

第三十节：逻辑运算符的“与”运算。

【30.1 “与”运算。】

不管是十进制还是十六进制，单片机底层的运算都是以二进制的形式进行的，包括前面章节的加减乘除运算，在单片机的底层处理也是以二进制形式进行。只不过加减乘除我们平时太熟悉了，以十进制的形式口算或者笔算也能得到正确的结果，所以不需要刻意把十进制的数据先转换成二进制，然后再模拟单片机底层的二进制运算。但是本节的逻辑“与”运算，在分析它的运算过程和规律的时候，必须把所有的数据都转化成二进制才能进行分析，因为它强调的是二进制的位与位之间的逻辑运算。我们知道，二进制中的每一位只能是 0 或者 1，两个数的“与”运算就是两个数被展开成二进制后的位与位之间的逻辑“与”运算。

“与”运算的运算符是“&”。运算规律是：两个位进行“与”运算，只有两个位都同时是 1 运算结果才能等于 1，否则，只要其中有一位是 0，运算结果必是 0。比如：

```
0&0 等于 0。
0&1 等于 0。
1&0 等于 0。
1&1 等于 1。
```

注意，上述的 0 和 1 都是指二进制的 0 和 1。

现在举一个完整的例子来分析“与”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&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“与”运算。

```
十进制的 12      ->    00001100
十进制的 9       ->    &00001001
“与”运算结果是 ->    00001000
```

第三步：把二进制的 00001000 转换成十六进制是：0x08。转换成十进制是 8。所以 12&9 的结果是 8。

上述举的例子只能分析“与”运算的规律，并没有看出“与”运算的意义所在。“与”运算有啥用途呢？其实用途很多，最常见的用途是可以指定一个变量二进制格式的某位清零，其它位保持不变。比如一个 unsigned char 类型的变量 b，数据长度一共是 8 位，从右往左：

想让第 0 位清零，其它位保持不变，只需跟十六进制的 0xfe 相“与”：b=b&0xfe。

想让第 1 位清零，其它位保持不变，只需跟十六进制的 0xfd 相“与”：b=b&0xfd。

想让第 2 位清零，其它位保持不变，只需跟十六进制的 0xfb 相“与”：b=b&0xfb。

想让第 3 位清零，其它位保持不变，只需跟十六进制的 0xf7 相“与”：b=b&0xf7。

想让第 4 位清零，其它位保持不变，只需跟十六进制的 0xef 相“与”：b=b&0xef。

想让第 5 位清零，其它位保持不变，只需跟十六进制的 0xdf 相“与”：b=b&0xdf。

想让第 6 位清零，其它位保持不变，只需跟十六进制的 0xbf 相“与”：b=b&0xbf。

想让第 7 位清零，其它位保持不变，只需跟十六进制的 0x7f 相“与”：b=b&0x7f。

根据上述规律，假设 b 原来等于十进制的 85（十六进制是 0x55，二进制是 01010101），要想把此数据的第 0 位清零，只需 b=b&0xfe。最终 b 的运算结果是十进制是 84（十六进制是 0x54，二进制是 01010100）。把它们展开成二进制格式的运算过程如下：

十进制的 85	->	01010101
十六进制的 0xfe	->	&11111110
“与”运算结果是	->	01010100

【30.2 例程练习和分析。】

现在编写一个程序来验证刚才讲到的“与”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=85; //十六进制是 0x55，二进制是 01010101。

    a=12&9;
    b=b&0xfe;

    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:8

十六进制:8

二进制:1000

第 2 个数

十进制:84

十六进制:54

二进制:1010100

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。

【30.3 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。