COMP1411 (Spring 2023) Introduction to Computer Systems

Individual Assignment 1 Duration: 00:00, 11-Feb-2022 ~ 23:59, 12-Feb-2022

Name	WANG Ruijie
Student number	22103808d

Question 1. [0.5 marks]

Suppose that `x` and `y` are unsigned integers.

Rewrite the following C-language statement by using `<<` and `-`.

```
y = x * 137;
```

Introducing new variables (other than `x` and `y`) is not allowed.

Show your steps. Only giving the final result will NOT get a full mark of this question.

Answer:

Knowing that 137 = 256 - 64 - 32 - 16 - 4 - 2 - 1 and u << k gives u * 2^k,

The statement can be rewritten as y = (x << 8) - (x << 6) - (x << 5) - (x << 4) - (x << 2) - (x << 1) - x.

Question 2. [1.5 mark]

Suppose that a, b, c and z are all 32-bit unsigned integers.

(1) Assume that the left-most bit is the highest bit. Write C-language statements to set the value of z, such that:
   a. the left-most 11 bits of z are the same as the left-most 11 bits of a;
   b. the right-most 14 bits of z are the same as the right-most 14 bits of b;
   c. the middle 7 bits of z are the same as the middle 7 bits of c.

Note that:

- You are only allowed to use bit shift operations and logic operations (including bit-wise operators, such as | ^ &) to set the value of z;
- NO arithmetic or if-then-else test (in any form) is allowed;

(2) Write another C statement to set the value of z to the same results in (1), but do NOT use masks.

(3) If a = 0xC9E3BA75, b = 0x268DBA83, and c = 0x63ABE432, what the be the resulting value of z? Please write the value of z in hex-decimal form starting with prefix 0x.

Show your steps. Only giving the final result will NOT get a full mark of this question.

Answer:

(1) a = a & 11111111111000000000000000000000 (11 bits of 1 and 21 bits of 0);
b = b & 00000000000000000011111111111111 (18 bits of 0 and 14 bits of 1);
c = c & 00000000000111111100000000000000 (11 bits of 0, 7 bits of 1 and 14 bits of 0);
z = a | b | c;

(2) a = a >> 21 << 21;
b = b << 18 >> 18;
c = c >> 14 << 25 >> 11;
z = a | b | c;

(3) (i) First, convert the hexadecimal numbers a, b and c to binary numbers 11001001111000111011101001110101, 10011010001101101110101000011 and 1100011101010111110010000110010.
(ii) Then, according to the description, the binary form of z should be 11001001111101111110010000110010.
(iii) Finally, convert z to its hexadecimal format: 0xC9F7E432.

Question 3. [1.5 marks]

Assume on a little-endian machine, a 32-bit single-precision floating-point number is stored in the addresses 0x0100 ~ 0x0103 is as follows:

Address	Byte in the Address
0x0100	0x3F
0x0101	0x02
0x0102	0x94
0x0103	0xC1

Convert the above floating-point number to a decimal number.

For the converted decimal number, leave only 3 digits after the decimal point and discard all the rest digits; DO NOT write the result in the exponential form of the power of 2 or 10.

Show your steps. Only giving the final result will NOT get a full mark of this question.

Answer:

Considering the endianness, the hexadecimal format of the number should be 0xC194023F, and convert it to binary 11000001100101000000001000111111. According to the IEEE Std 754, the sign bit is 1, the exponent is 10000011_2 which is equal to 131_{10}, and the fraction is $00101000000001000111111_2$. Hence, the real exponent in decimal format is 131 – 127 = 4, and the significand number is $1.00101000000001000111111_2$ whose decimal format is 1.1563185453414917. Therefore, the converted decimal result is 1.1563185453414917 * 2^4 which can be approximated to -18.501.

Question 4. [1.5 marks]

Consider a 12-bit floating-point representation based on the IEEE floating-point format:

- the highest bit is used for the sign bit,
- the sign bit is followed by 5 exponent bits, which are then
- followed by 6 fraction bits.

Question 1: What is the largest positive normalized number? Write the numbers in both the binary form and the decimal value.

Question 2: **Convert** the decimal number 23.875 into the above 12-bit IEEE floating-point format. Write the result in the binary form.

Show your steps for both Question 1 and Question 2. Only giving the final result will NOT get a full mark of this question.

Answer:

(1) Considering the number of exponent bits is 5, the bias number should be $2^{5-1} - 1 = 15$. For the exponent, Exp = E + Bias, and the largest Exp is 11111_2 that is equal to 31_{10}. Hence the largest number for the real exponent is 31 – 15 = 16. Since there are 6 fraction bits, the largest significand number is 1.111111_2. Therefore, the largest positive normalized number is $1.111111 * 2^{16}$ in binary form and $1.984375 * 2^{16}$ in decimal form.

(2) First, convert the decimal number 23.875 to its base-2 representation 10111.111. Then, adjust it to have the leading one: $10111.111 = 1.01111110 * 2^4$. Hence the exponent is 4, and the GRS is 110 where increment is needed. Then the rounded fraction should be 1.100000. Finally, Exp = 4 + 15 = 19 = 10011_2. Knowing that 23.875 is a positive number, the sign bit should be 0. Therefore, the converted representation is the 12-bit IEEE floating-point format should be 010011100000.