第 2 章　8086 微处理器结构与功能

2.1　8086 微处理器的外部结构

2.1.1　8086 微处理器的外部结构概述

微处理器通过引脚与外部的逻辑部件连接，完成信息的交换。CPU 的这些引脚信号称为微处理器级的总线，它应该能够完成下列功能。

（1）与存储器之间交换信息（指令及数据）。

（2）与 I/O 设备之间交换信息。

（3）能输入和输出必要的信号。

8086 CPU 芯片采用 40 条引脚的双列直插式封装形式。这些总线按功能可以分为四种，即数据总线、地址总线、控制总线、电源线。

（1）数据总线，用来传送信息（指令或数据）的总线。

（2）地址总线，指示正在传送的信息的来源或目的地址的总线。

（3）控制总线，管理和控制总线上活动的总线。

（4）电源引脚，包括 VCC 和地线。电源 VCC 为 4.5V～5.5V。

其中，部分引脚采用分时复用技术。8086 CPU 的引脚排列图如图 2.1.1 所示。

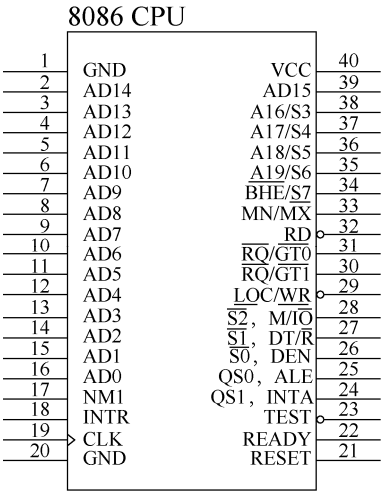

图 2.1.1　8086 CPU 引脚排列图

2.1.2 地址总线和数据总线

1．概述

数据总线用来传送 CPU 与内存储器或 I/O 设备的交换信息，该总线为双向三态总线。微处理器数据总线的条数决定了 CPU 和存储器或 I/O 设备一次能交换数据的位数，是区分微处理器是多少位的重要依据。

地址总线上的地址通常是 CPU 输出的，CPU 通过地址总线输出地址码来选择某一存储单元或某一称为 I/O 端口的寄存器，该总线为单向（输出）三态总线。地址码的位数决定了地址空间的大小。8088/8086 CPU 有 20 条地址线，访问存储器时用 20 条地址线，可寻址 1MB 内存单元；访问外部设备时用 16 条地址，可寻址 64K 个 I/O 端口

在 8086 CPU 中，数据/地址总线采用了分时复用引脚。

2．AD15～AD0 地址/数据总线

这 16 条线为分时复用的双重总线，在每个总线周期开始（*T*1）时用做地址总线的低 16 位（A15～A0），给出内存和 I/O 端口的地址。其余时间中数据总线用来传输数据。在传送数据时，用 BHE#信号区分字或字节传送。

8088 CPU 数据总线共 8 位，即 AD7～AD0，一次传送一个字节，但其内部结构是 16 位的，与 8086 CPU 相同。

3．A19～A16/S6～S3 地址/状态总线

这 4 条线也是分时复用的双重总线，在每个总线周期开始（*T*1）时用做地址总线的高 4 位（A19～A16），在存储器操作时为有效的高 4 位地址，在 I/O 操作中，这 4 条线为低电平。在总线周期的其余时间，这 4 条线指示 CPU 的状态信息，其中 S6 恒为低电平，S5 反映了标志寄存器中可屏蔽中断允许标志，而 S4、S3 表示正在使用哪个段寄存器，其编码如表 2.1.1 所示。

表 2.1.1　S4、S3 编码表

S4	S3	正在使用的段寄存器
0	0	附加段寄存器 ES
0	1	堆栈段寄存器 SS
1	0	访问存储器时用代码段寄存器 CS 或访问 I/O，不用段寄存器 或读中断向量，不用段寄存器
1	1	数据段寄存器 DS

2.1.3 控制总线

控制总线管理总线上的活动，用来传送从 CPU 发出的控制信息或外设送到 CPU 的状态信息，大部分是单向的，有一些是双向的。控制总线中的输出信号线，用来传输 CPU 送到其他部件的控制命令（如读写命令、中断响应等）；输入信号线，由外部向 CPU 输入控制及请求信号（复位、中断请求等）。

8086 CPU 的控制总线中有一条 MN/MX#线（引脚 33），称为最小/最大方式控制线，

用来控制 8086 CPU 的工作方式。当 MN/MX#接+5V 时，8086 CPU 工作于最小方式，由 8086 CPU 提供全部控制信号，用来构成一个小型的单处理器系统。当 MN/MX#接地时，8086 CPU 工作于最大方式，系统总线的部分控制信号由专门的总线控制器 8288 提供，8086 CPU 指示当前操作的状态信号（S2#、S1#、S0#）并送入 8288，8288 据此产生相应的系统控制信号。最大方式用于多处理器和协处理器结构。

在 8086 的控制总线中，有一部分控制线的功能与工作方式无关，而另一部分控制线的功能随工作方式不同而不同，现分述如下。下面主要介绍最大工作方式的引脚功能。

1．受 MN/MX#影响的控制总线

1）S2#、S1#、S0#—— 总线周期状态信号（三态、输出）

它们表示 CPU 外部总线周期的操作类型，由 CPU 送到系统中的总线控制器 8288，8288 根据这三个状态信号，产生存储器或 I/O 端口的读/写命令和中断响应信号，S2#、S1#、S0#的状态译码如表 2.1.2 所示。

表 2.1.2　S2#、S1#、S0#状态译码表

S2#	S1#	S0#	操作类型
0	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	0	0	取指
1	0	1	读存储器（数据）
1	1	0	写存储器
1	1	1	无效（无总线周期）

在总线周期的 *T*4 时钟周期期间，S2#、S1#和 S0#的任何变化，指示一个总线周期的开始，而在 *T*3 期间（或 *T*w，即等待周期期间）返回无效状态，表示一个总线周期的结束。在 DMA（直接存储器存取）方式下，S2#、S1#、S0#处于高阻状态。

在最小方式中，8086 CPU 与 S2#、S1#、S0#三个引脚相对应的信号分别为 M/IO#（存储器 I/O 控制）、DT/R#（数据发送/接收）和 DEN#（数据允许）。

2）RQ#/GT1#、RQ#/GT0#—— 请求/允许总线访问控制信号（双向）

这两条信息线是为多处理机应用而设计的，用于对总线控制权进行请求和应答，其特点是请求和允许功能是由一条线来实现的。

总线访问的请求/允许时序分为三个阶段：请求、允许和释放。首先，协处理器向 CPU 输入 RQ 信号请求使用总线；然后，在 CPU 的 *T*4 或下一个 *T*1 期间，CPU 输出一个宽度为一个时钟周期的脉冲 GT#给请求总线的协处理器，作为总线响应信号，从下一个时钟周期开始，CPU 释放总线。当协处理器使用总线结束时，再给出一个宽度为一个时钟周期的脉冲 RQ#给 CPU，表示总线请求结束，从下一时钟周期开始 CPU 又可以控制总线。

两条控制线可以同时接两个协处理器，RQ#/GT0#的优先级高。

在最小方式中，RQ#/GT1#、RQ#/GT0#引脚分别为总线请求信号 HOLD 和总线响应信

号 HLDA。

3）QS1、QS0——指令队列状态信号（输出）

它们提供 CPU 内部指令队列的状态，以便外部协处理器进行跟踪。

在最小方式中，QS1、QS0 引脚分别为 INTA#（中断响应）和 ALE（地址锁存允许）。

4）LOCK#——总线优先封锁信号（输出、三态）

该信号用来封锁外部处理器的总线请求，当 LOCK#输出低电平时，外部处理器不能控制总线，LOCK#信号是由指令在程序中设置的，若一条指令前加前缀 LOCK，则 CPU 执行这条指令时，LOCK#线输出低电平，并保持到指令执行结束，以防止在这条指令执行过程中被外部处理器的总线请求打断。

在保持响应期间，LOCK#线为高阻态。

在最小方式中，LOCK#引脚为 WR#（写控制）信号。

2．不受 MN/MX#影响的控制总线

1）RD#——读控制信号（三态、输出）

RD#有效时，表示 CPU 执行读操作，DMA 方式下，RD#处于高阻态。

2）READY——等待状态信号，准备就绪信号（输入）

当被访问的部件无法在 CPU 规定的时间内完成数据传送时，应由该部件向 CPU 发出请求信号，使 READY 变为低电平，使 CPU 处于等待状态，插入一个或几个等待周期 *T*w；当被访问的部件可以完成数据传输时，它使 READY 变为高电平，CPU 继续运行。

3）INTR——中断请求信号（输入）

此引脚可屏蔽中断请求，电平触发信号，可由软件复位内部的中断允许位加以屏蔽，CPU 在每一指令周期的最后一个时钟周期 *T*内采样 INTR 信号。

4）NMI——非屏蔽中断请求信号（输入）

NMI 为上升沿触发信号，不能用软件加以屏蔽。

5）TEST#——等待测试控制信号（输入）

在 WAIT（等待）指令期间，CPU 每隔 5 个时钟周期对 TEST#进行采样，若 TEST#变为高电平, 则 CPU 重复执行 WAIT 指令, 并对 TEST#进行测试, 循环于等待状态。若 TEST#为低电平，则 CPU 脱离等待状态，继续执行程序中的指令。

6）RESET——复位信号（输入）

当 RESET 为高电平时, CPU 停止正在运行的操作, 把内部的标志寄存器（Program Status Word，PSW）、段寄存器（Segment Register，SR）、指令指针寄存器（Instruction Pointer，IP）及指令队列复位为初始状态。

7）CLK——时钟信号（输入）

该信号为处理器和总线控制器提供基本的定时脉冲，其占空比为 1/3 以提供最佳的内部定时。

2.2　8086 微处理器的内部功能结构

8086 CPU 在内部采用了并行流水线结构，可以提高 CPU 的利用率和处理速度。从功能上讲，8086 CPU 由两个独立的逻辑单元组成：一个是总线接口单元（Bus Interface Unit BIU），另一个为执行单元（Execution Unit，EU）。其功能框图如图 2.2.1 所示。

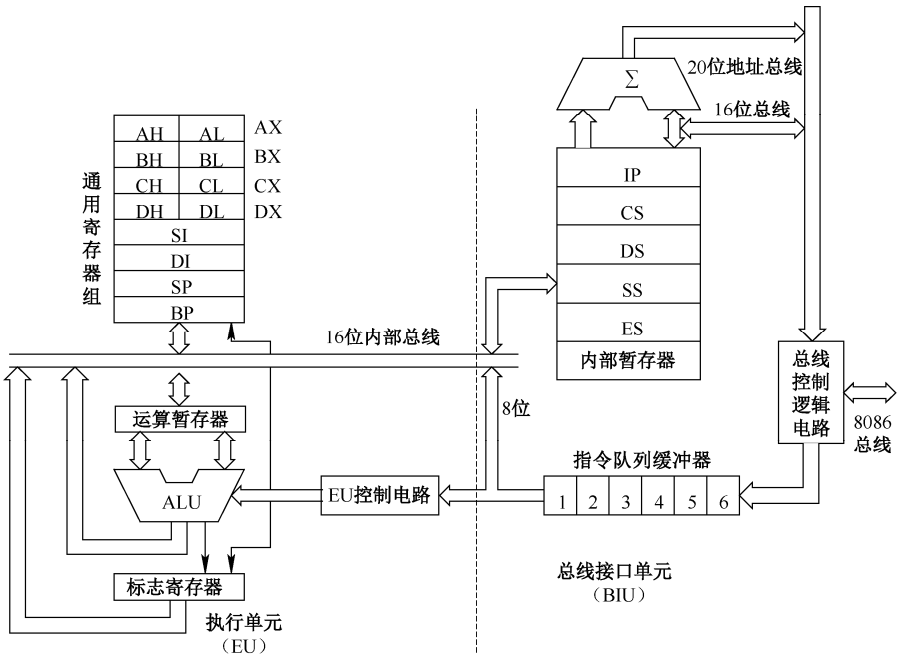

图 2.2.1　8086 CPU 的基本结构及功能框图

1．总线接口单元

总线接口单元负责 CPU 通过总线与存储器、I/O 设备之间的信息传送：取指令时，从存储器指定的地址取出程序指令送入指令队列供执行单元执行；执行指令时，根据执行单元命令对指定存储单元或 I/O 端口存取数据。

总线接口单元包括 4 个段寄存器（CS、DS、ES、SS）、指令寄存器（IP）、指令队列寄存器、物理地址加法器、总线控制逻辑等。

（1）4 个 16 位段地址寄存器，即代码段寄存器（CS），数据段寄存器（DS），附加段寄存器（ES）和堆栈段寄存器（SS），它们分别用于存放当前代码段、数据段、附加段和堆栈段的段基址。

（2）1 个 16 位的指令寄存器，即 IP。IP 用于存放下一条要执行指令的有效地址，即偏移地址（Effective Address，EA），IP 的内容由 BIU 自动修改，通常是进行加 l 修改。当执行转移指令、调用指令时，BIU 装入 IP 中的是转移目的地址。

（3）1 个 20 位物理地址加法器，将 16 位逻辑地址变换为存储器读/写所需要的 20 位物理地址。

（4）1 个 6 字节的指令队列缓冲器，预存 6 个字节的指令代码。

（5）总线控制逻辑，提供总线控制信号。

2．执行单元

执行单元主要负责从指令队列寄存器中获取指令，并对指令加以执行，完成指令所规定的操作。同时，它也负责算术/逻辑运算及进行内存有效地址的计算等。

执行单元由算术逻辑运算单元（Arithmetic Logic Unit，ALU）、暂存寄存器、状态标志寄存器（PSW）、通用寄存器组和执行控制单元等构成。执行单元包含以下几个部分。

（1）1 个 16 位算术逻辑运算单元，完成 8 位或 16 位的二进制运算。ALU 的核心是二进制加法器，其功能有二：一是进行 8 位或 16 位的二进制运算；二是按指令的寻址

方式计算出寻址单元的 16 位偏移地址，并将此偏移地址送到 BIU 中形成一个 20 位的实际地址。

（2）1 个 16 位的数据暂存器，协助 ALU 完成运算并可暂存参与运算的操作数。

（3）1 个标志寄存器，存放 ALU 运算结果特征或存放控制标志。

（4）4 个通用 16 位寄存器：AX、BX、CX、DX。

（5）4 个专用 16 位寄存器：SI、DI、SP、BP。

（6）1 个 EU 控制器，取指令控制和时序控制。

3．8086 CPU 的指令执行过程

CPU 的总线接口单元和执行单元相互独立，分别完成各自的操作，且按并行方式重叠操作，即可同时进行存取操作和执行指令的操作，实现了指令取指和执行的流水线操作技术，提高了 CPU 的利用率。8086 CPU 的指令执行过程如图 2.2.2 所示。

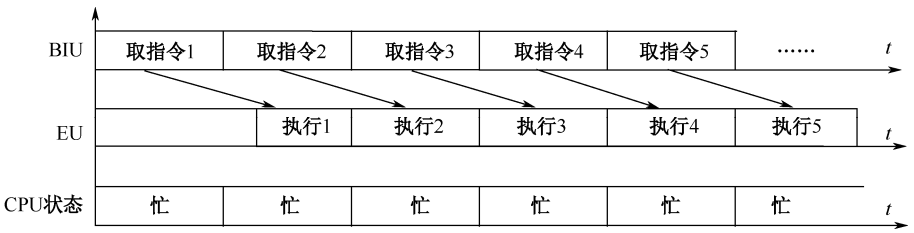

图 2.2.2　8088/8086 CPU 的指令执行过程

2.3　8086 微处理器的寄存器结构

8086 CPU 中可供编程使用的有 14 个 16 位寄存器，按其用途可分为 3 类 :通用寄存器、段寄存器和控制寄存器组。其中，通用寄存器分为数据寄存器与指针和变址寄存器两组。8086 微处理器的寄存器结构如图 2.3.1 所示。下面简要介绍这些寄存器的作用。

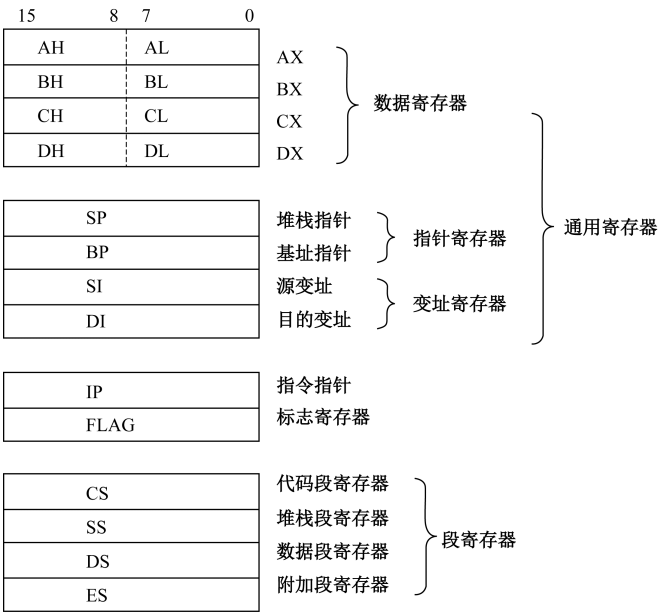

图 2.3.1　8086 CPU 的寄存器结构

1．通用寄存器

1）数据寄存器

数据寄存器包括 4 个 16 位的寄存器 AX、BX、CX 和 DX，一般用来存放 16 位数据，其中每一个又可根据需要将高 8 位和低 8 位分成独立的两个 8 位寄存器来使用，即 AH、BH、CH、DH 和 AL、BL、CL、DL 两组，用于存放 8 位数据，它们均可独立寻址、独立使用。这样，4 个 16 位寄存器可看做 8 个 8 位寄存器来使用。

每个寄存器又有它们的专用目的。

（1）AX——16 位累加器，使用频度最高，用于算术、逻辑运算及与外设传送信息等；AL 为 8 位累加器。

（2）BX——基址寄存器，常用做基址指针，指向一批连续存放操作数的基地址。

（3）CX——计数器，作为循环和串操作等指令中的隐含计数器。

（4）DX——数据寄存器，用来存放外设端口的 16 位地址，或双字长数据的高 16 位。

2）指针寄存器和变址寄存器

指针和变址寄存器包括指针寄存器 SP、BP 和变址寄存器 SI、DI，都是 16 位的寄存器且整体使用，一般用来存放逻辑地址的偏移量，即偏移地址。在任何情况下，都不能独立地形成访问内存的地址码。要形成 20 位的物理地址，需要由段寄存器提供段地址。它们之间的关系为

$$\text{物理地址}=\text{段地址}\times 16+\text{偏移地址}$$

（1）堆栈指针（SP）寄存器：用来指示栈顶的偏移地址。段地址由段寄存器 SS 提供。

（2）基址指针（BP）寄存器：在通过堆栈传递数据和地址时，基址指针寄存器用于存放要传递的数据和地址的偏移地址，也可以作为通用寄存器使用。

（3）源变址（SI）寄存器：用于变址操作。一般与 DS 联用，用来确定源操作数的存储单元地址。

（4）目的变址（DI）寄存器：用于变址操作。一般与 DS 联用，用来确定目的操作数的存储单元地址。

2．段寄存器

8086 CPU 的寻址空间有 1MB，物理地址需要 20 位的地址码。但 CPU 中的寄存器都是 16 位的。为解决该问题采用了存储器分段技术，利用段寄存器实现存储空间的分段结构。段寄存器提供段基址，即段寄存器用来存放这些内存段的首地址。段寄存器有 4 个：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）。

（1）代码段寄存器：存放当前被执行程序所在段的段基址。

（2）堆栈段寄存器：存放当前堆栈段的段基址。堆栈是内存中的一个特别存储区，主要用于在调用子程序和中断时，保留返回主程序的地址 CS:IP，以及保存进入子程序将要改变其值的寄存器的内容。

（3）数据段寄存器：存放当前使用的数据段的段基址。

（4）附加段寄存器：存放附加数据段的段基址。

其中，DS、ES 的初值都要由用户程序设置。

3．控制寄存器

8086 CPU 有两个控制寄存器：一是指令指针寄存器，二是处理器标志寄存器。

1）指令指针寄存器

8086 CPU 中设置了一个 16 位指令指针寄存器，用来存放将要取出的下一条指令在代码段中的偏移地址。在程序运行过程中，BIU 可修改 IP 中的内容，使它始终指向将要取出的下一条指令。指令在内存中的段地址由 CS 寄存器提供，下一条要取出的指令的内存地址由 IP 与 CS 联用构成，即 CS:IP 表示代码段中要处理的指令的逻辑地址，IP 和 CS 由系统执行，用户一般不能使用或修改。

2）处理器标志寄存器

8086 CPU 中设立了一个两字节的标志寄存器，标志由条件码标志和控制标志构成，其中，条件码标志反映了上次指令执行的结果状态信息，可用于条件转移指令的转移控制条件。8086 CPU 标志寄存器有 9 个有效标志位：6 个状态标志位，表示程序运行结果的状态信息，许多指令的执行都将自动地改变它，包括 CF、PF、AF、ZF、SF 和 OF；3 个控制标志位，可由用户根据需要用指令进行设置，用于控制处理器的具体工作方式，包括 IF、DF 和 TF。标志寄存器各标志位如图 2.3.2 所示。

D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
				OF	DF	IF	TF	SF	ZF		AF		PF		CF

图 2.3.2　8086 CPU 的标志寄存器

现将各标志位的定义说明如下。

（1）进位标志位（Carry Flag，CF）。如果做加法时最高位（字节操作是 D7 位，字操作是 D15 位）产生进位或做减法时最高位产生借位，则 CF=1，否则 CF=0。

（2）奇偶标志位（Parity Flag，PF）。如果操作结果的低 8 位中含有偶数个 1，则 PF=1，否则 PF=0。

（3）辅助进位标志位（Auxiliary Carry Flag，AF）。如果做加法时 D3 位有进位或做减法时 D3 位有借位，则 AF=1，否则 AF=0。

（4）零标志位（Zero Flag，ZF）。如果运算结果各位都为零，则 ZF=1，否则 ZF=0。

（5）符号标志位（Sign Flag，SF）。如果运算结果的最高位（字节操作是 D7，字操作是 D15）为 1，则 SF=1，否则 SF=0。

（6）溢出标志位（Overflow Flag，OF）。在加或减运算中结果超出 8 位或 16 位有符号数所能表示的数值范围时，产生溢出，OF=1，否则 OF=0。

（7）中断标志位（Interrupt Flag，IF）。当 IF=1 时，CPU 可响应可屏蔽中断请求；当 IF=0 时，CPU 不响应可屏蔽中断请求。可用指令设置：CLI 指令复位中断标志是 IF＝0；STI 指令置位中断标志是 IF＝1。

（8）单步标志位（Trap Flag，TF）。假如 TF=1，则 CPU 处于单步工作方式。在这种工作方式下，CPU 每执行完一条指令就会自动产生一次内部中断。在调试程序 DEBUG 中，T 命令就利用了这种中断。

（9）方向标志位（Direction Flag，DF）。在串处理指令中，若 DF=0，则表示串处理指令地址指针自动增量，即串操作由低地址向高地址进行；若 DF=1，则表示地址指针自动减

量，即串操作由高地址向低地址进行。DF 标志位可通过指令预置。

2.4 8086 微处理器的存储器组织

8086 CPU 地址总线有 20 条，可寻址 1MB，其物理地址为 00000H～FFFFFH，即微型计算机存储器寻址需 20 位地址码，但 CPU 内部寄存器和算术逻辑运算单元都是 16 位的，可寻址范围为 64KB。因此，如何扩大寻址范围成为一个必须解决的问题。

8088/8086 巧妙地采用了地址分段方法，将 1MB 的存储空间划分为若干个逻辑段，每段最多为 64KB 存储单元，段内任一个存储单元的地址可用相对于段起始地址的偏移量来表示，这个偏移量称为段内偏移地址（16 位）。同时，规定每个段的起始地址的低 4 位必须为 0，即逻辑段的起始地址的低 4 位二进制码必须是 0，这样段起始地址有效位只有 16 位，段的起始地址的高 16 位被称为该段的段地址，把它存放在相应的段寄存器中，而段内的相对地址可用系统中的 16 位通用寄存器来存放，被称为偏移地址。这种由段地址和偏移地址两部分共同表示的地址，称为逻辑地址，其格式为段地址:偏移地址。用户编程时采用逻辑地址。

在 8086 系统中，把整个存储空间分成许多逻辑段，每个逻辑段的容量≤64KB，允许它们在整个存储空间中浮动，各个逻辑段之间可以紧密相连，也可以相互重叠（完全重叠或部分重叠）。

一个存储单元可以拥有多个逻辑地址，但只可能拥有一个唯一的物理地址（00000H～FFFFFH），该 20 位地址在指令执行时由 CPU 内部总线接口单元的地址加法器形成，并进行硬件寻址。地址加法器的具体做法：段地址左移 4 位，然后加上偏移地址即可得到 20 位物理地址。物理地址形成示意图如图 2.4.1 所示。

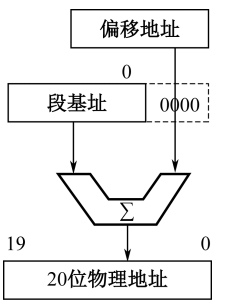

图 2.4.1　存储器物理地址的形成

在访问存储器时，段地址总是由段寄存器提供的。8086/8088 微处理器中有 4 个段寄存器（CS、DS、SS、ES），所以 CPU 可以通过这 4 个段寄存器来访问 4 个不同的段。用程序对段寄存器的内容进行修改，可访问所有的段。用户使用 8086/8088 汇编语言编写程序时，要把程序中的不同信息安排在不同的段，也就是说，用户源程序汇编后在存储器中存放时按照不同的信息放在不同的逻辑段中。而程序中的信息包括：程序（代码）信息、数据信息、堆栈信息。8086 CPU 存储器空间按信息特征分为程序区、数据区和堆栈区。

程序区：程序区是程序代码存放的区域，CS 存放正在运行程序的代码段首地址，CS:IP 指向代码段内要取出的指令。

数据区：数据区是数据存放的区域，DS 和 ES 分别存放正在运行程序的数据段和附加

数据段的首地址。

堆栈区：堆栈区存放正在运行的程序的堆栈段首地址，SS:SP 指向堆栈栈顶。

任一程序只要规定好 CS、DS、SS 和 ES 的值，便可分配到内存相应的地方，并开始运行。这种分段可由程序员或操作系统完成。

2.5 8086 微处理器的 I/O 组织

I/O 设备包括与外界通信和存储大容量信息用的各种外部设备。由于这些外部设备的复杂性和多样性，特别是速度比 CPU 低得多，因此 I/O 设备不能直接和总线相连接，必须通过 I/O 接口芯片进行相互联系。I/O 接口是保证信息和数据在 CPU 和 I/O 设备之间正常传送的电路。I/O 接口与 CPU 之间的通信是利用称为 I/O 端口的寄存器来完成的，一个 I/O 端口有一个唯一的 I/O 地址与之对应。

微机 I/O 端口的编址方式有两种：存储器映射方式和独立 I/O 编址方式。存储器映射方式就是 I/O 端口与存储器统一编址，它们共享一个地址空间。存储器映射方式不需要专门的 I/O 指令，I/O 数据存取与存储器数据存取一样灵活，I/O 端口要占用部分存储器地址空间。独立 I/O 编址方式就是 I/O 端口单独编址，I/O 地址空间独立于存储地址空间。独立 I/O 编址方式采用专门的 I/O 指令实现对 I/O 端口的操作。

8086 CPU 采用独立 I/O 编址方式。8086 CPU 利用 16 根地址线 A15～A0 来给 I/O 编址，所以能寻址 64KB 的 I/O 空间，其 I/O 地址为 0000H～FFFFH。I/O 端口的寻址不用分段，故而不用段寄存器，端口地址仍为 20 位，系统地址总线上的高 4 位总是 0。IBM-PC 系统只使用 A9～A0 等 10 根地址线为 I/O 端口编址。

习　题

2.1　CPU 在内部结构上由哪几部分组成？

2.2　CPU 的总线接口部件有哪些功能？CPU 的执行部件有什么功能？

2.3　8086/8088 微处理器中有哪些寄存器？通用寄存器中哪些可以作为地址指针使用？

2.4　简述 8086 CPU 标志寄存器各位的含义与作用。

2.5　8086 CPU 对存储器的管理为什么采用分段的办法？8086 CPU 的 20 位物理地址是怎样形成的？当（CS）=8000H,（IP）=0100H 时，其指向的物理地址等于多少？