## Computer Organization

What's in it for you? In-depth understanding of inner working of modern computers, their evolution, and the trade-offs present at the hardware-software boundary

Understanding of the hierarchy of layers that comprise computer systems -

- Hardware
- Systems Software
- Applications Software

## Summary

- The Big Picture
- Integrated Circuit Costs/Performance Evaluation
- Instruction Set Architectures
- Combinational (digital) logic/arithmetic
- Finite state machines
- Control: Hardware Vs. Microprogrammed
- Pipelining
- Exploiting Instruction-Level Parallelism (ILP)
- Memory Hierarchy
- I/O and Interconnection Network Issues
- Parallel Processing

## The Big Picture

Many levels of abstraction - hardware & software

Instructions and data naturally represented in binary

Important abstraction - Instruction Set Architecture

Five components of computers -

- Processor : datapath and control
- Memory
- Input and Output devices

Technology Trends affect price and performance

## Hardware Cost

$$DieCost = \frac{Wafer\ Cost}{Dies\ per\ Wafer \times Die\ Yield}$$

$$Die\ Yield = Wafer\ Yield \times \{1 + \frac{Defects\ per\ unit \times Die\ area}{a}\}^{-a}$$

Typical values: Wafer Yield = 90%, Defects per unit area = 1 $cm^2$, a = 2 or 3

The bigger the chip, the less likely it is to come out working correctly

## Performance Summary

$CPUTime = Instructions \times AverageCPI \times Cycletime$

Time is the measure of computer performance!

Need for good benchmarks and performance summaries

Amdahl's Law: Speedup is limited by unimproved part of program

## Design Principles for RISC Machines

1. Analyze the applications to find the key operations
2. Design a data path that is optimal for key operations
3. Design instructions that perform the key operations using the data path (registers+ALU)
4. Add new instructions only if they do not slow down the machine
5. Repeat this process for other resources

## RISC Open Issues

Register usage

Setting of condition codes

Byte ordering

Hardware interlocks

## RISC Versus CISC

One instruction per data path cycle

Load/store architecture (only register addressing permitted for ordinary instructions)

Pipelining - one instruction issued each cycle

No microcode (hardwired for the common case) - reduced chip area

Increased memory usage

Fixed-format instructions

Reduced Instruction set

Put the complexity in the compiler - delayed loads, reduced addressing modes, optimize register usage

Multiple register sets

## Arithmetic

2's complement

Floating point representation - sign, exponent, significand

Simple ALU and Multiplier design

Floating point unit design

Floating point rounding

Decimal to binary conversion

## Finite State Machines

Sequential system that contains state

Described with a state table

Set of states and two functions -

- Next state function
- Output function

Moore Machine - Output = F(Current State)

Mealy Machine - Output = F(Current State + Current Input)

## Datapath and Control

Replicated Versus Re-used function units

Single Versus Multiple Clock Cycles

Hardwired Versus Microprogrammed

Exception handling

## Pipelining

Improves clock rate and instruction throughput but not latency

Several cycles per instruction and replicated hardware

Compiler writer must understand pipeline to achieve best performance

Basic execution cycle (and possible pipeline stages) - Instruction Fetch, Instruction Decode, Execution (ALUop), Memory Access, and Write Back (Register Write)

## Factors That Result in Imperfect Speedups due to Pipelining

- Data hazards
- Control hazards
- Clock skew and latch overheads

## Exploiting Instruction-Level Parallelism (ILP)

Pipelining

Super-pipelining - Issue 1 instr./fast cycle

Super-scalar - Issue multiple scalar instructions per cycle (hazard resolution problem)

VLIW - Each instruction specifies multiple scalar operations (packing problem)

## Memory Hierarchy

The Principle of Locality: Programs access a relatively small portion of the address space at any instant of time

- Temporal locality: locality in time
- Spatial locality: locality in space

Three major categories of cache misses:

- Compulsory
- Conflict
- Capacity

Virtual memory invented as another level of the hierarchy

- Protection/sharing of single memory
- Illusion of large address space independent of physical layout
- TLBs important for fast translation/checking
- Segmented/page virtual memory schemes

## Computer System I/O

Interrupt handling Versus Polling

Memory mapped Versus I/O Instructions

Programmed I/O with Busy Waiting

Interrupt-driven I/O

DMA I/O

## Interconnection Network Issues

Switched Versus Shared Media

Circuit switched versus packet-switched networks

Hardware and software issues affect latency

Reliability and routing

## Parallel Computer Systems

Flynn's classification - SISD, SIMD, MIMD

Issues in MIMD:

- Data sharing - single versus multiple address spaces, shared memory versus message passing
- Process coordination - synchronization using locks/barriers versus messages
- Distributed (non-uniform access) versus centralized (uniform access) memory
- Connectivity - single shared bus versus network (with many different topologies)
- Fault tolerance

## Clocks

Free-running signal with a fixed cycle time

Edge-triggered clocking - state changes occur on clock edge

Constraint in synchronous or clocked system: Input must be valid immediately before and after active edge - *setup* and *hold* times

To avoid races -

$T_{period} > t_{prop} + t_{combinational} + t_{setup} + t_{skew}$

$t_{prop} + t_{combinational} - t_{skew} > t_{hold}$