

第三十二节：逻辑运算符的“异或”运算。

【32.1 “异或”运算。】

“异或”运算也是以位为单位进行运算的。位是指二进制中的某一位，位只能是 0 或者 1。两个数的“异或”运算就是转换成二进制后每一位的“异或”运算。

“异或”运算的符号是“ \wedge ”。运算规律是：两个位的“异或”运算，如果两个位都相同，那么运算结果就是 0；如果两个位不同（相异），则运算结果是 1。比如：

$0\wedge 0$ 等于 0。（两个位相同）
 $0\wedge 1$ 等于 1。（两个位相异）
 $1\wedge 0$ 等于 1。（两个位相异）
 $1\wedge 1$ 等于 0。（两个位相同）

现在举一个完整的例子来分析“ \wedge ”运算的规律。有两个 unsigned char 类型的十进制数分别是 12 和 9，求 $12\wedge 9$ 的结果是多少？分析步骤如下：

第一步：先把参与运算的两个数以二进制的格式展开。十进制转二进制的方法请参考前面第 14, 15, 16 节的内容。

十进制 12 的二进制格式是：00001100。
十进制 9 的二进制格式是： 00001001。

第二步：二进制数右对齐，按上下每一位进行“异或”运算。

十进制的 12	->	00001100
十进制的 9	->	\wedge 00001001
“异或”运算结果是	->	00000101

第三步：把二进制的 00000101 转换成十六进制是：0x05。转换成十进制是 5。所以 $12\wedge 9$ 的结果是 5。

【32.2 “异或”在项目中的应用。】

“异或”在哪些项目上经常应用？以我个人的项目经验，平时很少用“异或”，我本人在项目中用过两次“异或”，第一次是在某项目做串口通讯协议时，通过“异或”算法，增加一个校验字节，此校验字节是一串数据依次相“异或”的总结果，目的是为了增加数据传送时的抗干扰能力。第二次是把它用来对某变量的某个位进行取反运算，如何用“异或”来实现对某位进行取反的功能？要实现这个功能，首先要清楚“异或”运算有一个潜在的规律：任何一个位，凡是与 0 进行“异或”运算都保持不变，凡是与 1 进行“异或”运算都会达到取反的运算效果。因此，如果想某位实现取反的功能，只要把相关的位与“1”进行“异或”运算就可以实现取反的功能。二进制中的一个位要么是 0，要么是 1，不管是 0 还是 1，只要与 1 进行“异或”运算，是会达到取反的运算目的，0 的会变成 1，1 的会变成 0。请看以下这个例子：

$0\wedge 1$ 等于 1。（两个位相异）
 $1\wedge 1$ 等于 0。（两个位相同）

以上的例子只是列举了一个位，如果把一个字节的 8 位展开来，只要某位与“1”进行“异或”运算，都可以实现某位取反的功能。比如，一个十六进制的 0x55，如果要这个字节的低 4 位都取反，高 4 位不变，

只需要把该数据与十六进制的 0x0F 进行“异或”运算就可以达到目的。请看以下这个例子：

十六进制的 0x55	->	01010101
十六进制的 0x0F	->	^00001111
“异或”运算结果是	->	01011010

上述运算结果二进制的 01011010 转换成十六进制是 0x5A，转换成十进制是 90。

【32.3 例程练习和分析。】

现在编写一个程序来验证刚才讲到的“异或”运算：

程序代码如下：

```
/*---C 语言学习区域的开始。-----*/

void main() //主函数
{
    unsigned char a;
    unsigned char b;

    a=12^9;
    b=0x55^0x0F;

    View(a);           //把第 1 个数 a 发送到电脑端的串口助手软件上观察。
    View(b);           //把第 2 个数 b 发送到电脑端的串口助手软件上观察。

    while(1)
    {
    }
}

/*---C 语言学习区域的结束。-----*/
```

在电脑串口助手软件上观察到的程序执行现象如下：

开始...

第 1 个数

十进制:5

十六进制:5

二进制:101

第 2 个数

十进制:90

十六进制:5A

二进制:1011010

分析：

通过实验结果，发现在单片机上的计算结果和我们的分析是一致的。

【32.4 如何在单片机上练习本章节 C 语言程序？】

直接复制前面章节中第十一节的模板程序，练习代码时只需要更改“C 语言学习区域”的代码就可以了，其它部分的代码不要动。编译后，把程序下载进带串口的 51 学习板，通过电脑端的串口助手软件就可以观察到不同的变量数值，详细方法请看第十一节内容。