EECS150: Components and Design Techniques for Digital Systems

University of California
Dept. of Electrical Engineering and Computer Sciences
David E. Culler **Fall 2007**

Homework 8: Due Friday 11/5 2:10 pm.

In this homework set, we'll work with an 8-bit versions of IEEE 754 Floating Point – CS150 Floating Point, as shown bwlo

7	6	5	4	3	2	1	0
Sign Bit	Exponent			Significand			

CS150 FP has 1 sign bit, 3 exponent bits, and 4 significand bits. It uses the normalized version with the hidden 1 for the significand. The exponent is in excess 3.

Problem 1. To get practice with floating point operations, complete the following worksheet by hand. For the arithmetic problems, you should round your result toward zero (truncate). Show your work and note any overflow that occur.

Convert From Floating Point to Decimal	0 010 1110	0 110 0111	1 100 1001

Floating Point Addition (Write both FP and what your FP is in decimal)	0 010 1110 + 0 110 0111	0 010 1110 + 1 100 1001	0 110 0111 + 1 100 1001
Floating Point Multiplication (Write both FP and what your FP is in decimal)	0 010 1110 x 0 110 0111	0 010 1110 x 1 100 1001	0 110 0111 x 1 100 1001

Problem 2. In this problem you are going to construct the data path for a cs150 floating point adder. Your adder will need to adjust the normalized numbers so that they can be added, add the numbers and return a normalized solution. There are no denorms, NANs, or Infinities. Just the basics. Round toward zero. Draw the data path for this adder.

Write your FP adder in **verilog**.

In words, how would you change the adder if you need to check for infinity?

Problem 3. Linear Feedback Shift Register as a Random Number Generator

This problem is a more open-ended design exercise where you will need to produce a design from a description. You will design a subsystem that provides exponential random backoff support for transmission on a shared communication medium, such as might be used with Ethernet, wifi, or 802.15.4. The transmitter is a black-box that your

subsystem will interface to. When the transmitter starts a backoff sequence it asserts a signal "start" telling your subsystem to produce a random wait-time over the minimum wait interval (minWait). Your subsystem will produce an (n-bit) result that represents an unsigned number specifying the number of clock cycles that the transmitter should delay before attempting to transmit. It will assert a signal, "done", when this output is valid. After the wait, the transmitter determines whether the channel is busy. If not, it will transmit. If so, it will request a random wait time over an exponentially larger interval (twice the length) up to some maximum wait interval (maxWait). So the handshake is Start | Done | … | Next | Done | … | Next | Done, until the transmission succeeds. Each wait is a random length. The intervals that the waits are drawn from increase, up to a maximum value.

The clock frequency is 24 MHz, minWait is ~0.042 ms and maxWait is ~171 ms.

You will use an LFSR to generate the random numbers. (It can just run continuously.)

a) How many bits wide must the output be to specify the maximum wait time? How many of these will be used (i.e., potentially non-zero) for the minimum wait time?
b) How can you extract a random number over the maximum interval? The Minimum interval?
c) Draw the schematic for your design using flipflops, registers, xor gate, shift registers, combinational logic. Show the control and data lines that form the inputs and output of your subsystem. (Hint: use a shift register to generate a mask of the random number.)

Problem 4. In class we discussed how to implement SECDED. For eaxmple, with 8 information bits (d1-d8) we have 4 SEC parity bits (p1-p4) and one DED parity bit p. Before writing a word, we compute each Pi over its parity group. Then we compute p over the Di's and Pi's. Then we store all the Di, Pi, and P. On reading, we compute the parity over each group (an SEC parity bit and the information bits that it covers) to produce the check bits, c1-c4. And we compute the parity C over the entire word (information and SEC parity and DED parity) that was read.

- When an error occurs in a Di bit, what do the check bits c1-c4 contain? What is C?
- When an error occurs in a Pi bit, what do the check bits c1-c4 contain? What is C?
- When an error occurs in P bit, what do the check bits c1-c4 contain? What is C?