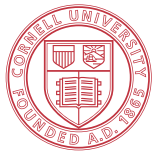
ECE 6775
High-Level Digital Design Automation
Fall 2025

Course Overview

Zhiru Zhang
School of Electrical and Computer Engineering



Cornell University

CSL

Agenda

- Important logistics
- Course motivation
- More course organization

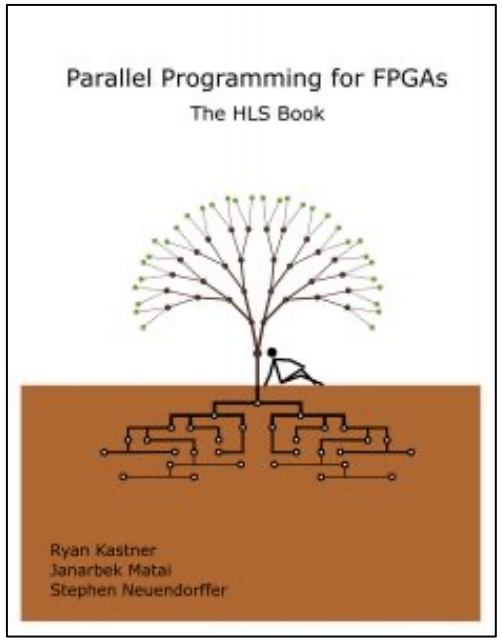
Class Resources

- Course website
  - https://www.csl.cornell.edu/courses/ece6775
  - Lectures slides, handouts, and other readings

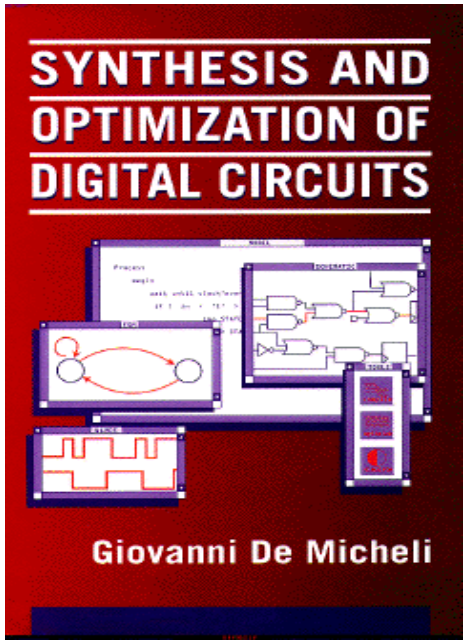
- Ed Discussion
  - Announcements and Q&A

- CMSX: course management system
  - Assignments and grades
  - Electronic submissions required

Course Texts



[e-book available online]



Get 1st edition
[Overhead slides available online]

Selected papers & software manuals

Seeking Help After Class

- Ed Discussion
  - Questions on lectures, assignments, projects, etc.
  - Monitored by course staff
- Instructor office hours (online)
  - Thursday 5:00-6:00pm, Zoom link posted on Ed
- Email instructor for personal issues/appointment
- PhD TA: Niansong Zhang (nz264)
  - TA OH schedule will be announced soon

Grading Scheme

- Class participation (4%)
  - Asking & answering questions during lectures
  - Contributing to discussions on Ed
- Paper readings (5%)
- Quizzes (6%)
- Prelim exam (20%)
- Assignments (30%)
- Final project (35%)

What this Course is About

Hardware/Software ***Co-Design***

- Specify applications/algorithms in software programs
- Synthesize software descriptions into special-purpose hardware components, namely, *accelerators*
  - Perform manual source-level code optimizations
  - Utilize automatic compiler & synthesis optimizations
  - Explore performance-cost trade-offs
- Realize the synthesized accelerators on FPGAs

This Course Dives into EDA

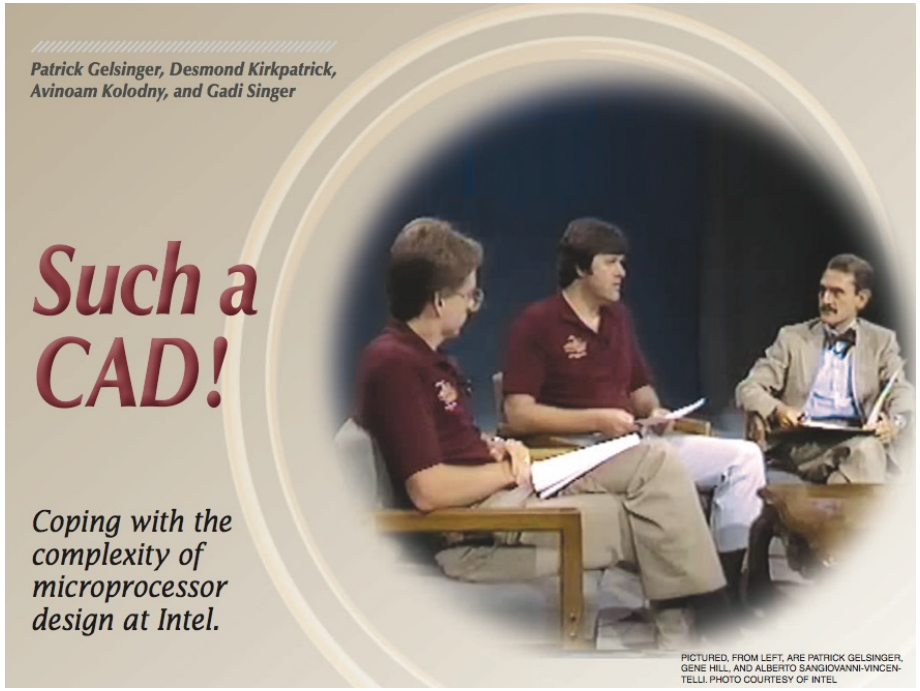
*E*lectronic *D*esign *A*utomation

- A general methodology for refining **a high-level description down to a detailed physical implementation** for designs ranging from
  - integrated circuits (including system-on-chips),
  - printed circuit boards (PCBs),
  - and electronic systems

- **Modeling**, **synthesis**, and **verification** at every level of **abstraction**

[source: NSF'09 EDA Workshop]

Significance of EDA



Patrick Gelsinger, Desmond Kirkpatrick, Avinoam Kolodny, and Gadi Singer. "Such a CAD!" *IEEE Solid-State Circuits Magazine*, 2010.

TABLE 1. INTEL PROCESSORS, 1971–1993.

PROCESSOR	INTRO DATE	PROCESS	TRANSISTORS	FREQUENCY
4004	1971	10 μm	2,300	108 KHz
8080	1974	6 μm	6,000	2 MHz
8086	1978	3 μm	29,000	10 MHz
80286	1982	1.5 μm	134,000	12 MHz
80386	1985	1.5 μm	275,000	16 MHz
Intel 486 DX	1989	1 μm	1.2 M	33 MHz
Pentium	1993	0.8 μm	3.1 M	60 MHz

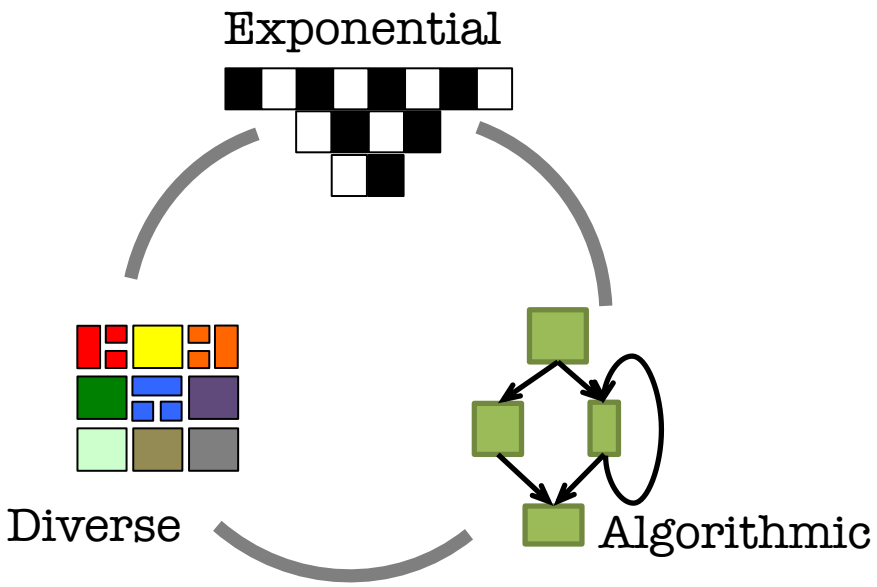
> "This incredible growth rate could not be achieved by hiring an exponentially growing number of design engineers. It was fulfilled by adopting new design methodologies and by introducing innovative design automation software at every processor generation."

E-D-A: My Other Interpretation

Exponential
in complexity (or **E**xtreme scale)

Diverse
increasing system heterogeneity
multi-disciplinary

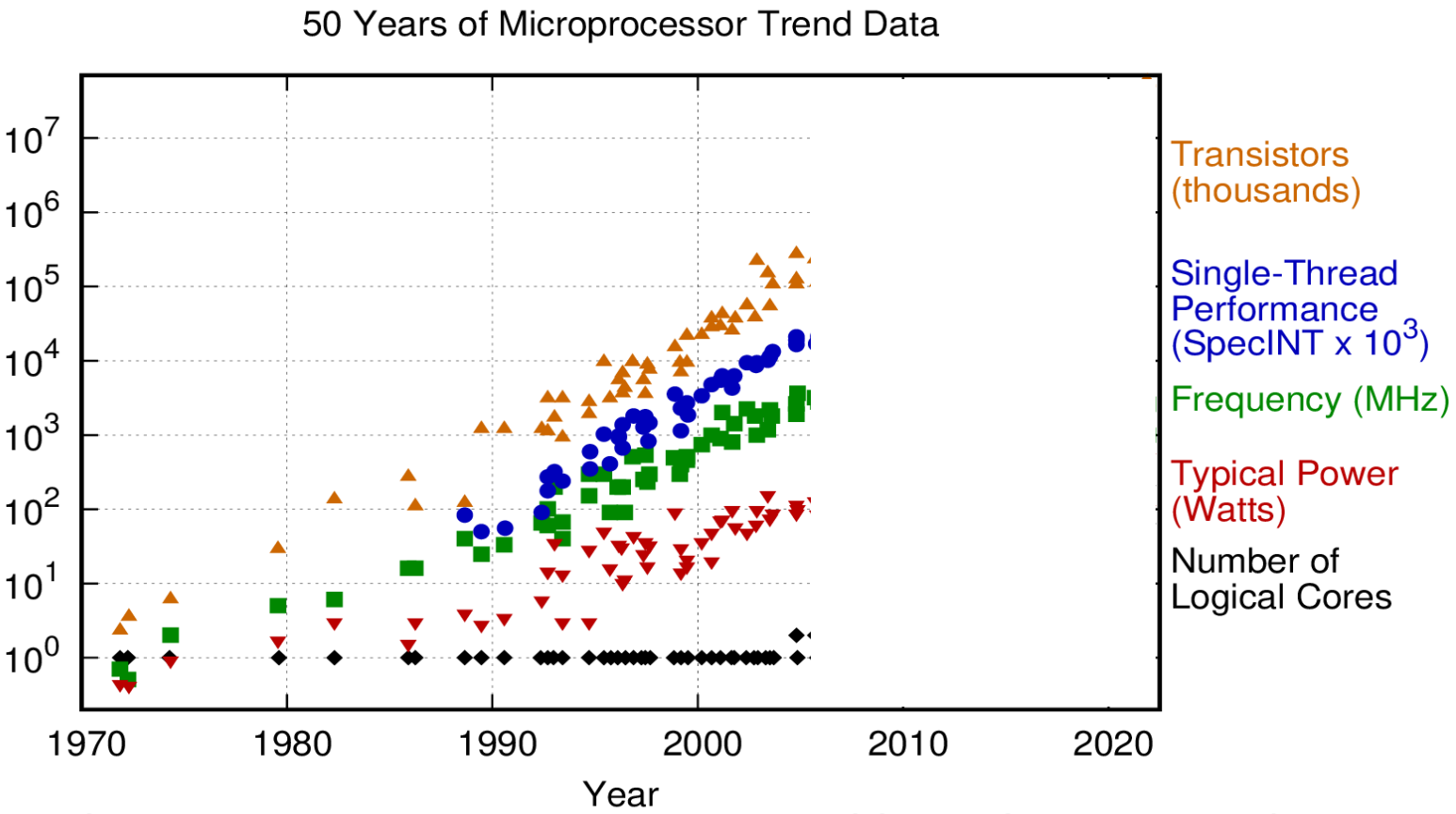
Algorithmic
intrinsically computational



Exponential: Technology Scaling (pre-2005)

Transistor counts doubled every ~2 years (**Moore's Law**)

Clock frequencies also rose exponentially (**Dennard scaling**)

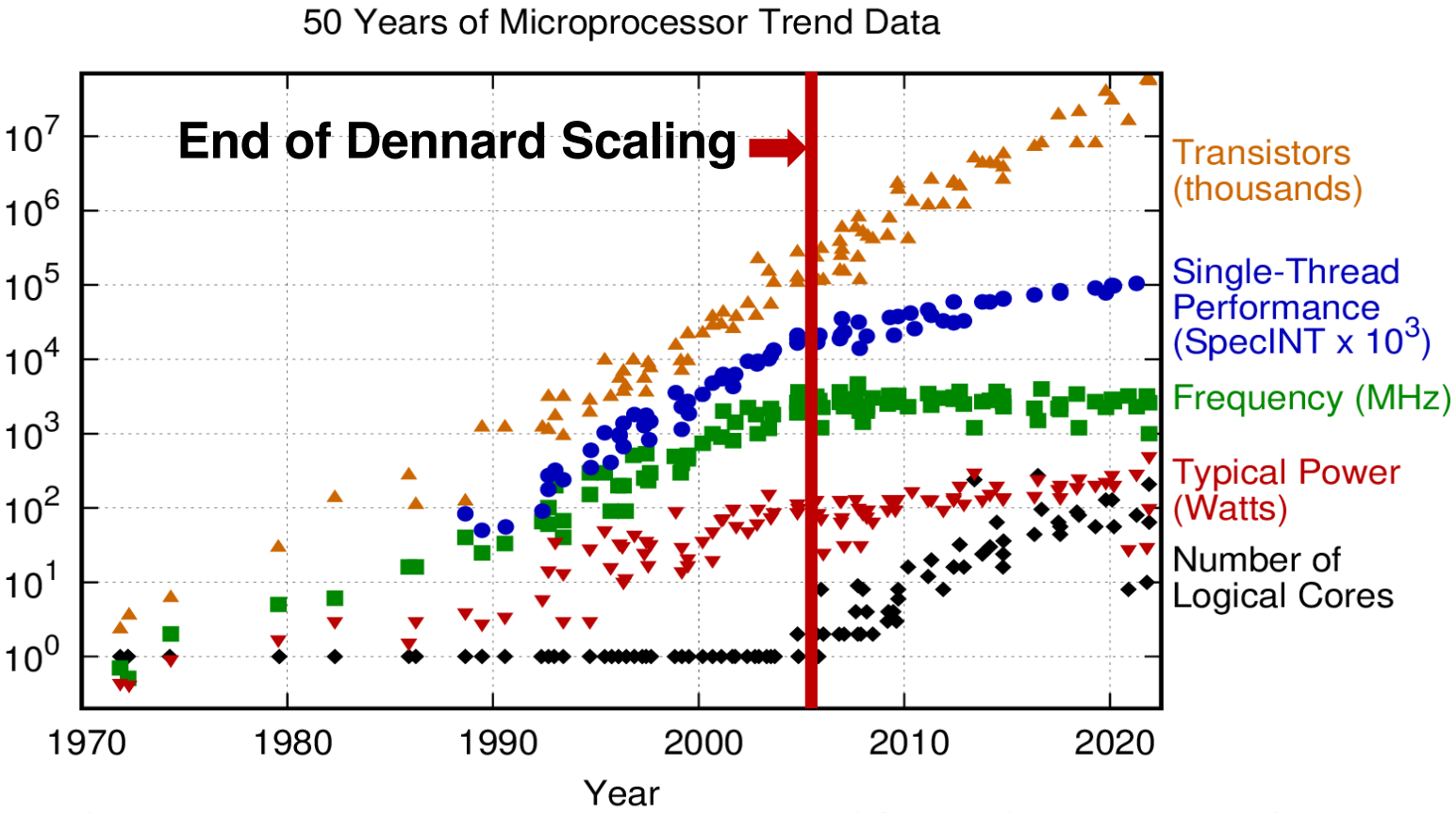


Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

Exponential: Technology Scaling (post-2005)

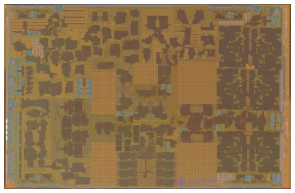
Transistor counts continue to scale

Frequency scaling plateaued as power becomes a limiting factor => led to multicore & greater emphasis on energy efficiency

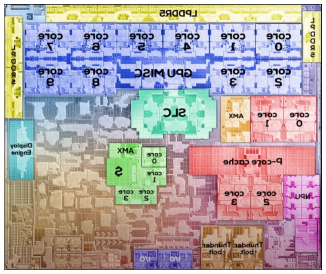


Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2021 by K. Rupp

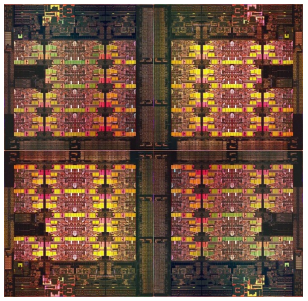
Era of Billion-Transistor Chips



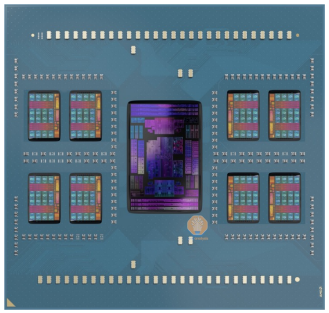
Apple A17 Pro
~19B transistors



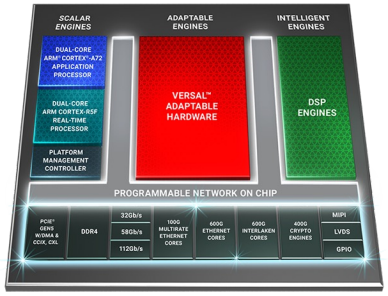
Apple M3
~28B transistors



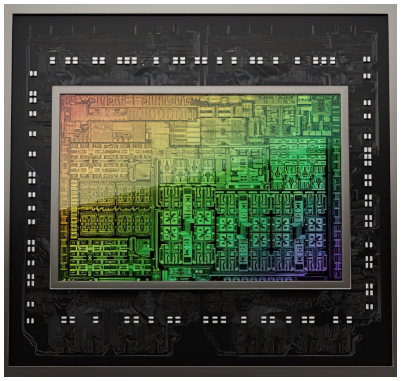
Intel Sapphire Rapids
(quad-chip module)
~48B transistors



AMD EPYC Bergamo
(9-chip module)
~82B transistors



AMD (Xilinx) Versal Premium
~92B transistors



NVIDIA Blackwell B200
~208B transistors

Power-Constrained Modern Computers

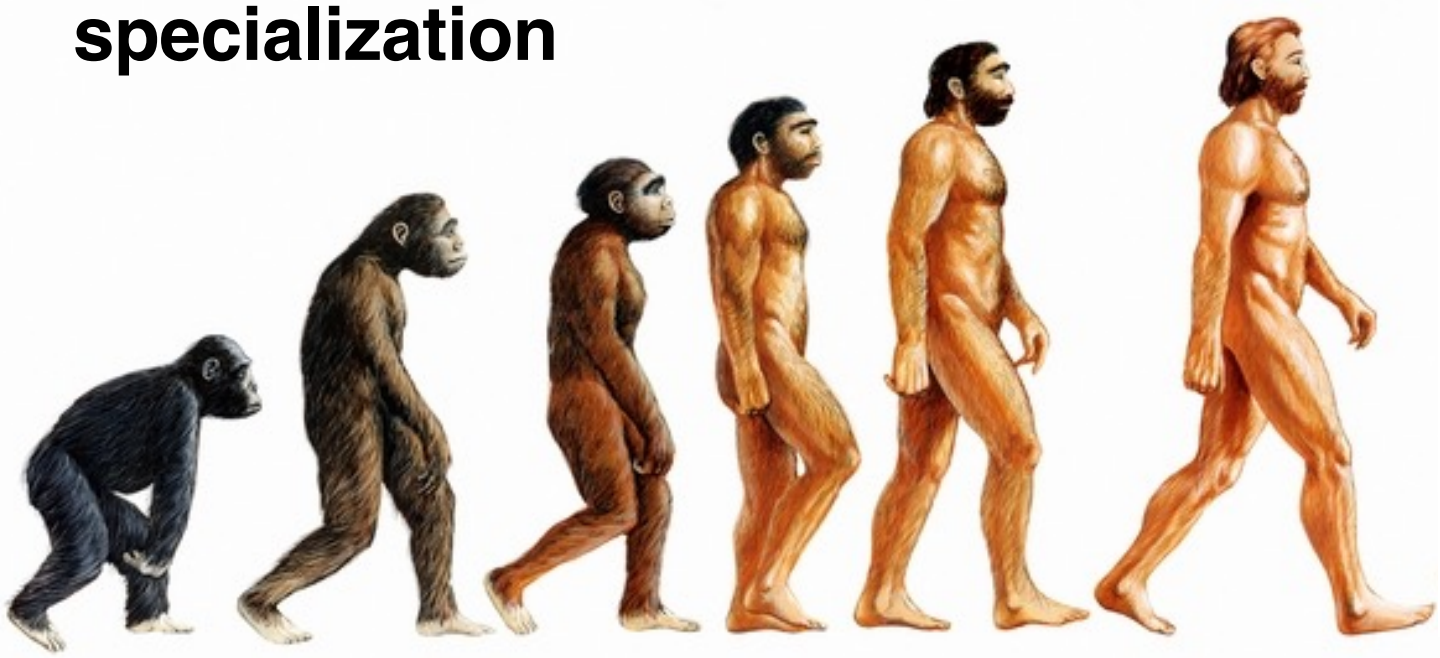
<<1W/chip ~1W/chip ~15W/chip ~50W/chip ~100W/chip >100W/chip

$$Power = \frac{Energy}{Second} = \frac{Energy}{Op} \times \frac{Ops}{Second}$$

To increase performance (Ops/sec) in a power-constrained regime, energy per operation must decrease—in other words, **energy efficiency** (Ops/Joule) **needs to improve!**

Advance of Civilization

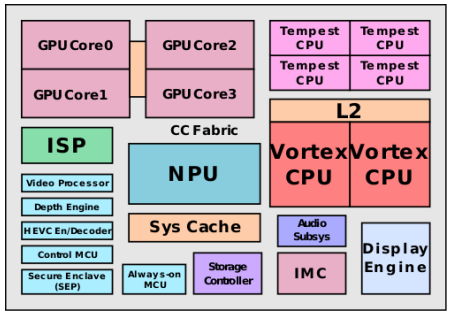
- For humans, Moore's Law scaling of the brain has ended a long time ago
  - Number of neurons and their firing rate did not change significantly
- Remarkable advancement of civilization via **specialization**



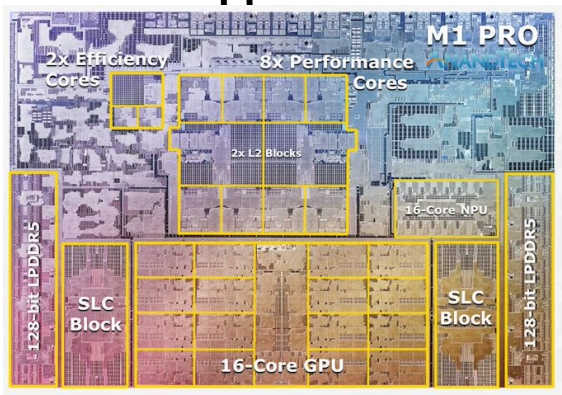
?

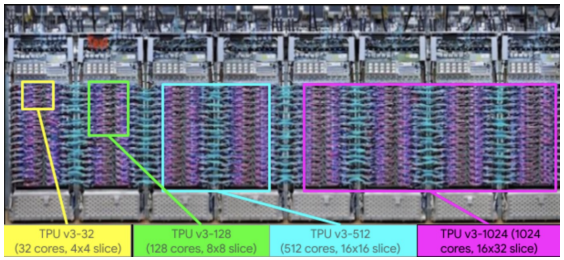
Diverse: Era of Hardware Heterogeneity

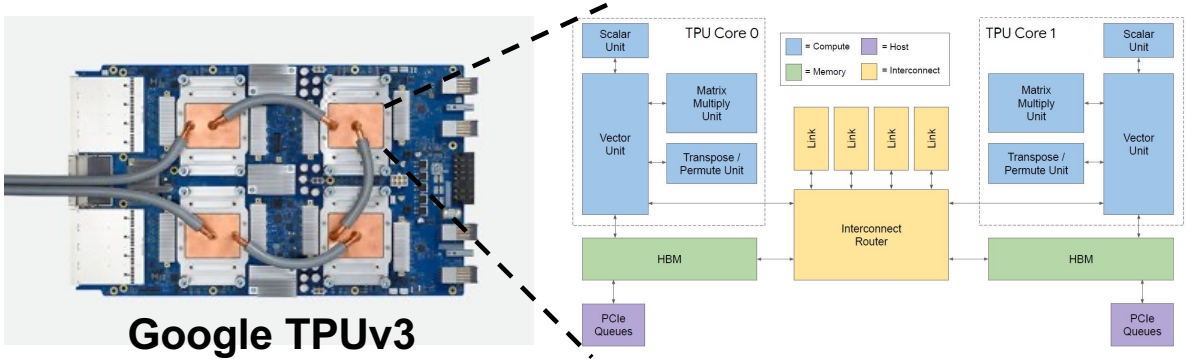
Apple 12 (iPhone X)



Apple M1 Pro





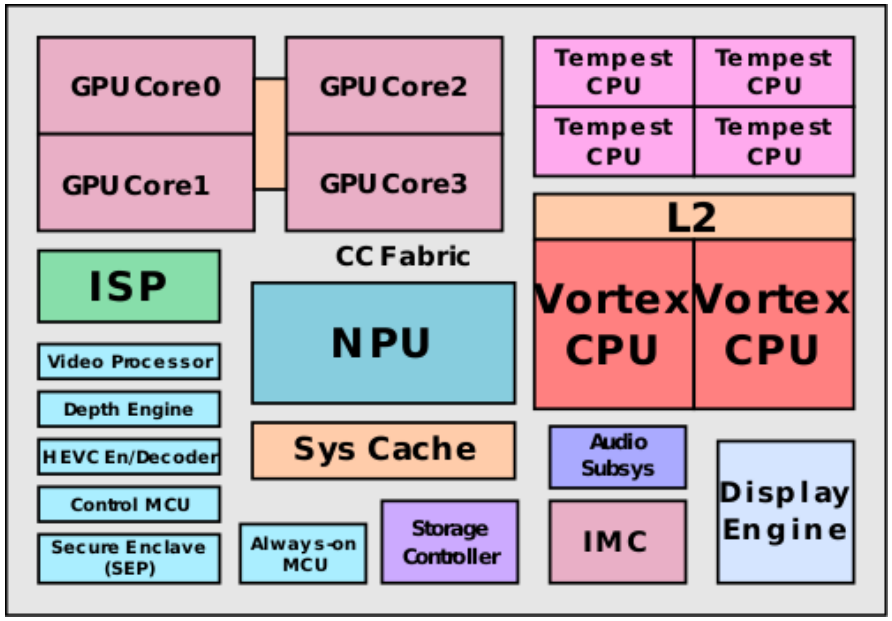


Google TPUv3

Special-purpose accelerators are increasingly used to improve performance & energy efficiency both in datacenters and at the edge

Hardware Specialization in Mobile Chips

System on chip (SoC)

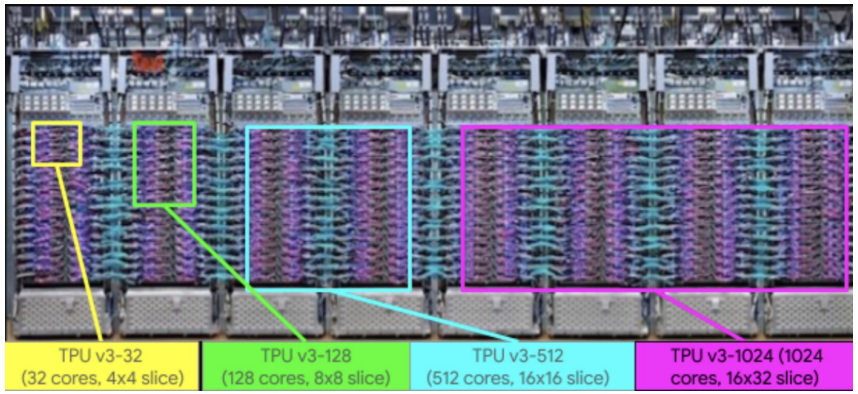


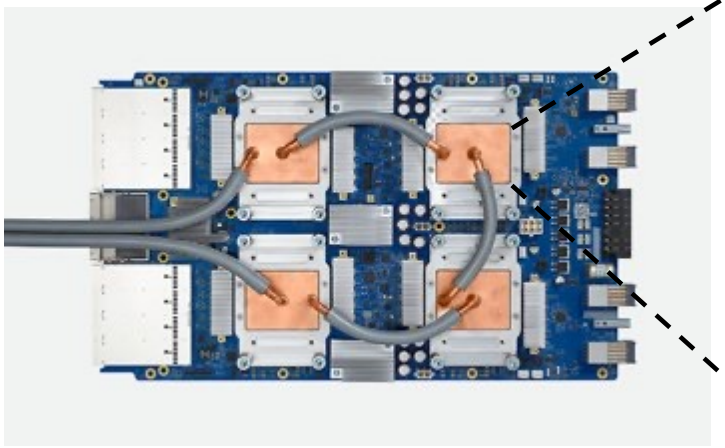
Apple 12 (iPhone X)

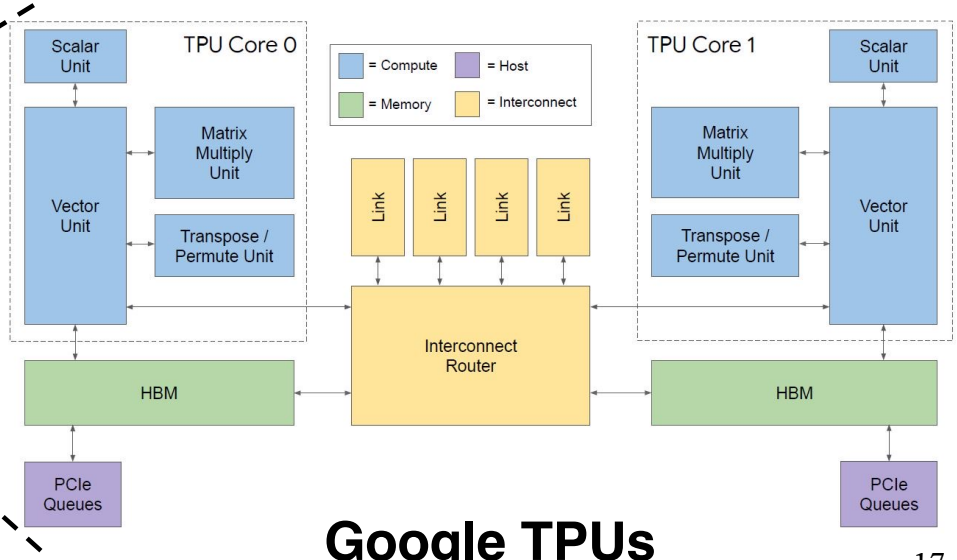
- Modern SoCs integrate a rich set of **special-purpose accelerators**
  - Speed up critical tasks
  - Reduce power consumption and cost
  - Increase energy efficiency

Hardware Specialization in Datacenters

Application- and domain-specific accelerators are being deployed for a rich mix of compute-intensive applications in cloud datacenters



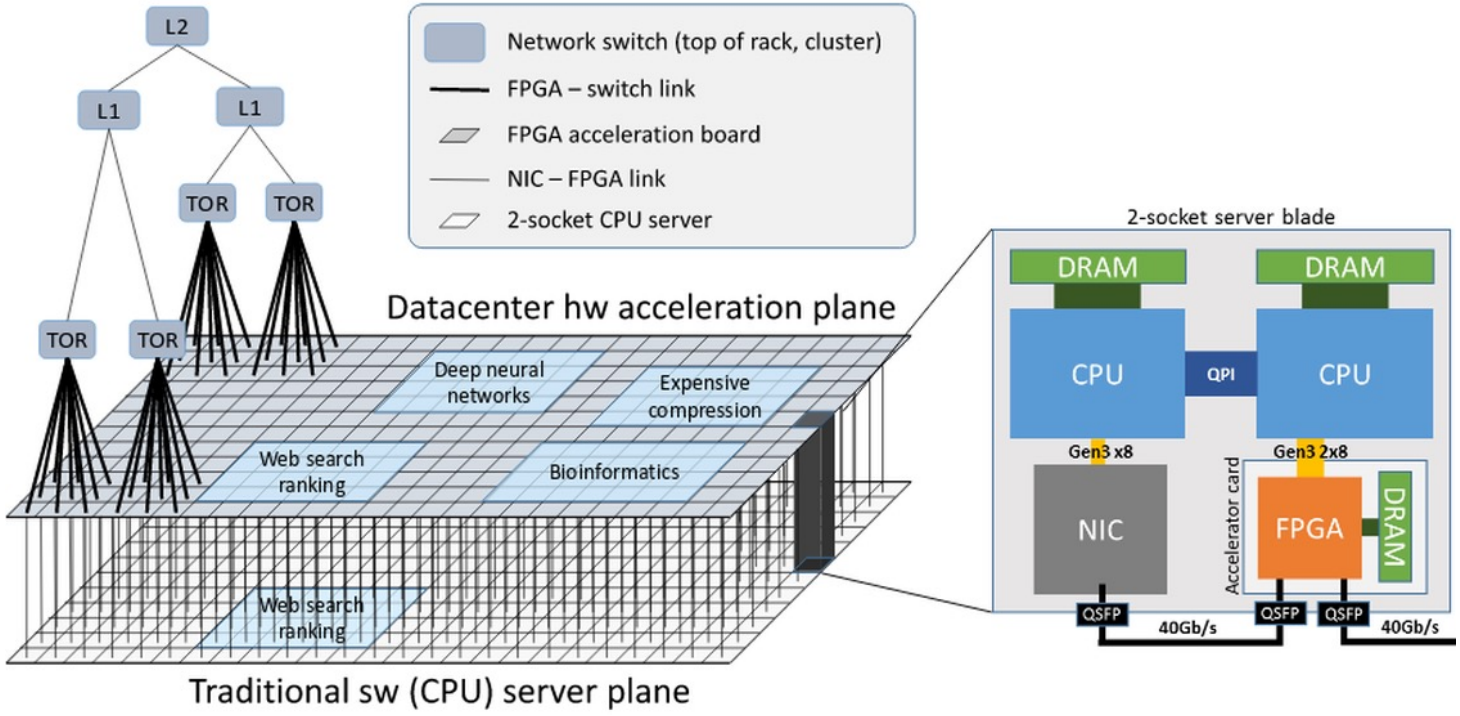




Google TPUs

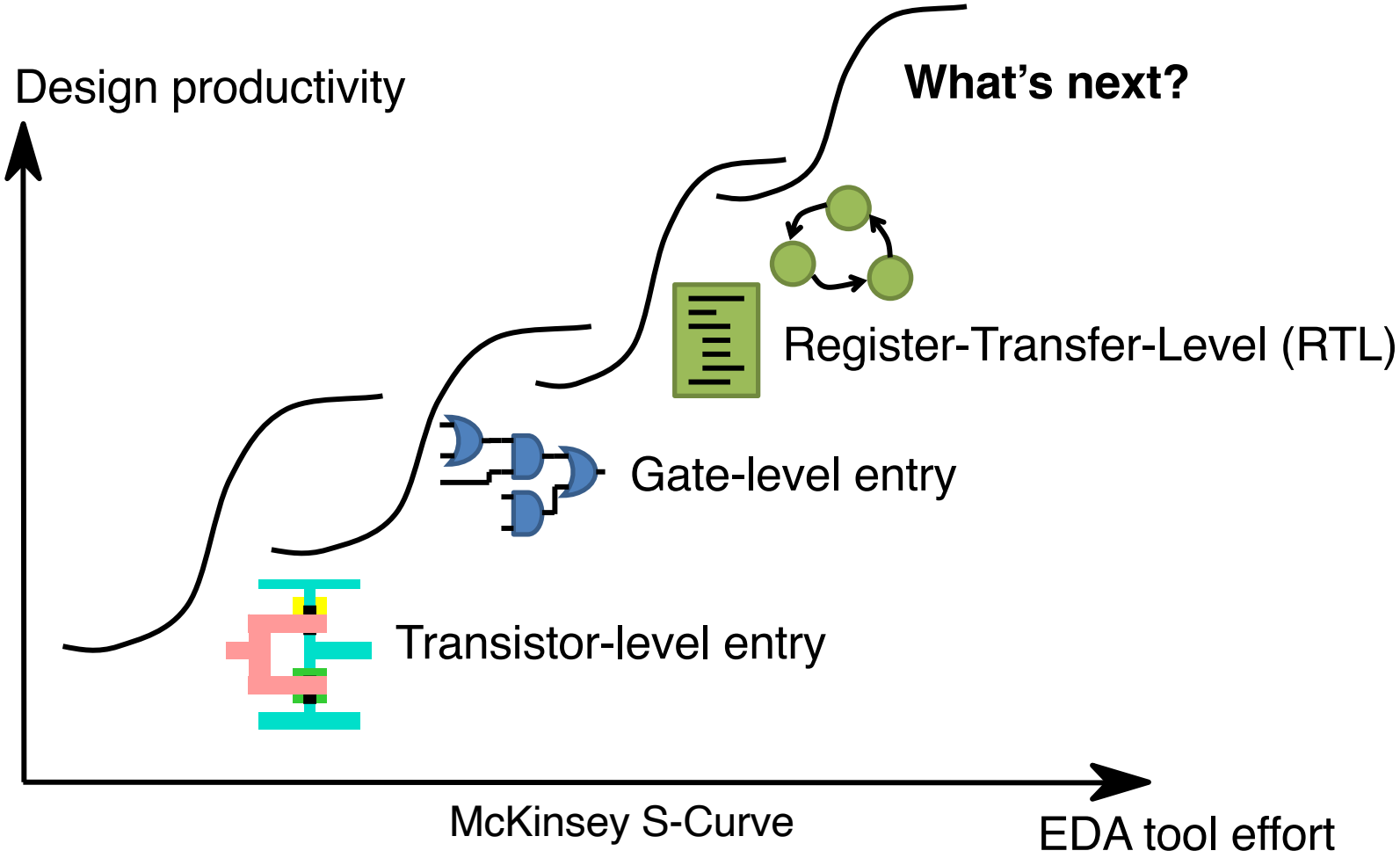
Hardware Specialization in Datacenters

Application- and domain-specific accelerators are being deployed for a rich mix of compute-intensive applications in cloud datacenters



Microsoft Cloud FPGA Platforms

Increasing Specialization Demands (Even) Higher Design Productivity



[source: Kurt Keutzer, UCB]

Motivation for High-Level Synthesis (HLS)

```
module dut(rst, clk, q);
  input rst;
  input clk;
  output q;
  reg [7:0] c;

  always @ (posedge clk)
  begin
    if (rst == 1b'1) begin
      c <= 8'b00000000;
  end
  else begin
    c <= c + 1;
  end

  assign q = c;
endmodule
```

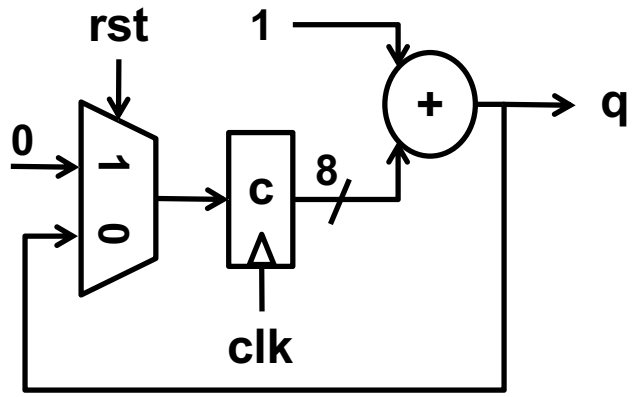
RTL Verilog

vs.

```
uint8 dut() {
  static uint8 c;
  c+=1;
}
```

Automated with HLS

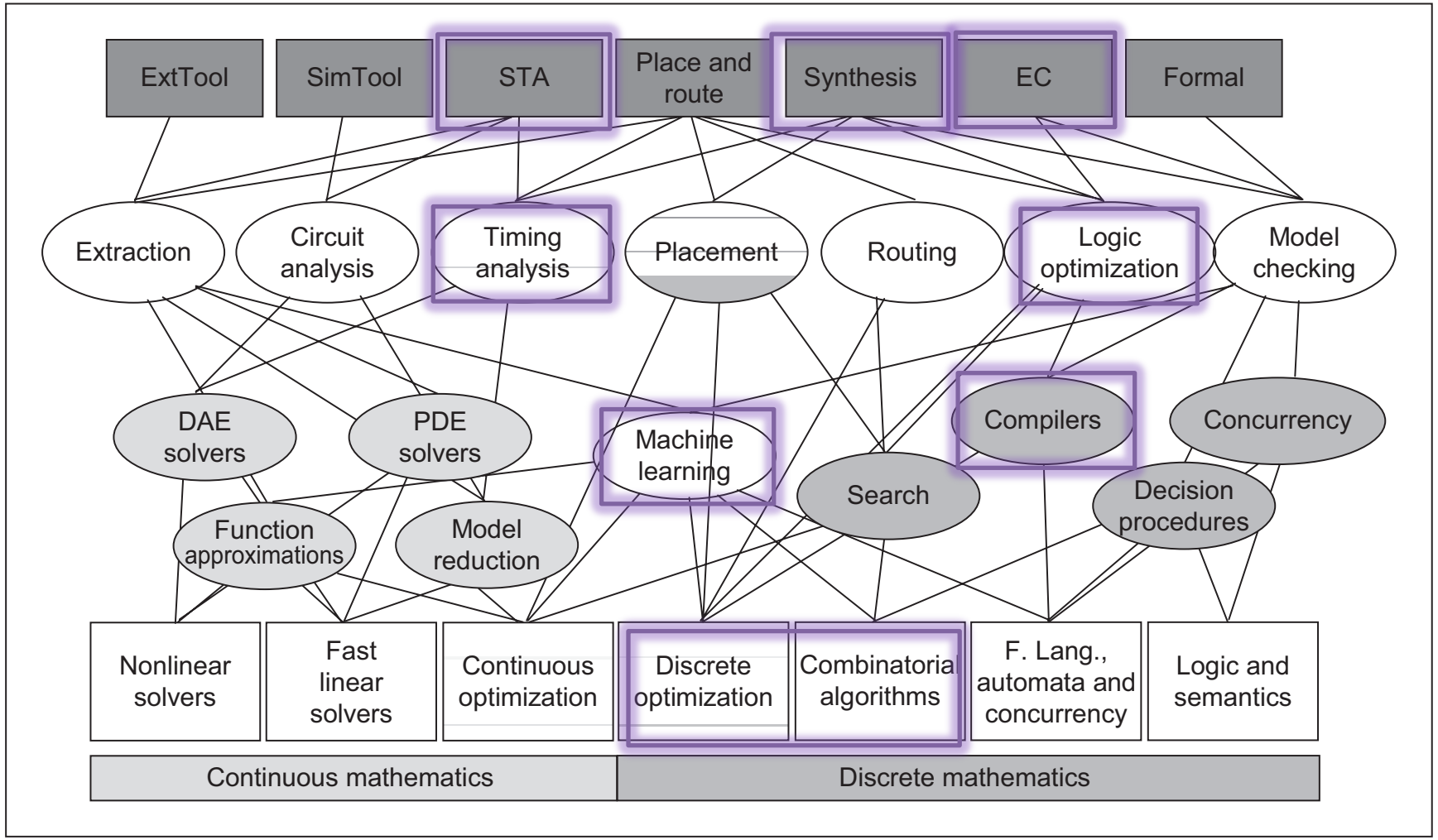
=



An 8-bit counter

Algorithms Drive Automation

Topics touched on in 6775



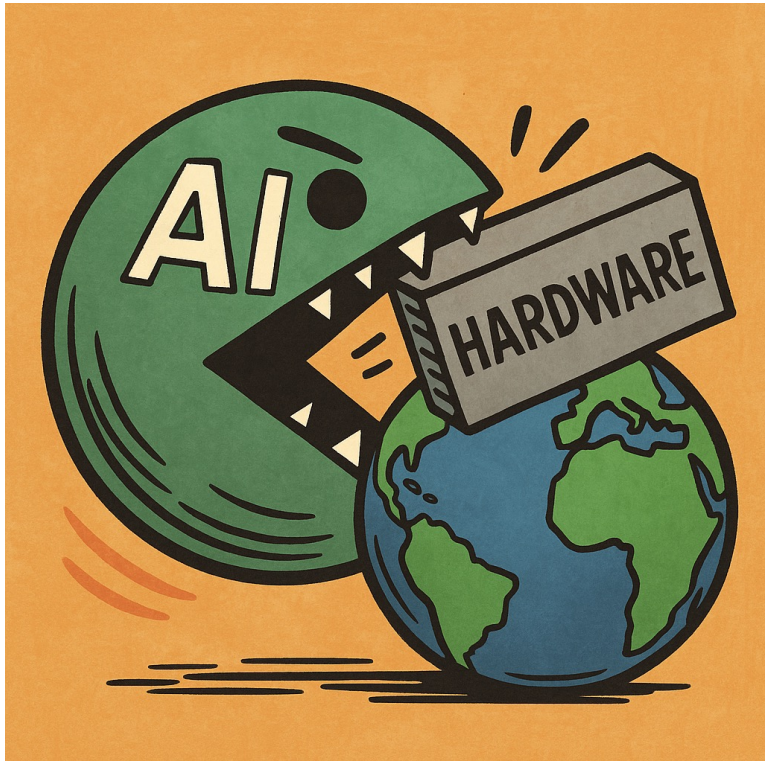
Key Algorithms in EDA

[source: Andreas Kuehlmann, Synopsys Inc.]

Another A We Can't Ignore

- **AI is "eating the world"**, rapidly transforming every domain

- Much of AI's potential is constrained by hardware
  - Constraints include **compute, memory, bandwidth, and energy**
  - Availability and cost of GPUs/AI chips are major hurdles
  - Need to **optimize for existing hardware** and **design for the next generation**



Course Organization

- Refer to syllabus for course organization details

Course Syllabus
ECE 6775 High-Level Digital Design Automation
Fall 2025, Tuesday and Thursday 11:40am-12:55pm, Hollister 366

---

1. Course Information

Lectures: TuTh 11:40am-12:55pm, 366 Hollister Hall
Website: http://www.csl.cornell.edu/courses/ece6775
CMS: https://cmsx.cs.cornell.edu
Ed: https://edstem.org/us/courses/81323

Instructor: Zhiru Zhang, zhiruz@cornell.edu
Office Hours: Thursday 5:00-6:00pm, Online

Course Texts:
- Lecture slides/notes on course website
- R. Kastner, J. Matai, and S. Neuendorffer, Parallel Programming for FPGAs, arXiv, 2018.
- G. De Micheli, Synthesis and Optimization of Digital Circuits, McGraw-Hill, 1994.

Supplementary Materials:
- S. Dasgupta, C.H. Papadimitriou, and U.V. Vazirani, Algorithms, McGraw-Hill, 2007. [link to online draft]
- Additional reference papers will be posted as a course reader.

2. Course Description and Objectives

Targeted specialization of functionality in hardware has become arguably the best means to achieving improved compute performance and energy efficiency for a plethora of emerging applications. Unfortunately, it is a very unproductive practice to design and implement special-purpose accelerators using the conventional register transfer level (RTL) methodology. For this reason, both academia and industry are seeing an increasing use of high-level synthesis (HLS) to automatically generate hardware accelerators from software programs.

Course Roadmap

- Lectures and paper* discussions
  - **Background**
    - Introduction
    - Hardware specialization
    - Algorithm basics
    - FPGA*
  - **High-level synthesis (HLS)**
    - C-based HLS
    - Front-end compilation
    - Scheduling
    - Resource sharing
    - Pipelining
  - **Advanced topics**
    - Domain-specific programming*
    - Deep learning acceleration*

Preferred Background

- Working knowledge of the following at undergraduate level
  - C/C++
  - Digital logic and basic computer architecture concepts (e.g., adders, clock, registers, pipelining)

- Experiences with the following would increase appreciation & productivity
  - Algorithms and data structures
  - RTL design for FPGA or ASIC

Learning Outcomes: The Tangibles

- High-level digital **design** methodologies
  - Design realistic accelerators *above RTL* using HLS design flow
  - Optimize accelerator performance using both manual source-level transformations and automatic HLS compiler optimizations

- High-level **design automation** algorithms
  - Fundamental compilation & synthesis techniques in HLS
    - e.g., SSA, scheduling, pipelining, resource sharing
  - Useful combinatorial optimization algorithms
    - e.g., graph algorithms, dynamic programming, greedy algorithms, integer linear programming

Learning Outcomes: The Intangibles

1. Develop a principled approach to analyzing accelerator design process and essential design factors (e.g., parallelism, resources, precision)

2. Gain comprehensive insights into accelerator design from the perspective of an HLS compiler

We will achieve these objectives through **a balanced mix of theoretical foundations** (lectures & homework) **and practical applications** (labs & project)

NOT Our Goals

- Teach you the design of microprocessors
- Cover the whole breadth of EDA
- Write RTL code
  - but it should still help you improve your hardware design skills
- Make you an *expert* FPGA programmer
  - FPGAs are mainly used as a platform to *prototype* accelerators

Assignments

- Two problem sets (8%)

- Four lab assignments (22%)
  - Design & programming assignments leveraging high-level synthesis tools and software compilers
  - Experiments to be conducted on **ecelinux** servers
    - % ssh -X <netid>@ecelinux-01.ece.cornell.edu
    - Necessary tools will be installed in common directories

Quizzes and Paper Readings

- Quizzes (6%)
  - You will need to answer pop quiz questions in most lectures
  - TWO lowest scores will be dropped
- Paper Readings (5%)
  - Two reading sessions
  - You are expected to read the paper or book chapter before the lecture, answer (more) quiz questions, and participate in discussions
  - Reading assignment will be announced at least one week in advance

Exam

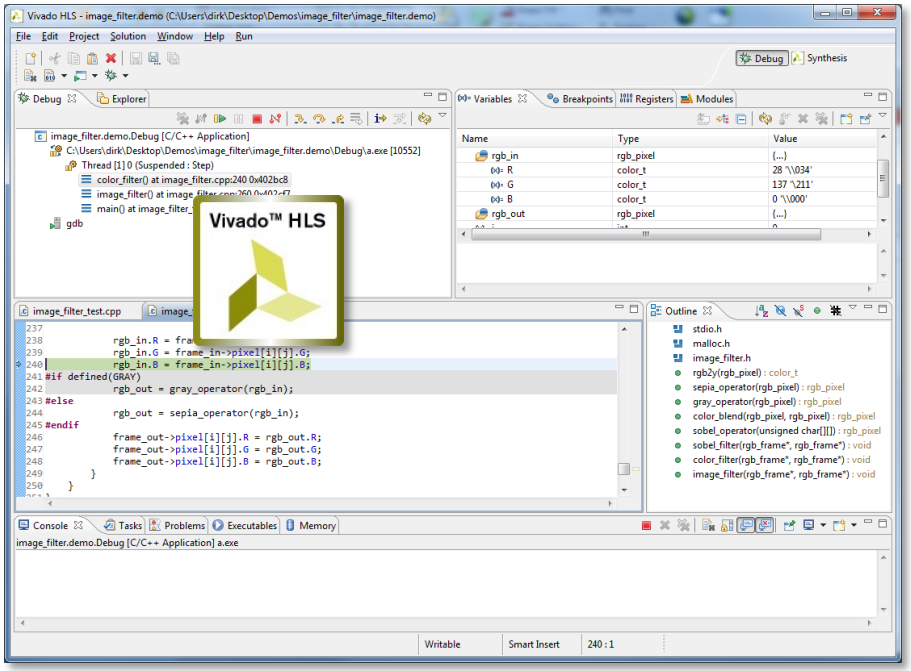
- In-class prelim (20%)
  - Open notes & open book
  - Tentative date: Tuesday October 28th
  - No sit-down final

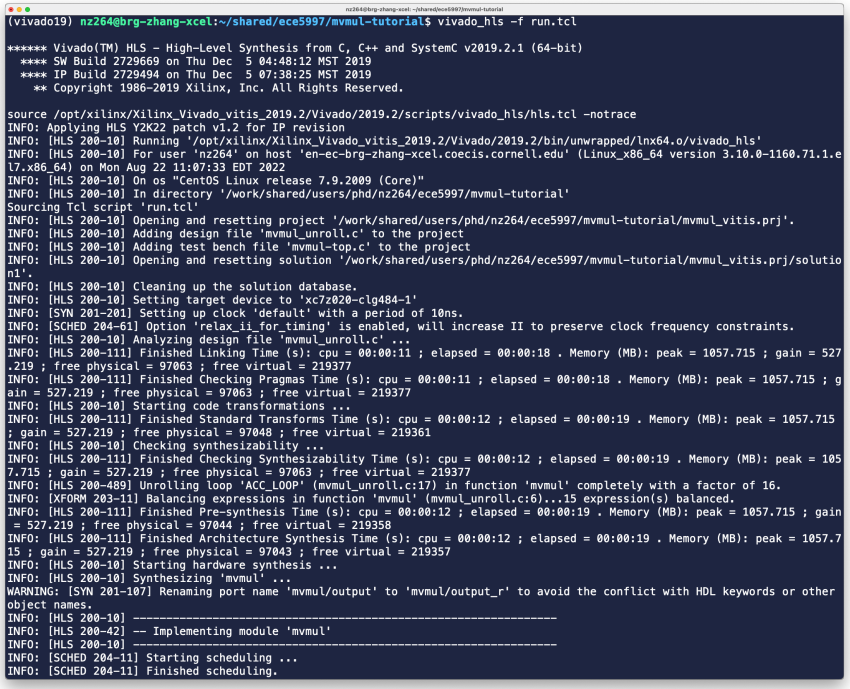
Final Project – 35%

- In-depth exploration of a research topic
  - (1) Designing new application-specific accelerators with HLS; OR
    (2) Devising new automation algorithms/tools
  - 3-4 students / team, depending on class size

- Timeline
  - Proposal due after prelim
  - Weekly meeting with the instructor to track progress
  - Demo before the final week
  - Final report due by the final exam date

High-Level Synthesis Tool



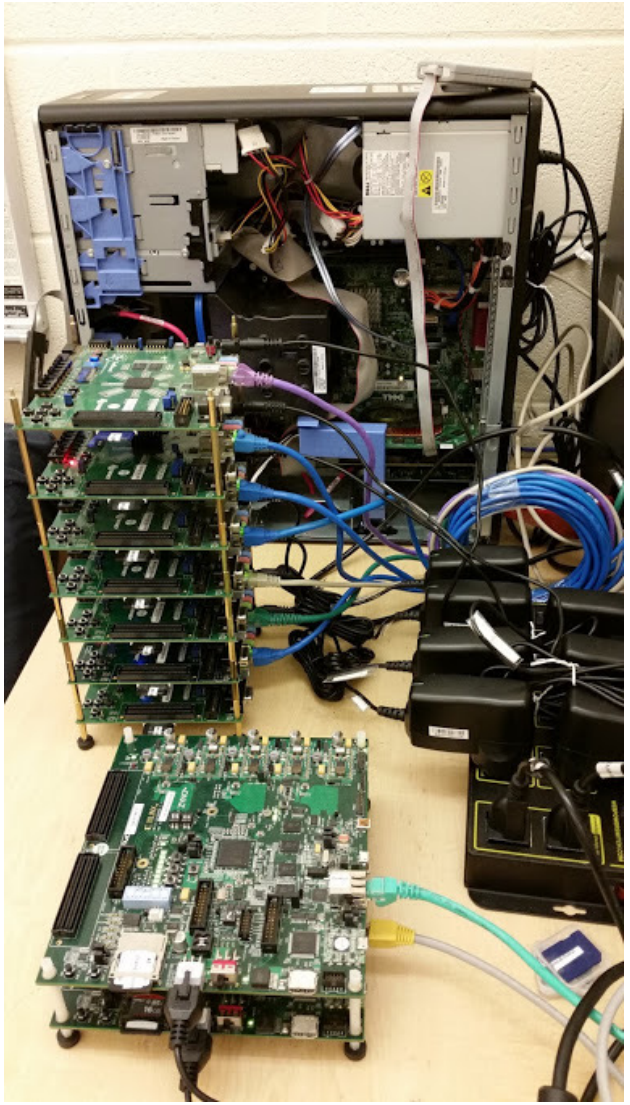


Tutorial on AMD Xilinx **Vivado HLS** v2019.2, Thursday 9/4

(we will be using the command-line interface in this course)

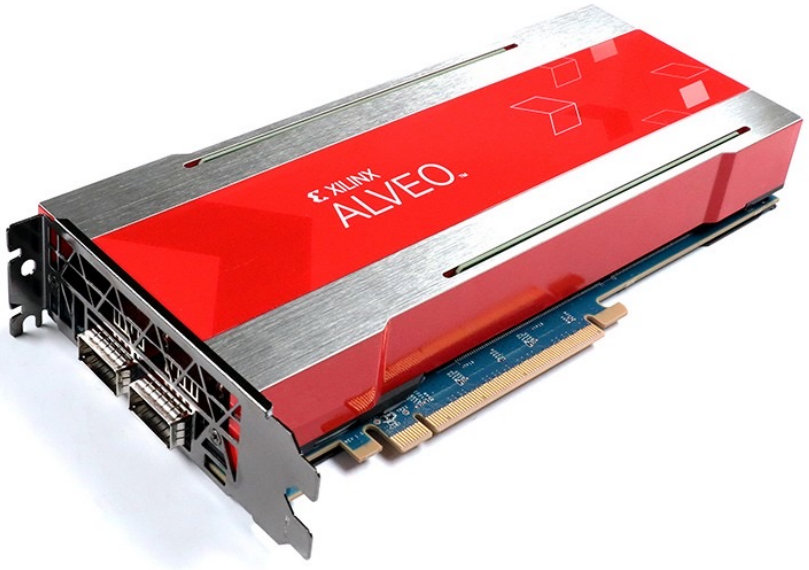
Local Cluster of Embedded FPGAs

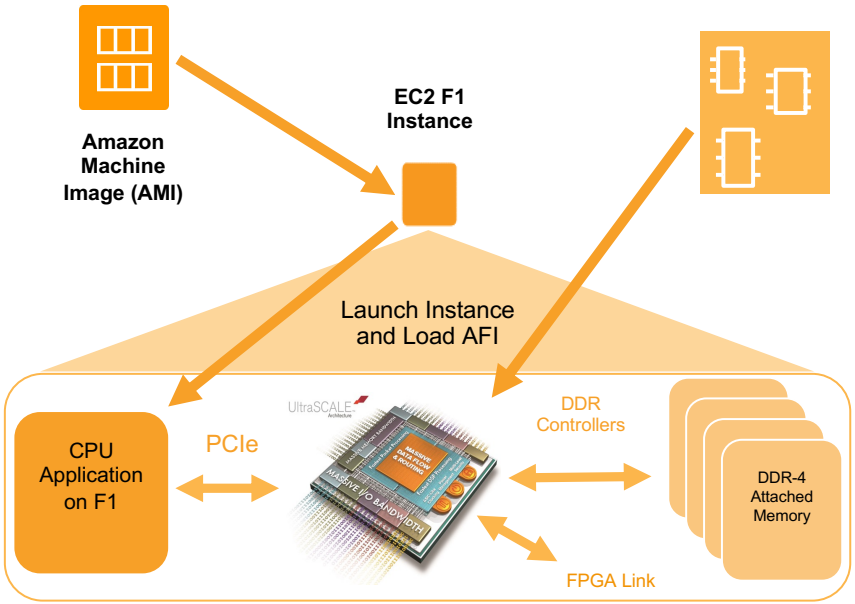
- For labs and project, we will use Zynq-based FPGA development boards (ZedBoard)
  - An FPGA SoC with a dual-core ARM CPU, which boots Linux



Datacenter FPGA Platforms

- For the final project, students can also choose to explore datacenter FPGA platforms such as AMD Xilinx Alveo U280 and AWS F1 cloud instances





Takeaway Points

- End of Dennard scaling leads to increasing **hardware specialization** to sustain improvement in performance and energy efficiency
- Increasing specialization and continued exponential growth in silicon capacity demands higher level of **design abstraction**
- HLS is a promising next step for EDA, which is fueled by sophisticated and yet scalable **algorithms**

Before Next Lecture

- Action items
  - Check out the course website
  - **Read through the course syllabus**
  - **Verify your login on ecelinux**
    - ssh -X <netid>@ecelinux.ece.cornell.edu

Acknowledgements

- These slides contain/adapt materials developed by
  - Prof. Jason Cong (UCLA)
  - Prof. David Z. Pan (UT Austin)