Course Description
CMPS 421

Advanced Computer Architecture

Catalog Description:
This course complements the Computer Architecture course and concentrates on the quantitative principles of computer architecture, instruction set and addressing design, instruction-level parallelism (ILP), compiler considerations for parallelism, cache and memory design, multiprocessor (including multi-core processors) and thread-level parallelism (TLP). A constant theme is how the hardware can achieve greater efficiency by exploiting various types of parallelism.

Prerequisite:
CMPS 321 or equivalent

Units:
5

Instructor:
Marc Thomas

ACM/IEEE Body of Knowledge Topics:
(Note that CMPS 320 covers AR1 and AR2; CMPS 224 covers AR3, and CMPS 321 covers AR4, and AR5 in detail and introduces AR6 and AR7.)

(AR6) Functional organization in a quantitative framework: Understand instruction set design, addressing modes and use of multiple functional units and pipelining in order to achieve parallelism.

(AR7) Multiprocessing: thread-level parallelism.

(AR8) Performance Enhancements at both hardware and compiler level of design.

(Laboratory) Become proficient in a software development environment which allows processor family emulation with code written in both C and assembly.

Texts:

Topics by Text Section:
(Chapter 1) Introduction to basic quantitative principles of processor design, power usage, and performance for desktop, server, and embedded computing.
(Section 1.3 and Appendix B) Instruction set architecture and addressing design.
(Partial: Chapter 5) Preliminary discussion of memory hierachy design; case studies.
(Appendix A) Pipelining: Basic and Intermediate concepts.
(Chapters 2 and 3) Exploiting instruction-level parallelism (ILP) dynamically in hardware; discussion of its theoretical limits.
(Section 2.2) Compiler design to exploit instruction-level parallelism, VLIW designs (briefly).
(Chapters 4 and 5) Thread-level parallelism and the associated cache-coherancy and virtual memory problems; full discussion of memory hierarchy.

Laboratory:
The laboratory session will parallel the lecture, illustrating the principles. We will use the MIPS Technologies SDE (Software Development Environment), which allows MIPS processor emulation with code written in both C and MIPS assembly, in order to investigate various features of RISC design.
ABET Outcome Coverage:

3a. An ability to apply knowledge of computing and mathematics appropriate to the discipline. An ability to understand how computer science relates to mathematics and the physical sciences. Laboratory/homework assignments and questions on the midterms and final require direct applications of the mathematical theory of algorithms pertinent to computer science.

3c. An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs. Embedded questions on midterms and final require direct applications of the principles of design under constraints.

3i. An ability to use current techniques, skills, and tools necessary for computing practice. Laboratory use of the SDE and the assignments made will address this outcome.

Grading:

Two midterms will be given, each worth 25%. I do not give make-up midterms; for an excused absence I count the other grades proportionately higher. One final exam, comprehensive but emphasizing the later material, will be given. It is mandatory and worth 25%. Homework and lab work are together worth the remaining 25%.

Attendance Policy:

Students are responsible for their own attendance. Course materials and assignments will be posted on the course website:

http://www.cs.csback.edu/~marc/code/cs421.html

Academic Integrity Policy:

Homeworks and labs may be worked on and strategy discussed in groups. However, unless otherwise stated, all assignments are individual assignments in that each student must turn in his/her own work; no direct copying is allowed. Refer to the Academic Integrity policy printed in the campus catalog and class schedule.