COSC-121, final exam
2013, spring
Study Guide

The exam will be closed book/notes. Questions will be qualitative and include necessary detailed information as needed. The focus will be on concepts.

Reading:

Patterson & Hennessy: Chapters 1, 4, and 7.
Lecture notes: Lec-4a-MIPSserial, Lec-4b-MIPSpiped, Lec-4c-pipeHazards, Lec-4d-ILP, Lec-5a-parallel.

A smattering of test material will be from prior midterm exams, but the bulk will be from our study of intruction-level parallelism and parallel architecture.

Parallelism simply means more hardware operating simultaneously applied to the same or a larger work load. Two students using two computers rather than sharing one, for instance. The students could be working on the same project together, or might not even know each other. In the first case, we might see them as working on the same data, an example of thread- or process-level parallelism, perhaps. In the second case, we might say they exhibit task-level parallelism if we assume they are working completely independently.

In general, independent actions can be done simultaneously. Where dependency arises, one action must wait for another to complete. For example, designing a book's cover and typesetting its table of contents are independent and can be done simultaneously, but both are dependent on completion of the book's writing. To the degree that indepenent actions can be scheduled simultaneously, parallelism is exploitable. The hardware must provide multiple units, but the architecture should avoid barriers to accessing independent actions.

Topics:

What the basic different types of parallel processing are (SISD, SIMD, MIMD, MISD). The fundamental difference between message-passing and shared-memory multi-processors. Why synchronization is needed.

The different regimes of parallelism: task, process, thread, instruction, and bit levels. That thread-level parallel-hardware supports multiple thread contexts. Why speedup is difficult to attain using parallelism (sequential fraction limitations, contention for resources).

Basic 1-cycle Harvard Architecture implementation of MIPS/LC3. Basic MIPS/LC3 pipelined architecture. Data and control hazard detection and data forwarding. Why clock speed increases as the depth of pipeling increases.

How delays due to dependencies undermine pipelining efficiency. How data dependencies affect both data operations and control operations. How exceptions, branches, and memory access operations affect pipeline performance. How data dependencies from LW (LDR) impacts branches and data operations.

How control dependencies affect data operation efficiency. Why branch-prediction is important. How speculation helps increase pipelining efficiency.

What a super-scalar (multi-pipeling) processor is (multi-instruction issue, multiple functional units).

How programmer, compiler, and hardware instruction scheduling helps performance.