Spring 2004 Final Exam Review

When / Where
Thursday, 13 May 2004, 17:30-19:30 (5:30-7:30 PM) CDT (Here).

Conditions
Closed Book, Closed Notes
Bring one 215 x 280 mm note sheet.
Cannot use communication devices.

Format
Two or three or maybe four problems.
One set of short answers.

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

Study Recommendations
Study homework assigned this semester—test questions may be based on homework.
Solve old 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.

Emphasis
Implementation Diagrams and Pipeline Execution Diagrams
They are a team, so study them together.

Instruction Use
Should be able to easily write MIPS programs.
Should be able to use other instructions in examples.
For example, SPARC, etc.
Not required to memorize instruction names, except for common MIPS instructions.

Cache Diagram, Address Bits, and Program
Determine cache structure from diagram. (Line size, etc.)
Determine hit ratio from program.
Write program to fill cache, maximize misses, etc.

Topics
Introductory Material
ISA v. Implementation.
Factors influencing ISA and implementation.
Design principles: Amdahl's law, locality.
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

Stack and accumulator architectures.
Memory/Memory, Register/Memory.

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, Stack, Accumulator

MIPS
Classification: RISC
Goals: ISA should allow simple, high-speed implementation.
Instruction types.
Know how to read and write MIPS and DLX 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 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.

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
Commitment (retirement): removal from reorder buffer entry.
Recovery
- Goal: Undo execution starting at some instruction.
- Reasons: exception, branch misprediction.
- Steps when register map backed up.
- Steps when register map not backed up.

Load/Store Unit
- Goal: Prevent stores from storing before they commit.
- Goal: Maintain program order of stores and loads.
- Store/load ordering rules.
- Load/store bypassing.

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.
- Branch target prediction.
- Return address stack.
- Predictability of indirect control transfers.

Memory
- General
  - Definitions: bus width ($w$), address space size ($a$), character size ($c$).
  - Connection of memory devices.
- Caches
  - Set Associative Caches
    - Definitions: block (line), index, tag, alignment, associativity
    - Connection of tag and data memory.
    - Special Cases: Direct mapped, Fully associative.
Cache Write Options
Write allocate or write around.
Write back or write through.

Victim (Block to replace) Selection
Least-Recently Used (LRU), Random (arbitrary).

Material from Homework Assignments
Homework 4 (implementing the complex instructions).
PPC 970 / POWER4.
Make sure that you understand the posted solutions.