University of Pennsylvania
Department of Electrical and Systems Engineering
Electronic Design Automation

ESE535, Spring 2013 — Assignment #8 — Monday, April 22

Due: Tuesday, May 7, 12PM (noon).

Resources You are free to use any books, articles, notes, or papers as references. Provide citations in your writeup as appropriate.

Collaboration There is no collaboration on this exercise.

Please include a statement on your final submission:

> I, *your-name-here*, certify that I have complied with the University of Pennsylvania's Code of Academic Integrity in completing this final exercise.

You can review the Code of Academic Integrity here: `http://www.upenn.edu/academicintegrity/ai_codeofacademicintegrity.html`

Questions Email instructor with any questions you have to clarify the assignment problems. Instructor will be traveling April 27–May 1 and May 6–7. Do not expect any response to questions after May 5th.

Writeup Turn-in assignments on blackboard (PDF preferred). See details on course web page. No handwriting or hand-drawn figures. State any assumptions you need to make.

Lateness This assignment cannot be turned in late for partial credit.

Grading You will be graded on the best 3 answers turned in. Each problem is worth 5 points, for a total of 15 possible points. You may choose to complete only three answers or to complete all four.

Hint Problems are not designed to be equally difficult.

Problems

1. Optimize the implementation of conditional blocks for delay and energy.

   On Day 14, we saw that conditional execution (*e.g.*, if-then-else blocks) could be implemented with separate basic blocks or as a single hyperblock using mux-conversion. In this problem, you will provide an algorithm to select the implementation of each conditional block in a computation in order to minimize the expected execution time while working within a target energy budget. For the sake of this problem, we will assume the selection of the conditional implementation is the only freedom that exists.

   Mux-converted hyperblocks typically reduce the delay at the expense of computing both sides of a conditional in parallel. Executing the non-taken side of a conditional will consume energy that is effectively wasted. It is possible that the average delay of a conditional may be shorter when executed as blocks (*e.g.* when the uncommon side of the conditional is longer than the commonly taken side). Consequently, we'll assume you already have analysis that can tell you the expected delay and energy for each of the options.

   Assume you have the following information available to you:

N	Total number of conditionals
E_{mux_i}	The Energy (pJ) for conditional i if implemented using mux-conversion
E_{block_i}	The Energy (pJ) for conditional i if implemented using separate basic blocks
T_{mux_i}	The expected Time (cycles) for conditional i if implemented using mux-conversion
T_{block_i}	The expected Time (cycles) for conditional i if implemented using separate basic blocks
A_i	Normalized fraction of conditional invocation that are conditional i
E_{cond}	Limit on total energy that can be spent across all N conditionals

   You are to determine:

c_i	a binary variable 0 - if conditional should be implemented as blocks 1 - if conditional should be implemented using mux-conversion

   (a) For a final set of c_i assignments, provide an equation that captures the expected normalized total execution time of all conditionals.

   (b) For a final set of c_i assignments, provide an equation that captures the total energy spent on all conditionals.

   (c) Provide an equation that captures the minimum and maximum achievable energy across all implementations?

   (d) Provide a description of and pseudocode for an algorithm to select the c_i's to minimize expected normalized total execution time for a specified conditional energy bound, E_{cond}.

2. Minimize cycle time by simultaneously optimizing placement and retiming.

   We saw in class that performing covering and retiming in sequence could lead to sub-optimal results. In this problem, you examine the problems with performing retiming and placement as separate steps and develop an algorithm to simultaneously deal with both placement and retiming.

   (a) As a default starting point, consider performing retiming before placement. Describe how this can lead to sub-optimal results?

   (b) What prevents us from performing retiming after placement?

   (c) Describe at least three approaches to integrating retiming with placement. For each approach give a one or two sentence description and identify 1–3 strengths of the approach and 1–3 weakness of the approach

   (d) Select one of the approaches and provide a more detailed description of and pseudocode for the algorithm.

3. Perform FSM encoding to minimize energy in a two-level implementation.

   We saw that we have freedom in the selection of the state encodings for Finite-State Machines. In class, we saw how to use that to minimize the Product Terms in a two-level implementation. Here, we would like to consider how to exploit that freedom to minimize switching energy for an FSM implement in two-level logic.

   Assume that you are given:

$\|S\|$	number of state
$\|I\|$	number of inputs to the FSM
$\|O\|$	number of outputs from the FSM
$P(E_{i,j})$	the probability of taking each outgoing edge from a state ($E_{i,j}$ is the edge from state i to state j)
$P(S_i)$	probability of being in state i

   (a) Formulate an equation to calculate the average energy cost for the FSM when implemented in two-level logic given that you have a specific encoding and associated two-level implementation.

   To write this equation, you may need to introduce additional variables and functions characterizing the implementation. Describe them similar to the ones we've already introduced above and in Problem 1.

   (b) Identify the factors in the equation that you can control with your optimization and how you can control them?

   (c) Describe what your encoding should try to do. Illustrate with a good and bad example.

   (d) Provide a description of and psuedocode for an algorithm to select the state encoding to minimize the average energy cost identified in Part (a).

4. Optimize expected completion time in an asynchronous, multi-level circuit graph.

   Consider asynchronous logic where, rather than waiting a worst-case delay for the output of a combinational logic block to clock it into a register, we instead use a dynamic data presence signal computed along with the logic. Here, the delay of a stage of logic may be data dependent, with some input cases being evaluated more quickly than others. How does this change the analysis of timing and the optimizations we perform?

   (a) Provide an example of a logic function that we would optimize differently for this case compared to our standard, synchronous case. Show and explain the different, decomposed multi-level logic implementations for the two cases. Decompose to 2-input ANDs, ORs, and inverts.

   (b) Formulate the timing analysis for an AND gate. Detail how to compute the PDF for the delay output of the gate from the input. The delay of the gate itself is a constant T_{and}.

   Hint: Assume the timing representation between gates includes a probability that the output value is 1 and a timing PDF (similar to the ones used for SSTA) for when the signal settles to 0 or 1 (hence two PDFs, one for each final output value).

   (c) Provide a description of and pseudocode for an algorithm to reduce the expected delay of a multi-level circuit. Assume you are given input probabilities and PDFs as specified for the signals between gates in Part (b).