Problem Set 10
Assigned 11/28/2007, Due 12/5/2007 at 2 PM

Problem 1. The following STD and STT define the behavior of a simple and not very compact FSM based on your first midterm Motion Detector design problem. Here we have modified the datapath to contain a single timer that takes its initial value from one of two sources. Note that the STT is in the more compact form used in lecture which places the next state under the input conditions.

Use the row reduction method to reduce the number of states. Show which rows you are able to combine. You will end at a point where there are rows that, by inspection, are clear that they can be combined, but they don’t quite satisfy the conditions of the row reduction method. Explain why and compress it the rest of the way.
**Problem 2.** In system design you will often run into forms of Amdahl’s Law, named for the designer of the IBM 360. He published a very short position paper in the 1967 National Computer Conference explaining why vector architectures (machines with special purpose instructions operating on arrays of data, rather than scalars) would not win over general purpose computers. This debate has continues over the years in almost every aspect of computing, including graphics accelerators, digital signal processing, and so on. A simple formalization of “the law of diminishing returns” is a very useful tool in analyzing trade-offs. In this problem you’ll derive a very general form of it.

Let $I$ be the improvement due to some enhancement $E$. For example, if $I$ were the improvement in performance, obtained from reducing the clock cycle time from 50 ns to 30 ns, $I = 50/30 = 1.67$, or a *speedup* of 67%. This is the simple case where the enhancement applies to every aspect of the system under analysis. Amdahl’s deals with what happens when the enhancement applies to only a portion of whole.

2.a. Suppose you have a design with a 120 ns cycle time, 80 ns of which is due to the critical path through combinational logic and 40 ns due to the setup, propagation, and skew associated with the registers. By using an alternative circuit design you reduce the propagation delay of all the combinational logic by a factor of 2, but the registers are unchanged. Calculated the new cycle time and show that the improvement due to this enhancement is 1.5 for a speedup of 50%. What fraction of the total cycle time, $f$, did your enhancement apply to?

2.b. If you work even harder and reduce the combinational propagation delay by another factor of 2, what is the Improvement? Speedup? Fraction of cycle time that this enhancement applies to? (Note, the cycle time is what you had before this particular enhancement, not what you had before 2.a).

2.c. If you work even harder still and reduce the combinational propagation delay by yet another factor of 2, what is the Improvement? Speedup? Fraction of cycle time that this enhancement applies to?

2.d. Give a general formula, $I(E, f) = \ldots$, that gives the improvement due to improving the time taken by a fraction $f$ of the total task by a factor of $E$. Graph this for $f$ ranging from 50% - 100% for a collection of $E = \{2, 5, 10\}$. Can you make a general statement about the improvement if you made a portion of the task infinitely fast (take zero time)? What properties must $E$ and $f$ satisfy to get a 10x improvement?
**Problem 3.** For the combinational circuit below, produce a set of test vectors \(<A,B,C>\) that test all stuck-at-0 and stuck-at-1 faults.

![Circuit Diagram](image)

**Problem 4.** Given that in CMOS circuits the power consumption is dominated by switching costs, show how you would augment the data path shown in Lecture 20, Slide 36 to reduce its power consumption. (Hint: we use “enable” control inputs on registers. Can we use something similar with function units?) Show for the each RTL statement on that slide what control inputs will be asserted by the controller to make the datapath do the register transfer. Include your additional power savings controls.