CSCE 434/834, Fall 2010

Homework 6 (Due November 29)

Note: Weights of problems are as indicated. This will be your final homework for the course. *You may discuss problems among yourselves but must write your own solutions independently.* Extra time is being allowed because this homework is a bit longer than others and may require extra reading on your part to solve some problems.

Problem 1 [40%].
(a) [10%] In class, we defined the binary "o" operation between two generate-propagate pairs (see slide #12 of *Adders*). We stated without proof that this operation is associative. For this problem, you are asked to prove this result.

(b) [10%] In general, the approach used to reduce the complexity of carry propagation using *parallel prefix computation* applies in general to any problem which can be formulated as a prefix computation problem for an associative operation. For example, the textbook, in Eq. 11.34, shows how the design of a priority encoder can be so formulated.

For this problem, you are asked to consider a binary comparison operation for unsigned numbers (choose one of <. <=, and =) and show that the computational problem can be defined as a prefix computation.

(c) [20%] Sketch Ripple, Lookahead, Brent-Kung, and Sklansky implementations of 8-bit unsigned A op B, where op is the comparison operation you have chosen.

Problem 2. [10%] Consider an n-stage dynamic Manchester carry chain (see the bottom diagram, slide #10 of *Adders*, for an example for n=4) built using minimum size transistors in the carry chain and a unit-size inverter at each output. The maximum parasitic (intrinsic) delay in this circuit would correspond to the propagation of the carry C_0 through all the stages to the final carry output. Estimate this delay, in the units of the delay of a unit-size inverter. Note that the problem can be formulated as a delay through a series-parallel RC network, corresponding to the resistance of transistors and the total capacitance at each C_i-bar node. The delay of such a network can be computed using the *Elmore* delay model (See Sec. 4.3.5, pp. 150-153). Comment on the order of complexity of this delay as a function of n.

Problem 3. [10%] Do Problem 11.18 from the textbook on radix-4 Booth encoding.

Problem 4. [10%] Construct an example input for which the critical path shown by blue dotted line of the array multiplier shown in slide #27 of *Other Datapath Circuits,* will be activated; providing a good justification in support of your answer.

Problem 5. [10%] Do Problem 12.2 from the textbook on estimating SRAM size.

Problem 6. [10%]
(a) Consider the *barrel shifter masking logic* shown in Figure 11.70 of the textbook. You are asked to generate a test for the fault line *kill x* stuck-at-0. Obtain the test, showing all the steps of your derivation. Your answer should be in the form <input-vector, expected output>.
(b) For the test input 1110 in slide #13 of *Test* you are asked to do fault simulation for single-stuck-at faults in the circuit, i.e. to find all such faults detected by the pattern. Show the steps of your computation to the final answer.