

83774

Intel 8086 微处理器

唐长钧 赵正校 朱煜清

上海交通大学 微机研究所
科技交流室

1986. 5

Intel 8086 微处理器

(上册)

唐长钧 赵正校 朱煜清

1986.5

目 录

8086微处理器电路系列

第一章	综述	(1)
第二章	8086十六位微处理器	(5)
第三章	8087数值处理器	(100)
第四章	8089 I/O处理器	(148)
第五章	8086外围芯片	(185)
第六章	系统结构	(205)

MIC-867十六位微型计算机

第七章	MIC-867十六位单板机的研制	(207)
第八章	MIC-867监控程序	(224)
第九章	MIC-867编辑程序	(235)
第十章	MIC-867汇编程序	(239)
第十一章	MIC-867系列扩充板(I)	(266)
第十二章	MIC-867系列扩充板(II)	(273)
第十三章	MIC-867使用说明	(284)
附录一	8086指令系统	(288)
附录二	8086代码表	(296)
附录三	8087指令系统	(306)
附录四	8087代码表	(310)
附录五	8089指令系统	(314)
附录六	8089代码表	(320)
附录七	多总线	(326)

第一章 综 述

当前常用的16位微处理器有 i8086、MC68000和Z8000。从单片的功能来看 MC68000最强，i8086最弱。但英特尔公司在设计CPU的同时，又设计了二个专用的协处理器：数值处理器8087和I/O处理器8089。这三片芯片组合在一起，其功能就超过了其他的16位微处理器。这种设计思想带来的一个好处是“灵活”。即用户可根据需要决定是否要加8087或8089，也就是说使用英特尔器件设计的16位处理器系统，其功能有较大的伸缩性。

在使用英特尔16位微处理器构成系统时，必需要有支持芯片。这些芯片型号见表1.1。8086系统的体系结构有如下特点：

- a. 为减轻 CPU 负担，並增强整个系统的功能，电路系列中的器件具有专门的功能。如表1.1所示，高速运算由专用的数值处理器8087执行；输入输出处理由专用的输入输出处理器8089执行；系统总线控制由专门的总线裁决器8289实现。
- b. 容易构成多处理器系统。
- c. 总线可分成系统总线和局部总线，因此可灵活地组成各种性能的系统。

表 1.1

芯 片	工 艺	引线端	功 能
8086	MOS	40	16位CPU(外部数据总线16位)
8087			数值处理器
8088			16位CPU(外部数据总线8位)
8089			I/O处理器
8259A		28	可编程中断控制器
8282	双极型	20	8 位锁存器(同相输出)
8283			8 位锁存器(反相输出)
8284		18	时钟发生器
8286		20	8 位总线驱动器(同相输出)
8287			8 位总线驱动器(反相输出)
8288			总线控制器
8289			总线裁决器

1. 处理器

(1) 8086/8088CPU

中央处理器8086和8088都是16位数据处理用的第三代微处理器，体系结构基本相同。两者的主要不同是8088的外部数据总线是8位，而8086则是16位。因此，在8088的系统中数据是以8位为单位传送的，在8086系统中则是以16位为单位传送的。显然，8086的信息流量比8088要大。

这二种处理器都有最小方式和最大方式二种动作方式。在最小方式中，由CPU自身发出存储器和输入输出的控制信号。在最大方式中，CPU只给8288状态信号，而控制信号由8288发出。这样在组成较大系统时，可确保必要的信号，从而弥补了CPU引线的不足。

(2) 8089I/O处理器

I/O处理器 8089 主要是为处理输入输出动作而设计的微处理器。其特点是可高效率地进行数据传送。8089通常根据CPU来的命令进行动作。执行动作完全可独立于CPU，因此可大大提高系统性能。

8089内部有二个独立的输入输出通道。每个通道具有CPU的功能以及控制通道的DMA功能，即各通道像CPU那样用执行程序来实现输入输出动作，并且最高可以以1.25MHZ的速度进行DMA传送。在DMA传送时，处理的数据可以是8位，也可以是16位。

8089承担了原来由CPU处理的输入输出动作，因而减轻了CPU的负担，提高了系统的流量。

(3) 8087数值处理器

以前，高精度的数值运算或浮点数的运算通常由程序来实现。其缺点是执行时间长。数值处理器8087是将复杂的数据运算用硬件来实现的数值运算处理器。它与8086用软件来实现数值运算相比，运算速度可提高100倍。

CPU和8087在工作时都监视着出现在总线上的指令（从内存取出），若是数值运算指令（即ESC指令），便由8087将该指令取入加以执行。在8087执行指令时，CPU可同时执行其他指令。

8087可处理的数据有16位、32位、64位的整数，18位组合二——十进制数以及32位、64位实数。但不管是哪种类型的数据，在8087内部运算时都化成80位实数进行运算。8087可直接处理的运算有加减乘除四则运算、平方根、比较、基本超越函数（如正切、反正切）等。

2. 8259A中断控制器

8259A是将用于8080/8085的中断控制器8259加以改进，使之适用于8086的中断控制器。8259A单独可进行8个中断源的控制。如将8259A级联起来，那末最多可以由8片8259A来控制64个中断源。每个中断源都有一定的优先级别。8259A经常监视着这些中断源，将提出中断请求、优先级最高的中断源所发出的中断请求信号传送给CPU。CPU在接受中断请求后，通知8259A给出该中断源的一个字节的向量值。

3. 总线接口

(1) 8282/8283八位锁存器

用以锁存地址数据总线上的地址。每个器件有 8 个锁存器，具有较强的驱动能力。

(2) 8286/8287 八位总线收发器

CPU 的总线仅可驱动 2mA 的电流。8286/8287 可驱动 32mA 的电流。

(3) 8288 总线控制器

它用于 8086 (或 8088) 的最大方式时的总线控制。它将 8086 (或 8088)、8089 的状态信号译码后，发出总线的控制命令。在处理器使用总线时，它就发出如存储器读、存储器写、I/O 读、I/O 写等命令。同时 8288 还发出锁存地址的选通信号。

(4) 8289 总线裁决器

几个处理器使用共有存储器或 I/O 设备时，它们都与系统总线相连，8289 就管理来自这些处理器的要求使用系统总线的请求，以避免总线竞争。当几个处理器同时请求使用系统总线时，8289 根据优先顺序，把系统总线使用权分配给优先级最高的处理器。

4. 总线结构

图 1.1 是 8086 系列的基本总线结构。有系统总线和局部总线二类。处理器通常连于局部总线。局部总线与系统总线通过总线接口相连接。系统总线可与几个局部总线相连。各处理器共有的存储器和 I/O 设备也连于系统总线。

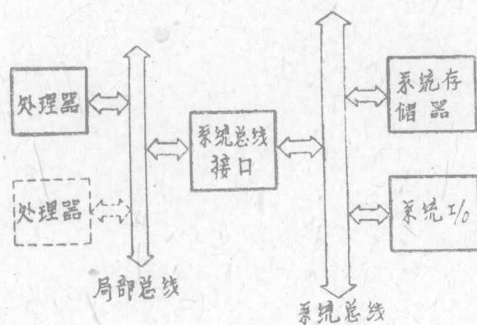


图 1.1 8086 系列的基本总线结构

(1) 局部总线

由于处理器本身包含总线管理功能，所以处理器与局部总线相连时，不需要其他总线控制电路。

(2) 系统总线

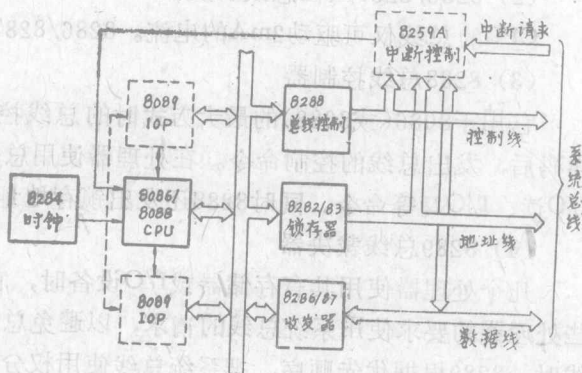
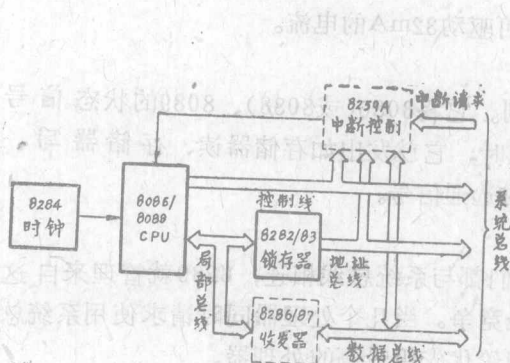
系统总线包括地址总线、数据总线、控制总线、中断线和总线管理线。通过这些引线，可方便地与存储器和 I/O 接口器件相连。控制线包括存储器读写和 I/O 读写信号。

系统总线与局部总线通过接口相连。该接口所使用的双极型集成电路数的多少视系统的大小不同而不同。8086 系列所采用的系统总线是“多总线”(multiBus)。

图 1.2 是 CPU 为最小方式时的系统结构图。在最小方式中，对存储器和 I/O 的控制信号由 CPU 自身发生。另外，需要 2~3 个 8 位锁存器 8282 或 8283。8286 或 8287 在存储器或 I/O 器件较多时使用。因此，在 CPU 为最小方式时，可用很少的器件构成一个系统。

图 1.3 是 CPU 为最大方式时的系统结构图。这时需要总线控制器 8288，由它发出各种控

制信号。在CPU为最大方式时,还可与协处理器(8087、8089)一起构成多处理器结构。



在系统总线上,若要增加其他的处理器构成多处理器模块,则在图1.3中再要增加 8289 总线裁决器。在系统总线上最多可以连接16个处理器模块。

第二章 8086十六位微处理器

2.1 内部结构

8086微处理器按内部功能可分为指令执行单元EU和总线接口单元BIU两部分，如图2.1所示。EU主要执行指令，BIU主要进行取指令、取操作数、写入结果等。这两个部分相互独

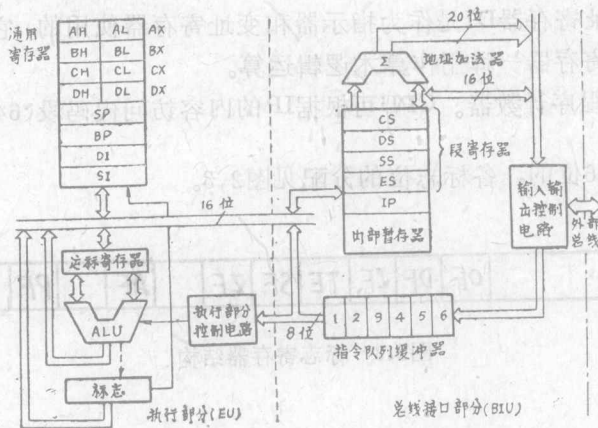


图2.1 8086内部结构框图

立工作，因此可在取指令的同时，进行运算等操作。EU所执行的指令都预先由BIU从存储器取出，这样缩短了指令的执行时间，提高了处理器的性能。

1. 指令执行单元EU

指令执行单元EU由内部的16位算术逻辑运算部件进行算术运算，根据其结果设置CPU的状态标志位。EU从BIU取得要执行的指令，在要访问存储器或输入输出设备时，向BIU发出地址以及数据读写的要求。

2. 总线接口单元BIU

总线接口单元BIU预先从存储器取出EU将要执行的指令(这动作称为预取)，预取的指令存入内部队列存储器中。8086CPU的内部队列有6个字节(8088为4个字节)。BIU在内部队列有空以及EU没向BIU取指令时，便从存储器预取指令。在通常情况下，EU和BIU可同时动作，从而提高了CPU的工作效率。但在EU执行转移指令、调用指令、返回指令等以后，BIU预取的指令就无效，必须重新预取指令。

2.2 寄存器组

8086的寄存器组完全包含8080及8085的寄存器组。图2.2画出了8086的寄存器组。其中斜线部分是8080、8085中也具有的寄存器。

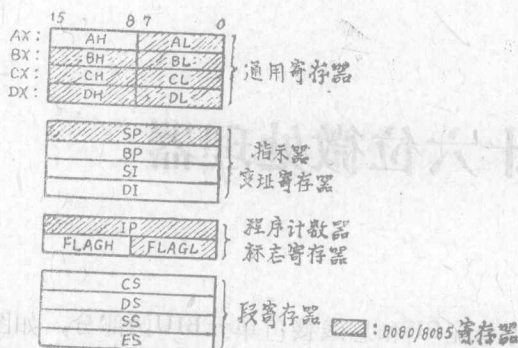


图2.2 8086 寄存器组

AX、BX、CX、DX 寄存器是16位通用寄存器，主要用于算术逻辑运算。这些寄存器可分为高8位或低8位来使用，因此便于字节数据的处理。对于不同的指令，这4个16位通用寄存器的作用各不相同，如下所示：

- AX: 累加器
- BX: 基址寄存器
- CX: 计数寄存器
- DX: 数据寄存器。

堆栈指示器 SP、基指示器 BP、源变址寄存器 SI、目的变址寄存器 DI 是作为指示器和变址寄存器使用的，它们都是16位寄存器，除特殊作用外和通用寄存器一样用作算术逻辑运算。

指令指示器 IP 即程序计数器。CPU 可根据 IP 的内容访问代码段(64K字节)范围的程序地址。

标志寄存器也是16位的，各标志位的分配见图2.3。

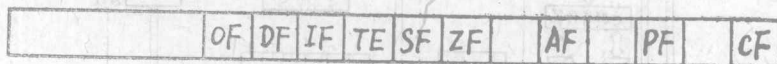


图2.3 标志寄存器结构

其中 CF 是进位标志位。运算指令执行时，当在最高位(字节运算时为 D₇，字运算时为 D₁₅)上产生进位、借位时，该标志位置“1”。

PF 是奇偶校验标志位。运算指令执行后，结果数值中含“1”的位数为偶数时，该标志位置“1”。

AF 是辅助进位标志位。运算指令执行中，从第3位到第4位有进位，或从第4位到第3位有借位时，该标志位置“1”。

ZF 是零标志位。运算指令执行后，结果为全零时，该标志位置“1”。

SF 是符号标志位。运算指令执行后，最高位的值为“1”时，该标志位置“1”。

TF 是陷阱标志位。该标志用于控制单步中断。在 TF = “1”时，每执行一条指令就产生单步中断。

IF 是中断允许标志位。该标志用于控制硬件中断(不可屏蔽中断除外)。在 IF = “1”时可以接受中断，IF = “0”时中断被屏蔽，不能接受中断。该标志通过 STI 和 CLI 指令置位和复位。

DF 是方向标志位。该标志用于指定字符串处理指令的方向。当 DF = “0”时，字符串处理由低位地址向高位地址处理；当 DF = “1”时，则从高位地址向低位地址处理。该标志位通过 STD、CLD 指令来置位、复位。

OF 是溢出标志位。运算指令执行后，结果数值超过能用2的补码(补数)所表示的范围(字节运算时为 -128~+127，字运算时为 -32768~+32767)时，该标志位置“1”。

8086可寻址的最大存储器空间是1兆字节。它可根据需要划分成许多段，每个段最大64K字节。这些段的起始地址都存放在段寄存器中。在8086中共有4个段寄存器，它们分别是代码段寄存器CS、数据段寄存器DS、堆栈段寄存器SS和附加段寄存器ES。这4个段寄存器分别与IP、SP、SI、DI、EA(有效地址)一起形成访问存储器的20位物理地址(见图2.4)。段

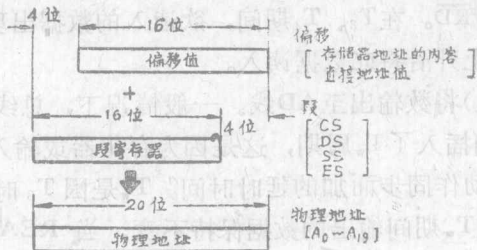


图2.4 物理地址的形成

寄存器与指示器、变址寄存器、有效地址的组合不是任意的而是有一定的规则，如表2.1所示。表中无段超越前缀一栏是指某条指令的编写中，不使用段超越前缀来指定所使用的段寄存器。在这种情况下，IP一定与CS相组合产生一个20位的地址；BP一定与SS相组合产生一个20位的地址等等。如果在指令中不加上段超越前缀，这种组合关系

是固定不变的。但为了程序编制的灵活性，要求这种组合关系不要一成不变。当在指令中加上段超越前缀，则可部分地改变这种情况。例如指令
MOV AL, [BP+0000] (代码为8A4600) 中没使用段超越前缀，则该指令功能是将 $SS \times 2^4 + [BP+0000]$ 地址(一个20位的地址)中的内容送到AL寄存器。如该指令加上CS段超越前缀，则指令

MOV AL, CS:[BP+0000] (代码为2E8A4600) 的功能是将 $CS \times 2^4 + [BP+0000]$ 地址中内容送到AL寄存器。

表2.1 段超越前缀的用法

存 储 器 存取方式	20 位 物 理 地 址	
	无段超越前缀	段 超 越 前 缀
代码(指令)地址	$(CS) * 2^4 + (IP)$	不能加段超越前缀
堆栈地址	$(SS) * 2^4 + (SP)$	不能加段超越前缀
数据存取 (BP的间接寻址方式除外)	$(DS) * 2^4 + EA$	$(ES)(或(SS)、(CS)) * 2^4 + EA$
数据存取 (BP的间接寻址方式)	$(SS) * 2^4 + EA$	$(ES)(或(DS)、(CS)) * 2^4 + EA$
字符串处理(源)	$(DS) * 2^4 + (SI)$	$(ES)(或(SS)、(CS)) * 2^4 + (SI)$
字符串处理(目的)	$(ES) * 2^4 + (DI)$	不能加段超越前缀

2.3 总线动作和定时

8086CPU 的一个总线周期至少包含有4个时钟周期。这4个时钟周期就称为 T_1 、 T_2 、 T_3 、 T_4 。在 T_1 期间CPU发出地址， T_3 、 T_4 期间在总线上进行数据传送。如果所寻址的存储器

或输入输出设备速度较慢，则需要较长的数据传送时间，因此在其发出“准备好”信号之前，CPU可在 T_3 、 T_4 之间增加等待周期 T_w 。每个 T_w 也为一个时钟宽度。

8086读周期和写周期的总线定时如图2.5所示。

读动作时，8086CPU在 T_1 期间从分时使用的地址数据(AD)总线上输出20位的地址。在 T_2 期间，使AD总线呈高阻态，并输出读控制信号 $\overline{RD}$ 。在 T_3 、 T_4 期间，欲读入的数据出现在AD总线上。CPU在 T_4 周期开始，也即 $\overline{RD}$ 信号的上升沿将此数据读入。

写动作时， T_1 周期将地址、 T_2 周期(延续到 T_4)将数输出至AD线。一般情况下，总线周期从 T_1 开始到 T_4 结束。图2.5(b)中，在 T_3 、 T_4 之间插入了 T_w 周期，这是因为存储器或输入输出设备的数据传送速度跟不上8086的速度，为使动作同步而加的延时时间。 T_w 是因 T_3 时准备就绪(READY)引线输入为低电平而引起的。在 T_w 期间输出的数据保持不变。当READY输入变为高电平时，CPU的定时就移至 T_4 周期。

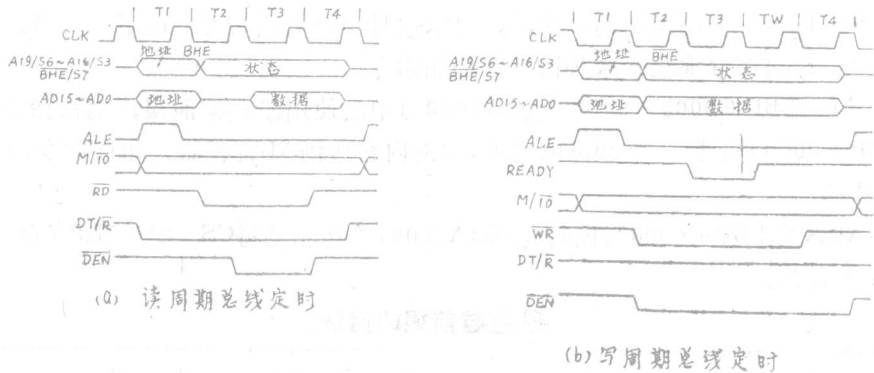


图2.5 读、写周期总线定时

2.4 8086引线端功能

图2.6 是8086的外引线图和名称。

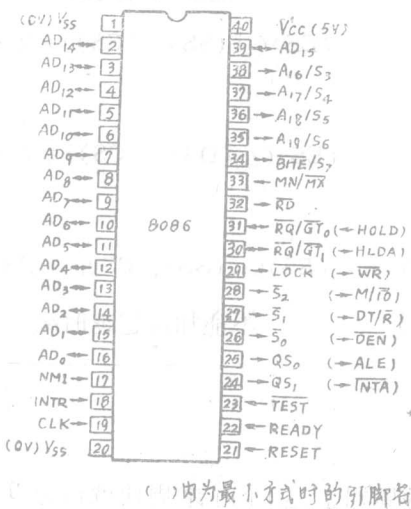


图2.6 8086外引线图

8086的工作方式可分为最小方式和最大方式二种，这由图2.6中MN/MX引线接高电平还是低电平来决定。8086的一部分外引线在这二种工作方式下功能是相同的，一部分外引线在这二种工作方式下功能是不同的，下面分别说明。

1. 二种工作方式下功能相同的外引线功能说明

AD₁₅~AD₀。——地址数据总线(输入输出)。分时用作存储器或输入输出设备的地址和数据线。在时钟周期 T_1 期间输出地址，在 T_2 — T_4 期间输入或输出数据。在“中断响应”和局部总线“保持响应”期间，输出高阻态。

A₁₉/S₆~A₁₆/S₃ ——地址状态线(输

INTA——中断响应(输出)。从此引线输出低电平,表示外来的中断请求已被接受。

ALE——允许地址锁存(输出)。为了把在AD总线上输出的地址信息锁存到锁存器,CPU从此引线输出允许锁存信号(高电平),它在时钟周期 T_1 期间有效。

DT/R——数据发送/接收(输出)。在使用8286/8287数据总线收发器的最小方式系统里,需要数据发送或接收,从此引线输出的信号(DT/R)用来控制数据流过收发器的方向。

DEN——数据允许(输出)。在使用收发器的最小系统里,由此引线输出的**DEN**信号提供给8286/8287,打开其通路。在每个访问存储器和输入输出设备读写周期以及**INTA**周期的 $T_2 \sim T_4$ 期间输出有效(低电平)。

HOLD, HLDA——保持请求(输入),保持响应(输出)。从保持请求引线输入其他主控器要求使用总线的请求信号(**HOLD**)。CPU收到保持请求信号后就经保持响应引线输出**HLDA**有效信号,表示请求已被接受。与此同时,CPU使总线和控制线处于高阻态,让出总线的使用权。在**HOLD**信号消失后,CPU中止**HLDA**信号,同时收回总线使用权。

3. 与最大工作方式(MN/MX为低电平)有关的外引线功能说明

S₂, S₁, S₀——状态(输出)。这3条引线输出信号的组合表明现在CPU执行的总线周期类型,如表2.3所示。

表 2.3

CPU总线周期类型

S_2	S_1	S_0	CPU 周 期
0	0	0	中断响应
0	0	1	读输入输出
0	1	0	写输入输出
0	1	1	暂 停
1	0	0	代码存取(读存储器)
1	0	1	读存储器
1	1	0	写存储器
1	1	1	非总线周期

RQ/GT₀, RQ/GT₁——请求/同意(输入输出)。这些引线在其他主控器向CPU请求使用总线时使用。每条引线都可双向传送信号。**RQ/GT₀**的优先级比**RQ/GT₁**高,**RQ/GT**的动作过程如下:

①一个主控器向**RQ/GT**引线发出一个时钟宽度的低电平,表示向CPU请求使用局部总线。

②CPU在下一个 T_4 或 T_1 期间,经本引线输出一个时钟宽度的脉冲,表示接受请求。同时,使总线处于高阻状态。

③提出总线请求的主控器使用完总线后,再向**RQ/GT**引线发出一个时钟宽度的脉冲,表示保持请求结束。CPU在下一个时钟周期开始,可再使用总线(收回总线使用权)。

LOCK——封锁(输出)。当此引线输出低电平(**LOCK**信号)时,表示本CPU封锁其他CPU使用总线。该信号一般用于多个处理器共用一个存储器时,防止同时访问存储器而用的。当在一条指令之前加上**LOCK**前缀(代码为**F0H**)时,**LOCK**引线就输出低电平,直到这条指令执行完。

QS₁、QS₀——队列状态(输出)。从引线**QS₁、QS₀**输出的信号向外部提供CPU中内部指令队列的信息:

QS₁ QS₀

- 0 0 队列无操作,即保持不变。
- 0 1 从队列取指令的第一个字节。
- 1 0 队列空。
- 1 1 从队列取指令第二个以后的字节。

2.5 中断

8086最多可处理256种中断。中断除外部中断外,还可由软件产生中断。此外,在某些特定条件下,CPU本身也会产生中断。中断产生的原因见图2.7

1. 外部中断

外部中断可通过8086的**INTR**引线和**NMI**引线通知8086。**INTR**一般与可编程中断控制器8259A相接。8259A又与各个中断源相接,接受中断源提出的中断请求,并将最高优先级的中断送至**INTR**,向CPU提出中断请求。

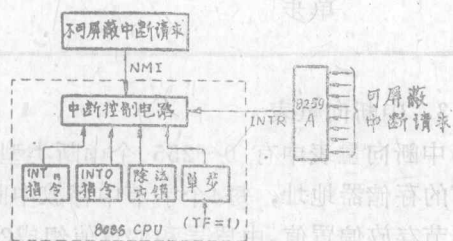


图2.7 8086中断源

一旦**INTR**有效,则CPU在执行本指令的最后时钟周期,根据允许中断标志位**IF**来决定是否接受此中断。若此时**IF**="0",CPU不接受中断而继续执行下面的指令;若**IF**="1",CPU接受该中断,进入中断处理,同时经**INTA**引线向外发出应答信号**INTA**。

INTR中断可由**CLI**(Clear **IF**)指令加以屏蔽,由**STI**(Set **IF**)指令解除屏蔽。

CPU接受中断后,发出二个**INTA**信号。第一个**INTA**通知8259A中断已被接受。第2个**INTA**要求8259A向总线发出选中的中断源的编号(1个字节),CPU读取该编号(中断类型码),并根据该编号转入相应的中断处理程序。

另一个外部中断请求输入引线是**NMI**。该引线通常是在出现电源波动,存储器出错,总线奇偶出错时通知CPU用的。**NMI**中断不可以屏蔽,比**INTR**的优先级也要高。

NMI中断类型码已预先确定,CPU不再要求其类型码,所以CPU不用发**INTA**信号。

2. 内部中断

内部中断有如下4种:

程序中可设置**INT**指令来产生中断。这时,中断发生在该指令执行的终了,也即软件中断。

当溢出标志位**OF**为"1",则在执行**INTO**指令时就产生中断。

在执行除法时,商超过规定的范围,CPU就产生中断。

如自陷标志 $TF = "1"$ ，则CPU每执行1条指令，就自动产生中断，此即单步中断，是很有力的调试手段。

内部中断有如下特点：

- ① 中断类型码包括在指令中或事先已确定。
- ② 不输出 $INTA$ 信号。
- ③ 除单步外，都是不可屏蔽的。
- ④ 除单步外，优先级都比外部中断高。

中断优先的次序如表2.4所示。

表 2.4

中断优先级次序

中 断	优 先 级
除法出错, $INTn$, $INTO$	最 高
NMI	↑
$INTR$	
单步	最 低

3. 中断向量表

中断向量表中有 $0 \sim 255$ 个中断类型，每个中断类型占存储器4个字节，故该表占有 $1K$ 字节的存储器地址，每4个字节中存放中断处理程序的入口地址。高2个字节存放段号，低2个字节存放偏置值，由段号和偏置值组成20位的中断处理程序入口地址（见图2.8）。

一旦发生中断，CPU就将标志寄存器的内容保存至堆栈区，代码段 CS 和程序计数器 IP 的内容也保存至堆栈中。根据中断类型码，从向量表中取出相应的段号和偏置值送至 CS 和 IP 中去，从而实现了转入中断处理程序的目的。

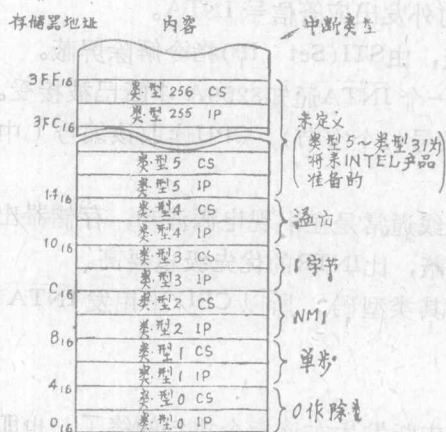


图2.8 中断向量表

4. 单步中断

若自陷标志位 TF 为“1”，则CPU就处于单步工作方式，每执行一条指令，就自动地产生类型1的中断。CPU将标志寄存器内容保存在堆栈之后，清除 TF 和 IF 。然后进入单步中断处理程序，单步中断处理一结束，由 $IRET$ (Interrupt Return) 指令恢复标志寄存器内容，CPU再返回单步工作方式。

单步动作是非常有用的调试程序的手段。例如在单步中断处理中可把每条指令执行后的寄存器内容， IP 的值，存储器中的变量等显示在 CPT 上提供给程序员。

8086中无直接置位或复位 TF 的指令，要改变其状态首先要把 TF 保存至堆栈，然后对堆

栈中该值进行修改。标志位进出堆栈可用PUSHF, POPF指令。

2.6 其他控制功能

1. 复位

8086一旦在复位引线上检出高电平,就中止全片的动作,进入复位状态。复位后CPU的状态如表2.5所示。复位后IP=0000H, CS=FFFFH。所以在复位引线输入电平由高到低后,CPU 即从FFFF0H地址开始执行程序。

表 2.5 复位后 8086 的状态

标志位	全部清除
IP寄存器	0000H
CS寄存器	FFFFH
DS寄存器	0000H
SS寄存器	0000H
ES寄存器	0000H
指令队列	清除

2. 暂停

CPU一执行HLT(Halt)指令,就进入暂停状态,停止一切动作。暂停状态一直保持到发生中断或系统复位时为止。CPU在暂停状态照样可接受HOLD引线(最小方式)或RQ/GT引线(最大方式)来的保持请求。保持请求消失后,CPU仍回到暂停状态。

3. 总线封锁

8086在最大方式时可产生总线封锁信号LOCK。它可以消除几个CPU共用存储器资源时引起的冲突。CPU在执行一个字节的封锁前缀LOCK(代码为FOH)时,就从引线发出此信

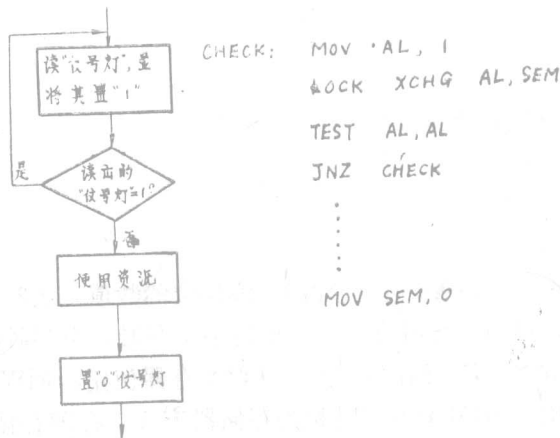
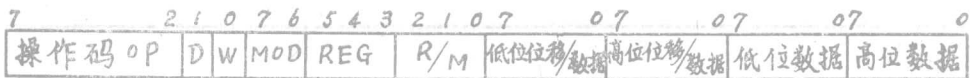


图 2.9 XCHG和LOCK使用举例

号，它一直保持到紧跟封锁前缀后的一条指令执行完。这时连接在公共总线上的其他处理器就不能使用总线。最典型的封锁前缀的使用例子如图2.9所示。这里使用了封锁前缀和XCHG指令来实现对总线的控制，以避免使用共有资源时产生的冲突。图中SEMAPHORE(信号灯)是共有存储器中的一个单元。当其内容为“0”，表示共有存储器可使用(即没有CPU在使用共有存储器)；当为“1”时，表示不可使用(即有CPU正在使用共有存储器)。规定系统中各CPU在使用共有存储器时要置“1”SEMAPHORE，用完后再将其置“0”。各CPU在使用共有存储器前必须先检查SEMAPHORE内容是否为“0”，是“0”可使用，否则就等待。为检查SEMAPHORE的状态，使用XCHG指令将SEMAPHORE状态取至AL寄存器。由于XCHG指令要进行8位数据的交换，须2个总线周期，而在这2个总线周期期间，有可能其他CPU也要检查SEMAPHORE。为了避免冲突，就在XCHG指令前加上LOCK前缀，这在XCHG执行的整个过程中，保证其他CPU不会使用总线去检查SEMAPHORE的状态，从而避免使用共有存储器而形成的冲突。

2.7 寻址方式

8086微处理中有下述几种寻址方式。这些寻址方式是指操作码中的MOD(2位)和R/M(3位)共5位字段来指定的(见图2.10)



其中 D 指定操作数是源还是目的
W 指定是字指令还是字节指令
MOD, R/M 定义寻址方式
REG 指定寄存器操作数

图2.10 机器字指令的一般形式

- (1) 寄存器
- (2) 直接地址
- (3) 寄存器间接地址(变址及基址)
- (4) 基址+变址
- (5) (基址或变址)+位移
- (6) 基址+变址+位移

这里提到的位移是指在指令的末尾所加的8位或16位的数值。表2.6列出了8086的各种寻址方式。当MOD=11时，称为寄存器方式，存取通用寄存器。寄存器的指令是由R/M字段进行的，在W=1时，选择AX~DI寄存器，作为16位寄存器使用，而W=0时，选择AL~DH寄存器作为8位寄存器使用。当MOD≠11时称为存储器方式，存取存储器。在MOD=00，R/M=110时为直接地址寻址方式，其余均是间接地址寻址方式。当MOD=00时没有位移，