



Samueli
School of Engineering

CS-M151B

Computer Systems Architecture

Blaise Tine
UCLA Computer Science

Agenda

- About me
- Logistic
- Q/A
- Course Introduction

About me

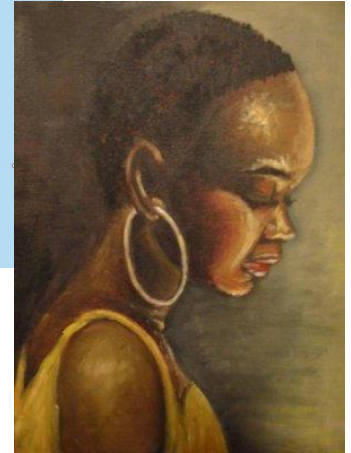
- **Name:** Blaise-Pascal Tine
- **Email:** blaisetine@cs.ucla.edu
- **DM me on campuswire**
- **Office Hours:** Fridays 4:30-6:00 PM
- **Location:** Eng VI 499
 - in-person

About me

- **Research Interests**
 - **Custom hardware extensions for GPUs**
 - Graphics, ML, security, etc...
 - ISA extensions
 - Compiler optimizations
 - **Neural Rendering Acceleration**
 - **Hardware design automation**
- **Looking for a H/W research project? – email me**
 - **Students with verilog or compiler background**

About me

- **Country of origin: Cameroon**
- **First language: French, Ngemba**
- **My hobbies:**
 - Painting: oil, acrylic, digital
 - Programming:
 - Old school: Pascal, BASIC
 - Web: HTML5, Javascript
 - Managed: Java, C#, Python
 - Native: C/C++
 - Hardware: Verilog, VHDL



Acrylic

“Coding is the art of creation – where logic meets imagination”

Teaching Assistants

- **TAs:** Jack Yu and Haiying H
- **Emails:** yiyaoyu@cs.ucla.edu and haiying1998@outlook.com
- **Office Hours:** TBD

Tell us about you

Your First Assignment:

- School/Department
- Background/Interests
- Why are you taking this course?
- **Free 1 bonus points ;-)**

Course Syllabus

- Computer Model, Abstractions, and Technologies
- Processor Design Metrics and Objectives
- Instruction Set Architecture Design
- Pipelining
- Data and Controls Hazards
- Branch Prediction
- Super-Scalar Processors and Out-of-Order Execution
- Cache and Memory Hierarchy
- Multicore architecture
- Power Management
- Advanced Topics: GPUs, Accelerators, and Custom Architectures

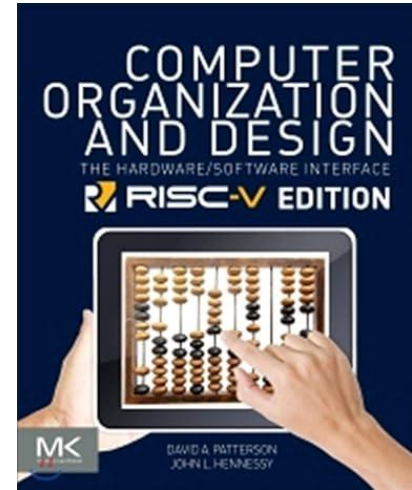
Reading Material

Book;

David A. Patterson and John L. Hennessy, [Computer Organization and Design: the Hardware/Software Interface: RISC-V Edition.](#)

Additional Course Materials:

Will be provided by the instructor.



Campuswire

- Self enrollment
- Check out Bruinlearn's announcement
- Encourage participation with bonus points
- TA and I will regularly check

Waiting List

- The class is full!
- Wait until next week if anyone drops out.

Homework

- 10 homework assignments
- Individual work
- Submitted on Friday.
- Due the following week.
- Multiple choice tests
- 2 pts each

Projects

- 3 projects
 - CPU pipeline implementation and optimization
 - Required knowledge of C++
 - 2 weeks duration
- Group project: teams of 2
 - Can be done individually if you prefer to work solo
- Encourage teamwork
 - Most learn better from peers
 - Most understand better by explaining to someone else
- Separate submission
- Must submit the same work you did with your partner
- 10 pts each, last project is 15 pts
- See Bruinlearn's Syllabus for the schedule

Exams

- Mid-term on 02/04/2026 – 20 pts
- Final on 03/19/2026 – 20 pts
- Use computer in class
- Will offer a review session

Attendance and Participation

- In class attendance
 - Check attendance via QR code
 - At least 70%
 - 5 pts
- In class participation
 - Or top 10 Campuswire participation
 - 2 bonus points
 - Use same QR code to register



Grading

- 20% Homework
- 35% Projects
- 20% Midterm
- 20% Final
- 5% Participation

- **Distribution**
 - A: 90+
 - B: 80+
 - C: 70+

Policies

- Late Assignment: -10% of the grade
 - You get 24 hours grace period to submit
- Bad submission: -5% of the grade
 - Wrong file name
 - Missing files
- Regrade
 - Must request within a week
 - Written justification
 - Be aware of full regrade!

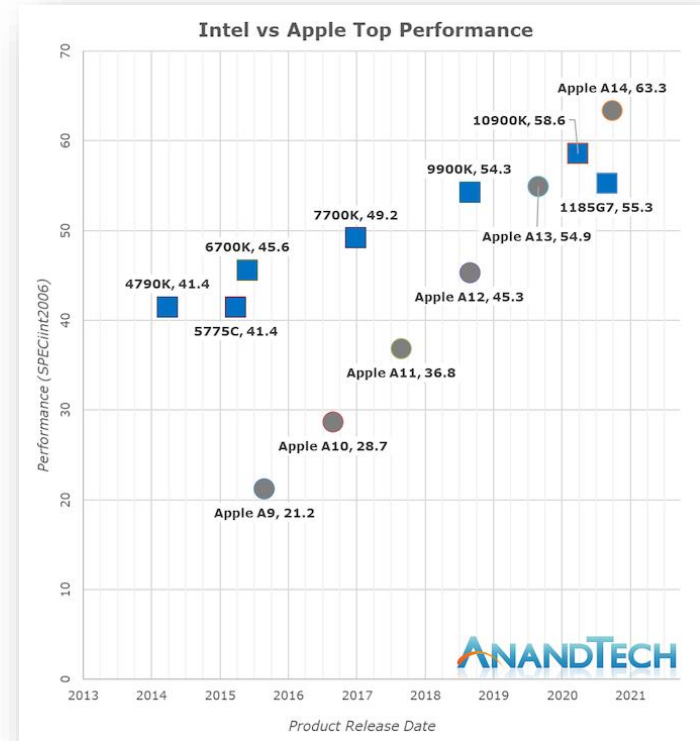
Q & A

Class Assignment

1. In which ways is the Apple M1 chip better than Intel and AMD processors?
2. How would you explain the success of the Apple M1 chip?
3. Why isn't everyone using the Apple M1?

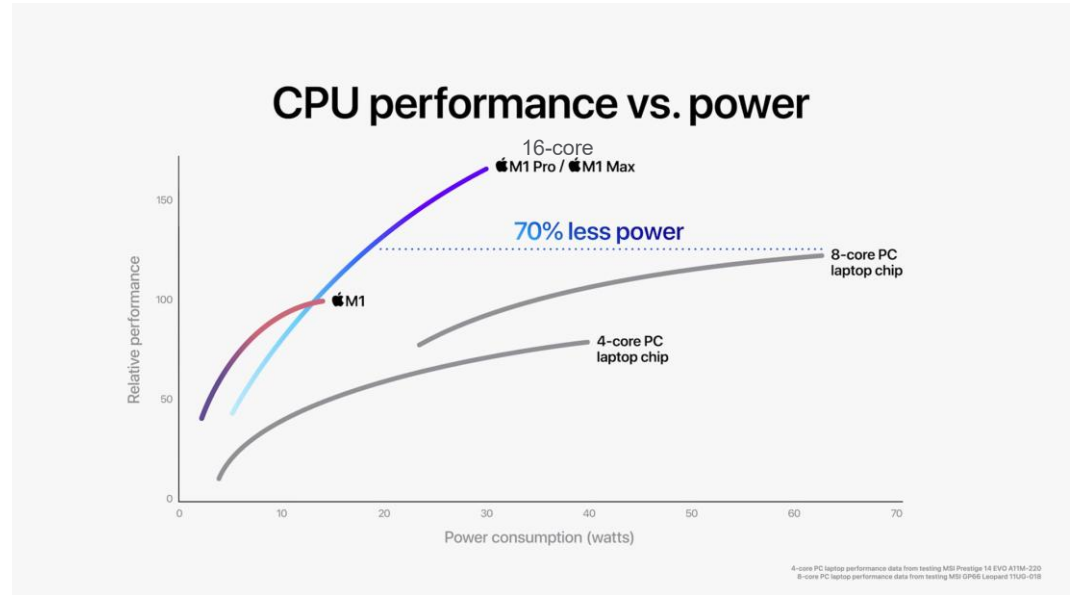
Apple Chip Progression

- From a mobile (phone) to desktop (MAC)
- Single-thread performance Increase (2015-2021)
 - Intel ~28%
 - Apple ~198% or 2.98x
- Intel still has an edge on single thread performance



Apple M1 Efficiency

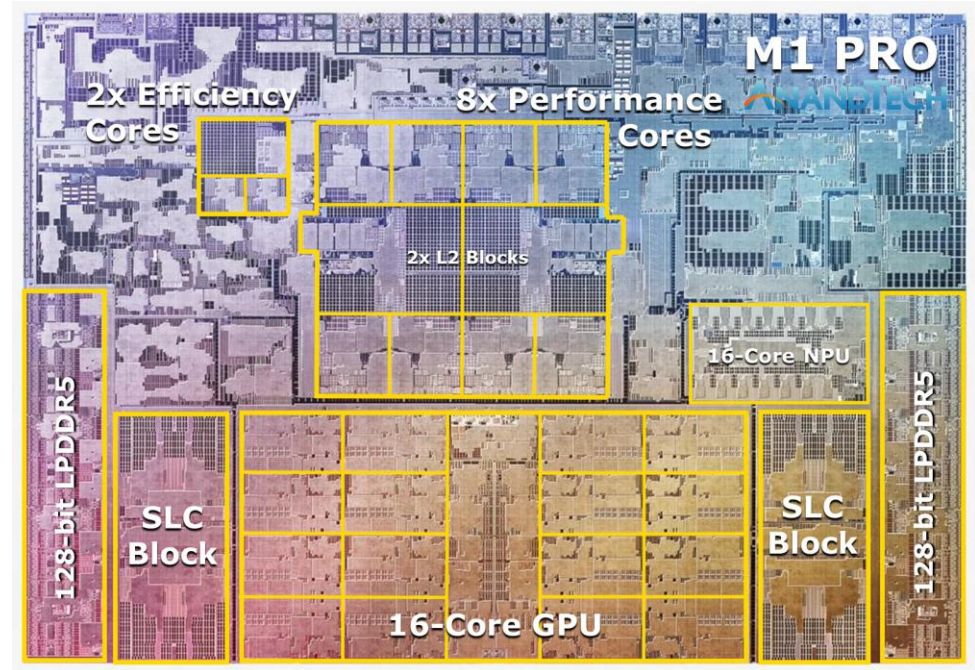
- 70% less power for same performance compared to competition!
- Performance per watt (flops per watt)
- How did they get here?



<https://www.apple.com>

Apple M1 PRO Chip layout

- Hybrid architecture
 - Efficient cores
 - Performance cores
- Neural Processing Units
- Internal DDR memory
- Shared GPU-CPU memory
- SLC (system-level cache)



Apple silicon characteristics

- Instruction set architecture (ISA)
 - ARM ISA
 - RISC architecture advantages
 - Smaller instructions set
 - Simpler pipeline
 - efficient decode and execute
 - Energy efficient processor
 - Easier compiler optimizations
 - RISC challenges
 - More instructions per work
 - e.g. memory operations

Apple silicon characteristics

- Process Technology
 - 5 nanometer for M1 chip from TSMC
 - Intel was still on 3rd generation 10 nm
 - Why it matters?
 - Higher transistor density
 - More cores per area => more flops per area
 - Higher clock speed due to shorter delay

Apple silicon characteristics

- Integrated components
 - Reduce communication overhead
 - Increase energy efficiency
 - Increase communication bandwidth
 - Bypass pins shortage
 - Integrated GPU
 - Integrated SSD controller

Apple silicon characteristics

- Near Memory Computing
 - Integrated Memory
 - Reduce memory latency
 - Reduce communication overhead
 - Increase energy efficiency
 - Increase communication bandwidth
 - Bypass pins shortage

Apple silicon characteristics

- Hardware specialization
 - Move common expensive tasks into silicon
 - Increase energy efficiency
 - Neural processing units (NPUs) (AI)
 - Media encode/decode engines (video, images)

Apple silicon characteristics

- Heterogenous multicore architecture
 - Little cores and big cores
 - high-performance "Firestorm" cores
 - high-efficiency "Icestorm" cores
 - optimize for performance or power efficiency as needed

Apple silicon characteristics

- Microarchitecture optimizations
 - Unified memory
 - CPU and GPU share the same memory pool
 - Eliminates unnecessary copies
 - Larger pool increases capacity / utilization

Apple silicon characteristics

- Hardware-software codesign
 - Apple own both the hardware and system stack
 - allows for tight integration
 - Controlled ecosystem of critical applications
 - Profiling and optimizations

The Classic CPU Performance Equation

- CPU time = Instruction Count x CPI x Clock Cycle Time
 - CPI = Clock Cycle per Instruction
- CPU time = $\frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$
- How do you increase performance?
 - Reducing the CPU time

Reducing CPU Time

- CPU time = $\frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$
- Reduce Instruction Count
 - Software, compiler, ISA
- Reduce CPI
 - Microarchitecture
- Increase Clock Rate
 - Process technology
 - Hardware design efficiency (propagation delay)

Dennard's (MOSFET) Scaling

- Dynamic Power = $\alpha * CFV^2$
 - Alpha = transistor switching rate (represents how active the transistor is)
 - C = capacitance
 - F = frequency
 - V = voltage
- Dennard's scaling (1974)
 - As transistors get smaller the capacitance and voltage decreases
 - So, you can increase frequency while keeping the power constant
 - Higher frequency means higher performance (CPU Time decreases)

Moore's law

- Gordon Moore predicted in 1965's the doubling of transistors count every two years.
- His prediction was fairly accurate in the decades that followed up to the 2010's

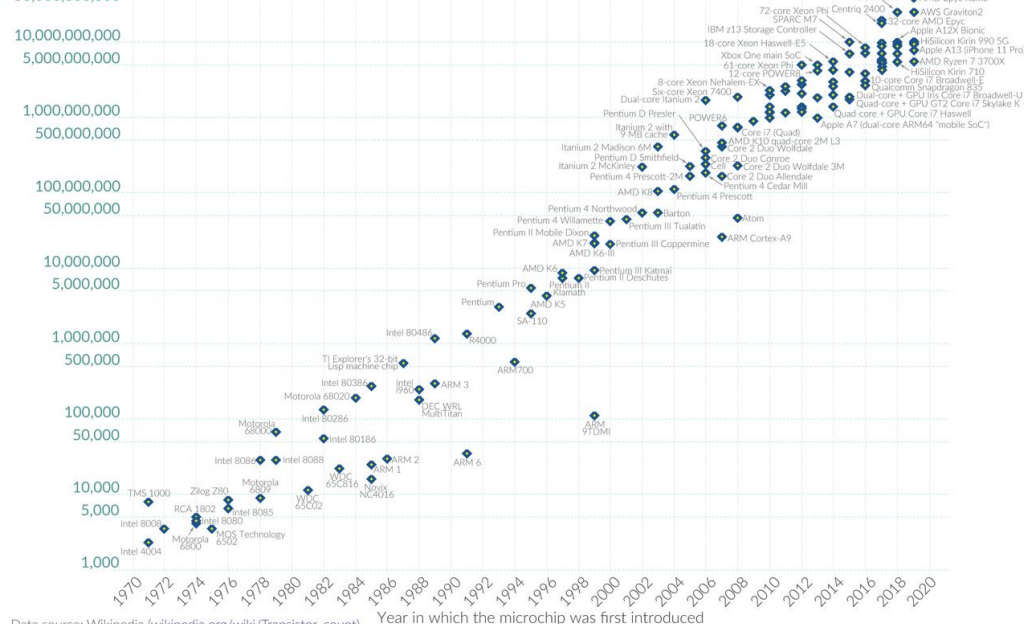
Moore's Law: The number of transistors on microchips doubles every two years

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important for other aspects of technological progress in computing – such as processing speed or the price of computers.

Our World
in Data

Transistor count

50,000,000,000



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)

OurWorldinData.org – Research and data to make progress against the world's largest problems.

Licensed under CC-BY by the authors Hannah Ritchie and Max Roser.

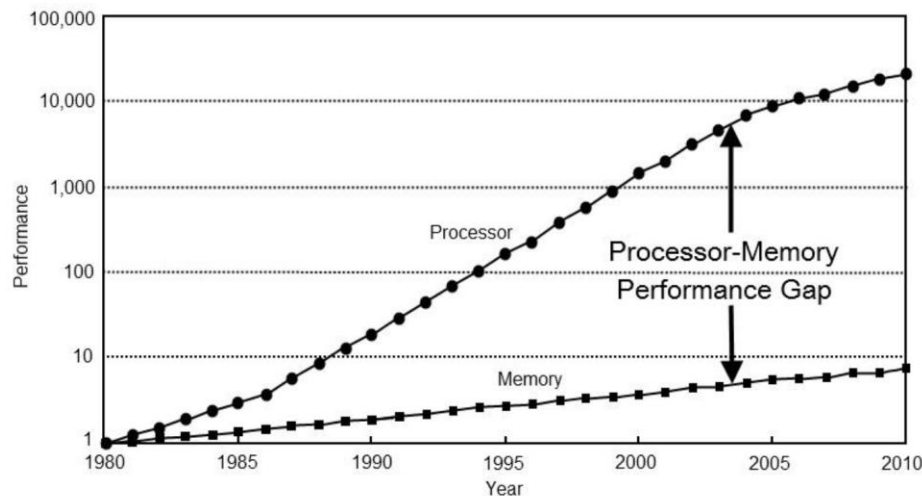
Breakdown of Dennard's Scaling (2006)

As transistors got too small

- Leakage current between transistor gate
- Threshold voltage couldn't be reduced further
 - Higher power density (watt/mm²)
- Heat dissipation from densely packed transistors
- Instability of transistors when approaching atomic scale

Memory Wall

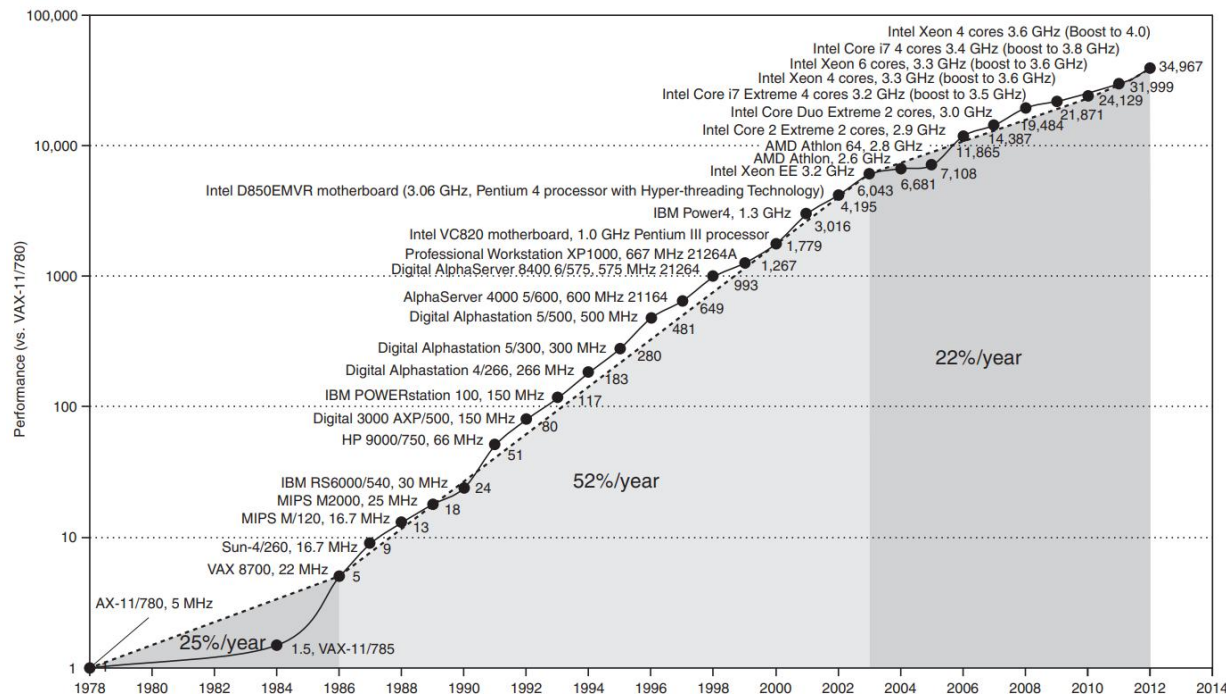
- DRAW technology scaling
 - Is slower than processor's
 - slower clock than CPU's
- Memory Latency
 - Getting data from memory
 - Takes more CPU cycles
- Multi-core area
 - Increases memory requests
 - Increases demand on memory bandwidth



Computer Architecture: A Quantitative Approach by John L. Hennessy, David A. Patterson, Andrea C. Arpaci-Dusseau

The big picture

- -1984
 - Process technology
- 1984-2004
 - Process technology
 - Microarchitecture optimizations
- 2004+
 - Power wall
 - Memory wall
 - ILP wall

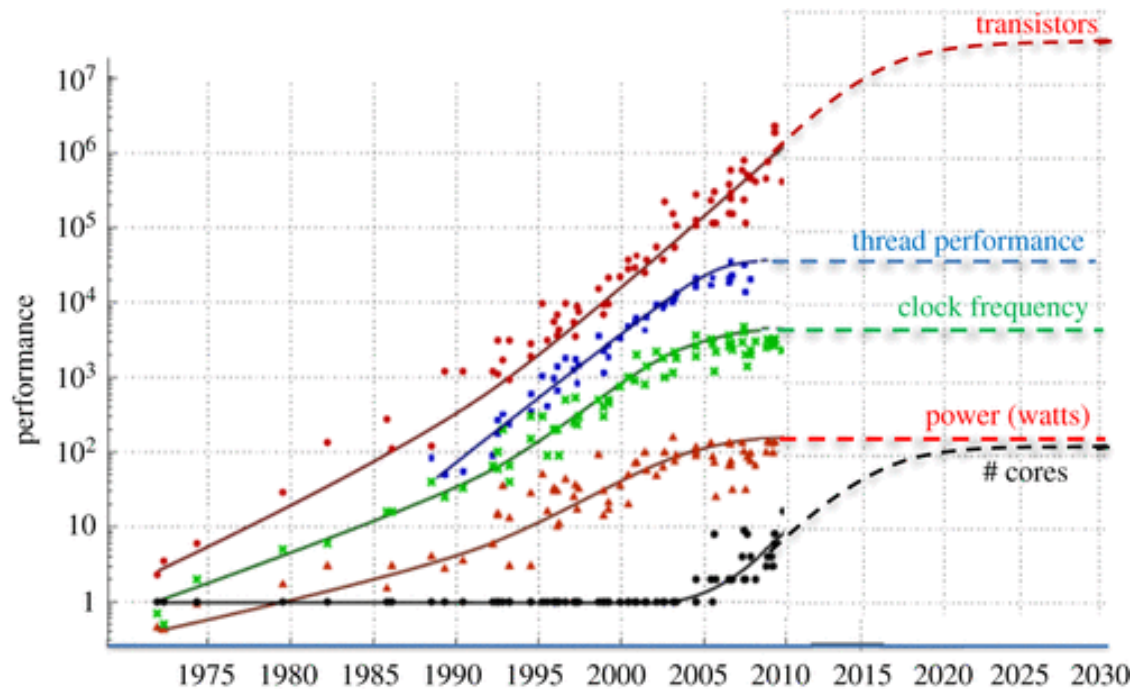


SPECint benchmarks – Patterson & Hennessy book

The end of Moore's Law

Multicore Area

- Increase the number of cores



Kunle Olukotun, Lance Hammond, Herb Sutter, Mark Horowitz
and extended by John Shalf.

Multicore area challenges

- Limitations application parallelism
- Inter-core communication
- Memory wall
- Heat dissipation
- Power management
- Physical limits

Great challenges for computer architects

- How do we continue to scale performance?
- It will require a holistic approach
- At all layers of the computing stack
 - Technology nodes
 - Microarchitecture
 - Compilers
 - Systems and applications
- The progress won't stop ;-)

What we will learn in this class

- The fundamentals of computer architecture
- Understand the design decisions when building a processor
- How to simulate and evaluate a processor

Q&A
