5月11日

1、算术左移-逻辑左移 二者操作方式相同

算术左移和逻辑左移一样都是右边补 0： 比如 00101011

算术左移一位:01010110

逻辑左移一位:01010110

对于二进制的数值来说左移 n 位等于原来的数值乘以 2 的 n 次方

比如 00011010 十进制是 26，左移两位后是 01101000 转成十进制是 104 恰好是 26 的 4 倍。

ps：这种倍数关系只适用于左移后被舍弃的高位不含 1 的情况，否则会溢出。

2、算术右移，逻辑右移

逻辑右移很简单，只要将二进制数整体右移，左边补 0 即可

如 10101101 逻辑右移一位为 01010110

算术右移符号位要一起移动，并且在左边补上符号位，也就是如果符号位是 1 就补 1 符号位是 0 就补 0

比如：11100 算术右移一位为 11110（符号位 1 跟着一起移动并且左边补了 1）

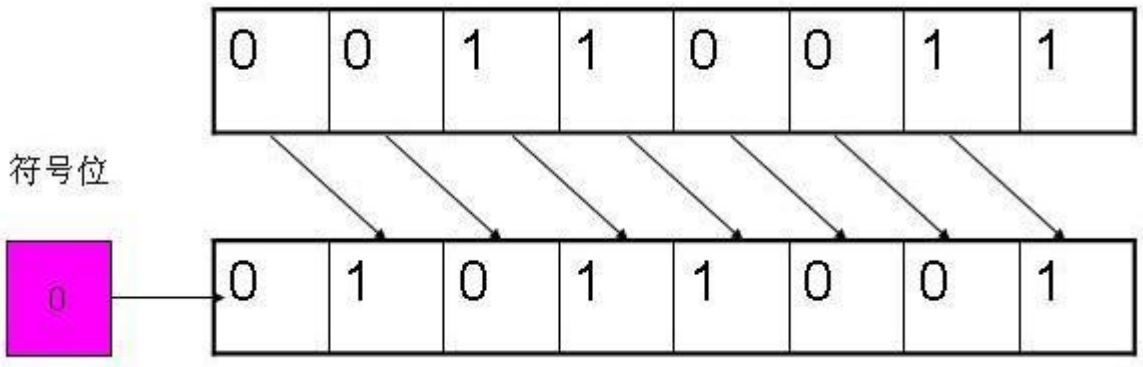
对于二进制的数值来说右移 n 位等于原来的数值除以 2 的 n 次方

负数

比如 10110100 十进制是 76（需要先将这个补码转换成原码之后再转换成十进制），右移两位后是 11101101 转成十进制是 19 恰好是 76 的 4 倍。

算术左移和算术右移主要用来进行有符号数的倍增、减半；

逻辑左移和逻辑右移主要用来进行无符号数的倍增、减半。



**扩展资料：**

移位操作是计算机指令中比较基本的操作，是位运算的一种。

在移位运算时，byte、short 和 char 类型移位后的结果会变成 int 类型，对于 byte、short、char 和 int 进行移位时，编译器未做任何优化的情况下（优化后不可预期），规定实际移动的次数是移动次数和 32 的余数，也就是移位 33 次和移位 1 次得到的结果相同。

移动 long 型的数值时，规定实际移动的次数是移动次数和 64 的余数，也就是移动 66 次和移动 2 次得到的结果相同。

算数左移位，即算术左移位，是一种带符号的左移位运算。

## ARM 语言中逻辑移位和算术移位的区别

比如一个有符号位的 8 位二进制数 11001101，逻辑右移就不管符号位，如果移一位就变成 01100110。

算术右移要管符号位，右移一位变成 10100110。

逻辑左移=算数左移，右边统一添 0

逻辑右移，左边统一添 0

算数右移，左边添加的数和符号有关 e.g:1010101010，其中[]位是添加的数字

逻辑左移一位：010101010[0]

算数左移一位：010101010[0]

操作方式一样

逻辑右移一位：[0]101010101

算数右移一位：1]101010101

算术左移和算术右移主要用来进行有符号数的倍增、减半；逻辑左移和逻辑右移主要用来进行无符号数的倍增、减半。记住这个就可以了。

算术左移和算术左移虽然方式是一样的，但他们表示的移位后数的范围是不一样的，有符号数左移（算术左移）位后的范围是-128——127【指 8 位】而无符号数（算术左移）左移的范围是 0——255.【指 8 位】。

其实不管是哪种移位（上述的）

1.汇编语言中的逻辑右移(SHR)是将各位依次右移指定位数，然后在左侧补 0,算术右移(SAR)是将各位依次右移指定位数，然后在左侧用原符号位补齐。

2.高级语言右移运算符（>>）是将一个数的二进位全部右移若干位低位移出部分舍弃，左补 0。

3.高级语言右移和汇编语言中的逻辑右移功能一样，但不同于算术右移。