CSE 30321 – Computer Architecture I – Fall 2008
Final Project Description and Deadlines
Assigned: November 11, 2008 **– Due:** (see this handout for deadlines)

This assignment can be done in groups of 3 or 4.

Introduction

Purpose:

In this final lab, you will be asked to apply what you have learned over the course of this semester in a capstone design project. More specifically, you will be asked to modify the 6-instruction processor design that you have worked with in your lab assignments to improve the baseline performance of a set of benchmarks.

- (Note: a baseline design will be provided.)

A Note on "Performance":

Throughout the semester, "improving performance" has generally meant lower execution time. However, in this project, we'll expand the definition a bit to encompass things like "lower cost," "more capability," etc. (I'll explain more below.)

Realistic Constraints:

For your project, you are free to focus on any of the above metrics when expanding the current design of the simple processor. However, there is one constraint – in order for you to be fully aware of practical factors that impact a processor design, your design needs to be built on realistic assumptions – e.g. realistic memory latencies, estimated fabrication costs, and programmability. The analyses and experiments that you present will be subject to these realistic implementation constraints.

In short:

You will need to propose a set of design optimizations, consider how a set of benchmarks will perform given your proposed optimizations (ideally better!), perform an initial evaluation, implement the hardware changes, show they work, and demonstrate that projected performance improvements are in fact realized.

Benchmarks

Overview:

For your final project, you'll need to quantify the performance of 3 benchmarks that will be specified below. Note that "in real life" an enhancement to a processor often doesn't make all code or tasks run faster. In some cases, an enhancement may only help with just a few types of applications. That said, the said enhancement should not make the performance of the other tasks significantly *worse* either. When you suggest your processor improvements, you should keep the above in mind (i.e. your enhancement should probably help at least 2 of the benchmarks while not adversely hurting the third; of course if your new design improves all 3, that's great too!)

Benchmarks:

- Sum up first n elements of Fibonacci.
- Manipulate data in array that is stored in memory.
- An "if-then-else" that will have you do one thing on an even numbered event, and another on an odd numbered event.

Baseline:

- We will provide initial code sequences for you to compare to.

What you should do:

- Consider how these benchmarks will perform given your proposed HW changes.
- Note: if you do something like adding support for interrupts, show that you can resume execution, etc. instead.

Design Considerations

Things to target:

As mentioned above, while you are essentially free to choose what enhancements you make to your design, do note that extra credit will be provided for the following:

- The design that offers the best area delay product (ADP). ADP quantitatively captures both design area and design latency. A good ADP is generally low and is indicative of a design that has minimal complexity but still offers low latency. To help you quantify ADP, we will provide estimates of transistor count per functional unit. When you have finished your design, you can simply add up the number of transistors that your design requires, multiply the number of transistors by the average area per transistor, and multiply this result by the time required to complete all 3 benchmarks.

- The design that offers the lowest execution time for all 3 benchmarks (irrespective of area).

- The best overall report.

- The most unique design (based on the opinions of the TAs and myself).

Project Ideas:

To help you get started, I've included a few design suggestions below. You certainly do not have to pick one of these design targets – these are simply included to help get you thinking. (Ultimately, you may want to think about what would help the 3 benchmarks referenced above perform better.)

- Extend the capability of the simple processor so it can handle exceptions and/or interrupts.

- Improve the performance of the simple processor in terms of array operations by adding some "powerful" instructions, e.g., instructions from the Pentium MMX instruction set.

- Improve the performance of the simple processor by introducing pipelining.

- Consider a multi-core design.

- Add a cache.

Depending on your personal interests, you may choose a project that is more "hardware" oriented or more "software" oriented. Please feel free to see me or the lab TAs to discuss your ideas.

Design Suggestions

To help you manage the design process more efficiently, I suggest the following process (which should not be new to you if you have been following closely the material discussed in class).

- Select which performance metrics you would like to improve based on your personal preference.
  - (Refer to notes from Lectures 5 and 6.)

- Identify several ways (in terms of changing the instruction set and/or the hardware organization) for improving the selected metrics and choose one or more to work on.
  - (Refer to notes from Lectures 7, 8, 18, 19, and 20 and possibly research other processor architectures.)

- Use the Register-Transfer Level (RTL) description to capture your ideas in Step 2.
  - (Refer to notes from Lecture 17.)

- Construct reasonable benchmarks and use them to estimate the performance/cost/functionality of your new RTL design. If not satisfactory, go to Step 3 or 2.

- Sketch necessary changes to the datapath and control units of the current version of the simple processor. Repeat the previous estimation step if necessary.
  - (Refer to notes that discuss Single Cycle, Multi-cycle, and Pipelined datapath development.)

- Implement your design in Verilog and test it using Xilinx ISE and/or ModelSim. You may use (modified) PSIM to help with your design but you still need to develop a Verilog version of your new processor.

- Download to the Basys Board is optional.
  - (Refer to relevant previous Lab assignments.)

Deadlines and Deliverables

This is a team-oriented effort. Each design team may consist of 3-4 students. Though all team members need to participate in the entire design process, each team member should take a lead role for at least one sub-task. This will give everyone a chance to play the role of a group leader as well as a group member. (You will definitely run into such situations after you enter the “real” world.)

Major deadlines are summarized below:

- Nov. 11th: Project assigned.
- Nov. 18th: Project proposal due in class.
- Nov. 25th: Progress report due in class.
- Dec. 11th: Demonstrations and presentations.
- Dec. 12th: Final report due.

The following grade distribution will be used:

- Proposal: 10%
- Progress report: 10%
- Presentation: 10%
- Final report: 70%
  - Working design: 35%
  - Technical depth: 15%
  - Writing: 20%

More details about the project requirements are given below.

Project Proposal

Each team should submit a **formal** project proposal (2-3 pages in length) by November 18th. The format of the proposal intends to emulate what may be required in a realistic working environment. Please take advantage of office hours this week to discuss your proposal!

The proposal should include:

- Statement of the problem (what metrics you will work on and why)
- Proposed solution (how you will improve the metrics)
- Final deliverables (what you expect to accomplish)
- Timeline that includes some intermediate deliverables
- Time and man hour estimates, division of responsibilities among the team members
- You should also have started working on the RTL description of your solution when you turn in the proposal – and preferably have complete one draft of the RTL description completed.

Progress Report

A progress report is required and is due in class on Nov. 25th. This helps you collect and organize the data and information for the final report. It can be informal and short but needs to include:

- A list of your accomplishments with evidence
  - (such as the RTL code, datapath diagram, Finite State diagram, etc.)
- Evaluation of your progress against what you have proposed.
- Explanation of any major problems encountered/tackled, any modifications that you have made to what you have proposed to achieve, and any needs for modifying your schedule.

Demonstration/Presentation

Around December 11th, you will need to demonstrate your project for me and the lab TAs. Each team will have approximately 10-15 minutes to present their project. This is the time for you to show off your work! Therefore you want to demonstrate that:

- You addressed the problem you proposed to solve.
- Your design does what you claim it does.
- Your experimental data reflected what you wanted to evaluate.
- You learned something new and interesting from doing the project.

Final Report

The final report is an important part of the project. It should be a formal technical report (3-5 pages in length not including attachments), complete but not wordy. Each team turns in one report. The report is due on December 12th by 5 pm. The report should include the following information:

- Detailed statement of problem and your strategy for addressing this problem
- Detailed description of your design
- Description of your benchmarks and experiments
- Discussion of your results
- Conclusion and future work

When preparing the final report, think about the reader. Assume the reader is your manager whose knowledge about computer architecture is about the level of an average student in this class. Use descriptive section titles to help with readability. Include figures and tables wherever necessary. (I don't think you would like to read a long technical report without any figures or tables.) Use the formal English language (e.g., no slang, no contractions). Also make sure that correct technical terms are used. PROOFREAD the report before turning it in.