CSC 2224:
Parallel Computer Architecture and Programming

Prof. Gennady Pekhimenko
University of Toronto
Fall 2022

The content of this lecture is adapted from the lectures of Onur Mutlu, Hadi Esmaeilzadeh, and Samira Khan

Summary

- Syllabus
  - Course Introduction, Logistics, Grading
- Paper Reviews and Presentation
  - How to handle these beasts
- Project
  - Team formation, Proposal, Milestones
- And Some Technical Details

Syllabus: Who Are We?

Gennady (Gena) Pekhimenko

Assistant Professor, Instructor

pekhimenko@cs.toronto.edu

http://www.cs.toronto.edu/~pekhimenko/

Office: BA 5232

PhD from Carnegie Mellon

Worked at Amazon, Microsoft Research, Nvidia, IBM

Research interests: computer architecture, systems, machine learning, compilers, hardware acceleration

Computer Systems and Networking Group (CSNG)

EcoSystem Group

VECTOR INSTITUTE

Anand Jayarajan

PhD Student, TA

anandj@cs.toronto.edu

MSc. from UBC

Research interests: stream processing, systems for machine learning

Computer Systems and Networking Group (CSNG)
EcoSystem Group

Course Information: Where to Get?

- Course Website:

  https://uoft-ecosystem.github.io/CSC2224-Website/

  Announcements, Syllabus, Course Info, Lecture Notes, Reading List and Reviews, Project Information

- Piazza: https://piazza.com/class/l7tbs42idbm76f/
  __Questions/Discussions, Syllabus, Searching for Teammates

- Your email

What is Computer Architecture?

- **Computer Architecture:**

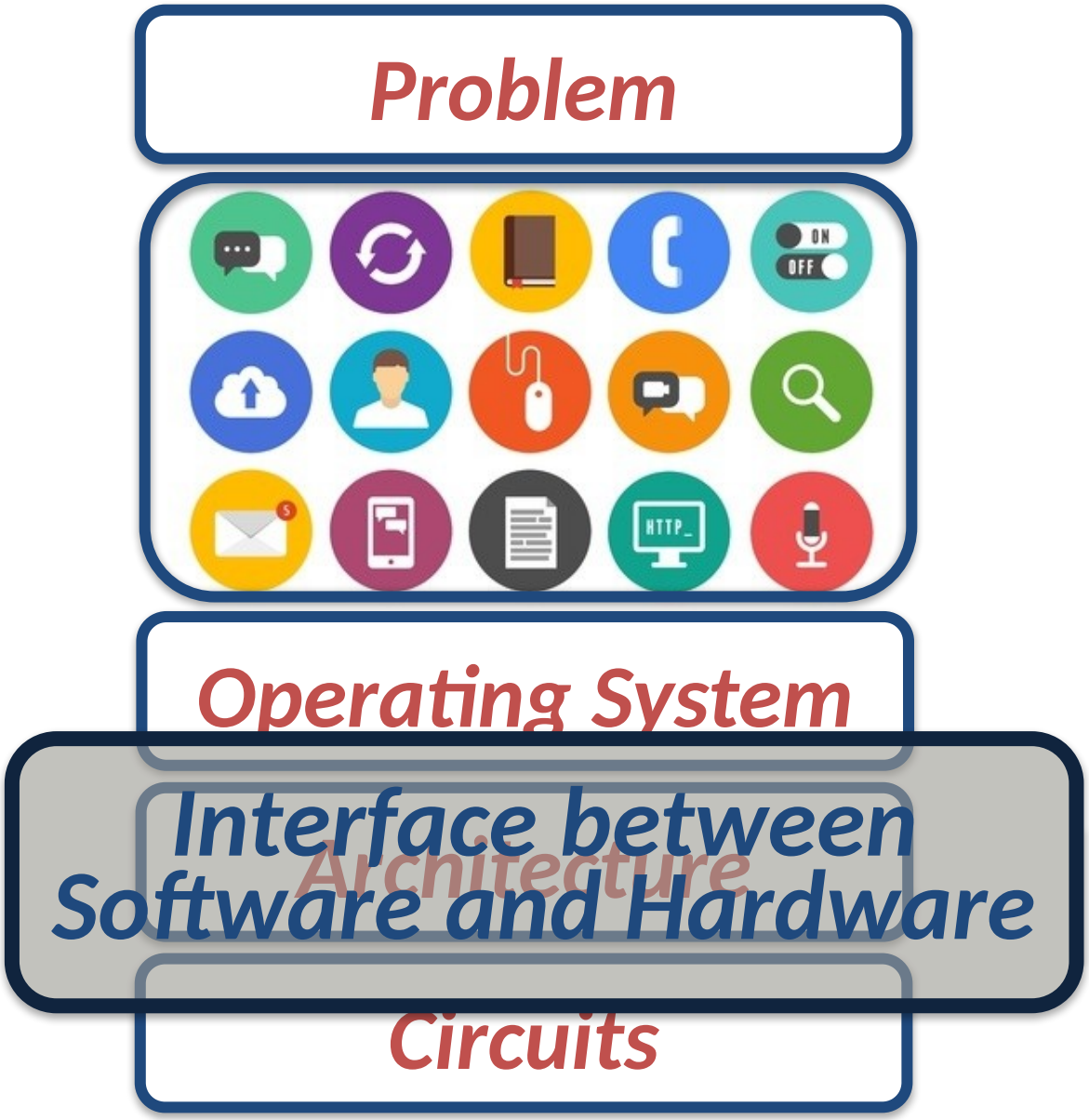
The science and art of designing, selecting, and interconnecting hardware components and designing the hardware/software interface to create a computing system that meets functional, performance, energy consumption, cost, and other specific goals.

- **Traditional definition:**

The term architecture is used here to describe the attributes of a system as seen by the programmer, i.e. and functional behavior as distinct from dataflow and controls, the logic design, implementation." Gene Amdahl, IBM Jo

Dr. Amdahl holding a 100gate LSI air-cooled chip. On his desk is a circuit board with the chips on it. This circuit board was for an Amdahl 470 V/6 (photograph dated March 1973).

Level of Transformations



The Power of Abstraction

- ***Isolation***
  - A higher level only needs to know about the interface to the lower level, not how the lower level is implemented
  - For example, high-level language programmer does not really need to know about the architecture

- ***Productivity***
  - No need to worry about decisions made in underlying levels
  - For example, programming in Java vs. C vs. assembly vs. binary vs. by specifying control signals of each transistor every cycle

Crossing the Abstraction Layers

- Should we ***always focus*** on our own layer?
- As long as ***everything goes well***, not knowing what happens in the underlying level (or above) is not a problem
- ***What if***
  - One of the layers reach a limit, there is no way to improve
  - There is a new disruptive change in technology that cannot be contained in a layer
  - New Applications that are too slow for today's system

Scope of this Course

- **Broad view of processor and memory design**
  - Beyond the ISA+microarchitecture levels
  - E.g., system-architecture interfaces and interactions
  - E.g., application-architecture interfaces and interactions
- **Out-of-the-box thinking is greatly encouraged**
  - E.g., research projects and readings on architectures that challenge the current dominant paradigms
    - processing in memory, approximate systems, persistent memory, neuromorphic computing, …
  - E.g., readings on topics that are traditionally covered less in computer architecture courses

What Will You Learn?

- Hardware/software interface, major components, and programming models of a modern computing platform
  - State-of-the-art as well as research proposals (lots of them)
  - Tradeoffs and how to make them
  - Emphasis on cutting-edge (research & state-of-the-art)
- Hands-on research in a computer architecture topic
  - Semester-long research project
  - Focus: How to design better architectures (not an intro course)
- Broaden your research vision
  - Will read papers not only from computer architecture
  - But, possibly, from circuits, programming language, operating systems, graphics, networking, …

Course Goals

- ***Goal 1:*** Rethink and redesign our computing model focusing on ***minimizing data movement and storage***
  - Understand the importance of cross layer research
  - Gain new insight from VLSI circuits, architecture design, systems, programming languages, and new application domains
  - Strong emphasis on ***critical analysis of research papers*** (through reading and literature review assignments)
- ***Goal 2:*** To provide the necessary **background and experience** to advance the state-of-the-art in computer architecture by **performing cutting-edge research**
  - Strong emphasis on ***developing new mechanisms that advance the state-of-the-art*** (through the course research project)

Advanced Graduate-Level Class

- Required background:
  - basic architecture
  - basic compilers
  - basic OS
  - programming skills
  - ***spirit, excitement, and dedication*** for deep exploration of a topic in computer architecture

CSC 2224: Parallel Computer Architecture and Programming Grading, Policies, Course Work

Prof. Gennady Pekhimenko
University of Toronto
Fall 2022

What Do I Expect from You?

- **Work hard**
- **Ask questions, think critically, participate in discussion**
- Critically review the assigned research papers & readings
  - Discuss/critique them online with peers and us
- Use Piazza and Write good reviews
- Start the research project early and focus
- Remember "Chance favors the prepared mind." (Pasteur)

Course Work

- **Project (50%):**
  - Groups of 2, sometimes 3
  - Proposal, Progress Report, Final Presentation and Final Report
  - Scaled down version of the real research projects
- **Paper Reviews (20%):**
  - 1-2 papers per week
  - 3-4 paragraphs

Course Work (2)

- **Paper Presentation (20%):**
  - Related works in your project area

- **Class Participation (10%)**
  - Very important

Grading

- Grading will be back-end heavy
- Most of your grade will be determined late
  - How you prepare and manage your time is important
  - But grades should not be the reason for taking this course

Paper Review: How to Handle

1. Brief summary
   - What is the problem the paper is trying to solve?
   - What are the key ideas of the paper? Key insights?
   - What is the key contribution to literature at the time it was written?
   - What are the most important things you take out from it?

2. Strengths (most important ones)

   Does the paper solve the problem well?

Paper Reviews (2)

3. Weaknesses (most important ones)

   This is where you should **think critically**. Every paper/idea has a weakness. This does not mean the paper is necessarily bad. It means there is room for improvement and future research can accomplish this.

4. Can you do (much) better? Present your thoughts/ideas.
5. What have you learned/enjoyed/disliked in the paper? Why?

Review should be short and concise (~ 3/4 a page to 1 page)

Advice on Reviewing

- When doing the reviews, be very critical
- Always think about better ways of solving the problem or related problems
- Do background reading
  - Reviewing a paper/talk is the best way of learning about a research problem/topic
- Think about forming a literature survey topic or a research proposal based on the paper (for future studies)

How to Submit?

- Email **PDF** with your review to csc2224arch@gmail.com

- Due Thursday 1pm each week

- Next week: "**Dark Silicon and the End of Multicore Scaling**", Hadi Esmaeilzadeh, ISCA 2011.

Research Project

- Your chance to explore in depth a computer architecture topic that interests you
- Perhaps even publish your innovation in a top computer architecture/systems/ML conference
- **Start thinking about your project topic from now!**
- Interact with me and Anand
- Use Piazza to discuss ideas, form teams based on interests, research areas

Research Project (2)

- Goal: Develop (new) insight
  - Solve a problem in a new way or evaluate/analyze systems/ideas
  - Type 1:
    - Develop new ideas to solve an important problem
    - Rigorously evaluate the benefits and limitations of the ideas
  - Type 2:
    - Derive insight from rigorous analysis and understanding of existing systems or previously proposed ideas
    - Propose potential new solutions based on the new insight
- The problem and ideas need to be concrete
- Problem and goals need to be very clear

Research Proposal: Key Questions

- What is the problem?
- Why is it hard?
- How is it solved today?
- What is the new technical idea?
- Why can we succeed now?
- What is the impact if successful?
- http://en.wikipedia.org/wiki/George_H._Heilmeier

Where to Get Ideas?

- Read a lot of papers; find focused problem areas to survey papers on

- We will provide some ideas for projects

- A good way of finding topics to survey or do projects on is:
  - Examining the provided project ideas and papers
  - Reading assigned papers in lectures and related/followup work
  - Examining papers from recent conferences (ISCA, MICRO, HPCA, ASPLOS, OSDI/SOSP, PLDI, MLSys…)

Project: Major Milestones

- Project Proposal (1–2 pages) - Sept. 29th
- Progress Report (2–3 pages) - Nov. 3rd
- Poster Presentation - Dec. 8th (tentative)
- Project Report (6–8 pages) - Dec. 15th (tentative)

Presentation

- After the project topic is selected, identify the key related works (1-3 papers)

- Signup for the day to present your related work (25-30 minutes)

- The exact schedule of the talks will be posted after project groups are formed

Presentation: What to Cover

- **Summary of the Work**
  - Problem, key ideas or insights, detailed mechanisms, and results
- **Strengths and Weaknesses**
  - Detailed discussion on both sides
- **Discussion**
  - Future research directions
  - Alternative ways of solving the problem
  - Importance of the problem
  - Anything interesting about the research direction
  - What did you like? What did you not like?

CSC 2224: Parallel Computer Architecture and Programming
Why Computer Architecture Matters

Prof. Gennady Pekhimenko
University of Toronto
Fall 2022

What has made computing pervasive?

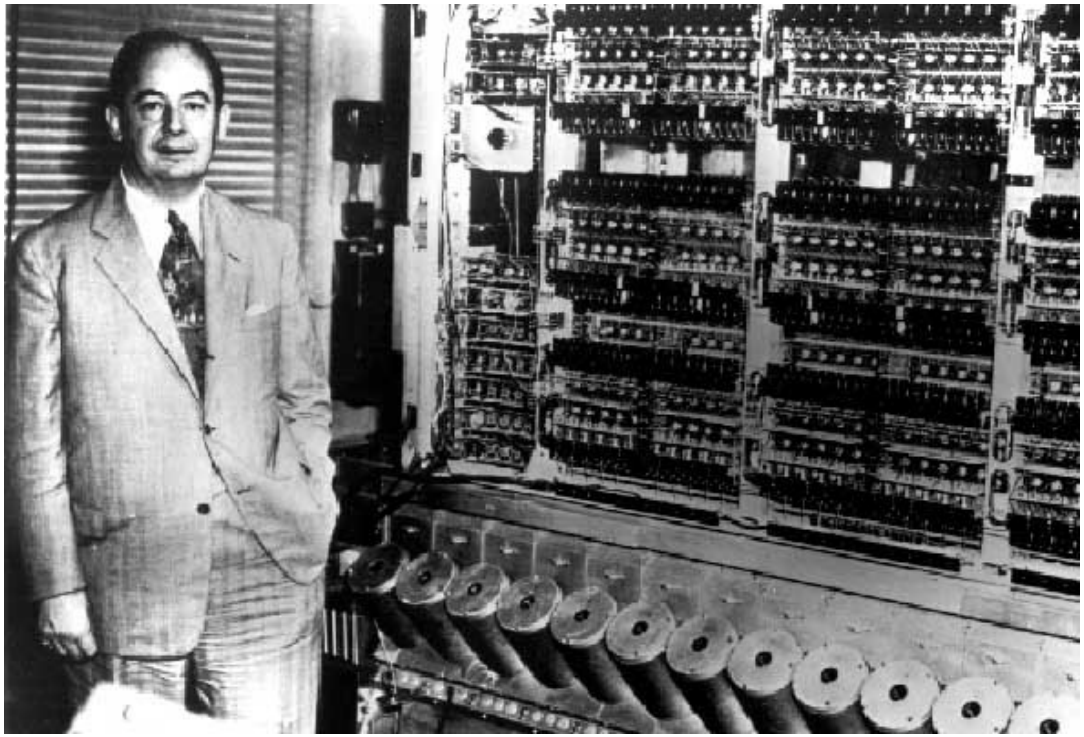
What is the backbone of computing industry?

Programmability

Networking

```
public class TcpClientSample
{
    public static void Main()
    {
        byte[] data = new byte[1024]; string input, stringData;
        TcpClient server;
        try{
            server = new TcpClient(" . . . . ", port);
        }catch (SocketException){
            Console.WriteLine("Unable to connect to server");
            return;
        }
        NetworkStream ns = server.GetStream();
        int recv = ns.Read(data, 0, data.Length);
        stringData  = Encoding.
          ASCII.GetString(data, 0, recv);
        Console.WriteLine(stringData);
        while(true){
            input = Console.ReadLine();
            if (input == "exit") break;
                newchild.Properties["ou"].Add
                ("Auditing Department");
                newchild.CommitChanges();
                newchild.Close();
```

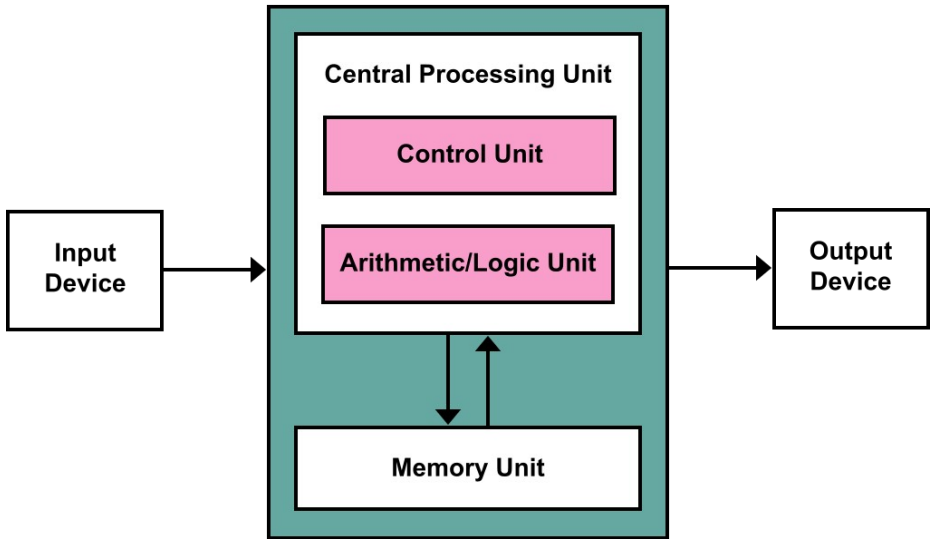
What makes computers programmable?



von Neumann architecture

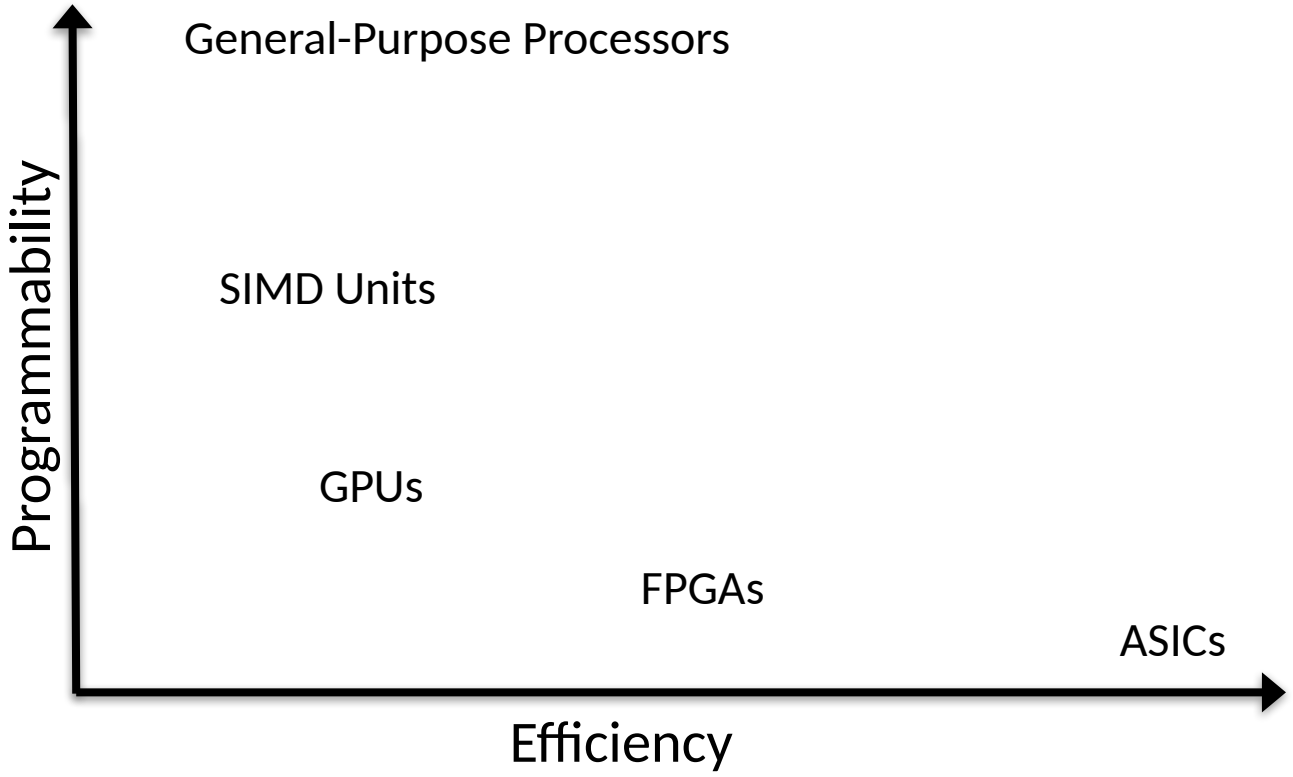
General-purpose processors

- Components
  - Memory (RAM)
  - Central processing unit (CPU)
    - Control unit
    - Arithmetic logic unit (ALU)
  - Input/output system
- Memory stores program and data
- Program instructions execute sequentially



Programmability versus Efficiency

Programmability versus Efficiency



WHAT IS THE DIFFERENCE BETWEEN THE COMPUTING INDUSTRY AND THE PAPER TOWEL INDUSTRY?

Industry of replacement

1971 2022

CAN WE CONTINUE BEING AN INDUSTRY OF NEW POSSIBILITIES?

Personalized healthcare

Virtual reality

Real-time translators

Moore's Law

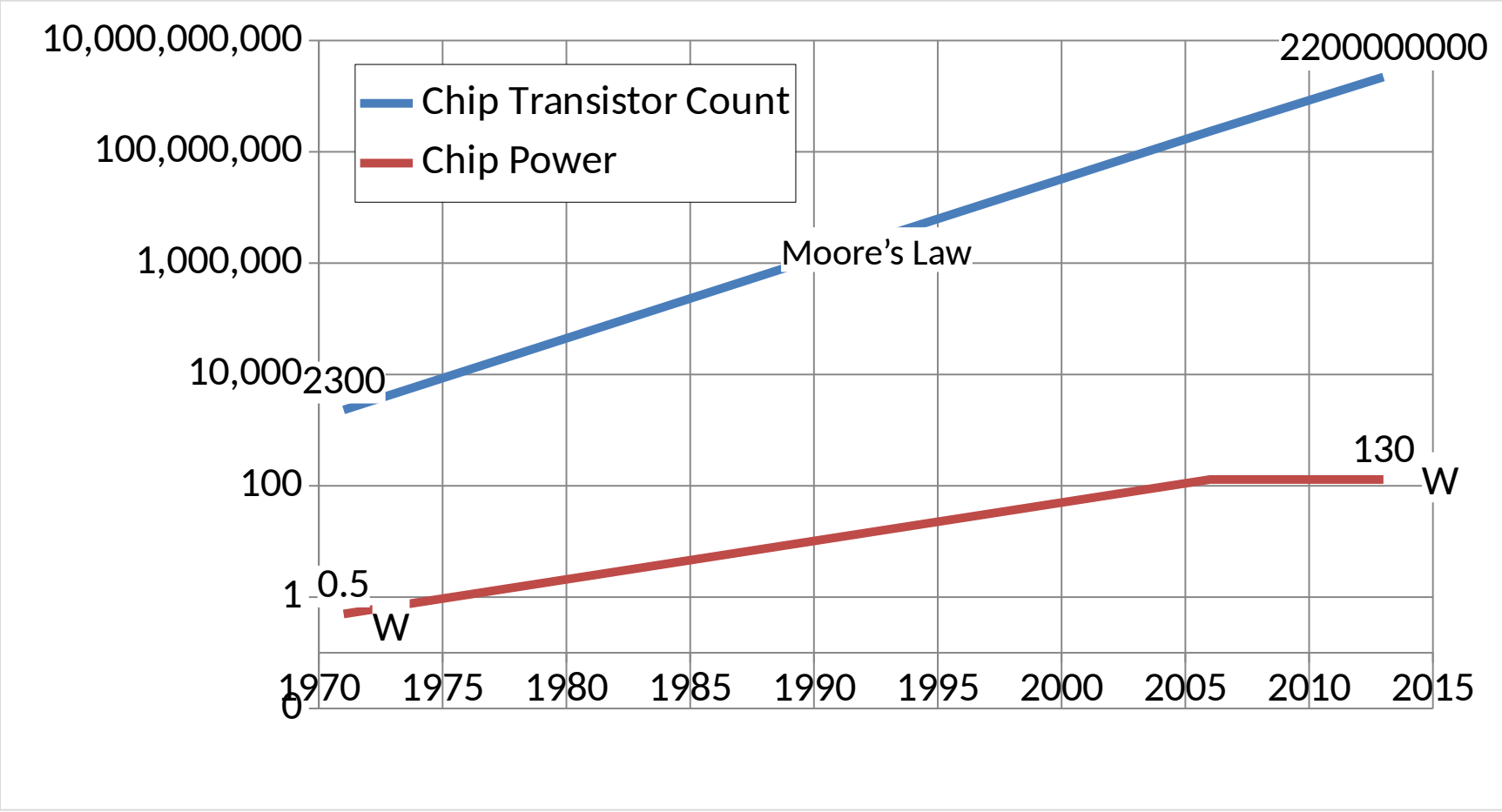
Or, how we became an industry of new possibilities

Every 2 Years

- Double the number of transistors
- Build higher performance general-purpose processors
  - Make the transistors available to masses
  - Increase performance (1.8×↑)
  - Lower the cost of computing (1.8×↓)

What is the catch?

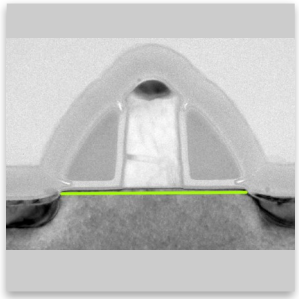
Powering the transistors without melting the chip



Dennard scaling:

Doubling the transistors; scale their power down

Transistor: 2D Voltage-Controlled Switch

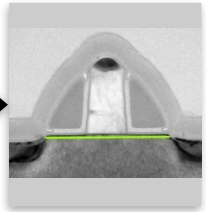


Dimensions

Voltage

Doping Concentrations

×0.7



Area — 0.5× ↓

Capacitance — 0.7× ↓

Frequency — 1.4× ↑

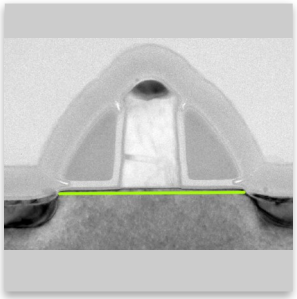
Power = Capacitance × Frequency × Voltage^2

Power — 0.5× ↓

Dennard scaling broke:

Double the transistors; still scale their power down

Transistor: 2D Voltage-Controlled Switch

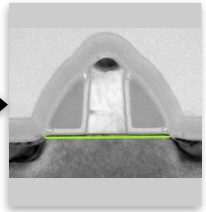


Dimensions

Voltage

Doping Concentrations

×0.7



Area 0.5× ↓

Capacitance 0.7× ↓

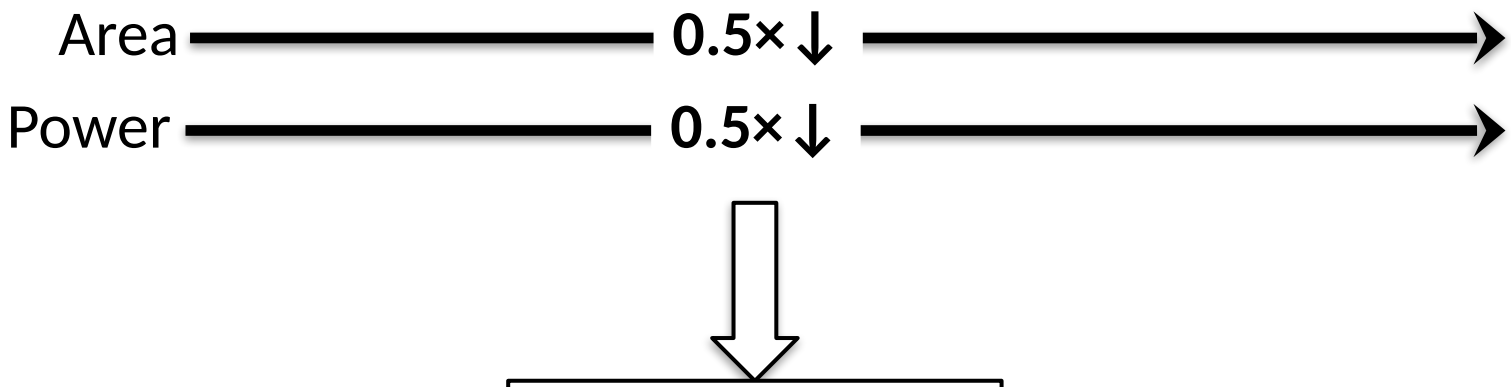
Frequency 1.4× ↑

Power = Capacitance × Frequency × Voltage^2

Power 0.5× ↓

Dark silicon

If you cannot power them, why bother making them?

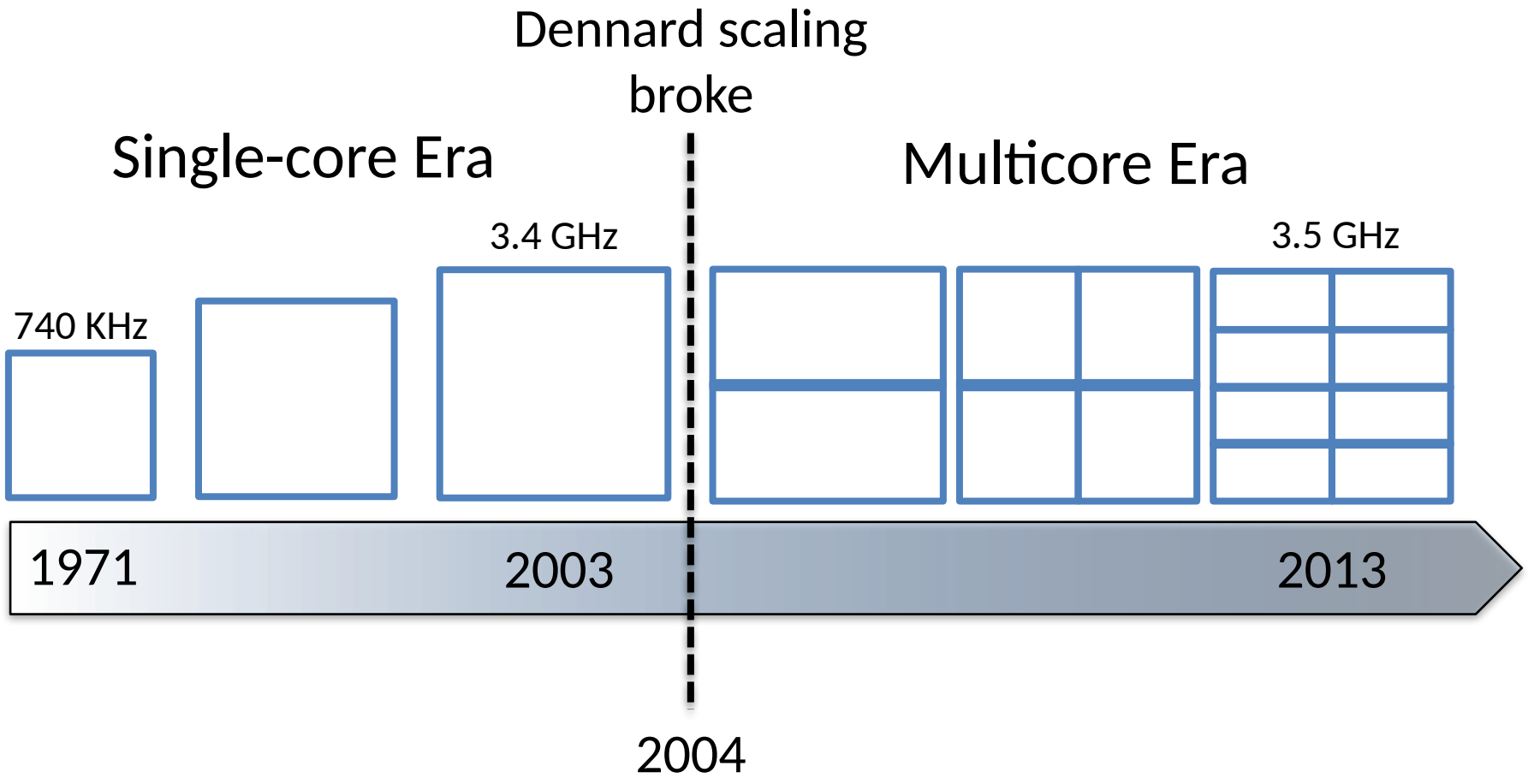


Dark Silicon

Fraction of transistors that need to be powered off at all times due to power constraints

Looking back

Evolution of processors

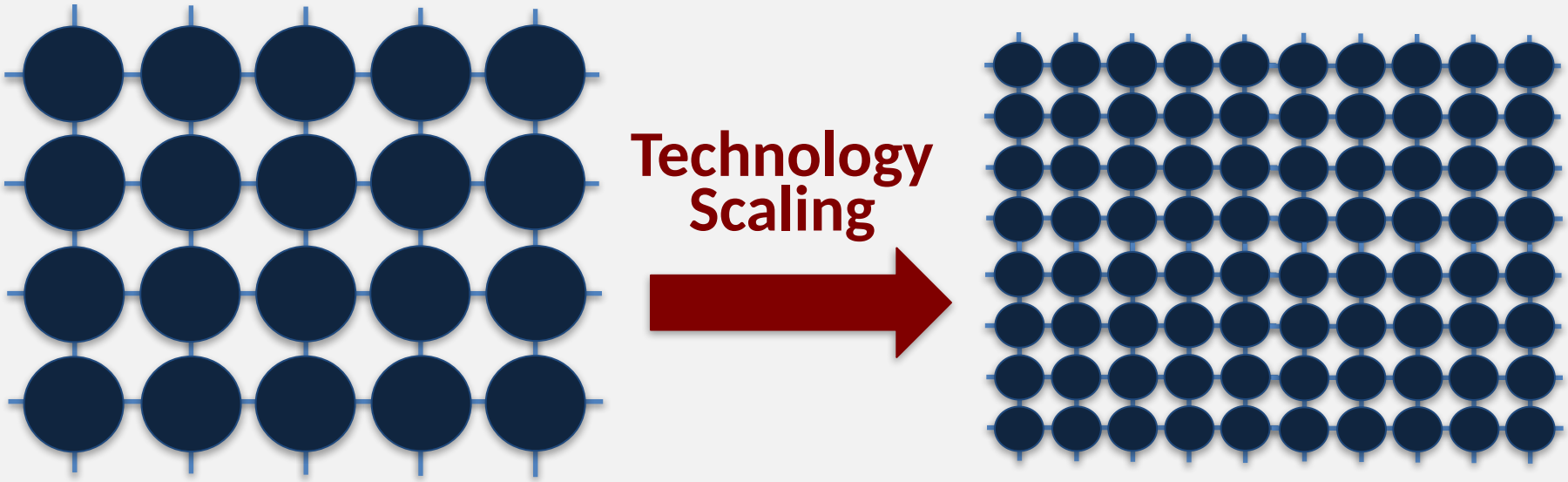


How about Memory/DRAM?

- Does it also have scaling issues?

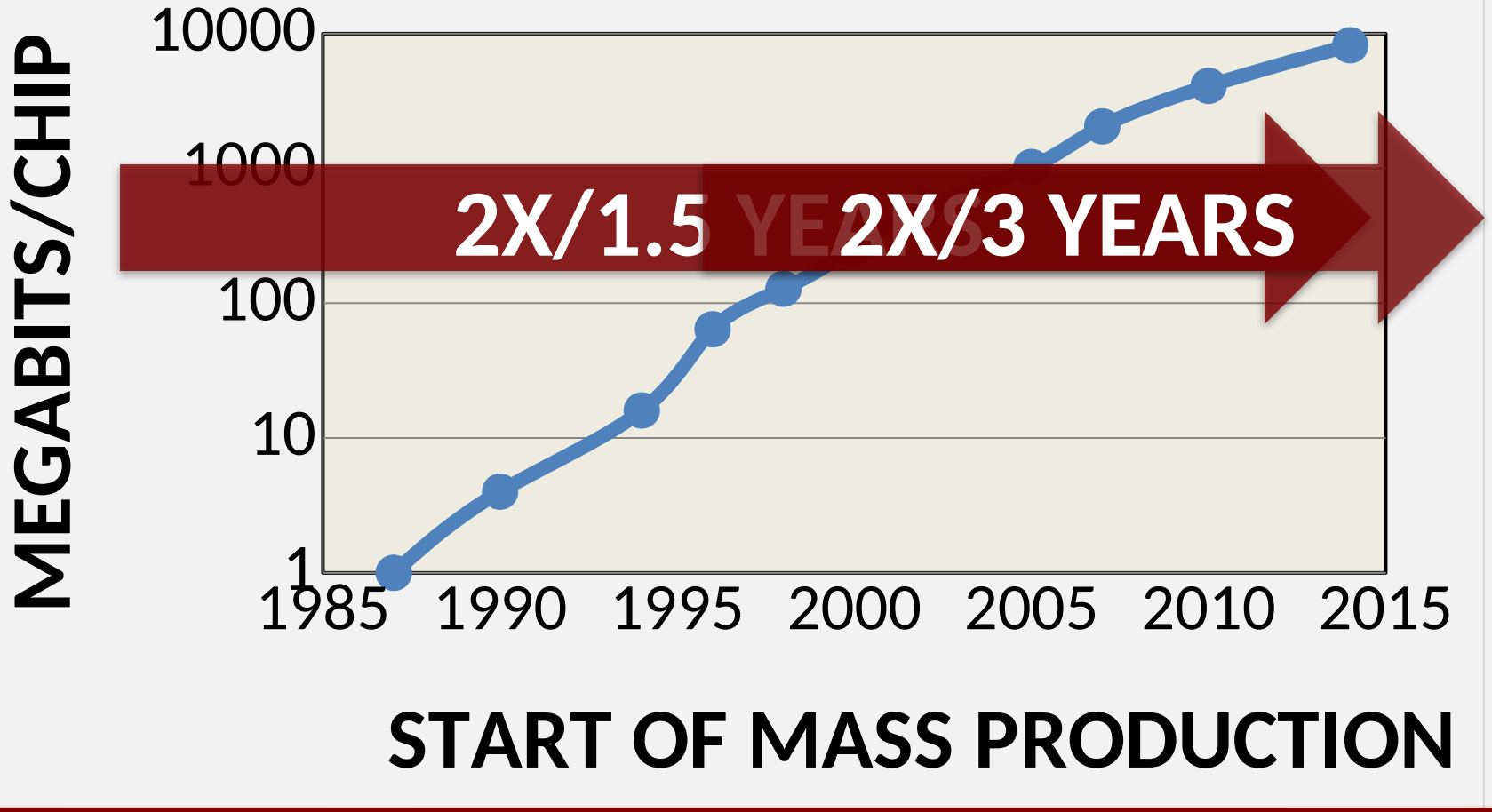
- What was DRAM trend over the last 30-40 years?

DRAM Scaling Challenge



DRAM scaling enabled high capacity

DRAM Scaling Trend



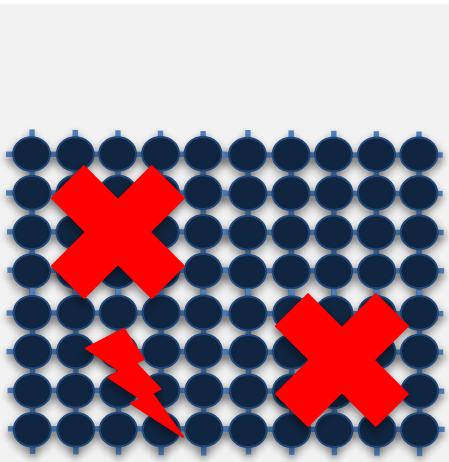
DRAM scaling is getting difficult

Source: Flash Memory Summit 2013, Memcon 2014

DRAM Scaling Challenge

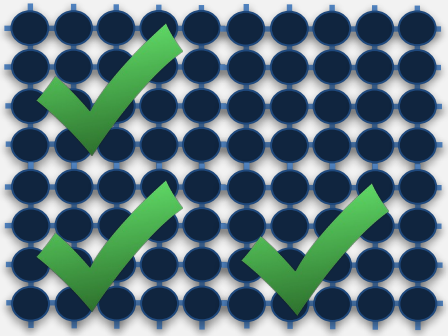
Manufacturing reliable cells at low cost is getting difficult

Traditional Approach

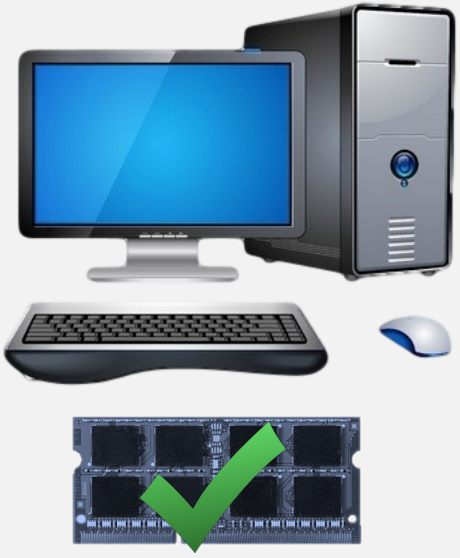


Unreliable DRAM Cells

Make DRAM Reliable



Reliable DRAM Cells



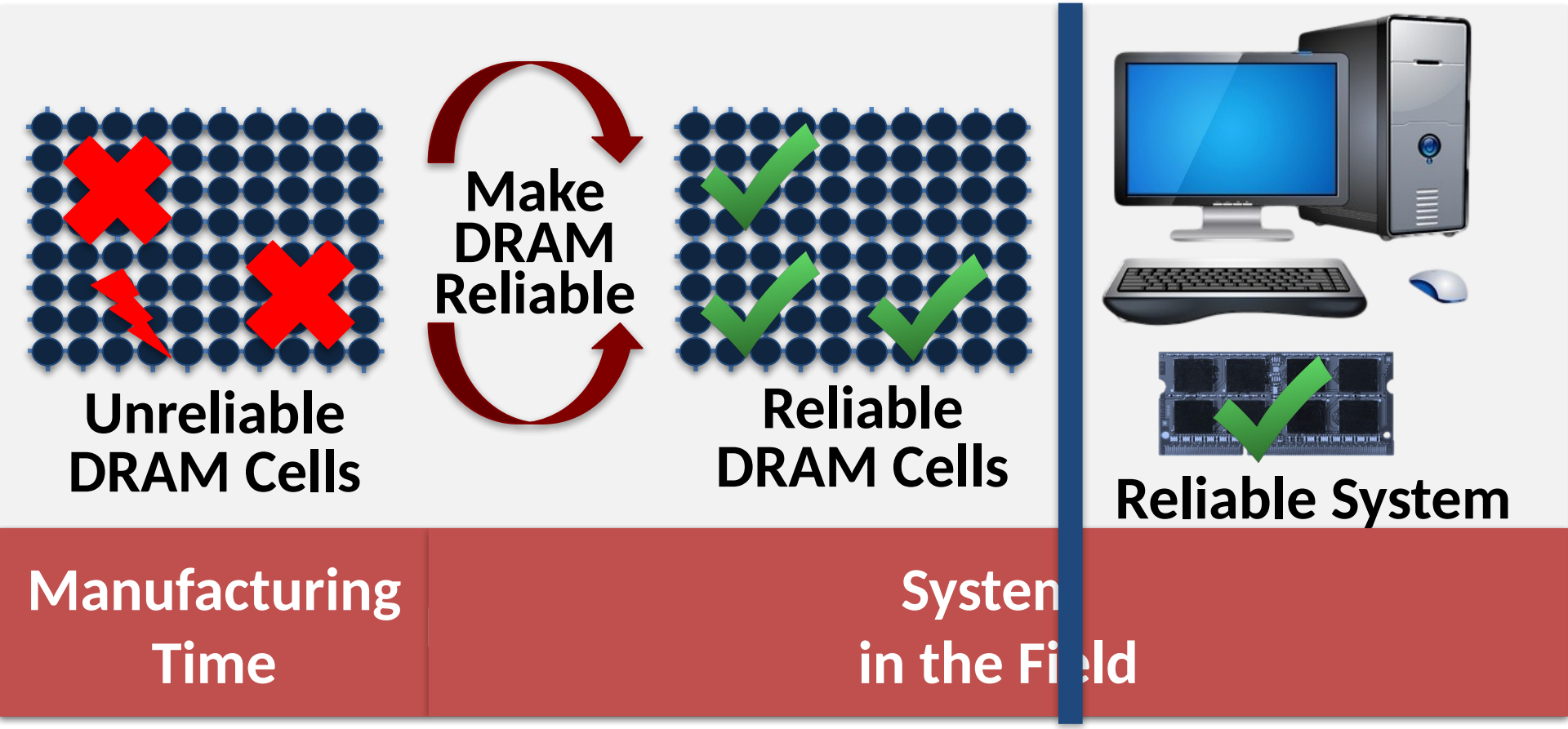
Reliable System

Manufacturing Time

System in the Field

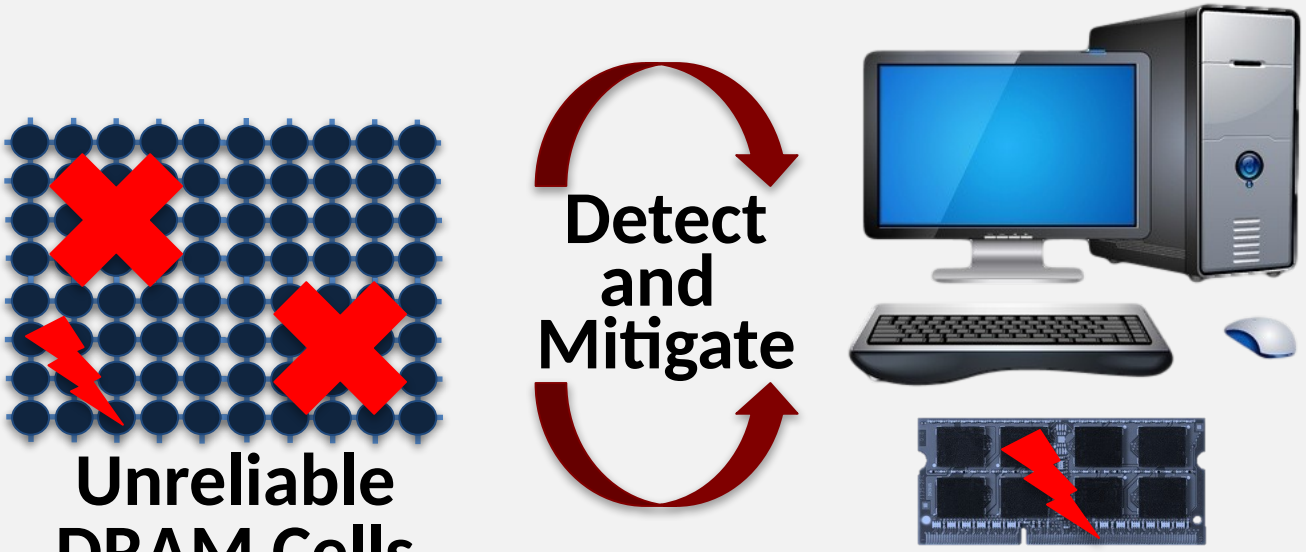
DRAM has strict reliability guarantee

New Approach



Shift the responsibility to systems

System-Level Detection & Mitigation



Detect and mitigate errors after the system has become operational

ONLINE PROFILING
Reduces cost, increases yield, and enables scaling

Breaking the Abstraction

OS needs to know about testing and tested pages

Need to implement the testing in the hardware

Need to know the circuit-level characteristics of the failures

- Samira Khan+, "**The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study**", ***SIGMETRICS 2014***

CSC 2224: Parallel Computer Architecture and Programming Advice on Doing Research

Prof. Gennady Pekhimenko
University of Toronto
Fall 2022

Advancing Research & Development

- We will talk a lot about this in this course

- Learning ***by example***
  - Reading and evaluating strong and seminal papers

- Learning ***by doing***
  - Semester-long research project

- Learning ***by open, critical discussions***
  - Online discussion of papers & ideas on Piazza and Paper Reviews

What is the Goal of Research?

- To generate new insight
  - that can enable what previously did not exist

- Research (in engineering) is a hunt for insight that can eventually impact the world

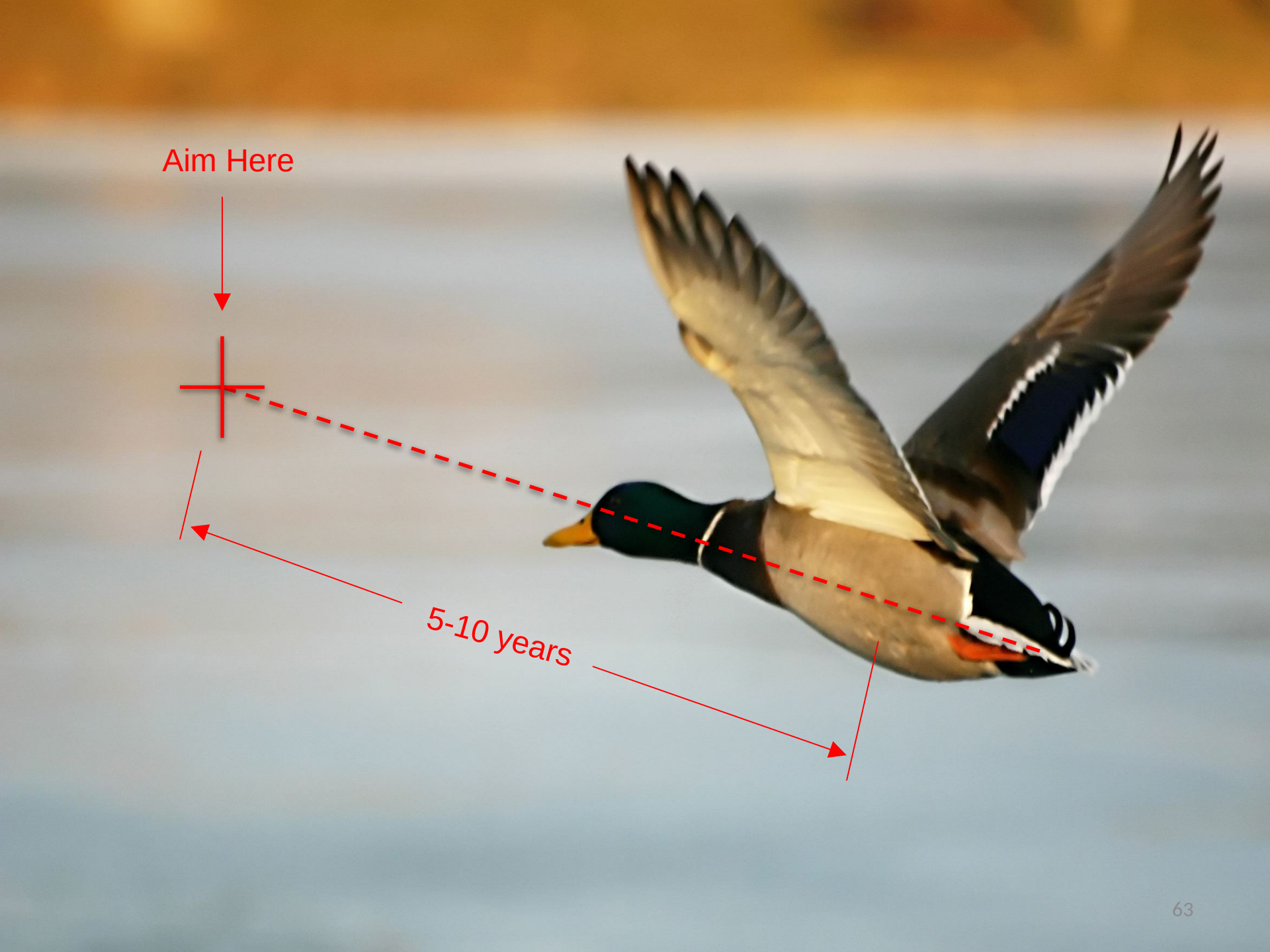
Basic Advice for Good RESEARCH

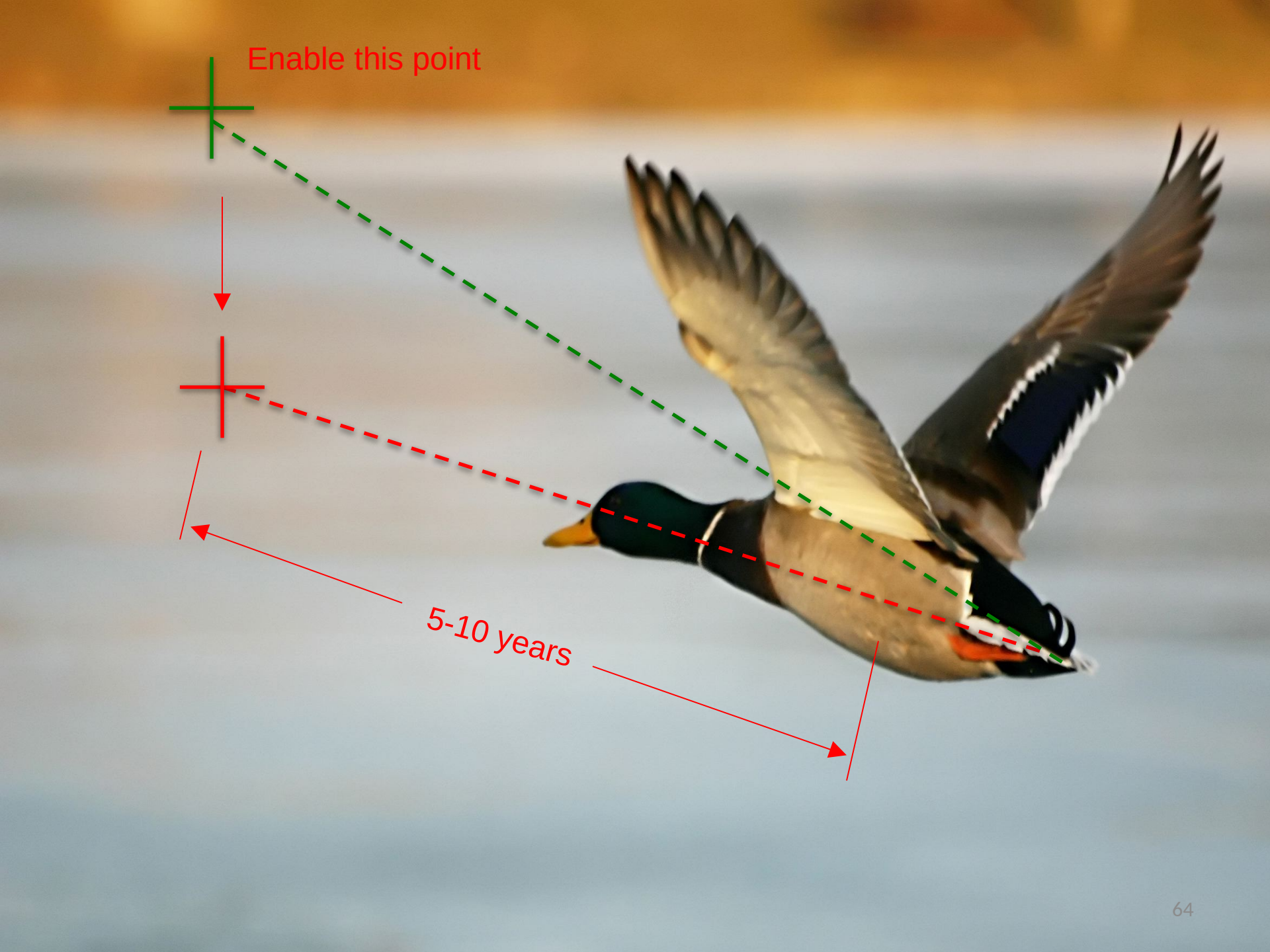
- Choose great problems to solve: Have great taste
  - Difficult
  - Important
  - High impact
- Read heavily and critically
- Think big (out of the box)
  - Do not restrain yourself to tweaks
- Aim high
- Write and present really well


Looking here for lost keys


Lost keys here
Looking here


Current Architecture Practice


Aim Here
5-10 years


Enable this point
5-10 years

The Research Formula

Reward

If you are wildly successful, what difference will it make?

Effort

Learn as much as possible with as little work as possible

Effort

Do the minimum analysis and experimentation necessary to make a point

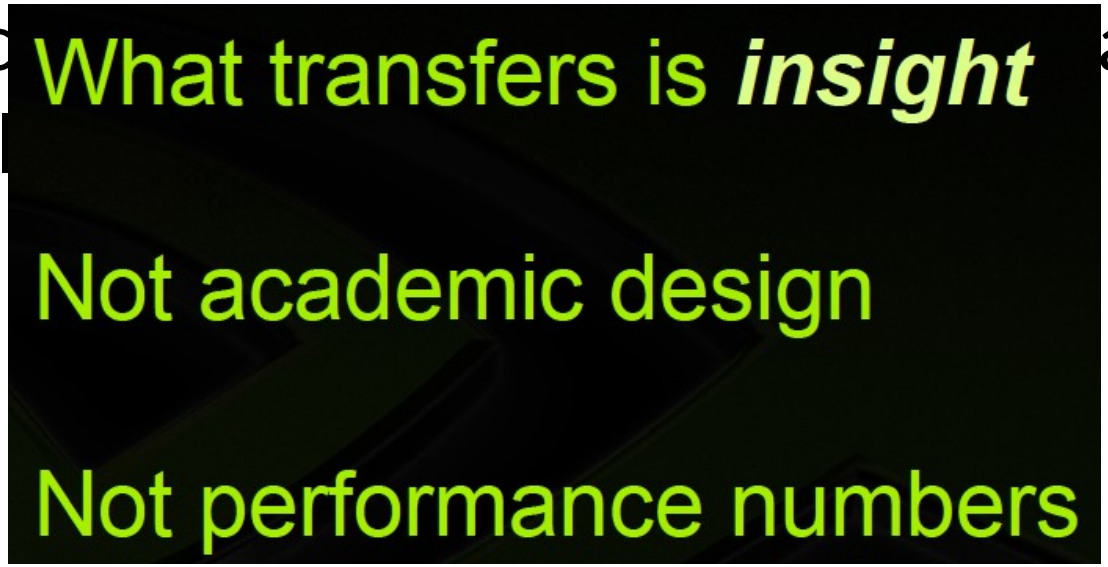
Research is a
hunt for insight

Need to get off the beaten path to find new insights

Recommended Talk

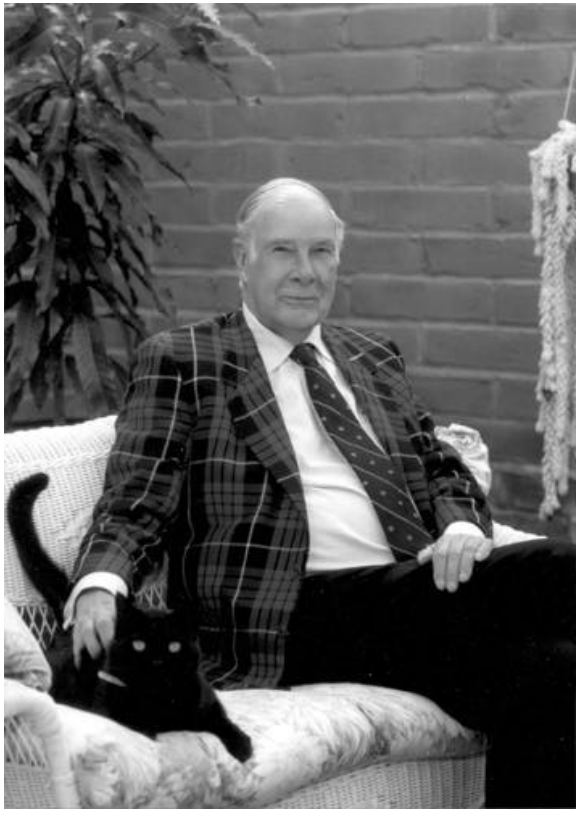
- Bill Dally,
  [Moving the needle: Effective Computer Architecture Research in Academy and Industry]

  ISCA 2010 Keynote Talk.

- Ac 

are from this ta

More Good Advice



"The purpose of computing is insight, not numbers"

Richard Hamming

CSC 2224: Parallel Computer Architecture and Programming Introduction, Logistics

Prof. Gennady Pekhimenko
University of Toronto
Fall 2022