INTEL 8086

微处理机与微计算机

（上册）

王兆全　　编写

国营第五七四厂研究所
一九八三年七月

前　言

自70年代初，出现微处理机以来，在十年左右的时间内，微型机的发展十分迅速。对科学技术和社会生活产生了广泛的影响，大规模集成电路是微处理机发展的支柱，并且它正以每年功能增加一倍，而价格下降一半的速度发展着。从71年四位微处理机诞生和72年英特尔公司的雏型8位微处理机出现，又经历了以INTEL8080／8085、M6800和Z—80为代表的8位的第二代微处理机。到78年以后又相继出现了INTEL8086、MC68000和Z—8000为代表的16位第三代微处理机。目前英特尔公司的32位微处理机已研制出来。也即将投放市场。最近两年我国微处理机的研究和应用工作也得到迅速地发展。国内已研制出了 DJS050和DJS060两个系列的微处理机。目前国内8位微处理机的教材和资料已经比较完整、丰富了。但是有关16位微处理机的资料还很缺乏，许多同志很想得到这方面的资料。作者在工作过程中整理编写了这本资料。本文主要介绍16位的INTEL8086微处理机。全书共分四章。第一章介绍8086cpu芯片的结构、性能和工作原理。第二章介绍8086的寻址方法和指令系统。对每条指令都给出了ASM—86汇编语言的汇编格式、指令的二进制编码、指令的操作、指令执行的周期和指令执行后对状态标志位的影响。对每条指令还做了必要的说明。第三章对组成微计算机的一些主要的芯片电路作了介绍。如存贮器、时钟、串行、并行接口、中断管理和总线管理等芯片。第四章对英特尔公司生产的以8086cpu为中心的16位的单板计算机ISBC86／12A 从电路结构和工作原理上做了较详细地介绍。

本文对于想了解8086的读者会有一定的帮助。扬廷善和綦希林同志审阅了本文，对其中不妥之处给予了指正。在此向这两位同志表示衷心地感谢。

由于编写时间较短，特别是编者水平有限，一定会有很多错误和不妥之处，恳请读者批评指正。

王兆全

1981.8

目　录

第一章 8086 微处理机

第二章 8086 微处理机指令系统

第一章 8086微处理机

1.1 概　述

8086微处理机是美国INTEL公司1978年投放市场的产品。8086CPU 性能的提高依赖于工艺和系统结构的改进（见：J·Mckevltt和J·Bayliss：“New Options from Big Chips”，IEEE Spectrum· Vol、16，NO、3.1979·3），以及两者之间的结合。8086CPU是用新开发的硅栅H－MOS工艺（N－channel depletion load·silicon gate technology，制造的第一台微处理机。H－MOS工艺使场效应晶体管按4微米的规模设计，并采用片上偏置的方法来提高器件的操作速度。并能使工作更加可靠。H—MOS工艺还提高了电路密度。整个16位数据处理和微程序控制机构（21位长微指令）所用的29000只晶体管全部集成在225平方密尔的芯片上。由于采用了这种高性能的H—MOS技术，片上的门延迟时间为2ns，这与高价格的STTL（肖特基TTL）门延迟时间相同，从而使片内的时钟频率高达5MHZ（200ns），在优选的情况下也可选用8MHZ（125ns）。比当时任何单片微处理机都快。由于4个CPU时钟脉冲周期相当于一个存贮周期，故8086CPU寻址存贮器的效率是很高的。8086选用的存贮器片子要求其存取周期为500～800ns，取数时间（从地址建立到数据形成的时间）为295～400ns。

8086是中档的8080／8085系列向16位的扩展。处理机的片子兼有8位和16位的处理能力。它的指令系统既包含了8080／8085的全部指令，又新增加了一套强有力的处理16位的指令。熟悉现有8080／8085的人员，只要应用与8080／8085基本相同的软件包和开发工具，就能把系统功能提高10倍。见图1—1。

8086微处理机采用40线双列直插式封装。寻址范围可达1兆字节。具有一个独立的64K字节的I／O空间。

为了减少芯片上的引线端的数目，采用了分时重迭使用的多总线结构，即地址总线，数据总线和状态控制总线。

8086CPU在一个总线周期中既可以传送8位也可以传送16位的数据，具有较大的输入输出能力。8086CPU有两种工作模式。工作模式的选择通过第33#腿（MN／$\overline{MX}$）来控制。这两种工作模式为：最大模式和最小模式。为了进一步提高8086CPU的工作速度，在8086CPU中有一个能够存放6个字节的指令队列（后续指令）寄存器。采用指令提前取出的技术。可予先取得一些指令码和操作数。

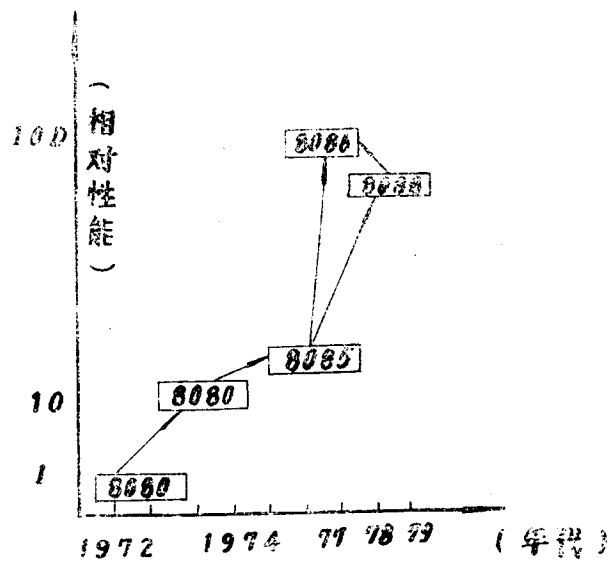


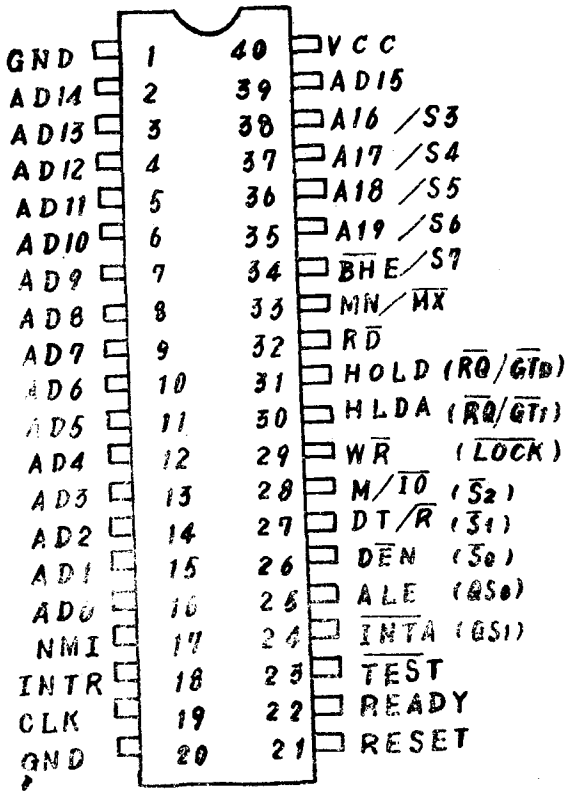
图1—1 INTE／系列CPU
相对性能

8086的中断功能很强。采用中断向量方式，每个中断向量由4个字节组成，共可得到256（2^8）个不同形式的中断。8259A可编程（Programmable interrupt cpntroller）中断控制器，与8086相兼容，使8086CPU对中断管理更加方便灵活。8259A可以接收8个不同的中断源的中断请求。如果将8259A做适当的级联，就可以接受64个中断源的信号。对每个中断源都给定一个优先权（级），根据这个优先权，系统就可以判断出各个中断源请求中断处理的紧急程度，中断优先权可以由软件来设定。

8086CPU可以接成多处理机工作方式。这对于多任务的复杂系统是很有用的。系统的某个任务可以指定给某个特定的微处理机。这样既简单而又快，同时处理任务的效率也高。组成多处理机系统还可以对高级任务进行并行处理。

为了使8086CPU的能力能充分地发挥出来，INTEL公司还提供了可供选择的多种功能的外围芯片，以便组成各种实用微计算机系统。为了更好地开发8086的软件和硬件，INTEL公司软件配备齐全，提供有完善的开发工具，为用户提供了十分便利地使用条件。就这点而论在微型计算机系统方面，它是处于领先地位的。

（下转第三页）



BHE	A_0	
0	0	整字
0	1	高8位对应于奇数地址
1	0	低8位对应于偶数地址
1	1	没有

<> <> <>

1.AD15—AD0（三态输入输出）数据／地址总线：

这些引脚构成了分时的M/IO（Memory／input output：存贮器／输入输出）地址（T_1）和数据（T_2、T_3、TwT_4）的多重总线。A_0对于数据总线的低位字节（引脚D_7-D_0）来说其作用与$\overline{BHE}$类似。T_1时间A_0是低电平，表示在存贮器或I/O的各种操作里，把一个字节传送到总线的低位部分去。按8位操作的设备，要依赖低位字节用A_0来形成片选功能。AD_{15}—AD_0这些引脚是高电平有效的，并在中断响应和局部总线保持响应（hold acknow ledge）期间，这些线浮离呈高阻抗状态。

1.2 8086CPU的结构

1.2.1 8086GPU管脚功能说明

（上接第二页）

8086CPU（Central Processing Unit——中央处理单元）被封装在一个单片电路上。这个单片电路采用40线标准的双列直插方式的封装（见图1—2）。这片电路只需要一个+5V的电源来供电。在介绍8086CPU的功能之前，我们先给出8086CPU芯片管脚的名称（见图1—3）和给出有关8086CPU管脚功能的说明。

下面所述的管脚功能即适用于8086CPU的最小模式也适用于最大模式。说明中的“局部总线”（Local bus）是指直接连接到8086的多重总线接口而不考虑附加的总线缓冲器。

公共信号		
名称	功能	类型
$AD_{15}\sim AD_0$	地址／数据总线	双向三态
$A_{19}/S_6—A_{16}/S_3$	地址／状态	输出三态
$\overline{BHE}/S_7$	总线高位允许／状态	输出三态
MN／$\overline{MX}$	最大／最小模式控制	输入
$\overline{RD}$	读控制	输入三态
$\overline{TEST}$	测试端控制等待	输入
READY	等待状态控制	输入
RESET	系统复位	输入
NMI	非屏蔽中断请求	输入
INTR	中断请求	输入
CLK	系统时钟	输入
VCC	+5V	
GND	地	

最小模式信号（MN／MX=VCC）		
名称	功能	类型
HOLD	保持请求	输入
HLDA	保持响应	输出
$\overline{WR}$	写控制	输出三态
M／$\overline{IO}$	存贮器／IO控制	输出三态
$\overline{DT}/R$	数据发送／接收	输出三态
$\overline{DEN}$	数据允许	输出三态
ALE	地址锁存允许	输出
$\overline{INTA}$	中断响应	输出

最大模式信号（$\overline{MN}$／MX=CND）		
名称	功能	类型
$\overline{RQ}/GT1.0$	请求／允许总线存贮控制	双向
$\overline{LOCK}$	总线优先权锁定控制	三态输出
S_2-S_0	总线周期状态	三态输出
QS_1,QS_0	指令队列状态	输出

图1—3　　8086芯片引脚定义

2.A19／S6、A18／S5、A17／S4、A16／S3（三态输出）

在T_1期间，这四条引脚对存贮器操作来说是十分重要的地址线。在I／O操作期间。这些引脚将保持低电平。在T_2、T_3、Tw和T_4周期中。在执行存贮器和I／O操作时，作为状态信息线。在每个时钟周期开始时修改允许中断状态标志位（S_6）的状态。$A_{17}／S_4$和$A_{16}／S_3$组合编码后的功能如下：

$A_{17}／S_4$	$A_{16}／S_3$	
0（LOW）	0	交换数据
0	1	堆栈
1（high）	0	代码或没有
1	1	数据

S_6 是0（LOW）

这些信息表示再定位寄存器现在正用来存取数据。在中断询问和局部总线“保持响应”期间，这些线浮离呈高阻抗状态。

3.$\overline{BHE}／S_7$（输出、三态）

在T_1期间，高位总线生效（允许）信号$\overline{BHE}$（bus high enable）用于使数据加到数据总线高8位即D_{15}～D_8上，使送数据依赖于高8位的设备，通常使用$\overline{BHE}$来实现片选功能。当在读、写和中断应答周期时，为了传送一个字节到高位总线去，在T_1期间$\overline{BHE}$应为低电平。在T_2、T_3和T_4期间，S_7的状态信息是有效的。这个信号在低电平时有效，并且在“保持”期间浮离呈高阻抗状态。在第一个中断询问周期的T_1期间S_7为低电平。

4.$\overline{RD}$（输出、三态）读

出现读选通脉冲时，处理机是执行存贮器读周期还是执行I／O读周期，由S_2引脚的状态决定。$\overline{RD}$信号用来读取属于8086局部总线上的组件中的内容，在T_2、T_3和Tw或任何读出周期期间，$\overline{RD}$为低电平时才有效，但当8086局部总线浮离呈高阻抗状态之前，在T_2期间保持高电平。

在“保持响应”期间这个信号浮离呈高阻抗。

5.READY（输入）准备就绪

READY信号是从被寻址的存贮器或I／O设备来的应答信号，以便完成数据的传送。RDY信号由M／IO送来，由8284时钟发生器来同步而形成准备就绪READY信号。这个信号是高电平有效。

6.INTR（输入）中断请求

中断请求是一个触发输入电平，它在每条指令的最后一个时钟周期被采样，以确定处理是否应当进入中断询问操作。通过查看系统存贮器中中断矢量表的不同位置来得到一个中断服务程序，可以用软件将允许中断状态标志位清除，达到中断屏蔽的目的。INTR由内部来同步。INTR是高电平有效。

7.$\overline{TEST}$（输入）测试状态

TEST由外部提供，这个信号由“等待测试”指令来检查。如果TEST这个输入信号为低电平，就继续执行“等待测试”后面的指令。如果$\overline{TEST}$为高电平处理机就进入一个空闲等待状态，重复执行“等待测试”指令。这个输入信号是由每个时钟的上升沿来览发同步的。

8.NMI（输入）非屏蔽中断

NMI是由边沿触发的一个外部输入信号。它能引起一个方式2的中断。通过一个中断矢量来得到中断子程序在存贮器中的入口。NMI的意思是不能用软件来将其屏蔽（Non-maskable interrupt），在当前指令的尾部一个从低电平变为高电平的信号就能起动中断。这个输入信号是由内部来同步的。

9.RESET（输入）复位信号

RESET引起处理机立即结束它的现行操作。复位信号必须保持其有效的高电平至少要有四个时钟周期。当RESET回到低电平时，处理机重新再启动。RESET是内部同步的。

10.CLK（输入）时钟信号

CLK时钟信号为处理机和总线控制器提供了基本的时序。它是非对称的，具有33%的占空比，提供了一个最佳的内部工作时序。

11.VCC电源

VCC是+5V±10%的电源输入引脚。

12.GND地

GND是接地引脚。

下面所说明的管脚的功能是8086处于最小模式（MN／$\overline{MX}$=VCC）工作方式的情况下。而且仅是在最小模式下来说明这些管脚的功能的。所有其它的管脚功能都如前面所述。

13.M／$\overline{IO}$（输出、三态）

此信号在逻辑上等效于最大模式工作情况的S_2。用其来区分是从存贮器中存取，还是从IO设备中存取。M／$\overline{IO}$在前一个总线周期的T_4状态变为有效，直到本周期的T_4状态仍然有效（M=高电平，IO=低电平）。在局部总线“保持响应”时，M／$\overline{IO}$浮离呈高阻抗状态。

14.$\overline{WR}$（输出三态）写

写选通脉冲表示：处理机正在执行存贮器写或I／O写周期。由M／$\overline{IO}$信号来决定是存贮器写还是I／O写。$\overline{WR}$对任何写周期的T_2、T_3和Tw有效。$\overline{WR}$本身是低电平有效，在局部总线“保持响应”状态时，$\overline{WR}$浮离呈高阻抗状态。

15.$\overline{INTA}$（输出三态）中断响应信

此信号由处理机发出，$\overline{INTA}$做为一个读选通脉冲用于中断响应周期。在每个中断响应周期的T_2、T_3和Tw，$\overline{INTA}$是低电平有效，中断响应信号在“保持响应”时，$\overline{INTA}$处于浮离状态呈高阻抗。

16.ALE（输出）允许地址锁存

ALE信号由处理机提供，用此信号将地址锁存在8282／8283地址锁存器中。在任何总线周期的T_1状态，它是高电平有效的。注意ALE信号永远不会浮离。

17.DT／$\overline{R}$（输出、三态）数据传送／接受

DT／$\overline{R}$需要在最小模式工作方式下，用8286／8287数据总线收发器在数据总线上传送／接收信息。DT／$\overline{R}$用来控制在数据总线上通过收发器数据的传输方向。在逻辑上，DT／$\overline{R}$信号等效于在最大模式时$\overline{S_1}$信号。它的时序类似于M／$\overline{IO}$（T=高电平、R=低电平）。在局部总线为“保持响应”时DT／$\overline{R}$浮离呈高阻抗状态。

18.$\overline{DEN}$（输出、三态）

在使用收发器的最小系统里，把$\overline{DEN}$作为输出信号提供给8286／8287。在每个存贮器和I／O存取和INTA周期$\overline{DEN}$是低电平有效。对于一个读和INTA周期，从T_2状态的中间直

到T_4状态中间的$\overline{DEN}$有效，而对于一个写周期从T_2状态的开始直到T_4状态的中间都是有效。在局部总线“保持响应”期间$\overline{DEN}$浮离呈高阻抗。

19.HOLD（输入）HLDA（输出）

HOLD指示另一台主控制机正在申请局部总线“保持”。为了响应、HOLD必须是高电平有效接收到“保持”申请的处理机，作为响应请求在T_4的中间或T_1发出HLDA（高）“保持响应”。与发送HLDA的同时，处理机将使局部总线和各控制线浮离。低电平的HOLD被检测后，就使HLDA降为低电平，并且当处理机需要起动另一周期时，将再启动局部总线和各控制线。

下列引脚功能说明是对8086／8288系统最大模式工作而言（即MN／$\overline{MX}$ = VSS）。而且仅是在最大模式下来说明这些管脚的功能的。所有其它的引脚功能则如前面所述。

20.$\overline{S_2}$、$\overline{S_1}$、$\overline{S_0}$（输出、三态）

这些状态线编码如下：

$\overline{S_2}$	$\overline{S_1}$	$\overline{S_0}$	
0（LOW）	0	0	中断响应
0	0	1	读I／O端口
0	1	0	写I／O端口
0	1	1	暂　　停
1（High）	0	0	代码存取
1	0	1	读存贮器
1	1	0	写存贮器
1	1	1	无　　效

在T_4、T_1和T_2期间，状态是有效的。当READY是高电平时，在T_3或Tw期间返至无效状态（1，1，1）。8288总线控制器使用这个无效的状态产生所有的存贮器和I／O存取控制信号。在T_4状态$\overline{S_2}$、$\overline{S_1}$、$\overline{S_0}$任何一个状态若发生变化都表示一个新的总线周期的开始，而在T_3或Tw状态返回到无效状态就表示了总线周期的结束。

这个信号在“保持响应”期间浮离呈高阻抗。

21.$\overline{RQ}$／$\overline{GT_0}$、$\overline{RQ}$／$\overline{GT_1}$（输入／输出）请求／允许

这个“请求／允许”引线是提供给其它的局部总线控制器使用的，以迫使本处理机在它的现行总线周期结束时让出局部总线。这两个引脚都是双向的，只是$\overline{RQ}$／$\overline{GT_0}$比$\overline{RQ}$／$\overline{GT_1}$有较高的优先权。$\overline{RQ}$／$\overline{GT}$在内部已装了一个负载（PU－UP）电阻，所以外面可以不接了。请求／允许工作过程如下：

①来自其它局部总线控制器一个时钟脉冲宽度的信号就表示对8086CPU的一个局部总线请求（hoad）。

②在CPU的下一个T_4或T_1期间，从8086发出的一个时钟宽度的脉冲到发来请求的总线控制器，表示8086已经允许局部总线浮离，并且在下一个时钟周期进入“保持响应”状态。这样CPU的总线接口单元在逻辑上，在保持响应”期间与局部总线就分开了。

③当来自其它局部总线控制器的请求结束时，就发出1个时钟脉冲宽度的一个脉冲，表示“保持”请求结束，CPU就可以在下一个时钟周期再度控制局部总线，并且使CPU进入T_4状态。局部总线每进行一次总线控制器和总线控制器之间的交换需要三个脉冲，在每次交

换以后必须有一个空闲的时钟周期。$\overline{RQ}/GT$是低电平有效。

22.$\overline{LOCK}$（输出、三态）封锁总线

当$\overline{LOCK}$信号是低电平有效时，$\overline{LOCK}$引脚端的输出信号表示其它的系统总线控制器不能获得对系统总线的控制。$\overline{LOCK}$信号是由前缀指令“LOCK”来触发的，$\overline{LOCK}$信号一直保持到执行完下一条指令。$\overline{LOCK}$信号是低电平有效，并且在“保持响应”期间被浮离呈高阻抗。

23.QS_1、QS_0（输出）

QS_1和QS_0提供了几个状态，允许外部处理机跟踪内部的8086指令队列。

QS_1	QS_0	
0（LOW）	0	无操作
0	1	来自指令队列操作码的第一个字节
1（high）	0	指令队列空
1	1	来自指令队列的其它字节

执行队列操作后的时钟周期期间，队列状态是有效的。

1.2.2执行单元和总线接口单元

微处理机要执行存放在存贮器中的一段程序，简单地说一般要做如下的一些工作。首先由控制器提供一系列的时钟周期，在这周而复始的时钟周期控制下，有规律地一步一步重复执行下述过程：

① 从存贮器中取出一条指令

② 读出并分析所取得的指令操作码

③ 执行这条指令

④ 根据指令的要求如果需要的话写下结果

⑤ 重复①～④的过程。

在从前的微处理机中都是这样一步一步串行执行的。先取操作码，然后分析操作码、执行这条指令最后写下结果。微处理机在执行取操作码和写下操作结果时，是要占用总线的。而在分析操作码和执行指令的时间却不占用总线。从另一方面来看，取指和写下结果与分析操作码和执行指令这两种工作分别是由不同的电路来完成的。这样在串行执行指令的过程中它们都有空闲的时间。因此执行指令所花的时间就比较长。8080／8085就是采用这种串行交替的工作方式（见图1—4）。为了减少指令执行的时间，8086CPU采用了并行交替的工作方式。8086CPU由两个独立的处理单元组成。执行单元（execution unit）EU和总线接口单元（Bus interface unit）BIU。EU只负责执行指令，而BIU则负责从存贮器中读取操作码。操作数和将指令执行的结果写入指令所要求的地址单元中。这两部分互相之间可以独立地进行工作。在一些情况下就可以同时进行取指令和执行指令的操作，从而减少了指令执行过程中的一些时间。指令可以由BIU部件予先取得，当EU要执行这条指令时就不必到存贮器中去取了，而只需从EU中直接得到（见图1—4）。图1—4示出了8086CPU与一般传统计算机所不同的交替工作的关系图。这个例子示出了执行三条指令所用的时间。下面分别介绍一下8086CPU的执行单元EU和总线接口单元BIU。

执行单元EU

图1—5给出了8086CPU的逻辑框图。右半部为BIU，左半部为EU。执行单元EU包括一

个16位的算术／逻辑单元（ALU）、一个16位的反应CPU状态和控制状态的状态标志寄存器（Flags）、一组通用寄存器和指令操作码的控制电路。所有的寄存器和数据传输通路都是16位的，用它们进行快速的内部数据传送。EU不与系统总线（或称外部世界）相联，它是从BIU的指令队列寄存器中取得指令和数据的。同样，当指令要求将信息存入到存贮器或外部设备中，或要从存贮器或外部设备中取得信息时，EU就向BIU发出请求来完成这些操作。对于EU要求BIU寻址的数据的地址的位移量是由EU本身计算出来的。EU将这个16位的位移量送到BIU中，然后由BIU形成一个20位的实际地址，达到寻址的目的。

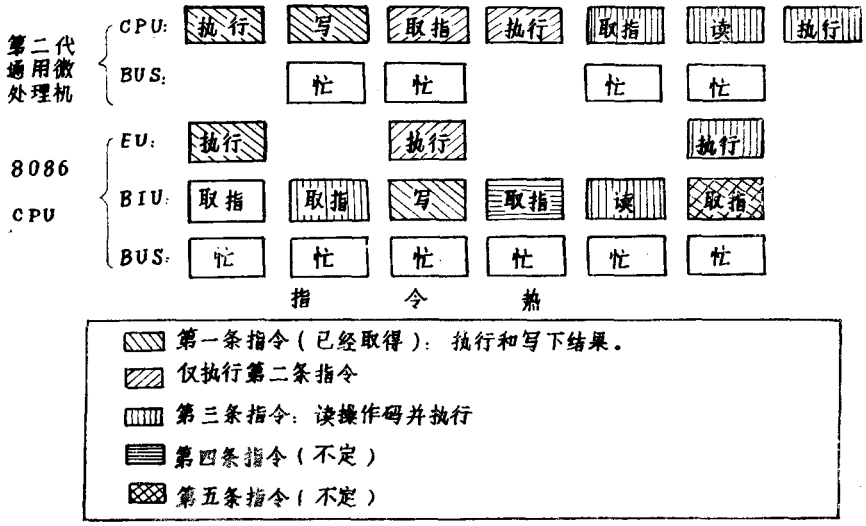


图1－4　交替取指和执行

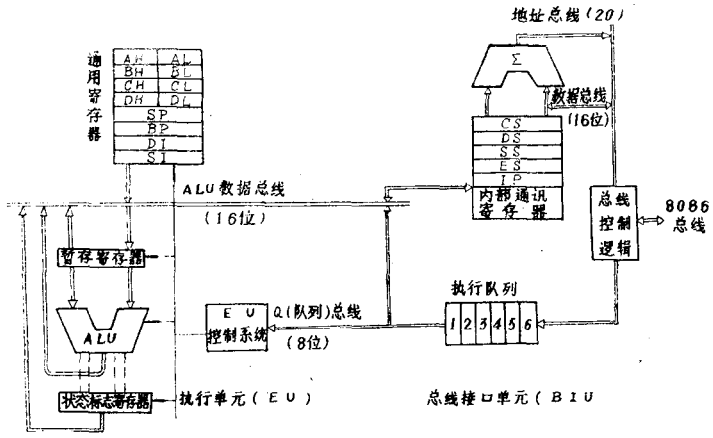


图1－5　8086CPU基本框图

总线接口单元BIU

总线接口单元BIU执行单元EU所要求的所有有关总线的操作。根据EU的命令BIU完成CPU与存贮器或I／O设备之间的数据传送。以后，我们还要详细地说明BIU和存贮器和I／O设备相互之间的工作关系。总线接口单元包括一组段寄存器、指令指示器、指令队列、地址产生器和总线控制器，在EU执行指令期间，BIU将向前看（looks ahead）从存贮器中预先先取得一些指令存放在指令队列（instruction stream queue）中。指令队列寄存器是一个六个字节的RAM存贮器，所以最多可放六个指令字节。取来的指令字节是顺序存放的。在大多数的情况下，BIU部件并不占用系统总线。系统总线可以提供其它部件使用。BIU一次可以取得两个指令字节。只要队列中有两个以上的指令字节空了的时候，不必由EU发出请求，BIU就会自动地去取指令。如果程序要求从奇效地址中取数，那么BIU就自动地从这个奇数地址中取出一个字节的数据，以后就顺序地从这个奇数地址后面的偶数地址中两个字节（字）两个字节地取数。

在大多数的情况下，指令队列中至少要有一个字节的指令，这样就可以使EU不必等待BIU去取指令。指令队列中所存放的都是即将被执行的指令。如果EU执行一条转移指令使程序产生转移时，存放在指令队列中的予先取来的指令就不能再被使用了。所要执行的指令要根据转移指令所给出的信息到一个新的地址单元中去取出新的指令来执行。在这种情况下EU就向BIU发出信息要求BIU从新的地址开始取指令，并将取得的第一个指令直接送到EU中。然后BIU再接着从这个新的地址开始的以后的地址中顺序取出指令重新填入到指令队列中，而指令队列中在转移指令以前所存放的指令就被冲掉了。这样就实现了程序的转移。

1.2.3总线操作和时钟电路

INTEL公司微处理机或微计算机的系统总线叫做多总线。多总线包括：地址总线、数据总线和控制状态总线。多总线是为了能在CPU存贮器、I／O以及外围控制等各种设备之间以高效兼容的方式进行数据传送。由于8086CPU也是采用40线封装的，所以地址线和数据线有些位是采用公共引脚的。又有些信号线则又要根据CPU的工作模式来决定它们的功能。采用多总线就能够很容易地构成各种系统。

总线操作

为了要说明分时（多）总线的操作，我们必须研究BIU的总线周期。从根本上来说总线主要是被用来完成信息传送的工作，从存贮器、I／O设备中读取数据，或向存贮器、I／O设备传送数据。这些事件是由执行各种不同功能的指令而随机产生的。如果要从存贮器或I／O设备中读取数据，那么首先就要给出所选定的存贮器或I／O设备的地址，接着再发出一个读控制信号，表示要从存贮器或I／O设备中读取（获取）数据。如果是在向存贮器或I／O设备写入（或称传送）数据时，那么首先要给出接受数据的存贮器或I／O设备的地址，接着就要发出一个写控制信号，同时给出要写的数据。在写周期，被选定的设备（存贮器或I／O）要在总线上接收数据，而在总线的读周期，被选中的设备要将数据送到总线上。在总线写周期的尾部，设备取得要写入的数据。在总线读周期的尾部，设备停止向总线送数。

如图1—6所示，每个总线周期至少由四个时钟周期组成。时钟周期也被称为T状态，我们用T_1、T_2、T_3和T_4来表示这四个时钟状期。在T_1状态期间，CPU将存贮器或I／O设备的地址置于总线上。在写总线周期时，从T_2到T_4这段时钟周期时间总线上一直保持要写的数据。在读总线周期期间，CPU在T_3和T_4状态期间在T_2受置于总线上的数据。在T_2状态时多路复用的地址/数据总线被浮离，并且允许CPU从写方式(输出地址)改变成为读方式(输入数据)。

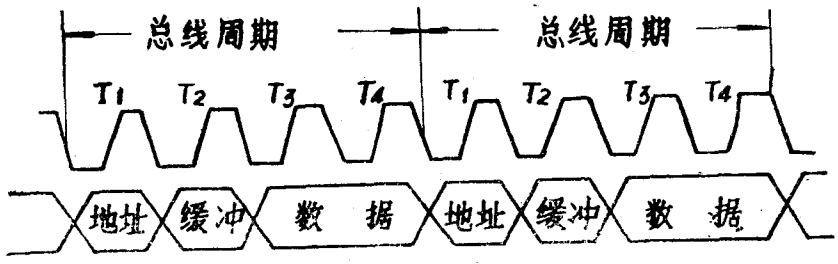


图1—6典型的BIU总线周期

这一点是很重要的，我们必须要注意，BIU只有在当指令的执行过程中根据指令的需要EU请求一个总线周期的时候，或者是必须要填写指令流队列的时候，才执行一个总线周期。然而时钟周期却是一直存在的。那么在两个总线周期之间仍然有BIU不执行任何操作的时钟周期。这些不起作用的时钟周期我们称之为空闲状态，用Ti（idle state）表示。有几种不同的条件可以产生空闲状态。如当允许外部的处理机存取总线时，本机将产生空闲状态。再如，在执行一条长指令时亦如此。

当8086CPU与慢速的存贮器或I／O设备交换数据时，由于8086CPU所选用的时钟频率很高。当CPU在T_1状态发出地址信号以后，要求在T_2—T_4状态期间完成数据的传送。在读总线周期由于存贮器或I／O设备工作速度比较低，在T_2—T_4状态期间可能还没有将要被读的数据置于总线上，或者是在写总线周期时的T_2—T_4状态期间，存贮器或I／O设备还没有来得及将数据取走，而写周期就结束了，这样就会产生数据传送时丢失的现象。为了避免这种情况的发生，8086CPU在T_3和T_4状态之间插入了一些必要的Tw（等待）状态。用来给予必要的时间补偿。在一个等待状态期间，数据在总线上保持不变。当设备完成了数据传送（发出或者接收了数据）时，就通过CPU片子READY管脚端发来一个信号，通知CPU退出等待状态而进入T_4状态。

下面我们来分析8086CPU在最小模式工作条件下读写总线时序。图1～7是8086读总线周期时间关系波形图。图1—8是8086写总线周期时间关系图。如图所示在T_1状态期间8086CUP在多路复用的地址／数据总线上给出一个20位的地址。在T_2状态期间，CPU将地址信号从总线上收回，后16位的地址／数据总线处于第三态（高阻抗），在读周期的情况下是这样的。而在写周期情况下，就将要写的数据置于地址／数据线的后16位上。这时地址／数据线是有效的。在T_3状态期间，在写总线周期时，在地址／数据线上保持写的数据。在读总线周期时，将被读的数据从低16位的地址／数据线上取来。在T_4状态总线周期就结束了。

大多数系统存贮器和外部设备，都要求在总线周期有一个稳定的地址信号。为了实现这一点8086CPU在每一个总线周期的T_1状态通过ALE管脚发出一个允许地址锁存的控制信号ALE（address Latch enable）。这个信号在最小模式下可以直接由微处理机给出。而在最大模式下这个信号可以间接地由8288总线控制器给出。在ALE的后沿地址信号有效时，利用这个信号将地址保留起来。而在T_2状态以后8086CPU发出的地址信号消失了的情况下，依然保留了地址信号。这样就将地址信号从地址／数据线上分离出来了。我们就可以将它做为系统的一个独立的地址信号线来使用，这时我们称这个线为地址总线。为了锁存地址，做为8086的系列产品，INTEL公司提供了一个非反相的8282和一个反相的8283地址锁存器，用

来进行地址锁存。这些电路除了完成地址锁存外，还增加了电流驱动的能力和减小了电容负载，见图1—9。

由于读写周期之间的时序不同，以及外部设备和存贮器对读的响应时间也不相同，所以数据总线要被多重复用。然而，多路复用的数据总线既可以被缓冲使用也可以被直接使用。当存贮器和I／O设备直接联在没有被缓冲的总线上时，在读周期中其根本的问题是被读的器件要防止在T_1状态时出现在总线上的地址信号被丢失。为了确保地址信号不被丢失，通过受CPU的读信号控制的允许输出功能端（不是器件的片选功能端）来控制器件的输出驱动器允许器件输出。INTEL公司的很多外部设备、ROM/EPROM和RAM电路提供了一个允许输出功能端允许这些器件与一个没有被缓冲的多路复用的地址／数据总线相联接。我们应当考虑采用缓冲的数据总线，使用缓冲的数据总线能够简化联接上的要求，同时既增加了电流驱动能力又减少了电容负载，图1—9绘出了采用INTEL公司的8286（非反相）和8287（反相的）8位总线收发器所组成的缓冲的数据总线。8286或8287总线收发器在8086CPU给出的控制信号$\overline{DEN}$（数据允许）和DT／$\overline{R}$（数据发送／接收）的控制下，控制数据在总线上的传送方向。为了保证能将在T_1状态期间被多路复用的地址／数据信号线上的地址信号被分离出来，由这些信号提供了适当的时间关系。

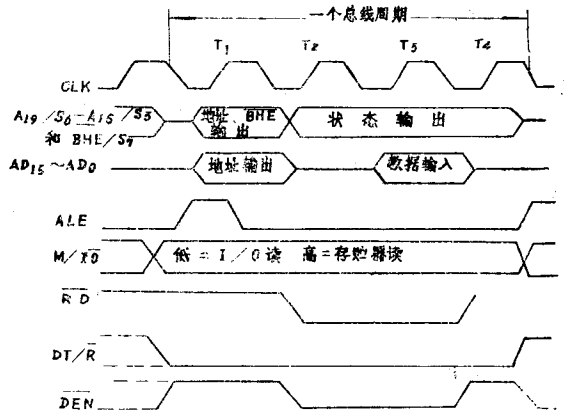


图1—7　8086读总线周期

我们把从地址锁存器到存贮器或I／O设备之间的地址线称为系统地址总线或简称为地址总线，而把从数据收发器或存贮器或I／O设备之间的数据线称为系统数据总线，简称为数据总线。

时钟电路

为了建立总线周期时间，8086CPU要求有一个外部时钟信号。为了这个目的，做为8086系列中的一个组成部分，INTEL公司提供了8284时钟产生／驱动器。除了用其为系统提供一个基本的时钟信号以外，还用其做为硬件复位和在总线周期中加入等待状态Tw。

8284时钟产生／驱动器要求有一个外部串接的振荡晶体，或者外部频率源。这个外部频率要为系统所要求频率的三倍，当8086CPU工作频率为5MHZ时，所要求的外部频率源的基本频率就要为15MHZ。这个频率经8284做三分频以后输出，这个输出的频率称为时钟。将这个时钟直接加在8086CPU的CLK输入端。8284还提供了一个被称为PCLK的另一个外部设备时钟。PCLK时钟的频率是CLK时钟频率的一半，由于8284使用晶体做为输入频率，所以电路中还有一个带缓冲的TTL电平的振荡器OSC（oscillator）输出。以上这些输出都可被外部系统设备所使用。

8284的硬件复位功能是在$\overline{RES}$输入端所加的信号的作用下通过内部一个施密特电路来完成的。当$\overline{RES}$输入端信号变低（也就是将$\overline{RES}$接地）时，RESET端的输出信号由CLK信号

同步启动这个信号要在外加的复位信号大于四个时钟周期才能被启动。

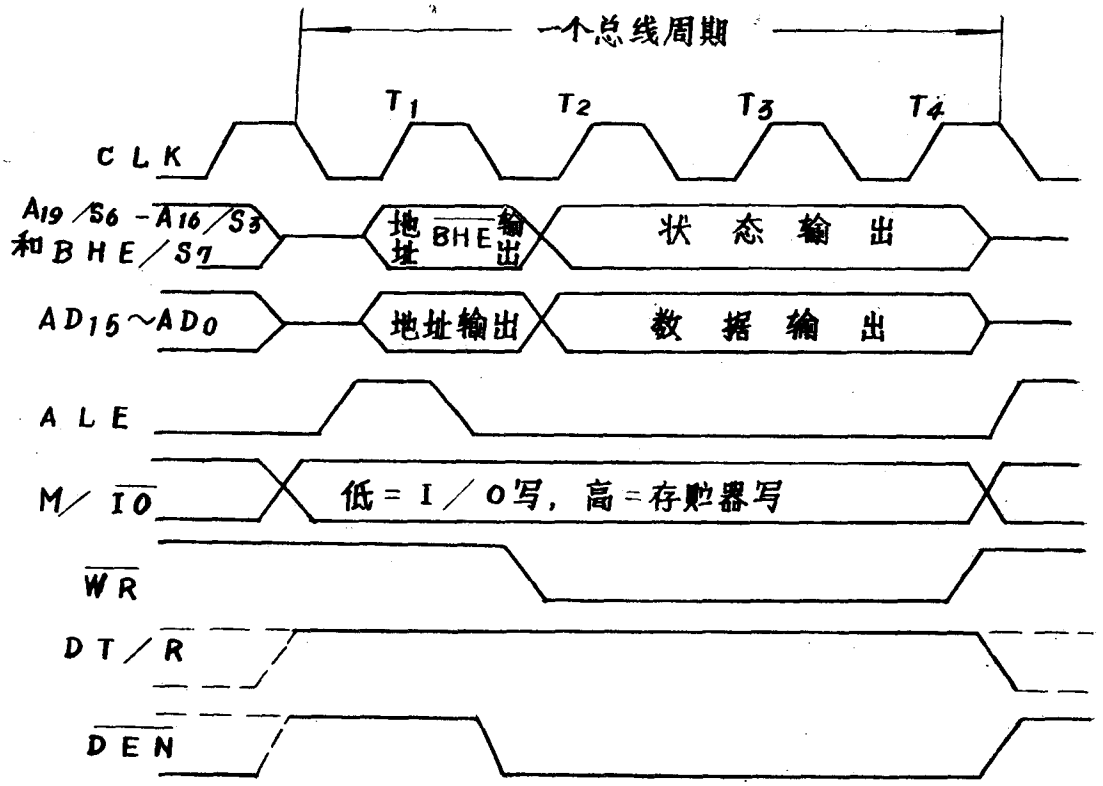


图1—8　8086写总线周期

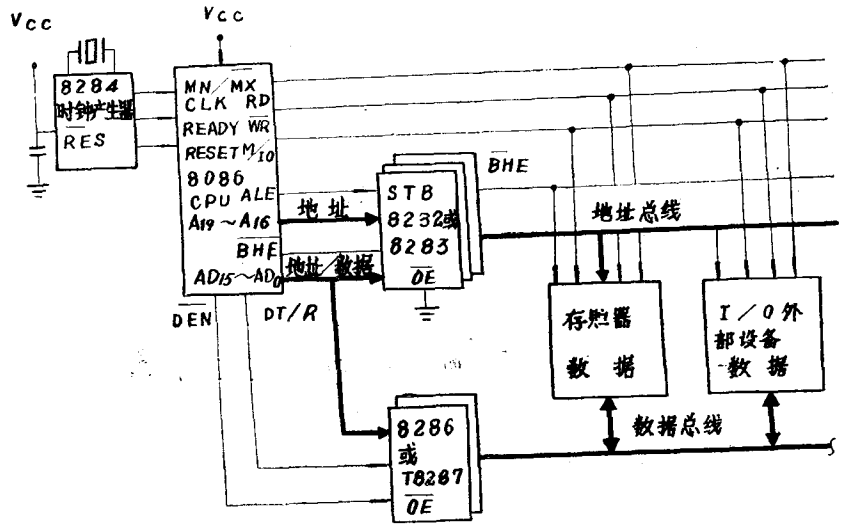


图1-9　8086最小模式缓冲数据总线

复位信号使8086CPU到内存的FFFF0H地址中取出一条指令并且执行这条指令。在$\overline{RES}$

端还外接了一个RC电路，当电源接通时自动产生一个复位信号（大于5ms）。8284的RESET信号的输出端是直接接在8086CPU的RESET信号的输入端上，此信号也可用做系统设备的系统复位信号。

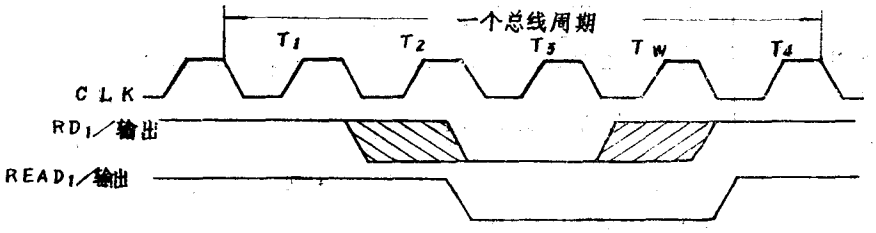


图1－10　等待状态时序

8086CPU总线周期等待状态的插入是通过8284的RDY_1或RDY_2（系统准备好信号）引入一个信号来实现的。当然RDY_1和RDY_2又能对应地受到$\overline{AEN}_1$和$\overline{AEN}_2$地址允许信号的制约。8284的READY输出信号是由CLK信号同步的。并且将其直接接在8086CPU的READY的输入端上。请见图1—10当被寻址的器件需要在一个总线周期中插入一个或多个等待状态时，在T_2状态结束之前RDY应变为低电平，由此使READY输出端在T_2状态结束时变为低电平，这个信号送到8086CPU的READY输入端，就使得在READY为低电平时期内的时钟状态成为等待状态Tw。这样就使Tw状态插入了。在退出等待状态时，RDY变为高电平，在Tw状态结束时使READY输出端变高，此高电平信号就通知CPU退出等待状态并允许总线周期进入T_4状态。以后我们还要对8284的功能做详细介绍。

1.2.4 通用寄存器

8086CPU的EU单元中有8个16位的通用寄存器。这8个寄存器分成两组，每组四个寄存器。其中一组称为数据寄存器（DATA register）组，有时也被称为H&L组(H表示高，L表示低）。这组寄存器中的四个寄存器是AX（AH和AL）累加器、BX（BH和BL）基址寄存器、CX（CH、CL）计数寄存器和DX（DH和DL）数据寄存器。另一组寄存器被称为指指器寄存器和变址寄存器，有时也被称为P&L组。这组寄存器中的四个寄存器是：SP堆栈指示器、BP基址指示器、SI源变址器和DI目标变址器（见图1—11）。

每个数据寄存器单独又可以分成高字节（H）和低字节（L）寄存器，它们可以分别被寻址。也就是说它们既可以被当做一个完整的16位的寄存器来使用，也可以分别被做为两个8位的独立的寄存器来使用，然而在8086CPU中的其它存寄器只能被做为一个完整的16位寄存器来使用。数据寄存器还并不仅限于在做算术和逻辑操作中使用。以后在讲指令系统时我们会看到在一些指令中不必被指定就可以使用某个特定的寄存器。表1-1中列出了这些寄存器的缩写符，在写指令时是非常有用的。表1—1绘出了这些寄存器在指令中的使用情况。指示器和变址寄存器在大多数算术和逻辑操作中也被用来分担一些功能。