Introduction

Nima Honarmand

CSE 502 — CompArch (1)

- Computer Architecture is
  … the ***science*** and ***art*** of selecting (or designing) and interconnecting hardware and software components to ***create computers*** …

- Computer Architecture has changed a lot in the past 10-15 years
  - When I was a student, it was mostly about superscalar, out-of-order pipelines and cache coherence
  - Now we have many-cores, GPUs, data centers and accelerators

CSE 502 — CompArch (2)

- These changes happened because we hit multiple walls in processor/system design
  - ILP wall
  - Power wall
  - Memory wall
  - Network wall (recently)
- Also because application domains emerged whose demands couldn't be met with conventional processors
  - AR and VR
  - LTE and future mobile protocols
  - Big data
  - Data driven science
  - Deep learning
  - etc.

CSE 502 — CompArch (3)

- In this course, first we learn what conventional processors look like internally
  - Instruction sets
  - Pipelining
  - Processor front-end (instruction fetch and decode)
  - Processor back-end (execution core)
  - Memory hierarchy (caches, DRAM, prefetching, etc.)
  - Superscalar and out-of-order execution
  - Branch prediction and speculation
  - Vector execution
  - …

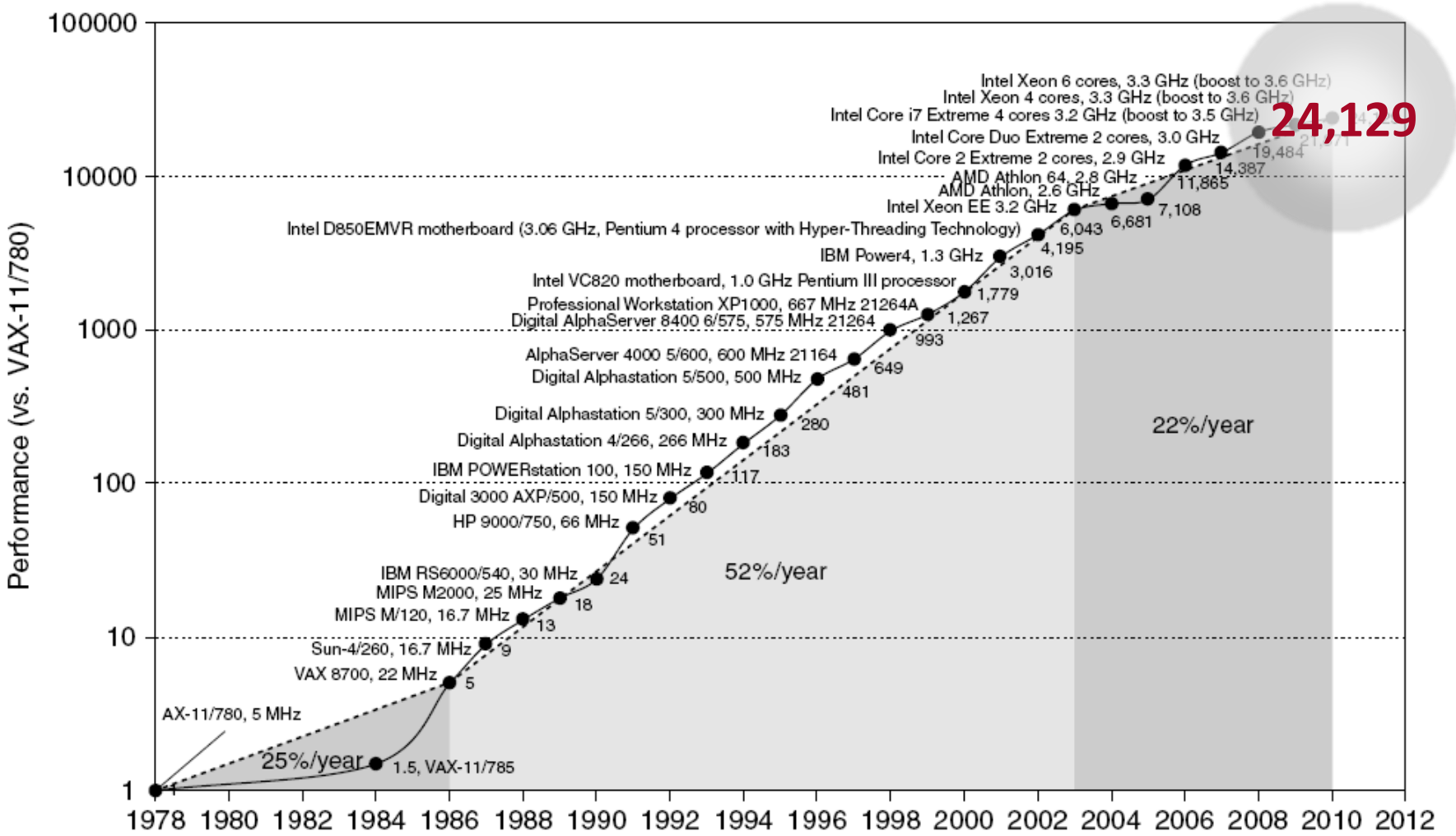
CSE 502 — CompArch (4)

- Then some more recent stuff (circa ~2000)
  - Multi-cores and multi-processors
  - Hardware multi-threading
  - Cache coherence and consistency
- And finally cutting-edge topics (as many as time permits)
  - GPUs
  - Warehouse Scale Computers
  - Accelerators
  - Etc.

CSE 502 — CompArch (5)

- "Computer Architecture" is an umbrella term
  - *Interface* exposed to system software (compiler and OS)
    - In particular, Instruction Set Architecture (ISA) of processors
  - *Micro-architecture*: internal organization of components
- This course is mostly about ***micro-architecture***
  - E.g., what's inside the processor (CPU)
  - What implications this has on software

Why Study CompArch (1)



Why Study CompArch (2)

Sources of performance improvement:

- Improvements in semi-conductor technology
  - Faster transistors
  - More transistors
- Improvements in computer architecture
  - *Computer architects work to turn the additional resources into speed/power savings and new functionality*

In this class, we will study some of the cool techniques invented by computer architects to make this possible!

Why Take CSE 502?

- You need one more qualifier/graduation requirement

✗ ***Bad answer!***

- You want to become a computer architect
- You want to learn what's inside a processor
  - Because you're curios (and there is no computer w/o a processor)
  - To write better/faster application code
  - To write system software (OS, compiler, etc.)
- Computer architecture is cool and intellectually fascinating
  - BTW, what is the most complicated man-made artifact?
  - Consider: there are billions of individually designed and verified transistors in a modern processor chip

✓ ***More like it!***

Course Format

- I will deliver most lectures
- There might be a few student presentations
  - Mostly on newer topics
- This is a project-intensive course
  - Learn why things are the way they are, first hand
  - We will “build” the behavioral model of a real CPU

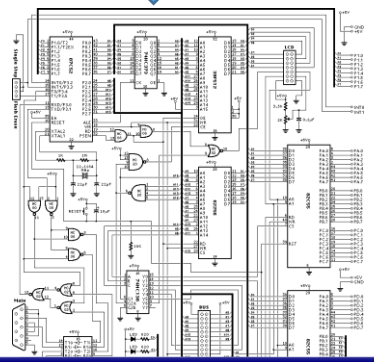
Hardware Design Process

Conceptual Design

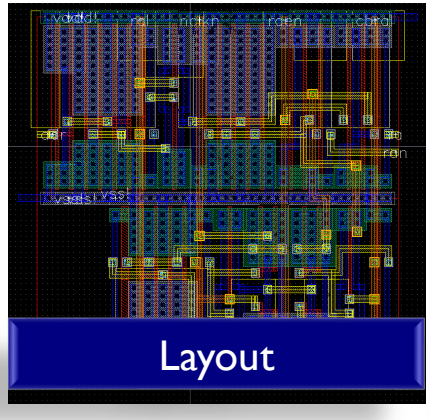
Behavioral Implementation

CSE 502

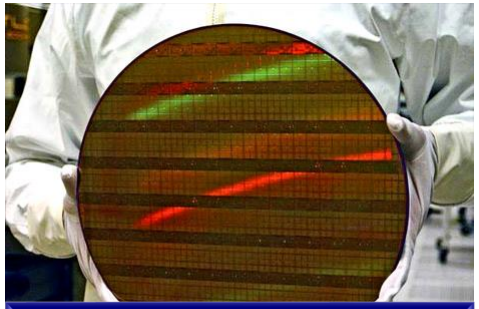
Evaluation



Structural Implementation



Layout



Manufacturing

Packaging

Course Project

- Goal: design and implement a SPARCv8 processor
  - Preferably a super-scalar, out-of-order one ☺
- We'll use SystemVerilog HDL for implementation
  - Don't panic! We'll cover the necessary background
  - Hopefully, will help you think and design like a HW designer
- I'll provide a cross-compiler and a simulation environment
  - You'll design and implement the processor
  - See course webpage for details
- No intermediate milestones for the course project
  - This is a graduate course; you should be self-disciplined
  - This flexibility is a rope; you can use it to climb or hang yourself

Grading

What?	Points
Quiz 0	0
Quiz 1	10
2 Homeworks	10 each
Course Project	Up to 125
Midterm	20
Final	20
Presentation (optional)	10
Total	**205**

Course Project	Points
5-Stage pipeline + direct-mapped caches	40
5-Stage pipeline + set-associative caches	45
Above + super-scalar pipeline	60
Above + out-of-order execution	80
Multi-cycle divider and pipelined multiplier on top of any of the above	5 extra
Branch prediction and speculation on top of the above	10-20 extra
SMT on top of the above	10-20 extra

- Guaranteed grades: [A, A-, B+, …] = [95, 90, 85, …]
  - I may use a curve on top of this if need be

Logistics (1/3)

- Books
  - Highly Recommended (but not strictly required)
    - *Modern Processor Design: Fundamentals of Superscalar Processors* (by **Shen** & **Lipasti**)
    - *Computer Architecture: A Quantitative Approach, 6th Ed* (by **Hennessy** & **Patterson**)
  - I highly recommend reading both books cover-to-cover if you are targeting systems research
- There will be other required readings
  - SPARC Architecture Manual
  - SystemVerilog tutorials
  - And a few papers or book chapters online

Logistics (2/3)

- Two quizes
  - Quiz 0 today
    - Does not affect your grade; just to help me assess class background
    - Completion is ***required***
    - If you missed the 1st class, come to office hours for it
  - Quiz 1 (in a month or so): to force you read the SystemVerilog tutorials and SPARCv8 manual
- Exams
  - Midterm and final
  - Include everything in class lectures and discussions, blackboard posts, course project and required readings

Logistics (3/3)

- Blackboard
  - Grades will be posted there
- Course forum and newsgroup
  - Also on Blackboard
  - "General Discussions" forum on blackboard
- Working in groups: only permitted on the project
  - Groups may be up to ***2 people***
  - Should let me know of your groups by ***Feb 19***

Academic Integrity Policy

- You may...
  - Discuss assignment, design, techniques
- You may ***not***…
  - Share code
  - Copy from internet
    - Exceptions are possible, but must receive explicit permission
  - In general, every character you hand in should be written by yourself
- If caught, you'll fail. No exceptions!

Quiz 0