

CS 3410: Advanced Computer Architecture

Bruce Childers  
 Dietrich School of Arts and Sciences  
 Department of Computer Science  
<http://www.cs.pitt.edu/~childers>  
[childers@cs.pitt.edu](mailto:childers@cs.pitt.edu)

January 6, 2015




---

---

---

---

---

---

---

---

### Seminar Course

- Primary focus will be **memory systems**, broadly interpreted.
  - Any aspect is “fair game” for the course.
  - Not a lot of background. Most are senior and have background. If you don’t have the background, do extra reading, talk to others.
- This is **your course!**
  - What you want to do, is what we will do.
  - What you put into it, is what you’ll get out of it.
- Seminar != Lecture course
  - We read papers, discuss results, innovate, and experiment.
  - I’m also a participant, but providing some degree of direction.

---

---

---

---

---

---

---

---

### Requirements

- Reading and presenting papers, discussing, doing a project
- Letter grade vs. Satisfactory/Not-satisfactory
  - Participation (30%, 50%)
  - Presentations (30%, 50%)
  - Project (40%)
- Reading 2-4 papers per week; will depend on how quickly we discuss papers (e.g., 6 page DAC vs. 40 page TACO journal)
- $((\#weeks - 1) * papers/week) / (participants - 1) = \#presentations$
- Papers selected by you (this is **your course!**)
  - I retain line item veto power. ☺
  - We want top notch, important work for discussion.

---

---

---

---

---

---

---

---

**Requirements**

- Readings
  - We will all suggest & then decide papers to read.
  - Select first set next week, then refine later.
- You are expected to **read every assigned paper** each week.
- A review (short) will be required; 24 hours before lecture.
  - See web site for a review form (childers/CS3410)
- Presentations
  - You can use them, if available online.
  - Otherwise, you will need to make one. You can “cut” from the paper to avoid drawing figures.
  - Target: 30 minutes + 10-20 minutes discussion
  - Discussion is a critique of the work. Let’s aim for positive, but look for shortcomings that might led to exciting new ideas.

---

---

---

---

---

---

---

---

**Requirements**

- Projects
  - Oriented around some aspect of memory
  - I will suggest projects. You are not required to do my suggestions, but you will need to make a proposal for alternatives.
- Proposal
  - Develop the suggestion into a) challenges, b) approaches, c) plan
- Development (doing the project)
- Status report weekly
- Presentation end of semester
- If you propose it, it’s accepted, you do it, then A.

---

---

---

---

---

---

---

---

**Schedule**

- **Santiago’s simulator (on Thursday)**
- **On Thursday, identify 3-4 papers you find “interesting”.**
  - Only look at the abstract & conclusion to judge. Learn to do this!
  - Be prepared to say why you selected something
  - Bring your list! We’ll decide on Thursday!
- Places to look for good papers
  - ISCA, MICRO, HPCA, ACM TACO, PACT, DAC, DATE, NVMW, IDEM, IEEE TC, ... you’ll also find good papers in “odd” places, such as USENIX ATC, OSDI, EuroSys, VLDB, etc... **memory is everywhere!!!**
- For now, just look for “interesting”. We may winnow list down to a set of topics. We’ll see.
- Starting next Tuesday, we will present/read.
- We may not meet every lecture, particularly later in the semester (projects). You may meet when I’m traveling!

---

---

---

---

---

---

---

---

**Custom Design (Specialization)**  
Implications to Computer Systems  
Research

---

---

---

---

---

---

---

---

The Trends

---

---

---

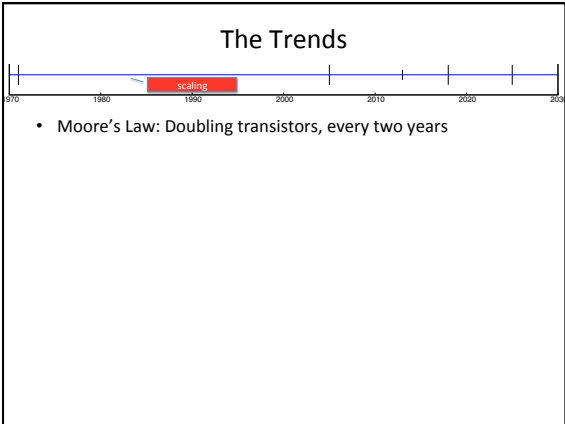
---

---

---

---

---



---

---

---

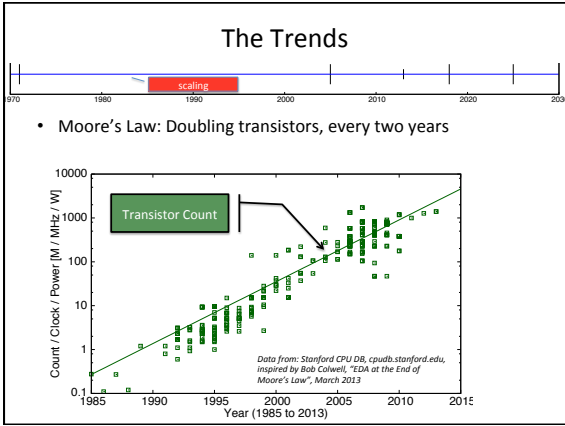
---

---

---

---

---




---

---

---

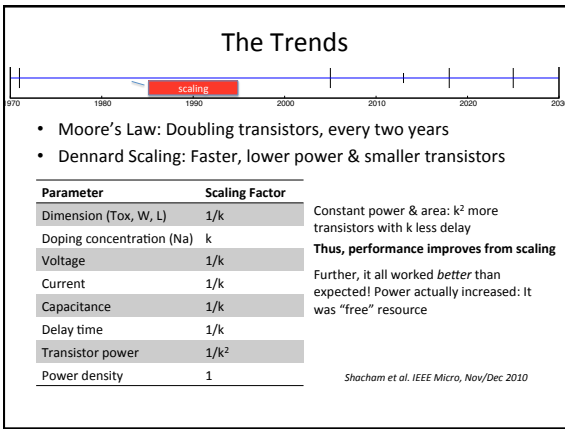
---

---

---

---

---




---

---

---

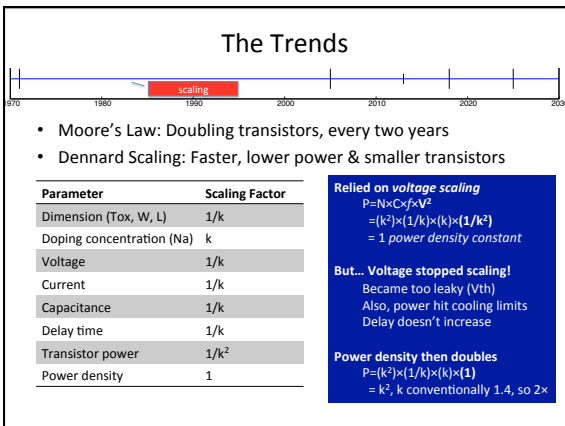
---

---

---

---

---




---

---

---

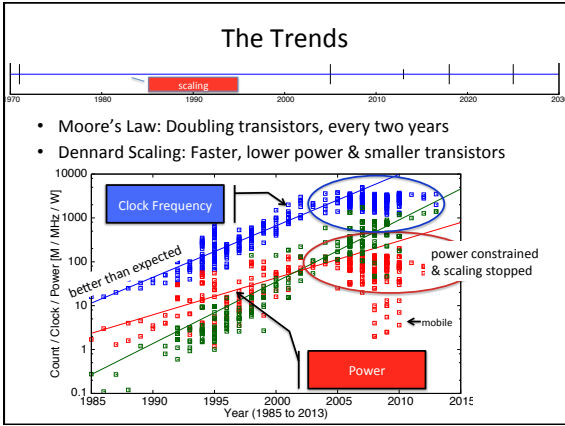
---

---

---

---

---




---

---

---

---

---

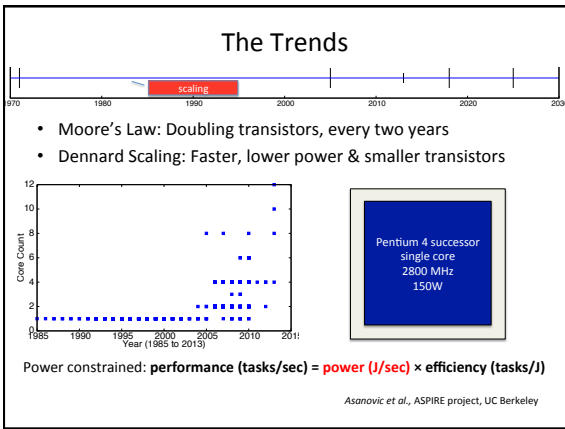
---

---

---

---

---




---

---

---

---

---

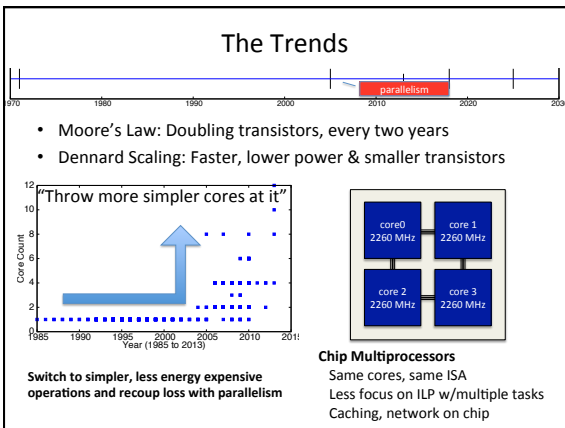
---

---

---

---

---




---

---

---

---

---

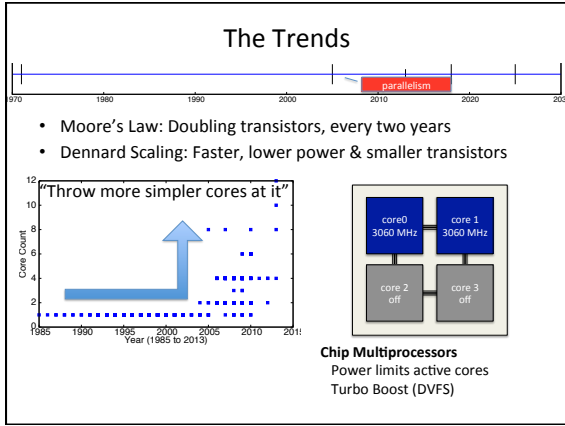
---

---

---

---

---



---

---

---

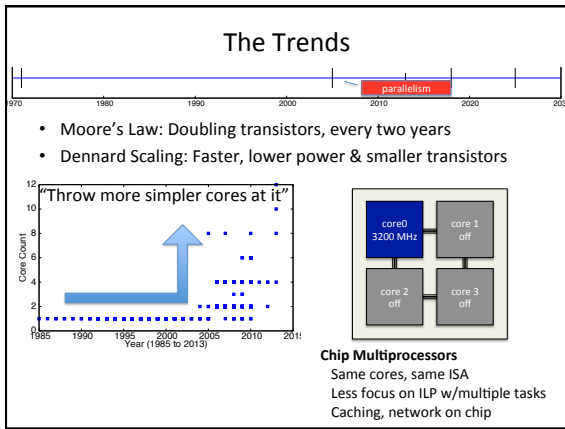
---

---

---

---

---



---

---

---

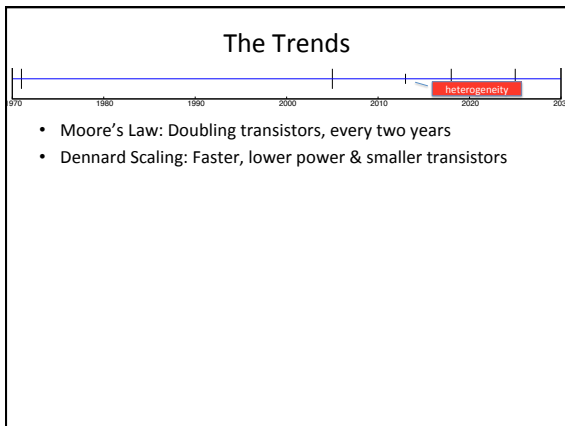
---

---

---

---

---



---

---

---

---

---

---

---

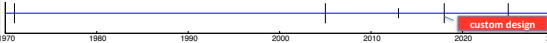
---








### The Trends



- Moore's Law: Doubling transistors, every two years
- Dennard Scaling: Faster, lower power & smaller transistors



Better design tools: lowering NRE costs  
 More economical: Reaches to more domains

- Differentiation: user experience, applications
- At both the Edge and the Cloud

Partially custom: Reduce expense, design time

- Automated, quick-turn composition
- A few truly custom blocks

Short time span: More design customization

- 12 to 18 month from design to deployed
- Aggressive EDA/design turns

---

---

---

---

---

---

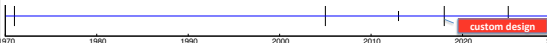
---

---

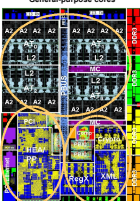
---

---

### The Trends



- Moore's Law: Doubling transistors, every two years
- Dennard Scaling: Faster, lower power & smaller transistors



**IBM PowerEN**  
 64 SMT app. cores  
 RegEx accelerator  
 XML accelerator  
 Crypto accelerator  
 Compression

*"deals with streaming data from multiple sources, often requiring repeated application of several standard algorithmic kernels. The **demand for high data rates and power efficiency points toward hardware acceleration of key functions**"*

Hardware Acceleration in the IBM PowerEN Processor: Architecture and Performance, Krishna et al., PACT 2012

---

---

---

---

---

---

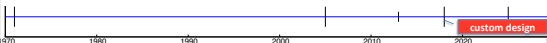
---

---

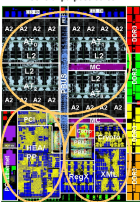
---

---

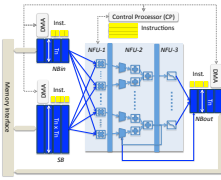
### The Trends



- Moore's Law: Doubling transistors, every two years
- Dennard Scaling: Faster, lower power & smaller transistors



**IBM PowerEN**  
 64 SMT app. cores  
 RegEx accelerator  
 XML accelerator  
 Crypto accelerator  
 Compression



**DianNao**: A Small-Footprint High-throughput Accelerator for Ubiquitous Machine Learning

---

---

---

---

---

---

---

---

---

---

### And Many More...

- Parallelism
- Heterogeneity
- Customization
- Vertical stacking (3D)
- Memory** + compute (Near data computation)
- Alternative **memory** technologies
- Etc.

---

---

---

---

---

---

---

---

### And Many More...

- Parallelism
- Heterogeneity
- Customization
- Vertical stacking (3D)
- Memory** + compute (Near data computation)
- Alternative **memory** technologies
- Etc.

#### A Selected Few General Implications

- Burden & metrics shift
- Dynamic, unanticipated landscape
- Agility, Flexibility and Legacy
- (Hello, EDA. Meet, CSR.)

---

---

---

---

---

---

---

---

### Implications: Burden & Metrics Shift

- Software delivers power, performance, reliability
  - Parallel, heterogeneous, custom: SW must *exploit* it to be beneficial.
  - Careful tuning: More customization, the more important.
- User\* oriented utility (“Apple Effect”)
  - **Power**: Judge decisions as benefit versus cost
  - **Quality of service**: Just enough at lowest cost to achieve experience
  - **Reliability**: Thermal, power, operating conditions
- New functionality: MEMS first-class differentiator
  - Accelerometers
  - Digital light projector
  - Pressure sensor
  - Bio/chem sensors, etc.



\* “User” may be “other computer”, “device”

---

---

---

---

---

---

---

---

**Implications: Dynamism Everywhere**

- Uncertainty in execution: Intentional & Unintentional
- Sea of resources (Intentional)
  - Easier, quicker design in more domains (beyond big volume)
  - More specialization in each domain (e.g., Amazon vs. Facebook)
  - Shorter timelines, more design spins
- Runtime variability (Unintentional)
  - Obey thermal design point: Throttling, activation/de-activation
  - Selection of current active resources (overheads, best choices)
  - Reliability: Thermal induced failure, recovery
- Opaque programming
  - Hardware design may not be transparent (and probably shouldn't be)
  - Portability across different designs
  - Shared accelerators (not the norm today!)
- At many points: Processor, memory, storage, & network

---

---

---

---

---

---

---

---

**Implications: Agility, Flexibility, Legacy**

- More than "just an application"
  - Both program and specification of design
  - Retain agnostic viewpoint – not everything will be custom
  - Communication
- System software must ride along too
  - System software requires customization
  - Equally difficult challenge for software
  - Ease correct integration and composition of new resource management, partitioning and mapping algorithms
  - Hierarchical and coordinated responsibilities
- Legacy foundation
  - Facilitate transition: Can't throw away the investment
  - Use accelerators: Give up some opportunity, utilization

---

---

---

---

---

---

---

---

**Attributes of Future Systems (Possible?)**

- Intention of task & automated generation (program+design)
- Communication (e.g., near data computation)
- Dynamic continuous binding
- Hierarchical resource management and optimization
- Deep monitor, control, coordination
- Composed, extendable system software

---

---

---

---

---

---

---

---

### Summary: Research Challenges

- Abstract, express, expose customization opportunities
  - Productivity, ease, managing complexity, & design cost
  - Agnostic SW with custom, multiple hardware instantiation
  - Methodologies, DSLs, Program/Chip generators/optimizers, APIs
- Programming and runtime paradigms
  - Highly tuned software to custom hardware
  - Just-in-time mapping (JIM): dynamic, continuous, context
  - Resource management: Cooperative, across layers & across time
  - Communication
- Custom system software (embodying paradigms)
  - Extensible & generated to new hardware customizations
  - Optimize & strip away abstractions when instantiated
  - System software in design flow: Modeling, verify
- (Legacy software is reality. Utilize and support.)

---

---

---

---

---

---

---

---

### CS 3410 Advanced Computer Architecture

- Topic: **Memory Sub-system Design!** A Renaissance period!
- Shifting away from processor to memory (power/performance)
  - Experimental methodology – sound science; with OCCAM

---

---

---

---

---

---

---

---

### CS 3410 Advanced Computer Architecture

- Topic: **Memory Sub-system Design!** A Renaissance period!
- Shifting away from processor to memory (power/performance)
  - Experimental methodology – sound science; with OCCAM
  - Increasing capacity, bandwidth (the rat race)
  - Main memory energy/power consumption
  - DRAM scaling is coming to an end. RIP.

---

---

---

---

---

---


---

---

**CS 3410 Advanced Computer Architecture**

Topic: **Memory Sub-system Design!** A Renaissance period!

- Shifting away from processor to memory (power/performance)
- Experimental methodology – sound science; with OCCAM
- Increasing capacity, bandwidth (the rat race)
  - Multi-core: Core counts are ever increasing
  - Changing applications: Data intensive (“big data”, analytics)
  - Consolidation: GPGPU, Cloud/Virtualization, MPSoC
- Main memory energy/power consumption
- DRAM scaling is coming to an end. RIP.




---

---

---

---

---

---

---

---

**CS 3410 Advanced Computer Architecture**

Topic: **Memory Sub-system Design!** A Renaissance period!

- Shifting away from processor to memory (power/performance)
- Experimental methodology – sound science; with OCCAM
- Increasing capacity, bandwidth (the rat race)
- Main memory energy/power consumption
  - Processors are relatively energy efficient – long term focus
  - Capacity, refresh increasing, leading to more energy consumption
  - Idleness may be wasteful – sparse data in applications
  - Upwards of 50% of total server power
- DRAM scaling is coming to an end. RIP.

---

---

---

---

---

---

---

---

**CS 3410 Advanced Computer Architecture**

Topic: **Memory Sub-system Design!** A Renaissance period!

- Shifting away from processor to memory (power/performance)
- Experimental methodology – sound science; with OCCAM
- Increasing capacity, bandwidth (the rat race)
- Main memory energy/power consumption
- DRAM scaling is coming to an end. RIP.
  - “No reliable way known to scale below 22nm”. ITRS.
  - Scaling has driven memory capacity increases but it’s ending?
  - Cap needs to be large enough to sense correctly/reliably
  - Big enough access transistor for low leakage/high retention

---

---

---

---

---

---

---

---

**CS 3410 Advanced Computer Architecture**

Topic: **Memory Sub-system Design!** A Renaissance period!

- Shifting away from processor to memory (power/performance)
- Experimental methodology – sound science; with OCCAM

Moving away from DRAM for many reasons

- Capacity limits of scaling, number of DIMMs, etc.
- Consumes significant power due to leakage and refresh
- End of reliable operation?

Clearly, we'd prefer NOT to move away from DRAM.  
But can we fix DRAM, or find alternatives to transition?

---

---

---

---

---

---

---

---

**CS 3410 Advanced Computer Architecture**

- Emerging technologies
  - New bit cell designs (rather, old is the new new): PCM, STT
  - Near data computation (in-memory computation)
  - Tiered, network-attached memory
  - Storage class memory (persistence at highest level of hierarchy)
  - Decoupling main memory from processor (e.g., HMC)
  - GPGPU high bandwidth (e.g., high bandwidth memory, HBM)
- Let's look at an example.

---

---

---

---

---

---

---

---

**The End**

Email: [childers@cs.pitt.edu](mailto:childers@cs.pitt.edu)  
Web: <http://www.cs.pitt.edu/~childers>

---

---

---

---

---

---

---

---