

## When / Where

Thursday, 14 December 2006, 17:30-19:30 (5:30 PM to 7:30 PM) CST (Here).

## Conditions

Closed Book, Closed Notes

Bring one  $215 \times 280$  mm note sheet.

Cannot use communication devices.

## Format


Two or three or maybe four problems.

One set of short answers.

## Resources

Check course Web page daily for hints, new resources.

Web page: <http://www.ece.lsu.edu/ee4720/index.html>

 feed: [http://www.ece.lsu.edu/ee4720/rss\\_home.xml](http://www.ece.lsu.edu/ee4720/rss_home.xml)

Solved tests and homework: <http://www.ece.lsu.edu/ee4720/prev.html>

Statically Scheduled MIPS Study Guide:...

... <http://www.ece.lsu.edu/ee4720/guides/ssched.pdf>

Dynamically Scheduled MIPS Study Guide:...

... <http://www.ece.lsu.edu/ee4720/guides/ds.pdf>

Cache Study Guide:...

... <http://www.ece.lsu.edu/ee4720/guides/cache.pdf>

## Study Recommendations

Study homework assigned this semester—**test questions often are based on homework.**

[Solve previous test problems](#), start with more recent problems.

Memorizing solutions is not the same as solving.

Following and understanding solutions is not the same as solving.

Use the solutions for brief hints and to check your own solutions.

## Implementation Diagrams and Pipeline Execution Diagrams

They are a *team*, so study them together.

## Instruction Use

Should be able to use MIPS and SPARC instructions in examples.

Not required to memorize instruction names, except for common MIPS instructions.

## Introductory Material

ISA v. Implementation.

Factors influencing ISA and implementation.

CPU Performance Equation

Benchmark types.

Compiling and Optimization

## SPEC Benchmark Suite

SPEC membership and their interests.

Benchmark programs (types, how they were selected).

Rules for running benchmarks and disclosing results.

## Compilers and Optimization

Steps in building and compiling.

Basic optimization techniques, compiler optimization switches.

Profiling.

Compiler ISA and implementation switches.

How programmer typically uses compiler switches (options).

## Instruction Set Design

Data Types: What to include, what to leave out.

Basic integer and floating point

Packed types: BCD, integer, saturating integer.

Size choices.

## Memory and Register Organization

Addressing Modes: What they do, which ones to include.

Register, Immediate, Direct, Register Deferred (Register Indirect), Displacement, Indexed, Memory Indirect, Autoincrement, Autodecrement, Scaled.

Control Transfer Instructions: Types, when to use.

Branch, Jump, Jump & Link, Call, Return

Format of displacements in instruction.

Specification of condition: condition code registers, integer registers, loop counter.

Delayed and predicated instructions; prediction hints.

Instruction Coding.

Fixed-length, variable-length, and bundled instructions.

Splitting of opcode field (as in MIPS type-R instructions).



ISA Classifications: RISC, CISC, VLIW

MIPS

Classification: RISC

Goals: ISA should allow simple, high-speed implementation.

Instruction types.

Know how to read and write MIPS programs.

Statically Scheduled MIPS Implementations

Unpipelined Implementation

Pipelined Implementations

Basic (2-cycle branch penalty).

Zero-cycle branch penalty.

Bypassed.

Dependency Definitions

Hazard Definitions

For a Given Pipelined Implementation

Show pipeline execution diagrams.

Show register contents at any cycle.

Determine control hardware.

Determine CPI.

## Interrupts and Exceptions and Traps

Difference between interrupt, exception, trap.

Causes of exceptions, role of handler.

Privileged Mode.

Pipeline activity leading to execution of handler.

Vectored traps (using trap table).

Precise exceptions, achieving with floating-point operations.

## Long Latency (FP) Operations

Types of operations. (Floating point and maybe load.)

Degree of pipelining: Initiation interval, latency.

Detecting functional unit structural hazards.

Detecting WB structural hazards: reservation register.

Detecting and handling RAW hazards: ID-stage v. pre-WB stall.

Handling WAW hazards.

## Loop Unrolling

Hand unrolling of loops.

## Dynamic Scheduling

### Register Renaming

How renaming allows out-of-order execution.

Where registers get renamed.

Role of register maps.

### Reorder Buffer

#### Normal Use

Issue: placement of instructions in reorder buffer.

Completion: updating reorder buffer entry

Commit (retirement): removal from reorder buffer entry.

## Recovery

Goal: Undo execution starting at some instruction.

Reasons: exception, branch misprediction.

Steps when register map not backed up.

## Multiple Issue

### Superscalar

Duplication of Resources.

For  $n$  instructions / cycle: Fetch, decode, rename, writeback, commit.

For  $< n$  instructions: load/store, floating-point units.

### Added Complexity

Instruction fetch inefficiency.

Rules for fetching a new group.

### VLIW

Difference with superscalar: instruction bundling.

Dependence information in bundles.

## Branch Prediction

One-level branch prediction, bimodal predictor.

Two-level (local,global,gshare) branch-prediction.

## Less Important Topics

Branch target prediction.

Return address stack.

Predictability of indirect control transfers.



## Memory

### General

Connection of memory devices.

## Caches

Cache structure, connection of memory devices.

Line size implications.

Computing hit ratio for given program.

### Set Associative Caches

Definitions: line (block), index, tag, alignment, associativity

Connection of tag and data memory.

Special Cases: Direct mapped, Fully associative.