U. Wisconsin CS/ECE 552
Introduction to Computer Architecture

Prof. Karu Sankaralingam

Introduction (Chapter 1)

www.cs.wisc.edu/~karu/courses/cs552/

Slides combined and enhanced by Karu Sankaralingam from work by Falsafi, Hill, Marculescu, Nagle, Patterson, Roth, Rutenbar, Schmidt, Shen, Sohi, Sorin, Thottethodi, Vijaykumar, & 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…
Role of Building Architect

Materials
- Steel
- Concrete
- Brick
- Wood
- Glass

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

Construction

Buildings
- Houses
- Offices
- Apartments
- Stadiums
- Museums

Design

Plans
Role of Computer Architect

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

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

Plans

Design

Manufacturing

Computer
- PCs
- Servers
- PDAs
- Mobile Phones
- Supercomputers
- Game Consoles
- Embedded

(4) CS/ECE 552 (Sankaralingam)
CS/ECE 552 in Context

- CS/ECE 352 – Gates to multiplexors & FSMs
- CS/ECE 354 – High-level language to machine language, a.k.a. Instruction Set Architecture (ISA)

- This course – CS/ECE 552 – puts it all together
  - Implement that logic that provides ISA
  - Must do datapath & control, but no magic!
  - Manage complexity through ABSTRACTION

- Follow-on courses explore trade-offs: 752 & 757

CS/ECE 552 (Sankaralingam)
Why Study Computer Design

• To design new computers: old designs become obsolete fast
  – new technologies - e.g., denser ICs (technology “push”)
  – new user demand - e.g., virtual reality (application “pull”)
• To be an informed user
  – a little auto mechanics helps owner rarely, but importantly
• To learn to deal with complexity via abstraction
  – problems that take months and years to complete
Abstraction

• Black boxes
• Difference between interface and implementation
  – Interface - WHAT something does
  – Implementation - HOW it does so
Abstraction - Example

• 2-to-1 Mux

• Interface:

  S

  X

  Y

  \[ S=0? \ O=X \]

  \[ : \ O=Y \]

  O

• Implementations
  – gates (fast or slow), pass transistors
What’s the Big Deal?

- E.g., x86 interface book
- Worse for computers, in general - a tower of abstraction
  - Application software
  - System software (OS and compiler/assembler/linker)
  - Hardware (CPU, memory, I/O)
- Each interface is complex and implemented with layer below
  - Abstraction keeps unnecessary details hidden
- Thousands of engineers to build one product
Basic Division of Hardware

- In space and time
  - In space
Basic Division of Hardware

• In time
  – Fetch the instruction from memory \texttt{add r1, r2, r3}
  – Decode the instruction - what does this mean? \texttt{read r2, r3}
  – Read input operands \texttt{add}
  – Perform operation \texttt{write to r1}
  – Write results \texttt{pc := pc + 4}
  – Determine next instruction
Why don’t old designs work?

- Evolutionary and revolutionary changes in technology

<table>
<thead>
<tr>
<th>Date</th>
<th>What</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1947</td>
<td>1st transistor</td>
<td>Bell Labs</td>
</tr>
<tr>
<td>1958</td>
<td>1st integrated circuit</td>
<td>Texas Instruments</td>
</tr>
<tr>
<td>1971</td>
<td>1st microprocessor Intel 4004</td>
<td>2300 transistors, 108 kHz</td>
</tr>
<tr>
<td>1978</td>
<td>Intel 8086</td>
<td>29K Transistors</td>
</tr>
<tr>
<td>1989</td>
<td>Intel 80486</td>
<td>1.2M Transistors</td>
</tr>
<tr>
<td>1995</td>
<td>Intel Pentium Pro</td>
<td>5.5M Transistors</td>
</tr>
<tr>
<td>2003</td>
<td>Intel Pentium4</td>
<td>55M Transistors</td>
</tr>
</tbody>
</table>
Moore’s Law(s)

• Technologists will double # transistors per chip doubles every two years (or 18 months)

• Or architects will double performance per chip doubles every two years (or 18 months)

• These can’t go on forever, but don’t underestimate a trillion dollar industry
First Microprocessor

- Intel 4004 [1971]
  - 4-bit data
  - 2300 transistors
  - 10 mm PMOS
  - 108 KHz (.0001 GHz)
  - 12 V
  - 13 mm$^2$
Some Older Chips!

Intel Pentium IV
- 42 million transistors
- 3.8 GHz
- 0.13μm process
- Could fit ~15,000 4004s on this chip!

IBM Cell
- 8 vector processors + 1 PPC
- 4 GHz
- 90nm process

Intel Itanium II (Montecito)
- 1.7 billion transistors
- ?? MHz
- 90nm process
Intel Ivy Bridge

3rd Generation Intel® Core™ Processor: 22nm Process

New architecture with shared cache delivering more performance and energy efficiency
GPUs: AMD Radeon 4870
GPUs: Nvidia GTX 280
A6 Chip
Bottom line

- Designers must know BOTH software and hardware
- Compilers, Operating Systems, Networks
- Both contribute to layers of abstraction of computers
- IC costs and performance
- Read the book - Chapter 1 done -
- Throughout the course, read the book (BEFORE lecture)
- Optional reading:
  - Soul of a New Machine, Trace Kidder
Syllabus

- Language of the computer: ISA
- Arithmetic
- Processor Design
- Performance
- Memory
- IO
- Multiprocessors, Advanced processors, GPUs
We will meet in 1221 CS henceforth