# Lab 11: Solutions
# Floating-Point Arithmetic

**Name (Print):** ____________________________________________ID____________

**Give brief answers to the following questions. You can edit this document and insert your answers after each question.**

**Due dates:**

**MW – Wed, May 2, beginning of class**
**TTH – Tue, May 1, beginning of class**

## Circle one: MW or TTH

**Note**: All problems refer to the 16-bit, **modified**, IEEE 754, base-2 floating point format that we discussed in class unless otherwise stated. The decimal-to-float converters that are online are **not** the same.

1. (2 pts) Calculate the decimal equivalent of the base-2 floating-point number $11010.01\times 2^{-3}$. (Remember that calculators are not permitted on exams.)

   **Ans.**

$$\begin{aligned} 11010.01\times 2^{-3} &= 11.01001 \\ &= 3+2^{-2}+2^{-5} \\ &= 3+\frac{1}{4}+\frac{1}{32} \\ &= 3.28125 \end{aligned}$$

2. (2 pts) Using the 16-bit floating point format that we discussed in class, how many different ways are there to represent the number 0? (Hint: Part of the answer is on Slide 6.)

   **Ans.**

   As long as the mantissa is zero, then the number is zero regardless of the value of the sign bit or the exponent. We have 8 exponent bits and 1 sign bit, so there are $2^9$ = 512 different ways to represent the number 0.

| $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ | $s$ | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ | $2^{-5}$ | $2^{-6}$ | $2^{-7}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $b_7$ | $b_6$ | $b_5$ | $b_4$ | $b_3$ | $b_2$ | $b_1$ | $b_0$ | $s$ | $f_1$ | $f_2$ | $f_3$ | $f_4$ | $f_5$ | $f_6$ | $f_7$ |

3. (2 pts) Convert the modified IEEE 754 floating point representation 0x84EC to decimal. Show your work. (Hint: See slides for conversion algorithms.)

   **Ans.** 0x84EC → −13.49

   x = 0x84EC = 1000 0100 1110 1100

   b = 1000 0100 = 132

   e = 132-128 = 4

   s = 1

   f = 110 1100 = ½ + ¼ + 1/16 + 1/32 = 0.8437

   $x = (-1)^s\ f\ *\ 2^e = (-1)^1\ (0.8437)\ (2^4) = -13.49$

4. (2 pts) Convert the decimal 0.0058 to the modified IEEE 754 floating point representation. Show your work.

   **Ans.** 0.0058 → 0x795F

   $x = 0.0058 = (-1)^0\ (0.00000001011111) = (-1)^0\ (0.1011111) * 2^{-7} = (-1)^s\ f\ * 2^e$

   e = -7

   b = 121

   s = 0

   f = 0.1011111

   x = 0111 1001 0101 1111

   = 7 9 5 F

5. (2 pts) Use lab11.asm to calculate the sum of

   0x8340 = 4<br>
   0x83C0 = –4

   Convert the sum to decimal. Is the sum correct?

   **Ans.**

   From lab11.asm, I get: 0x8340 + 0x83C0 = 0x8378 = 7.5. Your sum may be different – it depends on what was in your registers before adding. The sum registers are not cleared. Also, the FloatAdd routine is missing code, so it is unlikely that the answer you get is correct.

6. (2 pts) Convert each of the following to the modified IEEE 754 floating point representation.

   100352
   100353

   Do your answers make sense? Explain.

   **Ans.**

   The modified IEEE 754 floating point representation gives 0x9162 for both because our floating-point format can represent numbers only to within 0.4%, and so 100352 and 100353 both have the same floating-point representation.

7. (10 pts) Note that the `FloatMultiply` routine in lab11.asm does not return the two sample multiplications in normalized form. Create a copy of lab11.asm and rename it to lab11_normalized.asm. Create a new project called lab11_normalized using this file. Add a subroutine called `Normalize` which normalizes the products of the two samples in the code. Demonstrate the correct normalized products in a Watch window and explain the subroutine.

   **Student Name** ______________________________________________________

   **Instructor/TA signature** _________________________________ Date _________