

第2章 寄存器

检测点2.1

(1) 写出每条汇编指令执行后相关寄存器中的值。

```
mov ax,62627    AX=2627
mov ah,31H      AX=3127
mov al,23H       AX=3123
add ax,ax        AX=6246
mov bx,826CH     BX=826C
mov cx,ax        CX=826C
mov ax,bx        AX=826C
add ax,bx        AX=654C
mov al,bh        AX=6582
mov ah,bl        AX=6C82
add ah,ah        AX=4482
```

汇编语言(第3版)

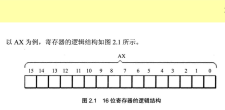
```
add al,6    AX=4488
add al,al    AX=4576
mov ax,cx    AX=6246
```

(2) 只能使用目前学过的汇编指令,最多使用4条指令,编程计算2的4次方。

```
mov ax,2
add ax,ax    2+2=4
add ax,ax    4+4=8
add ax,ax    8+8=16
```

2.4 物理地址 $2^8=16$
 $8+8=16$
 $2^8=16$

2.1 通用寄存器



AX、BX、CX、DX

- AX 可分为 AH 和 AL;
- BX 可分为 BH 和 BL;
- CX 可分为 CH 和 CL;
- DX 可分为 DH 和 DL。

注意,此时 al 是作为一个独立的 8 位寄存器来使用的,和 ah 没有关系,CPU 在执行这条指令时认为 ah 和 al 是两个不相关的寄存器。不要错误地认为,诸如 add al,93H 的指令产生的进位会存储在 ah 中,add al,93H 进行的是 8 位运算。

```
mov ax,bl    (在 8 位寄存器和 16 位寄存器之间传送数据)
mov bh,ax    (在 16 位寄存器和 8 位寄存器之间传送数据)
mov al,20000 (8 位寄存器最大可存放值为 255 的数据)
add al,100H  (将一个高于 8 位的数据加到一个 8 位寄存器中)
```

等都是错误的指令,错误的原因都是指令的两个操作对象的位数不一致。

2.2 字在寄存器中的存储

- 字节,记为 byte,一个字节由 8 个 bit 组成,可以存在 8 位寄存器中。
- 字,记为 word,一个字由两个字节组成,这两个字节分别称为这个字的高位字节和低位字节,如图 2.5 所示。

字: 0 1 0 0 1 1 1 0 0 0 1 0 0 0 0 0

高位字节 低位字节

图 2.5 一个字由两个字节组成

2.5 16 位结构的 CPU

概括地讲,16 位结构(16 位机、字长为 16 位等常见说法,与 16 位结构的含义相同)描述了一个 CPU 具有下面几方面的结构特性。

- 运算器一次最多可以处理 16 位的数据;
- 寄存器的最大宽度为 16 位;
- 寄存器和运算器之间的通路为 16 位。

2.6 8086CPU 给出物理地址的方法

8086CPU 有 20 位地址总线,可以传送 20 位地址,达到 1MB 寻址能力。8086CPU 又是 16 位结构,在内部一次性处理、传输、暂时存储的地址为 16 位。从 8086CPU 的内部结构来看,如果将地址从内部简单地发出,那么它只能送出 16 位的地址,表现出的寻址能力只有 64KB。

8086CPU 采用一种在内部用两个 16 位地址合成的方法来形成一个 20 位的物理地址。

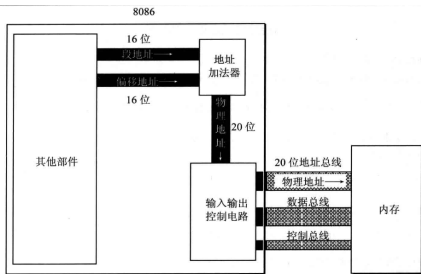


图 2.6 8086CPU 相关部件的逻辑结构

如图 2.6 所示,当 8086CPU 要读写内存时:

- CPU 中的相关部件提供两个 16 位的地址,一个称为段地址,另一个称为偏移地址;
- 段地址和偏移地址通过内部总线送入一个称为地址加法器的部件;
- 地址加法器将两个 16 位地址合成为一个 20 位的物理地址;
- 地址加法器通过内部总线将 20 位物理地址送入输入输出控制电路;
- 输入输出控制电路将 20 位物理地址送上地址总线;
- 20 位物理地址被地址总线传送到存储器。

地址加法器采用物理地址=段地址×16+偏移地址的方法用段地址和偏移地址合成物理地址。例如,8086CPU 要访问地址为 123C8H 的内存单元,此时,地址加法器的工作过程如图 2.7 所示(图中数据皆为十六进制表示)。

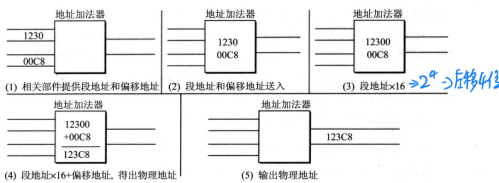


图 2.7 地址加法器的工作过程

2.8 段的 概念

物理地址	段地址	偏移地址
21F60H	2000H	1F60H
	2100H	0F60H
	21F0H	0060H
	21F6H	0000H
	1F00H	2F60H

2.9 段 寄 存 器

8086CPU 有 4 个段寄存器: CS、DS、SS、ES。

2.10 CS 和 IP

在 8086PC 机中,任意时刻,设 CS 中的内容为 M,IP 中的内容为 N,8086CPU 将从内存 M×16+N 单元开始,读取一条指令并执行。

2.11 修改 CS、IP 的指令

- 从 CS:IP 指向的内存单元读取指令,读取的指令进入指令缓冲器;
- IP=IP+所读取指令的长度,从而指向下一条指令;
- 执行指令。转到步骤(1),重复这个过程。

jmp ax, 指令执行前: ax=1000H, CS=2000H, IP=0003H
指令执行后: ax=1000H, CS=2000H, IP=1000H
jmp bx, 指令执行前: bx=0B16H, CS=2000H, IP=0003H
指令执行后: bx=0B16H, CS=2000H, IP=0B16H

“jmp 某一合法寄存器”指令的功能为:用寄存器中的值修改 IP。

jmp ax, 在含义上好似: mov IP,ax。

2.12 代 码 段

```
mov ax,0000    (8B 00 00)
add ax,0123H   (05 23 01)
mov bx,ax      (BB DB)
jmp bx         (FF E3)
```

这段长度为 10 个字节的指令,存放在 123B0H~123B9H 的一组内存单元中,我们就可以认为,123B0H~123B9H 这段内存是用来存放代码的,是一个代码段,它的段地址为 123BH,长度为 10 个字节。