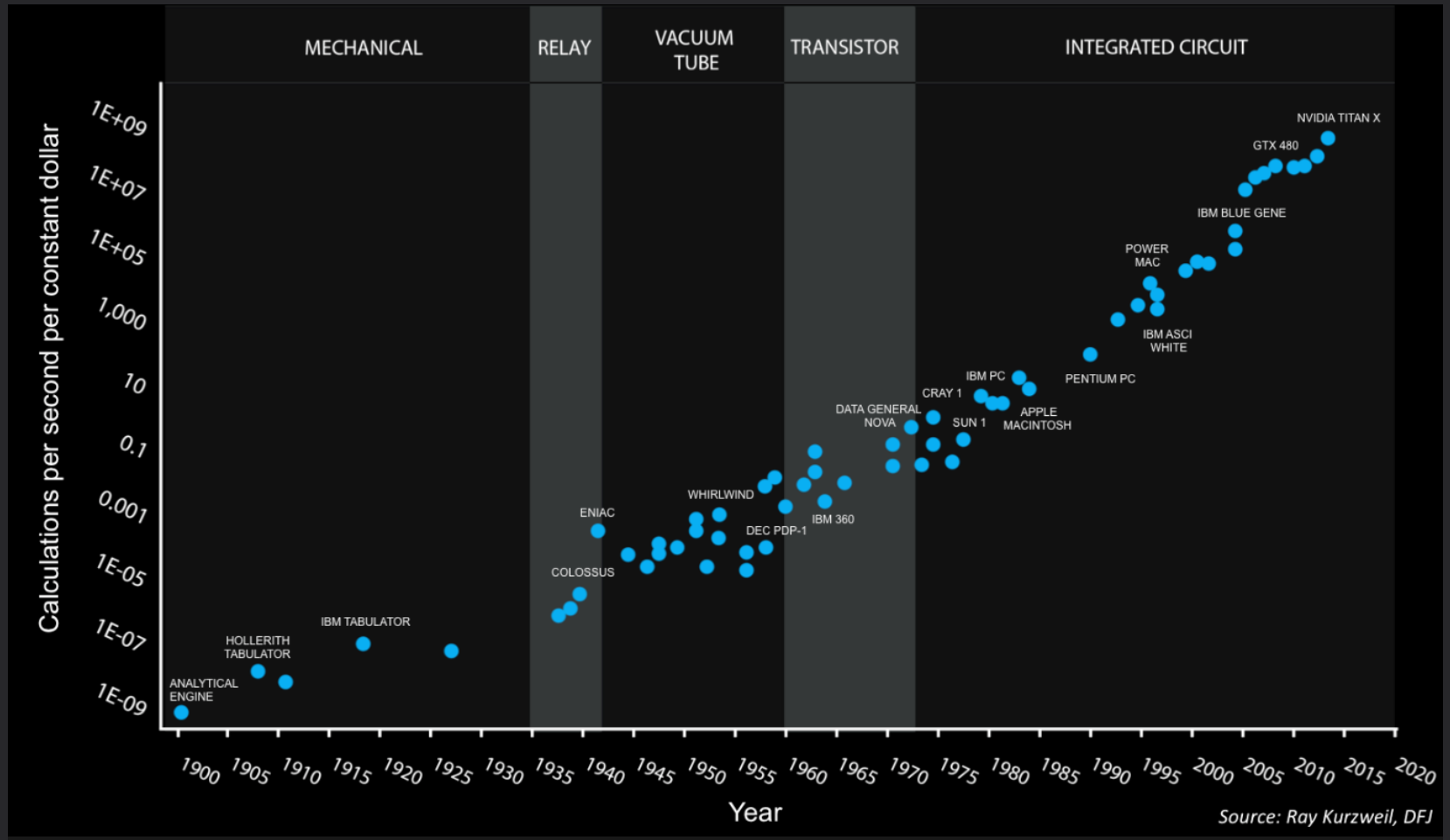
Computer Architecture

An introduction

The trend

- Exponential growth in performance per dollar
- How was this possible?
- Why did it happen?



By Steve Jurvetson. CC BY 2.0

The impact

New classes of computers

Desktop

Laptop

Smartphone

Easier programming

Less focus on performance

More focus on productivity

Ubiquity

Billions of computers worldwide

Zettabytes of data processed every year

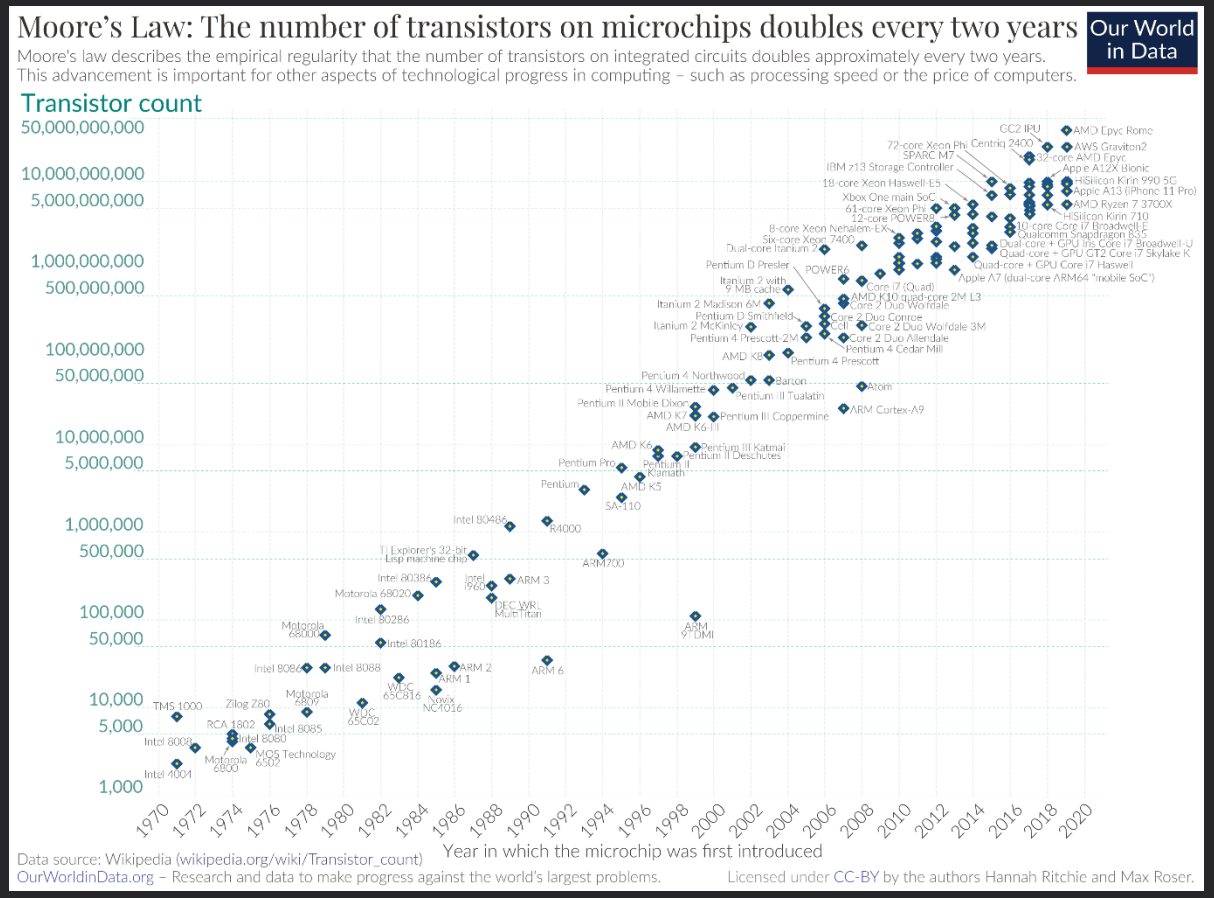
A cornucopia of transistors

Transistors, transistors, everywhere.

Moore’s Law

The number of transistors in an integrated circuit doubles every two years

- The minimum size of a transistor is its *feature size* (20 µm in 1968, 3 nm in 2022)
- Decreasing feature size would:
  - Increase the number of transistors
  - Improve the performance of a transistor
- What to do with all these transistors?
  - Computer architecture!
- P.S. Moore’s “law” is over



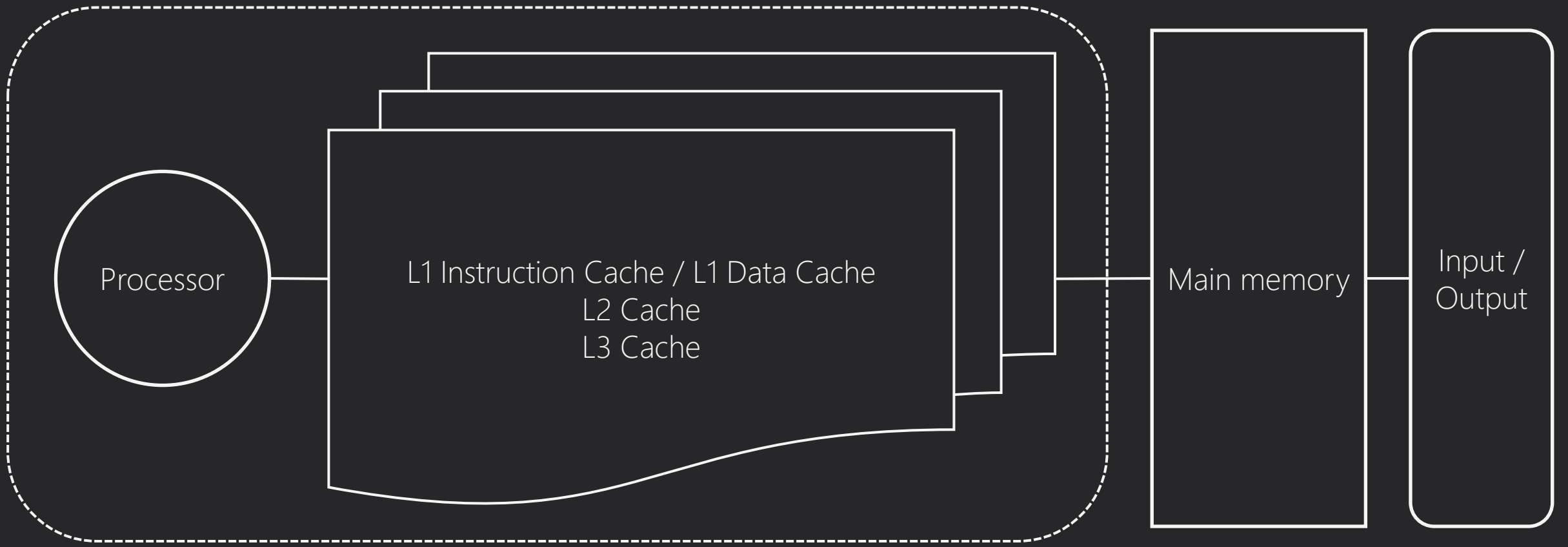
Instruction-level parallelism (ILP)

- To some degree, instructions may be evaluated in parallel
- ILP is done implicitly; no programmer intervention required!

Technique for exploiting ILP	Topic	Week
Pipelining (including forwarding and hazards)	Pipelined processors	2
Multiple-issue processors	Instruction scheduling	4
Dynamic instruction scheduling	Instruction scheduling	4
Branch prediction	Speculation	5

The hardware cache

Lots of transistors – why not store *more* data on-chip? Learn more in Week 3.



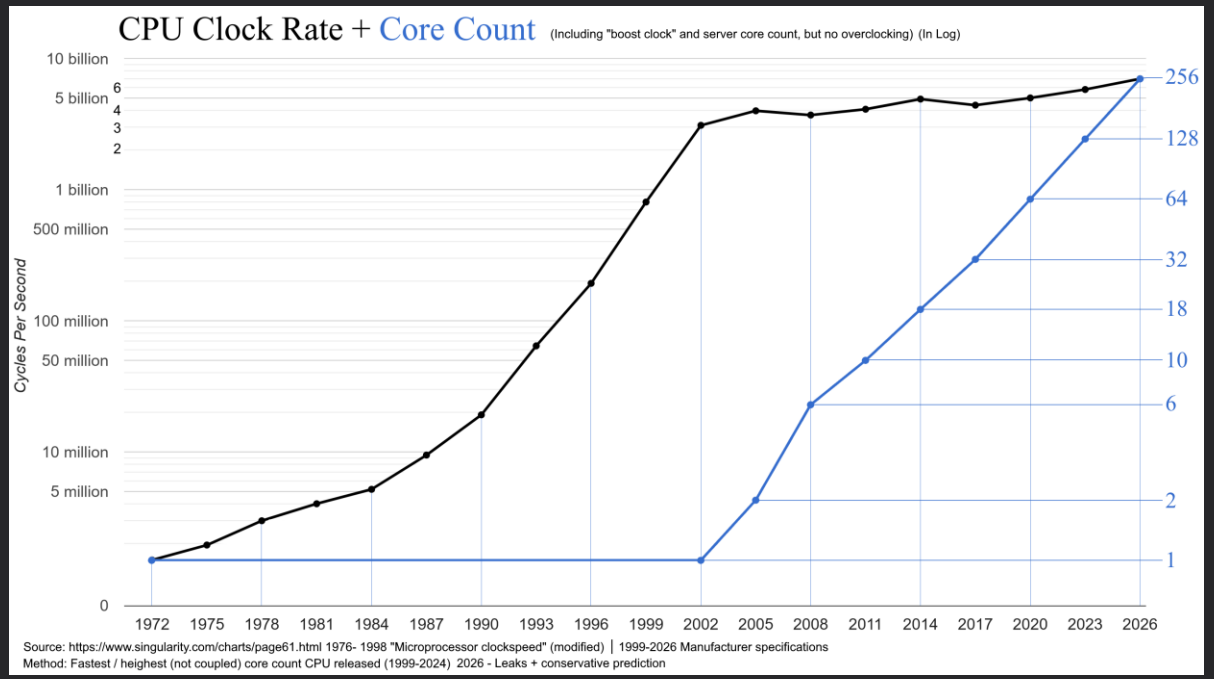
The power wall

When physical limitations ruin the party

Dennard scaling

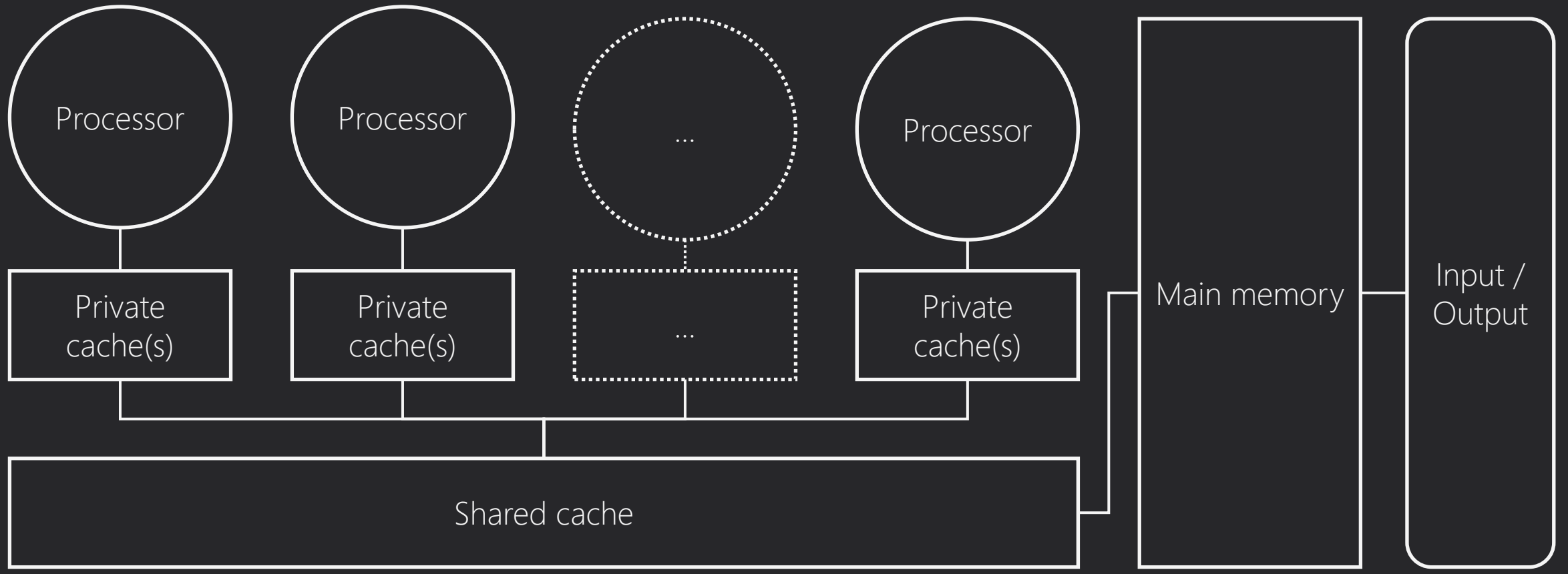
Power density stays constant, even as transistors get smaller.

- Dynamic energy consumed by a transistor when it switches: $E_{dyn} = \frac{1}{2}CV^2$
- Power is the product of energy and frequency: $P_{dyn} = \frac{1}{2}CV^2 \times f$
- Dennard scaling allowed increasing frequency
  - Why? Smaller transistors need less voltage
  - This is a performance gain
- Then, we hit the "power wall"
  - Can't drop voltage more without degrading reliability
  - Enter (stage right): the multi-core era



Thread-level parallelism (TLP)

- Threads can be executed in parallel. But programmers must be explicit about what each thread does
- Weeks 6 to 9



Amdahl's Law

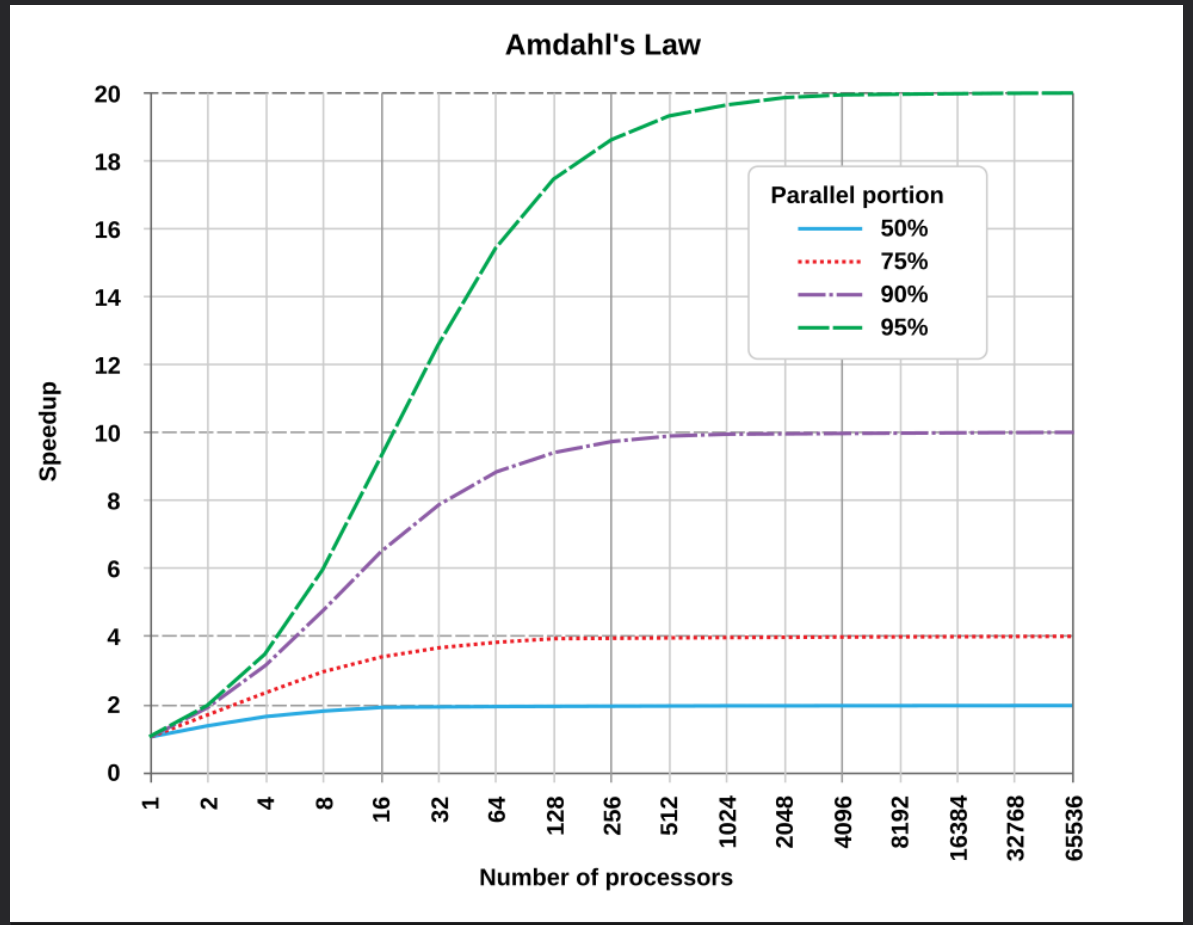
The speedup from enhancing one part of a system is limited by the fraction of time the improved part is used.

- Let $Speedup = \frac{Time_o}{Time_e}$
  - o is original
  - e is enhanced

- Amdahl's law: Enhance fraction of computation (f) by some speedup (S):

$$Speedup_e(f, S) = \frac{1}{(1 - f) + {}^{f}/_{S}}$$

- Implications of Amdahl's law
  - Small f means enhancement has little effect
  - Even with very large S, speedup is bounded by $\frac{1}{1-f}$



By Daniels220, CC BY-SA 3.0

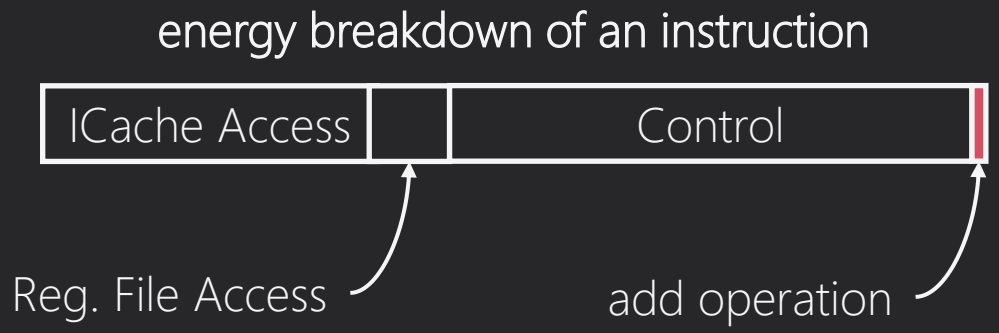
From general-purpose to specialized

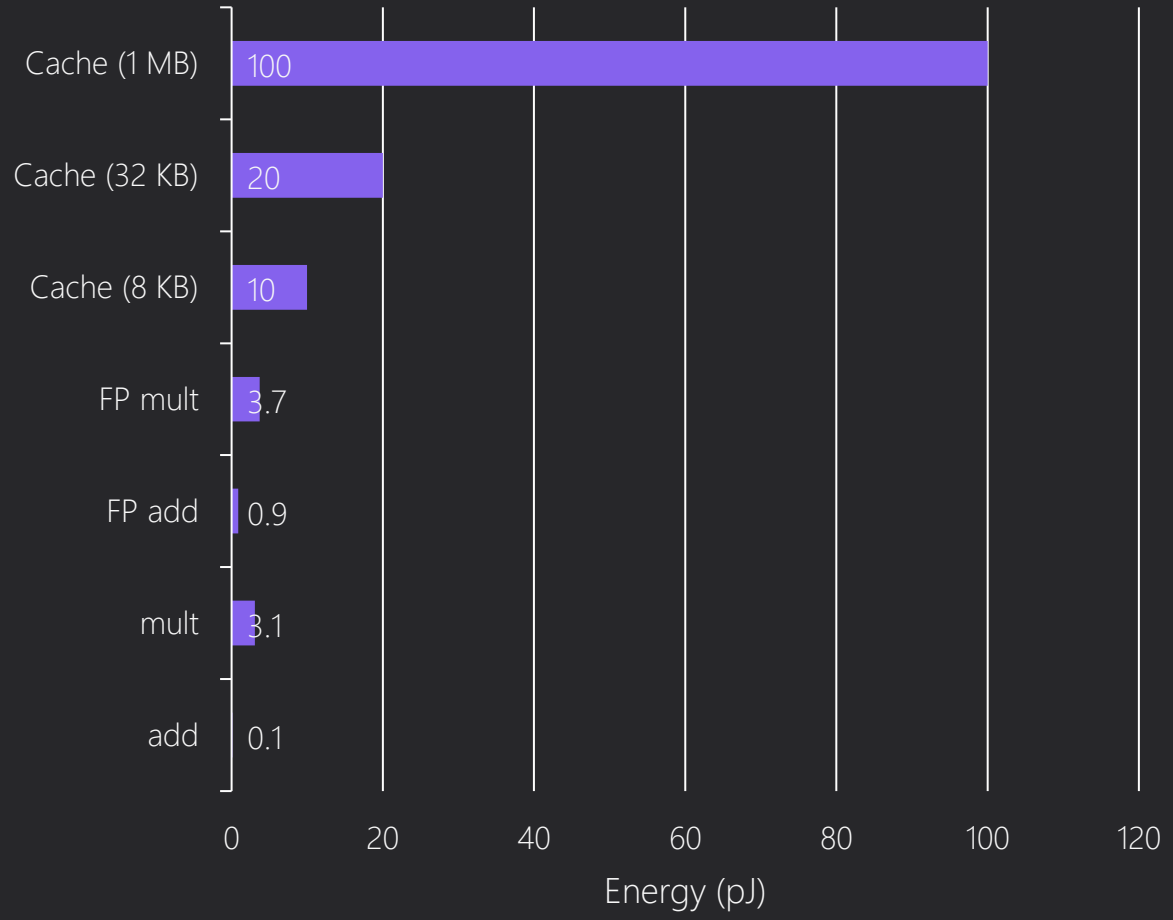
When power and energy budgets dominate

A power-limited world

Figures adapted from: Horowitz, Mark. "1.1 computing's energy problem (and what we can do about it)." 2014 IEEE international solid-state circuits conference digest of technical papers (ISSCC). IEEE, 2014.

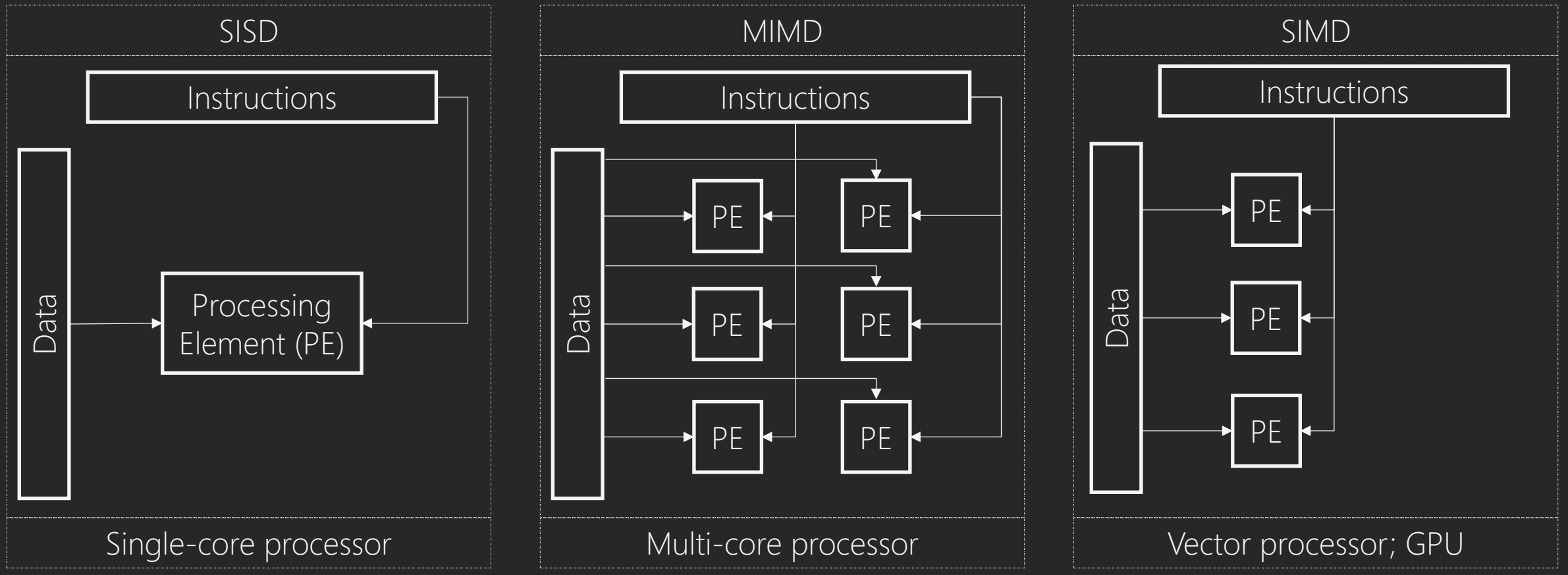
- Instructions have high energy overhead
- How can architects do better?





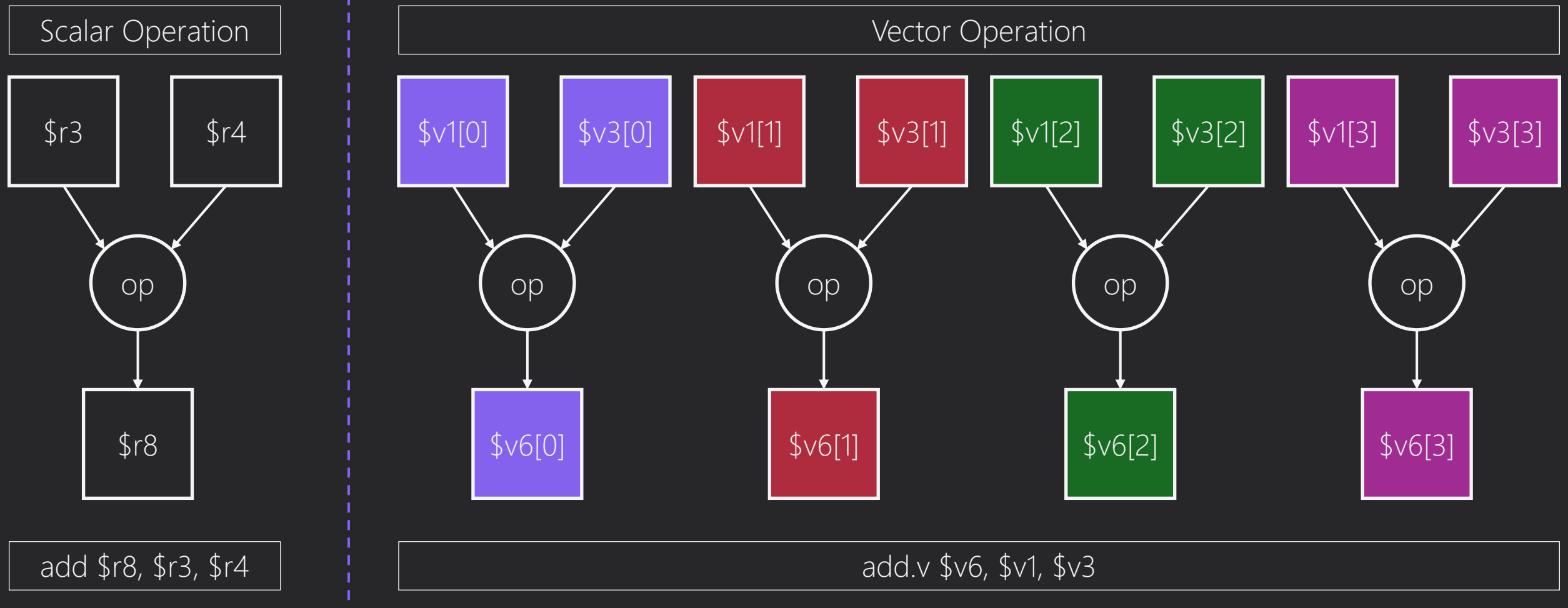
Flynn's Taxonomy

- Single instruction, single data (SISD)
- Multiple instruction, multiple data (MIMD)
- Single instruction, multiple data (SIMD)



Data-level parallelism

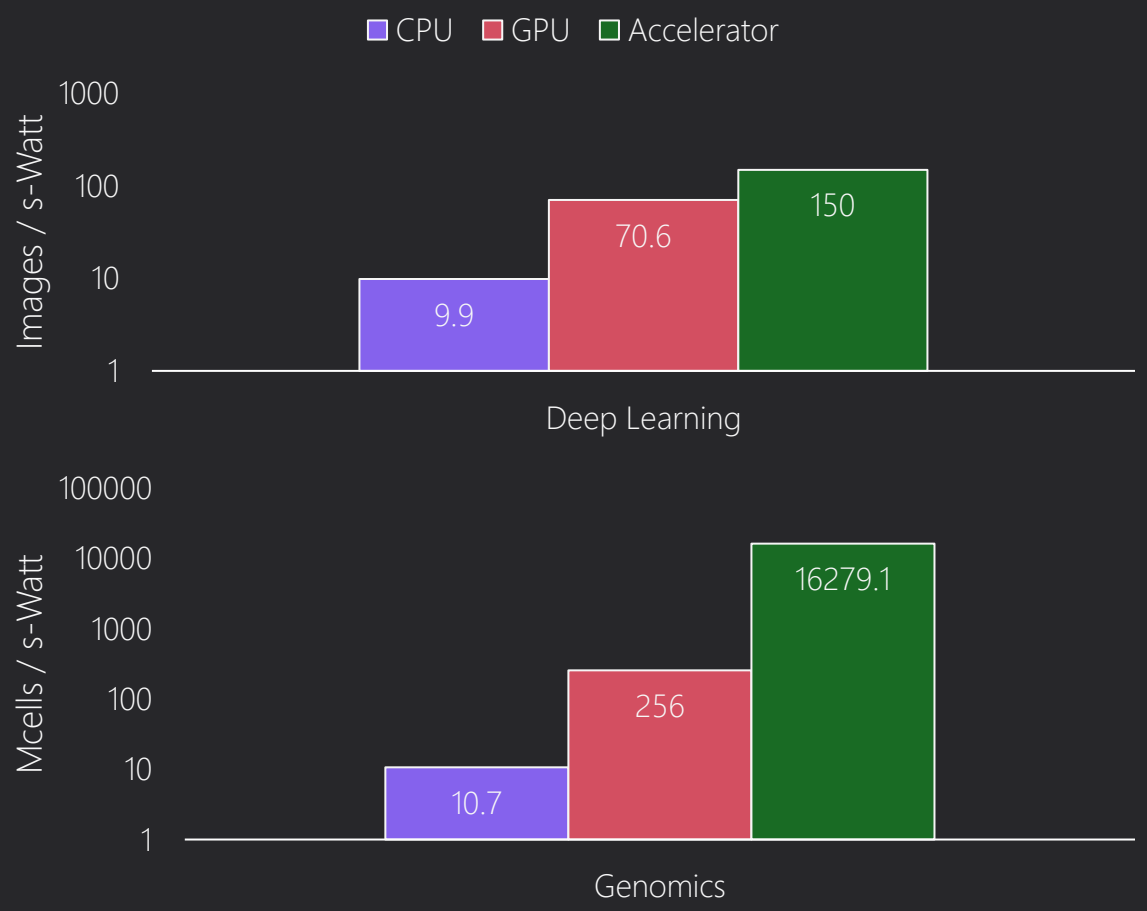
- Distribute the data, but perform the same operation(s)
- Week 10



Specialization

- Improve energy efficiency and speedup by creating hardware for specific algorithms
- Domain-specific accelerators are *not* general-purpose
  - They do one thing well
  - May be programmable with, e.g., specialized instructions
- Week 11

Charts adapted from: Dally, William J., Yatish Turakhia, and Song Han. "Domain-specific hardware accelerators." Communications of the ACM 63.7 (2020): 48-57.

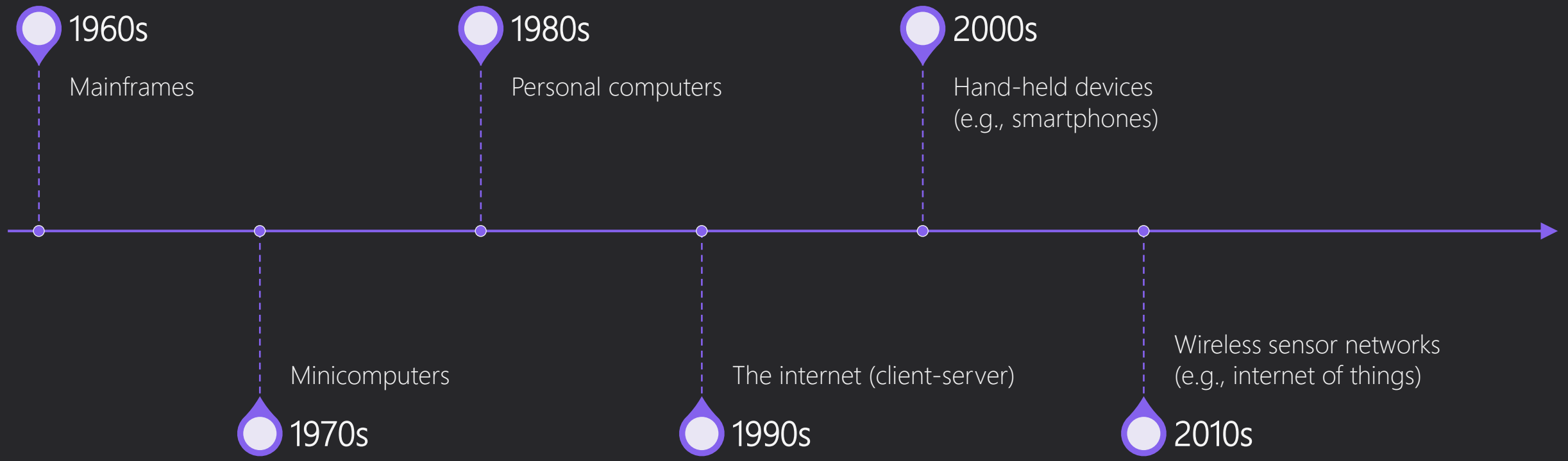


Conclusion

Recapping the important points

Bell's Law

"A new computer class forms and approximately doubles each decade, establishing a new industry" – Gordon Bell



Computer architects explore trade-offs

- How can I use transistors to increase performance?
  - ILP techniques and caches improve performance at the cost of power and energy
- How can I use transistors to be performant within some power budget?
  - TLP can do it, but requires programmer effort
  - DLP can do it, but requires programmer effort
  - Accelerators can do it, but developing hardware is harder than software
- Other trade-offs?

Computer architects use many tools

- Deriving “laws” from trends in empirical data
  - Moore’s Law, Dennard Scaling, Bell’s Law
- Building (simple) models and taxonomies to understand complex systems
  - Amdahl’s Law, Flynn’s Taxonomy
- Evaluating designs before (and after) they are built
  - Simulation, complex “calculators”, measurement

Computer architects design systems

- Given the technology available…
  - Like types of memory and disks, new packaging techniques
- … design a system that meets specific goals….
  - Like performance, energy, cost ($), security
- … for the (emerging) application domains in the market…
  - Like gaming, smart phones, embedded systems, scientific computing
- … in X months!
  - Time-to-market matters; competition is rampant