CIS 501
Computer Architecture

Unit 0: Introduction

Slides developed by Milo Martin & Amir Roth at the University of Pennsylvania
with sources that included University of Wisconsin slides
by Mark Hill, Guri Sohi, Jim Smith, and David Wood.

What is Computer Architecture?

- *"Computer Architecture* is the science and art of selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals." - WWW Computer Architecture Page
- An analogy to architecture of buildings…

What is ~~Computer~~ Architecture?

The role of a ***building*** architect:

Materials → Design → Plans → Construction → Buildings; Goals → Plans; Buildings → Plans

Materials
Steel
Concrete
Brick
Wood
Glass

Goals
Function
Cost
Safety
Ease of Construction
Energy Efficiency
Fast Build Time
Aesthetics

Buildings
Houses
Offices
Apartments
Stadiums
Museums

What is Computer Architecture?

The role of a ***computer*** architect:

"Technology" → Design → Plans → Manufacturing → Computers; Goals → Plans; Computers → Plans

"Technology"
Logic Gates
SRAM
DRAM
Circuit Techniques
Packaging
Magnetic Storage
Flash Memory

Goals
Function
Performance
Reliability
Cost/Manufacturability
Energy Efficiency
Time to Market

Computers
Desktops
Servers
Mobile Phones
Supercomputers
Game Consoles
Embedded

Important differences: age (~60 years vs thousands), rate of change, automated mass production (magnifies design)

Computer Architecture Is Different…

- Age of discipline
  - 60 years (vs. five thousand years)
- Rate of change
  - All three factors (technology, applications, goals) are changing
  - Quickly
- Automated mass production
  - Design advances magnified over millions of chips
- Boot-strapping effect
  - Better computers help design next generation

Design Goals

- **Functional**
  - Needs to be correct
    - And unlike software, difficult to update once deployed
  - What functions should it support (Turing completeness aside)
- **Reliable**
  - Does it ***continue*** to perform correctly?
  - Hard fault vs transient fault
  - Google story - memory errors and sun spots
  - Space satellites vs desktop vs server reliability
- **High performance**
  - "Fast" is only meaningful in the context of a set of important tasks
  - Not just "Gigahertz" – truck vs sports car analogy
  - Impossible goal: fastest possible design for all programs

Design Goals

- **Low cost**
  - Per unit manufacturing cost (wafer cost)
  - Cost of making first chip after design (mask cost)
  - Design cost (huge design teams, why? Two reasons…)
  - (Dime/dollar joke)
- **Low power/energy**
  - Energy in (battery life, cost of electricity)
  - Energy out (cooling and related costs)
  - Cyclic problem, very much a problem today
- Challenge: balancing the relative importance of these goals
  - And the balance is constantly changing
    - No goal is absolutely important at expense of all others
  - Our focus: *performance,* only touch on cost, power, reliability

Shaping Force: Applications/Domains

- Another shaping force: **applications** (usage and context)
  - Applications and application domains have different requirements
    - Domain: group with similar character
  - Lead to different designs
- **Scientific**: weather prediction, genome sequencing
  - First computing application domain: naval ballistics firing tables
  - Need: large memory, heavy-duty floating point
  - Examples: CRAY T3E, IBM BlueGene
- **Commercial**: database/web serving, e-commerce, Google
  - Need: data movement, high memory + I/O bandwidth
  - Examples: Sun Enterprise Server, AMD Opteron, Intel Xeon

More Recent Applications/Domains

- **Desktop**: home office, multimedia, games
  - Need: integer, memory bandwidth, integrated graphics/network?
  - Examples: Intel Core 2, Core i7, AMD Athlon
- **Mobile**: laptops, mobile phones
  - Need: **low power**, integer performance, integrated wireless
  - Laptops: Intel Core 2 Mobile, Atom, AMD Turion
  - Smaller devices: ARM chips by Samsung and others, Intel Atom
- **Embedded**: microcontrollers in automobiles, door knobs
  - Need: low power, **low cost**
  - Examples: ARM chips, dedicated digital signal processors (DSPs)
  - Over 1 billion ARM cores sold in 2006 (at least one per phone)
- **Deeply Embedded**: disposable "smart dust" sensors
  - Need: extremely low power, extremely low cost

Application Specific Designs

- This class is about **general-purpose CPUs**
  - Processor that can do anything, run a full OS, etc.
  - E.g., Intel Core i7, AMD Athlon, IBM Power, ARM, Intel Itanium
- In contrast to **application-specific chips**
  - Or **ASICs** (Application specific integrated circuits)
    - Also application-domain specific processors
  - Implement critical domain-specific functionality in hardware
    - Examples: video encoding, 3D graphics
  - General rules
    - \- Hardware is less flexible than software
    - \+ Hardware more effective (speed, power, cost) than software
    - \+ Domain specific more "parallel" than general purpose
      - But general mainstream processors becoming more parallel
- Trend: from specific to general (for a specific domain)

Constant Change: Technology

"Technology"
Logic Gates
SRAM
DRAM
Circuit Techniques
Packaging
Magnetic Storage
Flash Memory

Applications/Domains
Desktop
Servers
Mobile Phones
Supercomputers
Game Consoles
Embedded

Goals
Function
Performance
Reliability
Cost/Manufacturability
Energy Efficiency
Time to Market

- Absolute improvement, **different rates of change**
- New application domains enabled by technology advances

Technology Trends