CS152: Computer Systems Architecture
Introduction

Sang-Woo Jun

Winter 2021

UCI

Large amount of material adapted from MIT 6.004, “Computation Structures”,
Morgan Kaufmann “Computer Organization and Design: The Hardware/Software Interface: RISC-V Edition”,
and CS 152 Slides by Isaac Scherson

Why should we learn about computer architecture?

Software developer angle

Hardware architect angle

Why should software engineers learn about architecture?

Image source: www.smartlinkin.com.tw/Article/4826 (from Steve Jobs' 2007 iPhone introduction)

Computer architecture effects example 1

- Multiplying two 2048 x 2048 matrices
  - 16 MiB, doesn't fit in any cache
- Machine: Intel i5-7400 @ 3.00GHz
- Time to transpose B is also counted

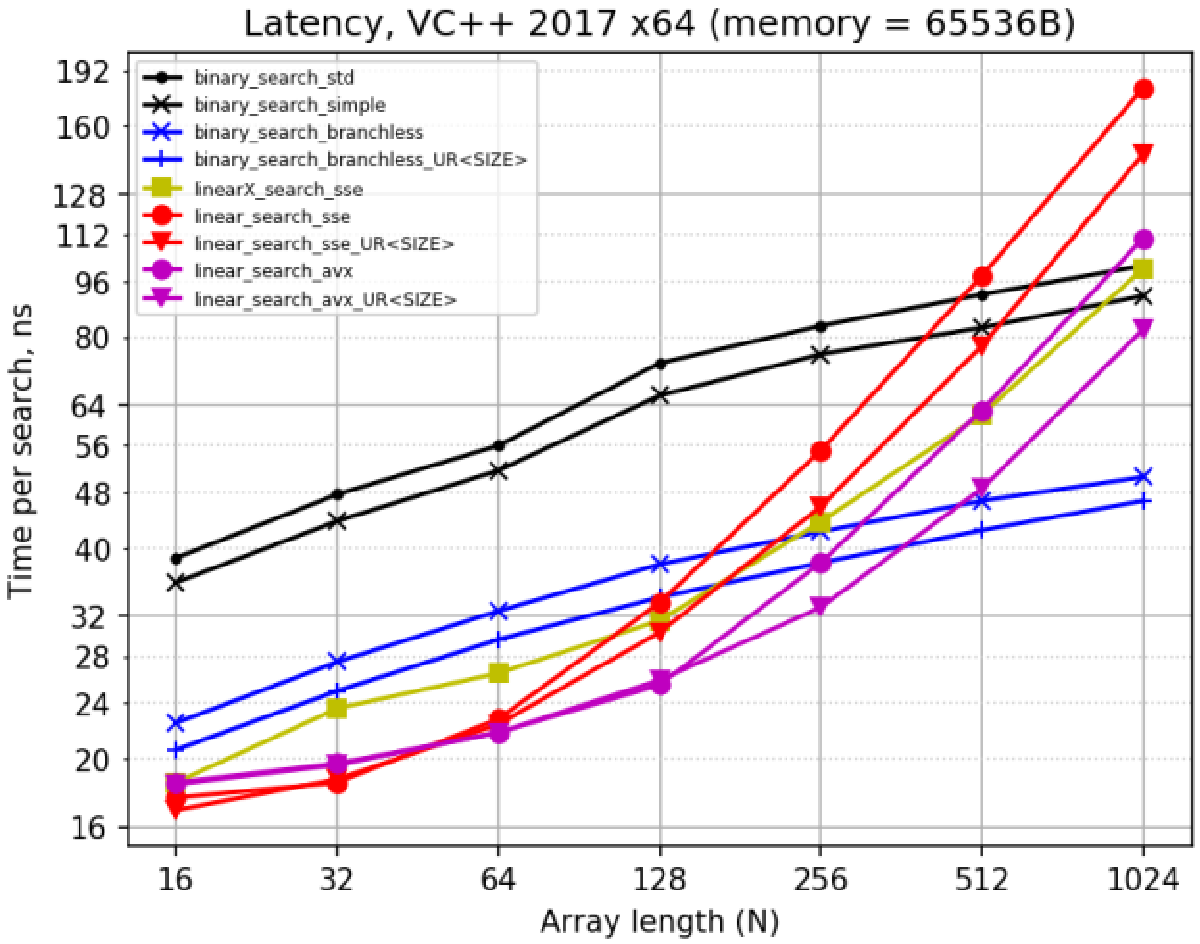
```
for (i=0 to N)
  for (j=0 to N)
    for (k=0 to N)
      C[i][j] += A[i][k] * B[k][j];
```

63.19 seconds

10.39 seconds
(6x performance!)

Computer architecture effects example 2

- Binary search vs. branchless binary search vs. linear search
  - Where does this difference come from, and how do I exploit this?
  - Architecture, assembly knowledge!

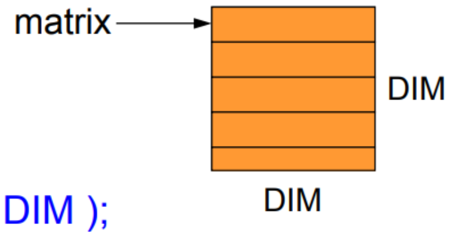


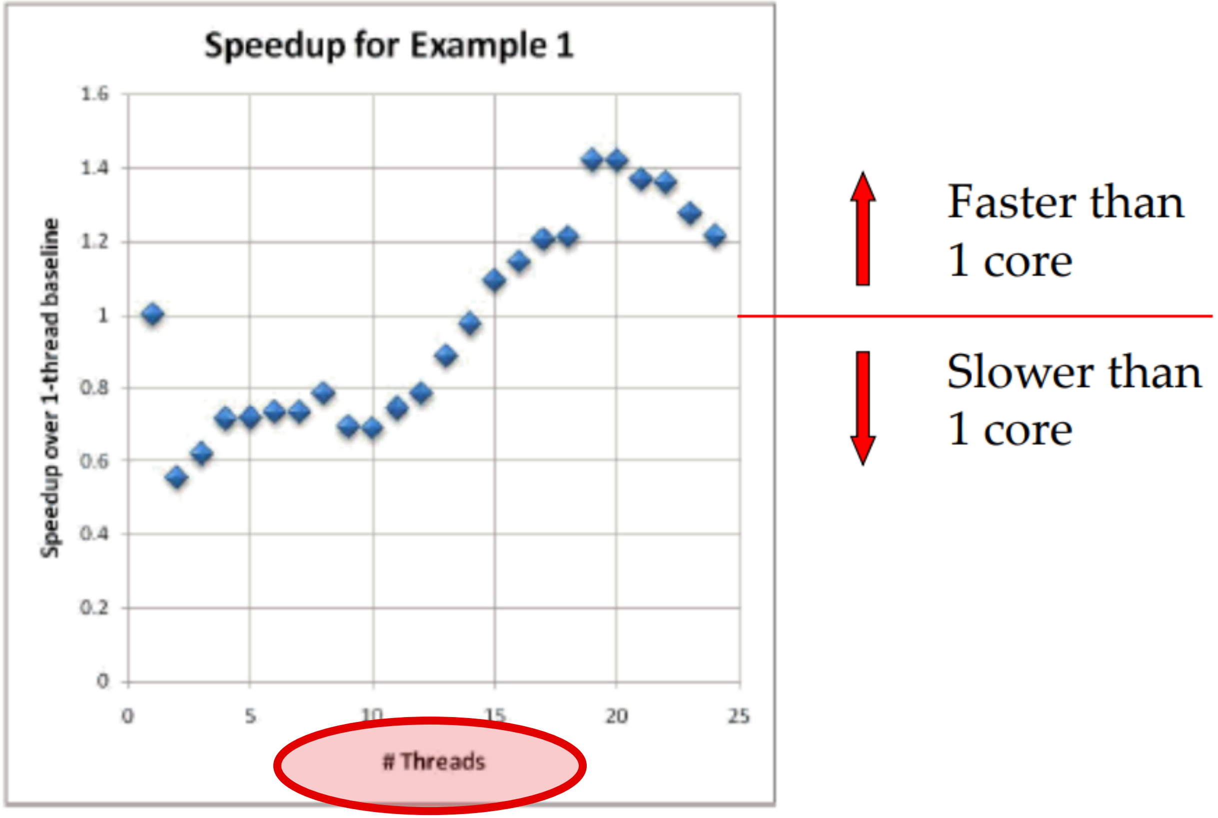
Source: "Performance comparison: linear search vs binary search." dirtyhandscoding.wordpress.com

Computer architecture effects example 3

```
int result[P];
// Each of P parallel workers processes 1/P-th of the data;
// the p-th worker records its partial count in result[p]
for (int p = 0; p < P; ++p )
  pool.run( [&,p] {
    result[p] = 0;
    int chunkSize = DIM/P + 1;
    int myStart = p * chunkSize;
    int myEnd = min( myStart+chunkSize, DIM );
    for( int i = myStart; i < myEnd; ++i )
      for( int j = 0; j < DIM; ++j )
        if( matrix[i*DIM + j] % 2 != 0 )
          ++result[p]; } );
pool.join();                    // Wait for all tasks to complete

odds = 0;                       // combine the results
for( int p = 0; p < P; ++p )
  odds += result[p];
```

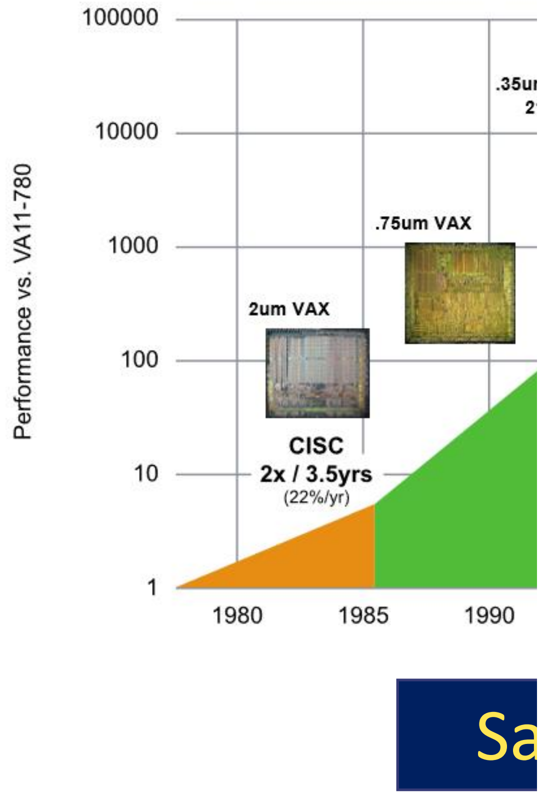


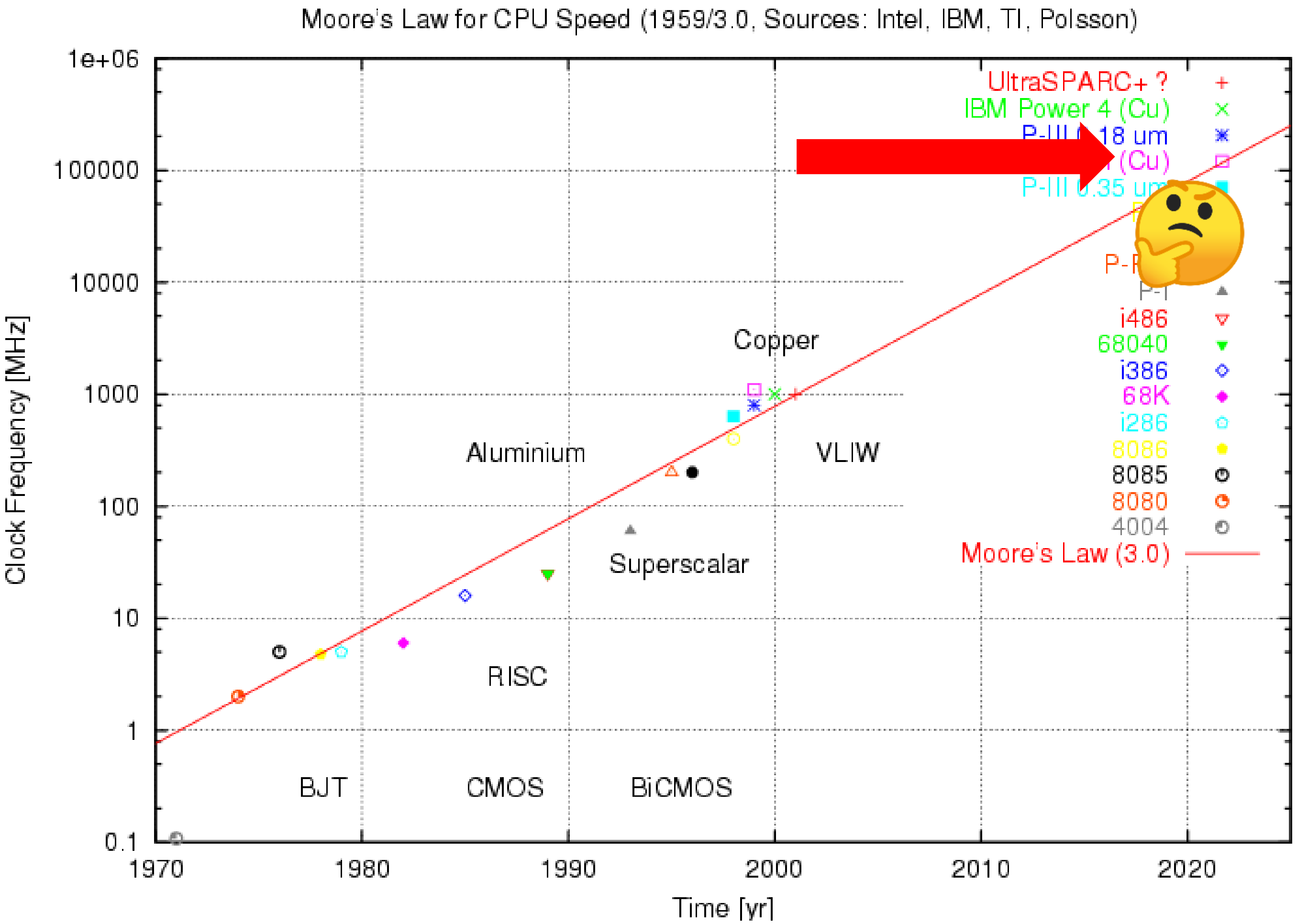


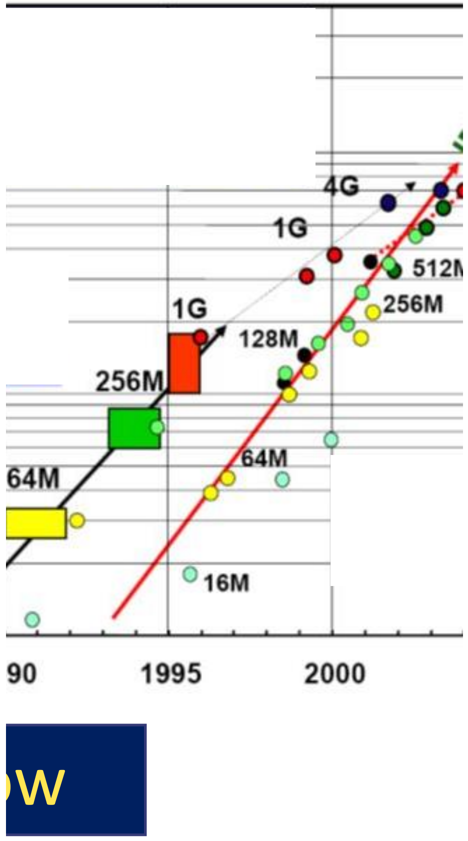
REALLY BAD scalability! Why?

Source: Scott Meyers, “CPU Caches and Why You care”

Why do we need computer architects?
-- The simpler past

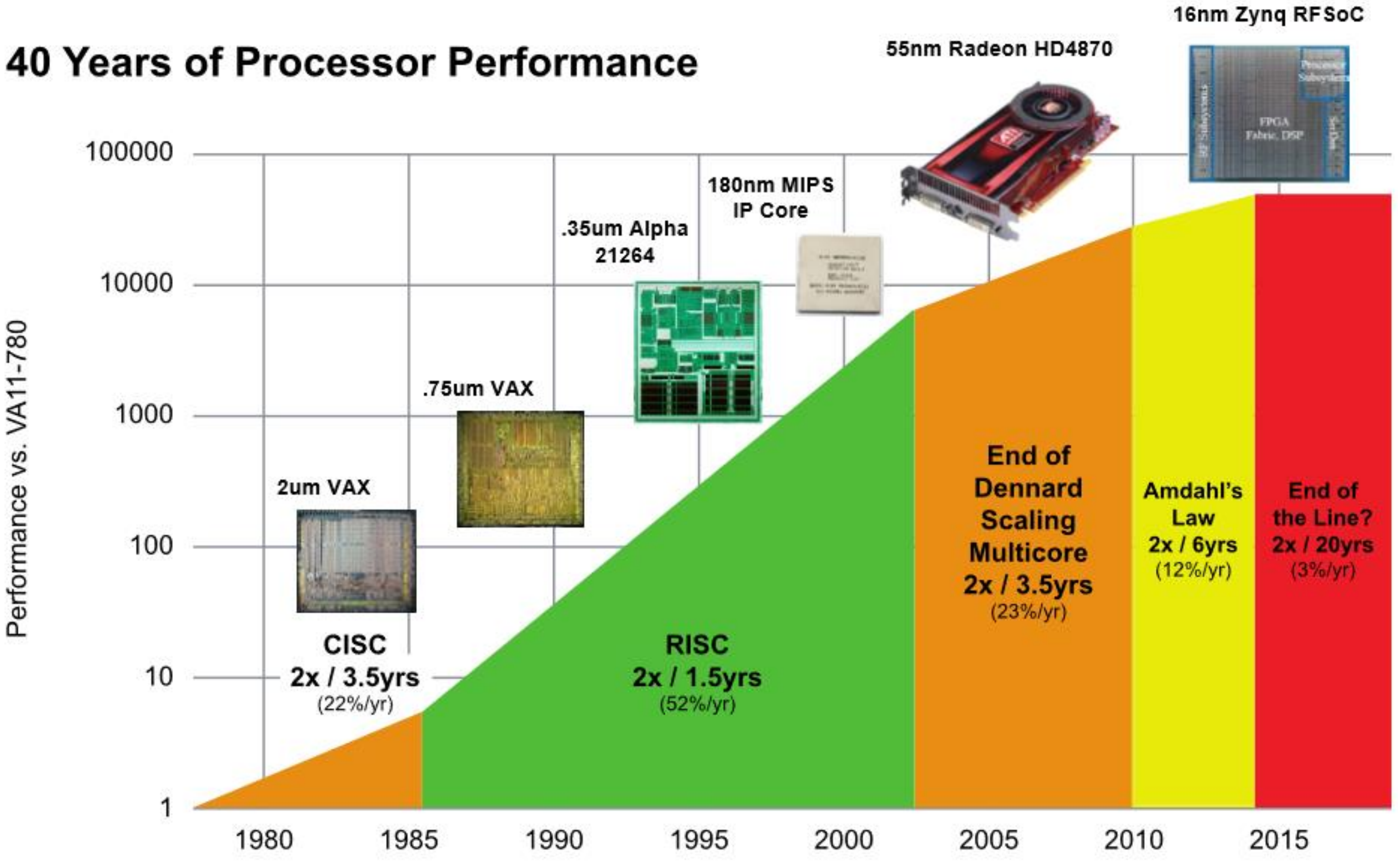






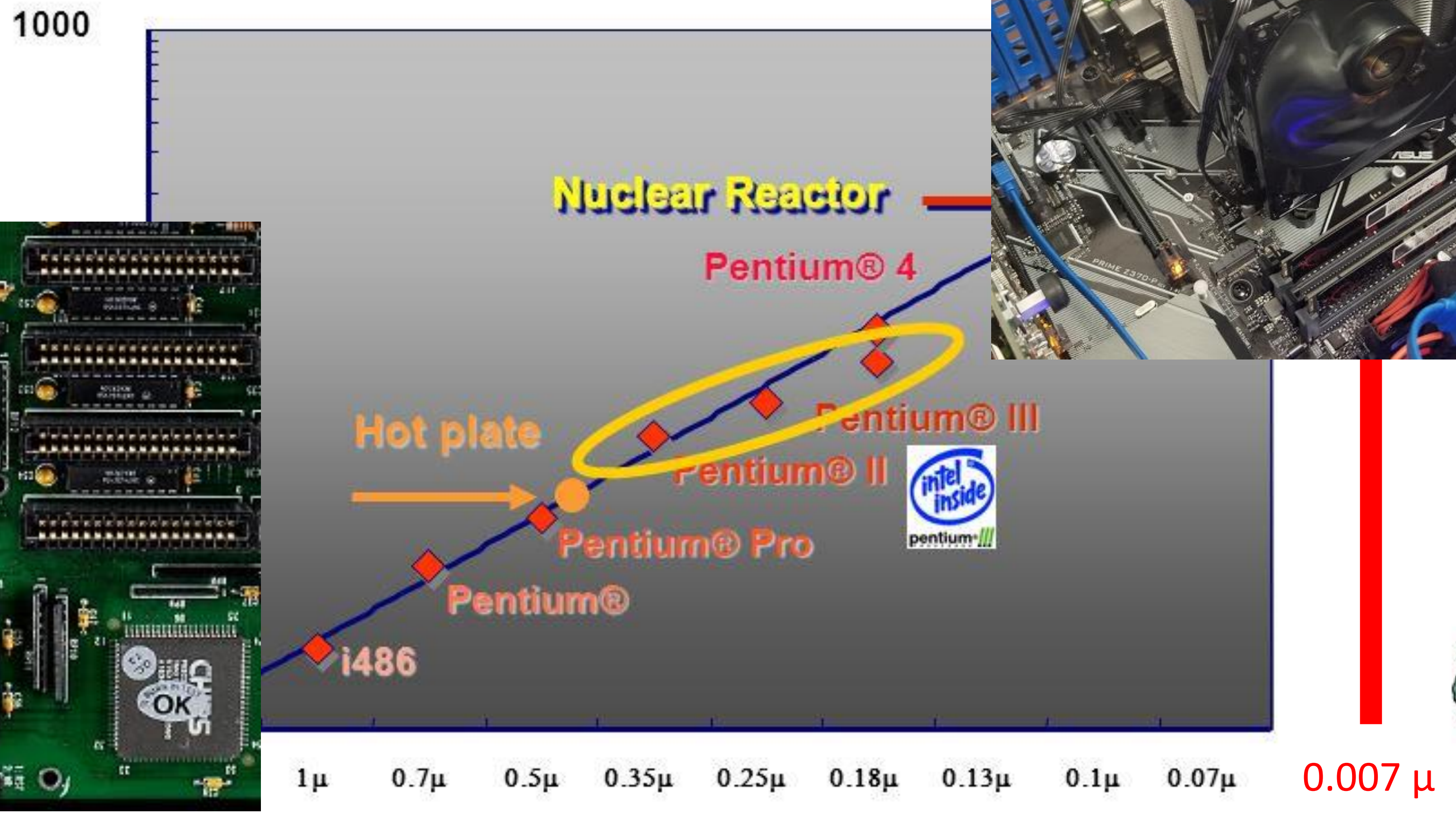
John Hennessy and David Patterson, “Computer Architecture: A Quantitative Approach”, 2018 (Cropped)
Bon-jae Koo, “Understanding of semiconductor memory architecture”, 2007 (Cropped)

Now: The end of Moore’s law and performance scaling



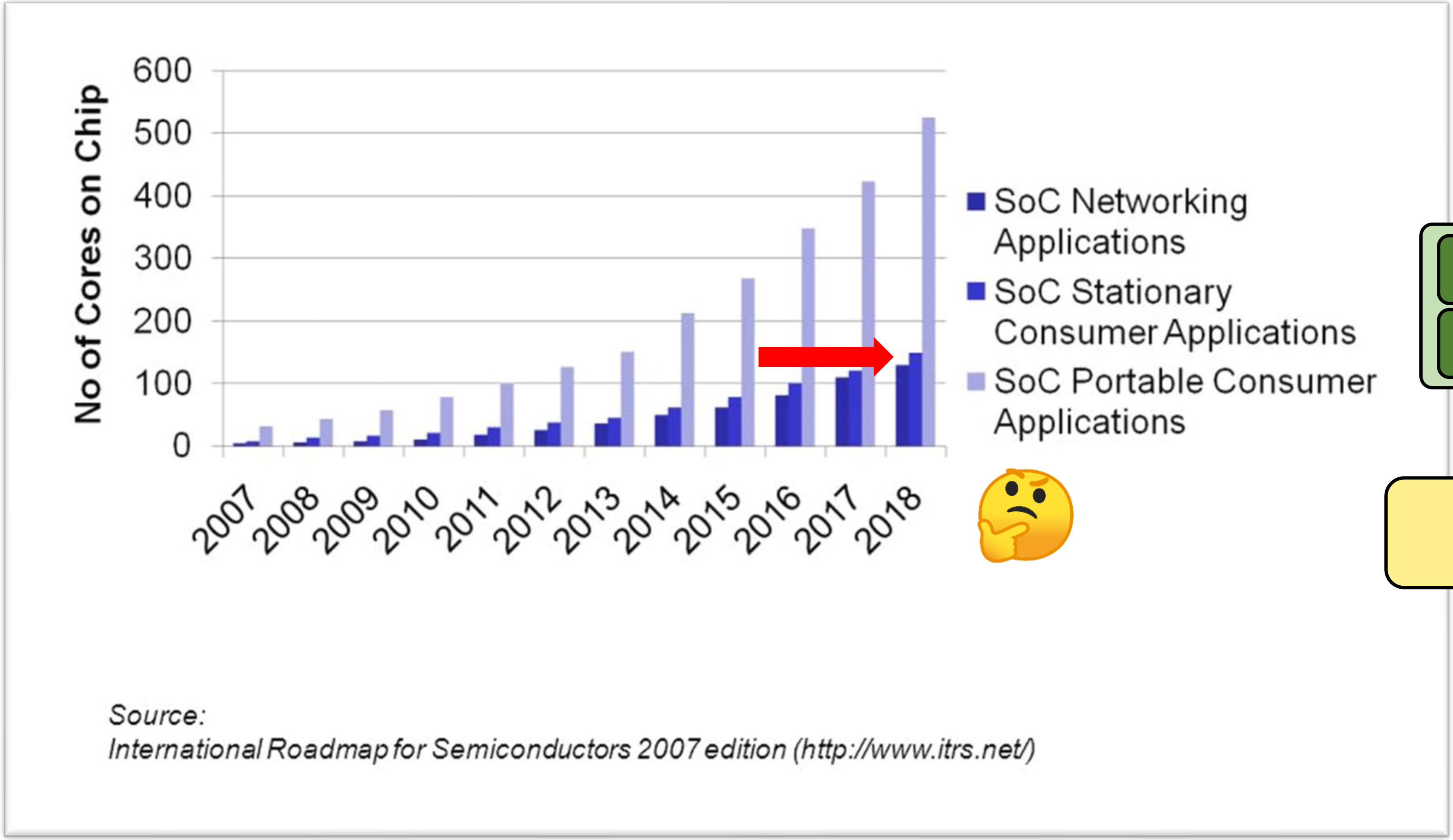
Source: John Hennessy and David Patterson, Computer Architecture: A Quantitative Approach, 6/e 2018

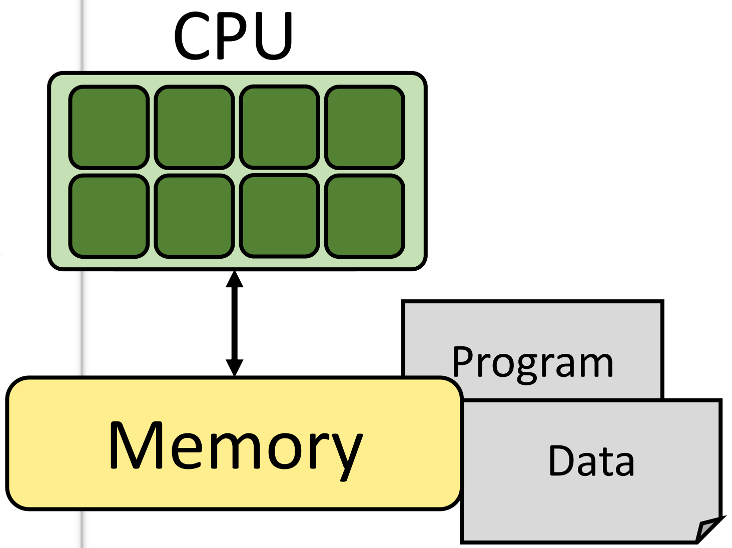
Running Into the Power Wall



* "New Microarchitecture Challenges in the Coming Generations of CMOS Process Technologies" – Fred Pollack, Intel Corp. Micro32 conference key note - 1999.

Crisis Averted With Manycores?





Bernd Hoefflinger, “ITRS 2028—International Roadmap of Semiconductors”, 2015

The Scale of Power Consumption

Department of Energy requests an exaflop machine by 2020

1,000,000,000,000,000,000 floating point operations per second

Using 2016 technology, 200 MW

Using 2019+ technology, 60 MW

Palo Verde Nuclear Generating Station

MIT Research nuclear reactor

3,937 MW

6 MW

Total residential power consumption of San Francisco: 168 MW

(Source: California Energy Commission 2018)

Lynn Freeny, Department of Energy

Also, scaling size is becoming more difficult!

- Processor fabrication technology has always reduced in size
  - As of end of 2020, 7 nm is cutting edge, working towards 5 nm

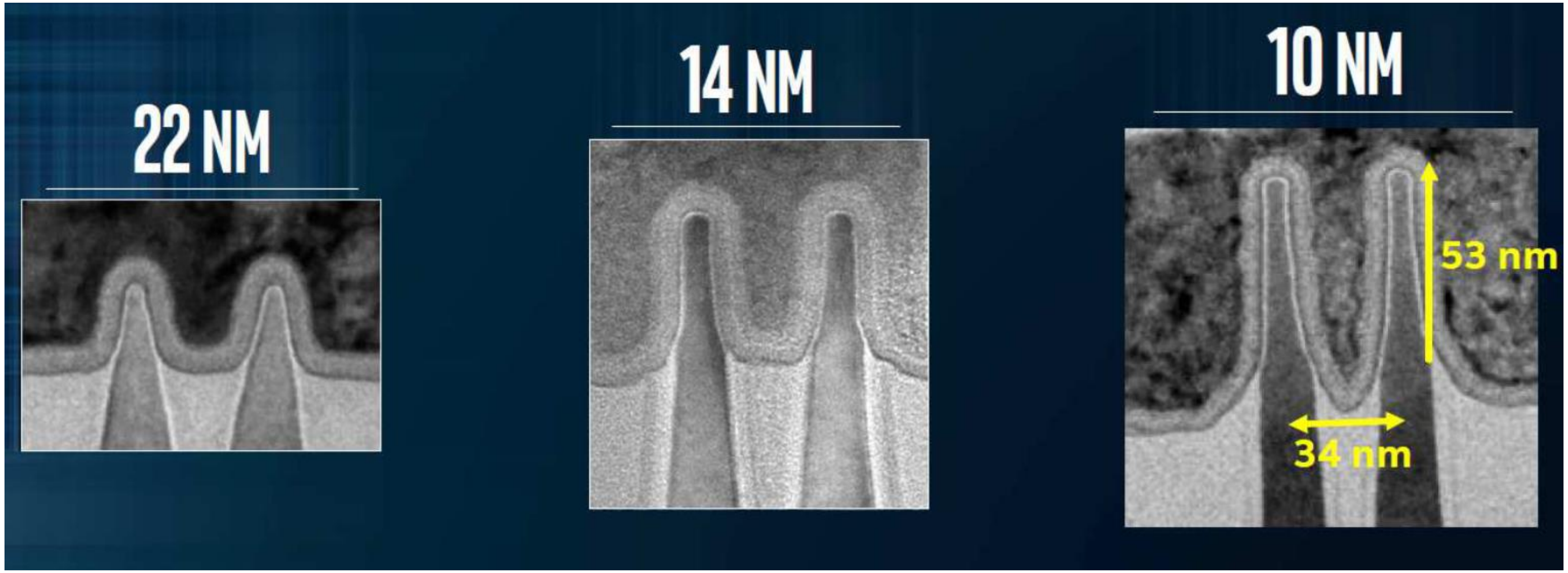


Image source: Intel

Number of Semiconductor Manufacturers with a Cutting Edge Logic Fab

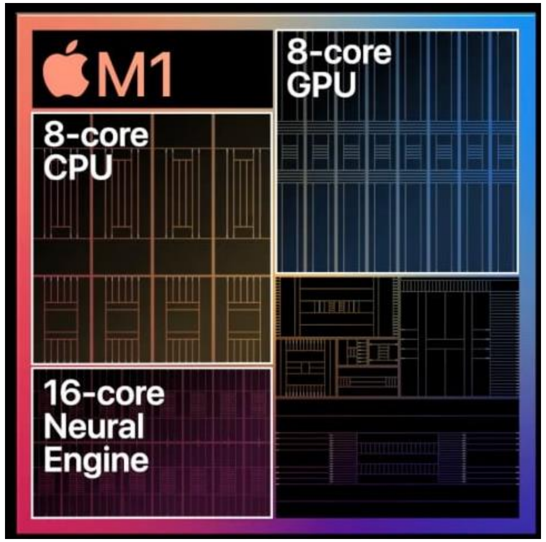
SilTerra										
X-FAB										
Dongbu HiTek										
ADI	ADI									
Atmel	Atmel									
Rohm	Rohm									
Sanyo	Sanyo									
Mitsubishi	Mitsubishi									
ON	ON									
Hitachi	Hitachi									
Cypress	Cypress	Cypress								
Sony	Sony	Sony								
Infineon	Infineon	Infineon								
Sharp	Sharp	Sharp								
Freescale	Freescale	Freescale								
Renesas (NEC)	Renesas	Renesas	Renesas	Renesas						
Toshiba	Toshiba	Toshiba	Toshiba	Toshiba						
Fujitsu	Fujitsu	Fujitsu	Fujitsu	Fujitsu						
TI	TI	TI	TI	TI						
Panasonic	Panasonic	Panasonic	Panasonic	Panasonic	Panasonic					
STMicroelectronics	STM	STM	STM	STM	STM					
HLMC	HLMC		HLMC	HLMC	HLMC					
UMC	UMC	UMC	UMC	UMC	UMC		UMC			
IBM	IBM	IBM	IBM	IBM	IBM	IBM				
SMIC	SMIC	SMIC	SMIC	SMIC	SMIC		SMIC			
AMD	AMD	AMD	GlobalFoundries	GF	GF	GF	GF			
Samsung	Samsung	Samsung	Samsung	Samsung	Samsung	Samsung	Samsung	Samsung	Samsung	Samsung
TSMC	TSMC	TSMC	TSMC	TSMC	TSMC	TSMC	TSMC	TSMC	TSMC	TSMC
Intel	Intel	Intel	Intel	Intel	Intel	Intel	Intel	Intel	Intel	Intel
180 nm	130 nm	90 nm	65 nm	45 nm/40 nm	32 nm/28 nm	22 nm/20 nm	16 nm/14 nm	10 nm	7 nm	5 nm

Only three players left?!

We can't keep doing what we used to

- Limited number of transistors, limited clock speed
  - How to make the ABSOLUTE BEST of these resources?

- Timely example: Apple M1 Processor
  - Claims to outperform everyone (per Apple)
  - How?
    - "8-wide decoder" [...] "16 execution units (per core)"
    - "(Estimated) 630-deep out-of-order"
    - "Unified memory architecture"
    - Hardware/software optimized for each other



What do these mean?

Not just apple! (Amazon, Microsoft, EU, …)

Image source: Apple

No better time to be an architect!

"There are Turing Awards waiting to be picked up
if people would just work on these things."
—David Patterson, 2018

And on that note…

Welcome to CS 152!

- We will learn:
  - How modern processors are designed to achieve high performance
  - under which restrictions, and
  - actually get hands-on experience with hardware design
  - using a sequence of gently guided labs.

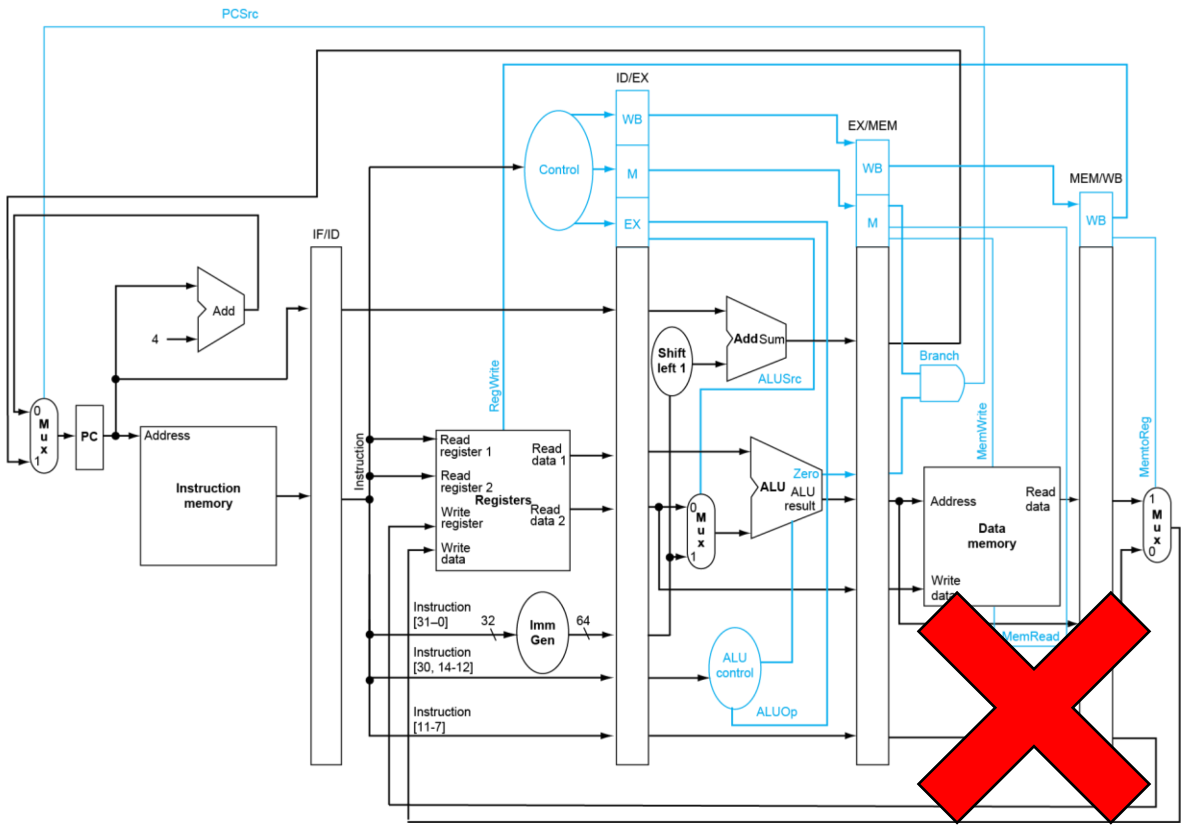
Course mechanics

- Lectures: Tuesdays, Thursdays at Zoom, 11:00 AM to 12:20 PM
- Recitations: Fridays at Zoom, 1:00 PM to 1:50 PM
  - May not always have lectures, but I will be there for questions

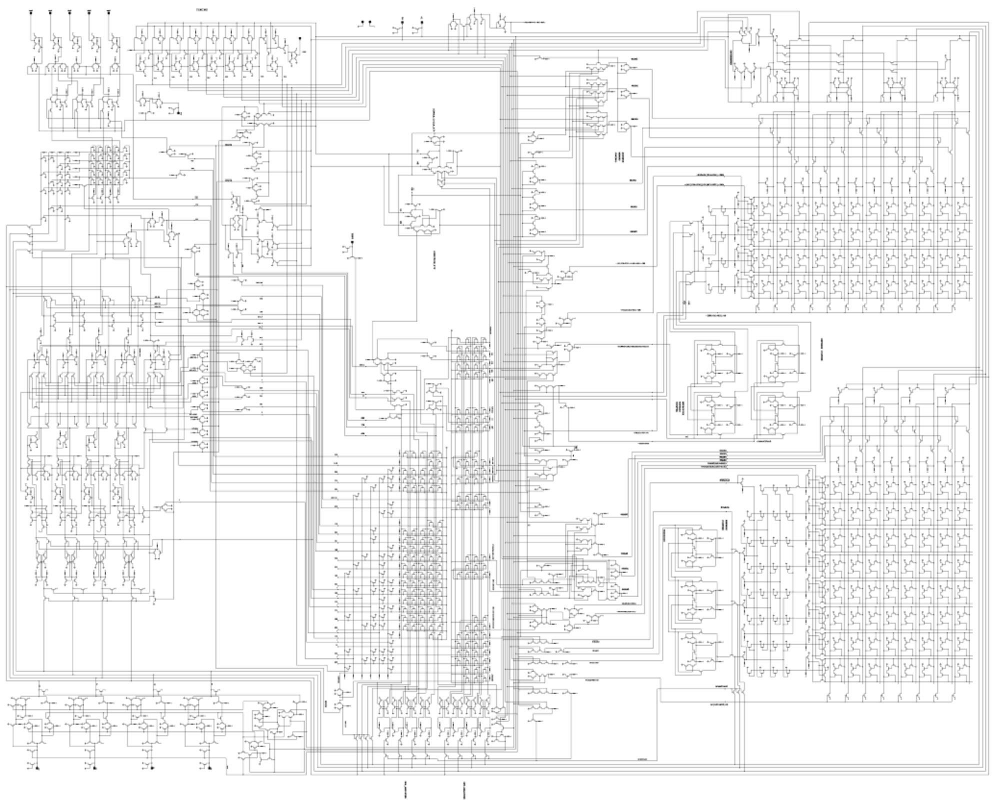
- Grading: 60% Labs, 40% Final, Curved
- Labs?
  - Will use a high-level hardware-description language (Bluespec)
  - By the end of the class, you will have a highly efficient CPU design that actually runs on metal! (FPGA)

What this class does and doesn't do

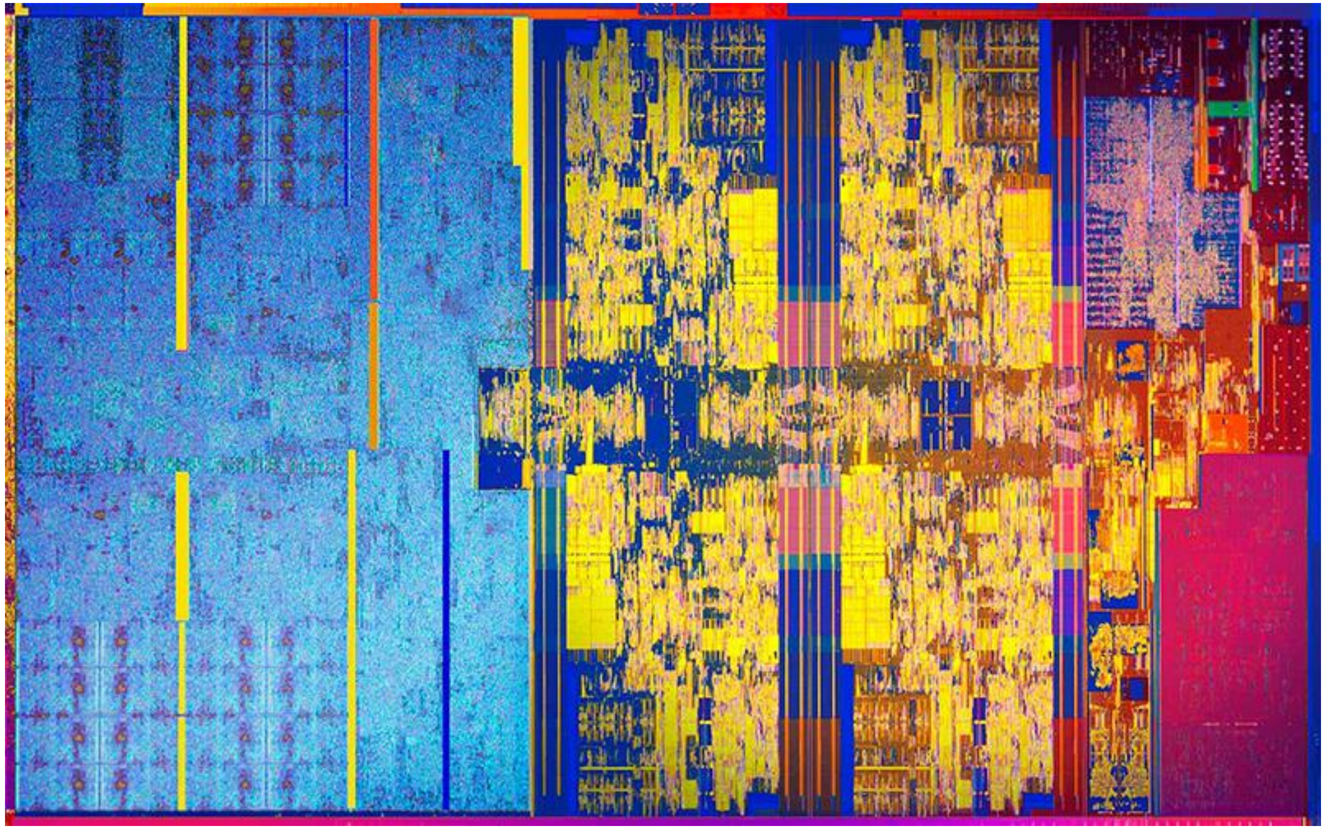
- ❑ It doesn't do
  - o Bit-level control signal management
    - (Not how modern processors are designed!)
  - o Details of the Intel x86 architecture
    - Very complicated and cluttered with backwards compatibility from the 70s
    - Interesting topic for after CS 152
- ❑ It does do
  - o Constructive computer architecture experience
    - Always working with a functional computer architecture design



Times have changed…



(1971) 2,250 transistors!
Intel 4004 Schematics drawn by Lajos Kintli and Fred Huettig for the Intel 4004 50th anniversary project



(2020) +1 Billion transistors!
Intel Core-i7 die (Source: Intel)

We will use modern tools

- RISC-V
  - Open-source Instruction-Set Architecture (ISA) based on what was learned in the past decades
- Bluespec
  - A high-level hardware-description language

Some important ideas in computer architecture

- ❑ Pipelining
- ❑ Caches and their design
- ❑ Branch prediction
- ❑ Virtual memory and privileges

Covered in CS 152

Covered in CS 250 and beyond

- ❑ Superscalar
- ❑ Simultaneous multithreading
- ❑ Speculative execution
- ❑ Out-of-Order Execution
- ❑ Vector operations
- ❑ Accelerators

and more!

Course outline

- Part 1: The Hardware-Software Interface
  - What is a ‘good’ processor?
  - Assembly programming and conventions
- Part 2: Recap of digital design
  - Combinational and sequential circuits
  - How their restrictions influence processor design
- Part 3: Computer Architecture
  - Computer Arithmetic
  - Simple and pipelined processors
  - Caches and the memory hierarchy
- Part 4: Computer Systems
  - Operating systems, Virtual memory