第二章 8086 微处理器

当今的高档微处理器是在 80386 的基础上发展起来的，而 80386 的基础是 8086。8086 和当今的高档微处理器不可比拟，但是它的指令流水线操作方式以及存储器的分段结构在微处理器发展史上是划时代的标志。

8086 是标准的 16 位 CPU，其内部和外部的数据总线都为 16 位。地址总线为 20 位，最大内存寻址空间为 2^{20}=l MB。其主频(即时钟频率)为 5 MHz，8086-1 的主频为 10 MHz。

2.1、8086 的编程结构

一、 8086 的功能结构与指令流水

1．8086 的指令流水

图 2．1 是 8086 的功能结构示意图。它由两大部分组成：执行部件(EU，Execute Unit)和总线接口部件(Interface Unit，BIU)。前者负责指令的译码和执行，后者负责所有涉及外部总线的操作。二者既独立工作，又相互配合。

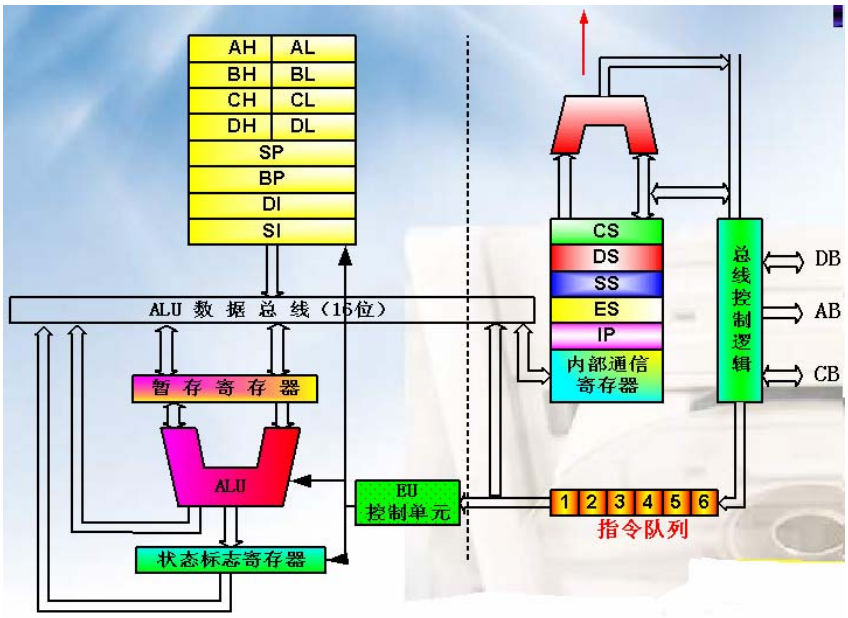

图 2．1

8086 以前的微处理器操作都是串行的，即对每一条指令而言，都是先取指令，再分析、执行。具体可表示为

取指令一[读操作数]一执行一[写操作数]一取(下一条)指令一……

方括号内的操作是操作数在内存中时所要进行的，当操作数是 CPIJ 内部的某个寄存器

时，不需要这一步。显然，在指令执行期间，总线(BUS)是空闲的，这一重要的系统资源没有得到充分利用。而 8086 在执行部件执行指令期间，总线接口部件可利用总线预取后续指令或进行存储器操作数的读或写。图 2．2 表示了 8086 的一段工作过程。可以看出，在图示的一段时间内，8086 执行了 3 条指令，并且预取了两条指令，存入内部的指令队列。图 2．2 揭示了一种现象——指令的并行操作，或者说指令的重叠执行，通常称之为指令流水线操作。

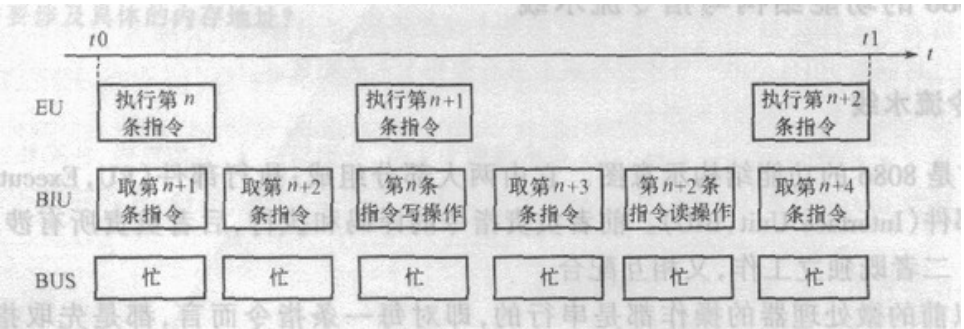

图 2.2　8086 的指令流水线操作

2．总线接口部件(BIU)

总线接口部件是 8086 同外部联系的接口。它负责所有涉及外部总线的操作，包括取指令、读操作数、写操作数、地址转换和总线控制等。

BIU 使用了指令预取队列(简称指令队列或队列)机构来实现指令流水线操作。该队列的容量为 6 个字节，即允许预取 6 个字节的指令代码。每当指令队列有两个或两个以上的字节空间，且执行部件未向 BIU 申请读／写存储器操作数时，BIU 顺序地预取后续指令的代码，并填入指令队列。

指令队列的工作方式是先进先出(First-in，First-out，FIFO)，或者说是管道方式，先预取的指令即排在前面的指令先被取走执行，后面的依次向前推进，腾出的空间再放新预取的指令。

当 EU 执行指令时发现是一条转移指令(又称分支转移指令或分支指令)，则 BIU 清除队列，从新的地址取指令，并立即送给 EU 去执行，然后，从后续的指令序列中取指令填满队列。可见，一遇到转移指令，就要重新预取指令队列中的下一条及下下一条指令。显然，这影响了机器的运行速度。8086，乃至 80386、80486，都存在这个缺陷，直到 Pentium 出现，引入了分支指令预测技术，才得以克服。

BIU 中有一个 16 位的专用加法器Σ，用来形成 20 位的物理地址。BIU 中还有一些专用的寄存器，其作用在本节的后面介绍。

3．执行部件(EU)

执行部件负责指令的译码和执行。它的组成和第一章介绍的运算器有许多相似之处，包括算术逻辑运算部件 ALU、通用寄存器、专用寄存器、标志寄存器等部件，这些部件的宽度都是 16 位。图 2．1 仅是一个组成结构示意图，不可能包括所有细节。对执行部件的理解主要在于它的总体功能以及各种寄存器的功能和应用。

二、 8086 的内部寄存器

从图 2．1 中可以看到，8086 内含 14 个 16 位寄存器。下面分五方面加以介绍。

1．通用寄存器

图 2．1 中的 AX、BX、CX 和 DX 是 4 个 16 位的通用寄存器。为了能处理字节数据，这 4 个寄存器都可看成由 2 个 8 位寄存器组成，它们可单独使用。例如，AX 可看成由高 8 位寄存器 AH 和低 8 位寄存器 AL 组成。在编程时，通用寄存器既可作为源操作数(其内容作为参与运算的数)，又可作为目的操作数(用来存放运算结果)。

2．指针和变址寄存器

图 2．1 中的 SP 和 BP 同属指针寄存器。SP 是堆栈栈指针寄存器，用来指示栈顶的位置。BP 为基址指针。图 2．1 中的 SI 和 DI 同属变址寄存器。SI 称为源操作数变址寄存器，DI 称为目的操作数变址寄存器。SP、BP、SI 以及 DI 寄存器的使用将在指令系统中详细介绍。

3．段寄存器

CS，DS，SS，ES 均为段寄存器，要了解段寄存器的使用，首先需要了解 8086 的内部存储器分段结构。

8086 的地址线为 20 位，最大寻址空间为 2^{20}=1 MB。8086 内部的寄存器都是 16 位，对地址的运算也是 16 位，而 16 位的最大寻址范围为 2^{16}=64 KB。为了能在 1 MB 空间内进行全范围寻址，即可访问到 1 MB 存储器的任何一个存储单元，把 l MB 存储器分成若干段(segment)，每一段最大为 64 KB，如图 2．3 所示。在分段时，要求段的起始单元的物理地址是 16 的整数倍，写成十六进制，最后一位应是 0，即 xxxx0H(x 为任一个十六进制数码，H 为十六进制后缀)。

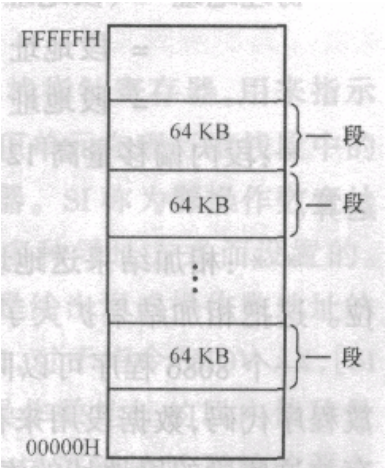

图 2.3　存储器分段

把段的起始单元的物理地址除以 16 的结果称为段地址，它为 16 位，写成十六进制是 4 位：XXXXH。显然，段地址决定了段在 lMB 空间中的位置。段内各存储单元相对段的起始单元都有一个距离，称为段内偏移量。

在对内存进行操作时，段地址先确定下来，然后给出不同的段内偏移量，就可以实现段内的寻址。段地址也是可以改变的，即段在 1MB 空间中的位置是可变的，因而可实现 1 MB 的全范围寻址。

由于采用了分段结构，因此可以把每一个存储单元看成是具有两种类型的地址：物理地址和逻辑地址。物理地址就是实际地址，它具有 20 位的地址值，它惟一地标识 1MB 存储空间的某一存储单元。CPU 与存储器之间的信息交换都是使用这个物理地址。逻辑地址是编程时所使用的地址，它由段地址和段内偏移量组成。逻辑地址和物理地址的关系为：物理地址=段地址×16+段内偏移量。由逻辑地址形成物理地址是由总线接口部件中的电路实现的，如图 2．4 所示。

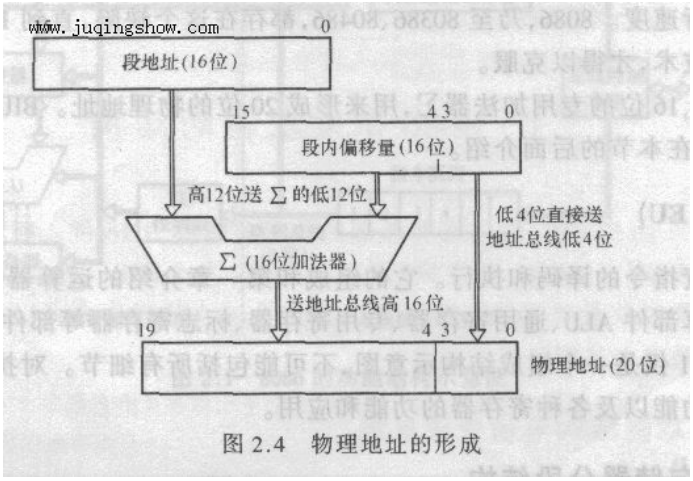

图 2.4　物理地址的形成

一个 8086 程序可以同时使用 4 个段：代码段、数据段、堆栈段和附加段。代码段用来存放程序代码，数据段用来存放程序中用到的数据，堆栈段作为堆栈，附加段也用来存放数据。存放这些段的段地址的寄存器分别是：

- 代码段寄存器：CS(Code Segment)；
- 数据段寄存器：DS(Data Segment)；
- 堆栈段寄存器：SS(Stack Segment)；
- 附加段寄存器：ES(Extra Segment)。

这 4 个段寄存器都是 16 位的。一个完整的程序在存储器中的存储分布如图 2．5 所示。

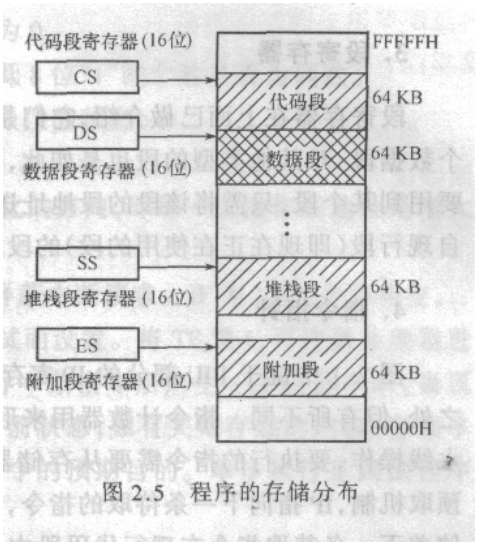

图 2.5　程序的存储分布

4．指令指针寄存器

图 2．1 中位于 BIU 部分的 IP 寄存器称为指令指针。8086 采用指令流水线操作方式，指令采用预取机制，IP 指向下一条待取的指令，即 IP 中存储了下一条待取指令的地址，确切地说，存储着下一条待取指令在现行代码段中的偏移量。取出的指令被送入指令队列。指令队列中的指令只有在到达队列的最前端，才能被送往 EU 执行。

5．标志寄存器

标志寄存器也是 16 位，但只使用了其中的 9 位，如图 2．6 所示。

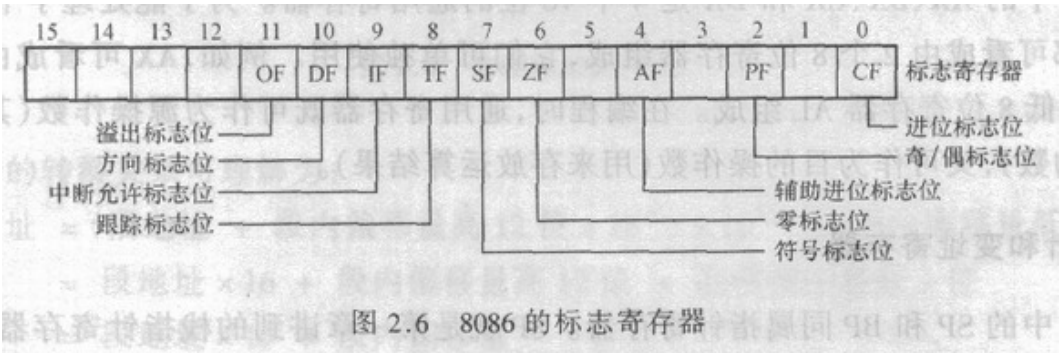

图 2.6　8086 的标志寄存器

这 9 个标志位可分为两类：一类反映指令执行的重要特征，为状态标志；另一类用于控制微处理器的操作，为控制标志。

1)状态标志位

①进位标志位(CF)：指出该指令是否在最高位产生了进位或借位。

②辅助进位标志位(AF)：对字节操作指令，反映该指令在执行时低 4 位是否产生了向高 4 位的进位或借位；对字操作指令，反映该指令在执行时低字节是否产生了向高字节的进位或借位。

③溢出标志位(OF)：反映该指令的运算结果是否超出范围。

④零标志位(ZF)：反映该指令的运算结果是否为 0。

⑤奇偶标志位(PF)：表示该指令的运算结果的低 8 位“1”的个数是否为偶数。

⑥符号标志位(SF)：反映该指令的运算结果符号位的状态。

2)控制标志位

①方向标志位(DF)：控制串操作指令的地址变化的方向，即每一次操作后地址是增大还是减小。

②中断允许标志位(IF)：控制是否允许响应可屏蔽中断请求。IF 为 1 时，允许响应。

③跟踪标志位(TF)：该标志位是为方便程序调试而设置。将 TF 置 1，可使微处理器单步执行方式。所谓“单步执行”，是指 CPU 每执行一条指令后产生一个内部中断，表现形式为程序暂停下来。此时，用户可以检查机器的当前状态，如有关寄存器的内容、存储单元的内容以及标志寄存器的内容等，检查是否达到指令的预期目的。单步执行是调试程序最有效的方法。

2.2、8086 的引脚功能

8086 有 40 个引脚，采用 DIP(Dual In—line Package，双列直插)封装，见图 2．13。

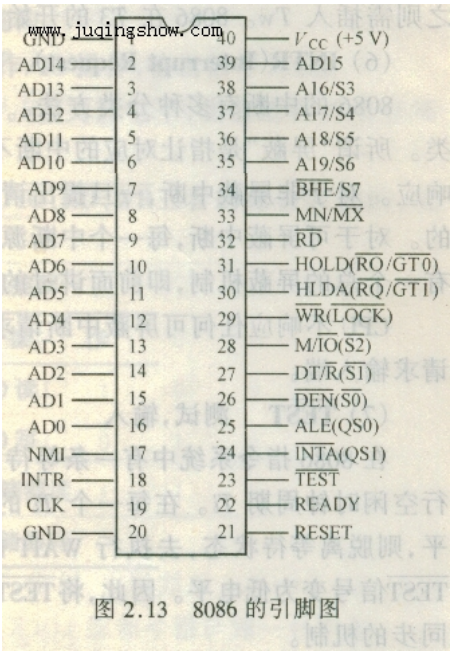

图 2.13　8086 的引脚图

8086 有两种组态模式：最小模式和最大模式。在最小模式时，8086 本身产生系统所需的全部控制信号，在构成微机系统时只能构成单处理器系统。在最大模式时，部分控制信号需借助于其他芯片(总线控制器 8288)，在构成微机系统时可构成多处理器系统(一个微机系统中可以有多片 8086 微处理器)。8086 的第 24～31 脚在两种模式下的引脚功能定义不一样。

1．两种模式功用相同的引脚

(1) ADl5～ADO(Address／Data) 地址／数据复用，双向，三态

先输出地址的低 16 位，然后作为数据线使用。这里要对 A0 作一些说明。A0 是 20 位地址的最低位，在 ALE 下降沿时被锁存到地址锁存器中。它除了作为地址的一位，参与地址译码外，还作为低 8 位数据的传送允许信号，即对于连接到低 8 位数据总线的存储器或 I／O 端口来说，在进行数据传送时须用 A0 选通。

(2) A19／$6 一 A：16／S3(Address／Status) 地址／状态复用，输出，三态

S6 指示 8086 当前是否使用总线。S5 指示中断允许标志位的状态。S4、S3 指明现在正在使用的段寄存器，S4、S3 的每一种组合对应一个段寄存器，组合“0，0”、“0，1”、“1，0”、“l，l”分别对应 ES、SS、CS、DS。

(3)$\overline{BHE}$／S7(Bus High Enable／s7) 控制／状态复用，输出，三态；

作为高 8 位数据传送允许信号$\overline{BHE}$；和 A0 的组合作用见表 2．2。

表 2.2 $\overline{BHE}$和 A0 的组合作用

$\overline{BHE}$	A0	功　能
0	0	进行 16 位字操作
0	1	在高 8 位数据总线和奇地址之间进行字节传送
1	0	在低 8 位数据总线和偶地址之间进行字节传送
1	1	保留(或不用)

(4)$\overline{RD}$　(Read)　读控制，输出，三态

低电平有效。有效时，表示 CPU 正在从存储器或 I／O 端口读出数据，究竟是读谁的数据需借助于信号 M／$\overline{IO}$。而$\overline{RD}$常被用来使能对方的数据输出机构。

(5)READY　准备好，输入

高电平有效。当 8086 采样该信号时，若发现为高电平，表示不需插入等待时钟，反之则需插入。

(6)INTR(Interrupt Request)可屏蔽中断请求，输入

8086 的中断有多种分类方法。从屏蔽角度分，可以分为可屏蔽中断和非屏蔽中断两类。所谓“屏蔽”是指让对应的中断不起作用，即不能提出中断请求或虽有请求但 CPIJ 不予响应。对于非屏蔽中断，一旦提出请求，CPU 必须无条件予以响应，即这类中断是

不可屏蔽的。对于可屏蔽中断，每一个中断源(引起中断的设备或 事件)都有自己的屏蔽机制，另外还有一个总的屏蔽机制，即前面讲过的中断允许标志位 IF。当 IF 为 0 时，CPU 不响应任何可屏蔽中断请求。只有在 IF 为 l 时 CPU 才予以响应。INTR 是可屏蔽请求输入端。

(7)$\overline{TEST}$ 测试，输入

在 8086 指令系统中有一条等待指令 WAIT。执行该指令时，8086 处于空闲状态，重复执行空闲时钟周期 Ti。在每一个 Ti 的开始测试$\overline{TEST}$，若为高电平，则继续插入 Ti，若为低电平，则脱离等待状态，去执行 WAIT 的下一条指令。可以说，等待指令是等待外部提供的$\overline{TEST}$信号变为低电平。因此，将$\overline{TEST}$和 wAIT 指令结合起来使用，可提供处理器与外部硬件同步的机制。

(8)NMI(Non-Maskable Interrupt) 非屏蔽中断请求，输入

(9)RESET 复位，输入

高电平有效(保持 4 个时钟周期以上)，使 8086 复位，即立即结束现行操作，清空指令队列，初始化 8086 的内部寄存器。CS 被置成 FFFFH，DS、 ES、SS、IP 和标志寄存器被清 0。随着 RESET 变为低电平，8086 开始再启动过程，首先执行地址 FFFF0 H 中的指令。

(10)CLK(Clock) 时钟输入

该引脚是 8086 的时钟信号输入端。时钟信号一般由时钟发生器 8284 提供，8284 将外接的石英振荡器的频率 3 分频后输出。当外接石英振荡器的频率为 15 MHz 时，8284 输出信号的频率为 5 MHz，这是 8086 的标准时钟频率。

(11)VCC +5 V 电源输入引脚

(12)GND 接地输入引脚

(13)MN／$\overline{MX}$(Minimum／Maximum)组态选择，输入

该引脚接+5 V 时，8086 工作在最大模式，接地时，工作在最小模式。

2．两种组态模式下，功能不同的引脚

1)最小模式下有关引脚的功能

(1)$\overline{WR}$(Write) 写控制，输出信号，三态

低电平有效。有效时，表示 CPU 正在向存储器或 I／0 端口写入数据，究竟是对谁

进行写，需借助于 M / $\overline{IO}$ 。$\overline{WR}$常被用来打开对方的数据输入机构。

(2)M / $\overline{IO}$ (Memory / Input and Output) 存储器操作 / IO 操作，输出信号，三态

该信号指示当前进行的是存储器操作还是 I / 0 操作。若为高电平，当前进行的是存储器操作；若为低电平，当前进行的是 I / O 操作。 在最小模式时，存储器和 I / O 的读 / 写控制是通过 M / $\overline{IO}$ 、$\overline{RD}$ 、、$\overline{WR}$三者的配合实现的，见表 2．3。

表 2.3 存储器和 I/O 的读/写控制

M/$\overline{IO}$	$\overline{RD}$	$\overline{WR}$	操 作
0	0	1	I/O 读
0	1	0	I/O 写
1	0	1	存储器读
1	1	0	存储器写

(3)$\overline{INTA}$(Interrupt Acknowledge) 中断响应，输出

CPU 响应可屏蔽中断请求后，进入中断响应周期，在中断响应周期中发出中断响应信号$\overline{INTA}$。8086 的中断响应周期为两个，所以在中断响应过程中会发出两个$\overline{INTA}$。

(4)ALE(Address Latch Enable) 地址锁存允许，输出，三态

有效时，表示复用地址线正在传送地址信息，其后沿可用来进行地址锁存。

(5)DT / $\overline{R}$ (Data Transmit / Receive) 数据收发控制，输出，三态

该信号表示当前总线上数据的流向。为高电平，表示数据从 CPU 流出，即 cPu 向外输出数据；为低电平，表示数据流向 CPU，即 CPU 接收数据。通常用这个信号作数据收发器的方向控制。

(6) $\overline{DEN}$ (Data Enable)数据允许，输出，三态

该信号低电平有效。有效时，表示允许数据传送。通常用这个信号作数据收发器的允许控制。

(7)HOLD 总线持有请求(简称总线请求)，输入

该信号高电平有效。有效时，表示有总线控制设备(又叫总线请求设备或总线主控设备)向 CPU 申请占有总线。

(8)HLDA(HOLD Acknowledge) 总线持有响应，输出

该信号高电平有效。有效时，表示 CPU 已响应总线持有请求并将总线释放，此时，cPu 的地址总线、数据总线及三态控制总线将全面呈现高阻，使提请求的总线控制设备可以顺利接管总线。

2)最大模式下有关引脚的功能

(1)$\overline{S2}$、$\overline{S1}$、$\overline{S0}$(status)状态信号，输出，三态

8086 通过它们的编码让外部电路(总线控制器 8288)了解 8086 当前的工作状态，见表 2．4。

表 2.4 $\overline{S2}$、$\overline{S1}$、$\overline{S0}$的编码

$\overline{S2}$、	$\overline{S1}$、	$\overline{S0}$	8086 工作状态	$\overline{S2}$、	$\overline{S1}$、	$\overline{S0}$	8086 工作状态
0	0	0	中断响应周期	1	0	0	取指令周期
0	0	1	I/O 读周期	1	0	1	存储器读周期
0	1	0	I/O 写周期	1	1	0	存储器写周期
0	1	1	暂停	1	1	1	过渡状态

(2)$\overline{LOCK}$总线锁定，输出，三态

该信号一般与指令前缀 LOCK 配合使用。当 CPU 执行一条加有 LOCK 前缀的指令时，该引脚输出有效电平，用来封锁其他总线主控设备，不允许它们此时提出 总线请求，直到 CPU 将该指令执行完为止。另外，在中断响应周期中该信号也一度有效，目的也还是利用该信号来封锁总线主控设备的请求，以确保 CPU 从数据 总线上正确获得中断向量。

(3)QSl、QSO(Queue Status) 指令队列状态，输出

通过它们的编码让外部电路(如协处理器 8087)了解指令队列当前的操作状态(细节略)。

(4)$\overline{RQ0}$ / $\overline{GT0}$、$\overline{RQ1}$ / $\overline{GT1}$(Request / Grant) 请求 / 同意，双向，三态

这两个引脚的作用和最小模式下的 HOLD、HLDA 相类似，也用于总线主控设备请求占用总线和获得 CPU 的允许，但机制有所不同(细节略)。

2.3、8086 最小/大模式下系统总线的形成

一、最小模式下系统总线的形成

当 8086 工作在最小模式时，所有的控制信号都由本身发出，没有特别的需要，控制信号不需要外加驱动电路，系统形成在于地址锁存和数据的双向驱动。图 2．14 是最小模式下的总线形成示意图。现说明如下：

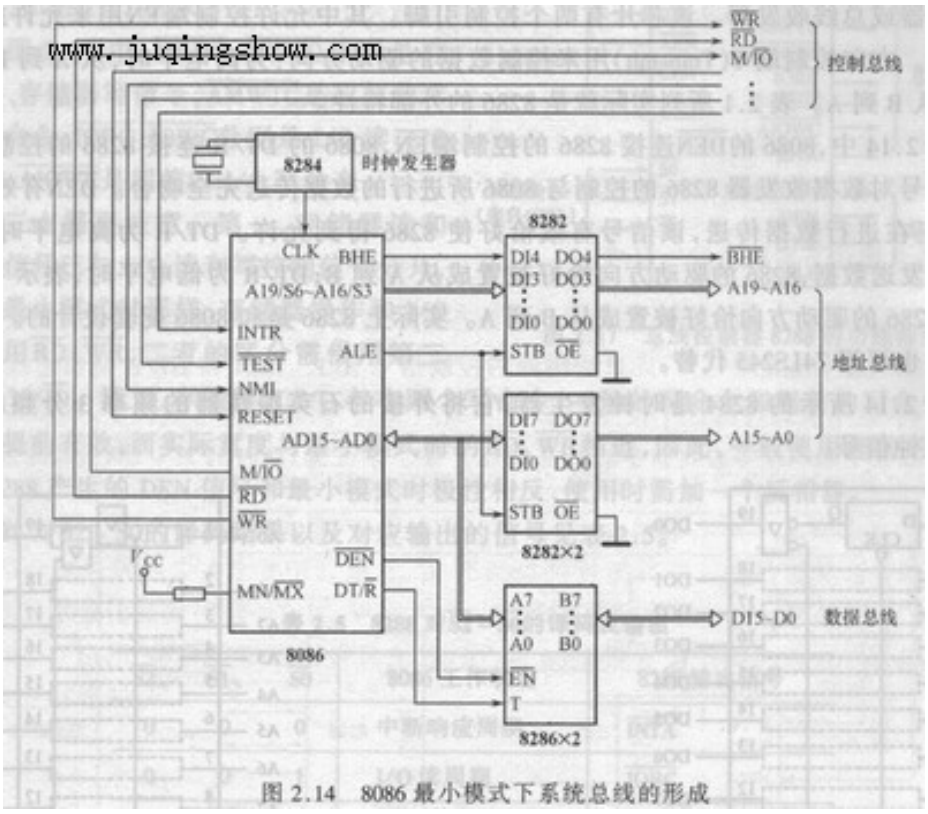

图 2.14　8086 最小模式下系统总线的形成

(1) 采用 3 个锁存器 8282 来锁存地址，并驱动 20 位的地址总线

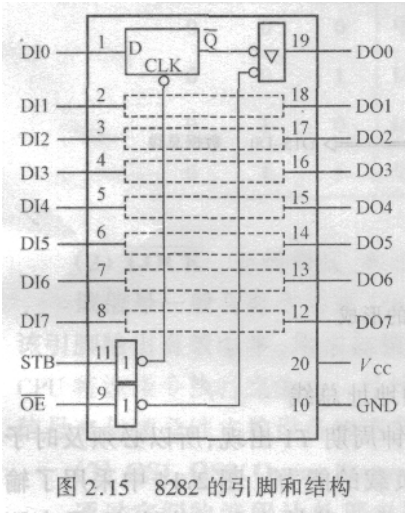

图 2.15　8282 的引脚和结构

由于复用线上的地址信号 A19～A16、A15 一 A0 只在时钟周期 T1 出现，所以必须及时予以锁存；此外还需要对 A19～AO 加以驱动，以增强它们带负载的能力。图 2．14 中采用了输出为三态的锁存器 8282，该芯片的引脚和结构见图 2．15，它有 8 位输入和 8 位输出；另有两个控制端：选通控制端 STB 和输出允许控制端$\overline{OE}$，前者用来控制锁存，后者控制数据的输出。 图 2．14 中，8282 的 STB 接 8086 的 ALE，ALE 为高电平时数据(是地址)直通，由高电平变低电平时(即后沿)数据被锁存；而$\overline{OE}$接地，处于常有效的状态，即该锁存器始终允许地址输出。 8282 也可用 74LS373 代替。

(2)采用 2 片 8286 来驱动数据总线

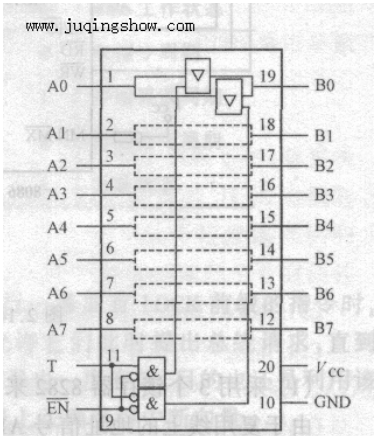

图 2.16　8286 的引脚和结构

如图 2.14 所示，数据总线采用双向驱动器 8286 来驱动。8286 的引脚和结构如图 2.16 所示，它可以朝两个方向驱动 8 位数据，适应了 CPU 发送和接收数据的需要，所以也称它为数据收发器或总线收发器。该芯片有两个控制引脚。其中允许控制端$\overline{EN}$用来允许或禁止芯片工作。方向控制端 T(Transmit)用来控制数据的驱动方向：为高电平时，从 A 到 B；为低电平时，从 B 到 A。表 2．1 所列实际就是 8286 的外部特性。在图 2．14 中，8086 的 $\overline{DEN}$连接 8286 的控制端$\overline{EN}$，8086 的 DT／$\overline{R}$连接 8286 的控制端 T。这两个信号对数据收发器 8286 的控制与 8086 所进行的数据传送完全吻合。 $\overline{DEN}$有效时，表示 8086 正在进行数据传送，该信号有效恰好使 8286 得到允许。DT／$\overline{R}$为高电平时，表示 CPU 向外发送数据，8286 的驱动方向恰好被置成从 A 到 B；DT／$\overline{R}$为低电平时，表示 CPU 接收数据，8286 的驱动方向恰好被置成从 B 到 A。实际上 8286 是和 8086 配套设计的。8286 也可用 74LS245 代替。

如图 2．14 所示的 8284 是时钟发生器，它将外接的石英振荡器的频率 3 分频后作为 8086 的时钟信号。

二、最大模式下系统总线的形成

8086 在最大模式时，一些控制信号需借助于总线控制器 8288 产生。

1．总线控制器 8288

总线控制器 8288 接收 8086 在最大模式时产生的工作状态编码信号$\overline{S2}$、$\overline{S1}$、$\overline{S0}$，识别 8086 将要执行的总线周期类型，产生相应总线周期所需的存储器访问、I / O 访问、中断响应等总线命令信号以及：DEN、DT / $\overline{R}$、ALE 等总线控制信号，如图 2．17 所示。

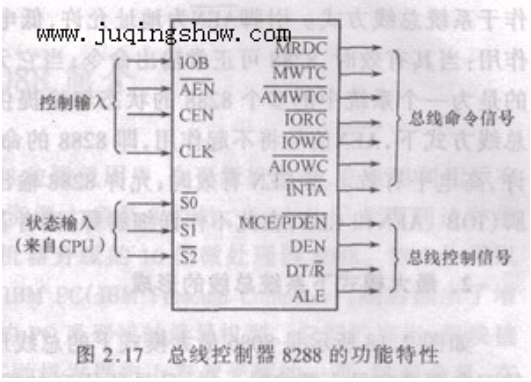

图 2.17　总线控制器 8288 的功能特性

如图 2．17 中，$\overline{MRDC}$、$\overline{MWTC}$分别是存储器读、存储器写命令，$\overline{AMWTC}$是超前的存储器写命令，$\overline{IORC}$、$\overline{IOWC}$分别是 I / O 读、I / O 写命令，$\overline{AIOWC}$是超前的 I / 0 写命令。 有三点需要注意。第一，存储器读和写控制信号已与 I / O 读和写控制信号分开，不再像最小模式时那样，存储器操作和 I / O 操作共用$\overline{RD}$、$\overline{WR}$，二者的区分需借用第三个信号 M / $\overline{IO}$。第二，存储器与 I / O 各有两个写命令：一般的写命令和超前的写命令。后者比前者提前有效，而实际宽度与最小模式时的$\overline{RD}$、$\overline{WR}$相近，因此，一般使用超前的写命令。第三，8288 产生的 DEN 信号和最小模式时极性相反，使用时需加一个反相器。 8288 对$\overline{S2}$～$\overline{S0}$的译码结果以及对应输出的信号见表 2．5。

表 2.5 8288 对$\overline{S2}$～$\overline{S0}$的译码及输出

$\overline{S2}$、	$\overline{S1}$、	$\overline{S0}$	8086 工作状态	8288 输出信号
0	0	0	中断响应周期	$\overline{INTA}$
0	0	1	I/O 读周期	$\overline{IORC}$
0	1	0	I/O 写周期	$\overline{IOWC}$, $\overline{AIOWC}$
0	1	1	暂　　停	—
1	0	0	取指令周期	$\overline{MRDC}$
1	0	1	存储器读周期	$\overline{MRDC}$
1	1	0	存储器写周期	$\overline{MWTC}$, $\overline{AMWTC}$
1	1	1	过渡状态	—

需要指出，8288 是一种有一定通用性的总线管理部件，它不仅可以为 CPU(如 8086)管理总线，还可以为 I / O 处理器(如 8089)管理与总线的接口。同时，它可以管理两种总线：系统总线和 I / 0 总线。所谓系统总线，是指总线为多个总线控制设备(或称总线主控器)所共享。所谓 I / 0 总线，是指总线为 某个处理器(CPU 或 I / O 处理器)所独自占有。8288 按哪种总线方式来管理由引脚 IOB 决定：IOB 为高电平时，8288 工作于 I / 0 总线方式；反 之，8288 工作于系统总线方式。引脚$\overline{AEN}$为地址允许，低电平有效。该信号仅在系统总线方式下发挥作用：当其有效时，8288 可正常输出命令；当它无效或刚开始有效的一段时间(约 115 ns，目的是为一个系统中的多个 8288 的状态切换提供时间)，8288 的命令线呈现高阻抗。在 I / O 总线方式下，$\overline{AEN}$信号将不起作用，即 8288 的命令输出总是被允许的。引脚 CEN 为命令允 许，高电平有效。当 CEN 有效时，允许 8288 输出有效的命令信号和控制信号。对这三个引脚(IOB、$\overline{AEN}$和 CEN)在此不作详细解释，读者若有需要，请查阅有关资料

2．最大模式下系统总线的形成

如图 2．18 所示是 8086 最大模式下的总线形成示意图。地址的锁存、驱动以及数据总线的双向驱动和最小模式时一样，只是所用到的控制信号 ALE、DEN 和 DT / $\overline{R}$ 是由总线控制器 8288 产生的(最小模式时这三个信号由 8086 本身产生)。在最大模式系统中，存储器读 / 写、I / O 读 / 写以及中断响应等总线命令信号由 8288 产生。请注意，此时的 MN / $\overline{MX}$ 引脚应接地。

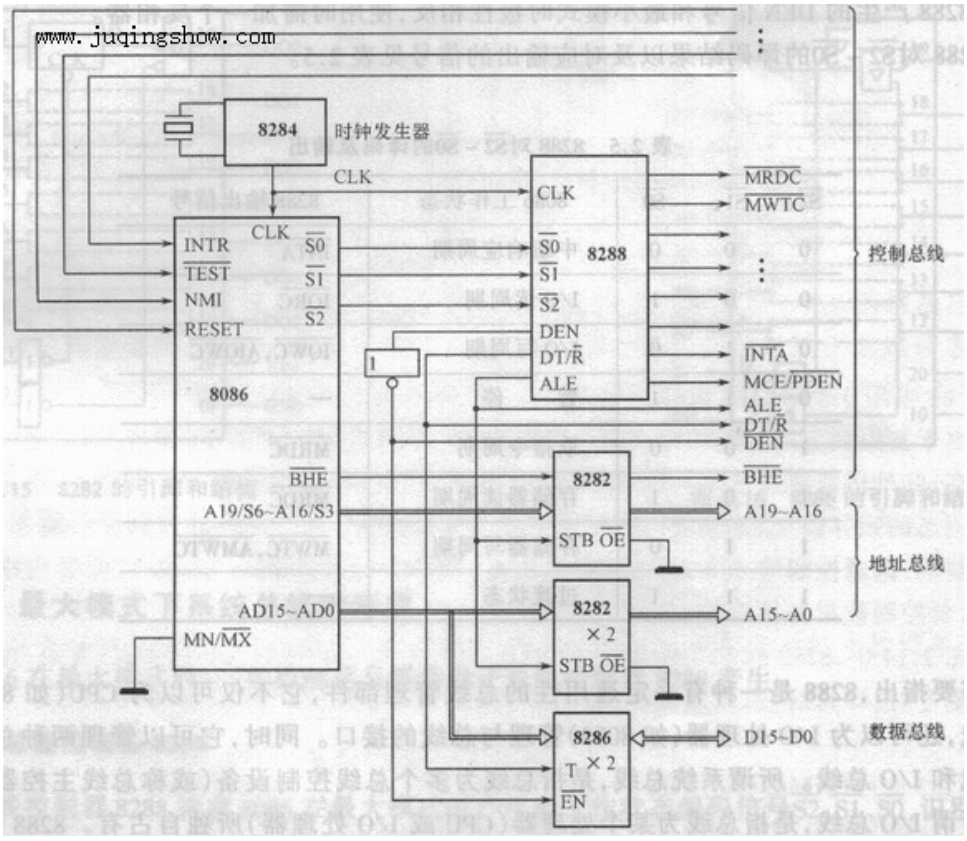

图 2.18　8086 最大模式下系统总线的形成