

# Computer Architecture Review

or

*CSE141's Greatest Hits*

*CSE 141*

*Allan Snively*

## Instruction Set Architecture

- What we learned
  - ISA types
  - ISA formats and tradeoffs
  - addressing modes
  - branch types
  - MIPS ISA

*CSE 141*

*Allan Snively*

# Instruction Set Architecture

- What we can do
  - identify and write simple code for various ISA types
  - identify and use several addressing modes (MIPS type, particularly)
  - write MIPS code (with cheat sheet)
  - encode or decode MIPS code to/from machine language
  - evaluate tradeoffs between ISAs

# Performance

- What we learned
  - speedup
  - execution time
  - benchmarks
  - Amdahl's law

# Performance

- What we can do
  - calculate CPI, ET, clock-rate, etc.
  - calculate speedup
  - apply Amdahl's law

# Computer Arithmetic

- What we learned
  - the computer uses *binary numbers*
  - number systems
  - negative numbers
  - addition, subtraction, multiplication, division
  - ALU design
  - fast adders
  - floating point numbers, operations

# Computer Arithmetic

- What we can do
  - manipulate binary numbers
  - do arithmetic on binary numbers
  - do arithmetic on fp numbers
  - simple adder, ALU design

# CPU Architecture

- What we learned
  - single-cycle cpu, multiple cycle cpu
  - datapaths
  - control logic
  - multiple-cycle control
    - FSM
    - microprogramming
  - exceptions

# CPU Architecture

- What we can do
  - construct datapath for new instructions
  - generate control logic (or FSM or microprogram) for new datapath or new instruction (don't memorize microprogram format, just learn concepts)
  - incorporate exceptions into datapath and control

# Pipelining

- What we learned
  - pipelined machine design, including
    - use of intermediate registers
    - pipelined control
  - data hazards, bubbles, and forwarding
  - branch hazards, bubbles/flushing, and simple branch prediction
  - advanced architectural concepts including branch prediction, superscalar execution, superpipelining, and out-of-order execution.

# Pipelining

- What we can do
  - design a slightly different pipelined machine
  - generate control for it
  - understand implications of all kinds of data hazards
  - understand implications of all kinds of branch hazards
  - reason about instruction schedules for pipelined, superscalar, out-of-order, or VLIW machines

# Memory

- What we learned
  - locality
  - memory hierarchies
  - hits, misses, miss penalties
  - cache performance (miss rates, MCPI, ET)
  - write-through, write-back, write-allocate, write-around
  - associativity
  - virtual memory, page tables
  - TLBs

# Memory

- What we can do
  - identify types of locality, types of misses (coherence, capacity, conflict)
  - identify hits and misses and manipulate cache structures for all types of caches
  - evaluate or predict cache performance
  - do LRU replacement
  - manipulate page table and TLB structures, identify TLB misses and page faults

11. (5 points) A 256 KB cache interprets addresses this way. What is the associativity of this cache? (Addresses are byte addresses)

16 bits	10 bits	6 bits
tag	index	block offset

# Multiprocessors

- What we learned
  - bus-based, network-based
  - UMA, NUMA
  - shared-memory, message-passing
  - cache coherency
  - multithreading