Computer Architecture / Security / Quantum / Where to Go Next

Yipeng Huang

Rutgers University

April 29, 2021

Very important to help develop next iteration of this course.

- https://sirs.ctaar.rutgers.edu/blue

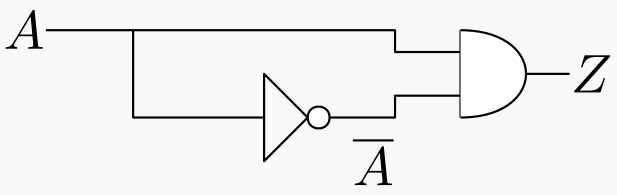
Computer Architecture

- ***<u>Microarchitecture support for Instruction Set Architectures</u>***
- Moore’s Law
- Data Level Parallelism
- Instruction Level Parallelism
- Hardware Security

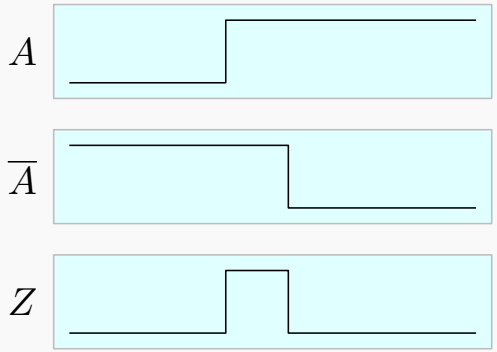
Synchronization. Clock signals.

Timing

Circuit:

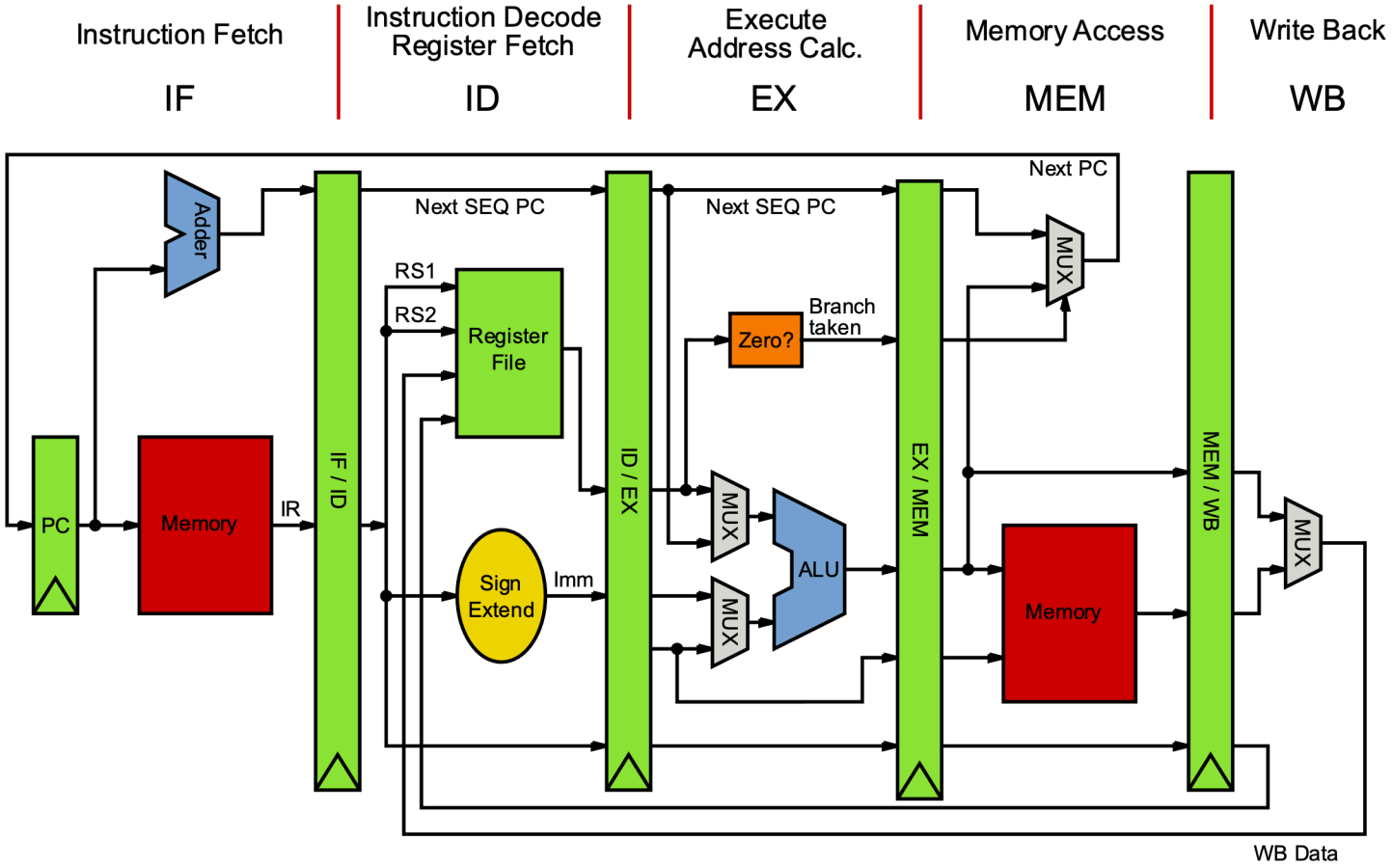


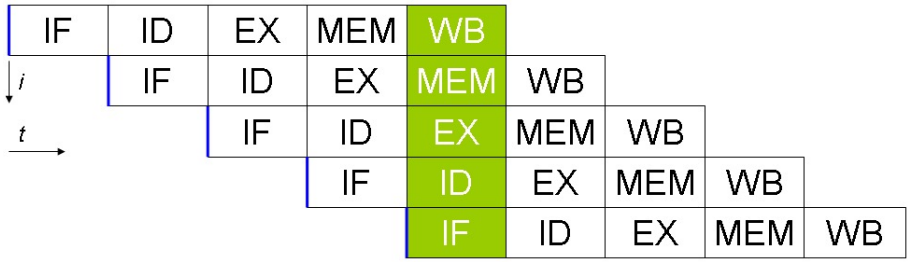
Voltages over time:



(A AND NOT A) should always evaluate to FALSE, need a clock signal to denote when values are valid, and to ignore transients.

The journey of an assembly instruction: pipeline





An assembly instruction needs multiple stages of combinational and sequential logic to execute.

At any given moment, several assembly instructions are in-flight.

Image sources: Wikimedia.

Microarchitectural support for Instruction Set Architectures

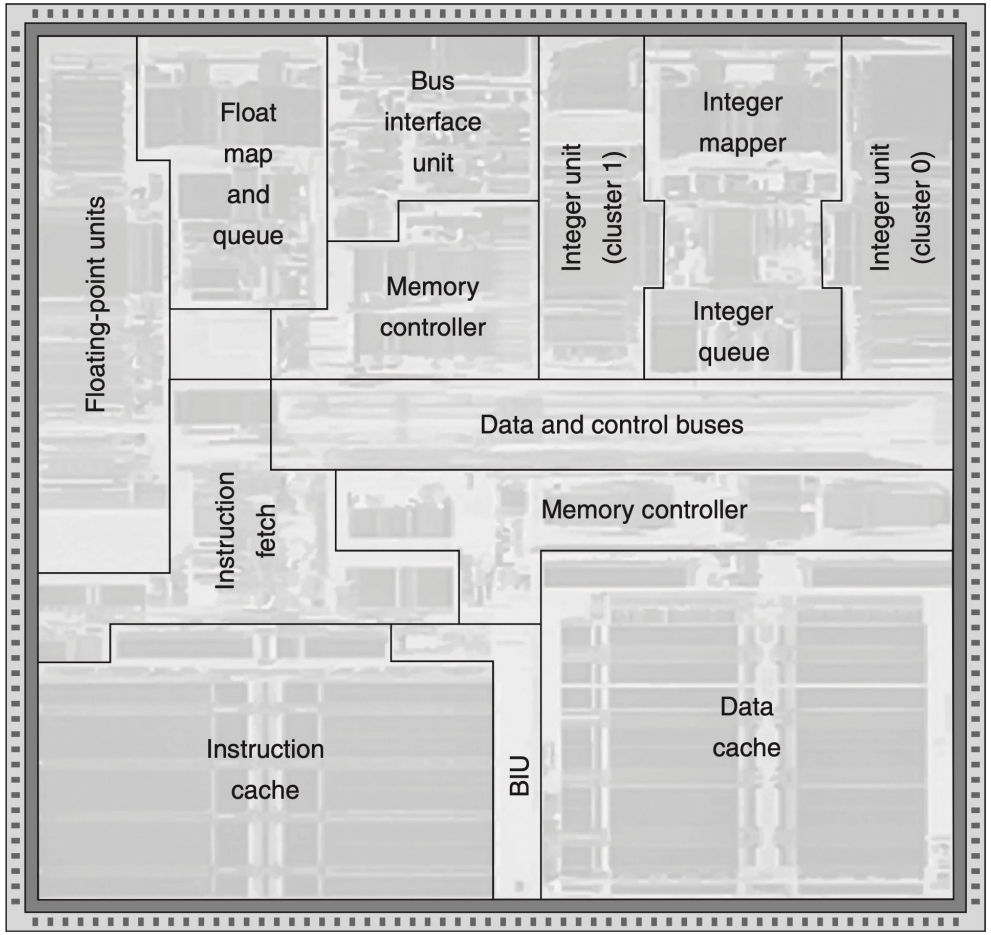


Figure 2.33 Floorplan of the Alpha 21264 [Kessler 1999].

Hennessy & Patterson. Computer Architecture: A Quantitative Approach

Computer Architecture

- Microarchitecture support for Instruction Set Architectures
- ***Moore's Law***
- Data Level Parallelism
- Instruction Level Parallelism
- Hardware Security

Dennard Scaling, Moore's Scaling, Power Wall

For a few decades, shrinking transistors drove clock speeds higher.

Dennard Scaling: Shrinking transistors also use less power.

Win-win.

Circa 2005, Dennard Scaling hit physical limitations.

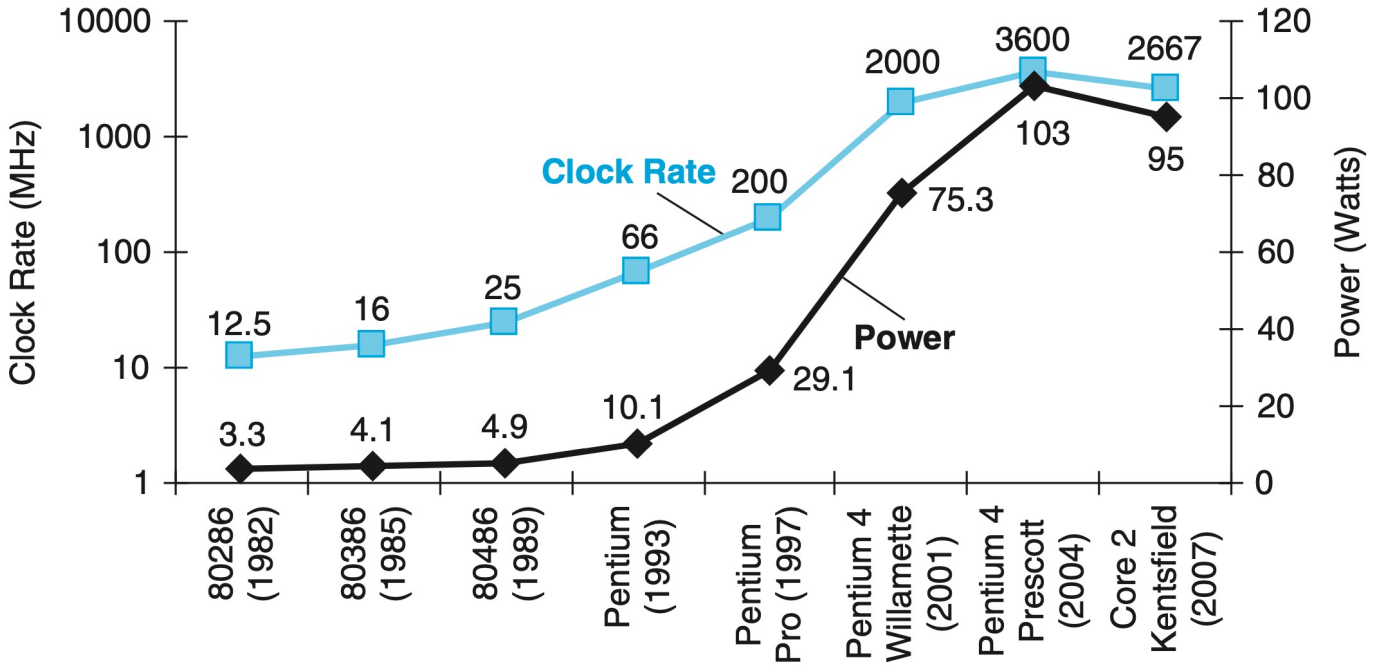


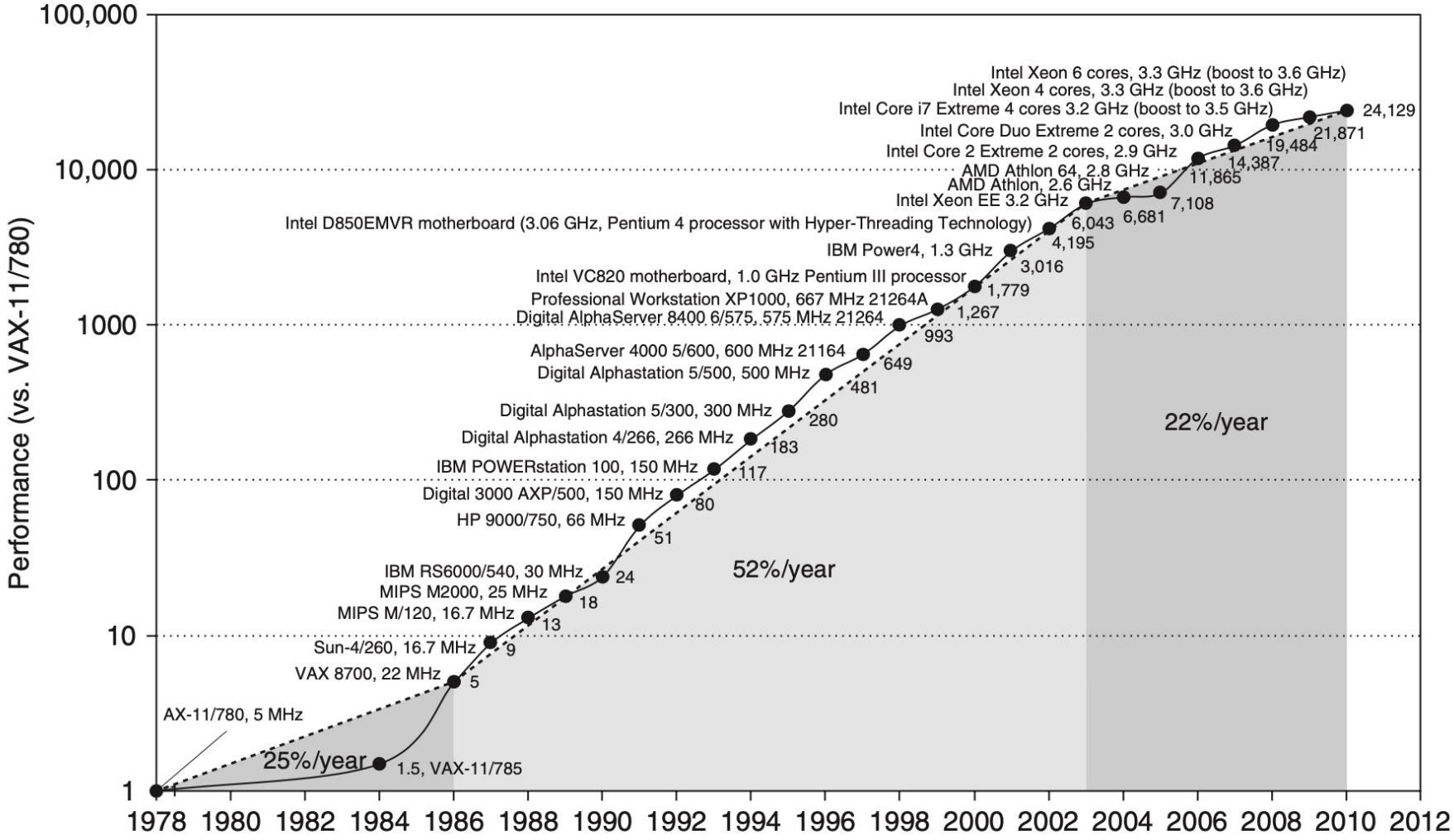
FIGURE 1.15 Clock rate and Power for Intel x86 microprocessors over eight generations and 25 years. The Pentium 4 made a dramatic jump in clock rate and power but less so in performance. The Prescott thermal problems led to the abandonment of the Pentium 4 line. The Core 2 line reverts to a simpler pipeline with lower clock rates and multiple processors per chip.

Patterson & Hennessy. Computer Organization and Design: The Hardware/Software Interface.

Computer Architecture

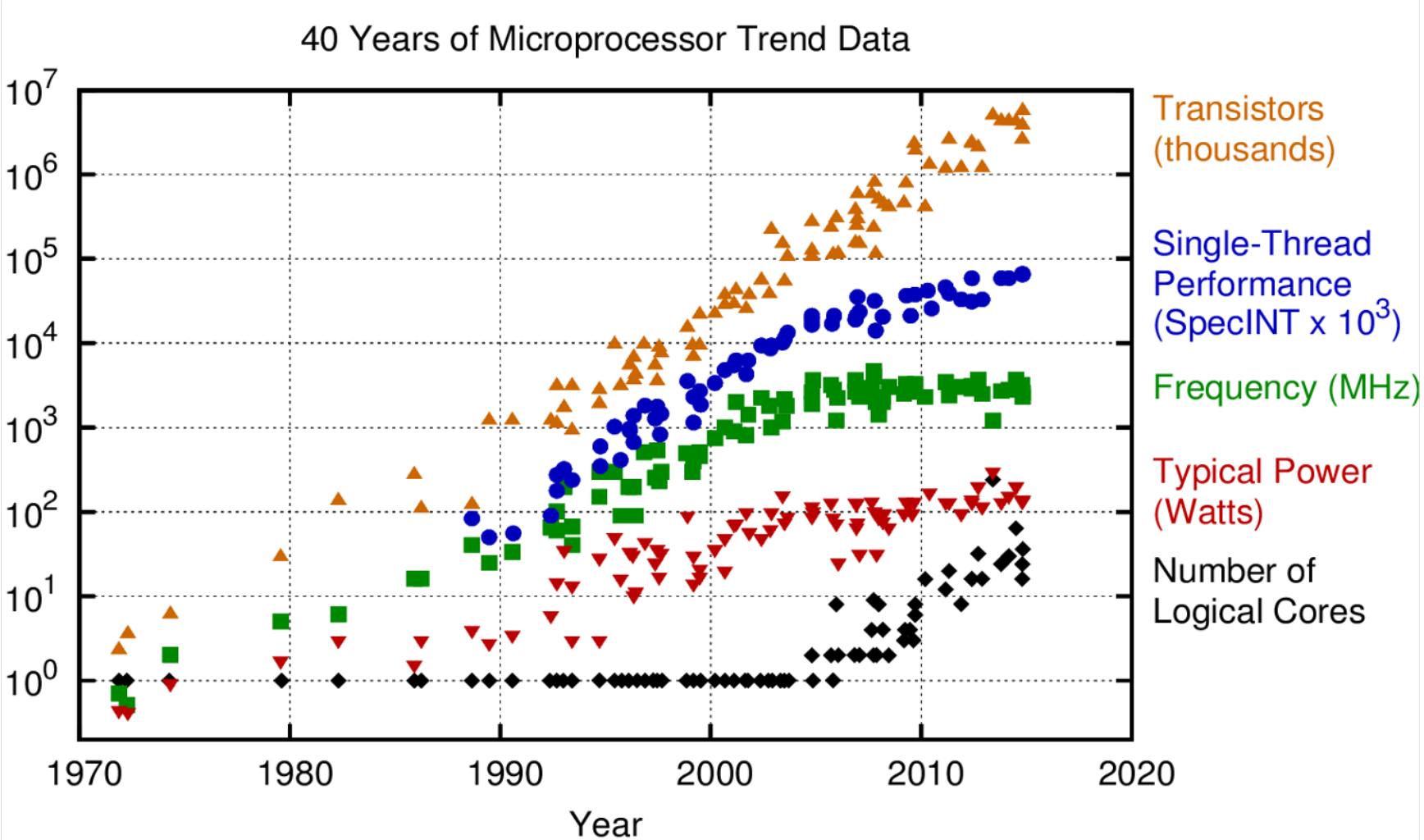
- Microarchitecture support for Instruction Set Architectures
- Moore’s Law
- ***<u>Data Level Parallelism</u>***
- Instruction Level Parallelism
- Hardware Security

Drivers of CPU performance: scaling and architecture



Hennessy & Patterson. Computer Architecture: A Quantitative Approach

Data Level Parallelism



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 2010-2015 by K. Rupp

Data Level Parallelism

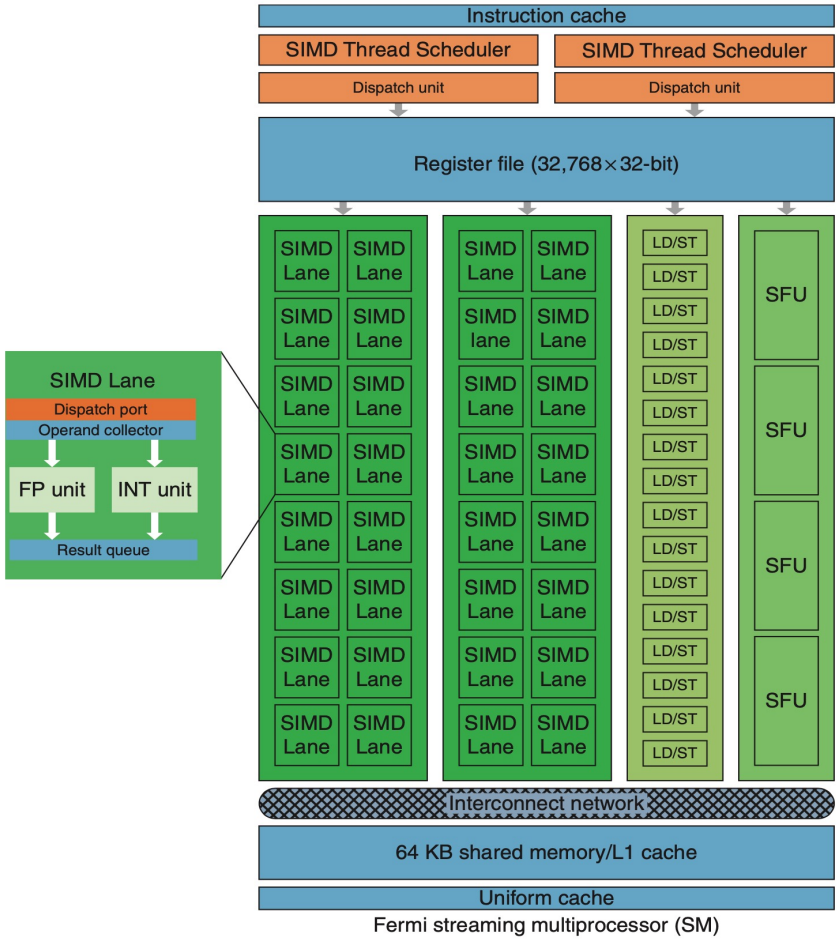


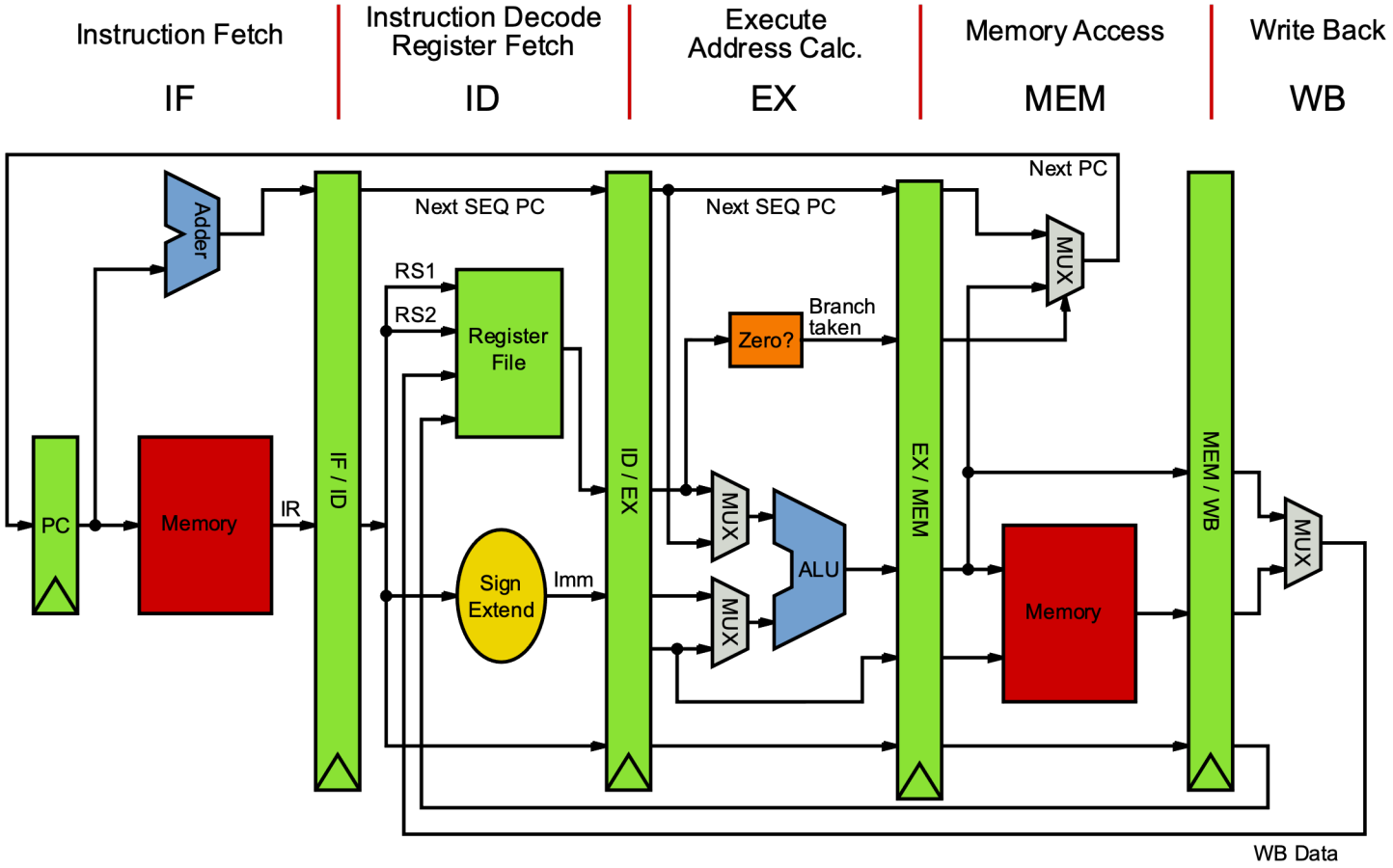
Figure 4.20 Block diagram of the multithreaded SIMD Processor of a Fermi GPU. Each SIMD Lane has a pipelined floating-point unit, a pipelined integer unit, some logic for dispatching instructions and operands to these units, and a queue for holding results. The four Special Function units (SFUs) calculate functions such as square roots, reciprocals, sines, and cosines.

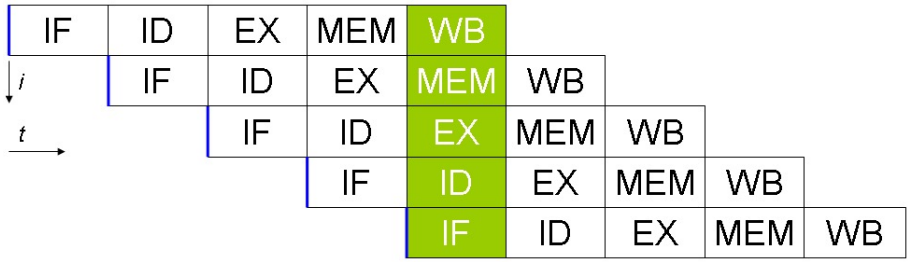
Hennessy & Patterson. Computer Architecture: A Quantitative Approach

Computer Architecture

- Microarchitecture support for Instruction Set Architectures
- Moore’s Law
- Data Level Parallelism
- ***<u>Instruction Level Parallelism</u>***
- Hardware Security

The journey of an assembly instruction: pipeline





An assembly instruction needs multiple stages of combinational and sequential logic to execute.

At any given moment, several assembly instructions are in-flight.

Image sources: Wikimedia.

Instruction Level Parallelism

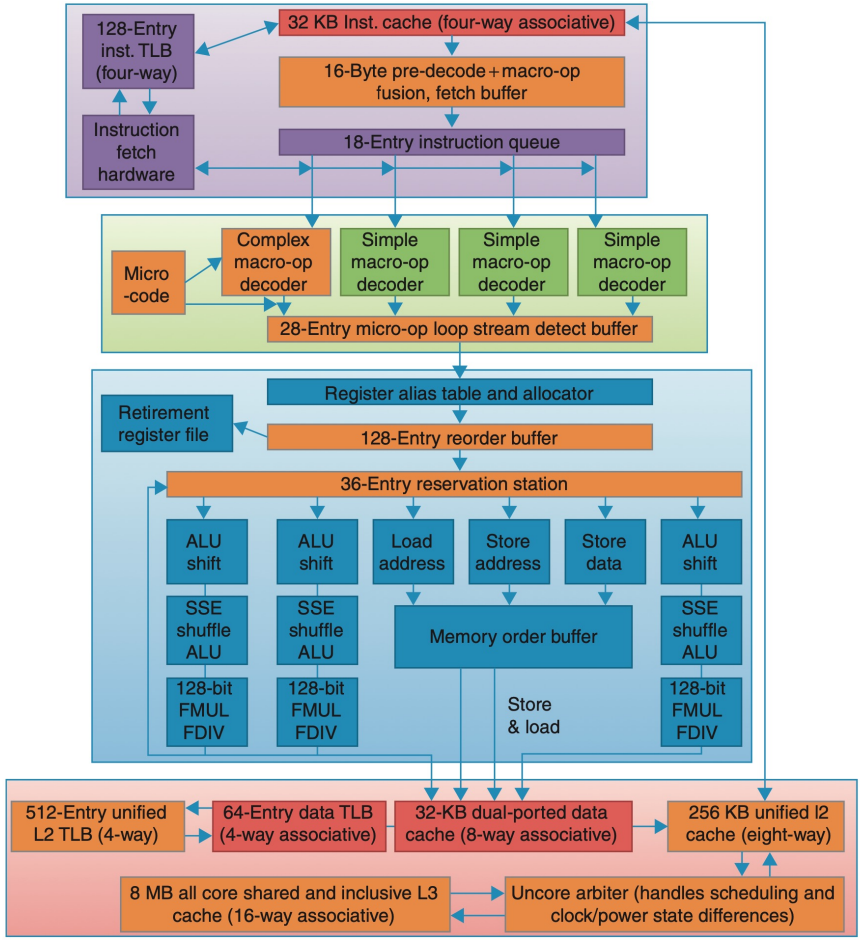


Figure 3.41 The Intel Core i7 pipeline structure shown with the memory system components. The total pipeline depth is 14 stages, with branch mispredictions costing 17 cycles. There are 48 load and 32 store buffers. The six independent functional units can each begin execution of a ready micro-op in the same cycle.

Hennessy & Patterson. Computer Architecture: A Quantitative Approach

Computer Architecture

- Microarchitecture support for Instruction Set Architectures
- Moore’s Law
- Data Level Parallelism
- Instruction Level Parallelism
- ***<u>Hardware Security</u>***

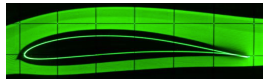
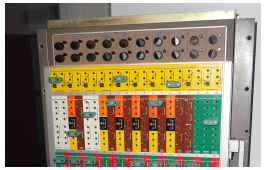
Limitations of microarchitecture trickery: security vulnarabilities

- See Prof. Mark Hill (U. Wisconsin) slides on Meltdown and Spectre

- Lipp, Moritz, et al. "Meltdown: Reading kernel memory from user space." *27th {USENIX} Security Symposium ({USENIX} Security 18)*. 2018.

Heterogenous architectures

- Accelerators
- New paradigms: Quantum computing

	1940s	1950s	1960s	1970s	1980s	1990s	2000s	2010s
Analog continuous-time computing	Analog computers for rocket and artillery controllers.	Analog computers for field problems. 	1st transistorized analog computer. 	Analog-digital hybrid computers.	…			
Digital discrete-time computing	Turing's *Bomba*.	1st transistorized digital computer.	Moore's law projection for transistor scaling.	Dennard's scaling for transistor power density.	VLSI democratized.	FPGAs introduced.	End of Dennard's scaling.	Cloud FPGAs: Microsoft Catapult. Amazon F1.
	Stored program computer.	Microprogram ming.	Instruction set architecture.	Reduced instruction set computers.		GPUs introduced.	CPUs go multicore.	
							Nvidia introduces CUDA.	ASICs: Google TPUs. DE Shaw Research Anton.

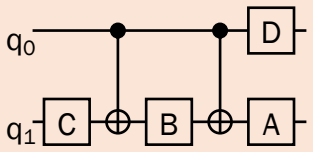
Heterogenous architectures

Transistor scaling and architectural abstractions drive digital revolution, make analog alternatives irrelevant

Scaling challenges drive heterogenous architectures

Image sources: AnalogMuseum.org, ComputerHistory.org

Quantum chemistry
& high energy physics

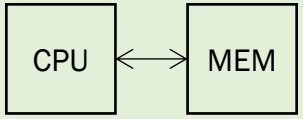


Quantum circuit computers

Digital
(discrete variables)

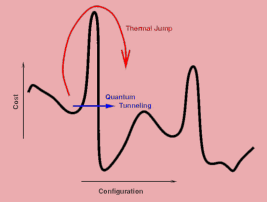
Quantum

General purpose
computation



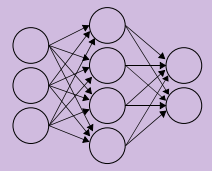
Classical digital
von Neuman architectures

High dimensional
nonlinear optimization



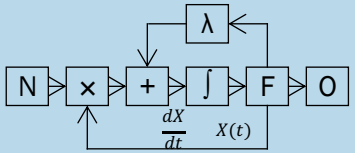
Quantum annealers

Computational neuroscience
& pattern recognition



Analog neuromorphic
networks

Fluid dynamics
& plasma physics



Analog continuous-time
accelerators

Classical

(continuous variables)

Analog

Image sources: Wikimedia.org

A game to show classical rules are insufficient

Premises:

- Physicists observe things obeying these rules.
- First, we try modeling these rules with conventional, “classical” intuition, but we will see it fails.

A game

"Teaching quantum information science to high-school and early undergraduate students" Economou, Rudolph, Barnes, Virginia Tech & Imperial College London

Even if you are familiar with quantum computing ground rules, worth revisiting and asking "why are the rules the way they are?"

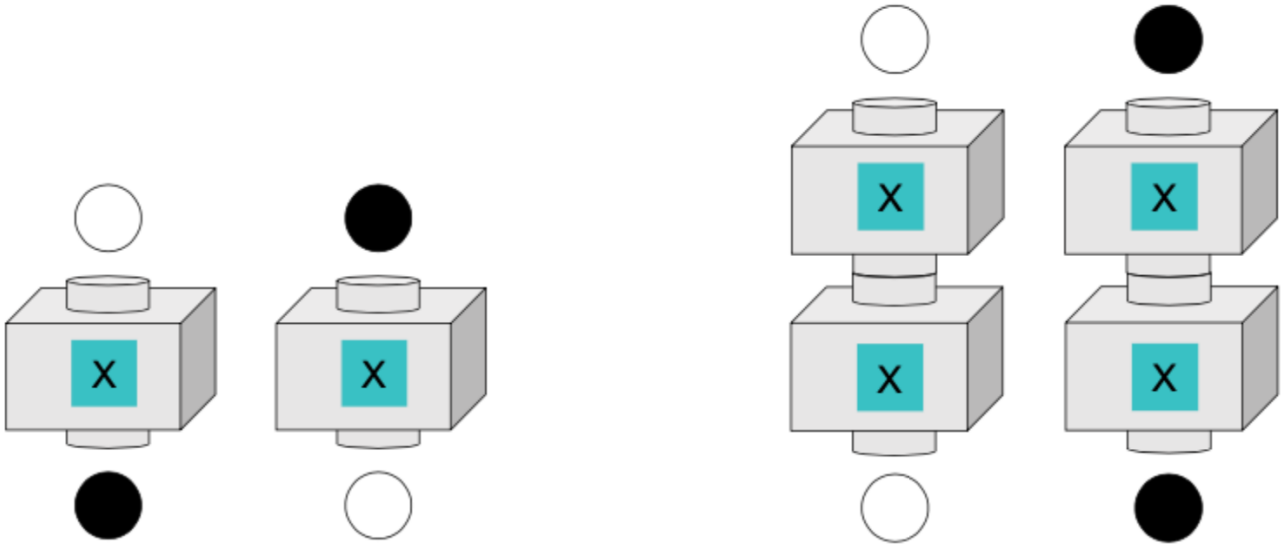


FIG. 2. Basic properties of NOT gates.

A game

"Teaching quantum information science to high-school and early undergraduate students" Economou, Rudolph, Barnes, Virginia Tech & Imperial College London

Even if you are familiar with quantum computing ground rules, worth revisiting and asking "why are the rules the way they are?"

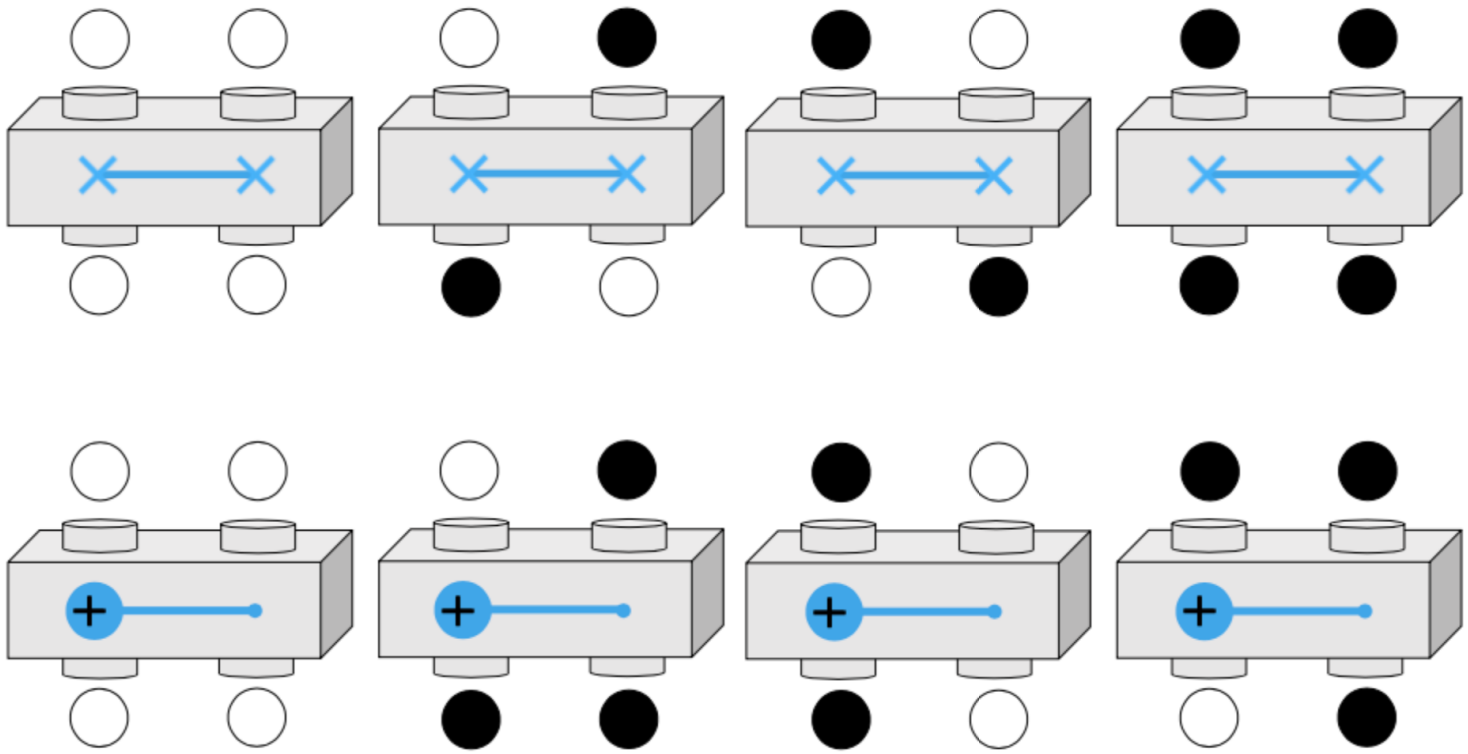


FIG. 3. SWAP and CNOT gates.

A game

"Teaching quantum information science to high-school and early undergraduate students" Economou, Rudolph, Barnes, Virginia Tech & Imperial College London

Even if you are familiar with quantum computing ground rules, worth revisiting and asking "why are the rules the way they are?"

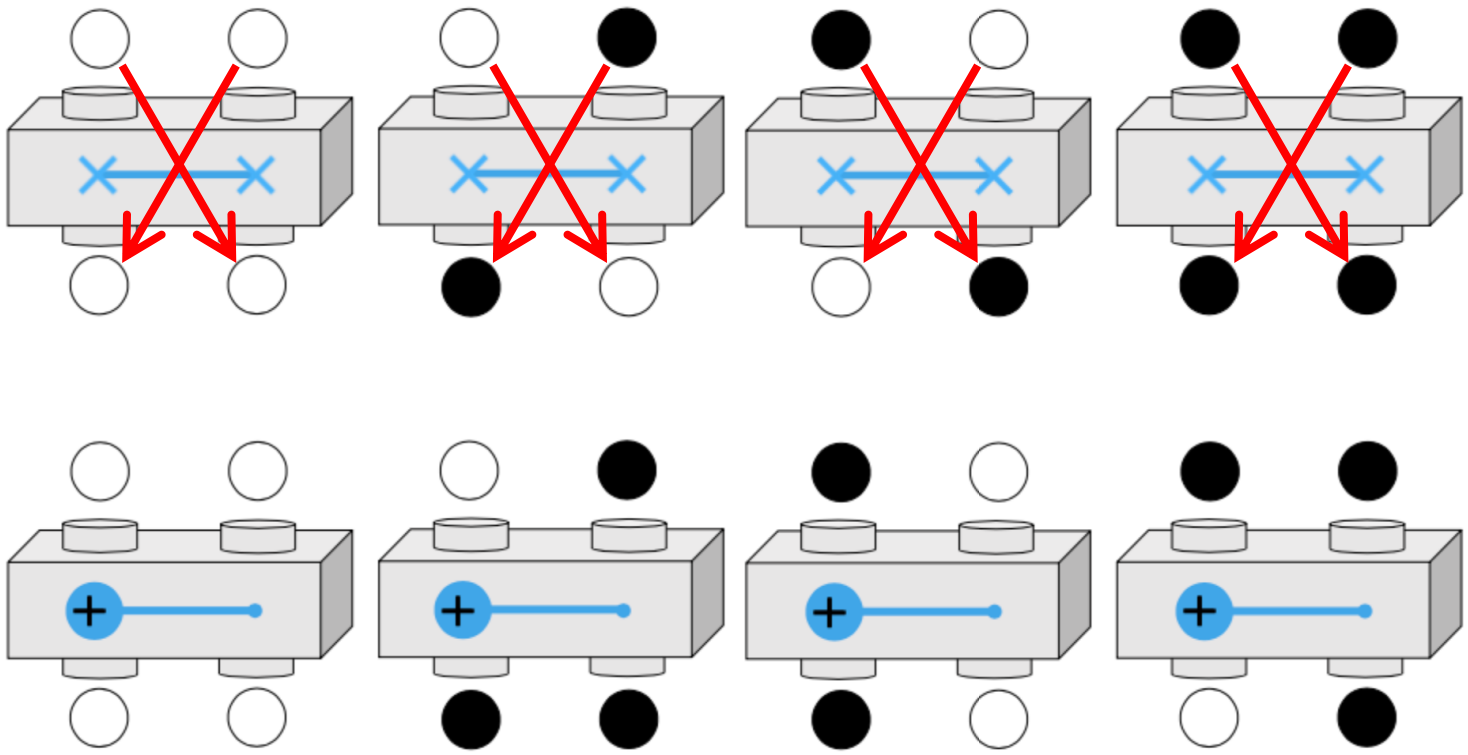


FIG. 3. SWAP and CNOT gates.

A game

"Teaching quantum information science to high-school and early undergraduate students" Economou, Rudolph, Barnes, Virginia Tech & Imperial College London

Even if you are familiar with quantum computing ground rules, worth revisiting and asking "why are the rules the way they are?"

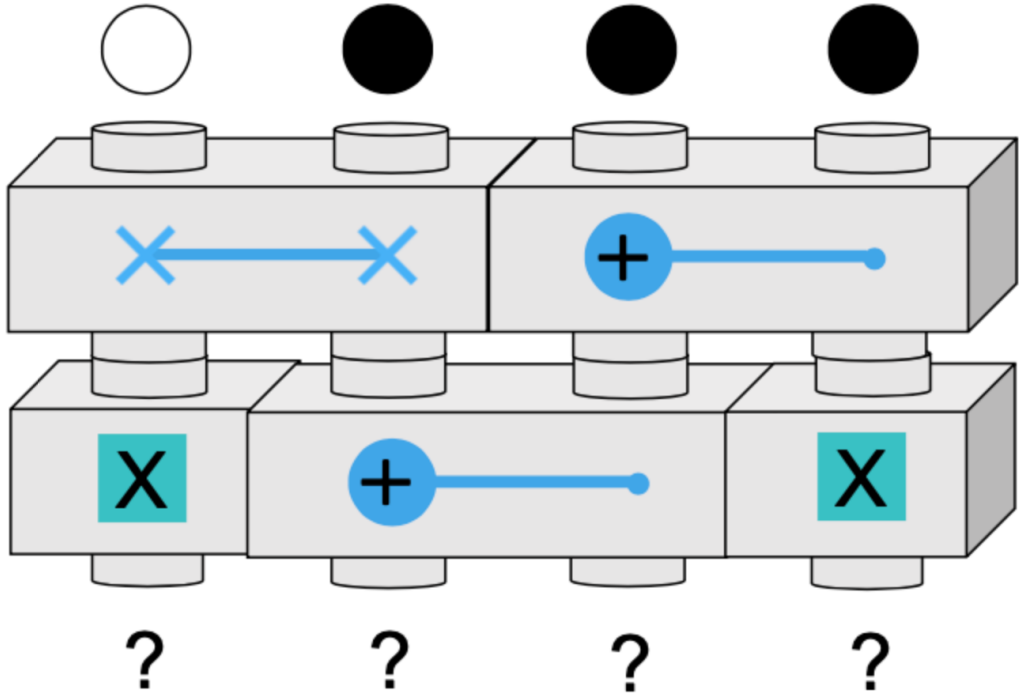


FIG. 4. Exercise with SWAP and CNOT gates.

A game

"Teaching quantum information science to high-school and early undergraduate students" Economou, Rudolph, Barnes, Virginia Tech & Imperial College London

Even if you are familiar with quantum computing ground rules, worth revisiting and asking "why are the rules the way they are?"

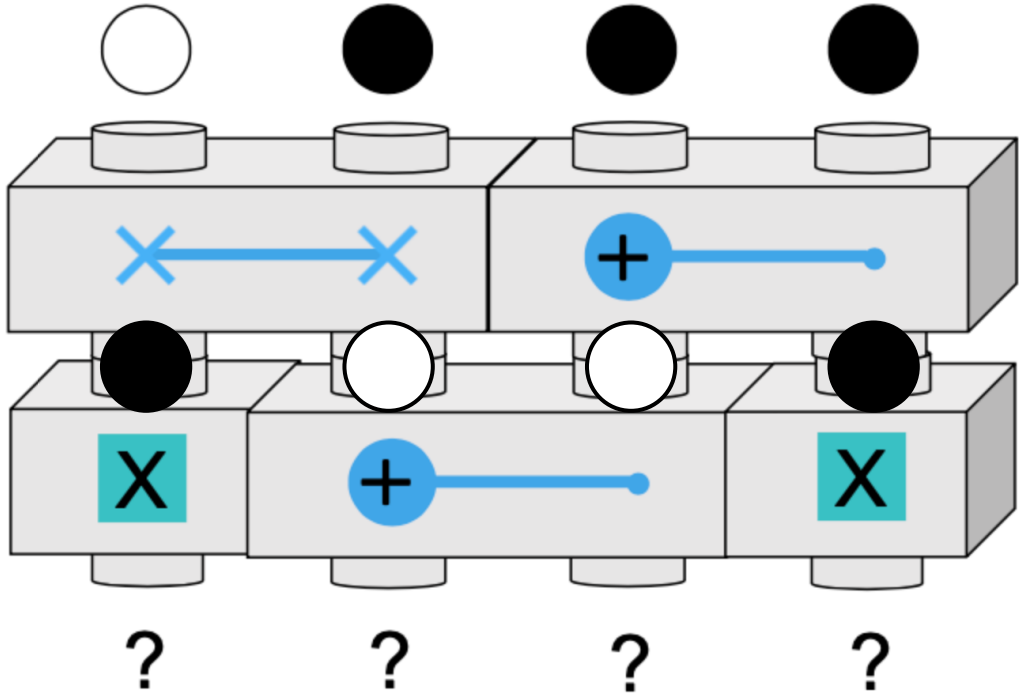


FIG. 4. Exercise with SWAP and CNOT gates.

A game

"Teaching quantum information science to high-school and early undergraduate students" Economou, Rudolph, Barnes, Virginia Tech & Imperial College London

Even if you are familiar with quantum computing ground rules, worth revisiting and asking "why are the rules the way they are?"

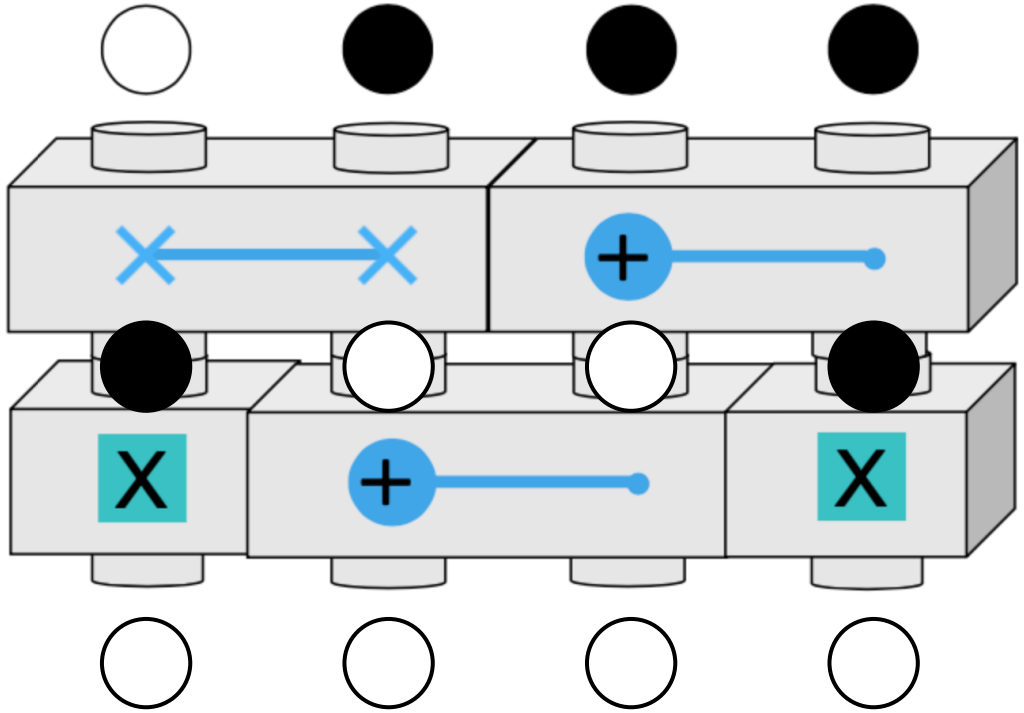


FIG. 4. Exercise with SWAP and CNOT gates.

A game

How far can we push this analogy??

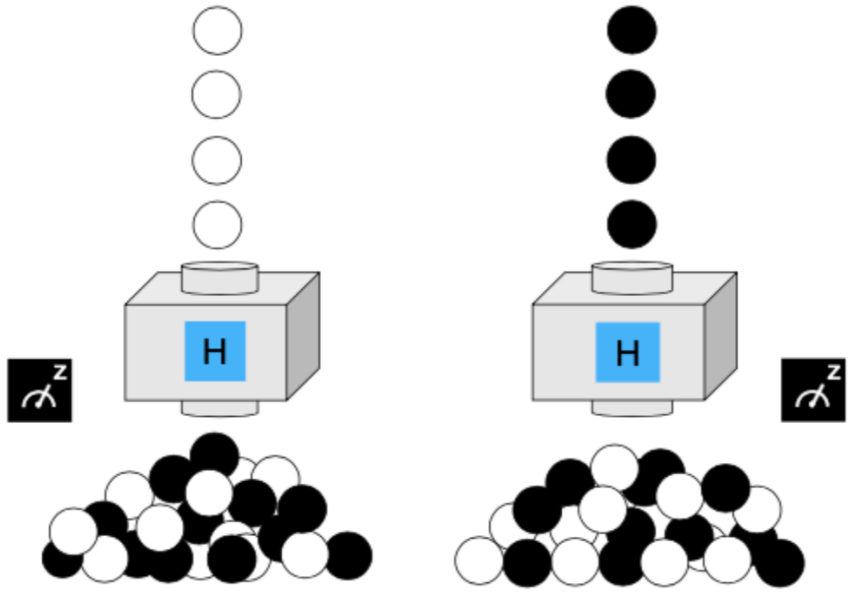


FIG. 5. Introducing the Hadamard gate as a box that produces random outputs upon measurement.

A game

How far can we push this analogy??

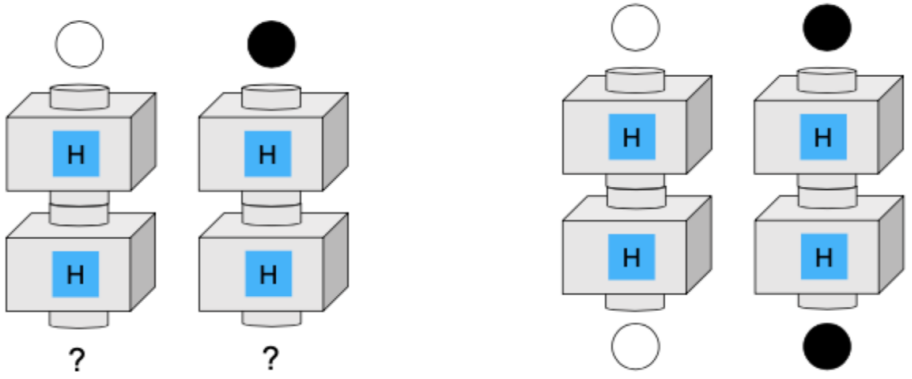


FIG. 6. Showing that the random output of a Hadamard gate goes away when two of them are stacked together.

A game

How far can we push this analogy??

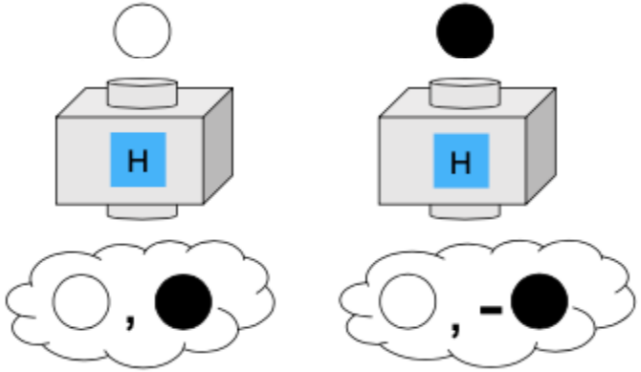
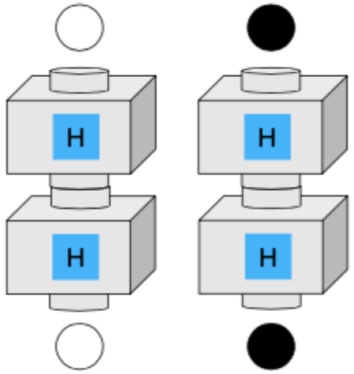


FIG. 7. Hadamard gates produce m

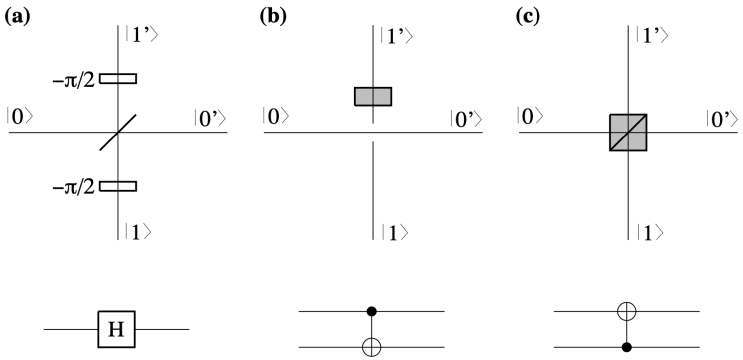
Optical Simulation of Quantum Logic

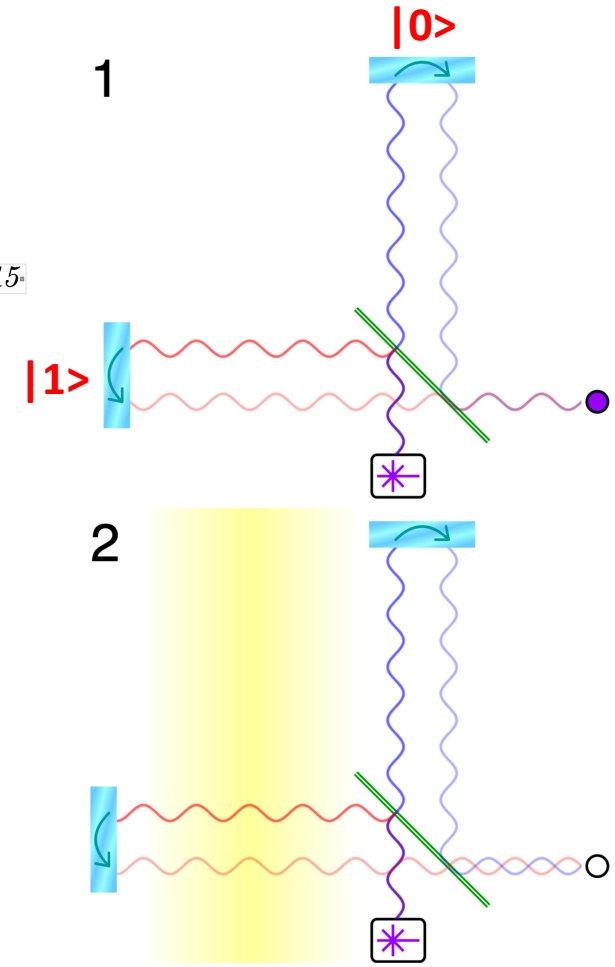
N. J. Cerf[1], C. Adami[1,2], and P. G. Kwiat[3]
[1] *W. K. Kellogg Radiation Laboratory and* [2] *Computation and Neural Systems*
California Institute of Technology, Pasadena, California 91125
[3] *Physics Division, P-23, Los Alamos National Laboratory, Los Alamos, New Mexico 87545*
(March 1997)



ard gate goes away when two of them are stacked together.

FIG. 1. Example of optical simulation of basic quantum logic gates. (a) Hadamard gate on a "location" qubit, using a lossless symmetric beam splitter. (b) Controlled-NOT gate using a polarization rotator. The location and polarization are the control and target qubit, respectively. (c) Same as (b) but the control and target qubits are interchanged by the use of a polarizing beam splitter.





https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Quantum_Tutorials_(Rioux)/Quantum_Optics/277%3A_A_Quantum_Circuit_for_a_Michelson_Interferometer

https://en.wikipedia.org/wiki/LIGO

A game to show classical rules are insufficient

Premises:

- Physicists observe things obeying these rules.
- First, we try modeling these rules with conventional, “classical” intuition, but we will see it fails.

- The states in our game cannot be thought as merely “probabilistic”

Outline

- Course logistics
- A game
- ***<u>1 qubit states: what is a qubit, basis states, superposition, Bloch sphere</u>***
- 1 qubit gates: Pauli-X, Hadamard, Quirk, unitary matrices
- 2 qubit states: basis states, tensor product
- 2 qubit gates: CNOT, $H \otimes H$, entanglement

Maybe first ponder: what is a (classical) bit?

Maybe first ponder: what is a (classical) bit?

- Binary abstraction (high/low voltage)
- Encoded at a specific instant in time
- Statistical property of many particles

1 qubit states: what is a qubit?

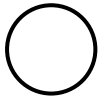
- A two-level quantum state (distinct from classical state)
- Carries more information than a classical bit

1 qubit states: what is a qubit?

Physically, can be:

- Quantized voltage and current (IBM, Google superconducting qubits)
- Electron spins (Intel solid state qubits)
- Atom energy states (UMD, IonQ ion trap qubits)
- Polarization of light in different directions
- Etc.

1 qubit states: basis states



$$\begin{bmatrix}1\\0\end{bmatrix}=|0\rangle$$

$$\begin{bmatrix}0\\1\end{bmatrix}=|1\rangle$$

1 qubit states: superposition

$$\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$\begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

1 qubit states: superposition

$$\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$\begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle$$

1 qubit states: superposition

$$\begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$\begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

$$|\psi\rangle = a_0|0\rangle + a_1|1\rangle$$

a_0, a_1 are amplitudes, are complex-valued

a_0 : how much "zeroness"

a_1 : how much "oneness"

$|a_0|^2 + |a_1|^2 = 1$

Outline

- Course logistics
- A game
- 1 qubit states: what is a qubit, basis states, superposition, Bloch sphere
- ***<u>1 qubit gates: Pauli-X, Hadamard, Quirk, unitary matrices</u>***
- 2 qubit states: basis states, tensor product
- 2 qubit gates: CNOT, $H \otimes H$, entanglement

One qubit gates: Hadamard gate

What would represent H?

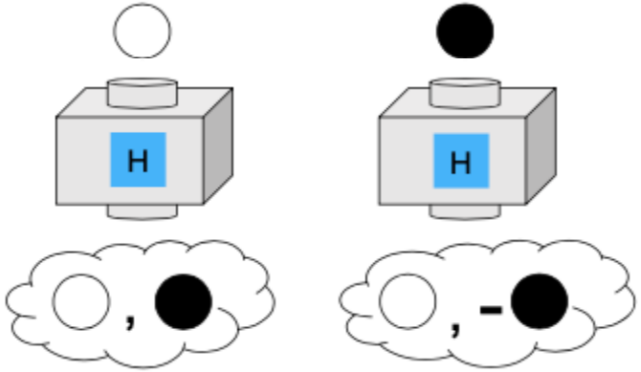


FIG. 7. Hadamard gates produce m

One qubit gates: Hadamard gate

$$H = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

$$H|0\rangle = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

$$H|1\rangle = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

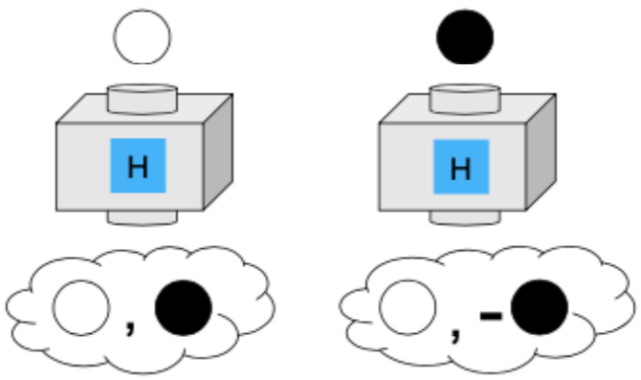


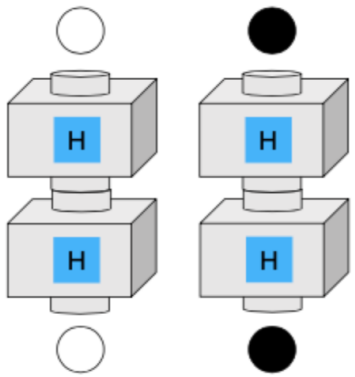
FIG. 7. Hadamard gates produce n

One qubit gates: Hadamard gate

$$HH|0\rangle = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

$$HH|1\rangle = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

$$HH = \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$$



rd gate goes away when two of them are stacked together.

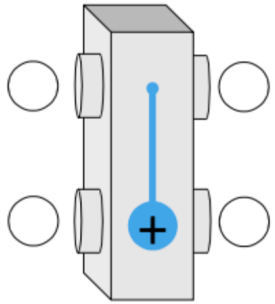
Outline

- Course logistics
- A game
- 1 qubit states: what is a qubit, basis states, superposition, Bloch sphere
- 1 qubit gates: Pauli-X, Hadamard, Quirk, unitary matrices
- ***2 qubit states: basis states, tensor product***
- 2 qubit gates: CNOT, $H \otimes H$, entanglement

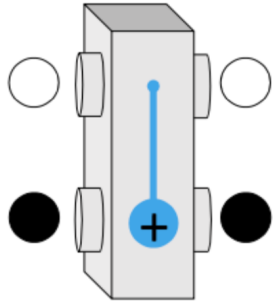
Outline

- Course logistics
- A game
- 1 qubit states: what is a qubit, basis states, superposition, Bloch sphere
- 1 qubit gates: Pauli-X, Hadamard, Quirk, unitary matrices
- 2 qubit states: basis states, tensor product
- ***2 qubit gates: CNOT, H$\otimes$H, entanglement***

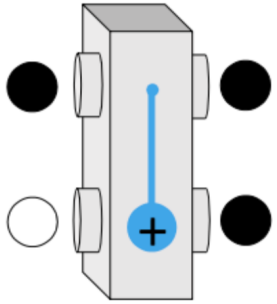
2 qubit gates: CNOT



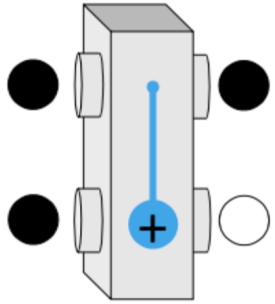
$$\mathrm{CNOT}|00\rangle = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix} \begin{bmatrix} 1 \\ \\ \\ \end{bmatrix} = \begin{bmatrix} 1 \\ \\ \\ \end{bmatrix} = |00\rangle = |0\rangle \otimes |0\rangle$$



$$\mathrm{CNOT}|01\rangle = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix} \begin{bmatrix} \\ 1 \\ \\ \end{bmatrix} = \begin{bmatrix} \\ 1 \\ \\ \end{bmatrix} = |01\rangle = |0\rangle \otimes |1\rangle$$

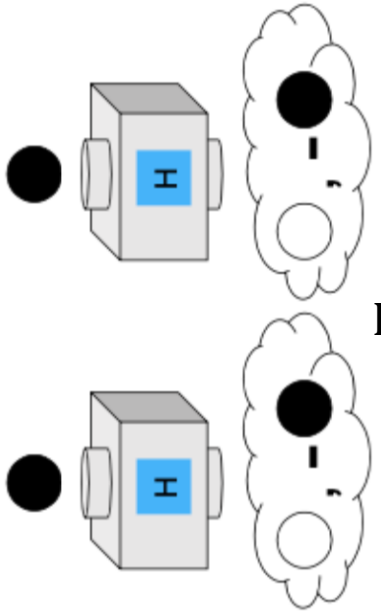


$$\mathrm{CNOT}|10\rangle = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix} \begin{bmatrix} \\ \\ 1 \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \\ 1 \end{bmatrix} = |11\rangle = |1\rangle \otimes |0\rangle$$



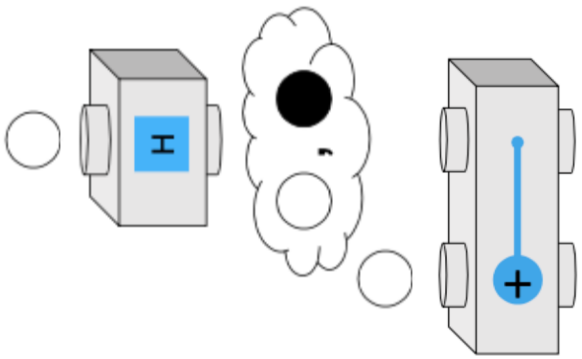
$$\mathrm{CNOT}|11\rangle = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix} \begin{bmatrix} \\ \\ \\ 1 \end{bmatrix} = \begin{bmatrix} \\ \\ 1 \\ \end{bmatrix} = |10\rangle = |1\rangle \otimes |1\rangle$$

2 qubit gates: H⊗H



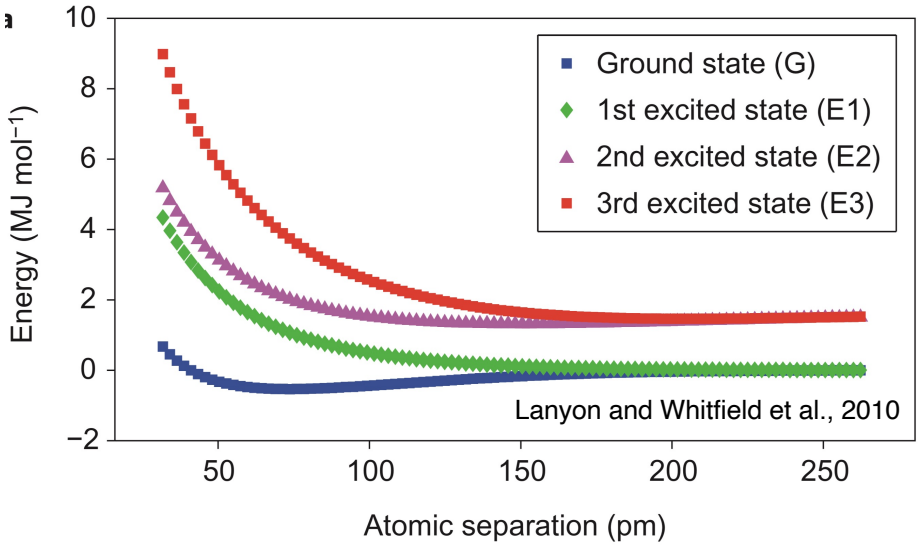
$$\mathrm{H} \otimes \mathrm{H}|11\rangle = \begin{bmatrix} \frac{1}{\sqrt{2}}H & \frac{1}{\sqrt{2}}H \\ \frac{1}{\sqrt{2}}H & \frac{-1}{\sqrt{2}}H \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & -1/2 & 1/2 & -1/2 \\ 1/2 & 1/2 & -1/2 & -1/2 \\ 1/2 & -1/2 & -1/2 & 1/2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1/2 \\ -1/2 \\ -1/2 \\ 1/2 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{bmatrix} \\ \frac{-1}{\sqrt{2}} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{bmatrix} \end{bmatrix} = H|1\rangle \otimes H|1\rangle$$

2 qubit gates: entanglement



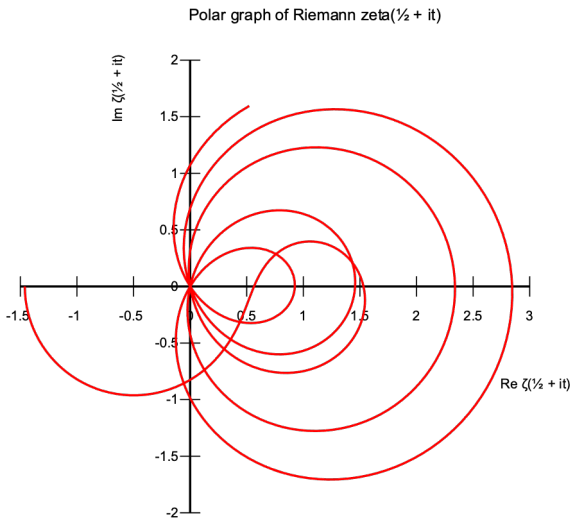
$$\mathrm{CNOT}(\mathrm{H} \otimes \mathrm{I}|00\rangle) = \mathrm{CNOT}(\mathrm{H}|0\rangle \otimes \mathrm{I}|0\rangle) = \mathrm{CNOT}\begin{bmatrix} \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\end{bmatrix} \\ \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\end{bmatrix} \end{bmatrix} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix}\begin{bmatrix} 1/\sqrt{2} \\ 0 \\ 1/\sqrt{2} \\ 0 \end{bmatrix} = \begin{bmatrix} 1/\sqrt{2} \\ 0 \\ 0 \\ 1/\sqrt{2} \end{bmatrix} = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

Motivation: Race to practical quantum computation



Quantum algorithms for chemical simulations

- Calculate properties of molecules directly from governing equations
- Use quantum mechanical computer to simulate quantum mechanics!



Shor's quantum algorithm for factoring integers

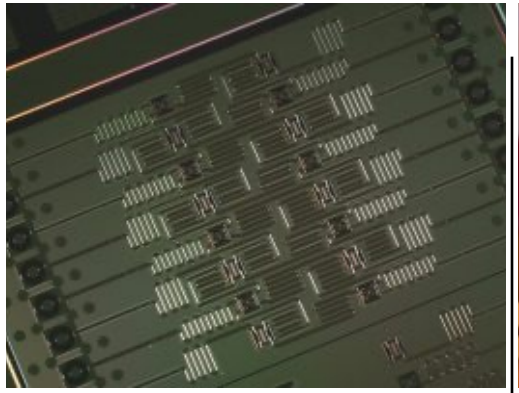
- Factor large integers to primes in polynomial time complexity
- Surpasses any known classical algorithm taking exponential time complexity

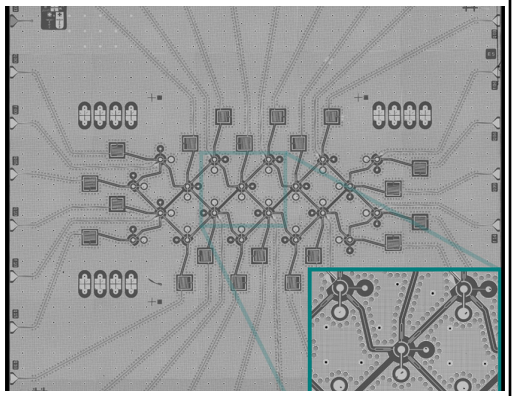
Hundreds of algorithms @ QuantumAlgorithmZoo.org

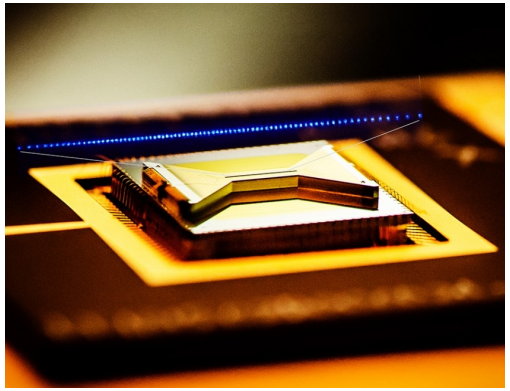
Motivation: Race to practical quantum computation

Superconducting qubits

Trapped ion qubits







IBM

Google

Intel

Rigetti

University of Maryland / IonQ

Many research teams now competing towards more reliable and more numerous qubits.

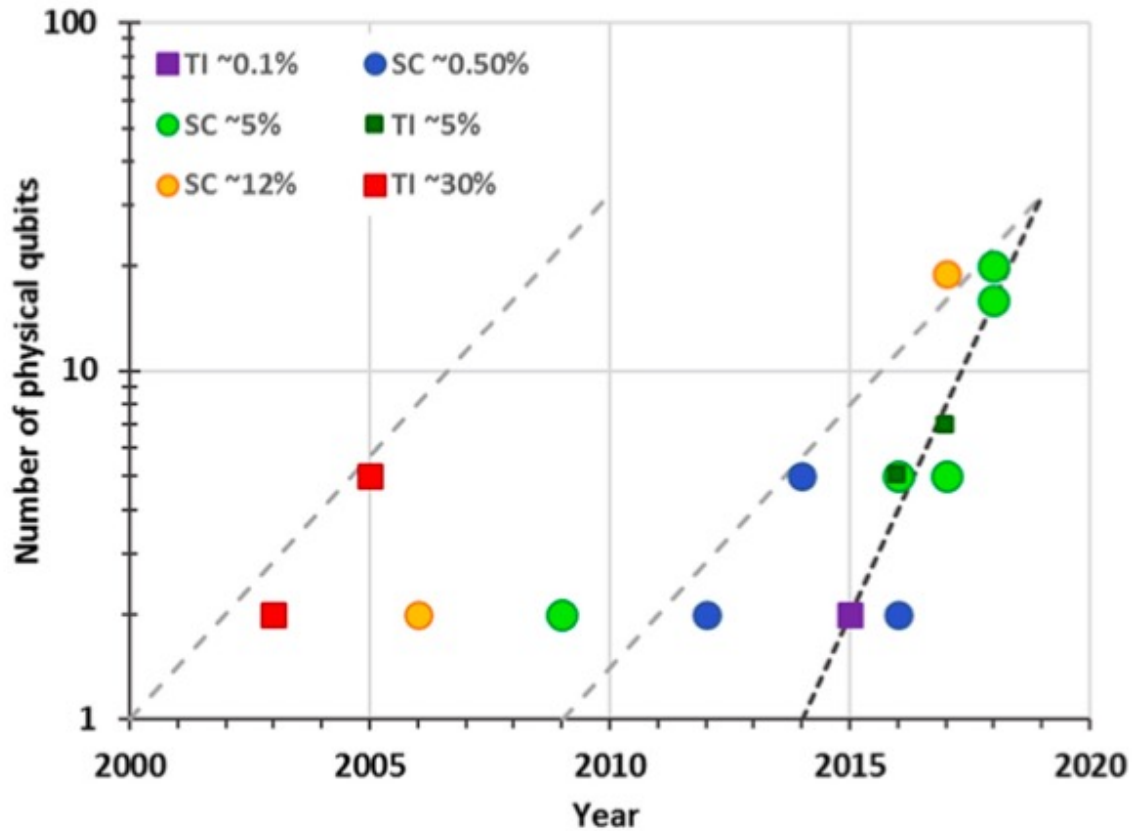


FIGURE 7.2 The number of qubits in superconductor (SC) and trapped ion (TI) quantum computers versus year; note the logarithmic scaling of the vertical axis. Data for trapped ions are shown as squares and for superconducting machines are shown as circles. Approximate average reported two-qubit gate error rates are indicated by color; points with the same color have similar error rates. The dashed gray lines show how the number of qubits would grow if they double every two years starting with one qubit in 2000 and 2009, respectively; the dashed black line indicates a doubling every year beginning with one qubit in 2014. Recent superconductor growth has been close to doubling every year. If this rate continued, 50 qubit machines with less than 5 percent error rates would be reported in 2019. SOURCE: Plotted data obtained from multiple sources [9].

Quantum Computing Progress and Prospects. National Academies Press.

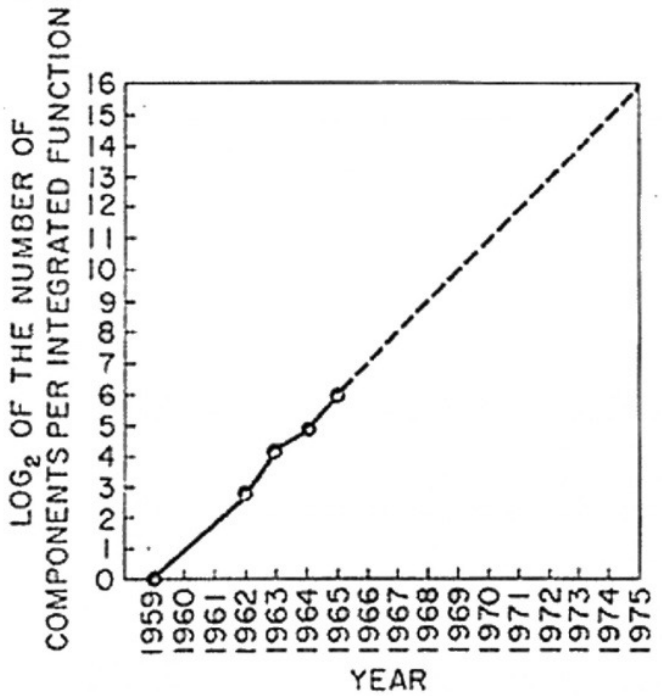


Fig. 2 Number of components per integrated function for minimum cost per component extrapolated vs time.

Intel

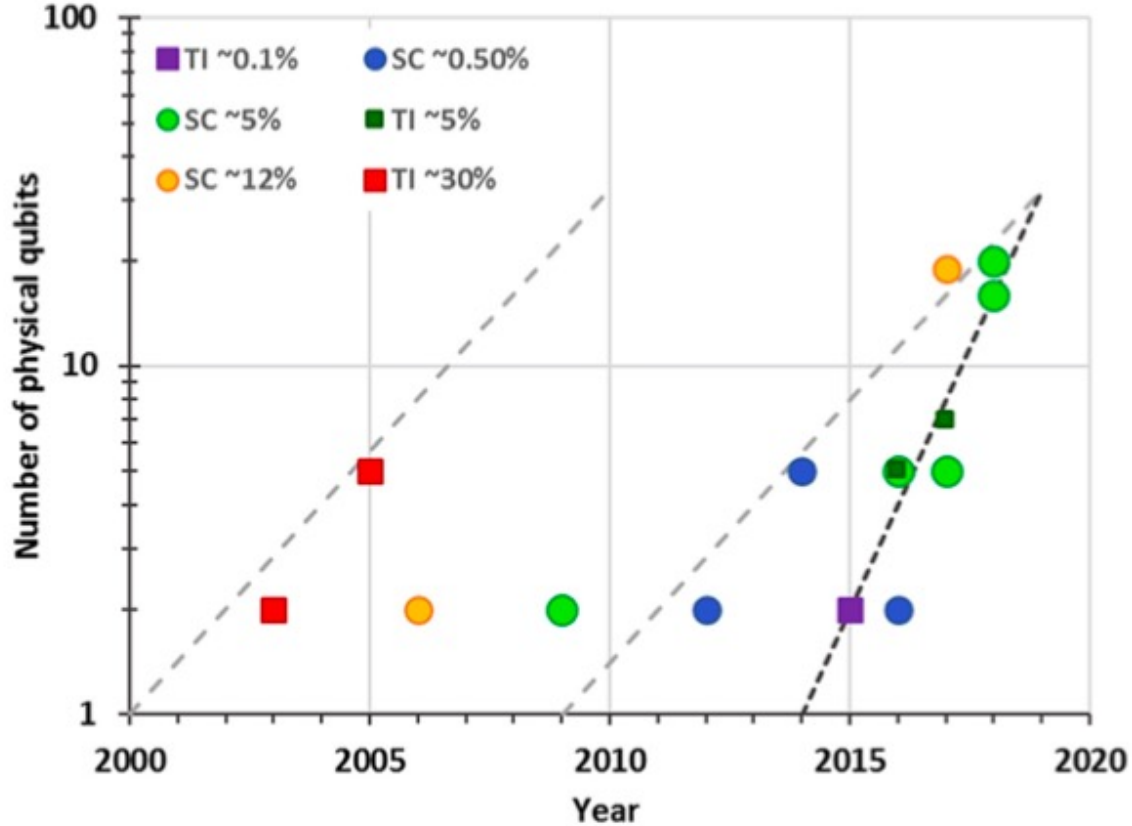


FIGURE 7.2 The number of qubits in superconductor (SC) and trapped ion (TI) quantum computers versus year; note the logarithmic scaling of the vertical axis. Data for trapped ions are shown as squares and for superconducting machines are shown as circles. Approximate average reported two-qubit gate error rates are indicated by color; points with the same color have similar error rates. The dashed gray lines show how the number of qubits would grow if they double every two years starting with one qubit in 2000 and 2009, respectively; the dashed black line indicates a doubling every year beginning with one qubit in 2014. Recent superconductor growth has been close to doubling every year. If this rate continued, 50 qubit machines with less than 5 percent error rates would be reported in 2019. SOURCE: Plotted data obtained from multiple sources [9].

Quantum Computing Progress and Prospects. National Academies Press.

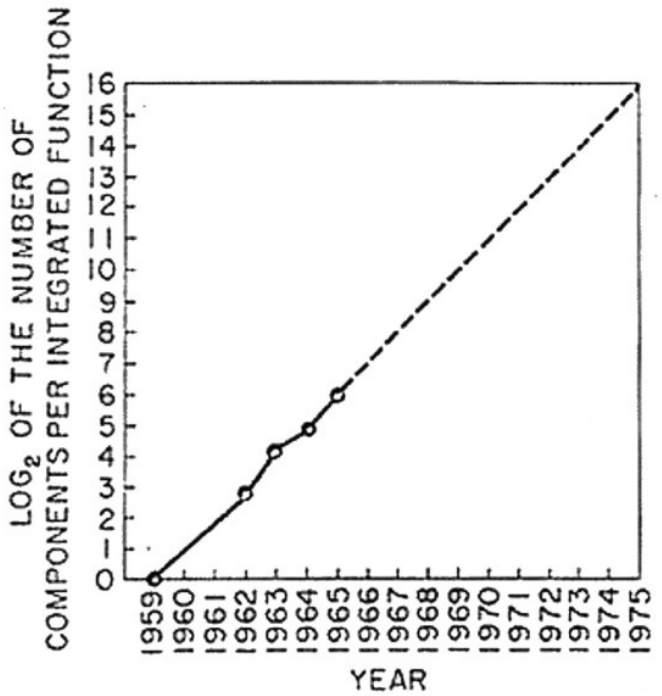
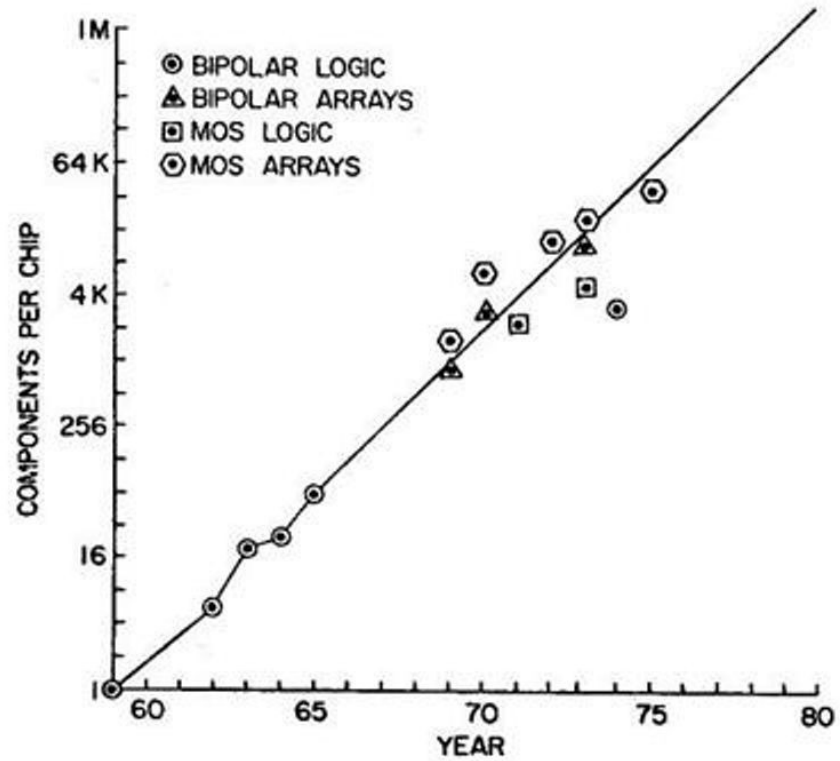


Fig. 2 Number of components per integrated function for minimum cost per component extrapolated vs time.

Intel



Computer History Museum

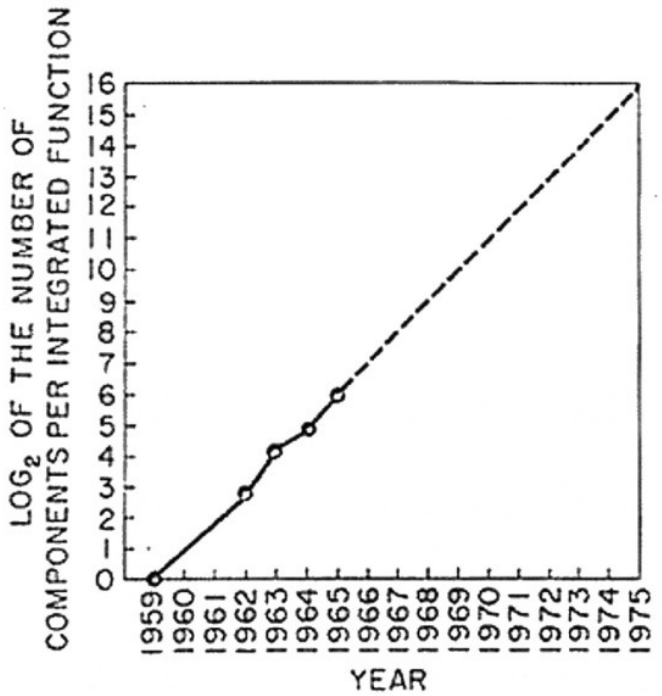
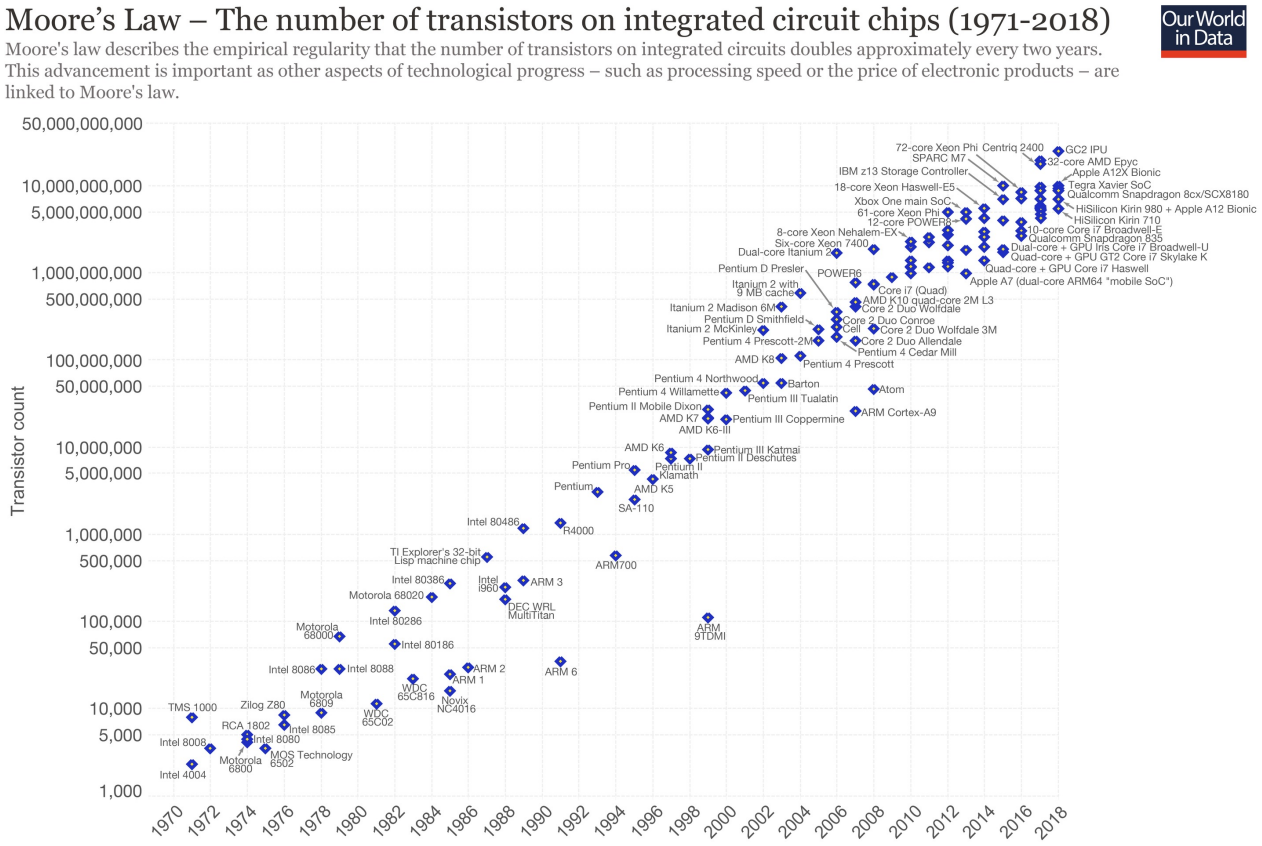


Fig. 2 Number of components per integrated function for minimum cost per component extrapolated vs time.

Intel



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Wikipedia

How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Gidney and Ekerå. 2019.

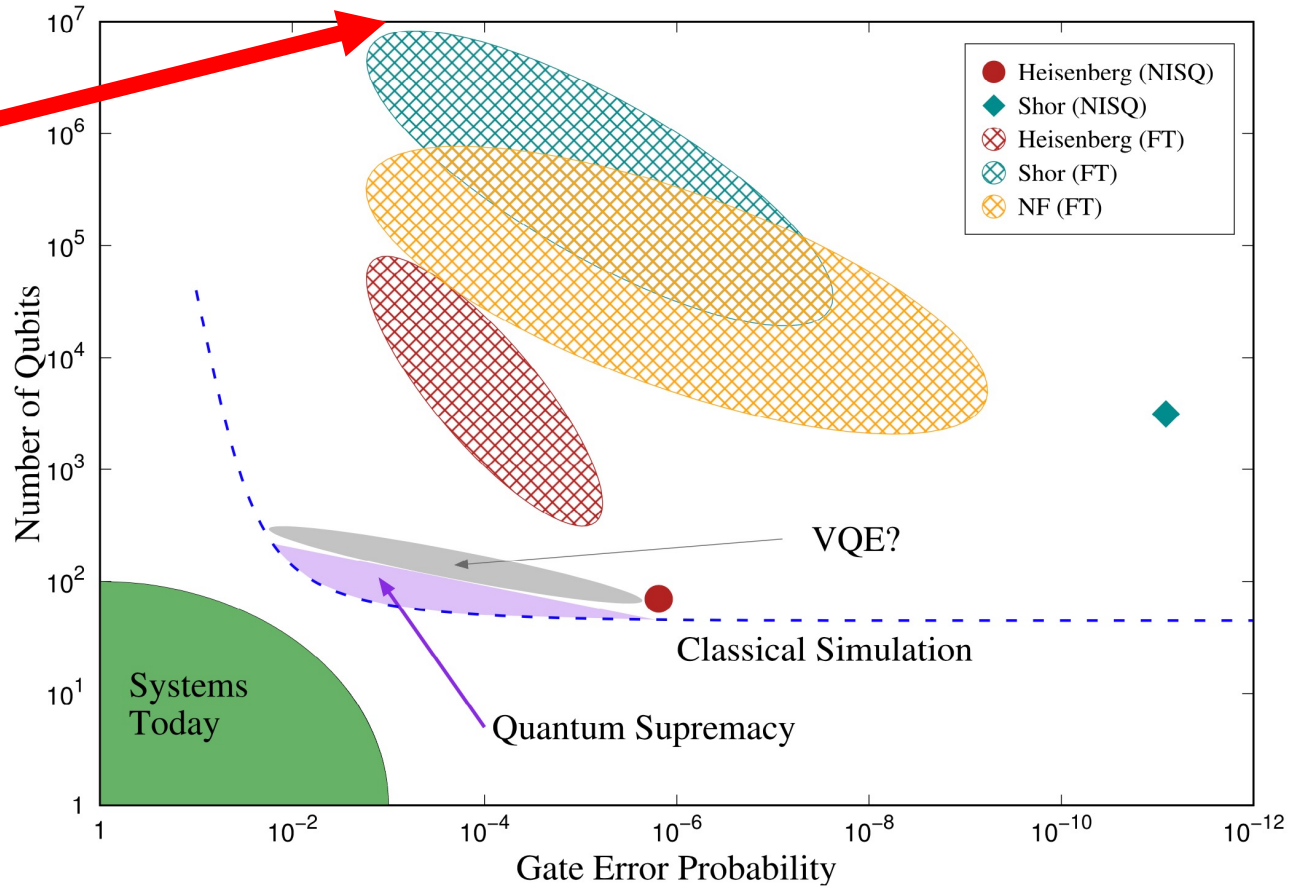


Fig. 2. ***Performance space of quantum computers, measured by the error probability of each entangling gate in the horizontal axis (roughly inversely proportional to the total number of gates that can be executed on a NISQ machine), and the number of qubits in the system in the vertical axis. Blue dotted line approximately demarcates quantum systems that can be simulated using best classical computers, while the green colored region shows where the existing quantum computing systems with verified performance numbers lie (as of September 2018). Purple shaded region indicates computational tasks that accomplish the so-called "quantum supremacy," where the computation carried out by the quantum computer defies classical simulation regardless of its usefulness. The different shapes illustrate resource counts for solving various problems, with solid symbols corresponding to the exact entangling gate counts and number of qubits in NISQ machines, and shaded regions showing approximate gate error requirements and number of qubits for an FT implementation (not pictured are the regions where the error gets too close to the known fault-tolerance thresholds): cyan diamond and shaded region correspond to factoring a 1024-bit number using Shor's algorithm [14], magenta circle and shaded region represent simulation of a 72-spin Heisenberg model [20], and orange shaded region illustrates NF simulation [21].***

An Outlook for Quantum Computing. Maslov et al.

Broad view of open challenges in quantum computer engineering

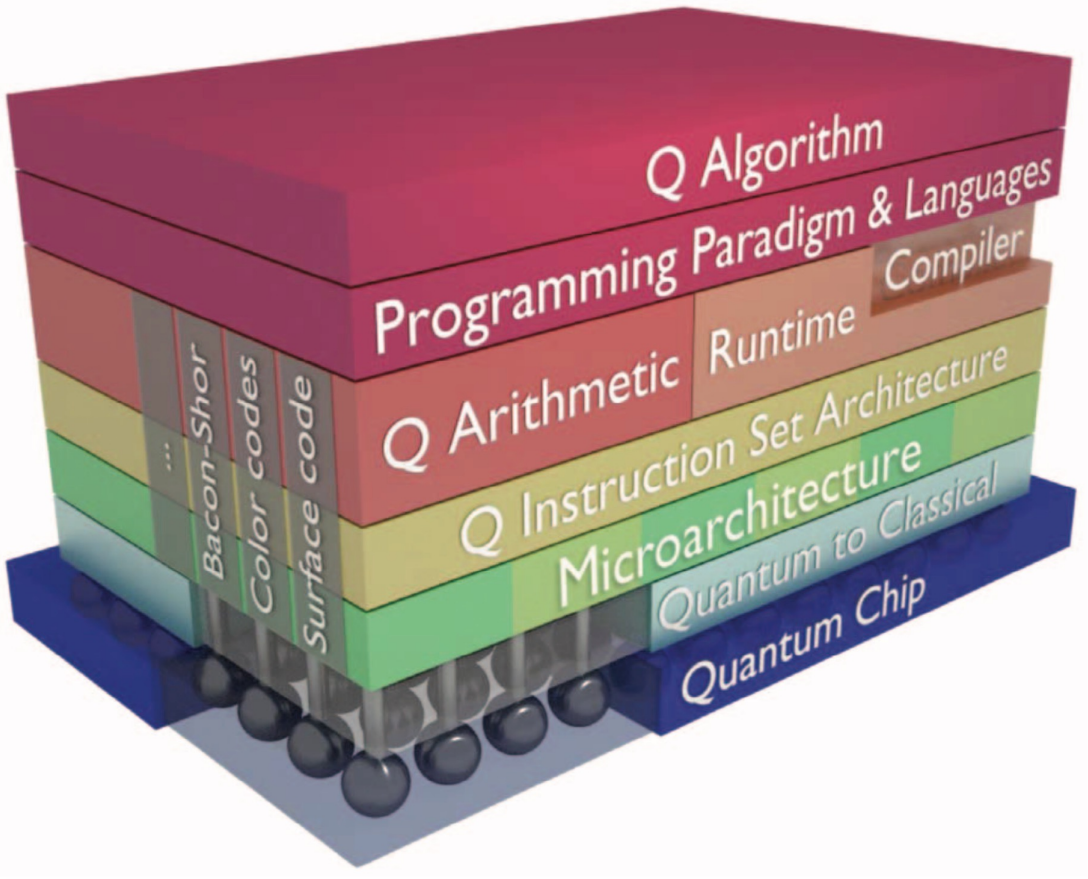


Figure 1. Overview of the quantum computer system stack.

A Microarchitecture for a Superconducting Quantum Processor. Fu et al.

- A complete view of full-stack quantum computing.
- In short, challenges are in finding and building abstractions.
- In each layer, why we don't or can't have good abstractions right now.
- Recent and rapidly developing field of research.

All the quantum computer abstractions we don’t yet have right now

0. Quantum computer support for quantum computer engineering
1. Fault-tolerant, error-corrected quantum algorithms
2. Mature, high level quantum programming languages
3. Universally accepted quantum ISAs
4. Uniform, fully connected quantum device architectures
5. Reliable quantum gates and qubits

Semantic gap

- Need languages, abstractions…

Tools gap

- Need optimizing compilers, simulators, debuggers…

Infrastructure gap

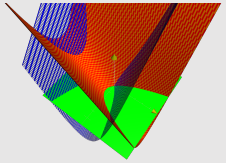
- Need more abundant, more reliable qubits…

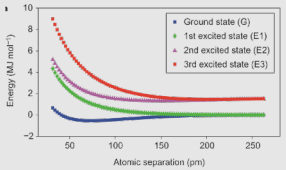
Educational gap

- Need researchers, students…

Nonlinear scientific computation

Quantum simulation & optimization





New and extreme workload challenges

Multicore CPUs, GPUs, FPGAs, ASICs, analog, quantum, etc.

Limitations in transistor scaling

Dennard's scaling already ended

Moore's law increasingly costly to sustain

Open challenges in emerging architectures:

Problem abstractions

- How do you accurately solve big problems?

Programming abstractions

- Can you borrow ideas from conventional computing?

Architecture abstractions

- How to interface with the unconventional hardware?

Image sources: Lanyon and Whitfield et al., 2010

A guide to the rest of the CS:APP textbook

CS:APP Chapter	Rutgers CS / ECE course in AY21-22 for further study
4. Processor Architecture	ECE 563 Computer Architecture I
5. Optimizing Program Performance	CS 314 Principles of Programming Languages / CS 415 Compilers
7. Linking	(CS 415 Compilers)
8. Exceptional Control Flow	CS 416 Operating Systems Design
9. Virtual Memory	CS 416 Operating Systems Design
10. System-Level I/O	CS 416 Operating Systems Design
11. Network Programming	CS 352 Internet Technology
12. Concurrent Programming	CS 214 Systems Programming / ECE 451 Parallel & Distributed Computing

2021 Fall: ECE 493. Soljanin. Quantum Computing Algorithms. (seniors only).
2022 Spring: Physics 421. Schnetzer. An Introduction to Quantum Computing.
2022 Fall: CS 583. Huang. Quantum Computing: Programs and Systems.

Very important to help develop next iteration of this course.

- https://sirs.ctaar.rutgers.edu/blue