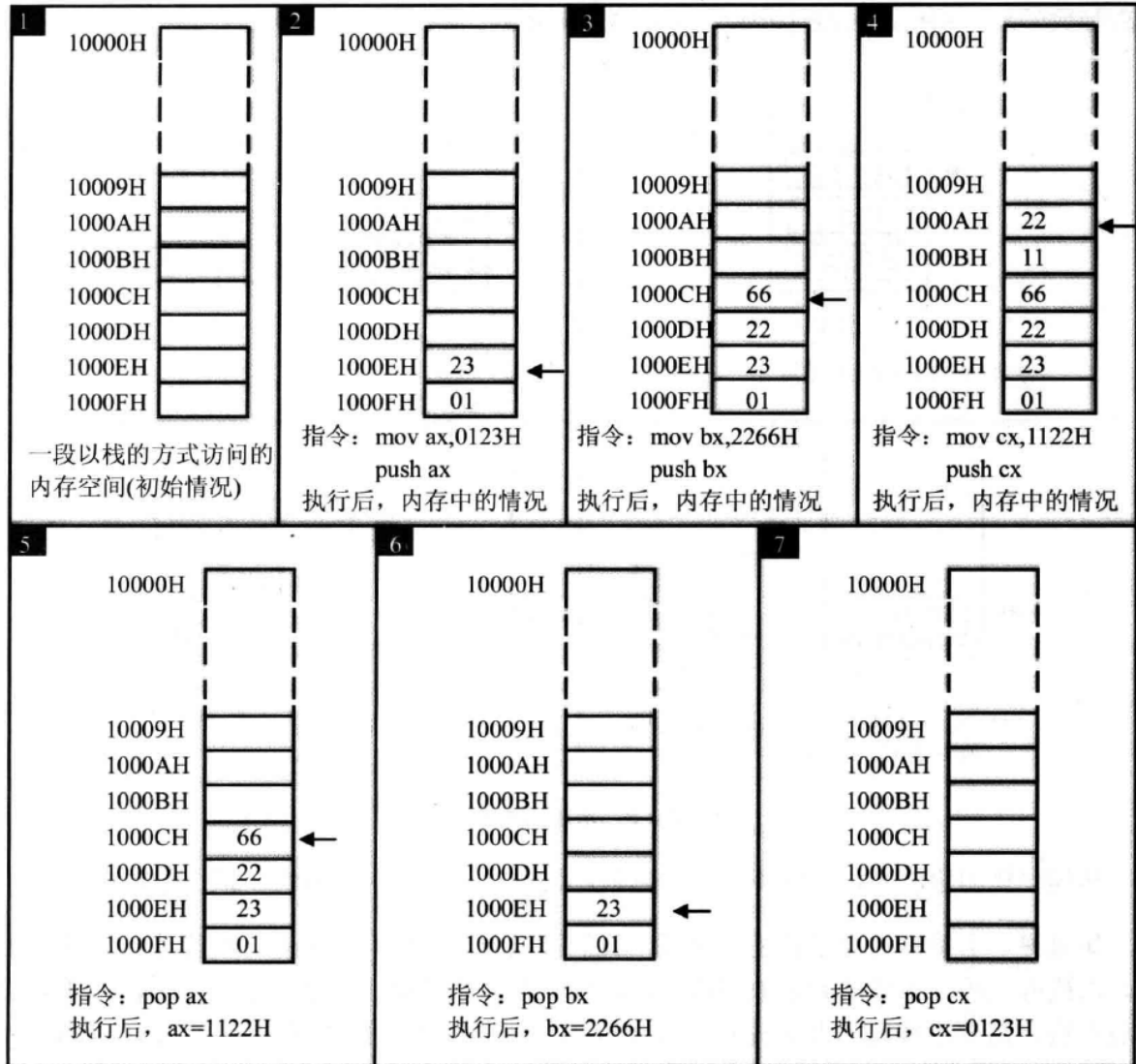


# 栈操作指令

- `push ax`: 将ax中数据送入栈中
- `pop ax`: 将栈顶数据取出送入ax
- `ss:sp` 指向栈顶

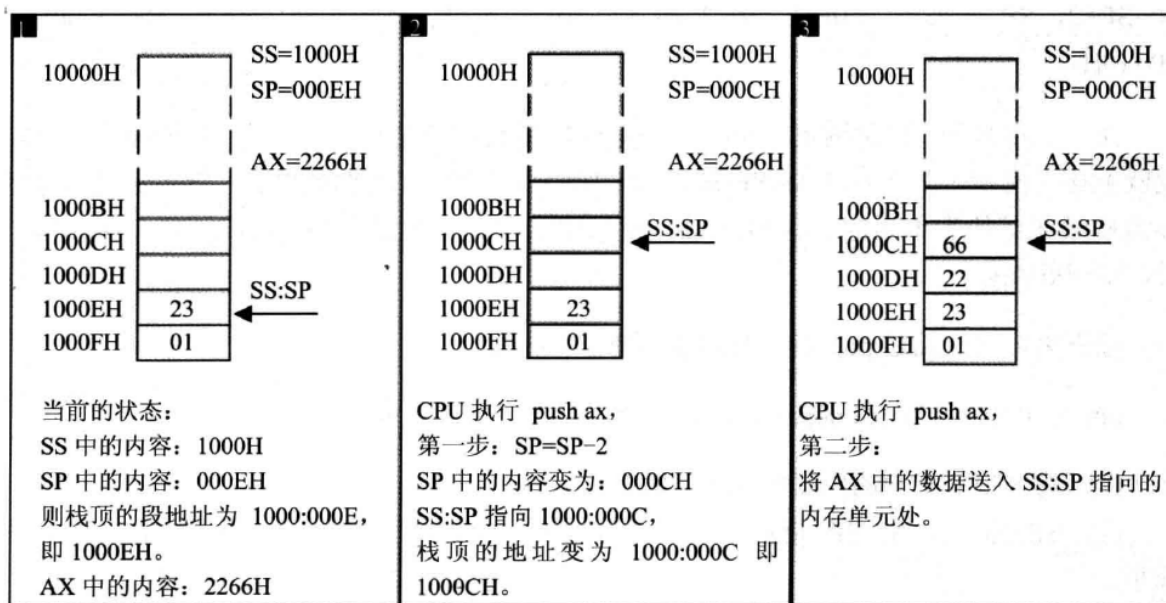


```
mov ax, 0123H
push ax
mov bx, 2266H
push bx
mov cx, 1122H
push cx
pop ax
pop bx
pop cx
```

`push ax` 执行，由下列两步完成

1.  $SP = SP - 2$

2. 将 `ax` 中数据送入新的 `SS:SP` 指向位置



## CF标志位

flag的第0位是CF，在进行无符号运算的时候，记录运算结果的最高有效位向更高位的进位值，或者从更高位的借位值。

## OF标志位

记录有符号数是否溢出。

## CMP指令

指令格式： `cmp 操作对象1, 操作对象2`

指令功能：计算 操作对象1-操作对象2，但并不保存结果，

下列为根据无符号数的比较结果进行转移的条件转移指令：

指令	含义	检测的相关标志位
je	等于则转移	zf=1
jne	不等于则转移	zf=0
jb	低于则转移	cf=1
jnb	不低于则转移	cf=0
ja	高于则转移	cf=0 且 zf=0
jna	不高于则转移	cf=1 或 zf=1

## 练习1

各寄存器的初始值: CS=2000H, IP=0, DS=1000H, AX=0, BX=0;

- ① 写出 CPU 执行的指令序列(用汇编指令写出)。
- ② 写出 CPU 执行每条指令后, CS、IP 和相关寄存器中的数值。
- ③ 再次体会: 数据和程序有区别吗? 如何确定内存中的信息哪些是数据, 哪些是程序?

10000H	B8	} mov ax, 2000H	20000H	B8	} mov ax, 6622H
10001H	00		20001H	22	
10002H	20		20002H	66	
10003H	8E	} mov ds, ax	20003H	EA	} jmp 0ff0:0100
10004H	D8		20004H	00	
10005H	A1	} mov ax, [0008]	20005H	01	
10006H	08		20006H	F0	
10007H	00	} mov ax, [0002]	20007H	0F	} mov bx, ax
10008H	A1		20008H	89	
10009H	02		20009H	C3	
1000AH	00		2000AH		
1000BH			2000BH		
1000CH			2000CH		

## 练习2

编程，将 10000H~1000FH 这段空间当作栈，初始状态栈是空的，将 AX、BX、DS 中的数据入栈。

思考后看分析。

分析：

代码如下。

```
mov ax,1000H
mov ss,ax      ;设置栈的段地址，SS=1000H，不能直接向段寄存器 SS 中送入
               ;数据，所以用 ax 中转。

mov sp,0010H   ;设置栈顶的偏移地址，因栈为空，所以 sp=0010H。

               ;上面的 3 条指令设置栈顶地址。编程中要自己注意栈的大小。

push ax
push bx
push ds
```