16.482 / 16.561: Computer Architecture and Design

Fall 2014

Homework #2

Due **Thursday, 9/18/14**

Notes:

- While typed submissions are preferred, handwritten submissions are acceptable.
- Any electronic submission must be in a single file. Archive files will not be accepted.
- Electronic submissions should be e-mailed to Dr. Geiger at Michael_Geiger@uml.edu.
- This assignment is worth a total of 100 points.

1. (10 points) Show how the "refined multiply hardware" (slide #22 from Lec. 2) multiplies the 8-bit values 55 and 13, using an approach similar to the one demonstrated in lecture.

2. (15 points) Calculate the time required for each of the two hardware multipliers shown in class (slides #22 and #26 from Lecture 2) to multiply two 16-bit numbers, if each step of the operation requires 5 ns. For the first multiplier, assume that an addition operation is always performed—the multiplier either adds the multiplicand or 0—and that the shift operation is performed as a separate step.

3. (25 points) Convert each of the following decimal values into single-precision IEEE floating-point format. Show all steps, including how you calculate the fraction and biased exponent stored in the number. *(Note: I encourage you to convert each result into hexadecimal, which will help ensure that your assignments are graded and returned relatively quickly!)*

a. 4.375
b. -19
c. 0.6640625
d. -99.25
e. 2.014 (determine the closest approximation you can)

4. (25 points) Convert each of the following IEEE single-precision floating-point values into decimal values. Show all steps of your work. In some cases, you may need to simply approximate the result.

a. 0x40d80000
b. 0xc0f80000
c. 0x3f380000
d. 0xbf550000
e. 0xabcd1234

5. (25 points) Compute the result of each floating-point arithmetic arithmetic operation below, in which each of the values is encoded in single-precision IEEE floating-point format. Recall that:

   - For floating-point addition, align the binary points, add the significands, then normalize the result.
   - For floating-point multiplication, add the exponents (taking care to only account for the bias once), multiply the significands, normalize the result, and then determine the sign.

All arithmetic should be done in binary, and results should be re-encoded in single-precision IEEE floating-point format.

a. `0x40980000 + 0x40100000`
b. `0x41700000 * 0x3f400000`
c. `0x3fd00000 + 0xc0a00000`
d. `0xc1680000 * 0xbe000000`