CS 6290
Intro & Trends
Milos Prvulovic - Fall 2007
Prerequisites

- I will assume you have detailed knowledge of
  - Pipelining
    - Classic 5-stage pipeline, pipeline hazards, stalls, etc.
  - Caches
    - Tag/Index/Offset, hit/miss, set-associativity, replacement policies, write-through/write-back, etc.
  - Assembler and ISAs
    - RISC, load/store, instruction encoding, caller-saved/callee-saved registers, stack pointer, frame pointer, function call/return code, etc.

- If you don’t remember a few of these
  - Read Appendices A, B, and C
  - Review the CS2200 textbook

- If you don’t know what some of these are
  - Take CS2200 before you take this class, or
  - Read the CS2200 textbook before next week’s lectures
Logistics

• Internet
  - [http://www.cc.gatech.edu/~milos/CS6290F07](http://www.cc.gatech.edu/~milos/CS6290F07)
    • temporary address
  - Email: milos@cc

• Office hours: TBA

• TAs:
  - Minjang Kim, minjang@cc, office hours TBA
  - Sunjae Park, sunjae.park@gatech, office hours TBA

• Textbook:
    by Hennessy and Patterson
What is Architecture?

- **Original sense:**
  - Taking a range of building materials, putting together in desirable ways to achieve a building suited to its purpose

- **In Computer Science:**
  - Similar: how parts are put together to achieve some overall goal
  - Examples: the architecture of a chip, of the Internet, of an enterprise database system, an email system, a cable TV distribution system

Adapted from David Clark’s, What is “Architecture”? 
Why Computer Architecture?

• Exploit advances in technology
  – Make things Faster, Smaller, Cheaper, ...

• Which enables new applications
  – Shrek 20 years ago?

• Make new things possible
  – Accurate one-month weather forecasts? Cure for cancer? Life-like virtual reality?

• The advancement of computer architecture is vital for the advancement of all other areas of computing!
Today:

• Trends in Computer Industry
  – setting the stage for the what’s, why’s and how’s to come through the rest of this course
Moore’s Law (1965)

• Transistors per inch square
  – Twice as many after ~1.5-2 years

• Related trends
  – Processor performance
    Twice as fast after ~18 months
  – Memory capacity
    Twice as much in <2 years
Moore’s Not-Exactly-Law

• Not a law of nature
  – But fairly accurate over 42 years and counting
• No exponential is forever
  but we can delay “forever”
  (Gordon Moore in 2003)
• More about Moore’s Law at
  http://www.intel.com/research/silicon/mooreslaw.htm
How to use 1B (or more) transistors?

1. Instruction set architecture (ISA)
   - interface between HW and SW
   - different ISAs may be more/less effective for different target application areas

2. Microarchitecture
   - Techniques below the ISA level (transparent)
   - ex. pipelining, caching, branch prediction, superscalar, dynamic scheduling, clock-gating, ...

Review appendix C
Review appendix A
Review 1.3 and appendix B
Performance Trend

Doubling the number of people on a project doesn’t speed it up by 2x

Similarly, 2x transistors does not automatically get you 2x performance

Possible because of continued advances in computer architecture.

Much of computer architecture is about how do you organize these resources to get more done.
Price Trends (Pentium III)

Raw performance *and* performance per $ improves with time
Price Trends (DRAM memory)

Similar trends for main memory: capacity, and capacity per $ both getting better.
Why Do You Care About Prices?

• Target market, target prices place a limit on the cost of my processor
  – price = what I sell the part for
  – cost = what it costs me

• Design decisions affect the cost (and price)
  – Ex. adding more cache may improve performance, but increase cost

• Price-performance is often what we’re trying to balance

So what determines price/cost?
# Depends on the Class of Processor

<table>
<thead>
<tr>
<th>Feature</th>
<th>Desktop</th>
<th>Server</th>
<th>Embedded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Price of system (USD)</td>
<td>$500-$5K</td>
<td>$5K - $5M</td>
<td>$10 - $100K (ex. high-end network routers)</td>
</tr>
<tr>
<td>Price of CPU (per processor)</td>
<td>$50 - $500</td>
<td>$200 - $10K</td>
<td>$0.01 - $100</td>
</tr>
<tr>
<td>Critical design issues</td>
<td>Price-performance, graphics performance</td>
<td>Throughput, availability, scalability</td>
<td>Price, power, application-specific performance</td>
</tr>
</tbody>
</table>
Desktop Systems

- **Examples**
  - Intel Core 2 Duo
  - AMD Opteron

- **Applications:** everything (general purpose)
  - Office, Internet, Multi-media, Video Games...

- **Goals**
  - performance, price/performance
  - power → affects cost, noise, size
Servers

• Examples
  – IBM Power
  – Sun Niagara (T1)
  – Intel Xeon

• Applications
  – infrastructure: file server, email server, ...
  – business: web, e-commerce, databases

• Goals
  – Throughput (transactions/second)
  – Availability (reliability, dependability, fault tolerance ...)
  – Cost not a major issue
Embedded

• Examples
  – Xscale, ARM, MIPS, x86, ... (many varieties)

• Applications
  – cell phones, mp3 players, game consoles, consumer electronics (refrigerator, microwave), automobiles, ... (many varieties)

• Goals
  – Cost, Power
  – Sufficient performance, real-time performance
  – Size (CPU size, memory footprint, chip count...)

Georgia Tech College of Computing
Fabrication Costs

- CPU (die) size greatly affects cost of all systems (desktop/server/embedded)
  - Current CPUs 1-2 cm²
  - Embedded much smaller
    - cost and footprint really matters in cell phone or iPod
Yield

13/16 working chips
81.25% yield

1/4 working chips
25.0% yield
Yield (2)

52 die, 81.25% yield → 42.25 working parts / wafer

17 die, 25.0% yield → 4.25 working parts / wafer

Assuming $250 per wafer:
- $5.92 per die
- $58.82 per die
Cost/Yield Equations
(approximations)

Cost of Die = \frac{\text{Cost of wafer}}{\text{Dies per wafer} \times \text{Die yield}}

Dies per wafer = \frac{\pi \times (\text{Wafer diameter} / 2)^2}{\text{Die area}} - \frac{\pi \times \text{Wafer diameter}}{\sqrt{2 \times \text{Die area}}}

Die yield = \text{Wafer yield} \times \left(1 + \frac{\text{Defects per unit area} \times \text{Die area}}{\alpha}ight)^{-\alpha}

Number of completely bad wafers

Typical: 0.4 defects per cm² in 90nm, but improves with time

Parameter related to complexity of manufacturing, typical \( \alpha = 0.4 \)
Interaction of Price and Performance

• Add a new architectural feature to chip
  – for more performance, for less power, etc.

• Chip die size increases
  – Fewer dies per wafer
  – More defective dies

• Die testing more expensive
  – Must test whether feature works

• Die package more expensive
  – Larger package, maybe more pins
  – If feature needs more power, may need better heat sink
Goal of Processor Design

• Maximize performance
• Within the constraints of
  – Peak power, average power, thermals, reliability, manufacturing costs, implementation complexity, verification complexity, time-to-market, cost to manufacturer (Intel), cost to OEM (Dell), cost to end-customer (you)
  – Which really just says:
    • Maximize performance per $$$

• Huge, multi-variable optimization problem!
  – Not all variables are independent
The Rest of this Course

• Not as much *explicit* focus on price, although it will be kept in mind

• How do you organize these millions/billions of transistors to implement the ISA
  – data-processing (workers)
  – control-logic (managers)
  – memory (warehouse)
  – parallel systems (multiple worksites)