装的，DJS-050和Intel系列的微型机兼容，DJS-060系列和Motorola系列的微型机兼容。此外，以Z80为CPU的单板机TP-801产量也较大。而目前国内生产的仿IBM PC/XT的长城0520系列机和仿Apple机的紫金-Ⅱ及中华学习机年产量均超过万台。

§1-2 微型机的分类

人们可以从不同的角度对微型机进行分类。有人按机器组成来分，将微型机分为位片式、单片式、多片式；有人按制造工艺来分，将微型机分为MOS型和双极型。由于微型机性能的高低在很大程度上取决于核心部件微处理器，所以，最通常的作法是把微处理器的字长作为微型机的分类标准。

当前，可以见到以下几类微处理器构成的微型计算机：

1．4位微处理器

最初的4位微处理器就是Intel 4004，后来改进为4040。目前常见的是4位单片微型机，即在一个芯片内集中了4位的CPU、RAM、ROM、I/O接口和时钟发生器。这种单片机价格低廉，但运算能力弱、存储容量小，存储器中存放固定程序。这些特点使它们广泛用于各类袖珍计算器进行简单运算，或者用于家用电器和娱乐器件中进行简单的过程控制。

2．8位微处理器

8位微处理器被推出时，微型机技术已经比较成熟。因此，在8位微处理器基础上构成的微型机系统通用性较强，它们的寻址能力可以达到64K字节，有功能灵活的指令系统和较强的中断能力。另外，8位微处理器有比较齐备的配套电路。这些因素使8位微型机的应用范围很宽，广泛用于事务管理、工业控制、教育、通信行业。8位微处理器也常用来构成智能终端，而8位微型机则被许多家庭用作学习机和个人计算机。

常见的8位微处理器为Zilog的Z80、Intel的8080/8085、Motorola 6800/6802和Rockwell 6502。由这些微处理器构成的微型计算机在国内占有相当数量。

3．16位微处理器

16位微处理器不仅在集成度和处理速度、数据总线宽度等方面优于前几类微处理器，而且在功能和处理方法上也作了改进。在此基础上构成的微型计算机系统在性能方面已经和70年代的中档小型计算机相当。

16位微处理器中最有代表性的是 Intel 8086/8088和 Motorola 68000。以 Intel 8086/8088为 CPU 的16位微型机 IBM PC/XT 是目前的主流机型，它拥有的用户在计算机世界首屈一指，以至于在设计更高档的微型机时，都尽量保持对它兼容。

4。32位微处理器

这是当前性能最优的微处理器，典型产品为 Intel 80386、Motorola 68020，它们的主频率高达 20～40MHz，平均指令执行时间为 0.05μs。近两年，32位微型机已投入使用，但由于价格还比较高，因此，一般作为工作站进行计算机辅助设计、工程设计，或者作为微型机局域网中的资源站点。

除此以外，还有一类叫位片式处理器，利用这类微处理器，可以构成不同字长的微型计算机。位片机结构灵活，主要用于对中、小型计算机的仿真，或者用于高速实时控制专用系统中，也常用在分布式系统和高速智能外设。目前用得较多的位式片微处理器有 Intel 3000系列（2位）、AM 2900系列（4位）、MC 10800系列（4位）、F 100220系列（8位）。

§ 1-3 微处理器、微型计算机和微型计算机系统

在上一节，我们按照微处理器的档次对微型机作了分类，但是，必须注意，微处理器、微型计算机和微型计算机系统这三者的概念和含义是不同的。图1-1表明了它们之间的关系。

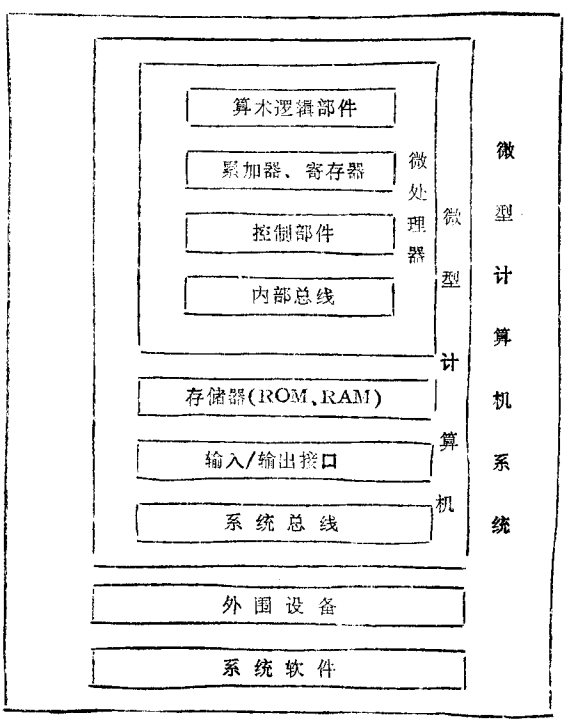

图 1-1 微处理器、微型计算机和微型计算机系统

一、微处理器

微处理器一般也称为CPU，它本身具有运算能力和控制功能。多数CPU是单片的，有时也会见到多片型的，即几个片子合起来完成一个CPU的功能。

微处理器是微型计算机的核心。尽管各种CPU的性能指标各不相同，但是有共同的特点。

首先，CPU一般都具备下列功能：

- 可以进行算术和逻辑运算；
- 可保存少量数据；
- 能对指令进行译码并执行规定的动作；
- 能和存储器、外设交换数据；
- 提供整个系统所需要的定时和控制；
- 可以响应其他部件发来的中断请求。

另外，CPU在内部结构上都包含下面这些部分：

- 算术逻辑部件（ALU）；
- 累加器和通用寄存器组；
- 程序计数器（指令指针）、指令寄存器和译码器；
- 时序和控制部件。

CPU内部的算术逻辑部件是专门用来处理各种数据信息的，它可以进行加、减、乘、除算术运算和与、或、非、异或等逻辑运算。比较低档的CPU不能进行乘、除运算，这种情况下，可以用程序来实现。

累加器和通用寄存器组用来保存参加运算的数据以及运算的中间结果，也用来存放地址。累加器也是寄存器，不过，它有特殊性，即许多指令的执行过程以累加器为中心。往往在运算指令执行前，累加器中存放一个操作数，指令执行后，由累加器保存运算结果，另外，输入/输出指令一般也通过累加器来完成。

程序计数器指向下一条要执行的指令。由于程序一般存放在内存的一个连续区域，所以，顺序执行程序时，每取1个指令字节，程序计数器便加1。指令寄存器存放从存储器中取出的指令码。指令译码器则对指令码进行译码和分析，从而确定指令的操作，并确定操作数的地址，再得到操作数，以完成指定的操作。

指令译码器对指令进行译码时，产生相应的控制信号送到时序和控制逻辑电路，从而，组合成外部电路所需要的时序和控制信号。这些信号送到微型计算机的其他部件，以控制这些部件协调工作。

实际上，微处理器的控制信号分为两类：一类是通过对指令的译码，由CPU内部产生的，这些信号由CPU送到存储器、输入/输出接口电路和其他部件；另一类是微型机系统的其他部件送到CPU的，通常用来向CPU发出请求，如中断请求、总线请求等。

二、微型计算机

微型计算机由CPU、存储器、输入/输出接口电路和系统总线构成。

CPU 如同微型计算机的心脏，它的性能决定了整个微型机的各项关键指标。

存储器包括随机存取存储器（RAM）和只读存储器（ROM）。

输入/输出接口电路是用来使外部设备和微型机相连的。

总线为 CPU 和其他部件之间提供数据、地址和控制信息的传输通道。

图1-2表示了微型计算机的基本结构。

特别要提到的是微型计算机的总线结构，它是一个独特的结构。有了总线结构以后，系统中各功能部件之间的相互关系变为各个部件面向总线的单一关系。一个部件只要符合总线标准，就可以连接到采用这种总线标准的系统中，使系统功能得到扩展。

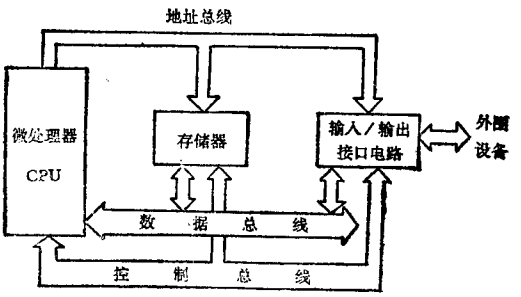

图 1-2 微型计算机的基本结构

尽管各种微型机的总线类型和标准有所不同，但大体上都包含 3 种不同功能的总线，即数据总线 DB(Data Bus)、地址总线 AB(Address Bus)和控制总线 CB(Control Bus)。

数据总线用来传输数据。从结构上看，数据总线是双向的，即数据既可以从 CPU 送到其他部件，也可以从其他部件传送到 CPU。数据总线的位数(也称为宽度）是微型机的一个很重要的指标，它和微处理器的位数相对应。和其他类型的计算机一样，在微型机中，数据的含义也是广义的。数据总线上传送的不一定是真正的数据，而可能是指令代码、状态量，有时还可能是一个控制量。

地址总线专门用来传送地址信息。因地址总是从 CPU 送出去的，所以和数据总线不同，地址总线是单向的。地址总线的位数决定了 CPU 可以直接寻址的内存范围。比如，8位微型机的地址总线一般是16位，因此，最大内存容量为 $2^{16}=64$K 字节；16位微型机的地址总线为20位，于是，最大内存容量为 $2^{20}=1$M 字节。

控制总线用来传输控制信号。其中包括 CPU 送往存储器和输入/输出接口电路的控制信号，如读信号、写信号、中断响应信号等；还包括其他部件送到 CPU 的信号，比如，时钟信号、中断请求信号、准备就绪信号等。

三、微型计算机系统

以微型计算机为主体，配上系统软件和外设之后，就成了微型计算机系统。系统软件包括操作系统和一系列系统实用程序，比如编辑程序、汇编程序、编译程序、调试程序等。有了系统软件，才能发挥微型机系统中的硬件功能，并为用户使用计算机提供了方便手段。外设用来使计算机实现数据的输入/输出，最通用的外设包括键盘、显示器、磁盘控制器、打印机等。

§ 1-4 微型计算机的应用

由于微型机具有体积小、价格低、耗电少和可靠性高等优点，所以应用范围十分广阔。微型机不仅在科学计算、信息处理、事务管理和控制等方面占重要地位，并且在日

常生活中也发挥了不可缺少的作用。归纳起来，目前有如下几个应用方面：

1．科学计算

现在，不少微型机系统具有较强的运算功能，特别是多个微处理器模块构成的系统，其功能往往可与大型机相匹敌，而成本却低到足以使大型机趋于淘汰。比如，美国Seguent公司用30个Intel 80386集合起来，构成Symmetry计算机，速度为120MIPS，达到IBM 3090系列最高档大型机的性能，价格却不到后者的十分之一。用更多的微处理器构成的并行处理机甚至可以超过大型机的速度和性能。比如，Intel公司用128个微处理器构成的IPSC机，速度达512 MIPS，这比任何一种商用大型机的速度都要高。Intel公司用一台32个微处理器构成的IPSC机运行一个广泛使用的科学计算程序，结果比Cray公司的大型机X-MP/2快40%，后者的速度为70 Megaflops（以百万次计算的每秒浮点操作速度），而前者的速度为100 Megaflops。

2．信息处理和事务管理

在短时间内完成对大量信息的处理是进入信息时代的必然要求。微型计算机配上数据库管理软件以后，可以很灵活地对各种信息按不同的要求进行分类、检索、转换、存储和打印，加上一些专用部件（如传感器）以后，还可以处理光、热、力、声音等物理信号。

3．过程控制

过程控制是微型机应用最多、也是最有效的方面之一。现在，制造工业和日用品生产厂家中都可见到微型机控制的自动化生产线，微型机在这些部门的应用为生产能力和产品质量的迅速提高开辟了广阔前景。

4．仪器、仪表控制

在许多仪器、仪表中，已经用微处理器代替传统的机械部件或分离的电子部件，这使产品降低了价格，而可靠性和功能却得到了提高。

此外，微处理器的应用还导致了一些原来没有的新仪器的诞生。在实验室里，出现了用微处理器控制的示波器——逻辑分析仪，它使电子工程技术人员能够用以前不可能采用的办法同时观察许多信号的波形和相互之间的时序关系。在医学领域，出现了用微处理器作为核心控制部件的CT扫描仪和超声扫描仪，加强了疾病的诊断手段。

5．家用电器和民用产品控制

由微处理器控制的洗衣机、冰箱在现在已经是很普通的民用电器了。此外，微处理器控制的自动报时、自动空调、自动报警系统也已经进入发达国家的家庭。还有，装有微处理器的娱乐产品往往将智能融于娱乐中；以微处理器为核心的盲人阅读器则能自动扫描文本，并读出文本的内容，从而为盲人带来福音。确切地讲，微处理器在人们日常生活中的应用所受到的主要限制不是技术问题，而是创造力和技巧上的问题。

当前，微型机技术正往两个方向发展：一个是高性能、高价格的方向，从这方面不断取得的成就可能使微型机代替价格昂贵、功能优越的巨型机；另一个是价格低廉、功能专一的方向，这方面的发展使微型机在生产领域、服务部门和日常生活中得到越来越广泛的应用。

第二章　8086微处理器

8086是Intel系列的16位微处理器，它是采用HMOS工艺技术制造的，内部包含约29000个晶体管。

8086有16根数据线和20根地址线。因为可用20位地址，所以可寻址的地址空间达2^{20}即1M字节。

8086工作时，只要一个5V电源和一相时钟，时钟频率为5MHz。后来，Intel公司推出的8086-1型微处理器时钟频率高达10MHz，8086-2型微处理器时钟频率达8MHz。

几乎在推出8086微处理器的同时，Intel公司还推出了一种准16位微处理器8088。推出8088的主要目的是为了与当时已有的一整套Intel外围设备接口芯片直接兼容使用。8088的内部寄存器、内部运算部件以及内部操作都是按16位设计的，但对外的数据总线只有8条。在本章的讲述过程中，我们对8088也将作出说明。

§ 2-1　8086的编程结构

要掌握一个CPU的工作性能和使用方法，首先应该了解它的编程结构。所谓编程结构，就是指从程序员和使用者的角度看到的结构，当然，这种结构与CPU内部的物理结构和实际布局是有区别的。在编程结构图（图2-1）中可以看到，从功能上，8086分为两部分，即总线接口部件BIU（Bus Interface Unit）和执行部件EU（Execution Unit）。

图2-1就是8086的编程结构图。

一、总线接口部件

总线接口部件的功能是负责与存储器、I/O端口传送数据。具体讲，总线接口部件要从内存取指令送到指令队列；CPU执行指令时，总线接口部件要配合执行部件从指定的内存单元或者外设端口中取数据，将数据传送给执行部件，或者把执行部件的操作结果传送到指定的内存单元或外设端口中。

总线接口部件由下列各部分组成：

4个段地址寄存器，即

CS——16位的代码段寄存器，

DS——16位的数据段寄存器，

ES——16位的扩展段寄存器，

SS——16位的堆栈段寄存器；

16位的指令指针寄存器IP；

20位的地址加法器；

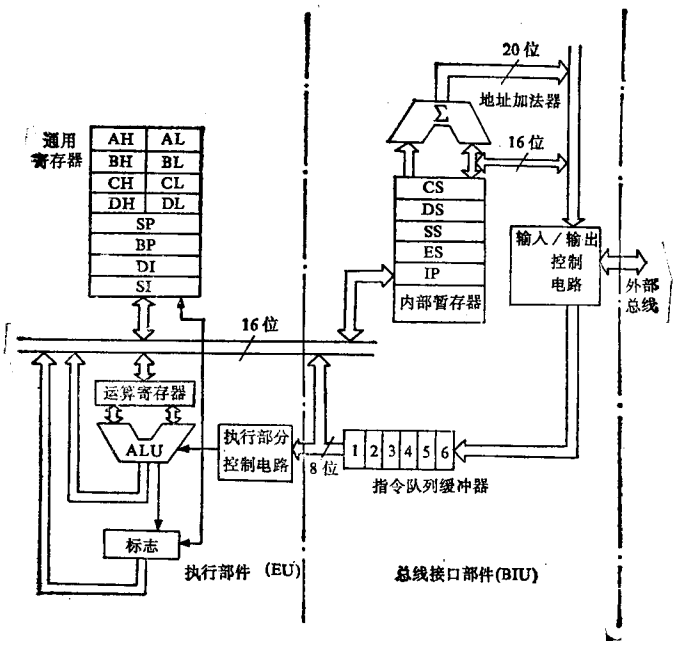

图 2-1 8086的编程结构

6 字 节的指令队列。

对总线接口部件，我们作下面两点说明：

① 8086 的指令队列为 6 个 字节，8088的指令队列为 4 个 字节。不管是 8086 还是 8088，都会在执行指令的同时，从内存中取下面 1 条指令或几条指令，取来的指令就放在指令队列中。这样，一般情况下，CPU 执行完一条指令就可以立即执行下一条指 令，而不需要像以往的计算机那样让 CPU 轮番地进行取指令和执行指令的 操 作，从而提高了 CPU 的效率。

② 地址加法器用来产生 20 位地址。上面已经提到，8086 可用20位地址寻址1M字节的内存空间，但 8086 内部所有的寄存器都是 16 位的，所以需要由一个附加的机构来根据 16 位寄存器提供的信息计算出 20 位的物理地址，这个机 构 就是 20 位的地址加法器。

比如，一条指令的物理地址就是根据代码段寄存器 CS和指令指针寄存器 IP 的内容得到的。具体计算时，要将段寄存器的内容左移 4 位，然后再与 IP 的内 容相加。假设 CS＝FE00H，IP＝0200H，此时指令的物理地址为 FE200H。

二、执行部件

执行部件的功能就是负责指令的执行。

从编程结构图可见到，执行部件由下列部分组成：

4 个 通用寄存器，即 AX、BX、CX、DX；

4 个专用寄存器，即基数指针寄存器 BP，堆栈指针寄存器 SP，源变址寄存器 SI，目的变址寄存器 DI；

标志寄存器；

算术逻辑单元。

对执行部件，有四点说明：

① 4个通用寄存器既可以作为16位寄存器使用，也可以作为8位寄存器使用。比如，BX寄存器作为8位寄存器时，分别称为BH和BL，BH为高8位，BL为低8位。

② AX寄存器也常称为累加器，8086指令系统中有许多指令都是通过累加器的动作来执行的。当累加器作为16位来使用时，可以进行按字乘操作、按字除操作、字输入/输出和其他字传送等；当累加器作为8位来使用时，可以实现按字节乘操作、按字节除操作、字节输入/输出、其他字节传送和十进制运算等。

③ 算术逻辑部件主要是加法器，绝大部分指令的执行都是由加法器完成的。

④ 标志寄存器共有16位，其中，7位未用，所用的各位含义如下：

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

熟悉8位微处理器Intel 8080和Intel 8085的读者一眼就会看出，在0～7位的标志CF、PF、AF、ZF、SF是和8080/8085的标志一样的。

根据功能，8086的标志可以分为两类：一类叫状态标志，另一类叫控制标志。状态标志表示前面的操作执行后，算术逻辑部件处在怎样一种状态，这种状态会像某种先决条件一样影响后面的操作。控制标志是人为设置的，指令系统中有专门的指令用于控制标志的设置和清除，每个控制标志都对某一种特定的功能起控制作用。

状态标志有6个，即SF、ZF、PF、CF、AF和OF。

符号标志SF (Sign Flag)：它和运算结果的最高位相同。我们知道，当数据用补码表示时，负数的最高位为1，所以符号标志指出了前面的运算执行后的结果是正还是负。

零标志ZF (Zero Flag)：如果当前的运算结果为零，则零标志为1；如果当前的运算结果为非零，则零标志为0。

奇偶标志PF (Parity Flag)：如果运算结果的低8位中所含的1的个数为偶数，则PF为1，否则为0。

进位标志CF (Carry Flag)：当执行一个加法运算使最高位产生进位时，或者执行一个减法运算引起最高位产生借位时，则CF为1。除此之外，循环指令也会影响这一标志，这在后面还会讨论。

辅助进位标志AF (Auxiliary Carry Flag)：当加法运算时，如果第三位往第四位有进位，或者当减法运算时，如果第三位从第四位有借位，则AF为1。辅助进位标志一般在BCD码运算中作为是否进行十进制调整的判断依据。

溢出标志OF (Overflow Flag)：当运算过程中产生溢出时，会使OF为1。所谓溢出，就是当字节运算的结果超出了范围－128～＋127，或者当字运算的结果超出了范围－32768～＋32767时称为溢出。计算机在进行加法运算时，每当判断出低位往最高有效位产生进位，而最高有效位往前没有进位时，便得知产生了溢出，于是OF为1；或

者反过来，每当判断出低位往最高位无进位，而最高位往前却有进位时，便得知产生了溢出，于是OF为1。在减法运算时，每当判断出最高位需要借位，而低位并不向最高位产生借位时，OF置1；或者反过来，每当判断出低位从最高位有借位，而最高位并不需要从更高位借位时，OF置1。

为对上述状态标志有更具体的了解，举两个例子。

比如，执行下面两个数的加法：

```
  0010  0011  0100  0101
+ 0011  0010  0001  1001
------------------------
  0101  0101  0101  1110
```

由于运算结果的最高位为0，所以，SF=0；而运算结果本身不为0，所以，ZF=0；低8位所含的1的个数为5个，即有奇数个1，所以，PF=0；由于最高位没有产生进位，所以，CF=0；又由于第三位没有往第四位产生进位，所以，AF=0；由于低位没有往最高位产生进位，最高位往前也没有进位，所以，OF=0。

又比如，执行下面两个数的加法：

```
  0101  0100  0011  1001
+ 0100  0101  0110  1010
------------------------
  1001  1001  1010  0011
```

显然，由于运算结果的最高位为1，所以，SF=1；由于运算结果本身不为0，所以ZF=0；由于低8位所含的1的个数为4个，即含有偶数个1，所以，PF=1；由于最高位没有往前产生进位，所以，CF=0；运算过程中，第三位往第四位产生了进位，所以，AF=1；由于低位往最高位产生了进位，而最高位没有往前产生进位，所以OF=1。

当然，在绝大多数情况下，一次运算后，并不对所有标志进行改变，程序也并不需要对所有的标志作全面的关注。一般只是在某些操作之后，对其中某个标志进行检测。

控制标志有3个，即DF、IF、TF。

方向标志DF (Direction Flag)：这是控制串操作指令用的标志。如果DF为0，则串操作过程中地址会不断增值；反之，如果DF为1，则串操作过程中地址会不断减值。

中断标志IF (Interrupt Enable Flag)：这是控制可屏蔽中断的标志。如果IF为0，则CPU不能对可屏蔽中断请求作出响应；如果IF为1，则CPU可以接受可屏蔽中断请求。

跟踪标志TF (Trap Flag)：如果TF为1，则CPU按跟踪方式执行指令。

这些控制标志一旦设置之后，便对后面的操作产生控制作用。

三、总线接口部件和执行部件的动作管理

总线接口部件和执行部件并不是同步工作的，但是尽管不同步，两者的动作仍然是有管理原则的，体现在下面几个方面：

① 每当8086的指令队列中有2个空字节，或者8088的指令队列中有1个空字节时，总线接口部件就会自动把指令取到指令队列中。

② 每当执行部件准备执行一条指令时，它会从总线接口部件的指令队列前部取出指令的代码，然后用几个时钟周期去执行指令。在执行指令的过程中，如果必须访问存储器或者输入/输出设备，那么，执行部件就会请求总线接口部件进入总线周期去完成访问内存或者输入/输出端口的操作；如果此时总线接口部件正好处于空闲状态，那么，会立即响应执行部件的总线请求。但有时会遇到这样的情况，执行部件请求总线接口部件访问总线时，总线接口部件正在将某个指令字节取到指令队列中，此时，总线接口部件将首先完成这个取指令的总线周期，然后再去响应执行部件发出的访问总线的请求。

③ 当指令队列已满，而且执行部件对总线接口部件又没有总线访问请求时，总线接口部件便进入空闲状态。

④ 在执行转移指令、调用指令和返回指令时，下面要执行的指令就不是在程序中紧接着排列的那条指令了，而总线接口部件往指令队列装入指令时，总是按顺序进行的，这样，指令队列中已经装入的字节就没有用了。遇到这种情况，指令队列中的原有内容被自动清除，总线接口部件会接着往指令队列中装入另一个程序段中的指令。

为了更深入了解 8086 和 8088 的工作特点，我们来比较一下 8086/8088 系统的工作与传统的计算机有什么不同。

传统的计算机都是按照下面这样的步骤来工作的：

① 从指令指针所指的内存单元中取一条指令送到指令寄存器。

② 对指令进行译码，而指令指针进行增值，以指向下一条指令。

③ 执行指令。如果所执行的是转移指令、调用指令或者返回指令，则重新设置指令指针的值，以指向下一条要执行的指令。

可见，传统的计算机在执行指令时，总是相继地进行提取指令和执行指令的动作，也就是说，指令的提取和执行是串行进行的。

相比之下，在 8086/8088 中，指令的提取和执行是分别由总线接口部件和执行部件完成的，总线控制逻辑和指令执行逻辑之间是既互相独立又互相配合的。正是这种互相配合但又非同步的工作方式，使得 8086/8088 可以在执行指令的同时进行提取指令的操作。在 8086 或 8088 CPU 中，执行部件可以不停地一条接一条地执行事先已经进入指令队列中的指令。只有当遇到转移指令、调用指令和返回指令时，或者当某一条指令的执行过程中，需要访问内存的次数过于频繁，以至于总线接口部件没有空闲从内存将指令提取到指令队列中时，才需要执行部件等待总线接口部件提取指令。而这些情况相对来说是较少发生的。

8086/8088CPU 中，总线接口部件和执行部件的这种并行工作方式，有力地提高了工作效率，这也正是 8086/8088 成功的原因之一。

四、8086的总线周期的概念

为了取得指令和传送数据，就需要 CPU 的总线接口部件执行一个总线周期。为了便于叙述后面的内容，先对总线周期的概念作一个介绍，在讲完 8086 的基本配置以后，再详细地讲解总线操作。

在 8086/8088 中，一个最基本的总线周期由 4 个时钟周期组成，时钟周期是 CPU 的基本时间计量单位，它由计算机主频决定。比如，8086 的主频为 5MHz，1 个时钟周

期就是200ns；8086-1的主频为10MHz，1个时钟周期为100ns。在1个最基本的总线周期中，习惯上将4个时钟周期分别称为4个状态，即T_1状态、T_2状态、T_3状态和T_4状态。

① 在T_1状态，CPU往多路复用总线上发出地址信息，以指出要寻址的存储单元或外设端口的地址。

② 在T_2状态，CPU从总线上撤消地址，而使总线的低16位浮置成高阻状态，为传输数据作准备。总线的最高4位（A_{19}～A_{16}）用来输出本总线周期状态信息。这些状态信息用来表示中断允许状态、当前正在使用的段寄存器名等。

③ 在T_3状态，多路总线的高4位继续提供状态信息，而多路总线的低16位（8088则为低8位）上出现由CPU写出的数据或者CPU从存储器或端口读入的数据。

④ 在有些情况下，被写入数据或者被读取数据的外设或存储器不能及时地配合CPU传送数据。这时，外设或存储器会通过"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，于是CPU会在T_3之后插入1个或多个附加的时钟周期T_w。T_w也叫等待状态，在T_w状态，总线上的信息情况和T_3状态的信息情况一样。当指定的存储器或外设完成数据传送时，便在"READY"线上发出"准备好"信号，CPU接收到这一信号后，会自动脱离T_w状态而进入T_4状态。

⑤ 在T_4状态，总线周期结束。

需要指出的是，只有在CPU和内存或I/O接口之间传输数据，以及填充指令队列时，CPU才执行总线周期。可见，如果在1个总线周期之后，不立即执行下1个总线周期，那么，系统总线就处在空闲状态，此时，执行空闲周期。

在空闲周期中，可以包含1个时钟周期或多个时钟周期。这期间，在高4位上，CPU仍然驱动前一个总线周期的状态信息，而且，如果前一个总线周期为写周期，那么，CPU会在总线低16位上继续驱动数据信息；如果前一个总线周期为读周期，则在空闲周期中，总线低16位处于高阻状态。

图2-2表示了一个典型的总线周期序列。

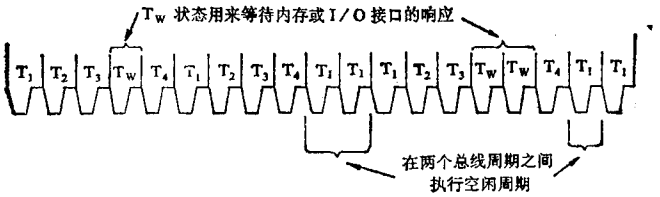

图 2-2 典型的8086总线周期序列

§ 2-2 8086的引腿信号和工作模式

一、最小模式和最大模式的概念

为了尽可能适应各种各样的使用场合，在设计8086/8088CPU芯片时，就使得它们可以在两种模式下工作，即最大模式和最小模式。

所谓最小模式，就是在系统中只有8086或者8088一个微处理器。在这种系统中，

所有的总线控制信号都直接由 8086 或 8088 产生，因此，系统中的总线控制逻辑电路被减到最少。这些特征就是最小模式名称的由来。

最大模式是相对最小模式而言的。最大模式用在中等规模的或者大型的 8086/8088 系统中。在最大模式系统中，总是包含有两个或多个微处理器，其中一个主处理器就是 8086 或者 8088，其他的处理器称为协处理器，它们是协助主处理器工作的。

和 8086/8088 配合的协处理器有两个，一个是数值运算协处理器 8087，一个是输入/输出协处理器 8089。

8087 是一种专用于数值运算的处理器，它能实现多种类型的数值操作，比如高精度的整数和浮点运算，也可以进行超越函数（如三角函数、对数函数）的计算。由于在通常情况下，这些运算往往通过软件方法来实现，而 8087 是用硬件方法来完成这些运算的，所以，在系统中加入协处理器 8087 之后，会大幅度地提高系统的数值运算速度。

8089 在原理上有点象带有两个 DMA 通道的处理器，它有一套专门用于输入/输出操作的指令系统，但是，8089 又和 DMA 控制器不同，它可以直接为输入/输出设备服务，使 8086 或 8088 不再承担这类工作。所以，在系统中增加协处理器 8089 后，会明显提高主处理器的效率，尤其是在输入/输出频繁的场合。

关于 8086/8088 到底工作在最大模式还是最小模式，这完全由硬件决定。

二、8086/8088 的引腿信号和功能

图 2-3 是 8086 和 8088 的引腿信号图。

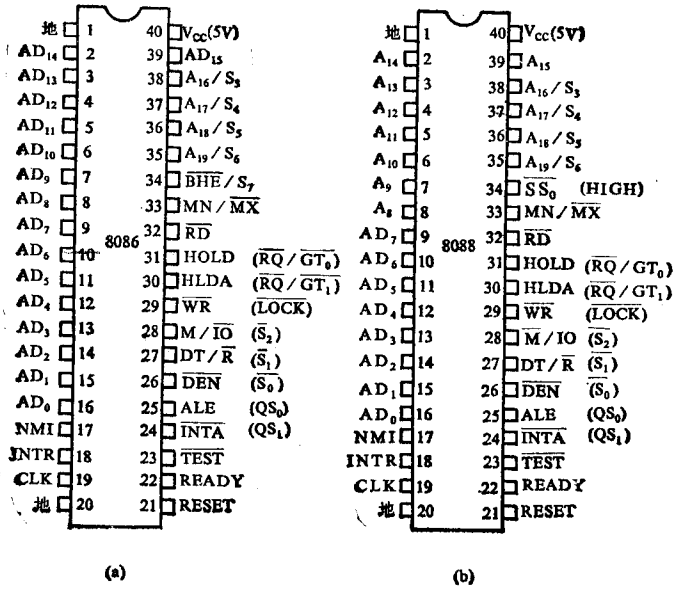

图 2-3 8086/8088 的引腿信号（括号中为最大模式时引腿名）
（a）8086 的引腿信号；（b）8088 的引腿信号。

对于 8086/8088 的引腿信号，首先要注意下列几点：

① 8086/8088 的数据线和地址线是复用的，所以常把 8086/8088的总线称为多路总线，即某一时候总线上出现的是地址，另一时候，总线上出现的是数据。正是这种引腿的分时使用方法才能使 8086/8088 用 40 条引腿实现 20 位地址、16 位数据及众多的控制信号和状态信号的传输。不过 8086 和 8088 是有差别的，由于 8088 只能传输 8 位数据，所以 8088 只有 8 个地址引腿兼为数据引腿；而 8086 是按 16 位传输数据的，所以有 16 个地址/数据复用引腿。

② 除了第 28 腿和第 34 腿以外，8086 和8088的控制线引腿定义是一样的。在最小模式时，8088 的第 28 腿为$\overline{M}$/IO，而 8086 的第 28 腿为 M/$\overline{IO}$。这是为了使 8088 和 Intel 系列的 8 位微处理器 8080/8085 兼容。8086 的第 34 腿为 $\overline{BHE}/S_7$，而 8088 的第 34 腿为 SS_0。这是因为8086有 16 根数据线，可用高 8 位数据线传送 1 个字节，也可用低 8 位数据线传送 1 个字节，还可一次传送 1 个字，$\overline{BHE}$ 信号就是用来区分这几类传输的。8088 只能进行 8 位传输，所以第 34 腿只用来指出状态信息，而不作复用。

③ 第 21 腿（RESET）是输入复位信号用的。大部分计算机系统中都有一根对系统进行启动的复位线，复位线和系统中所有的部件相连。在系统开机时，有一个脉冲发送到复位线上，表示现在系统进行启动，此时，CPU 和各部件都会接收到这个复位脉冲；此外，在操作员按下 RESET 键时，也会有一个复位脉冲发送到复位线上，使系统重新启动。复位脉冲的有效电平为高电平。

当然我们很关心 CPU 的启动状态到底是什么样的。

在 8086/8088 系统中，CPU 被启动后，处理器的标志寄存器、指令指针寄存器 IP、段寄存器 DS、SS、ES 和指令队列都被清除，但是代码段寄存器 CS 被设置为 FFFFH。因为IP＝0000，而CS＝FFFFH，所以，此后，8086/8088 将从地址 FFFF0H 开始执行指令。通常，在安排内存区域时，将高地址区作为只读存储区，而且，在 FFFF0H单元开始的几个单元中放一条无条件转移指令，转到一个特定的程序中。这个程序往往实现对系统作初始化、引导监控程序或者引导操作系统等功能，这样的程序叫做引导和装配程序。

④ 第 22 引腿用于从内存或 I/O 接口往CPU输入“准备好”（READY）信号。“准备好”信号用来告诉 CPU，在下一个时钟周期中，内存或外设将在总线上放一个输入数据；或者将在下一个时钟周期中，内存或外设已从数据总线上接收一个数据。不管是输入还是输出，CPU 和总线控制逻辑电路都会在下一个时钟周期后，完成当前的总线周期。有关时钟周期和总线周期的含义，我们已在前面说明。

⑤ 第 23 腿（$\overline{TEST}$）是在多处理器系统中使用的，后面我们再作具体讲述。第32 腿（$\overline{RD}$）指出当前要执行一个输入操作。在最小模式中，第 32 腿还和第 28 腿（M/$\overline{IO}$，8088 中为$\overline{M}$/IO）一起使用，以区分当前进行的是CPU和内存之间的数据传输还是 CPU 和 I/O 设备之间的数据传输。第 29 腿在最小模式中用来指出将要执行一个输出操作，并且和第 28 腿（M/$\overline{IO}$或$\overline{M}$/IO）一起指出是往内存单元写数据还是往 I/O 设备写数据。

⑥ 高 4 位地址和状态线复用。在总线周期的前一部分时间，A_{19}/S_6～A_{16}/S_3腿用来输出高 4 位地址，在总线周期的其余部分时间，则用来输出状态信息。

为了对 8086/8088 各引腿信号有个具体的了解，下面，我们逐一介绍它们。

1．GND、V_{CC}，地和电源

第 1、20 腿为地；第 40 腿为电源，8086 和 8088 均用单一的 +5V 电压。

2．AD_{15}～AD_0(Address Data Bus) 地址/数据复用引腿，双向工作

第 2～16 腿分别为 AD_{14}～AD_0，第 39 腿为 AD_{15}。需要说明的一点是，在 8088 中，AD_{15}～AD_8实际上不作复用，它们只用来输出地址，称为 A_{15}～A_8。

作为复用引腿，在总线周期的 T_1 状态用来输出要访问的存储器或 I/O 端口的地址，T_2～T_3 状态，对读周期来说，是处于浮空状态，而对写周期来说，则是传输数据。

需要特别指出的是，在 8086 系统中，常将 AD_0 信号作为低 8 位数据的选通信号，因为每当 CPU 和偶地址单元或偶地址端口交换数据时，在 T_1 状态，AD_0 引腿传送的地址信号必定为低电平，在其他状态，则用来传送数据。而 CPU 的传输特性决定了只要是和偶地址单元或偶地址端口交换数据，那么，CPU 必定通过总线低 8 位即AD_7～AD_0传输数据。可见，如果在总线周期的 T_1 状态，AD_0 为低电平，实际上就指示了在这一总线周期的其余状态中，CPU 将用总线低 8 位和偶地址单元或偶地址端口交换数据。因此，AD_0 和下面讲到的 $\overline{BHE}$ 类似，可以用来作为接于数据总线低 8 位上的 8 位外设接口芯片的选通信号。这一点，在后面讲述接口芯片的章节中会作进一步的说明。

AD_{15}～AD_0 在 CPU 响应中断时，以及系统总线“保持响应”时，都被浮置为高阻状态。

3．A_{19}/S_6～A_{16}/S_3 (Address/status)地址/状态复用引腿，输出

第 35～38 腿分别为 A_{19}/S_6～A_{16}/S_3，这些引腿在总线周期的 T_1 状态，用来输出地址的最高 4 位，在总线周期的 T_2、T_3、T_w 和 T_4 状态时，用来输出状态信息。

其中 S_6 为 0 用来指示 8086/8088 当前与总线相连，所以，在 T_2、T_3、T_w 和 T_4 状态，8086/8088 总是使 S_6 等于 0，以表示 8086/8088 当前连在总线上。

S_5 表明中断允许标志的当前设置，如为1，表示当前允许可屏蔽中断请求，如为 0，则禁止一切可屏蔽中断。

S_4 和 S_3 合起来指出当前正在使用哪个段寄存器，具体规定如表 2-1 所示。

表 2-1 S_4、S_3 的代码组合和对应的含义

S_4	S_3	含义
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，或者未用任何段寄存器
1	1	当前正在使用DS

当系统总线处于“保持响应”状态时，A_{19}/S_6～A_{16}/S_3 被浮置为高阻状态。

4．$\overline{BHE}/S_7$ (Bus High Enable/Status) 高 8 位数据总线允许/状态复用引腿，输出

在总线周期的 T_1 状态，8086 在 $\overline{BHE}/S_7$ 引腿输出 $\overline{BHE}$ 信号，表示高 8 位数据线

D_{15}～D_8上的数据有效；在T_2、T_3、T_4及T_w状态，$\overline{BHE}/S_7$引腿输出状态信号S_7，不过，在当前的芯片（8086、8086-1、8086-2)设计中，S_7并未被赋于任何实际意义。

$\overline{BHE}$信号和A_0合起来告诉连接在总线上的存储器和接口，当前的数据在总线上将以何种格式出现。归纳起来，一共有如表 2-2 所示的 4 种可能格式。

表 2-2 $\overline{BHE}$和A_0的代码组合和对应的操作

$\overline{BHE}$	A_0	操　　作	所用的数据引腿
0	0	从偶地址开始读/写一个字	AD_{15}～AD_0
1	0	从偶地址单元或端口读/写一个字节	AD_7～AD_0
0	1	从奇地址单元或端口读/写一个字节	AD_{15}～AD_8
0	1	从奇地址开始读/写一个字	AD_{15}～AD_8
1	0	（在第一个总线周期，将低 8 位数字送到AD_{15}～AD_8，在第二个总线周期，将高 8 位数字送到AD_7～AD_0）	AD_7～AD_0

在 8086 系统中，如果要读/写从奇地址单元开始的一个字，需要用 2 个总线周期，这是特别要指出的一点。

在 8088 中，第 34 腿不是$\overline{BHE}_7/S_7$，而是被赋于另外的信号。在最大模式时，此引腿恒为高电平；在最小模式中，则为$\overline{SS}_0$，它和 DT/$\overline{R}$、$\overline{M}$/IO一起决定了 8088 芯片当前总线周期的读/写动作。

5．NMI(Non-Maskable Interrupt) 非屏蔽中断输入引腿

非屏蔽中断信号是一个由低到高的上升沿。这类中断不受中断允许标志 IF 的影响，也不能用软件进行屏蔽。每当 NMI 端进入一个正沿触发信号时，CPU 就会在结束当前指令后，进入对应于中断类型号为 2 的非屏蔽中断处理程序。

6．INTR (Interrupt Request) 可屏蔽中断请求信号输入

第 18 腿为可屏蔽中断请求信号输入端。可屏蔽中断请求信号为高电平有效。CPU 在执行每条指令的最后一个时钟周期会对 INTR 信号进行采样，如果 CPU 中的中断允许标志为 1，并且又接收到 INTR 信号，那么，CPU 就会在结束当前指令后，响应中断请求，进入一个中断处理子程序。

7．$\overline{RD}$ (Read) 读信号输出

第 32 腿为读信号输出端，此信号指出将要执行一个对内存或 I/O 端口的读操作。具体到底是读取内存单元中的数据还是 I/O 端口中的数据，这决定于 M/$\overline{IO}$ 信号。在一个执行读操作的总线周期中，$\overline{RD}$ 信号在T_2、T_3和T_w状态均为低电平。在系统总线进入“保持响应”期间，$\overline{RD}$ 引腿被浮置为高阻状态。

8．CLK (Clock) 时钟输入

第 19 腿为时钟输入端，8086 和 8088 要求时钟信号的占空比为 33%，即 1/3 周期为高电平，2/3 周期为低电平。8086 和 8088 的时钟频率要求为 5MHz，8086-1 的时钟频率为 10MHz，8086-2的时钟频率则为 8MHz。时钟信号为 CPU 和总线控制逻辑电路提供定时手段。

9．RESET(Reset) 复位信号输入

第21腿为复位信号输入端，高电平有效。8086/8088 要求复位信号至少维持 4 个时钟周期的高电平才有效。复位信号来到后，CPU 便结束当前操作，并对处理器标志寄存器、IP、DS、SS、ES 及指令队列清零，而将 CS 设置为FFFFH。当复位信号变为低电平时，CPU 从 FFFF0H 开始执行程序。

10．READY (Ready) "准备好"信号输入

第 22 腿为"准备好"信号输入端。"准备好"信号实际上是由所访问的存储器或者 I/O 设备发来的响应信号，高电平有效。"准备好"信号有效时，表示内存或 I/O 设备准备就绪，马上就可进行一次数据传输。CPU 在每个总线周期的 T_3 状态开始对 READY 信号进行采样。如果检测到 READY 为低电平，则在 T_3 状态之后插入等待状态 T_w。在 T_w 状态，CPU 也对 READY 进行采样，如 READY 仍为低电平，则会继续插入 T_w，所以，T_w 可以插入 1 个或多个。直到 READY 变为高电平后，才进入 T_4 状态，完成数据传送过程，从而结束当前总线周期。

11．$\overline{\text{TEST}}$ (Test) 测试信号输入

第 23 腿为测试信号输入端，低电平有效。$\overline{\text{TEST}}$ 信号是和指令WAIT 结合起来使用的，在 CPU 执行 WAIT 指令时，CPU 处于空转状态进行等待；当 8086 的 $\overline{\text{TEST}}$ 信号有效时，等待状态结束，CPU 继续往下执行被暂停的指令。后面，我们要讲到，WAIT 指令是用来使处理器与外部硬件同步用的。

12．MN/$\overline{\text{MX}}$ (Minimum/Maximum Mode Control) 最小/最大模式控制信号输入

第 33 腿为最小模式和最大模式的选择控制端，这一引腿的输入决定了 8086/8088 到底工作在最小模式，还是工作在最大模式。如果此引腿固定接为 + 5V，则 CPU 处于最小模式；如果接地，则 CPU 处于最大模式。

上述信号是 8086/8088 工作在最小模式和最大模式时都要用到的。此外，8086/8088 第 24～31 腿还有8个控制信号，它们在最小模式和最大模式下有不同的名称和定义。

三、最小模式

当 8086/8088 的第 33 腿 MN/$\overline{\text{MX}}$ 固定接到 + 5V 时，就处于最小工作模式了。最小模式下第 24～31 腿的信号含义如下。

1．$\overline{\text{INTA}}$ (Interrupt Acknowledge) 中断响应信号输出

在最小模式下，第 24 腿作为中断响应信号的输出端，用来对外设的中断请求作出响应。对于 8086/8088 来讲，$\overline{\text{INTA}}$ 信号实际上是位于连续周期中的两个负脉冲，在每个总线周期的 T_2、T_3 和 T_w 状态，$\overline{\text{INTA}}$ 端为低电平。第一个负脉冲通知外部设备的接口，它发出的中断请求已经得到允许；外设接口收到第二个负脉冲后，往数据总线上放中断类型码，从而 CPU 便得到了有关此中断请求的详尽信息。

2．ALE (Address Latch Enable) 地址锁存允许信号输出

第 25 腿在最小模式下为地址锁存允许信号输出端，这是 8086/8038 提供给地址锁存器 8282/8283 的控制信号，高电平有效。在任何一个总线周期的 T_1 状态，ALE 输出有效电平，以表示当前在地址/数据复用总线上输出的是地址信息，地址锁存器将 ALE