APPLIED COMPUTER SCIENCE

ACS-2906-002

Computer Architecture and System Software

Winter 2025

Laboratory 04

Motivation

The goal of this laboratory is to reinforce Bit-Level Operations.

Questions

1. Consider an 8-bit IEEE floating point format with a sign bit, a 3-bit exponent field ($k = 3$) and a 4-bit fraction field ($n = 4$). The bias of the exponent field is 3.

   a) What is the bit representation of the smallest, strictly positive, floating point number in this format? (Do not convert to decimal).

   b) Is the number in a) in normalized or denormalized form?

   c) What is the value of the number in a) in decimal? You can express your answer as a fraction or a power of 2.

   d) What is the decimal value of the largest, non-infinite, floating point number in this format?

2. Write a Java method that converts a variable of type float to floating point binary representation. The signature of your method should be:

   ```
   public static String toBinaryString(float floatBits)
   ```

   You should use the Java built-in function `Float.floatToRawIntBits()` that returns the decimal value of its IEEE float-point bit vector. For example, the following code prints the decimal value of 1236, which is 1150976000. This decimal value in binary is 0100 0100 1001 1010 1000 0000 0000 0000, which is the IEEE Floating point representation for 1236.

   ```
   public static void main(String[] args) {
   float x = 1236f;
   System.out.println( Float.floatToRawIntBits(x) );
   ```

```
}
-----------------------------------------------
Output
1150976000
```

Using the decimal value of the IEEE format, you should use masks, logical, and shift operations to extract the different components of IEEE, namely, the sign (s), exponent (exp), and fractionary (frac). Remember that the IEEE format has 32 bits, of which 1 is for the sign, 8 for the exponent, and 23 for the fractionary or mantisa.

Use it to produce the following output.

Sample Code:

```
float x = 12345f;
float y = 3.12345f;

float result = x + y;

floatToBinaryString(x);
floatToBinaryString(y);
floatToBinaryString(result);
```

Output:

```
s    exp                    frac
- --------      -----------------------------
0 1000 1100  1. 1000 0001 1100 1000 0000 000

s    exp                    frac
- --------      -----------------------------
0 1000 0000  1. 1000 1111 1100 1101 0011 011

s    exp                    frac
- --------      -----------------------------
0 1000 1100  1. 1000 0001 1110 0000 1111 110
```

Evaluation:

- You **must** comment your code to explain that you understand underlying functionality to receive full marks.

Submission instructions

Submit your laboratory solutions via Nexus.