Computer Architecture

- Rely on abstraction layers to manage complexity
  - Von Neumann Machine
Technology advances at an astounding rate—

- 19th century: attempts to build mechanical computers
- Early 20th century: mechanical counting systems (cash registers, etc.)
- Mid 20th century: vacuum tubes as switches
- Since: transistors, integrated circuits

- 1965: Moore’s law (Gordon Moore)
  - Predicted doubling of IC capacity every 18 months
  - Has held for five decades, appears to be slowing down

- Drives functionality, performance, cost
  - Exponential improvement for 50+ years
# Semiconductor History

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</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 IC</td>
<td>Jack Kilby (MSEE ’50) @TI Winner of 2000 Nobel prize</td>
</tr>
<tr>
<td>1971</td>
<td>1st microprocessor</td>
<td>Intel (calculator market)</td>
</tr>
<tr>
<td>1974</td>
<td>Intel 4004</td>
<td>2300 transistors</td>
</tr>
<tr>
<td>1978</td>
<td>Intel 8086</td>
<td>29K transistors</td>
</tr>
<tr>
<td>1989</td>
<td>Intel 80486</td>
<td>1M transistors</td>
</tr>
<tr>
<td>1995</td>
<td>Intel Pentium Pro</td>
<td>5.5M transistors</td>
</tr>
<tr>
<td>2006</td>
<td>Intel Montecito</td>
<td>1.7B transistors</td>
</tr>
<tr>
<td>2015</td>
<td>Oracle SPARC M7</td>
<td>10B+ transistors</td>
</tr>
</tbody>
</table>
Computer Architecture

• Instruction Set Architecture (IBM 360)
  – ... the attributes of a computing system as seen by the programmer. I.e. the conceptual structure and functional behavior, as distinct from the organization of the data flows and controls, the logic design, and the physical implementation. -- Amdahl, Blaaw, & Brooks, 1964

• Machine Organization (microarchitecture)
  – ALUS, Buses, Caches, Memories, etc.

• Machine Implementation (realization)
  – Gates, cells, transistors, wires
752 In Context

• Prior courses
  – 352 – gates up to multiplexors and adders
  – 354 – high-level language down to machine language interface or instruction set architecture (ISA)
  – 552 – implement logic that provides ISA interface
  – CS 537 – provides OS background (co-req. OK)
• This course – 752 – covers advanced techniques
  – Modern processors that exploit ILP
  – Modern memory systems that exploit MLP
• Additional courses
  – ECE 757 covers parallel and multiprocessing
  – ECE 755 covers VLSI design
Why Take 752?

• To become a computer designer
  – Alumni of this class helped design your computer
• To learn what is *under the hood* of a computer
  – Innate curiosity
  – To better understand when things break
  – To write better code/applications
  – To write better system software (O/S, compiler, etc.)
• Because it is intellectually fascinating!
  – What is the most complex man-made single device?
Computer Architecture

• Exercise in engineering tradeoff analysis
  – Find the fastest/cheapest/power-efficient/etc. solution
  – Optimization problem with 100s of variables
• All the variables are changing
  – At non-uniform rates
  – With inflection points
  – Only one guarantee: Today’s right answer will be wrong tomorrow
• Two high-level effects:
  – Technology push
  – Application Pull
Technology Push

• What do these two intervals have in common?
  – 1776-1999 (224 years)
  – 2000-2001 (2 years)

• Answer: Equal progress in processor speed!

• The power of exponential growth!

• Driven by Moore’s Law
  • Devices per chip doubles every 18-24 months

• Computer architects turn additional resources into
  • Speed
  • Power savings
  • Functionality
Performance Growth

Unmatched by any other industry!
[John Crawford, Intel]

• **Doubling every 18 months (1982-1996): 800x**
  – Cars travel at 44,000 mph and get 16,000 mpg
  – Air travel: LA to NY in 22 seconds (MACH 800)
  – Wheat yield: 80,000 bushels per acre

• **Doubling every 24 months (1971-1996): 9,000x**
  – Cars travel at 600,000 mph, get 150,000 mpg
  – Air travel: LA to NY in 2 seconds (MACH 9,000)
  – Wheat yield: 900,000 bushels per acre
Technology Push

• Technology advances at varying rates
  – E.g. DRAM capacity increases at 60%/year
  – But DRAM speed only improves 10%/year
  – Creates gap with processor frequency!

• Inflection points
  – Crossover causes rapid change
  – E.g. enough devices for multicore processor (2001)

• Current issues causing an “inflection point”
  – Power consumption
  – Reliability, variability
  – Packaging innovations
Application Pull

• Corollary to Moore’s Law:
  Cost halves every two years
  
  *In a decade you can buy a computer for less than its sales tax today.* –Jim Gray

• Computers cost-effective for
  – National security – weapons design
  – Enterprise computing – banking
  – Departmental computing – computer-aided design
  – Personal computer – spreadsheets, email, web
  – Mobile computing – GPS, location-aware, ubiquitous
  – Wearable computing – activity/health monitoring, etc.
  – Voice web search
Application Pull

• What about the future?
  – For many modeling applications, scaling up resolution blows up computational demand (e.g. weather)
  – Machine learning: model size increases seem to keep providing better and better accuracy

• Must dream up applications that are not cost-effective today
  – Realism in games and virtual worlds (graphics, physics, AI)
  – Virtual reality (Hololens), telepresence
  – Big data analytics, large-scale optimization
  – Personal assistants (AI/ML)
  – Image & video processing, analysis, contextual semantics

• This is your job!

[Canziani et al., 2016]
Trends

- Moore’s Law for device integration [source: Intel]
- Chip power consumption
- Single-thread performance trend
Dynamic Power

\[ P_d \approx \sum_{i \in U} k_i C_i V^2 A_i f \]

- Static CMOS: current flows when active
  - Combinational logic evaluates new inputs
  - Flip-flop, latch captures new value (clock edge)
- Terms
  - C: capacitance of circuit
    - wire length, number and size of transistors
  - V: supply voltage
  - A: activity factor
  - f: frequency
- Future: Fundamentally power-constrained
Multicore Mania

- First, servers
  - IBM Power4, 2001
- Then desktops
  - AMD Athlon X2, 2005
- Then laptops
  - Intel Core Duo, 2006
- Cellphones
  - Dual/quad/octo, big.LITTLE
Why Multicore

<table>
<thead>
<tr>
<th></th>
<th>Single Core</th>
<th>Dual Core</th>
<th>Quad Core</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core area</td>
<td>A</td>
<td>~A/2</td>
<td>~A/4</td>
</tr>
<tr>
<td>Core power</td>
<td>W</td>
<td>~W/2</td>
<td>~W/4</td>
</tr>
<tr>
<td>Chip power</td>
<td>W + O</td>
<td>W + O’</td>
<td>W + O’’</td>
</tr>
<tr>
<td>Core performance</td>
<td>P</td>
<td>0.9P</td>
<td>0.8P</td>
</tr>
<tr>
<td>Chip performance</td>
<td>P</td>
<td>1.8P</td>
<td>3.2P</td>
</tr>
</tbody>
</table>

Mikko Lipasti -- University of Wisconsin
Amdahl’s Law

\[ \text{Speedup} = \frac{1}{(1 - f) + \frac{f}{n}} \]

\[ \lim_{{n \to \infty}} \frac{1}{1 - f + \frac{f}{n}} = \frac{1}{1 - f} \]

\text{f – fraction that can run in parallel}
\text{1-f – fraction that must run serially}
Fixed Chip Power Budget

- **Amdahl’s Law**
  - Ignores (power) cost of n cores

- **Revised Amdahl’s Law**
  - More cores → each core is slower
  - Parallel speedup < n
  - Serial portion (1-f) takes longer
  - Also, interconnect and scaling overhead
Fixed Power Scaling

- Fixed power budget forces slow cores
- Serial code quickly dominates

Chip Performance vs. # of cores/chip

- 99.9% Parallel
- 99% Parallel
- 90% Parallel
- 80% Parallel

Mikko Lipasti -- University of Wisconsin
Focus of this Course

• How to make serial portion fast
  – Fast serial portion also helps parallel portion!

• State-of-the-art processor design
  – Pipelining review (online lectures)
  – Superscalar, out-of-order processors
  – Branch prediction

• Advanced memory systems
  – Cache review (online lecture)

• Multicore and multithreaded processors
Instruction Set Processing

The ART and Science of Instruction-Set Processor Design
[Gerrit Blaauw & Fred Brooks, 1981]

**ARCHITECTURE** (ISA) programmer/compiler view
- Functional appearance to user/system programmer
- Opcodes, addressing modes, architected registers, IEEE floating point

**IMPLEMENTATION** (μarchitecture) processor designer view
- Logical structure or organization that performs the architecture
- Pipelining, functional units, caches, physical registers

**REALIZATION** (Chip) chip/system designer view
- Physical structure that embodies the implementation
- Gates, cells, transistors, wires
Iron Law

Processor Performance = \frac{\text{Time}}{\text{Program}}

= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}

(code size) \quad (\text{CPI}) \quad (\text{cycle time})

Architecture --> Implementation --> Realization

Compiler Designer    Processor Designer    Chip Designer
Iron Law

• Instructions/Program
  – Instructions executed, not static code size
  – Determined by algorithm, compiler, ISA

• Cycles/Instruction
  – Determined by ISA and CPU organization
  – Overlap among instructions reduces this term
  – Constrained by energy per instruction (EPI)

• Time/cycle
  – Determined by technology, organization, clever circuit design
  – Constrained by power limitations
Our Goal

- Minimize time, which is the product, NOT isolated terms

- Common error to miss terms while devising optimizations
  - E.g. ISA change to decrease instruction count
  - BUT leads to CPU organization which makes clock slower
    - Reduced CPI causes large increase in EPI

- Bottom line: terms are inter-related
Textbooks

• **Recommended course textbook:**

• **Recommended textbook:**
Expected Background

• ECE/CS 552 or equivalent
  – Design simple uniprocessor
  – Simple instruction sets
  – Organization
  – Datapath design
  – Hardwired/microprogrammed control
  – Simple pipelining
  – Basic caches

• High-level programming experience
  – C/UNIX skills – modify simulators
Course Context

• Assume canonical RISC ISA
  – Register-register ALU ops
  – Load from memory (cache)
  – Store to memory
  – Branches, jumps, calls, returns

• Modern CISC (x86) processors
  – Translate to equivalent primitives
    • Later: how the translation is done
About This Course

• Readings and Paper Reviews
  – Will be posted on website (one list for each midterm)
  – Make sure you keep up with these! Not necessarily discussed in lecture.

• Lecture
  – Attendance required
  – Some lectures will be delivered online
  – Overscheduled in first half; will cancel many lectures in 2nd half

• Homework
  – Homework assigned but not graded
  – Learning tool to help prepare for midterm
About This Course

• Pop Quizzes
  – Not announced ahead of time
  – Will drop one for final grade to accommodate occasional absence
  – Make sure you are ahead on readings!

• Exams
  – Midterm 1: Wed 10/25 in class
  – Midterm 2: Wed 12/20 10:05am-12:05pm (final exam time slot)
  – Keep up with reading list!
About This Course

• Course Project
  – Research project
    • Replicate results from a paper
    • Or attempt something novel

• Final project includes a written report and an oral presentation
  – Proposal due 10/30
  – Progress report due 11/22
  – Presentations during class time 12/11, 12/13
  – Final reports due 12/13
About This Course

• Grading
  – Quizzes & paper reviews 20%
  – Midterm 1 25%
  – Midterm 2 25%
  – Project 30%

• Web Page (check regularly)
  – http://ece752.ece.wisc.edu
About This Course

• Office Hours
  – Prof. Lipasti: EH 3621, TBD
  – Or, catch me after class

• Communication channels
  – E-mail to instructor, class e-mail list
    • compsci752-1-f17@lists.wisc.edu
  – Web page
  – Office hours
About This Course

• Other Resources
  – Computer Architecture Colloquium – Tuesday 4-5PM, 1221 CSS
  – Computer Engineering Seminar – Friday 12-1PM, EH4610
About This Course

• Lecture schedule:
  – MWF 11:00-12:15
  – Cancel approx. 1 of 3 lectures, mostly in second half of semester
  – Allows us to get ahead on topics to enable broader range for project work
<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Introduction, Technology challenges</td>
</tr>
<tr>
<td>1</td>
<td>Superscalar Organization</td>
</tr>
<tr>
<td>2</td>
<td>Instruction Flow</td>
</tr>
<tr>
<td>3</td>
<td>Register Data Flow</td>
</tr>
<tr>
<td>4</td>
<td>Memory Data Flow</td>
</tr>
<tr>
<td>5</td>
<td>Advanced Register Data Flow</td>
</tr>
<tr>
<td>6</td>
<td>Case Studies</td>
</tr>
<tr>
<td>7</td>
<td>Midterm 1 in-class on 10/25, Case Studies</td>
</tr>
<tr>
<td>8</td>
<td>Advanced Memory Hierarchy</td>
</tr>
<tr>
<td>9</td>
<td>Multiple threads, Case studies</td>
</tr>
<tr>
<td>10</td>
<td>Advanced topics</td>
</tr>
<tr>
<td>11</td>
<td>Lecture canceled, project work</td>
</tr>
<tr>
<td>12</td>
<td>Lecture canceled, project work</td>
</tr>
<tr>
<td>13</td>
<td>Lecture canceled, project work</td>
</tr>
<tr>
<td>14</td>
<td>Project talks, Course Evaluation, Final reports</td>
</tr>
<tr>
<td>Finals</td>
<td>Midterm 2 Wednesday 12/20 10:05pa</td>
</tr>
</tbody>
</table>
Wrapping Up

• Next lecture on technology challenges
  – Sets the stage for the whole course

• View review lecture online
  – Pipelining Review, 2 lectures with audio narration
  – http://ece752.ece.wisc.edu

• Reading list and review schedule on web page

• Be prepared for discussion/pop quiz

Final thought:

*Talking about music is like dancing about architecture.*
(Thelonious Monk)