《计算机组成原理》
作业3

2023

33920212204567　任宇

3-1

1. 全加器：用门电路实现两个二进制数相加并求出和的组合线路，称为一位全加器

2. 半加器：半加器电路是对两个输入数据位相加，输出一个结果位和进位，没有进位输入的加法器电路

3. 进位生成函数：n位加法器中，每一位定义一个函数：Gi = Ai &Bi。只要Gi为1，不管低位有没有进位，一定能生成向高位的进位，这个函数称为进位生成函数

4. 进位传递函数：n位加法器中，每一位定义一个函数：Pi = Ai | ，只要Pi为1，那么，低位来的进位一定能传递到高位，这个函数称为进位传递函数

5. 算术移位：算术移位是对带符号数进行的，移位前后符号位不变，包括算术左移和算术右移

6. 逻辑移位：逻辑移位是对无符号数进行的移位，包括逻辑左移和逻辑右移

7. 阵列乘法器：阵列乘法器采用类似于人工计算的方法进行乘法运算，用大量与门阵列同时产生手动乘法中的各乘积项，同时将一位全加器按照需要构成全加器阵列。

8. 原码恢复余数除法：过程为：①对被除数和除数取绝对值，先做被除数-除数。②判断余数是否小于0，若是，则加上除数的补，否则减去除数的补。③移位，重复步骤②，直到移位次数等于除数的位数，上商的次数为移位次数+1。

3-1

9. 原码不恢复余数除法：第一次被除数对除数做一次差，判断最高位是否溢出，如果第一位商为0，则继续下去。移位后的处理由上一次的商决定，如果上一次商0，则对余数进行左移1位后-除数；若上一次商为1，则对余数左移1位后+除数

10. 阵列除法：利用一个可控加减CAS单元组成的流水阵列，有四个输出，四个输入。P=0时，做加法，P=1时，做减法

11. 串行进位：串行进位又称为行波进位，每一级进位直接依赖于前一级的进位，进位信号逐级形成

12. 先行进位：先行进位即高位进位和低位进位同时产生的进位

13. 对阶：浮点加减运算的第一步，让参与运算的两个浮点数的阶码相同

14. 规格化：规格化又叫做规格化数，是一种表示浮点数的规格化的表示方法，通过修改阶码并同时移动尾数的方法使其满足这种规范。

15. 保留附加位：浮点加减运算时，对阶操作要对阶码小的尾数进行右移，保留附加位即保留右移后的若干位，在舍入处理时丢掉附加位。

3-2

（1）[2009] 一个 C 语言程序在一台 32 位机器上运行，程序中定义了 3 个变量 x、y、z，其中 x 和 z 是 int 型，y 为 short 型。当 $x=127$，$y=-9$ 时，执行赋值语句 $z=x+y$ 后，x、y、z 的值分别是 _____。

A. x = 0000007FH，y = FFF9H，z = 00000076H

B. x = 0000007FH，y = FFF9H，z = FFFF0076H

C. x = 0000007FH，y = FFF7H，z = FFFF0076H

D. x = 0000007FH，y = FFF7H，z = 00000076H

答：D，原因如下：

x=127，补码为：0000 0000 0000 0000 0000 0000 0111 1111，即0000007FH

y=-9，补码为：1111 1111 1111 0111=FFF7H，有符号拓展，前面补1，即FFFFFFF7H，

相加后得00000076H

（2）[2010] 假定有 4 个整数用 8 位补码分别表示 r_1= FEH，r_2= F2H，r_3= 90H，r_4= F8H，若将运算结果存放在一个 8 位的寄存器中，则下列运算会发生溢出的是 _____。

A. $r_1 \times r_2$　　B. $r_2 \times r_3$　　C. $r_1 \times r_4$　　D. $r_2 \times r_4$

答：B，原因如下：

r1=FEH=1111 1110,取反加1，1000 0010，即-2

r2=F2H=1111 0010,取反加1，1000 1110，即-14

r3=90H=1001 0000,取反加1，1111 0000，即-112

r4=F8H=1111 1000,取反加1，1000 1000，即-8

易得r2*r3溢出

3-2

（3）[2013] 某字长为 8 位的计算机中，已知整型变量 x、y 的机器数分别为 $[x]_{补}=11110100$，$[y]_{补}=10110000$。若整型变量 $z=2\times x+y/2$，则 z 的机器数为 _____。

A. 11000000　　B. 00100100　　C. 10101010　　D. 溢出

答：A，原因如下：

z的机器数= 11101000 + 11011000 = 11000000

（4）[2018] 假定带符号整数采用补码表示，若 int 型变量 x 和 y 的机器数分别是 FFFF FFDFH 和 0000 0041H，则 x、y 的值以及 $x-y$ 的机器数分别是 _____。

A. $x=-65$，$y=41$，$x-y$ 的机器数溢出

B. $x=-33$，$y=65$，$x-y$ 的机器数为 FFFF FF9DH

C. $x=-33$，$y=65$，$x-y$ 的机器数为 FFFF FF9EH

D. $x=-65$，$y=41$，$x-y$ 的机器数为 FFFF FF96H

答：C，原因如下：

FFFF FFDFH=1111 1111 1111 1111 1111 1111 1101 1111

取反加1， 1000 0000 0000 0000 0000 0000 0010 0001，即x=-33

0000 0041H=0000 0000 0000 0000 0000 0000 0100 0001 即y=65

-y即-65， 1000 0000 0000 0000 0000 0000 0100 0001

取反加1， 1111 1111 1111 1111 1111 1111 1011 1111， x-y=x+(-y)

得到 1111 1111 1111 1111 1111 1111 1001 1110， 即FFFF FF9EH

3-2

（5）[2018] 整数 x 的机器数为 1101 1000，分别对 x 进行逻辑右移 1 位和算术右移 1 位操作，得到的机器数各是 _____。

A. 1110 1100、1110 1100　　B. 0110 1100、1110 1100

C. 1110 1100、0110 1100　　D. 0110 1100、0110 1100

答：B，原因如下：

逻辑右移1位 0110 1100 算术右移1位 1110 1100

（6）[2009] 浮点数加减运算过程一般包括对阶、尾数运算、规格化、舍入和判断溢出等步骤。设浮点数的阶码和尾数均采用补码表示，且位数分别为 5 位和 7 位（均含 2 位符号位）。若有两个数 $X=2^7\times 29/32$，$Y=2^5\times 5/8$，则用浮点加法计算 $X+Y$ 的最终结果是 _____。

A. 001111100010　　B. 001110100010　　C. 010000010001　　D. 发生溢出

答：D，原因如下：

X补= 00 111，00 11101　　Y补= 00 101，00 11000

对阶　Y补= 00111 00 00110（00）

尾数求和01，00000（00），右规一位，溢出

3-2

（7）[2015] 下列有关浮点数加减运算的叙述中，正确的是_____。

Ⅰ. 对阶操作不会引起阶码上溢或下溢

Ⅱ. 右规和尾数舍入都可能引起阶码上溢

Ⅲ. 左规时可能引起阶码下溢

Ⅳ. 尾数溢出时结果不一定溢出

A. 仅Ⅱ、Ⅲ

B. 仅Ⅰ、Ⅱ、Ⅳ

C. 仅Ⅰ、Ⅲ、Ⅳ

D. Ⅰ、Ⅱ、Ⅲ、Ⅳ

答：D，原因如下：

I正确

II正确：右规时，阶码增大，尾数舍入过程需要右规调整，可能出现阶码上溢

III正确：左规时，尾数增大，阶码减小，有可能下溢

IV：正确，浮点数中以阶码作为溢出判断的标准

3-4-(3)

3.4 已知 x 和 y，用变形补码计算 $x+y$，并判断结果是否溢出。

(1) $x=0.11010$，$y=0.10111$。

(2) $x=0.11101$，$y=-0.10100$。

(3) $x=-0.10111$，$y=-0.11000$。

答：x补= 11.01001 y补= 11.01000
x补+y补= 10.10001
发生溢出，且为负溢出

3-5-(3)

3.5 已知 x 和 y，用变形补码计算 $x-y$，并判断结果是否溢出。

（1）$x=0.11011$，$y=0.11101$。

（2）$x=0.10111$，$y=0.11110$。

（3）$x=-0.11111$，$y=-0.11001$。

答：x补= 11.00001 y补= 11.00111 －y补= 00.11001

x补+(-y补)=11.11010

没有发生溢出

3-6-(2)

3.6 用原码一位乘法计算 $x \times y$。

（1）$x=-0.11111$，$y=0.11101$。

（2）$x=-0.11010$，$y=-0.01011$。

答： 符号位异或，数据位计算，|x补|=0.11010

```
      00.00000 01011
     +00.11010
----------------------
      00.11010 01011
右移  00.01101 00101
     +00.11010
----------------------
      01.00111 00101
右移  00.10011 10010
     +0
----------------------
右移  00.01001 11001
      00.01001 11001
     +00.11010
----------------------
      01.00011 11001
右移  00.10001 11100
     +0
----------------------
右移  00.01000 11110

x * y = 0.01000 11110
```

3-7-(2)

3.7 用补码一位乘法计算 $x \times y$。

（1）$x=0.10110$，$y=-0.00011$。

（2）$x=-0.011010$，$y=-0.011101$。

答： x补=1.100110 y补=1.100011 -x补=0.011010

```
      00.000000 11000110
     +00.011010
----------------------
      00.011010 11000110
右移  00.001101 01100011
     +0
----------------------
右移  00.000110 10110001
     +11.100110
----------------------
      11.101100 10110001
右移  11.110110 01011000
      11.110110 01011000
     +0
----------------------
右移  11.111011 00101100
     +0
----------------------
右移  11.111101 10010110
     +00.011010
----------------------
      00.010111 10010110
右移  00.001011 11001011
     +0
----------------------
      00.001011 11001011
```

x * y = 00.001011 110010

3-8-(2)

3.8 用原码不恢复余数法计算 $x \div y$。

（1）$x=0.10101$，$y=0.11011$。

（2）$x=-0.10101$，$y=0.11000$。

答： |x|=00.10101 |y|=00.11000 -|y|补=11.01000

```
      00.10101 000000
     +11.01000
----------------------
      11.11101 000000
左移  11.11010 00000
     +00.11000
----------------------
      00.10010 000001
左移  01.00100 00001
     +11.01000
----------------------
      00.01100 000011
左移  00.11000 00011
     +11.01000
```

```
      00.00000 000111
左移  00.00000 00111
     +11.01000
----------------------
      11.01000 001110
左移  10.10000 01110
     +00.11000
----------------------
      11.01000 011100
     +00.11000
----------------------
      00.00000 011100
```

综上， x / y ((-21/32)/(3/4))= -0.11100(-7/8)，余数为0

3-9-(2)

3.9 设数的阶码为3位，尾数为6位（均不包括符号位），按机器补码浮点运算规则完成下列 $[x+y]_{补}$ 运算。

（1）$x=2^{011}\times 0.100100$，$y=2^{010}\times(-0.011010)$。

（2）$x=2^{-101}\times(-0.100010)$，$y=2^{-100}\times(-0.010110)$。

答：　　x= 1 011 1.011110　　y= 1 100 1.101010

对阶 x= 1 100 1.101111

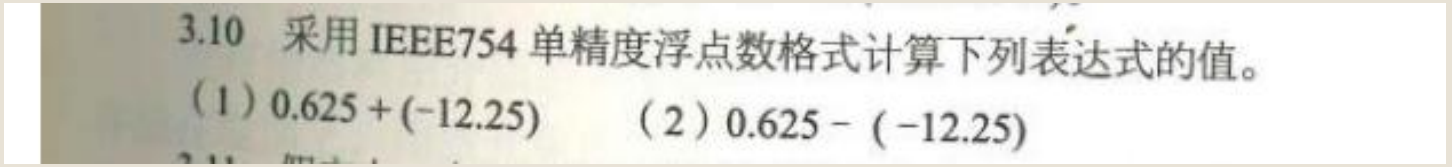
尾数相加　1.101010

+1.101111

1.011001　　尾数已经规格化

[x+y]补=1 100 1.011001

3-10-(2)


3.10 采用 IEEE754 单精度浮点数格式计算下列表达式的值。

（1）0.625 + (-12.25)　　（2）0.625 - (-12.25)

答：　0.625=0.101=1.01*2^(-1)　　对应浮点数 0 0111 1110 010 0000 0000 0000 0000 0000

　　12.25=1100.01=1.10001x2^3 对应浮点数 0 1000 0010 100 0100 0000 0000 0000 0000

　　对阶 0 1000 0010 000 1010 0000 0000 0000 0000

即　0.100 0100 0000 0000 0000 0000

　　+0.000 1010 0000 0000 0000 0000

　　0.100 1110 0000 0000 0000 0000

即　0 1000 0010 100 1110 0000 0000 0000 0000 对应真值 1.100111*2^3 = 12.875

3-11

3.11 假定在一个 8 位字长的计算机中运行如下 C 语言程序段。

```
unsigned int x=134;
unsigned int y=246;
int m=x; int n=y;
unsigned int z1=x-y;
unsigned int z2=x+y;
int k1=m-n;
int k2=m+n;
```

若编译器编译时将 8 个 8 位寄存器 R1 ～ R8 分别分配给变量 x、y、m、n、$z1$、$z2$、$k1$ 和 $k2$。请回答下列问题（提示：带符号整数用补码表示）。

（1）执行上述程序段后，寄存器 R1、R5 和 R6 中的内容分别是什么？（用十六进制表示）

答：

x = 1000 0110	R1 = 1000 0110 = 86H
y = 1111 0110	R2 = 1111 0110 = F6H
m = 1000 0110	R3 = 86H
n = 1111 0110	R4 = F6H
z1 = 1001 0000	R5 = 90H
z2 = 0111 1100	R6 = 7CH
k1 = 1001 0000	R7 = 90H
k2 = 0111 1100	R8 = 7CH

3-11

3.11 假定在一个 8 位字长的计算机中运行如下 C 语言程序段。

```
unsigned int x=134;
unsigned int y=246;
int m=x; int n=y;
unsigned int z1=x-y;
unsigned int z2=x+y;
int k1=m-n;
int k2=m+n;
```

若编译器编译时将 8 个 8 位寄存器 R1 ～ R8 分别分配给变量 x、y、m、n、$z1$、$z2$、$k1$ 和 $k2$。请回答下列问题（提示：带符号整数用补码表示）。

（1）执行上述程序段后，寄存器 R1、R5 和 R6 中的内容分别是什么？（用十六进制表示）

（2）执行上述程序段后，变量 m 和 $k1$ 的值分别是多少？（用十进制表示）

答:(1) x = 1000 0110　　R1 = 1000 0110 = 86H　　(2) m = -122
y = 1111 0110　　R2 = 1111 0110 = F6H　　k1 = -112
m = 1000 0110　　R3 = 86H
n = 1111 0110　　R4 = F6H
z1 = 1001 0000　　R5 = 90H
z2 = 0111 1100　　R6 = 7CH
k1 = 1001 0000　　R7 = 90H
k2 = 0111 1100　　R8 = 7CH

3-11

（3）上述程序段涉及带符号整数加减、无符号整数加减运算，这 4 种运算能否利用同一个加法器及辅助电路实现？简述理由。

（4）计算机内部如何判断带符号整数加减运算的结果是否发生溢出？上述程序段中，哪些带符号整数运算语句的执行结果会发生溢出？

答：(3) 可以，可控加减法器既可以完成无符号数加减运算，也可以完成有符号数加减运算

(4) 判断带符号整数加减运算有3种方法：

1. 根据运算过程中最高数据位的进位与符号位的进位是否一致进行检测
2. 根据操作数和运算结果的符号位是否一致进行检测
3. 利用变形补码的符号位进行检测

上述程序中“int k2 = m + n”会发生溢出，因为运算结果=-132，超出了8位二进制数补码的表示范围-128～+127。