计算机系统基础

数的表示及运算（2）

王晶

jwang@ruc.edu.cn，信息楼310

2022年10月

题目1

- 写一个C表达式，在下列描述的条件下产生1，其他情况产生0，假设X是int类型。代码中不能使用==或！=进行测试。
  - ✓x的任何位都等于1；
  - ✓x的任何位都等于0；
  - ✓x的最低有效字节中的位都等于1；
  - ✓x的最高有效字节中的位都等于1；

!(~x & 0xFF000000)

题目2

- int为32位，float和double分别是32位和64位IEEE格式
  - Int x =random();
  - Int y = random();
  - Int z = random();
  - Double dx = (double)x;
  - Double dy = (double)y;
  - Double dz = (double)z;
- 对于下面的每个C表达式，判断是否恒为1。如果是请说明原理，如果不是请举出反例。
  - A. (float)x == (float)dx
  - B. dx-dy == (double)(x-y)
  - C. (dx+dy)+dz == dx+(dy+dz)
  - D. (dx*dy)*dz == dx*(dy*dz)
  - E. dx/dx == dz/dz

Mathematical Properties of FP Add

- 是否符合阿贝尔群的特征
  - ✓ 加法的封闭性? — ***Yes***
    - 但是有可能产生无穷或NaN
  - ✓ 交换律? — ***Yes***
  - ✓ 结合律? — ***No***
    - Overflow and inexactness of rounding
    - `(3.14+1e10)-1e10 = 0, 3.14+(1e10-1e10) = 3.14`
  - ✓ 0 是加法单位元? — ***Yes***
  - ✓ 每个元素都有加法逆元? — ***Almost***
    - Yes, 除了infinities & NaNs
- 单调性
  - ✓ $a \geq b \Rightarrow a+c \geq b+c$? — ***Almost***
    - 除了infinities & NaNs

Mathematical Properties of FP Mult

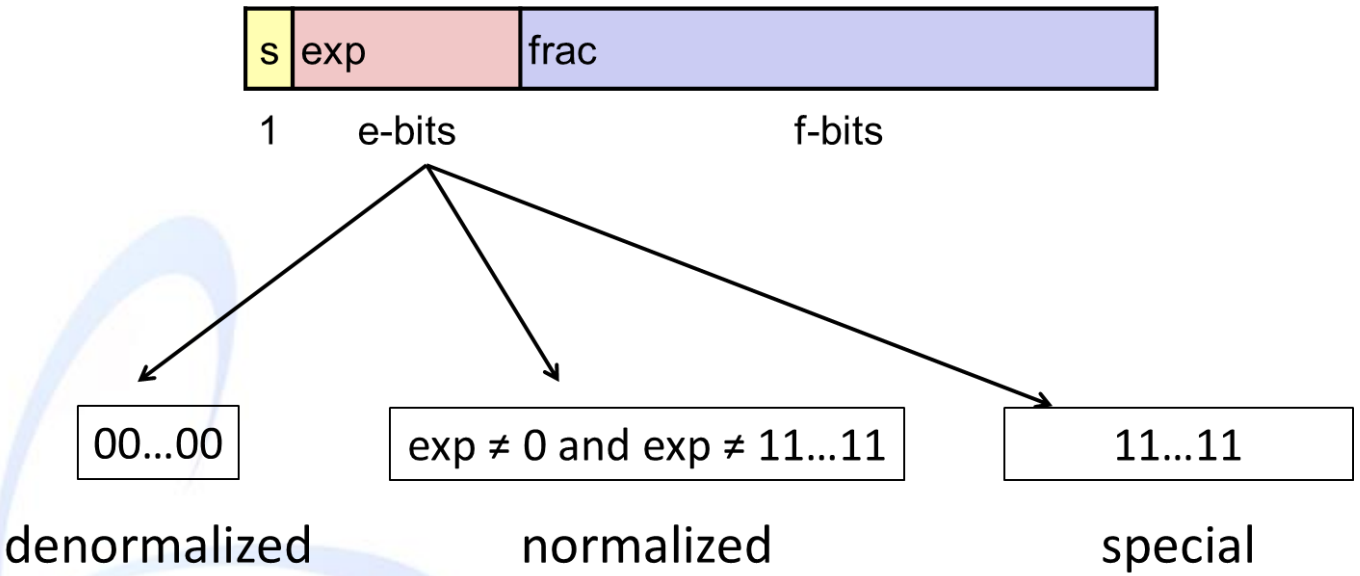
- 属性
  - ✓ 乘法是否封闭？ ***Yes***
    - 但可能产生 infinity or NaN
  - ✓ 交换律? ***Yes***
  - ✓ 结合率? ***No***
    - 可能导致溢出，或者由于舍入带来的不精确
    - Possibility of overflow, inexactness of rounding
    - Ex: `(1e20*1e20)*1e-20= inf, 1e20*(1e20*1e-20)= 1e20`
  - ✓ 1是乘法的单位元？ ***Yes***
  - ✓ 乘法对加法的分配率？ ***No***
    - 可能导致溢出，或者由于舍入带来的不精确
    - `1e20*(1e20-1e20)= 0.0, 1e20*1e20 – 1e20*1e20 = NaN`
- 单调性 ***Almost***
  - ✓ $a \geq b \ \& \ c \geq 0 \Rightarrow a * c \geq b * c$?
    - 除了infinities & NaNs

题目2

- int为32位，float和double分别是32位和64位IEEE格式
  - ✓ Int x =random();
  - ✓ Int y = random();
  - ✓ Int z = random();
  - ✓ Double dx = (double)x;
  - ✓ Double dy = (double)y;
  - ✓ Double dz = (double)z;
- 对于下面的每个C表达式，判断是否恒为1。如果是请说明原理，如果不是请举出反例。
  - ✓ A. (float)x == (float)dx ✓
  - ✓ B. dx-dy == (double)(x-y) ✗ x=0，y=Tmin
  - ✓ C. (dx+dy)+dz == dx+(dy+dz) ✓ `(3.14+1e10)-1e10 = 0, 3.14+(1e10-1e10) = 3.14`
  - ✓ D. (dx*dy)*dz == dx*(dy*dz) ✗
  - ✓ E. dx/dx == dz/dz ✗

题目3

- 编写如下函数，求浮点数f的绝对值|f|。如果f是NaN，那么应该直接返回f（注意NaN不要对f做任何修改）。
- 其中float_bits等价于unsigned，是float数字的二进制形式
  - ✓ typedef unsigned float_bits;
- /* Compute |f|. If f is NaN, then return f. */
- float_bits float_absval (float_bits f);



NaN: exp = 111…1, frac ≠ 000…0

题目3

```
float_bits float_absval (float_bits f) {
    // 读取尾数和阶码
    Unsigned exp = f>>23 & 0xFF;
    Unsigned frac = f & 0x7FFFFF;

    // 判断是否为NaN
    If (exp == 0xFF && frac != 0)
        Return f;

    // 符号位清0
    Unsigned mask = 1<<31;
    Unsigned abs = f & ~mask;

    Return abs;
    }
```

题目4

- 实现如下函数，对于浮点数f，计算2.0*f。如果f是NaN，你的函数应该简单返回f。
- /* Compute 2*f. If f is NaN, return f. */
- float_bits float_twice(float_bits f);

题目4答案

```
Float_bits float_twice (float_bits f) {
    Unsigned sign = f>>31;
    Unsigned exp = f>>23 & 0xFF;
    Unsigned frac = f & 0x7FFFFF;
    If (exp == 0) {                          //非规格化浮点数
        Frac = 2*frac;
        If (frac > 0x7FFFFF) {               // 变为规格化浮点数，多了一个隐含1
            Frac = frac&0x7FFFFF;            // 隐含1
            Exp = 1;                         //阶码变1
            }
    }
    else if (exp < 0xFF) {                   //规格化浮点数
        exp++;
        If (exp == 0xFF) {
            Frac = 0;              // 转为∞
        }
    }else if (frac != 0} {                   // NaN
        Return f;
    }
    Return (sign<<31} | (exp<<23) | frac;
}
```

从二进制表示拆分符号位、阶码和尾数

计算机系统基础

程序的机器级表示

王晶

jwang@ruc.edu.cn，信息楼310

2022年10月

Summarizing

- C Control
  - ✓ if-then-else
  - ✓ do-while
  - ✓ while, for
  - ✓ switch
- Assembler Control
  - ✓ Conditional jump
  - ✓ Conditional move
  - ✓ Indirect jump (via jump tables)
- Standard Techniques
  - ✓ Loops converted to do-while or jump-to-middle form
  - ✓ Large switch statements use jump tables
  - ✓ Sparse switch statements may use decision trees (if-elseif-elseif-else)

课堂练习1

- int comp (data_t a, data_t b) {
  - return a COMP b;
- }
- 64位机环境下，对于下面每个汇编指令序列，确定哪种数据类型data_t和比较操作COMP会导致编译器产生如下代码：
  - A. cmpl %esi, %edi; setl %al int, <
  - B. cmpw %si, %di; setge %al short, >=
  - C. cmpb %sil, %dil; setbe %al unsigned char, <=
  - D. cmpq %rsi, %rdi; setne %al long, unsigned long, 或指针，!=

课堂练习2

填写C语言代码：

```
Long test (long x, long y) {
    long val = ____8x____;
    if (___y>0____) {
        if (___x<y____) {
            val = ____y-x_____;
        else
            val = ____y&x_____;
    } else if (____y<=-2____)
        val = ____x+y____;
    return val;
}
```

```
# x @ %rdi, y @ %rsi
Test:
  leaq    0(,%rdi,8), %rax
  testq   %rsi, %rsi
  jle     .L2
  movq    %rsi, %rax
  subq    %rdi, %rax
  movq    %rdi, %rdx
  andq    %rsi, %rdx
  cmpq    %rsi, %rdi
  cmovge %rdx, %rax
.L2:
  addq    %rsi, %rdi
  cmpq    $-2, %rsi
  cmovle %rdi, %rax
  ret
```


14

练习3

程序填空，变量映射关系，
并解释函数功能

```
Int fun_a(unsigned x) {
  int val = 0;
  while (    x     ) {
    val ^= x;
    x>>=1
  }
  return val&0x01;
}
```

- x@$ebp+8

```
  Movl 8(%ebp), %edx
  Movl $0, %eax
  Testl %edx, %edx
  Je .L7
.L10:
  xorl %edx, %eax
  shrl %edx  ;逻辑右移1位
  jne .L10
.L7:
  andl $1, %eax
```

课堂练习4

- A. 根据汇编代码，填写C代码缺失的部分；
- B. 解释循环前为什么没有初始测试，也没有初始跳转到循环内部的测试部分；
- C. 用自然语言描述这个函数的功能

```
fun_b(unsigned long x) {
        Long val = 0;
        Long i;
        For (… ; … ; …) {
                …
        }
        Return val;
}
```

```
# x in %rdi
1 func_b:
2     Movl $64, %edx
3     movl $0, %eax
4   .L10:
5     movq %rdi, %rcx
6     andl  $1, %ecx
7     addq %rax, %rax
8     orq $rcx, %rax
9     shrq %rdi
10    subq $1, %rdx
11    jne .L10
12    ret
```

练习4答案

```
fun_b(unsigned long x) {
        long val = 0;
        long i;
        for (i=64; i!=0 ; i--) {
            Val = (val << 1) | (x&1)
             X >>= 1;
        }
        Return val;
    }
```

```
# x in %rdi
1 func_b:
2     Movl $64, %edx
3     movl $0, %eax
4   .L10:
5     movq %rdi, %rcx
6     andl  $1, %ecx
7     addq %rax, %rax
8     orq $rcx, %rax
9     shrq %rdi
10    subq $1, %rdx
11    jne .L10
12    ret
```

赋初值

循环

更新和判断

- 2）因为循环次数是常量
- 3）将x镜像反转

17

课堂练习5（3.31）

```
long switcher(long a, long b,
long c, long *dest)
{
    long val;
    switch(a) {
    case ____ :
        c = ____;
        /* Fall Through */
    case ____ :
    val = ____;
    break;
    case ____ :
    case ____ :
    val = ____;
    break;
    case ____ :
    val = ____;
    break;
    default:
    val = ____;
}
    *dest=val
}
```

```
Switcher:
   cmpq   $7, %rdi
   ja     .L2
   jmp    *.L4(,%rdi,8)
   .section          .rodata
.L7:
   xorq   $15, %ris
   movq %rsi, %rdi
.L3:
   leaq   112(%rdx), %rdi
   jmp    .L6
.L5:
   leaq    (%rdx, %rsi),%rdi
   salq   $2, %rdi
   jmp    .L6
.L2:
   movq     %rsi, %rdi
.L6:
   movq     %rsi,  (%rcx)
   ret
```

```
.L4:
  .quad     .L3
  .quad     .L2
  .quad     .L5
  .quad     .L2
  .quad     .L6
  .quad     .L7
  .quad     .L2
  .quad     .L5
```

练习5答案

```
long switcher(long a, long b,
long c, long *dest)
{
    long val;
    switch(a) {
    case ____ :
        c = ____;
        /* Fall Through */
    case ____ :
    val = ____;
    break;
    case ____ :
    case ____ :
    val = ____;
    break;
    case ____ :
    val = ____;
    break;
    default:
    val = ____;
}
    *dest=val
}
```

3

```
Switcher:
   cmpq   $7, %rdi
   ja     .L2
   jmp    *.L4(,%rdi,8)
   .section          .rodata
.L7:                          # Case 5
   xorq   $15, %ris
   movq %rsi, %rdi
.L3:                           # Case 0
   leaq   112(%rdx), %rdi
   jmp    .L6
.L5:                          # Case 2/7
   leaq    (%rdx, %rsi),%rdi
   salq   $2, %rdi
   jmp    .L6
.L2:                          # Case 1/6
   movq     %rsi, %rdi
.L6:                          # Case 4
   movq     %rsi,  (%rcx)
   ret
```

2

```
.L4:
  .quad     .L3   # a = 0
  .quad     .L2   # a = 1
  .quad     .L5   # a = 2
  .quad     .L2   # a = 3
  .quad     .L6   # a = 4
  .quad     .L7   # a = 5
  .quad     .L2   # a = 6
  .quad     .L5   # a = 7
```

1