## **Department of Electrical and Computer Engineering**

### **The University of Texas at Austin**

EE 460N, Spring 2017

Problem Set 5 Solutions

Due: April 17, before class

Yale N. Patt, Instructor

Chirag Sakhuja, Sarbartha Banerjee, Jon Dahm, Arjun Teh, TAs

## 

## **Instructions**

You are encouraged to work on the problem set in groups and turn in one problem set for the entire group. The problem sets are to be submitted on Canvas. Only one student should submit the problem set on behalf of the group. The only acceptable file format is PDF. Include the name of all students in the group in the file.

*You will need to refer to the assembly language handouts and the LC-3b ISA on the course website.*

**Problem 1**

Determine the decimal value of the following IEEE floating point numbers.

* 1. -3.25
  2. 1.25 x 2^(-128)
  3. Negative Infinity

**Problem 2**

Using a residue number system with two moduli, represent all of the decimal values between 0 and 11 inclusive when the moduli are

| 4 3 | 6 2  
-----------------  
 0 | 0 0 | 0 0  
 1 | 1 1 | 1 1  
 2 | 2 2 | 2 0  
 3 | 3 0 | 3 1  
 4 | 0 1 | 4 0  
 5 | 1 2 | 5 1  
 6 | 2 0 | 0 0  
 7 | 3 1 | 1 1  
 8 | 0 2 | 2 0  
 9 | 1 0 | 3 1  
10 | 2 1 | 4 0  
11 | 3 2 | 5 1

**Problem 3**

Using the Booth Multiplication Algorithm, multiply the two unsigned 10-bit numbers 0011011110 and 0001110010. Show the intermediate results after each step.

Multiplicand is 0011011110.

|  |  |  |
| --- | --- | --- |
| Cycle | Multiplier | Temp register |
| 0 | 0001110010 | 0000000000---------- |
| 1 | --00011100 | 000110111100-------- |
| 2 | ----000111 | 00000110111100------ |
| 3 | ------0010 | 1111001111011100---- |
| 4 | --------00 | 000110001011011100-- |
| 5 | ---------- | 00000110001011011100 |

The result is 00000110001011011100.

**Problem 4**

The exponent field has a value 63. Therefore the exponent is 63-63=0. The value is 1.0000001111111111 \* 2^0. This value is 1.0156.

**Problem 5**

The bias for the 1-8-23 IEEE format is 127.  
  
 3EE00000h = 0 01111101 11000000000000000000000 = 1.11 \* 2^-2  
 3D800000h = 0 01111011 00000000000000000000000 = 1.0 \* 2^-4  
  
Adjusting exponents, we get  
  
 1.11 \* 2^(-2)  
 0.01 \* 2^(-2)  
  
Since the exponents are now the same, we can add the two fractions. The result has the same exponent as the operands. The result is: 10.00 \* 2^(-2). When normalized to IEEE format, this  
becomes 1.00 \* 2^(-1),

IEEE format: 0 01111110 00000000000000000000000 which in hex is 3F000000h.

**Problem 6**

Model 0.001: 1-7-8  
Smallest positive normalized number is 0 0000001 00000000 = 1.0 \* 2^(1-63) = 2^-62  
Largest postive normalized number is 0 1111110 11111111 = 1.11111111 \* 2^(126-63) = 1.11111111 \* 2^63  
  
This model has (8+1) binary bits of precision. Since 1024 (2^10) is approximately 1000 (10^3); 10 binary bits of precision correspond to about 3 decimal digits of precision. Hence, this model has 9 \* (3/10) = 2.7 decimal digits of precision  
  
Model 0.002: 1-5-10  
Smallest positive normalized number is 0 00001 0000000000 = 1.0 \* 2^(1-15) = 2^-14  
Largest postive normalized number is 0 11110 1111111111 = 1.1111111111 \* 2^(30-15) = 1.1111111111 \* 2^15  
  
This model has (10+1) binary bits of precision. Hence, this model has 11 \* (3/10) = 3.3 decimal digits of precision.  
  
The model chosen would depend on the application - if a higher range is desired, then model 0.001 is better; if precision is more important, then model 0.002 will be preferred.

**Problem 7**

1. 5 bits
2. 1
3. Write each number in in the 9-bit floating point representation, below:

|  |  |
| --- | --- |
| Value | Representation |
| 48 | 0 110 10000 |
| 19.5 | 0 101 00111 |
| 5/16 | 0 000 01010 |
| 0 | 0 000 00000 |
| -1 | 1 001 00000 |
| -infinity | 1 111 00000 |