作业-2章

9. (1) int x = -32768 ~~0xffffff80~~ 0xffff8000

(2) short y=522 0x020a

(3) unsigned z=65530 ~~0x0000fff9~~ ~~0x0000fffa~~ 0x0000fffa

(4) char c='@' 0x0040

(5) float a=-1.1 0xbf8ccccd

(6) double b=10.5 0x4025 0000 0000 0000

10. (1) int x: FFFF0006H x=65526

(2) short y: DFFCH y=-8196

(3) unsigned z: FFFFFFFAH z=65530

(4) char c: 2AH c=42

(5) float a: C448 0000H a=-800

(6) double b: C024 8000 0000 0000 b=-10.25

11. (1) char *mystring1: 68H 65H 6CH 6CH 6FH 2CH 77H 6FH 72H 6CH 64H 0AH 00H

hello, world

(2) char *string2: 77H 65H 20H 61H 72H 65H 20H 68H 61H 70H 70H 79H 21H 00H

we are happy!

12. (1) char *mystring1 = "./myfile"

0x2e 0x2f 0x6d 0x79 0x66 0x69 0x6c 0x65

(2) char *mystring2 = "OK, good!"

0x4f 0x4b 0x2c 0x67 0x6f 0x6f 0x64

18. strlen(str2) > strlen(str1) 时结果不正确

strlen 返回类型为 unsigned, 始终为大于等于0的值

与0比较时

当 str2 较长, strlen(str1) - strlen(str2) 为 TRU

出现错误。

21. $M = 2^4 - 1 = 15$, $N = 2^2 = 4$

22. (1) 0~65535 ~~(2) $-2^{15} \sim 2$~~ (2) $2^{-15} - 1 \sim 1 - 2^{-15}$

(3) -32768~32767 (4) -32768~32767

(5) 最大正数: $(1-2^{-7}) \cdot 2^{127}$ 最小正数: 2^{-135}

最大负数: -2^{-135} 最小负数: $(1-2^{-7}) \cdot 2^{127}$

23. (1) +1.75 = 1.10 1 1B S=0, e=127+0=127, f=110 0---0B

+1.75 = 0x3fd00000

(2) +19 = 10011B. S=0, e=127+4=131, f=0011 0-0B

~~+19: 0x8f980000~~ 0x41980000

(3) $-\frac{1}{8}$ = -0.001B, S=1, e=127-3=124, f=0

~~$-\frac{1}{8}$: 0xbd000000~~ $-\frac{1}{8}$ = 0xbe000000

(4) 258 = 100000010B, S=0, e=127+8=135, f=00000010-0B

258: 0x43810000

24. 4098 = 0x00001002 = 0000 0000 0000 0000 0001 0000 0000 0010

= 0x45801000 ~~= 01000100~~

= 0100 0101 1000 0000 0010 0000 0000 0000

框中两部分相同，因为 e-bias=12；~~小数点后12位即~~ 尾数前12位与整数末12位相同。

28. 大端

0x100	0x101	0x102	0x103	……	0x108	0x109	0x10a	0x10b
0xbe	0x0	0x00	0x00		0x40	0x00	0x50	0x00

0x

28. 大端 100H 101H 102H 103H 104H 105H
内容 beH 00H 00H 00H 00H 00H
106H 107H 108H 109H 10aH 10bH 10cH 10dH
08H 00H 40H f0H 00H 00H \ \
10eH 10fH 110H 111H 112H 113H 114H 115H
\ \ \ \ 00H 64H \ 64H
小端 100H 101H 102H 103H ··· 108H 109H 10aH 10bH
内容 00H 00H 00H beH 00H 00H f0H 40H
··· 112H 113H
64H 00H

29. 表示. X x Y y X+Y x+y OF SF CF X-Y x-y OF SF 0
无符号 0xB0 176 0x8C 140 0x3C 60 1 0 1 0x24 36 0 0 0
带符号 0xB0 -80 0x8C -116 0x3C 60 1 0 1 0x24 36 0 0 0
无符号 0x7E 126 0x5D 93 0xDB 219 1 1 0 0x21 33 0 0 0
带符号 0x7E 126 0x5D 93 0xDB -37 1 1 0 0x21 33 0 0 0

31. 不能.

改为 unsigned long long arraysize = count * (unsigned long long)
if (arraysize > INT_MAX) sizeof(int);
return -1;
int *myarray = (int*) malloc(arraysize);

34. (2) (x*x) >= 0 x = 2^{17}-1 为假

(2) (x-1<0) || x>0 x = INT_MIN时为假

(3) x<0 || -x<=0 永真。否则, x>=0 且 -x>0, 满足x>=0时由于负数在INT范围内多于正数, -x也为负数或0不成立.

(4) x>0 || -x>=0 x = INT_MIN时为假

(5) x&0xf != 15 || (x<<28)<0, != 优先级高, x=0x0f为假

(6) x>y == (-x<-y) x = INT_MIN, y=0 为假

(7) ~x+~y == ~(x+y) 左 = -x-y-2, 右 = -x-y-1 恒假

(8) (int)(ux-uy) == -(y-x) 永真。~~(int)(ux-uy) = [ux-uy]补~~
(int)(ux-uy) = [x-y]补 = [x]补 + [-y]补 = [-y+x]补 = [-(y-x)]补

~~(9) ((x>>2)<<2)<<x~~

(9) ((x>>2)<<2) <= x ~~假设 x = 0x8~~

永真。末2位为正权, 消除仅导致减小.

(10) (x*4)+(y*8) = (x<<2)+(y<<3)

永真。等价 (*2^k, <<k)

(11) x/4 + y/8 == (x>>2)+(y>>3)　x=-1, y=-1 时假

(12) x*y == ux*uy 永真。x*y与ux*uy低32位相同

(13) x+y == ux+uy 永真。x+y与ux+uy机器数运算规则均相同。

(14) x*~y + ux*uy == -x 永真。左式: x*(-y-1)+x*y
= -x

35. (1) dx*dx >=0 永真。符号位一定为0，单独计算得。

(2) (double)(float)x == dx　x=0xffffffff 假

(3) dx+dy=(double)x+y ; x=y=0x ffffffff 假

(4) (dx+dy)+dz == dx+(dy+dz) 永真。double可精确表示int，且对阶时阶码不足52位。

(5) dx*dy*dz == dz*dx*dy 非永真，过程中可能会

(6) dx/dx == dy/dy　dx=0, dx/dx=NaN, 假

41. (1)(2)(3)(4)(5)(6)

```
#include <stdio.h>
typedef unsigned float_bits;

float_bits float_abs(float_bits f){
    if((f>>23 & 0xff)== 0xff && (f & 0x7fffffff))
        return f;
    ~~if(f>>31)~~
    return f & 0x7fffffff;
}

float_bits float_neg(float_bits f){
    if((f>>23 & 0xff)==0xff && (f & 0x7fffff))
        return f;
    return f ^ 0x80000000;
}

~~float_bits float_neg(float_bits f){~~
    ~~if((f>>23 & 0xff)==0xff && (f & 0x7fffff))~~
        ~~return f;~~
    ~~if(f>>31)~~
```

~~return f+1~~

```
float_bits float_half(float_bits f){
    unsigned sign=f>>31;
    unsigned exp=f>>23 & 0xff;
    unsigned frac = f & 0x7fffff;
    if ((exp=0xff) && frac)
        return f;
    if((exp=0xff) && frac)
        return f;
    if((exp==0) || (exp==0xff))
        return f;
    if((exp==0) && frac)
        return sign<<31 | frac>>1;
    exp=(exp-0xff) & 0xff;
    if(exp)
        return sign<<31 | exp<<23 | frac;
    return sign<<31 | (frac | 0x800000)>>1
```

```
float_bits float_twice(float_bits f){
    unsigned sign = f>>31;
    unsigned exp = f>>23 & 0xff;
    unsigned frac = f & 0x7fffff;
    if ((exp==0xff) && frac)
        return f;
    if(( (exp==0) || (exp == 0xff)) && (frac==0))
        return f;
    if ((exp==0) && (frac)){
        if(frac & 0x400000){
            return sign << 31 | exp<<23 | (frac & 0x3fffff)<<1;
        else
            return sign | frac <<1;
    }else{
        exp = exp+1;
        if(exp != 0xff)
            return sign<<31 | exp<<23|frac;
        else
            return sign<<31 |exp<<23;
    }
}
```

```
float_bits float_i2f(int i){
    if(i==0) return 0;
    else if(i==0x80000000) return 0xcf000000;
    int sign= i>>31 & 0x1;Bias=127; left_most_one=0;
    if(sign==1) i=~i+1;
    for(int j=30; j>=0; --j){
        if(1<<j&i){
            left_most_one=j;
            break;
        }
    }
    int E=left_most_one, exp=E+Bas;
    unsigned frac=0;
    if(E<=23){
        frac = ((1<<E)-1)&i;
        return (sign<<31)|(exp<<23)|frac<<(23-E);
    }else{
        int mov=E-23;
        frac=(i>>mov)&0x7fffff;
        int round_1=((i&((1<<mov)-1))>(1<<(mov-1));
        int round_2=(i&((1<<mov)|(1<<(mov-1))))==
        ((1<<mov)|(1<<mov-1)));
```

```
if (round_1 || round_2){
    frac++;
}
if (frac & 0x80000000){
    frac = frac & 0x7fffff;
    exp++;
}
}
return (sign<<31) | (exp<<23) | frac;
}
```

~~float_bits f~~

```
int float_f2i(float_bits f){
    int S = uf>>31;
    int E = uf>>23 & 0xff;
    int frac = uf && 0x7fffff;
    if (E>158){
        return 0x80000000u;
    }else if(E<127){
        return 0;
    }else{
```

```
frac = frac | 0x800000;
if(E>150){
        frac = frac<<(E-150);
}else{
        frac = frac>>(150-E);
}
if(S){
        frac = ~frac+1;
}
return frac;
}
```