CS 211: Computer Architecture

Instructor: Prof. Bhagi Narahari
Dept. of Computer Science
Course URL: www.seas.gwu.edu/~narahari/cs211/

Course Information

- Course materials placed at
  - www.seas.gwu.edu/~narahari/cs211/
  - All lecture notes, homeworks, and announcements
  - Check at least once a week – before class.
- Late submission of homework will be penalized

Course Requirements

- Prerequisites: data structures, discrete math, computer organization
- Requirements:
  - Midterm – 35%
  - Final – 30%
  - Homework assignments+Project – 35%
    - Includes programming homeworks/projects
    - Includes a programming project
    - Students "may" be permitted to substitute project for a term paper—will have to meet me during beginning of January with a term paper proposal.
- Academic Integrity Policy
  - Absolutely no collaboration on homeworks
  - Programming projects can be done in 3 person teams – no collaboration between teams

Programming projects

- Part of homework includes programming using Simple Scalar simulator
  - Project will be based on using this
  - Students placed into 3 person teams for programming projects – team selection by Feb 15th
- www.simplescalar.com
- Reading assignment for next week: visit simple scalar website, read tutorial, figure out installation instructions.
- Machines in Staughton Hall room 307
  - Linux machines
  - Simplescalar will be installed on some of them – TA will cover this in office hours
Course Outline

• Computer Organization Review – Self study
• Architecture challenges, design objectives, thumb rules, emerging issues
• (I) Processor architectures:
  ➢ Instruction level parallel (ILP) processors
  ➢ Pipelined, superscalar, and EPIC/VLIW..vector
  ➢ Midterm – date to be decided…plan for 8th or 9th week
  ➢ Compiler optimization techniques
  ➢ challenges of embedded systems
  ➢ Memory organization
• (II) Multiprocessor architectures:
  ➢ SIMD, MIMD, Interconnection networks, clusters
• (III) Parallel Algorithms and Comp.Model

Course Information

• Textbook: Hennessy and Patterson, Computer Architecture: A quantitative approach; Pub. Morgan Kauffman
• course topic to book chapter mapping will be placed on website
• Website will contain lecture materials and homeworks, as well as references
  ➢ URL for course
    www.seas.gwu.edu/~narahari/cs211
  ➢ Keep checking website at least once a week for announcements and postings

Computer Architecture – Course Objectives

• Study the role of computer architecture in system/program performance
  ➢ What aspects of architecture design affect performance of application
  ➢ What are the key components of CA ?
  ➢ What is the architecture of today’s high performance processors ?
  ➢ How to extract maximum performance out of today’s processors ?
  ➢ What are the emerging trends in CA ?
  ➢ Take a quantitative approach to CA
• What the course is not
  ➢ Detailed exposition on hardware design
  ➢ Semiconductor technology details
  ➢ Case studies

Perspective

• Computer architecture design is directly linked to underlying technology
  ➢ Semiconductor
  ➢ Compiler technology
  ➢ Computational models
• Our goal as software designers is to run an application program efficiently on the architecture
  ➢ Compiler plays a key role
  ➢ what is the interplay between architecture features and application program properties
  ➢ Bottom line is performance of application
Review the basics of the interplay between hardware and software

Brief review of technology trends and implications on emerging architecture designs

Hardware
- Medium to compute functions

Software
- Functions to compute

Computational Model connects them

Functions to compute

- Programming language

- Turing Machines, Recursive Functions

What does Computer Architecture deal with – The Multi-Level Concept

Different levels, each with its unique functionality
- Problem-Oriented Language Level (prog languages)
- Assembly Language Level
- Operating System machine level
- Conventional Machine Level (Instruction Set Architecture -- ISA)
- Micro-architecture level (Microprogramming level)
- Digital Logic Level (program in VHDL, Verilog)
  - Device & Semiconductor Level
Another View of Architecture Levels

- Network Application Layer
  - web computing, CORBA
- Network of Processors - networked architecture
  - Cluster Computing
- System level architecture
  - Multiple interconnected processors
  - CPU, Memory, I/O
- Processor architecture
  - processor design and programming
- Microarchitecture
- Device architecture
  - Gallium Arsenide, TTL, Optical

Hardware support must scale
(e.g. HPL-PD)
Eg. Clock dilation
Sensitive to hidden hardware costs

The Backdrop

- Who will program these machines?
  - Programmers
- What do they expect?
  - Performance
- How?
  - Write HLL program and Compile
  - Compilation is key to performance
    - Requires Hardware/Software interaction at ISA level
    - Knowledge of architecture, application, algorithm

The AAA rule for designers

- Algorithm
- Application
- Architecture
Review: Computer Organization Basics

- What are the components of a CPU
- What is a microarchitecture level?
- What is an ISA - Instruction set architecture?
- How does a sample processor design look?
  - A simple processor architecture
- What is the basic concept of pipelining

Trends In Technology, Applications, Architectures

Original Food Chain Picture

Big Fishes Eating Little Fishes

1998 Computer Food Chain

Now who is eating whom?
Why Such Change in 10 years?

- Performance
  - Technology Advances
    - CMOS VLSI dominates older technologies (TTL, ECL) in cost and performance
    - Computer architecture advances improves low-end
      - RISC, superscalar, RAID, ...
  - Price: Lower costs due to …
    - Simpler development
      - CMOS VLSI: smaller systems, fewer components
    - Higher volumes
      - CMOS VLSI: same dev. cost 10,000 vs. 10,000,000 units
    - Lower margins by class of computer, due to fewer services
  - Function
    - Rise of networking/local interconnection technology

Memory Capacity
(Single Chip DRAM)

Year | size(Mb) | cyc time
--- | --- | ---
1970 | 1000 | 300 ns
1975 | 10000 | 300 ns
1980 | 100000 | 300 ns
1985 | 1000000 | 300 ns
1990 | 10000000 | 300 ns
1995 | 100000000 | 300 ns
2000 | 1000000000 | 300 ns

Technology Trends: Microprocessor Capacity

"Graduation Window"
- Alpha 21264: 15 million
- Pentium Pro: 5.5 million
- PowerPC 620: 6.9 million
- Alpha 21164: 9.3 million
- Sparc Ultra: 5.2 million

CMOS improvements:
- Die size: 2X every 3 yrs
- Line width: halve / 7 yrs

Performance Trends (Summary)

- Workstation performance (measured in Spec Marks) improves roughly 50% per year (2X every 18 months)
- Improvement in cost performance estimated at 70% per year
Architecture is an iterative process:
- Searching the space of possible designs
- At all levels of computer systems

Creativity
Good Ideas
Mediocre Ideas
Bad Ideas

Cost / Performance Analysis

Computer Engineering Methodology

Implementation Complexity
Evaluate Existing Systems for Bottlenecks
Analysis

Implementation
Technology Trends
Evaluate New Designs and Organizations

Workloads
Design

Performance

- How do you measure performance?
- Throughput
  - Number of tasks completed per time unit
- Response time/Latency
  - Time taken to complete a task
- Which measure chosen depends on user community
  - System admin vs single user submitting homework

Measurement Tools

- Benchmarks, Traces, Mixes
- Hardware: Cost, delay, area, power estimation
- Simulation (many levels)
  - ISA, RT, Gate, Circuit
- Queuing Theory
- Rules of Thumb
- Fundamental “Laws”/Principles
The Bottom Line: Performance (and Cost)

- Time to run the task (ExTime)
  - Execution time, response time, latency
- Tasks per day, hour, week, sec, ns ... (Performance)
  - Throughput, bandwidth

### Boeing 747
- DC to Paris: 6.5 hours
- Speed: 610 mph
- Passengers: 470
- Throughput (pmph): 286,700

### BAD/Sud Concorde
- DC to Paris: 3 hours
- Speed: 1350 mph
- Passengers: 132
- Throughput (pmph): 178,200

"X is n times faster than Y" means

\[
\frac{\text{ExTime}(Y)}{\text{Performance}(X)} = \frac{\text{ExTime}(X)}{\text{Performance}(Y)}
\]

- Speed of Concorde vs. Boeing 747
- Throughput of Boeing 747 vs. Concorde

Amdahl’s Law: Speedup

- Application takes X time
- How to run it faster
  - Enhance/optimize a portion of it
  - Which portion
  - Can we enhance all of it
  - Note that we are talking of solving the enhanced part in a different way, and possibly using different (more costly) resources

\[
\text{ExTime}_{\text{new}} = \text{ExTime}_{\text{old}} \times \left(1 - \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}} \right) + \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}
\]

\[
\text{Speedup}_{\text{overall}} = \frac{\text{ExTime}_{\text{old}}}{\text{ExTime}_{\text{new}}} = \frac{1}{1 - \frac{\text{Fraction}_{\text{enhanced}}}{\text{Speedup}_{\text{enhanced}}}}
\]
**Principles of Computer Architecture Design**

- Common case fast
  - Focus on improving those instructions that are frequently used
  - Even optimizing compilers end up using only 10% of instructions in ISA
- Amdahl’s Law
  - Parts of program that cannot be enhanced
- Locality
  - Spatial
  - Temporal

**Locality**

- 90% time spent on 10% of code
- Temporal Locality
  - If you use something then you will use it again soon
- Spatial Locality
  - If you use something then you will use something nearby
- Examples: Word proc, CAD
  - 80% of program instructions executed were from 3-5% of the code
  - 90% of inst. executed were from 9-12% code

**Architecture Design: Summary**

- Design to last through trends
- Understand the principles
  - Make common case fast
  - Amdahl’s law
  - Locality

**Emerging trends in Processor Design**

- CISC to RISC
  - Based on speeding up common instructions
- What’s the trend in Semiconductor technology and its impact on new types of processor architectures?
  - Some aspects to consider:
    - Delay: switching time of transistor – impacts clock cycle
    - Feature size: size of transistor – impacts amount of logic in processor
    - Interconnect delay: clock cycle/delay in sending signal across the interconnect lines on a chip
Impact Of Decreasing Feature Size

- Interconnect delay greater has impact than gate delays
- "...wires are not keeping pace with scaling of other features. In fact, for CMOS processes below 0.25 micron ... an unacceptably small percentage of the die will be reachable during a single clock cycle."
- "Architectures that require long-distance, rapid interaction will not scale well ..."
  - "Will Physical Scalability Sabotage Performance Gains?" D.Matzke, Chief architect TI, IEEE Computer (9/97)

As Wire Delays Become Significant...

- Focus on architectures that
  - do not involve long distance communication
  - distribute control and data processing logic

Verification And Test

- With increasing chip complexity, verification and test costs form a significant component of the overall cost
  - Based on trends in previous slide
  - Scaling current superscalar architectural techniques is likely to exacerbate the test and verification cost factor
  - Long testing process will also affect time to market
  - Impact of high costs?
    - Keep architecture simple and regular
Available Instruction-level parallelism

(Wall '93, DECWRL)

<table>
<thead>
<tr>
<th>Application</th>
<th>Perfect Model</th>
<th>Superb Model</th>
<th>Good Model</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>12</td>
<td>13</td>
</tr>
</tbody>
</table>

CS 211: Computer Architecture, Bhagi Narahari

From Previous Slides...

- Lots of hardware parallelism available
  - can accommodate approx. 50 pentiums on one die in few years

However,

- Conventional architectures and compilation
  - cannot expose enough parallelism in applications
  - even the “superb” model yields an ILP < 10 on average

- Need for new architectures and compilation techniques!

Emerging Architecture Directions

- Reconfigurable Processors—better for special purpose applications
  - let compiler handle everything
  - no commitment to a particular architecture
  - compiler generates architecture and code for it

- Explicitly Controlled Architectures (Very Large Instruction Word - VLIW)
  - simplify architectures as much as possible
  - architectural template is a known, conventional one
  - compiler handles a lot of processor’s decision making
    - explicitly control issue, scheduling, allocation
  - Explicitly Parallel Instruction Computing (EPIC)
    - subset of explicitly controlled architectures

Explicitly Parallel Processors

- Lots of hardware parallelism available
  - build processors that execute multiple instructions in parallel.
- Intel’s IA-64 project, Itanium Processor
- As we shall see, compiler plays even more important role
Sequential Processor
Sequential Instructions
Processor
Execution unit

Instruction Level Parallelism: Shrinking of the Parallel Processor
- Put multiple processors into one chip
- execute multiple instructions in each cycle
- move from multiple processor architectures to multiple issue processors
- Two classes of Instruction Level Parallel (ILP) processors
  - Superscalar processors
  - Explicitly Parallel Instruction Computers (EPIC)
    - also known as Very Large Ins Word (VLIW)

ILP Processors: Superscalar
Sequential Instructions
Superscalar Processor
Scheduling Logic
Instruction scheduling/parallelism extraction done by hardware

ILP Processors: EPIC/VLIW
Serial Program (C code)
Compiler
Scheduled Instructions
EPIC Processor
Compiler vs. Processor

Compiler

Hardware

Superscalar

Dataflow

Determine Dependences

Determine Independences

Bind Operations to Function Units

Bind Transports to Busses

Execute

Determine Dependences

Determine Independences

Bind Operations to Function Units

Bind Transports to Busses

The Embedded Processor

What?

A programmable processor whose programming interface is not accessible to the end-user of the product. The only user-interaction is through the actual application.

Examples:

- Sharp PDA’s are encapsulated products with fixed functionality
- 3COM Palm pilots were originally intended as embedded systems. Opening up the programmers interface turned them into more generic computer systems.

Some Interesting numbers

- The Intel 4004 was intended for an embedded application (a calculator)
- Of today's microprocessors
  - 95% go into embedded applications
  - SH744 (Hitachi): best-selling RISC microprocessor
  - 50% of microprocessor revenue stems from embedded systems
- Often focused on particular application area
  - Microcontrollers
  - DSPs
  - Media Processors
  - Graphics Processors
  - Network and Communication Processors

Summary: What's up with Architecture Trends?

- Moore's law: density doubles every 18 months
  - smaller processors, faster clocks
  - leads to more powerful and smaller processors!
  - Small computing platforms like Palmtop computers, Palm, WinCE
  - what next?
- Network technology
  - communicating processors
  - future is in a network of embedded processors
  - program from a central location
  - switch on microwave from office; examine refrigerator contents through web
- BUT … programming these is a *@&#&&!
  - Software and compiler support needed!!!

Next...

- Computational Model for ILP
- Instruction Scheduling problem in ILP
- Pipelining as a simple Instruction Level Parallel (ILP) Processor architecture
  - Introduction to Pipelining
  - Control of Pipelines (Instruction Scheduling)
- Superscalar and VLIW/EPIC

Next class...

- Review Computer Organization materials placed on course website and Chapter 2.
- Read intro to Simple Scalar simulator
  - www.simplescalar.com
- Topics for Next class:
  - Chap 2, Appendix A
Review: Computer Organization Basics

- What are the components of a CPU
- What is a microarchitecture level?
- What is an ISA - Instruction set architecture?
- How does a sample processor design look?
  - A simple processor architecture
- what is the basic concept of pipelining