
 **练习题 2.12** 对于下面的值, 写出变量 x 的 C 语言表达式。你的代码应该对任何字长 $w \geq 8$ 都能工作。我们给出了当 $x=0x87654321$ 以及 $w=32$ 时表达式求值的结果, 仅供参考。

- A. x 的最低有效字节, 其他位均置为 0。[0x00000021]。
- B. 除了 x 的最低有效字节外, 其他的位都取补, 最低有效字节保持不变。[0x789ABC21]。
- C. x 的最低有效字节设置成全 1, 其他字节都保持不变。[0x876543FF]。

 **练习题 2.13** 从 20 世纪 70 年代末到 80 年代末, Digital Equipment 的 VAX 计算机是一种非常流行的机型。它没有布尔运算 AND 和 OR 指令, 只有 bis(位设置)和 bic(位清除)这两种指令。两种指令的输入都是一个数据字 x 和一个掩码字 m 。它们生成一个结果 z , z 是由根据掩码 m 的位来修改 x 的位得到的。使用 bis 指令, 这种修改就是在 m 为 1 的每个位置上, 将 z 对应的位设置为 1。使用 bic 指令, 这种修改就是在 m 为 1 的每个位置, 将 z 对应的位设置为 0。

为了看清楚这些运算与 C 语言位级运算的关系, 假设我们有两个函数 bis 和 bic 来实现位设置和位清除操作。只想用这两个函数, 而不使用任何其他 C 语言运算, 来实现按位 | 和 ^ 运算。填写下列代码中缺失的代码。提示: 写出 bis 和 bic 运算的 C 语言表达式。

```
/* Declarations of functions implementing operations bis and bic */
int bis(int x, int m);
int bic(int x, int m);

/* Compute x|y using only calls to functions bis and bic */
int bool_or(int x, int y) {
    int result = _____;
    return result;
}

/* Compute x^y using only calls to functions bis and bic */
int bool_xor(int x, int y) {
    int result = _____;
    return result;
}
```

2.1.8 C 语言中的逻辑运算

C 语言还提供了一组逻辑运算符 ||、&& 和 !, 分别对应于命题逻辑中的 OR、AND 和 NOT 运算。逻辑运算很容易和位级运算相混淆, 但是它们的功能是完全不同的。逻辑运算认为所有非零的参数都表示 TRUE, 而参数 0 表示 FALSE。它们返回 1 或者 0, 分别表示结果为 TRUE 或者为 FALSE。以下是一些表达式求值的示例。

表达式	结果
!0x41	0x00
!0x00	0x01
!!0x41	0x01
0x69&&0x55	0x01
0x69 0x55	0x01

可以观察到, 按位运算只有在特殊情况下, 也就是参数被限制为 0 或者 1 时, 才和与