2.88

格式 A		格式 B	
位	值	位	值
1 01110 001	-9/16	1 0110 0010	-9/16
0 10110 101	208	0 1110 1010	208
1 00111 110	$-7/2^{10}$	1 0000 0111	$-7/2^{10}$
0 00000 101	$5/2^{17}$	0 0000 0000	0
1 11011 000	-4096	1 1111 0000	-∞
0 11000 100	768	0 1111 0000	+∞

(如果按照最接近的格式 B 表示的值第四行应该为零,第五行应为负无穷,但如果向正无穷舍入(也就是向上舍入)的话,第四行结果可能是 0 0000 0001 值为 $1/2^{10}$,第五行可能是最小的规格化数(1 1110 1111))

2.92

```
float_bits float_negate(float_bits f){
      unsigned int sign_bit = (f >> 31) & 1;
      unsigned int exp = (f >> 23) & 0xFF;
      unsigned int frac = f & ((1 << 23) - 1);

       if(exp == 0xFF && frac){
            return f;
       }else{
            return (!sign_bit << 31) | (exp << 23) | frac;
       }
}
```

2.96

```
int float_f2i(float_bits f){
      unsigned int sign_bit = (f >> 31) & 1;
      unsigned int exp = (f >> 23) & 0xFF;
      unsigned int frac = f & ((1 << 23) - 1);
      int bias = (1 << 7) - 1;
      int ans = 0;

       if(exp < bias){
            ans = 0;
       }else if(exp - bias >= 31){//int 不能表示或者 Tmin
            return 0x80000000;
```

```
        }else{
                frac += 1 << 23;//m = 1 + f
                exp -= bias;//exp = e - bias
                if(exp >= 23){
                        frac <<= (exp - 23);
                }else{
                        frac >>= (23 - exp);
                }
                ans = frac;
        }

        if(sign_bit > 0){//是负数
                return -ans;
        }else{
                return ans;
        }
}
```