IBM-PC (0520)

微型計算机系統

李汉章　王秋未　夏涛　编

2

《计算机技术》编辑部
中国计算机技术服务公司北京分公司

目　录

第六章　8086指令系统详解

（按其功能分类法）

第七章　汇编语言

第六章　8086指令系统详解

（按其功能分类法）

第一节　数据传送指令：

1. MOV——传送指令

用途：

1）从寄存器向寄存器传送数据。

2）从存贮单元向寄存器传送数据。

3）从寄存器向存贮单元传送数据。

MOV 指令传送的数据可以是 8 位的也可以是 16 位的。它一共有七种不同类型的传送指令。每种类型可以有多种使用方法。其编码依赖于被传送的数据的类型以及这个数据所在的单元。汇编语言将根据这两个因素的具体情况来生成正确的指令目标代码。

如果目的操作数为寄存器，则指令操作码中相应于 d 的那位就置“1”，否则置“0”。如果类型为字，则指令操作码中对应于 W 码那位应置“1”，否则为“0”。

例1：将 CX 的内容传送到 AX 寄存器中。

编码：

1	0	0	0	1	0	d	W
mod		reg			r/m		

设：CX 的内容为 2423；AX 的内容为 3569；

执行完 MOV　AX、CX 指令后。

AX 的内容为 2423；CX 的内容为 2423；

执行过程如图 6-1 所示。

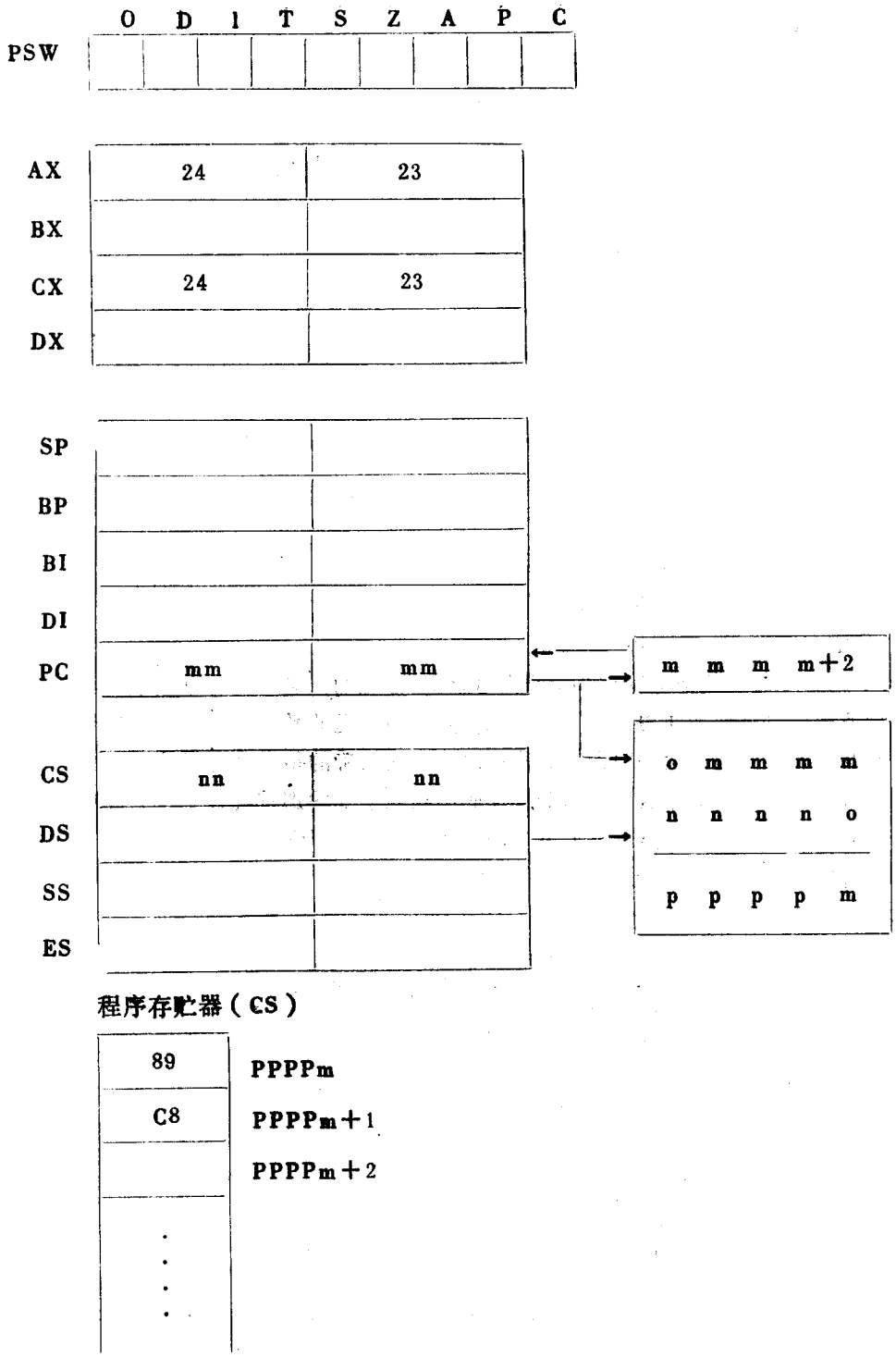


图6-1　例1的执行过程

注意：1.在这个指令中不可以指定段寄存器。

2.不影响标志位。

例2：把立即数装入到寄存器中。

立即数可以是 8 位或者 16 位的。通过立即寻址方式装入到寄存器中。

设：立即数为 1 2 3 4、CX 的内容为任意的。执行 MOV CX，1 2 3 4 H 指令之后。则 CX 的内容为 1 2 3 4。执行过程如图 6-2 所示。

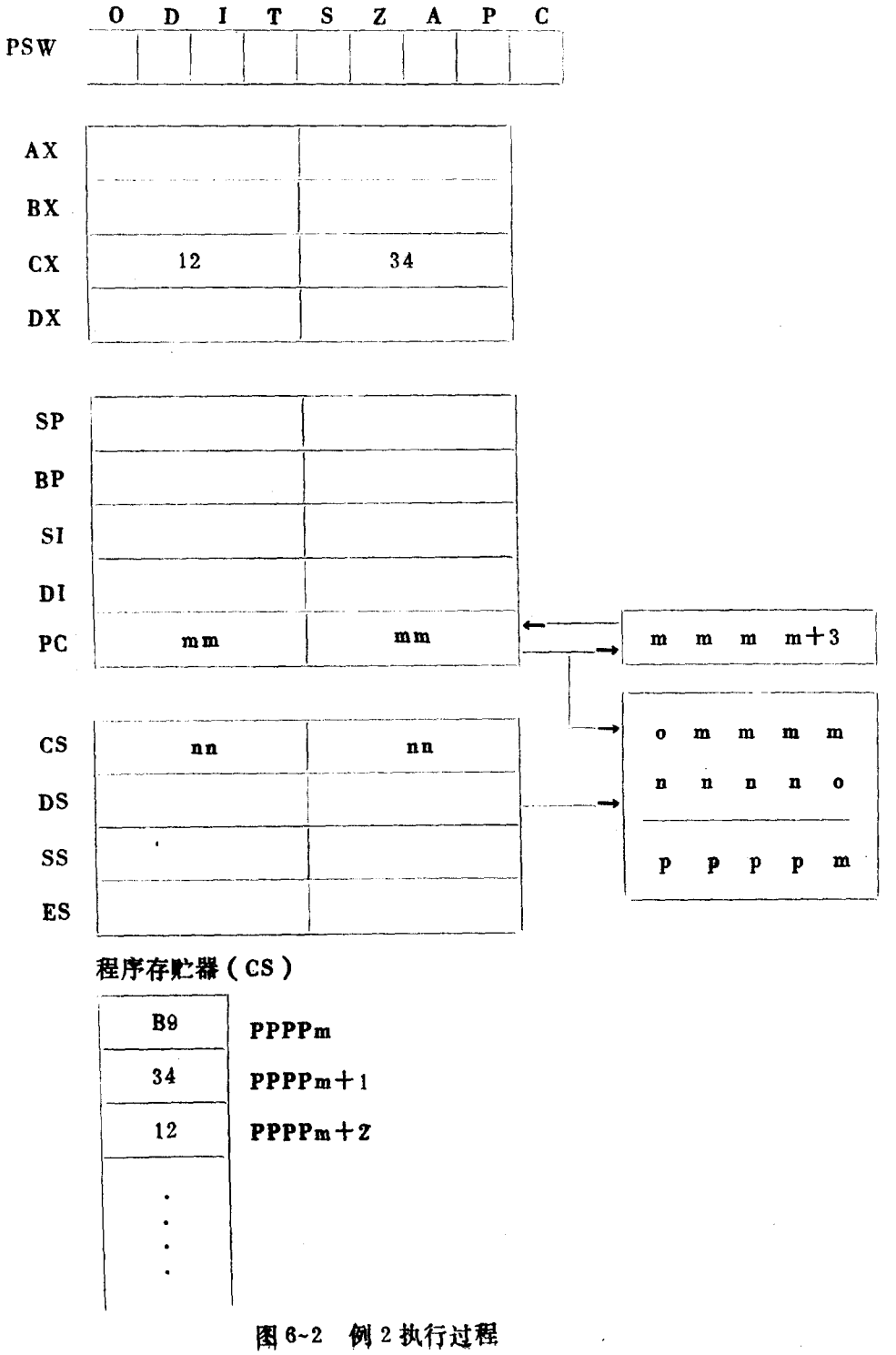
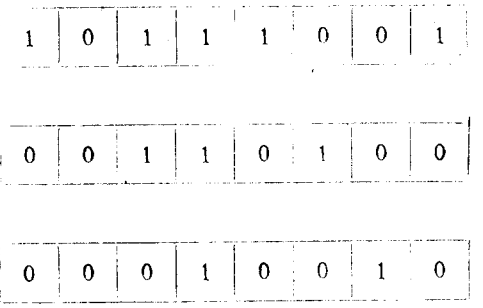


图 6-2　例 2 执行过程

注意：1.不影响标志位。

2.这条指令不能装入到段寄存器中。

3.编码：



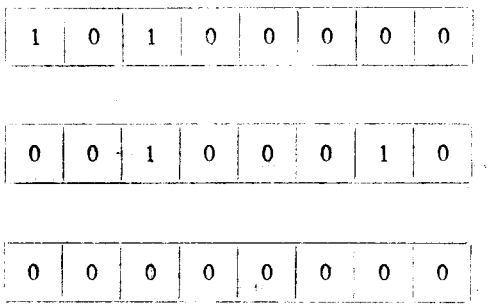
例3 把存贮器的内容装入到累加器中。

MOV AL,〔1050H〕

将存贮单元1050（相对于DS寄存器）中的内容传送到AL寄存器中。不影响标志位。

设1050单元的内容为22。

指令编码：



执行完MOV AL,〔105 OH〕指令后

数据段1050 H单元的内容送LA寄存器，执行过程如图6-3所示。

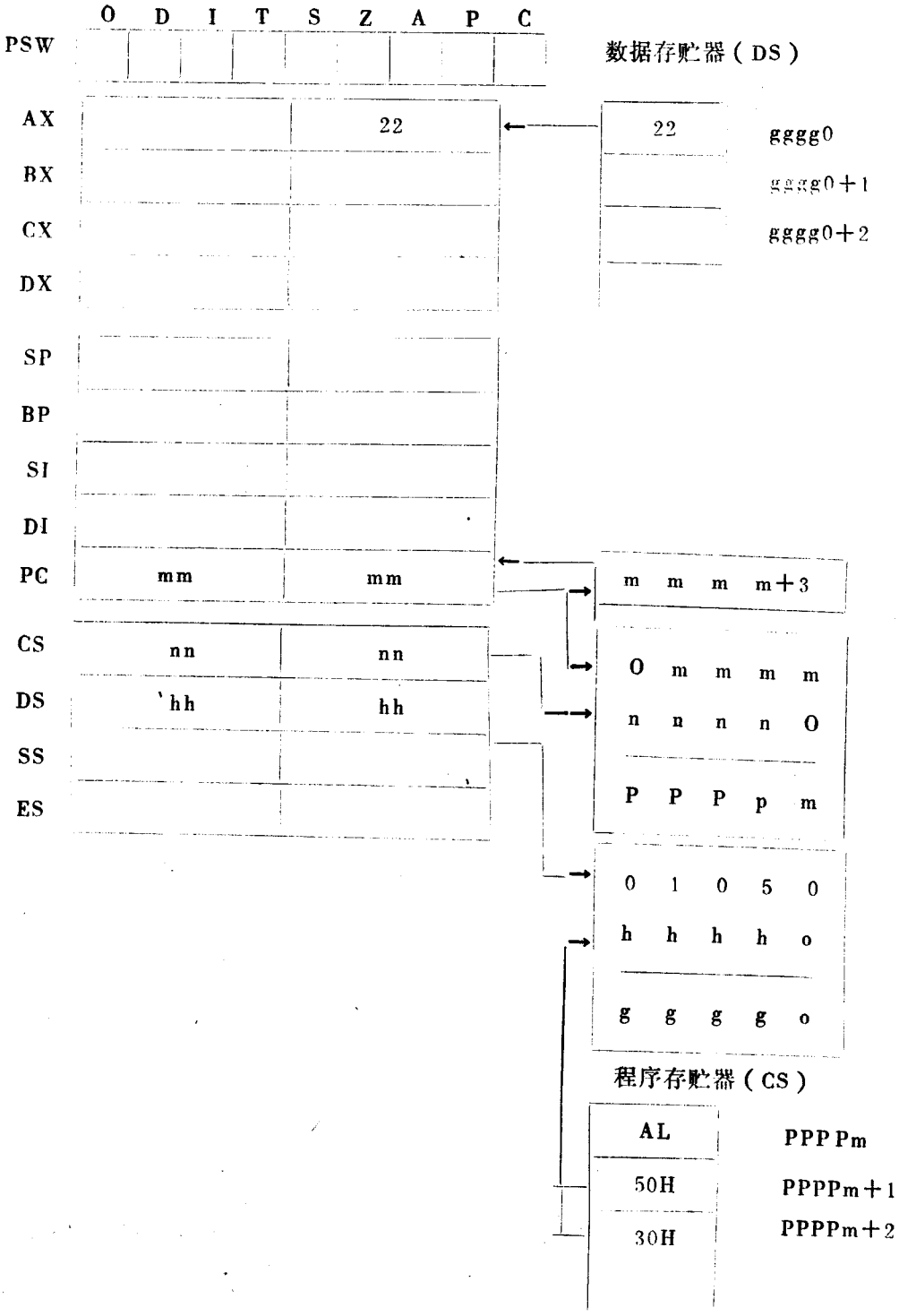


图 6~3　例 3 执行过程

例 4：把累加器的内容传送到存贮器中。

MOV 〔1050 H〕, AX

这条指令的功能是将 AX 中的内容传送到 1050 单元和 1051 单元之中。(相对于 DS 寄存器)。不影响标志位。

设 AX 的内容为 2233 。

指令编码：

1	0	1	0	0	0	1	1
0	0	1	1	0	0	1	1
0	0	1	0	0	0	1	0

执行完指令后，数据段 1050 和 1051 单元的内容分别为 33 和 22 。其过程见图 6-4 。

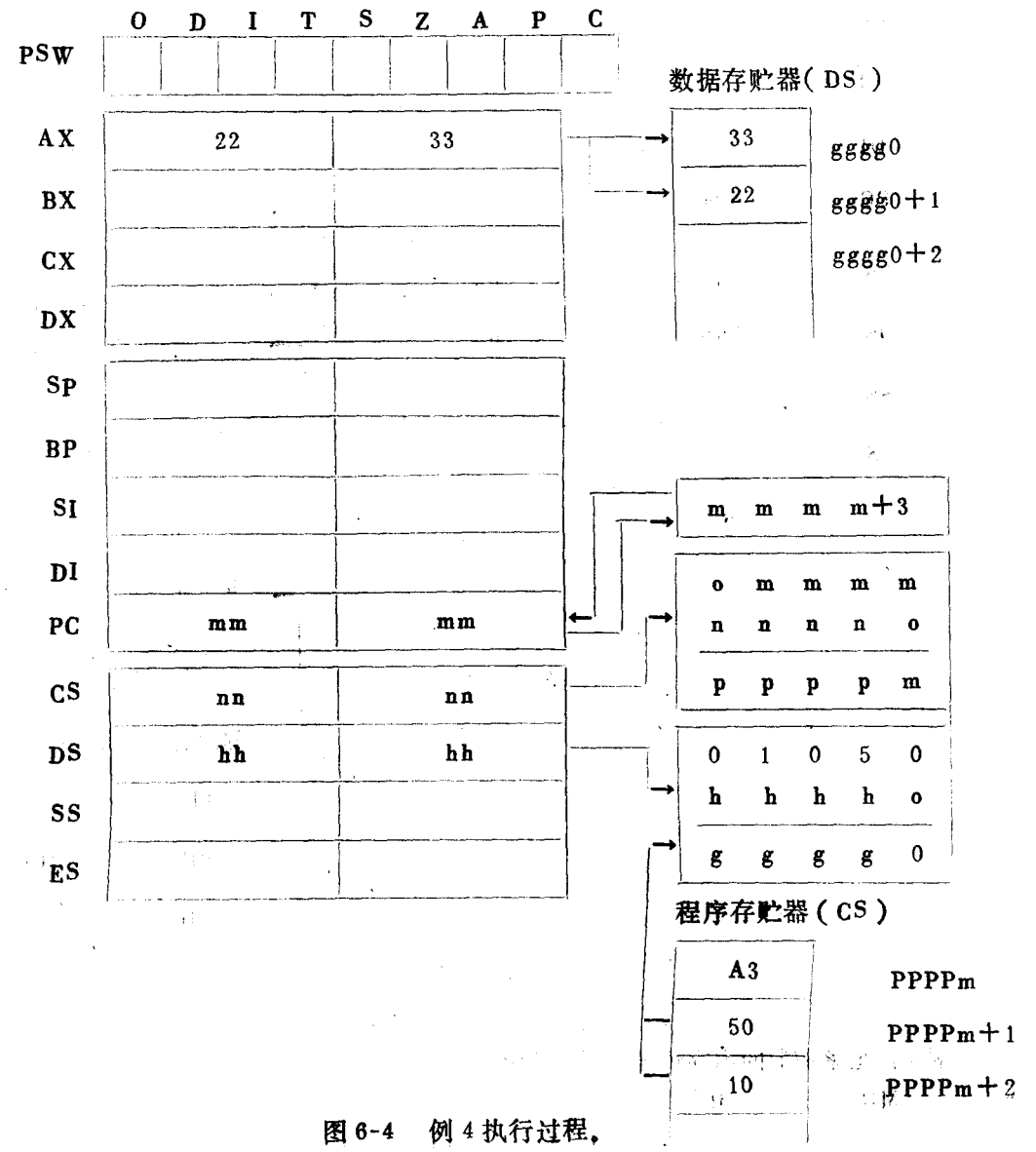


图 6-4　例 4 执行过程。

例 5 ：将寄存器或存贮单元中的16位数据传送到段寄存器中去。

指令编码

1	0	0	0	1	1	1	0

mod	o	reg	r/m

reg 占两位。指出段寄存器

00 ——— ES

10 ——— SS

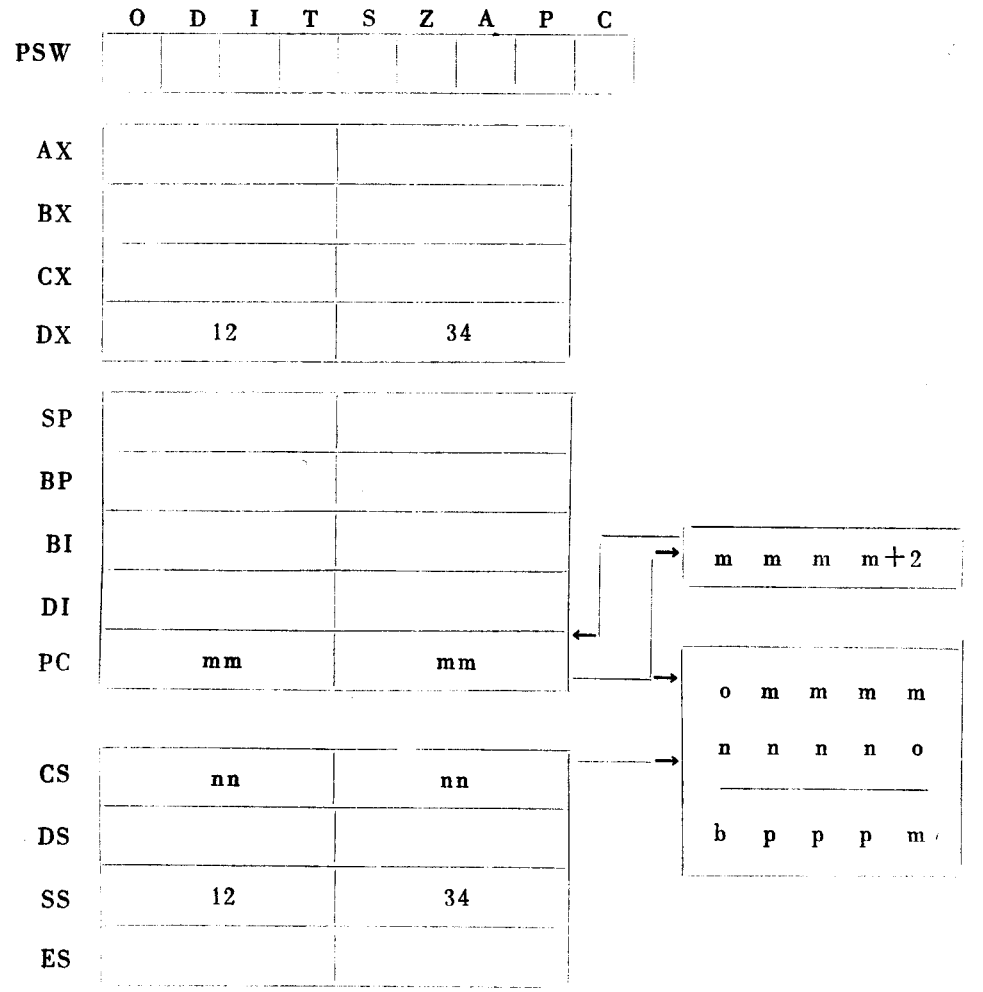
11 ——— DS

mod，r/m指出存贮器/寄存器操作数

MOV SS，DX

将 DX 的内容传送到 SS 寄存器

设 DX 的内容为 1234，SS 的内容为任意的。执行完该指令后。SS 寄存器的内容为 1234。执行过程见图 6-5。



程序存贮器（CS）

8E	PPPPm
D2	PPPPm+1
	PPPPm+2

图 6-5　例 5 执行过程

注意：1. 如果 reg ＝01。此操作数就无意义了。

这样就禁止用户直接地存入到 CS 寄存器中。只有执行 Jmp，CALL，RET，IRET，INT 指令时才会改变 CS 寄存器的内容。

2. 用于初始化时序，指令段域。

3. 该指令结束时，才采样中断。而是在该指令的下一条指令结束时才采样中断。这样就允许当装入 SS 和 PC 时在无中断的情况下，存满 32 位指针。这是很关键的一点。

例 6：将段寄存器中 16 的位数据传送到寄存器或存贮单元中。我们用这样一条指令：

MO6〔2100H〕，DS

设存贮单元 22100 的内容是 00H；22101 的内容是 20H。DS 的内容为2000。此指令对标志位不影响。

编码：

1	0	0	0	1	1	0	0

mod	reg	r/m

mod 和 r/m 用来指定存贮器或寄存器操作数。

reg 是 2 位，用来指定段寄存器。

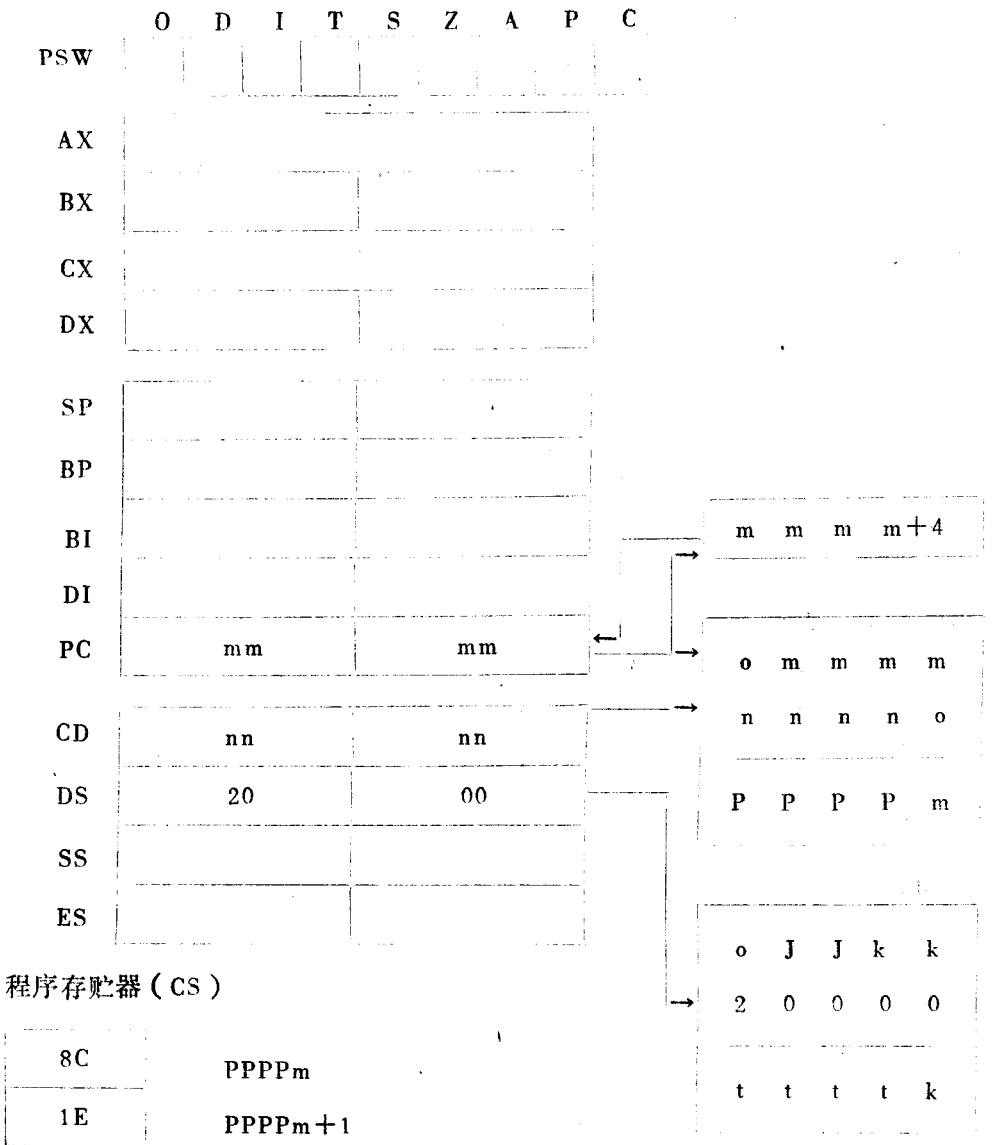
rr＝00　ES

01　CS

10　SS

11　DS

执行过程如图 6-6 所示。



程序存贮器（CS）

8C	PPPPm
1E	PPPPm+1
KK	PPPPm+2
JJ	PPPPm+3

数据储存器（DS）

00	ttttk
20	ttttk+1
	ttttk+2

图 6-6　例 6 执行过程

例 7：把操作码后面的立即数装入到指定的寄存器或存贮器单元中。（无论 8 位或 16 位数据都可以传送）。

我们用这样一条指令：

MOV〔BX〕, 371AH

设：DS 寄存器中的内容是 DOOOH；BX 寄存器中的内容是 0016H；执行完该指令之后。存贮单元 D0016H 中的内容是 1AH，D0017H的内容是 37H 。此指令不影响标志位。

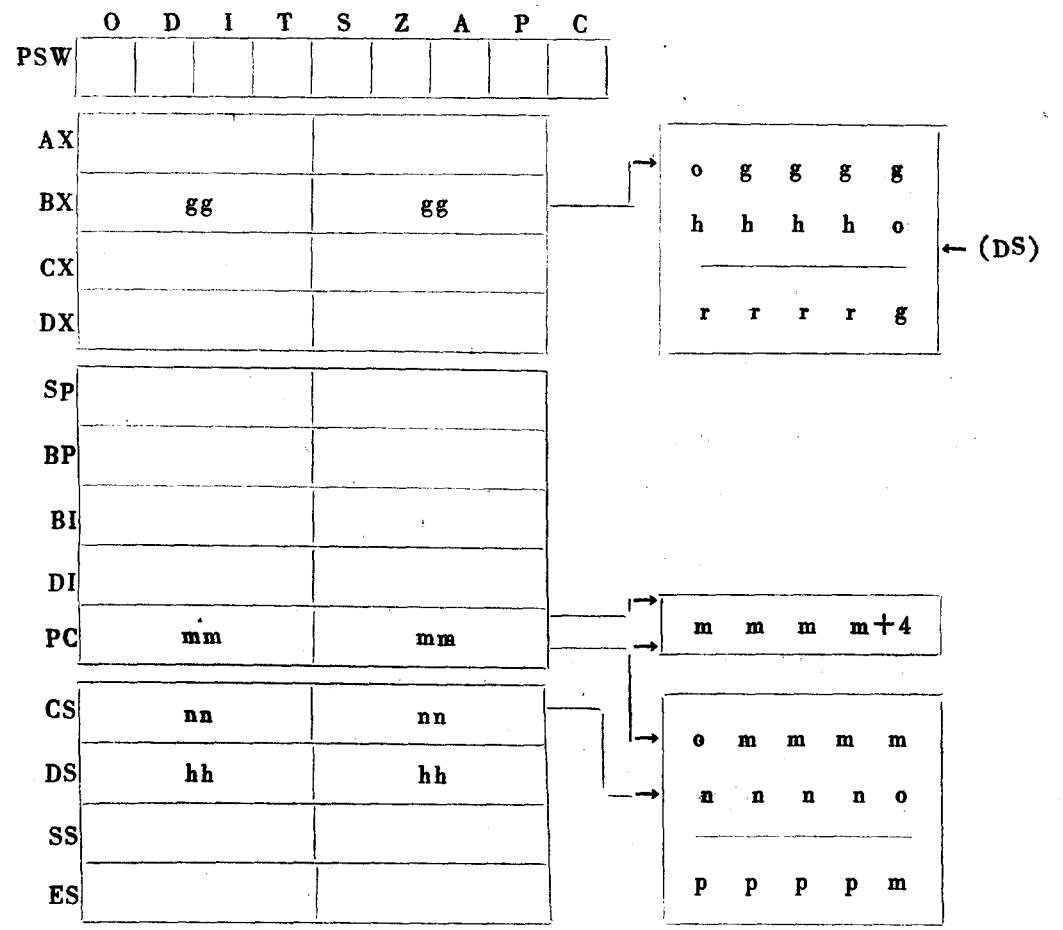
指令编码：

1	1	0	0	0	1	1	W
mod		0	0	0	r/m		
0	0	0	1	1	0	1	0
0	0	1	1	0	1	1	1

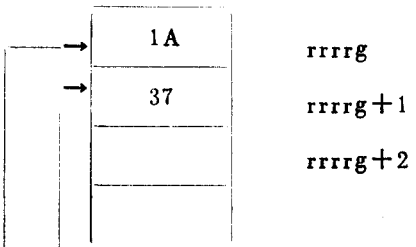
当W＝0时，是 8 位操作数。

当W＝1时，是 16 位操作数。

mod 000 r/m是寻址方式字节。执行过程如图 6-7。



数据存贮器（DS）



程序存贮器（CS）

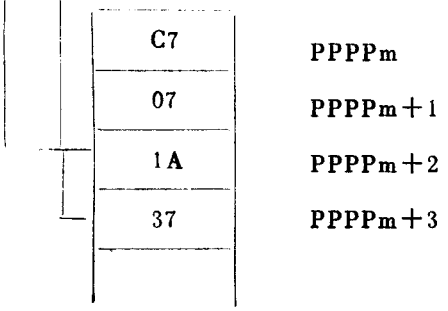


图 6-7 例 7 的执行过程

1.本指令中不可指定段寄存器

2.本指令典型地不用于将立即数送入寄存器中。

2.PUSH——入栈指令

这是一种将字压入堆栈的指令。它一共有三种类型。

它的具体操作是：

1.)堆栈指示器减2

（SP）－2→（SP）。

2.)将指令中指定的操作数内容存入 SP 指向的栈顶单元中去。SP 的内容为相对于在SS中的“栈基地址”的偏移地址。

〔（SP+1）：（SP）〕←（SRC）

第一种类型是：写入到堆栈中。即将指定的寄存器或者存贮器单元的内容压入到栈顶上。这是一个 16 位的压入操作。

指令编码

1	1	1	1	1	1	1	1

mod	1	1	0	r/m

此指令为两字节指令，第一字节为指令的操作码字节，第二个字节为寻址方式字节。

例 1：设 DS 寄存器的内容为 2700 H；

BX 寄存器的内容为 0300 H；

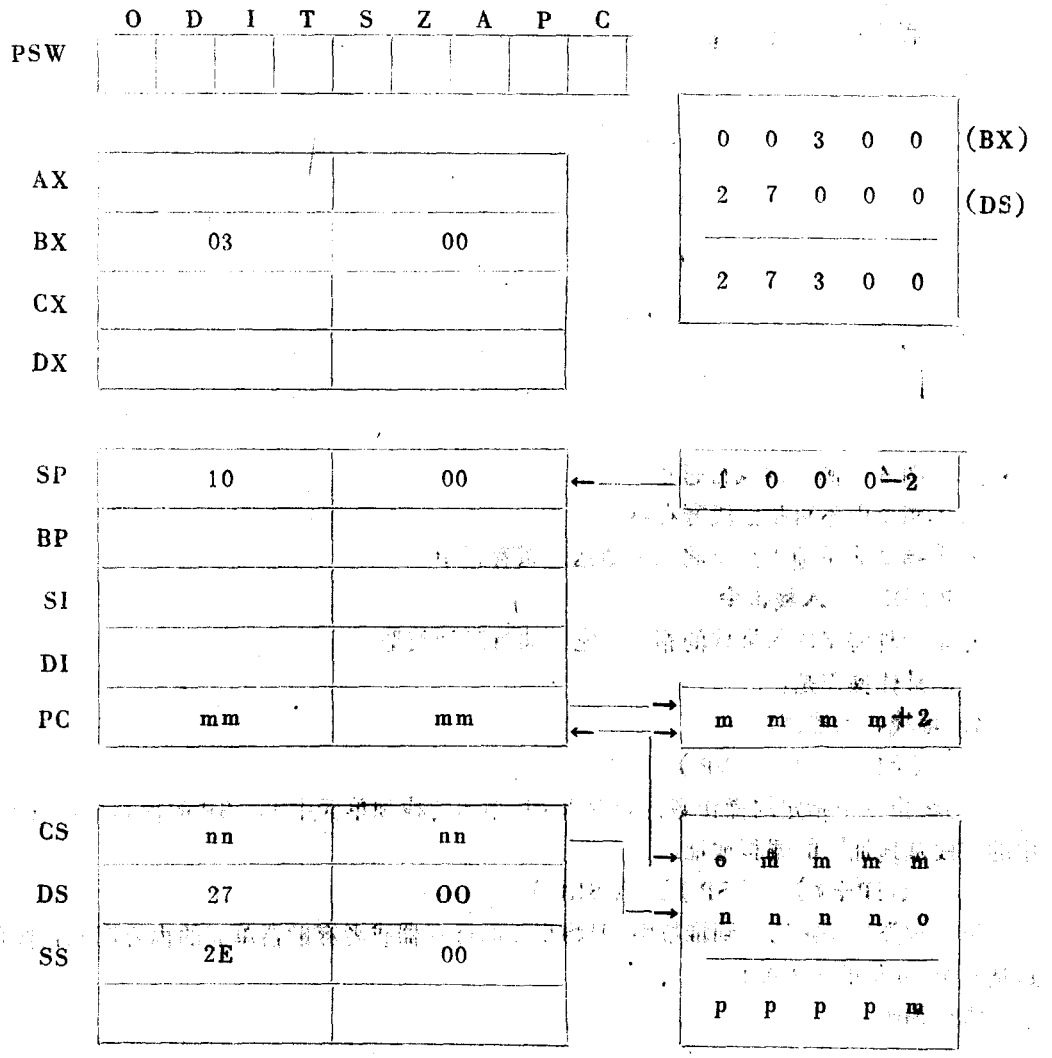
SP 寄存器的内容为 1000 H；

SS 寄存器的内容为 2E00 H；

存贮单元 27300 和 27301 的内容为 B010H。

执行 PUSH〔BX〕指令后；

存贮单元 2EFFFH的内容是 B0H；存贮单元 2EFFEH 的内容是 10H；SP 寄存器的内容为 OFFEH。不影响标志位。执行过程如图 6-8。



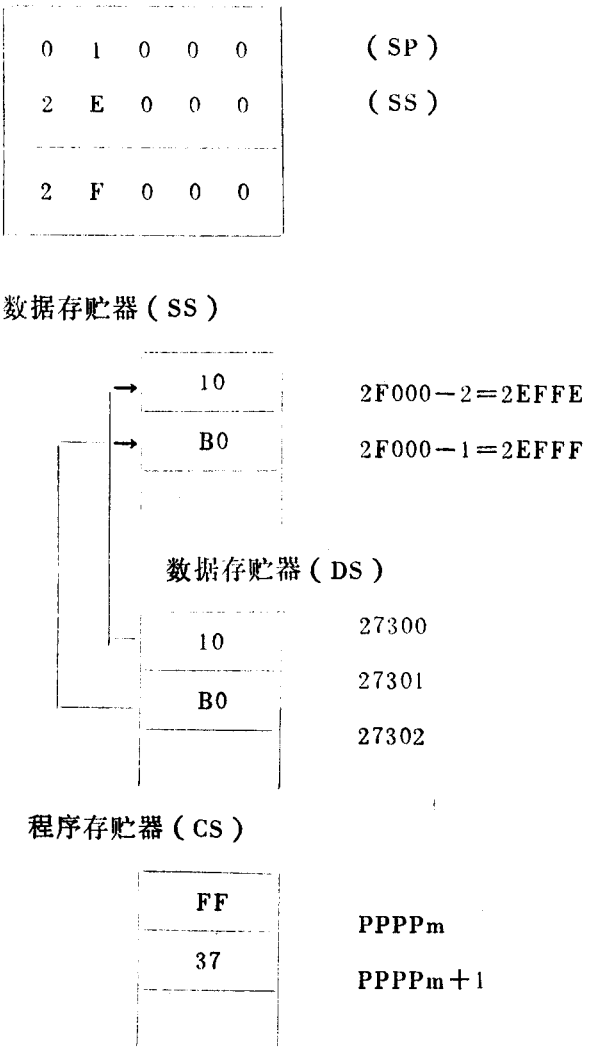


图 6-8　例 1 执行过程

注意：这条指令典型地不是用在将寄存器的内容压入堆栈。可用指令 PUSH reg 执行这个功能。因为它只占用程序存贮器的一个字节。

第二种类型是：写入到栈顶。即用这个指令压入指定的 16 位寄存器的内容到栈顶。

指令编码：

0	1	0	1	0	r	r	r

rrr：3 位；指定要压入的 16 位寄存器。

rrr＝000　AX

001　CX

010　DX

001 BX

100 SP

101 BP

110 SI

111 DI

例 2：将 SI 寄存器的内容（16位）压入到堆栈中。

PUSH SI

这条指令具体的执行步骤如下：

1. 堆栈指针减 1 。

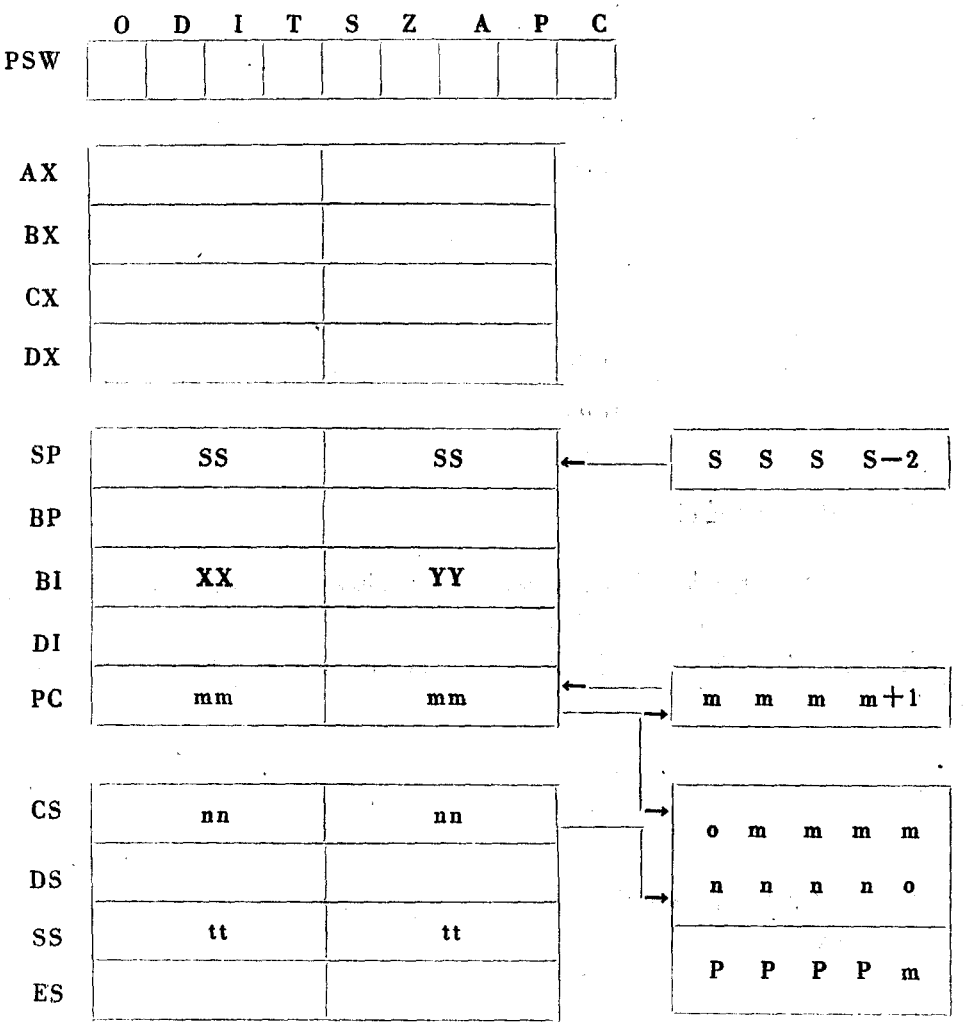
2. 指定的寄存器的高 8 位存入到由堆栈指针和堆栈段寄存器指定的存贮器单元中。

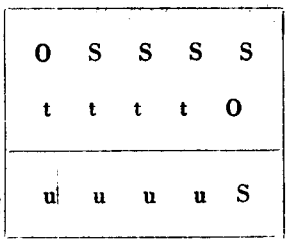
3. 堆栈指针减 1 。

4. 指定的寄存器的低 8 位存入到由堆栈指针和堆栈段寄存器指定的存贮器单元中。

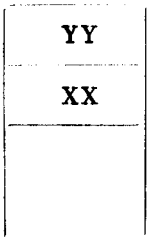
堆栈指针总是指向栈顶，实际上在 8086 中是进行着 16 位传送，不影响标志位。

执行过程如图 6-9 所示。





数据贮存器（SS）



程序存贮器（CS）

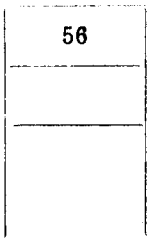


图 6-9　例 2 执行过程

注意：1.此指令不能用于压入段寄存器或标志寄存器。可由 PUSH Seyrey。PUSHF 指令执行这些操作。

2.在堆栈指针初始化之后。此指令就可以使用了。实际上，也只在现堆栈指针初始化之后。才应该使用这条指令。

第三类型是：写入到栈顶。用指令将指定的 16 位段寄存器的内容压入到栈顶。

段寄存器的编码为

0 0 0 S S 1 1 0

SS 是两位。指定要压入堆栈的段寄存器。

SS＝00　　ES

SS＝10　　SS

SS＝11　　DS

注意：如果 SS＝01，则是非法操作。