COMP 230: Computer Architecture and Organization

Homework 5

Assigned: October 20, 2017 *Due: November 03, 2017*

Complete this assignment on a separate sheet of paper.

1. Using a table similar to the one used in class (similar to that shown in Figure 3.6), calculate 7 * -42 using the hardware described in figure 3.5. Assume 8 bit signed integers. Show the contents of each register at each step.

2. Using a table similar to the one used in class (similar to that shown in Figure 3.10 in the text), calculate 74 divided 21 using the hardware described in Figure 3.11. Use 8 bit signed integers. Show the contents of each register at each step.

3. What decimal number is represented by the bit pattern `0xC128000`, interpreted as a single-precision floating point number?

4. Write 4.225×10^1 as a single-precision string of bits. Give your answer in hexadecimal.

5. We have talked about how you can divide by a power of two by simple shifting to the right. Unfortunately, this doesn't always work for negative numbers — a negative number has a 1 as the most significant bit, but a right shift puts a 0 as the new most significant bit, meaning the result is positive! A natural idea would be to always *sign extend*: when we do a right shift, put in 1s if the number is negative and 0s otherwise. Give an example where this does not work.

6. Convert each of the following C statements to MIPS assembly language. Assume the variables `f, g, h, i`, and `j` are 32-bit integers as declared in a C or C++ program and are stored in registers `$s0` through `$s4`, respectively. The variables `w, x, y`, and `z` are single-precision floating point numbers and are stored in registers `f0` through `f3`, respectively. Arrays are denoted with uppercase letters, with the base address of `A` in register `$s5`, the base address of `B` in `$s6`, etc. Remember that floating point arguments are passed in registers `$f12` through `$f15`.

   (a) f = A[g] % h;

   (b)
   ```
   float ctof(float celsius) {
     float x = (9.0 / 5.0) * celsius;
     return x + 32.0;
   }
   ```

7. Exercise 3.29 from the text.

8. Consider an 8-bit floating point format with 1 sign bit, 2 exponent bits, 5 fraction bits, and a bias of 1. Treat all 0 or all 1 exponent fields as special, just like normal single- or double-precision numbers (see the figure 3.13 on page 199).

   (a) How many different numbers can be represented?

   (b) What is the smallest number that can be represented with this format?

   (c) What is the largest number that can be represented?

   (d) What is the smallest normalized, positive (and non-zero) number that can be represented?

   (e) What is the smallest positive (and non-zero) number that can be represented?

(f) Come up with a number that is within the range of numbers this format can represent, but cannot be represented exactly. Also, what number would be used to approximate it in this format?

9. Occasionally people will use a *fixed* point representation for fractional numbers, where the binary point is in a fixed location. Part of the bits represent the integer portion of a number, and the rest represent the fractional part. Consider an 8-bit fixed point representation with the first 4 bits representing the (signed) integer part and the last 4 bits representing the fractional part.

   (a) How many different numbers can be represented?

   (b) What is the smallest number that can be represented with this format?

   (c) What is the largest number that can be represented?

   (d) What is the smallest positive (and non-zero) number that can be represented?

   (e) Come up with a number that is within the range of numbers this format can represent, but cannot be represented exactly. Also, what number would be used to approximate it in this format?