第二章课后习题

2.55

运行结果如下：

又 int、float 运行结果可知，此机器为小端序。

2.59

设最初两个数为 x、y，生成的数 z 的最低有效位来自 x，其余位来自 y，则有

z = (x & 0x000000FF) || (y & 0xFFFFFF00)

2.63

```c
#include <stdio.h>

int w;
unsigned srl(unsigned x,int k){
    /* Perform shift arithmettically */
    unsigned xsra = (int) x >> k;
    int le_srl = 0;
    for(int i = 0;i <= 31-k;i++){
        le_srl <<= 1;
        le_srl += 1;
    }
    xsra = xsra & (unsigned)le_srl;
    return xsra;
}

int sra(int x,int k){
    int xsrl = (unsigned) x >> k;
    int le = 0;
    unsigned le_sra;
    if(x < 0){
        for(int i = 0;i <= 31-k;i++){
            le *= 2;
            le += 1;
        }
        le_sra = (unsigned)le ^ 0xFFFFFFFF;
        xsrl = (int)le_sra ^ xsrl;
    }
    return xsrl;
}

int main(){
```

```
32.
33.     w = 8 * sizeof(int);
34.     printf("int 的位数为%d\n",w);
35.
36.     unsigned x = 0x80000002;
37.     unsigned y = srl(x,1);
38.     printf("y = %x\n",y);
39.
40.     int a = 0x80000002;
41.     int b = sra(a,1);
42.     printf("b = %x\n",b);
43.
44.     return 0;
45. }
```

2.67

A. int 占 4 个字节，共 32 位。在

Int beyond_msb = 1 << 32;

中，移位数量超出了字长，在 sun spark 上并不会进行取模操作，C 语言对此也没有特别规定，故产生警告。

B.

```
1.  int bad_int_size_is_32(){
2.      int i = 1 ;
3.      int set_msb = 1 << 31;
4.      int beyond_msb = 1 << 31 << 1;
5.      printf("set_msb = %d\n",set_msb);
6.      printf("beyond_msb = %.8x \n",beyond_msb);
7.      return set_msb && !beyond_msb;
8.  }
```

C.

```
1.  int bad_int_size_is_32(){
2.      int i = 1;
3.      int set_msb = 1 << 15 << 15 << 1;
4.      int beyond_msb = 1 << 15 << 15 << 2;
5.      printf("set_msb = %d\n",set_msb);
6.      printf("beyond_msb = %.8x \n",beyond_msb);
7.      return set_msb && 7!beyond_msb;
8.  }
```

2.71

（1）如果抽出的是负数，这段代码返回的是无符号数。

（2）

```
int xbyte(packed_t word,int bytenum){
    return ((int)word <<(24-(bytenum << 3)))>>24;
}
```

2.75

```
#include <stdio.h>

int w;

unsigned unsigned_high_prod(unsigned x,unsigned y){
    int W = signed_high_prod(x,y);
    int sighed_x = x >> (w-1);
    int sighed_y = y >> (w-1);
    return W + x * sighed_y + y * sighed_x;
}


int main(){
    int x = 0x12345678;
    int y = 0x12345678;
    w = 32;
    printf("%d\n",unsigned_high_prod(x,y));
    return 0;
}
```

2.79

```
#include <stdio.h>

int threefourths(int x){
    int x_div2 = x >> 1;
    int x_div4 = x >> 2;
    int x_mul2 = x << 1;
    int x_last = ((x * 0x0011) + (x_mul2 * 0x0011))>>2;
    return x_div2 + x_div4 + x_last;
}


int main(){
    int x = 0x7845678;
    printf("%d\n",threefourths(x));
    return 0;
}
```

2.83

(1) x = Y/(2 **k - 1)

(2) (a) 5/7

(b) 2/5

(c) 19/63

2.87

描述	Hex	M	E	V	D
-0	**0x8000**	**0**	**-14**	-0	-0.0
最小的>2 的值	**0x4001**	**1025/1024**	**1**	**1025*2^(-9)**	**2.001953**
512	**0x6000**	**1**	**9**	512	512.0
最大的非规格化束	**0x03FF**	**1023/1024**	**-14**	**1023*2^(-24)**	**0.000061**
-∞	**0xFC00**	---	---	-∞	-∞
3BB0	3BB0	**123/64**	**-1**	**123*2^(-7)**	**0.960938**

2.91

(1) 3.141593

(2) 11.001001001……

(3) 第 9 位

2.95

```
typedef unsigned float_bits;
float_bits float_half(float_bits f){
    unsigned sign = f & 0x80000000;
    unsigned exp = (f>>23) & 0xff;
    unsigned flac = f & 0x7fffff;
    unsigned flag = f & 0x03;
    unsigned usd_f = f & 0x7fffffff;
    if(exp == 0xff){
        return f;
    }else if(exp == 0)
    {
        /* code */
        flac = flac >> 1;
        if(flag == 0x3){
            flac += 1;
        }
    }else if(exp == 1){
        usd_f >> 1;
        if(flag == 0x3){
            usd_f += 1;
        }
        exp = usd_f >> 23;
        flac = usd_f & 0x7fffff;
```

```
    }else{
        exp = exp - 1;
    }
    return sign << 31 | exp << 23 | flac;
}
```