

### 第三章 运算方法和运算部件

#### 2. 简单回答下列问题。(参考答案略)

- (1) 为何在高级语言和机器语言中都要提供“按位运算”？为何高级语言需要提供逻辑运算？按位运算和逻辑运算的差别是什么？
- (2) 如何进行逻辑移位和算术移位？它们各用于哪种类型的数据？
- (3) 移位运算和乘除运算具有什么关系？
- (4) 高级语言中的运算和机器语言（即指令）中的运算是有什么关系？假定某一个高级语言源程序 **P** 中有乘、除运算，但机器 **M** 中不提供乘、除运算指令，则程序 **P** 能否在机器 **M** 上运行？为什么？
- (5) 为什么用一个 **ALU** 和移位器就能实现定点数和浮点数的所有加、减、乘、除运算？
- (6) 影响加/减运算速度的关键问题是什么？可采取什么改进措施？

#### 5. 以下是两段 C 语言代码，函数 **arith()** 是直接 C 语言写的，而 **optarith()** 是对 **arith()** 函数以某 **M** 和 **N** 编译生成的机器代码反编译生成的。根据 **optarith()**，可以推断函数 **arith()** 中 **M** 和 **N** 的值各是多少？

```
#define M
#define N
int arith (int x, int y)
{
    int result = 0;
    result = x*M + y/N;
    return result;
}
```

```
int optarith (int x, int y)
{
    int t = x;
    x <<= 4;
    x -= t;
    if (y < 0) y += 3;
    y >> 2;
    return x+y;
}
```

#### 参考答案：

对反编译结果进行分析，可知：对于 **x**，指令机器代码中有一条“**x** 左移 4 位”指令，即：**x=16x**，然后有一条“减法”指令，即 **x=16x-x=15x**，所以，根据源程序，知 **M=15**；对于 **y**，有一条“**y** 右移 2 位”指令，即 **y=y/4**，根据源程序，知 **N=4**。（当 **y<0** 时，**(y+3)/4=y/4**，若不调整，则“-1>>2=-1 而本来-1/4=0”，故使-1+3=2，2/4=0）

#### 7. 利用 SN74181 和 SN74182 器件设计一个 16 位并行进位补码加/减运算器，画出运算器的逻辑框图，并给出零标志、进位标志、溢出标志、符号标志的生成电路。

#### 参考答案：（略）

9. 已知二进制数  $x = 0.1010$ ,  $y = -0.1101$ 。请按如下要求计算, 并把结果还原成真值。

- (1) 求  $[x+y]_{\text{补}}$ ,  $[x-y]_{\text{补}}$ 。
- (2) 用原码一位乘法计算  $[x*y]_{\text{原}}$ 。
- (3) 用布斯乘法计算  $[x*y]_{\text{补}}$ 。
- (4) 用不恢复余数法计算  $[x \div y]_{\text{原}}$  的商和余数。
- (5) 用不恢复余数法计算  $[x \div y]_{\text{补}}$  的商和余数。

**参考答案: (略)**

13. 假设浮点数格式为: 阶码是 4 位移码, 尾数是 6 位补码 (采用双符号位), 用浮点运算规则分别计算在不采用任何附加位和采用 2 位附加位 (保护位、舍入位) 两种情况下的值。(假定对阶和右规时采用就近舍入到偶数方式)

- (1)  $(15/16)*2^7 + (2/16)*2^5$
- (2)  $(15/16)*2^7 - (2/16)*2^5$
- (3)  $(15/16)*2^5 + (2/16)*2^7$
- (4)  $(15/16)*2^5 - (2/16)*2^7$

**参考答案: 采用 2 位附加位的情况:**

$x = (15/16)*2^7$  的浮点数表示为: **1111, 00.1111**;  $y = (2/16)*2^5$  的浮点数表示为: **1101, 00.0010**

对阶: **1111+0011=0010 (+2)**; 对  $y$  进行: **1111, 00.000010**

尾数相加: **00.111100+00.000010=00.111110**

舍入: **1111, 01.0000**

右规: 右规前阶码已经为 **1111**, 所以结果“溢出”。

(其余略)

14. 采用 IEEE754 单精度浮点数格式计算:  $0.75 + (-65.25)$

**参考答案: (略)**

15. 采用十进制数 (NBCD 码) 加法运算的方法, 计算下列各式。并讨论在十进制 BCD 码加法运算中如何判断溢出。

- (1)  $234 + 567$
- (2)  $548 + 729$

**参考答案: (略)**

先确定位数, 最高位有进位, 则“溢出”