Project

Important Dates

Assigned:	February 20, 2007
Checkpoint 1:	March 1, 2007
Meeting:	March 15-16, 2007
Checkpoint 2:	March 27, 2007
Final Presentation:	April 17, 2007
Final Report:	April 24, 2007

Description

The project should include a proposal and an evaluation of a concept to improve microprocessor performance. To accomplish this, you should first generate an idea for improving performance. Once you have a conceptual idea, you should state an unambiguous hypothesis that you can prove or disprove through analysis and/or simulation. The evaluation of your hypothesis should include well thought out experiments and analysis, as appropriate. Finally, you should describe your results both in a presentation and write-up that clearly shows your concept, hypothesis, findings, and methodology.

Deliverables

Checkpoint 1: For the first checkpoint, you should have solidified your concept, generated a clear hypothesis, and proposed a plan for evaluating the hypothesis. For this checkpoint, you will have to give a short presentation (~5 minute talk plus 5-10 minutes of questions/discussion) in class and submit a one page write-up describing your concept, hypothesis, and plan.

Checkpoint 2: For the second checkpoint, you should be midway through your evaluation with any experimental software at least partially working. You should have data that clearly motivates your work and have some preliminary results. For this checkpoint, you will have to give a longer presentation (~10 minute talk plus ~5 minutes for questions/discussion) in class briefly reviewing your hypothesis and describing your motivation, preliminary results, and any changes from your evaluation plan.

Final Report: The completed project should include your original concept and hypothesis, as well as your findings and methodology for the project. You will have to present these results in a class presentation (20-25 minutes plus discussion) and submit a well organized write-up of approximately 10 pages.

Meetings

You must meet as a team with Prof. Rixner at least once during the semester, between the checkpoints, to ensure that you are making reasonable progress and are headed in the right direction. You may also meet with Prof. Rixner about the project at any other time you would like guidance.

Teams

You should work in groups of 2-3 people on the project. Each member of the team must deliver at least one of the checkpoint or final presentations.

Possible Concepts

These concepts are given to help start you thinking about ideas for a project. Do not limit yourself to only thinking about these ideas. Feel free to modify them to suit your needs or come up with completely different topics. Also, many of the papers we will read in class leave questions unanswered. This can be a good starting point to investigate solutions to the open issues other people have identified.

Classifying memory accesses: Different types of memory accesses can exhibit different behavior. Consider having the compiler append additional information to memory accesses to differentiate the different types of accesses (e.g. strided stream access, normal access, etc.). Then, techniques for taking advantage of the different types of accesses can be explored, such as the use of stream buffers, compiler controlled memory, or other extensions. What kinds of information could be encoded in memory operations that could be used by the processor (e.g. latency tolerance, miss probability) and how would that information be used?

Bandwidth: Study the amount of data and instruction bandwidth required at the various levels of the processor to keep large numbers of arithmetic units busy. How many arithmetic units would be practical if there is infinite bandwidth? Can you develop efficient ways to provide enough bandwidth to reduce/minimize bandwidth stalls of the pipeline?

Instruction scheduling: There are several mechanisms for increasing the IPC of a processor, such as allowing instructions to execute out of order or by using simultaneous multithreading. Are there other interesting scheduling mechanisms or algorithms that you can develop? Consider scheduling groups of instructions (annotated by the compiler) simultaneously, rather than individually. Or consider other methods of combining static scheduling at compile time and dynamic scheduling at run time.

Media processing: Media processing applications place different demands on the memory hierarchy and computation resources than familiar general-purpose programs. As general-purpose processors execute a larger fraction of media processing applications, they must be modified to allow efficient execution of both types of applications. Study the different characteristics of the two types of applications and find the similarities and differences. Consider evaluating the applicability of features from media processors and DSPs on general-purpose processors that execute media and non-media applications.

Locality of computation: As semiconductor technology advances, wires are becoming more of a bottleneck in terms of delay, area, and power. At the same time, larger numbers of function units require more wiring for issue and result forwarding. There are many approaches to reducing this communication burden. The complete connection provided in most systems today can be replaced with a more efficient interconnect. This interconnect can then be exploited by allowing the compiler (or run-time hardware) to place operations on nearby function units. Other optimizations are also possible.

Projects From Other Classes

The final projects from previous semesters of this class are available on the web page. Also, the final project reports and presentations for similar classes taught at other universities are available on the web. You may look at the projects of others to help you understand and evaluate the scope and complexity of various project ideas. However, do not duplicate someone else's project.