# Demystifying Quantum Computing

Santiago $\longleftrightarrow$ Leiden 2026 • Lecture 1 of 3

Alfons Laarman

Leiden University
Santiago 2026

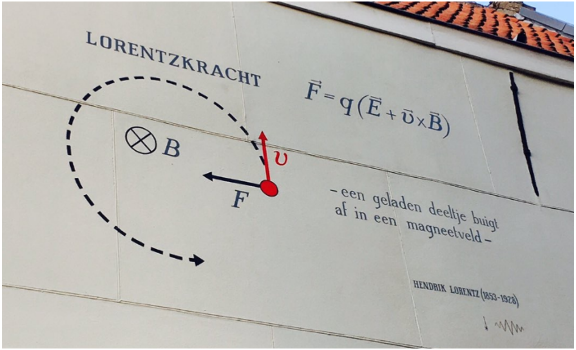
Universiteit Leiden
The Netherlands

# Leiden Scientists and Inventions


Hendrik Lorentz

`https://muurformules.nl/`

# Leiden Scientists and Inventions




Hendrik Lorentz

https://muurformules.nl/

# Leiden Scientists and Inventions


Willebrord Snellius

`https://muurformules.nl/`

# Leiden Scientists and Inventions


Christiaan Huygens

https://muurformules.nl/

# Leiden Scientists and Inventions


Uhlenbeck / Goudsmit

https://muurformules.nl/

# Leiden's Contributions to Physics

- Photonics
- Classical Mechanics
- Quantum Mechanics
- Magnetism
- Thermodynamics
- Superconductivity
- ...

# Leiden's Contributions to Physics

- Photonics
- Classical Mechanics
- Quantum Mechanics
- Magnetism
- Thermodynamics
- Superconductivity
- ...

For the next three hours you will have to do with a mere computer scientist (me).

# Leiden's Contributions to Physics

- Photonics
- Classical Mechanics
- Quantum Mechanics
- Magnetism
- Thermodynamics
- Superconductivity
- . . .

For the next three hours you will have to do with a mere computer scientist (me).

But computer science is just getting started, and we are already winning physics prizes:

# Santiago 2026 — a three-lecture series

14:30 – 15:15 Demystifying Quantum Computing

15:30 – 16:15 Demystifying Quantum Computing

16:30 – 17:15 Quantum Circuit Analysis with #SAT

16:30 – 17:15 Knowledge Representation for Quantum Information

# Santiago 2026 — a three-lecture series

14:30 – 15:15 Demystifying Quantum Computing

15:30 – 16:15 Demystifying Quantum Computing

16:30 – 17:15 Quantum Circuit Analysis with #SAT

16:30 – 17:15 Knowledge Representation for Quantum Information

We will interleave the tutorial part.

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition.

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓
- A measurement changes the quantum state.

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓
- A measurement changes the quantum state. ✓

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓
- A measurement changes the quantum state. ✓
- A computational basis state can be entangled.

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓
- A measurement changes the quantum state. ✓
- A computational basis state can be entangled. ✗

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓
- A measurement changes the quantum state. ✓
- A computational basis state can be entangled. ✗
- Quantum computers provide exponential speedups over classical computers.

# Quantum Computing Quiz: True or False?

- The single-qubit Hadamard gate can create superposition. ✓
- A measurement changes the quantum state. ✓
- A computational basis state can be entangled. ✗
- Quantum computers provide exponential speedups over classical computers. ?

Can we illuminate superposition, entanglement, and measurements based on concepts that we computer scientists understand well?

## Can we illuminate superposition, entanglement, and measurements based on concepts that we computer scientists understand well?

Takeaway: we will see that quantum computing concepts can be understood based on our intuition from classical circuits and probabilistic computing.

## Can we illuminate superposition, entanglement, and measurements based on concepts that we computer scientists understand well?

Takeaway: we will see that quantum computing concepts can be understood based on our intuition from classical circuits and probabilistic computing.

"Quantum computing becomes easy once one takes the physics out"

— Scott Aaronson

## Can we illuminate superposition, entanglement, and measurements based on concepts that we computer scientists understand well?

Takeaway: we will see that quantum computing concepts can be understood based on our intuition from classical circuits and probabilistic computing.

"Quantum computing becomes easy once one takes the physics out"

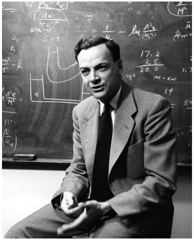
— Scott Aaronson

It is even easier if you also take out linear algebra.

Alfons Laarman

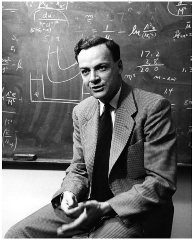
# Quantum Computing

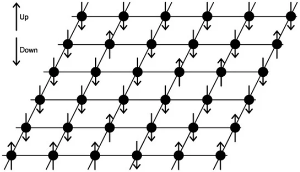
*"Nature isn't classical, d****t, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."*



# Quantum Computing

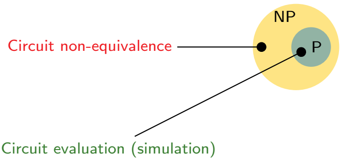
*"Nature isn't classical, d****t, and if you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."*



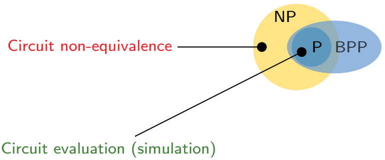



Simulating a quantum mechanical system requires exponentially many calculations.

# New complexity classes: BQP and QMA



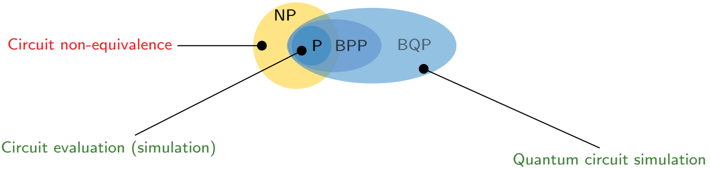
# New complexity classes: BQP and QMA




**Definition (Probabilitistic Turing Machine)**

Turing Machine with a probability distribution over transitions.

# **New complexity classes:** BQP **and** QMA




## Definition (Probabilitistic Turing Machine)

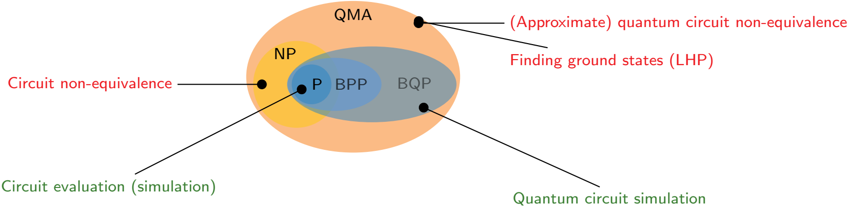
Turing Machine with a probability distribution over transitions.

## Definition (Deutsch's Quantum Turing Machine)

Roughly:

1. Take a Probabilistic Turing Machine (the computational model in BPP)
2. Swap out the (full) probability matrix in the transition relation for a unitary matrix

# New complexity classes: BQP and QMA




## Definition (Probabilitistic Turing Machine)

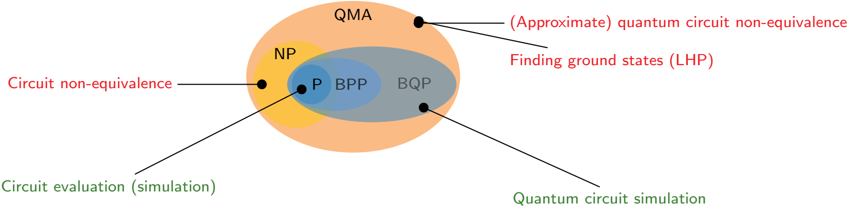
Turing Machine with a probability distribution over transitions.

## Definition (Deutsch's Quantum Turing Machine)

Roughly:

1. Take a Probabilistic Turing Machine (the computational model in BPP)
2. Swap out the (full) probability matrix in the transition relation for a unitary matrix

# New complexity classes: BQP and QMA




## Definition (Probabilitistic Turing Machine)

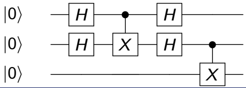
Turing Machine with a probability distribution over transitions.

## Definition (Deutsch's Quantum Turing Machine)

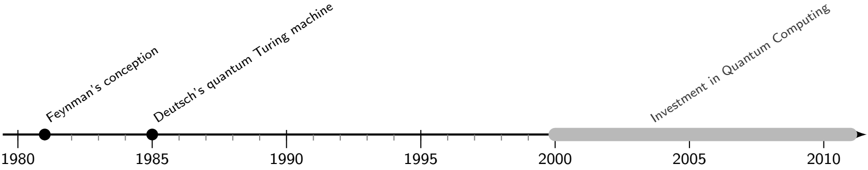
Roughly:

1. Take a Probabilistic Turing Machine (the computational model in BPP)
2. Swap out the (full) probability matrix in the transition relation for a unitary matrix

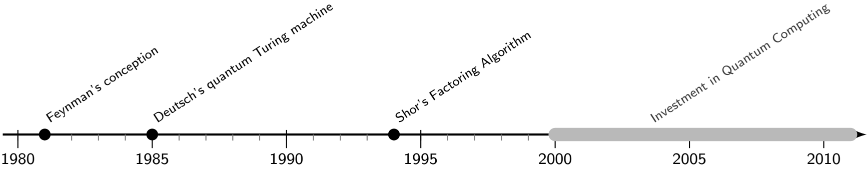
Quantum Turing machines are not used: We will use circuits as computational model.



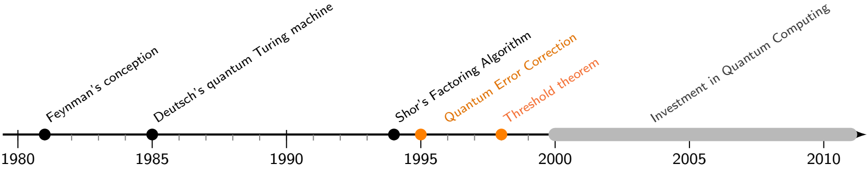
# Quantum Computing Timeline



# Quantum Computing Timeline

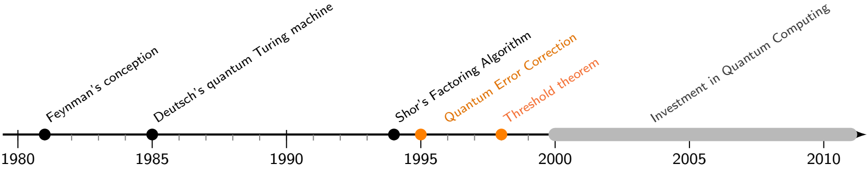


# Quantum Computing Timeline




The goal: fully error-corrected quantum computation as formalized in quantum Turing Machine and quantum circuits.

# Quantum Computing Timeline




The goal: fully error-corrected quantum computation as formalized in quantum Turing Machine and quantum circuits.

Many horses (physical phenomena) to bet on:

# Towards Quantum Supremacy

## The main open questions

- Is there a quantum algorithm that outperformans classical algorithms in practice?

# Towards Quantum Supremacy

## The main open questions

- Is there a quantum algorithm that outperformans classical algorithms in practice?
- Is there a problem where quantum computers have exponential advantage?
  (or is BQP = BPP?)

# Towards Quantum Supremacy

## The main open questions

- Is there a quantum algorithm that outperformans classical algorithms in practice?
- Is there a problem where quantum computers have exponential advantage?
  (or is BQP = BPP?)

$\Rightarrow$ Opportunity:

- finding new quantum algorithms
- charting the classical/quantum boundary
- quantum foundations: non-locality, contextuality, ...

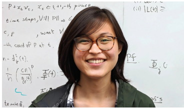
# Towards Quantum Supremacy

## The main open questions

- Is there a quantum algorithm that outperformans classical algorithms in practice?
- Is there a problem where quantum computers have exponential advantage?
  (or is BQP = BPP?)

$\Rightarrow$ Opportunity:

- finding new quantum algorithms
- charting the classical/quantum boundary
- quantum foundations: non-locality, contextuality, ...



Classical algorithms can surprise us: Ewin Tang [1] gave an efficient *classical* algorithm for the recommender problem (at age 18) — charting that boundary is valuable in itself.

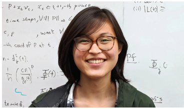
# Towards Quantum Supremacy

## The main open questions

- Is there a quantum algorithm that outperformans classical algorithms in practice?
- Is there a problem where quantum computers have exponential advantage?
  (or is BQP = BPP?)

⇒ Opportunity:

- finding new quantum algorithms
- charting the classical/quantum boundary
- quantum foundations: non-locality, contextuality, ...



Classical algorithms can surprise us: Ewin Tang [1] gave an efficient *classical* algorithm for the recommender problem (at age 18) — charting that boundary is valuable in itself.

## ... progress remains important regardless of quantum supremacy

- Even without a quantum computer, Nature keeps posing "quantum-hard" questions: simulating physical systems (BQP), finding ground states (QMA), ...

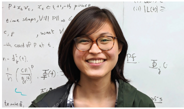
# Towards Quantum Supremacy

## The main open questions

- Is there a quantum algorithm that outperformans classical algorithms in practice?
- Is there a problem where quantum computers have exponential advantage?
  (or is BQP = BPP?)

$\Rightarrow$ Opportunity:

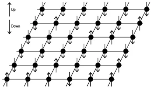
- finding new quantum algorithms
- charting the classical/quantum boundary
- quantum foundations: non-locality, contextuality, ...



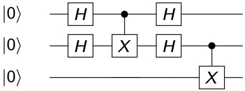
Classical algorithms can surprise us: Ewin Tang [1] gave an efficient *classical* algorithm for the recommender problem (at age 18) — charting that boundary is valuable in itself.

## ... progress remains important regardless of quantum supremacy

- Even without a quantum computer, Nature keeps posing "quantum-hard" questions: simulating physical systems (BQP), finding ground states (QMA), . . .
- Progress on quantum circuits translates back to these physics applications.




$\longleftarrow$



# Demystifying Quantum Computing

"Quantum computing becomes easy once one takes the physics out"

— Scott Aaronson

It is even easier if you also take out the linear algebra.

Alfons Laarman

# Tutorial: Quantum computing in a nutshell

**Didactic approach:** start from what computer scientists know, and *push out* to quantum.

**Classical** circuits

$\wedge \quad \vee \quad \neg$


[1] D. Aharonov, "A simple proof that toffoli and hadamard are quantum universal," *arXiv preprint quant-ph/0301040*, 2003.

[2] M.-O. Renou, D. Trillo, M. Weilenmann, T. P. Le, A. Tavakoli, et al., "Quantum theory based on real numbers can be experimentally falsified," *Nature*, vol. 600, 2021.

# Tutorial: Quantum computing in a nutshell

**Didactic approach:** start from what computer scientists know, and *push out* to quantum.

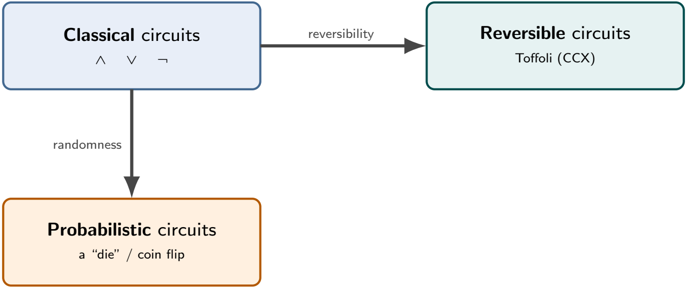


[1] D. Aharonov, "A simple proof that toffoli and hadamard are quantum universal," *arXiv preprint quant-ph/0301040*, 2003.

[2] M.-O. Renou, D. Trillo, M. Weilenmann, T. P. Le, A. Tavakoli, et al., "Quantum theory based on real numbers can be experimentally falsified," *Nature*, vol. 600, 2021.

# Tutorial: Quantum computing in a nutshell

**Didactic approach:** start from what computer scientists know, and *push out* to quantum.



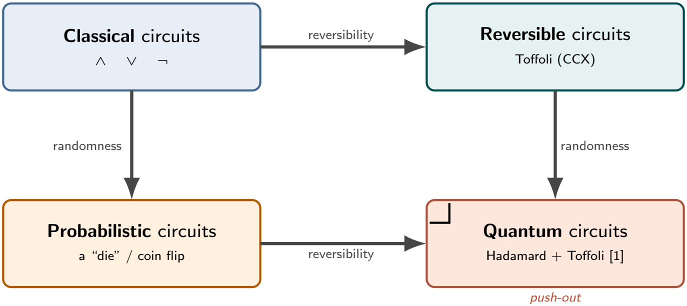


[1] D. Aharonov, "A simple proof that toffoli and hadamard are quantum universal," *arXiv preprint quant-ph/0301040*, 2003.

[2] M.-O. Renou, D. Trillo, M. Weilenmann, T. P. Le, A. Tavakoli, et al., "Quantum theory based on real numbers can be experimentally falsified," *Nature*, vol. 600, 2021.

# Tutorial: Quantum computing in a nutshell

**Didactic approach:** start from what computer scientists know, and *push out* to quantum.




**What we skip:** complex numbers [2] (we use reals.

[FM'24] `https://arxiv.org/abs/2407.11675`


[1] D. Aharonov, “A simple proof that toffoli and hadamard are quantum universal,” *arXiv preprint quant-ph/0301040*, 2003.

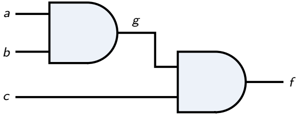
[2] M.-O. Renou, D. Trillo, M. Weilenmann, T. P. Le, A. Tavakoli, et al., “Quantum theory based on real numbers can be experimentally falsified,” *Nature*, vol. 600, 2021.

# Roadmap

- **Quantum computing in a nutshell**
  - Classical & reversible circuits
  - Randomness: the probabilistic die
  - Superposition: the reversible die (Hadamard)
  - Entanglement
  - Linear algebra

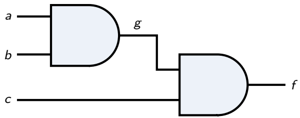
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):



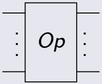
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):



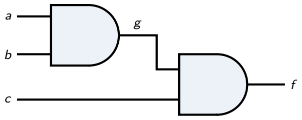

## Reversibility

Reversible gates implement a bijection
#outputs = #inputs



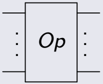
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):




### Reversibility

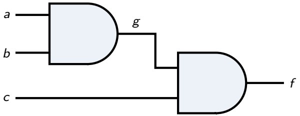
Reversible gates implement a bijection
$\#$outputs $=$ $\#$inputs




How to make an AND ($\wedge$) gate reversible?

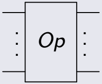
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):

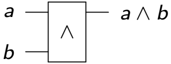



### Reversibility

Reversible gates implement a bijection
#outputs = #inputs

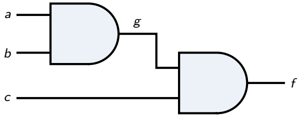



How to make an AND ($\wedge$) gate reversible?



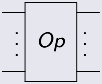
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):

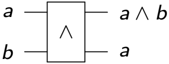



## Reversibility

Reversible gates implement a bijection
#outputs = #inputs

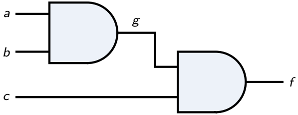



How to make an AND ($\wedge$) gate reversible?



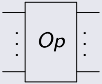
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):

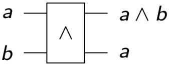


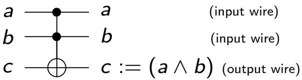

## Reversibility

Reversible gates implement a bijection
#outputs = #inputs



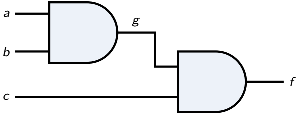

How to make an AND ($\wedge$) gate reversible?



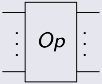
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):

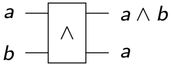


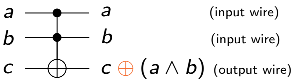

## Reversibility

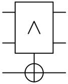
Reversible gates implement a bijection
#outputs = #inputs




How to make an AND ($\wedge$) gate reversible?



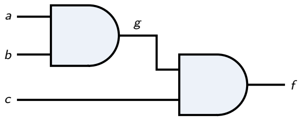




(A Toffoli gate)

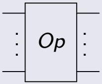
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):

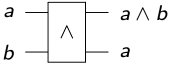


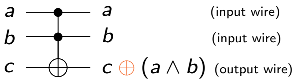

**Reversibility**

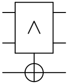
Reversible gates implement a bijection
#outputs = #inputs



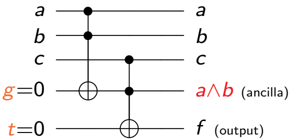

How to make an AND ($\wedge$) gate reversible?



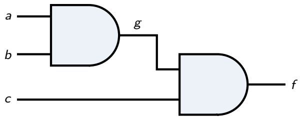

(A Toffoli gate)




Output $t = f$ — but ancilla $g$ holds garbage $a \wedge b$.

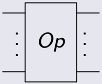
# From classical to reversible circuits

A classical circuit, $f = a \wedge b \wedge c$ (two ANDs — loses info):

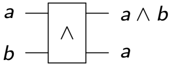


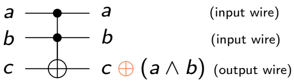

**Reversibility**

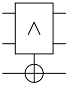
Reversible gates implement a bijection
#outputs = #inputs



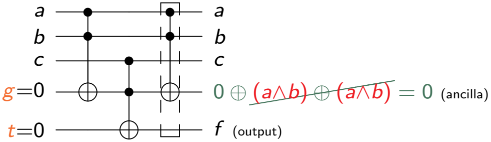

How to make an AND ($\wedge$) gate reversible?



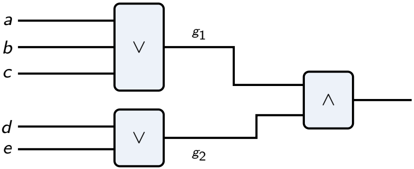

(A Toffoli gate)




Uncompute: repeat the first Toffoli $\Rightarrow$ ancilla $g$ reset to 0 (clean, reusable).

# Reversible Circuit Exercise

**Your turn.** Make the following classical circuit reversible:

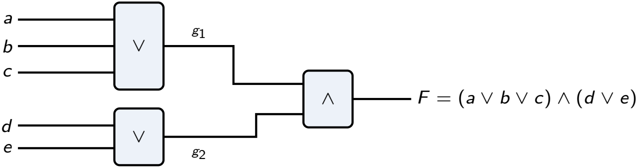



$F = (a \vee b \vee c) \wedge (d \vee e)$

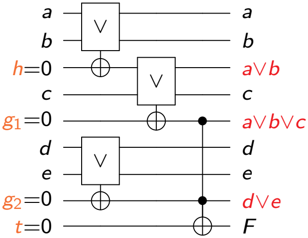
You may use the fact that NOT $-\oplus-$ is reversible: $-\oplus-\oplus- = ——$ .

# Reversible Circuit Exercise

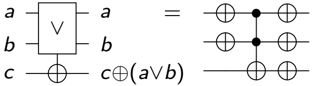
**Your turn.** Make the following classical circuit reversible:




**Solution A** — two ∨ gadgets: $h{=}a{\vee}b$, then $g_1{=}h{\vee}c$; then AND $\to t$:

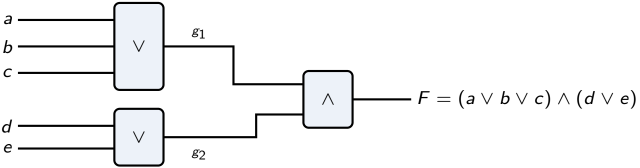



where

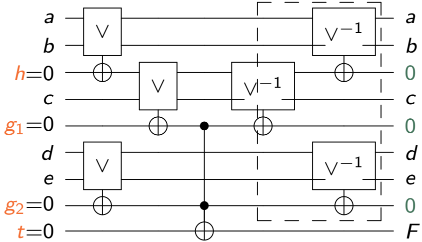


# Reversible Circuit Exercise

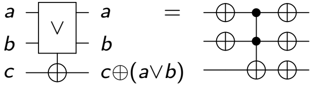
**Your turn.** Make the following classical circuit reversible:




**Solution B** — uncompute every ancilla (each $\vee^{-1}$) so all return to 0:

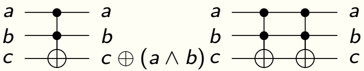



where



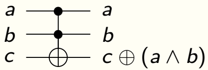
# Any classical circuit can be made reversible

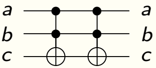
**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)



# Any classical circuit can be made reversible

**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)



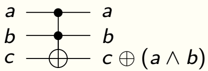



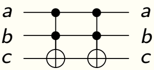

## Theorem (Toffoli is universal)

*Any boolean function can be implemented using only Toffoli gates & ancilla bits.*

## Any classical circuit can be made reversible

**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)



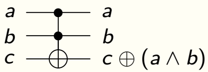

### Theorem (Toffoli is universal)

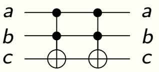
*Any boolean function can be implemented using only Toffoli gates & ancilla bits.*

### Proof.

# Any classical circuit can be made reversible

**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)




## Theorem (Toffoli is universal)

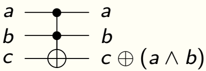
*Any boolean function can be implemented using only Toffoli gates & ancilla bits.*

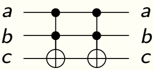
## Proof.

$\{NOT, AND\}$ is universal.

# Any classical circuit can be made reversible

**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)




## Theorem (Toffoli is universal)

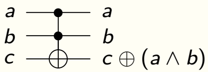
*Any boolean function can be implemented using only Toffoli gates & ancilla bits.*

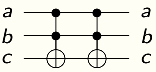
## Proof.

$\{NOT, AND\}$ is universal. Toffoli implements both using at most two ancillas:

# Any classical circuit can be made reversible

**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)



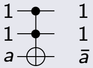

## Theorem (Toffoli is universal)

*Any boolean function can be implemented using only Toffoli gates & ancilla bits.*

## Proof.

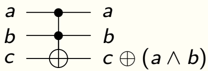
$\{NOT, AND\}$ is universal. Toffoli implements both using at most two ancillas:

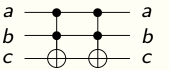
$NOT$:



# Any classical circuit can be made reversible

**Toffoli gate** is reversible:
(=controlled-controlled-NOT gate)



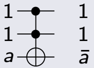

## Theorem (Toffoli is universal)

*Any boolean function can be implemented using only Toffoli gates & ancilla bits.*

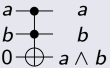
## Proof.

$\{NOT, AND\}$ is universal. Toffoli implements both using at most two ancillas:

$NOT$:

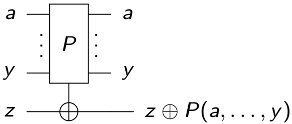



$AND$:

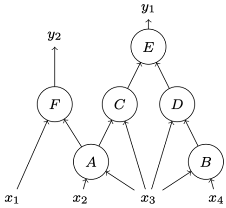


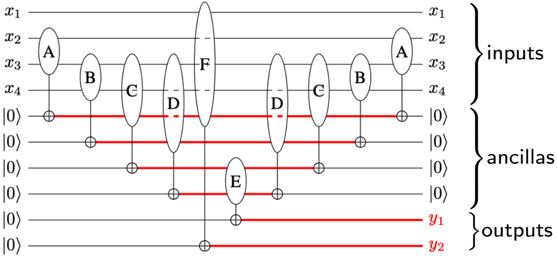

□

Note that we can generalize Toffoli's definition for any predicate $P$.

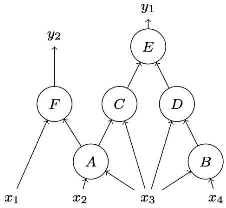


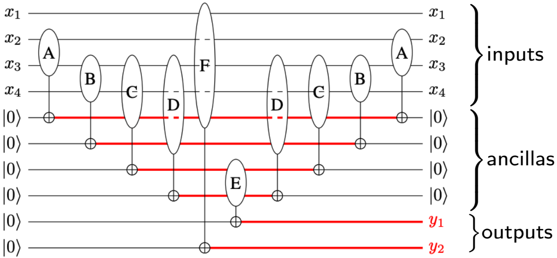
# Reversible space and time



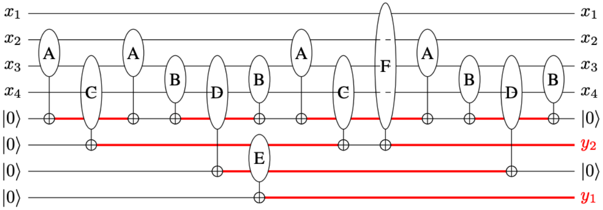



# Reversible space and time

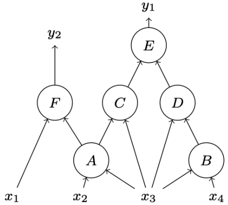


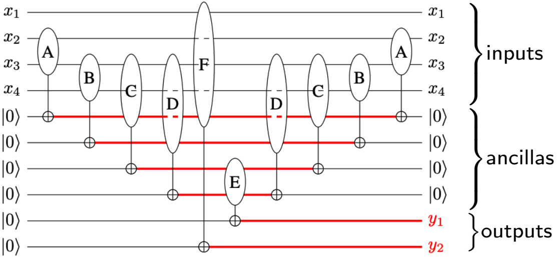




**Your turn.** Do we need one ancilla bit per gate? Expensive: Space = Time.

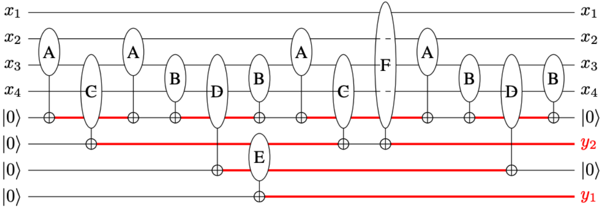


# Reversible space and time



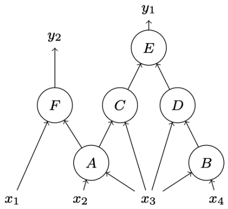



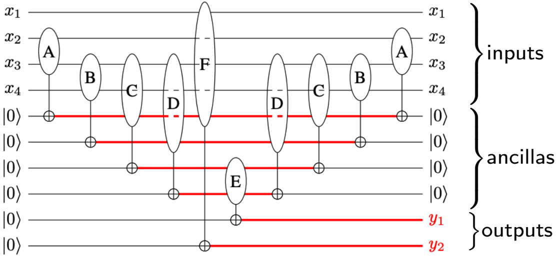

**Your turn.** Do we need one ancilla bit per gate? Expensive: Space = Time.



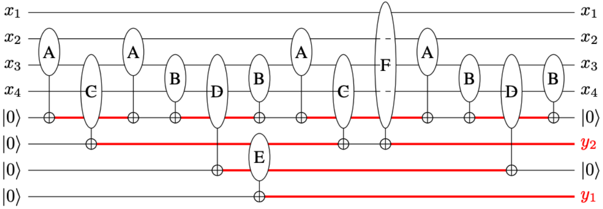

[Meuli et al. (2019) Reversible Pebbling Game for Quantum Memory Management]

# Reversible space and time




**Your turn.** Do we need one ancilla bit per gate? Expensive: Space = Time.

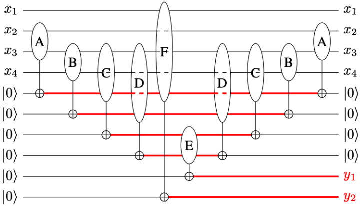



[Meuli et al. (2019) Reversible Pebbling Game for Quantum Memory Management]

[Lange, McKenzie, & Tapp (2000) Reversible space equals deterministic space]

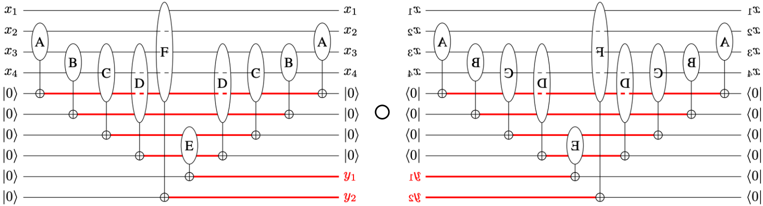
# Uncomputation?

**Your turn.** How to fully uncompute this circuit?



# Uncomputation?

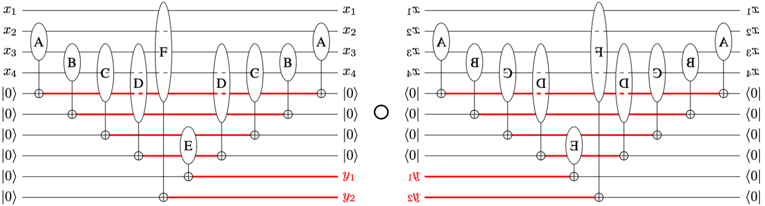
**Your turn.** How to fully uncompute this circuit?




Flip the circuit!

# Uncomputation?

**Your turn.** How to fully uncompute this circuit?




Flip the circuit!

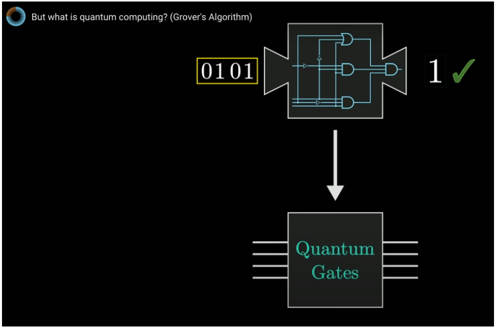
It also happens to be the case that in linear algebra $(AB \dots Z)^{-1} = Z^{-1} \dots B^{-1}A^{-1}$...

$$CCX^{-1} = CCX \qquad H^{-1} = H$$

# But how are reversible circuits related to quantum computing?

A reversible circuit **is** a quantum circuit!

A reversible circuit **is** the oracle in Grover's algorithm!




Recipe for the oracle implementation for the Grover search quantum algorithm:

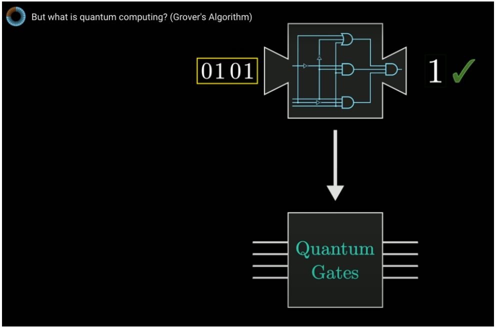
1. Take CNF formula $F$
2. Turn it into a reversible circuit $C_F$
3. Plug it into Grover's algorithm

https://www.3blue1brown.com/lessons/grover/

# But how are reversible circuits related to quantum computing?

A reversible circuit **is** a quantum circuit!

A reversible circuit **is** the oracle in Grover's algorithm!

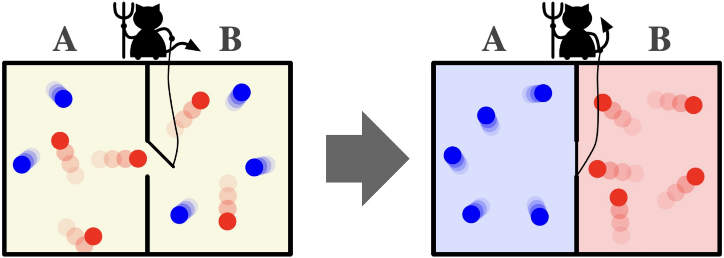



Recipe for the oracle implementation for the Grover search quantum algorithm:

1. Take CNF formula $F$
2. Turn it into a reversible circuit $C_F$
3. Plug it into Grover's algorithm
4. Only quadratic speedup over brute force!

`https://www.3blue1brown.com/lessons/grover/`

# Physics connection: Maxwell's daemon




Maxwell's demon opens the door only for fast molecules — sorting hot from cold, seemingly lowering entropy *for free*.

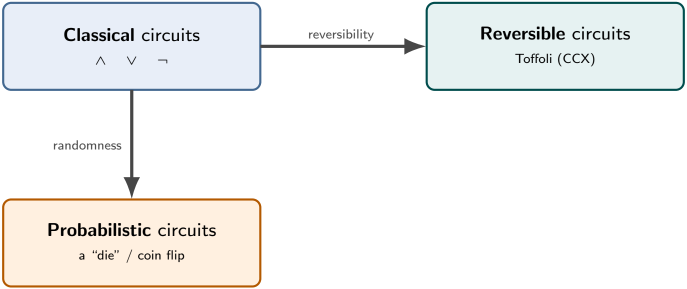
Landauer's principle: *erasing* one bit costs $\geq k_B T \ln 2$ of heat. The demon must erase its memory — the 2$^{\text{nd}}$ law survives.

So irreversible gates (AND, erase) must dissipate; reversible ones (Toffoli, uncompute) need not — computation can approach zero energy cost.

Quantum computing is reversible by necessity.

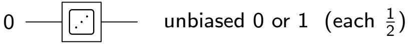
# Roadmap

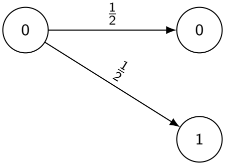
- **Quantum computing in a nutshell**
  - Classical & reversible circuits
  - Randomness: the probabilistic die
  - Superposition: the reversible die (Hadamard)
  - Entanglement
  - Linear algebra



# Adding a probabilistic gate to reversible circuits

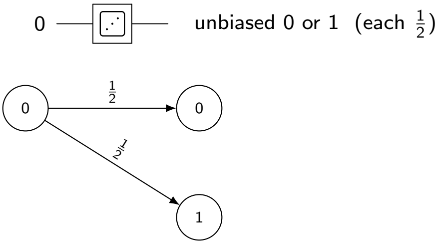
Now add one probabilistic gate — a fair die ⚂:



# Adding a probabilistic gate to reversible circuits

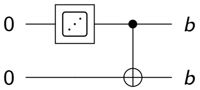
Now add one probabilistic gate — a fair die :

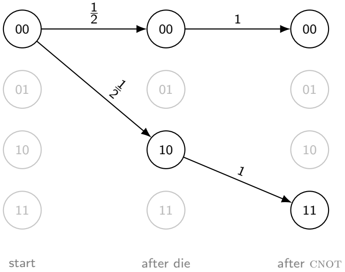



One fixed input, a random output — our new ingredient.

# This induces a distribution over bit strings

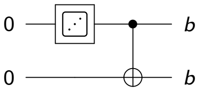
A classical “Bell pair”: roll the die, then copy it with a CNOT (controlled not).

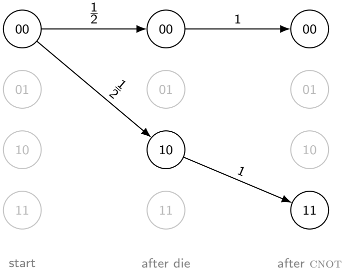


# This induces a distribution over bit strings

A classical "Bell pair": roll the die, then copy it with a CNOT (controlled not).



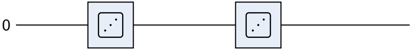



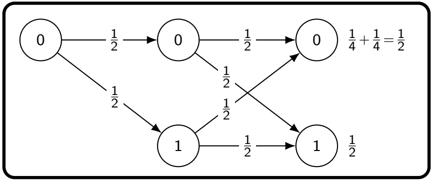

Perfectly correlated: $\frac{1}{2}(00) + \frac{1}{2}(11)$. $n$ random bits $\Rightarrow$ an exponential ($2^n$) vector:

$$\frac{1}{2}(00) + \frac{1}{2}(11) \; = \; \begin{array}{cl} \begin{bmatrix} 1/2 \\ 0 \\ 0 \\ 1/2 \end{bmatrix} & \begin{array}{l} 00 \\ 01 \\ 10 \\ 11 \end{array} \end{array}$$

# The die: does it matter when you open the box?

Two dice in a row. Watch the output distribution as we open the box at different points:

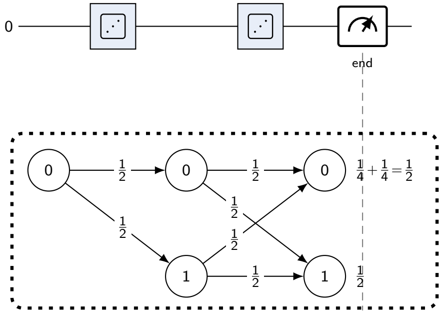



The box is closed — nothing measured yet.

# The die: does it matter when you open the box?

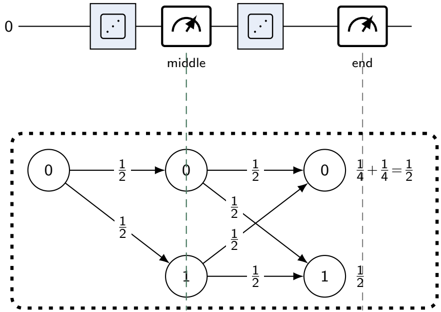
Two dice in a row. Watch the output distribution as we open the box at different points:




Open at the end: outcome 0/1 with $\frac{1}{2}(0) + \frac{1}{2}(1)$.

# The die: does it matter when you open the box?

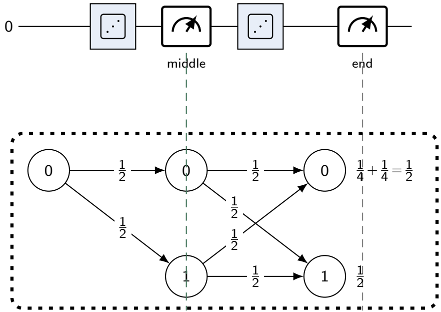
Two dice in a row. Watch the output distribution as we open the box at different points:




Open in the middle too: collapse to 0/1, then a fresh die — still $\frac{1}{2}(0) + \frac{1}{2}(1)$.

# The die: does it matter when you open the box?

Two dice in a row. Watch the output distribution as we open the box at different points:



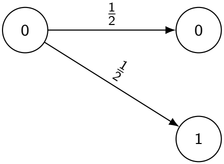

The distribution never changes: classical measurement is passive.

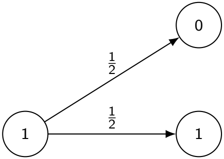
# Punchline: randomness costs reversibility

How to reverse?

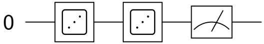
# Punchline: randomness costs reversibility

How to reverse?

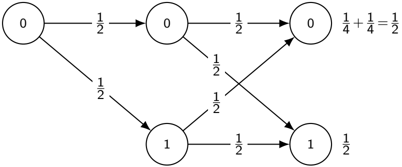



Rolling twice does not undo a roll — the result stays random:



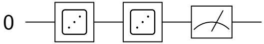

still $0/1$, each $\frac{1}{2}$



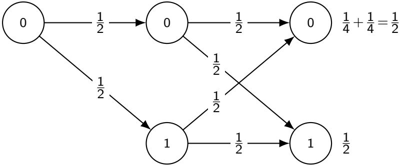
# Punchline: randomness costs reversibility

How to reverse?


Rolling twice does not undo a roll — the result stays random:




still $0/1$, each $\frac{1}{2}$

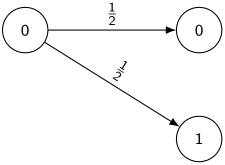


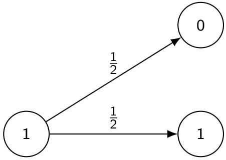

## Why?

A die is many-to-one (inputs 0 and 1 give the same outcomes) — it has no inverse.
Reversibility is (obviously) lost.

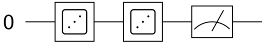
# Punchline: randomness costs reversibility

How to reverse?

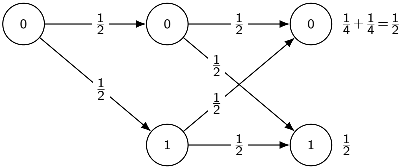



Rolling twice does not undo a roll — the result stays random:




still 0/1, each $\frac{1}{2}$



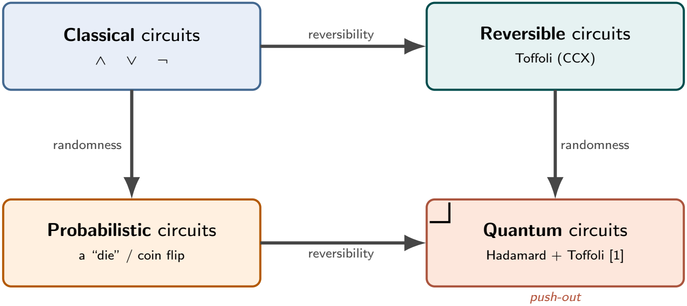

## Why?

A die is many-to-one (inputs 0 and 1 give the same outcomes) — it has no inverse.
Reversibility is (obviously) lost.

Can we make randomness *reversible* again?

# Roadmap

- **Quantum computing in a nutshell**
  - Classical & reversible circuits
  - Randomness: the probabilistic die
  - Superposition: the reversible die (Hadamard)
  - Entanglement
  - Linear algebra



# Making it quantum: the Hadamard is a reversible “die”

A Hadamard gate is like a reversible die:
- one roll always gives 0 or 1 with 50%.

$$|0\rangle \; -\!\boxed{H}\!- \; \text{unbiased } 0/1 \text{ (measurement)}$$

# Making it quantum: the Hadamard is a reversible "die"

A Hadamard gate is like a reversible die:
- one roll always gives 0 or 1 with 50%.

$$|0\rangle \;—\boxed{H}—\; \text{unbiased } 0/1 \text{ (measurement)}$$

- Two rolls maps $|0\rangle$ to $|0\rangle$ and $|1\rangle$ to $|1\rangle$ (identity).

$$|0\rangle \;—\boxed{H}—\boxed{H}—\; |0\rangle$$

$$|1\rangle \;—\boxed{H}—\boxed{H}—\; |1\rangle$$

# Making it quantum: the Hadamard is a reversible "die"

A Hadamard gate is like a reversible die:
- one roll always gives 0 or 1 with 50%.

$$|0\rangle \;-\boxed{H}-\; \text{unbiased } 0/1 \text{ (measurement)}$$

- Two rolls maps $|0\rangle$ to $|0\rangle$ and $|1\rangle$ to $|1\rangle$ (identity).

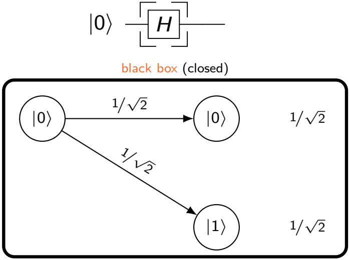
$$|0\rangle \;-\boxed{H}-\boxed{H}-\; |0\rangle$$

$$|1\rangle \;-\boxed{H}-\boxed{H}-\; |1\rangle$$

For this, we need "negative probabilities" (amplitudes).

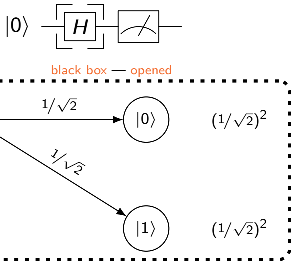
# Behind the Hadamard: the same box, new weights

The same automaton — but the weights are now amplitudes (which may be negative);



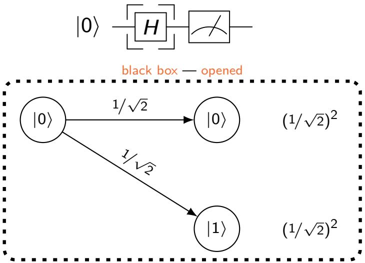
# Behind the Hadamard: the same box, new weights

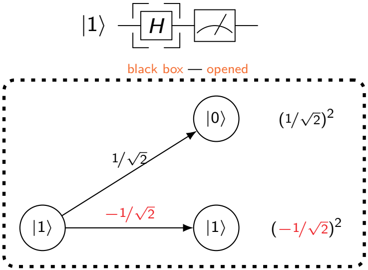
The same automaton — but the weights are now amplitudes (which may be negative); measuring squares them.



# Behind the Hadamard: the same box, new weights

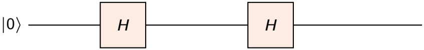
The same automaton — but the weights are now amplitudes (which may be negative); measuring squares them.

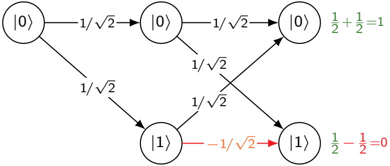


# Two Hadamards: now it does matter when you open the box

Amplitudes on the paths into each output add — and can cancel:

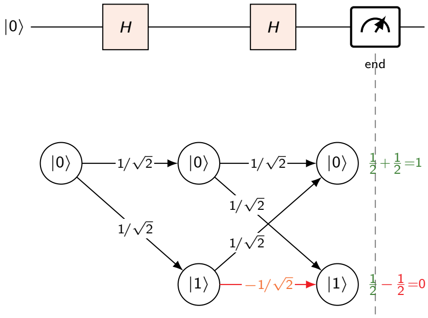



The box is closed.

# Two Hadamards: now it does matter when you open the box

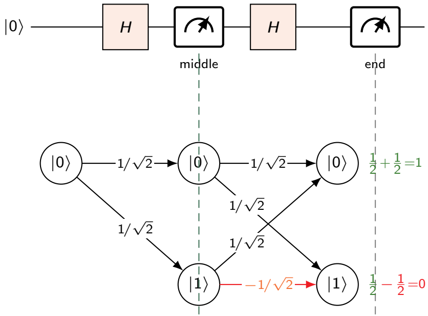
Amplitudes on the paths into each output add — and can cancel:




Open at the end: constructive to 0, destructive to 1 $\Rightarrow$ outcome 0 with certainty.

# Two Hadamards: now it does matter when you open the box

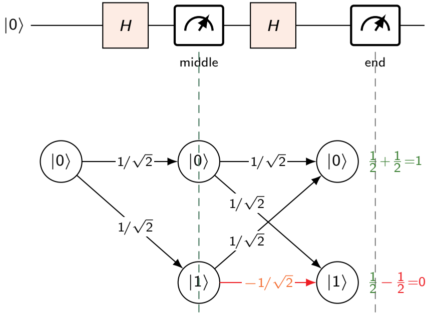
Amplitudes on the paths into each output add — and can cancel:




Open in the middle: collapse to 0/1, then a fresh $H \Rightarrow$ back to 50/50 — interference gone!

# Two Hadamards: now it does matter when you open the box

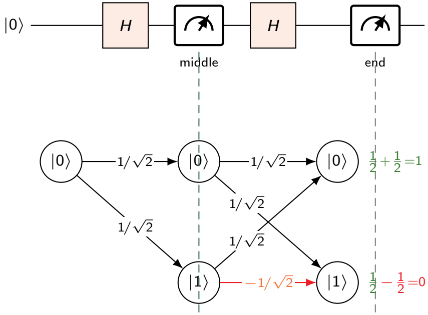
Amplitudes on the paths into each output add — and can cancel:




Quantum: it matters *when* you open the box.

# Two Hadamards: now it does matter when you open the box

Amplitudes on the paths into each output add — and can cancel:

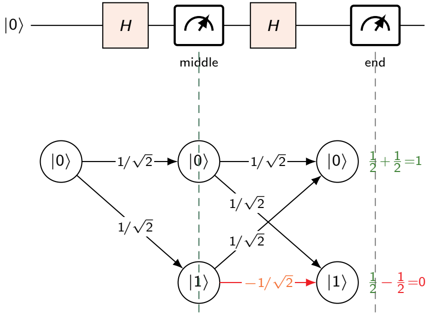



Quantum: it matters *when* you open the box.

**Contrary to the probabilistic case, the superposition 'seems real:' Nature allows the system to be in many states at once.**

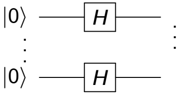
# Two Hadamards: now it does matter when you open the box

Amplitudes on the paths into each output add — and can cancel:




Quantum: it matters *when* you open the box.

**Contrary to the probabilistic case, the superposition 'seems real:' Nature allows the system to be in many states at once.**




$$\sum_{x \in \{0,1\}^1} \left(\frac{1}{\sqrt{2}}\right)^n |x\rangle$$

## Superposition

### Takeaway

Now you understand superpositions. This is what distinguishes quantum from probabilistic systems.
It follows from the negative amplitudes; orthogonality in a real vector space:
$\begin{bmatrix}1 & 1\end{bmatrix} \cdot \begin{bmatrix}1 \\ -1\end{bmatrix} = 0$
We will see that superpositions also lead entanglement of qubits / particles.

# Superposition

## Takeaway

Now you understand superpositions. This is what distinguishes quantum from probabilistic systems.
It follows from the negative amplitudes; orthogonality in a real vector space:

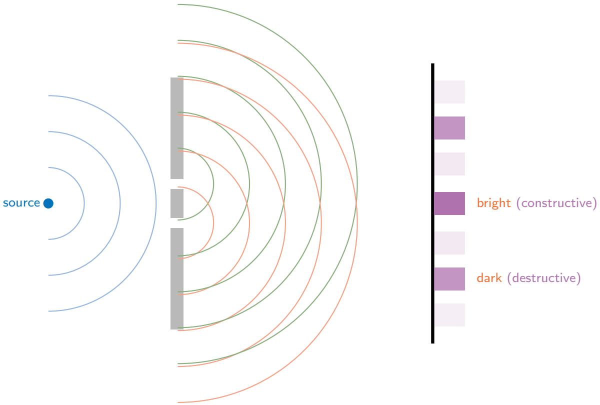
$$\begin{bmatrix} 1 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix} = 0$$

We will see that superpositions also lead entanglement of qubits / particles.

Now you can join the more philosophical discussion about interpretations:

| Interpretation | Core idea | |
|---|---|---|
| Copenhagen | Wavefunction encodes probabilities; measurement yields a definite outcome. | |
| Many-Worlds (Everett) | Wavefunction never collapses. | All possible outcomes occur in branching |
| de Broglie–Bohm (Pilot Wave) | Particles have definite positions guided by a wave. | |
| Objective Collapse | Wavefunction collapse is a real physical process. | |
| QBism | Quantum states represent an agent's personal beliefs. | |
| Relational QM | Properties exist only relative to other systems. | |
| Consistent Histories | Quantum mechanics assigns probabilities to entire histories. | |
| Ensemble (Statistical) | Wavefunction describes an ensemble, not individual systems. | |
| Transactional | Quantum events arise from advanced and retarded wave exchanges. | |

# Physics connection: the double-slit experiment



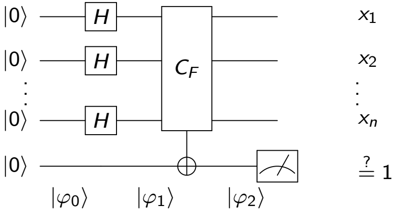

Particles behaves like waves; the slits split the wave source; on the screen the waves add or subtract:

- bright: $+\frac{1}{2} + \frac{1}{2}$ (reinforce)
- dark: $+\frac{1}{2} - \frac{1}{2} = 0$ (cancel)

A *single* particle interferes with itself — our negative weights made visible.

# Bringing it all together: can a quantum computer solve SAT?

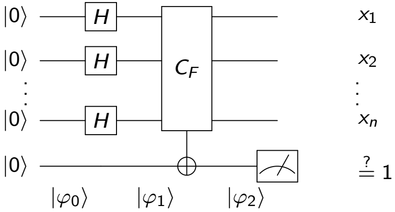
Let $F$ be a CNF formula over variables $x_1, x_2, \ldots, x_n$ and $C_F$ be its reversible circuit (omitting ancillas):




$$|\varphi_0\rangle = |0\rangle^{\otimes n} |0\rangle \quad \xrightarrow{H} \quad |\varphi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} |\vec{x}\rangle\, |0\rangle \quad \xrightarrow{C_F} \quad |\varphi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} |\vec{x}\rangle\, |F(\vec{x})\rangle$$

# Bringing it all together: can a quantum computer solve SAT?

Let $F$ be a CNF formula over variables $x_1, x_2, \ldots, x_n$ and $C_F$ be its reversible circuit (omitting ancillas):

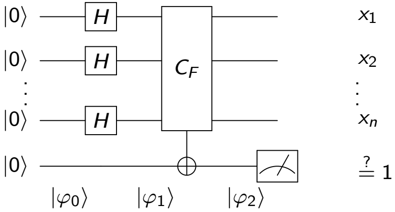



$$|\varphi_0\rangle = |0\rangle^{\otimes n}|0\rangle \quad \xrightarrow{H} \quad |\varphi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{\vec{x}}|\vec{x}\rangle|0\rangle \quad \xrightarrow{C_F} \quad |\varphi_2\rangle = \frac{1}{\sqrt{2^n}}\sum_{\vec{x}}|\vec{x}\rangle|F(\vec{x})\rangle$$

Post-selecting output $= 1$ on $|\varphi_2\rangle$ leaves a uniform superposition over the satisfying assignments — measure and read off a solution!

# Bringing it all together: can a quantum computer solve SAT?

Let $F$ be a CNF formula over variables $x_1, x_2, \ldots, x_n$ and $C_F$ be its reversible circuit (omitting ancillas):



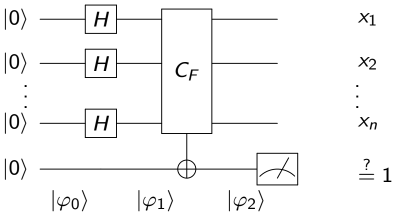

$$|\varphi_0\rangle = |0\rangle^{\otimes n}|0\rangle \quad \xrightarrow{H} \quad |\varphi_1\rangle = \frac{1}{\sqrt{2^n}}\sum_{\vec{x}} |\vec{x}\rangle\,|0\rangle \quad \xrightarrow{C_F} \quad |\varphi_2\rangle = \frac{1}{\sqrt{2^n}}\sum_{\vec{x}} |\vec{x}\rangle\,|F(\vec{x})\rangle$$

Post-selecting output $= 1$ on $|\varphi_2\rangle$ leaves a uniform superposition over the satisfying assignments — measure and read off a solution!

**Show of hands — does this solve SAT?**

# Bringing it all together: can a quantum computer solve SAT?

Let $F$ be a CNF formula over variables $x_1, x_2, \ldots, x_n$ and $C_F$ be its reversible circuit (omitting ancillas):




$$|\varphi_0\rangle = |0\rangle^{\otimes n} |0\rangle \quad \xrightarrow{H} \quad |\varphi_1\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} |\vec{x}\rangle\, |0\rangle \quad \xrightarrow{C_F} \quad |\varphi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{\vec{x}} |\vec{x}\rangle\, |F(\vec{x})\rangle$$

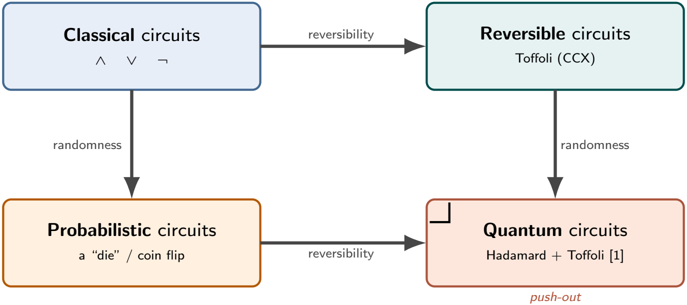
Post-selecting output $= 1$ on $|\varphi_2\rangle$ leaves a uniform superposition over the satisfying assignments — measure and read off a solution!

**Show of hands — does this solve SAT?**

Not for free: $\Pr[\text{out} = 1] = \dfrac{\#\text{SAT}(F)}{2^n}$ is often exponentially small $\Rightarrow$ exponentially many tries.
(If post-selection *were* free: PostBQP $=$ PP $\supseteq$ NP [Aaronson].)

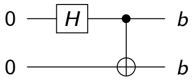
# Roadmap

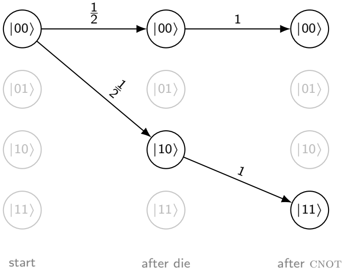
- **Quantum computing in a nutshell**
  - Classical & reversible circuits
  - Randomness: the probabilistic die
  - Superposition: the reversible die (Hadamard)
  - Entanglement
  - Linear algebra



# Entanglement

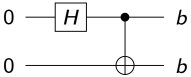
A classical "Bell pair": roll the die, then copy it with a CNOT (controlled not).

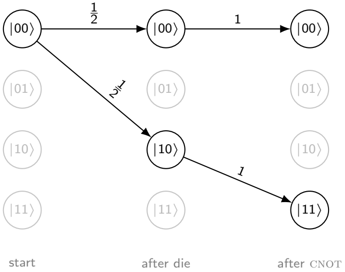


# Entanglement

A classical "Bell pair": roll the die, then copy it with a CNOT (controlled not).



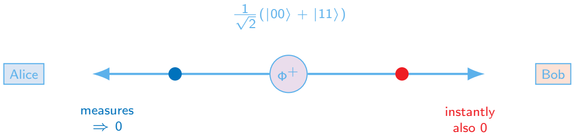




Perfectly correlated: $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$.

$n$ random bits $\Rightarrow$ an exponential ($2^n$) vector:

$$\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle = \begin{array}{cc} \begin{bmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{bmatrix} & \begin{array}{c} 00 \\ 01 \\ 10 \\ 11 \end{array} \end{array}$$

# Physics connection: EPR and "spooky action at a distance"

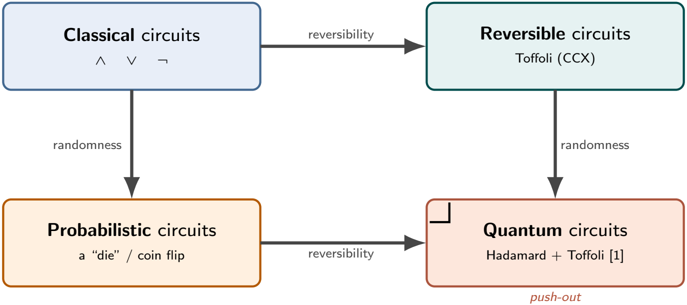



Measuring one half instantly fixes the other — however far apart. Einstein's "spukhafte Fernwirkung" (EPR, 1935).

But no signal travels: the outcomes are correlated, not controllable. Entanglement is a resource, not a telephone.

# Roadmap

- **Quantum computing in a nutshell**
  - Classical & reversible circuits
  - Randomness: the probabilistic die
  - Superposition: the reversible die (Hadamard)
  - Entanglement
  - Linear algebra



# Bra–ket / Dirac Notation

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

# Bra–ket / Dirac Notation

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

$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

# Bra–ket / Dirac Notation

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

$$|00\rangle = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \qquad |01\rangle = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \qquad |10\rangle = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} \qquad |11\rangle = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

$$|000\rangle = \underbrace{\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}}_{|0\rangle|0\rangle|0\rangle} \qquad |001\rangle = \underbrace{\begin{bmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix}}_{|0\rangle|0\rangle|1\rangle} \qquad \ldots \qquad |111\rangle = \underbrace{\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{bmatrix}}_{|1\rangle|1\rangle|1\rangle}$$

$\ddots$

## Bra–ket / Dirac Notation

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

$$|00\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \qquad |01\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \qquad |10\rangle = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \qquad |11\rangle = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$|000\rangle = \underbrace{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{|0\rangle|0\rangle|0\rangle} \qquad |001\rangle = \underbrace{\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}_{|0\rangle|0\rangle|1\rangle} \qquad \ldots \qquad |111\rangle = \underbrace{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}}_{|1\rangle|1\rangle|1\rangle}$$

$$\ddots$$

$$\langle 000| = |000\rangle^\dagger = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\langle 0|\langle 0|\langle 0|}$$

(conjugate & transpose)

# Bra–ket / Dirac Notation

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

$$|00\rangle = \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \qquad |01\rangle = \begin{bmatrix}0\\1\\0\\0\end{bmatrix} \qquad |10\rangle = \begin{bmatrix}0\\0\\1\\0\end{bmatrix} \qquad |11\rangle = \begin{bmatrix}0\\0\\0\\1\end{bmatrix}$$

$$|000\rangle = \underbrace{\begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}}_{|0\rangle|0\rangle|0\rangle} \qquad |001\rangle = \underbrace{\begin{bmatrix}0\\1\\0\\0\\0\\0\\0\\0\end{bmatrix}}_{|0\rangle|0\rangle|1\rangle} \qquad \dots \qquad |111\rangle = \underbrace{\begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\1\end{bmatrix}}_{|1\rangle|1\rangle|1\rangle}$$

$$\ddots$$

$$\langle 000| = |000\rangle^\dagger = \underbrace{\begin{bmatrix}1&0&0&0&0&0&0&0\end{bmatrix}}_{\langle 0|\langle 0|\langle 0|}$$ (conjugate & transpose)

Tricks:

$\langle i| M |j\rangle = M_{i,j}$ (matrix entry $i, j$)

$\langle i| |j\rangle = \delta_{i,j}$ (1 iff $i = j$)

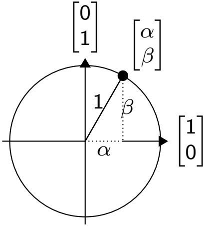
$|\varphi\rangle \langle\varphi|$ (density matrix/operator of the pure state $|\varphi\rangle$)

# Qubits, more generally (I)

A one-qubit quantum state is a vector $\vec{s} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix} \in \mathbb{R}^2$ (or more generally $\in \mathbb{C}^2$) on the unit circle (i.e. $\alpha^2 + \beta^2 = 1$ ):

$\vec{s} = \alpha\,|0\rangle + \beta\,|1\rangle$

$\vec{s} = \alpha \begin{bmatrix}1 \\ 0\end{bmatrix} + \beta \begin{bmatrix}0 \\ 1\end{bmatrix} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$

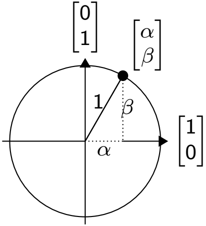


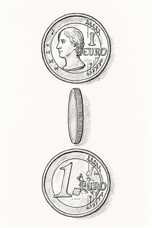
# Qubits, more generally (I)

A one-qubit quantum state is a vector $\vec{s} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \in \mathbb{R}^2$ (or more generally $\in \mathbb{C}^2$) on the unit circle (i.e. $\alpha^2 + \beta^2 = 1$ ):

$\vec{s} = \alpha |0\rangle + \beta |1\rangle$

$\vec{s} = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$



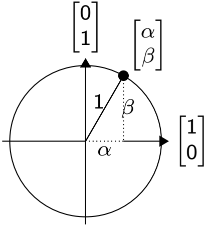



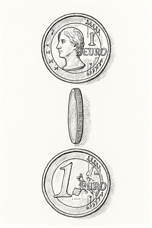
# Qubits, more generally (I)

A one-qubit quantum state is a vector $\vec{s} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix} \in \mathbb{R}^2$ (or more generally $\in \mathbb{C}^2$) on the unit circle (i.e. $\alpha^2 + \beta^2 = 1$ ):

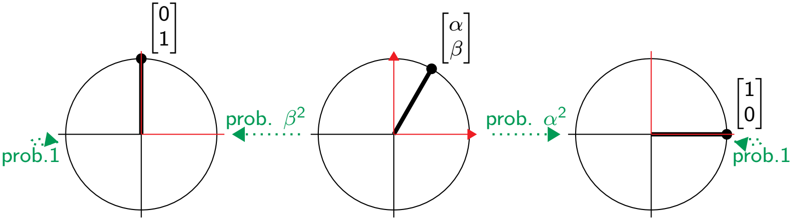
$\vec{s} = \alpha\,|0\rangle + \beta\,|1\rangle$

$\vec{s} = \alpha \begin{bmatrix}1 \\ 0\end{bmatrix} + \beta \begin{bmatrix}0 \\ 1\end{bmatrix} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$




A measurement 'collapses' the state:

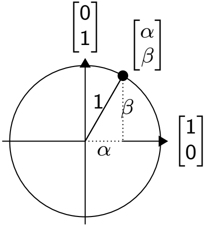


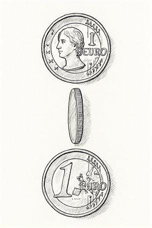
# Qubits, more generally (I)

A one-qubit quantum state is a vector $\vec{s} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix} \in \mathbb{R}^2$ (or more generally $\in \mathbb{C}^2$) on the unit circle (i.e. $\alpha^2 + \beta^2 = 1$ ):

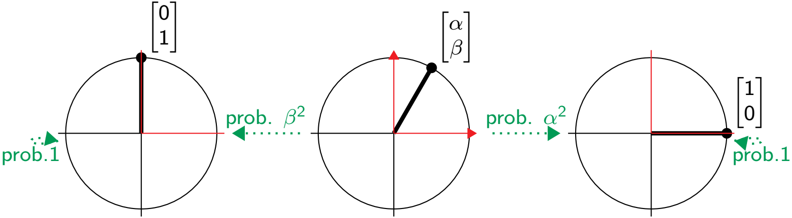
$\vec{s} = \alpha\,|0\rangle + \beta\,|1\rangle$

$\vec{s} = \alpha \begin{bmatrix}1 \\ 0\end{bmatrix} + \beta \begin{bmatrix}0 \\ 1\end{bmatrix} = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$

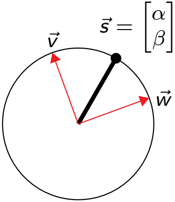


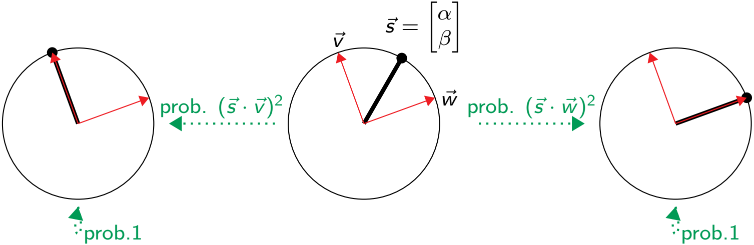
A computational basis measurement 'collapses' the state:



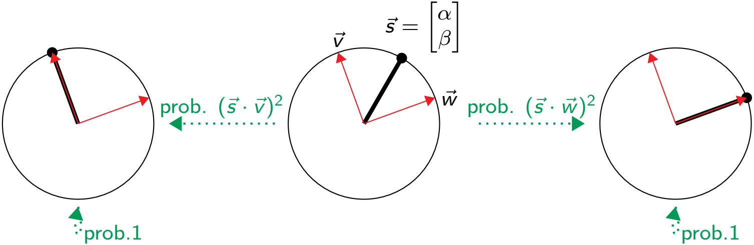
# Qubits, more generally (I)



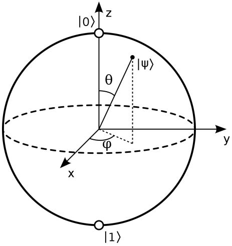
# Qubits, more generally (I)



# Qubits, more generally (I)

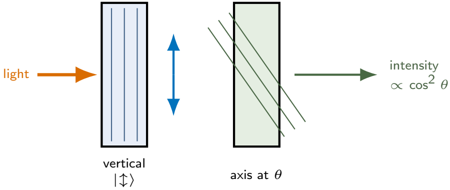



With complex numbers, we get a Bloch sphere:




$|\vec{s} \cdot \vec{w}|^2$

# Physics connection: polarizing filters and the Born rule



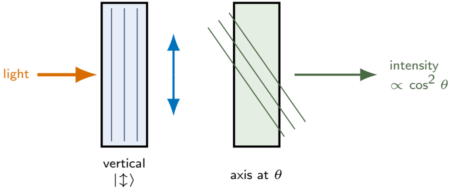

A filter projects the polarization onto its axis:

$$\text{amplitude} = \cos\theta, \qquad \text{prob} = \cos^2\theta.$$

The squaring is the Born rule: prob $= |\text{amplitude}|^2$.

A filter is a measurement in a rotated basis — $|\updownarrow\rangle$ read along $\theta$: $|\langle\theta|\updownarrow\rangle|^2 = \cos^2\theta$.

# Physics connection: polarizing filters and the Born rule




A filter projects the polarization onto its axis:

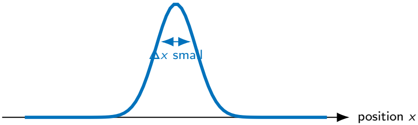
$$\text{amplitude} = \cos\theta, \qquad \text{prob} = \cos^2\theta.$$

The squaring is the Born rule: prob $= |\text{amplitude}|^2$.

A filter is a measurement in a rotated basis — $|\updownarrow\rangle$ read along $\theta$: $|\langle\theta|\updownarrow\rangle|^2 = \cos^2\theta$.

Three filters $0^\circ - 45^\circ - 90^\circ$: inserting the middle one lets light through two crossed filters — measurement changes the state!

# Physics connection: Heisenberg's uncertainty principle





$$\Delta x \cdot \Delta p \ \geq \ \frac{\hbar}{2}$$

Pin down position and momentum spreads out — you cannot sharply know both.

A qubit tells the same story: a measurement in the computational basis $(|0\rangle, |1\rangle)$ leaves the conjugate basis $(|+\rangle, |-\rangle)$ maximally uncertain.

Measurement collapses one observable — at the cost of the other.

# Many qubits: an amplitude distribution over bit strings

One qubit: $\vec{s} = \alpha \left|0\right\rangle + \beta \left|1\right\rangle$, a vector on the unit circle ($\alpha^2 + \beta^2 = 1$).

# Many qubits: an amplitude distribution over bit strings

One qubit: $\vec{s} = \alpha\,|0\rangle + \beta\,|1\rangle$, a vector on the unit circle ($\alpha^2 + \beta^2 = 1$).

*n* qubits: an amplitude for *every* bit string $\vec{b} \in \{\,0, 1\,\}^n$:

$$|\varphi\rangle \;=\; \sum_{\vec{b} \in \{\,0,1\,\}^n} \alpha_{\vec{b}}\,|\vec{b}\rangle \;=\; \begin{bmatrix} \alpha_{00\ldots0} \\ \alpha_{00\ldots1} \\ \vdots \\ \alpha_{11\ldots1} \end{bmatrix}, \qquad \sum_{\vec{b}} \alpha_{\vec{b}}^2 = 1.$$

# Many qubits: an amplitude distribution over bit strings

One qubit: $\vec{s} = \alpha\,|0\rangle + \beta\,|1\rangle$, a vector on the unit circle ($\alpha^2 + \beta^2 = 1$).

$n$ qubits: an amplitude for *every* bit string $\vec{b} \in \{\,0, 1\,\}^n$:

$$|\varphi\rangle \;=\; \sum_{\vec{b} \in \{\,0,1\,\}^n} \alpha_{\vec{b}}\,|\vec{b}\rangle \;=\; \begin{bmatrix} \alpha_{00\ldots0} \\ \alpha_{00\ldots1} \\ \vdots \\ \alpha_{11\ldots1} \end{bmatrix}, \qquad \sum_{\vec{b}} \alpha_{\vec{b}}^2 = 1.$$

- A superposition is exactly such a vector — like a probability distribution, but amplitudes may be negative (in general: complex).
- $2^n$ amplitudes for $n$ qubits: the exponential cost Feynman warned about.
- Measuring all qubits returns $\vec{b}$ with probability $|\alpha_{\vec{b}}|^2$ (Born rule).

# Many qubits: an amplitude distribution over bit strings

One qubit: $\vec{s} = \alpha\,|0\rangle + \beta\,|1\rangle$, a vector on the unit circle ($\alpha^2 + \beta^2 = 1$).

$n$ qubits: an amplitude for *every* bit string $\vec{b} \in \{\,0,1\,\}^n$:

$$|\varphi\rangle \;=\; \sum_{\vec{b}\in\{\,0,1\,\}^n} \alpha_{\vec{b}}\,|\vec{b}\rangle \;=\; \begin{bmatrix} \alpha_{00\ldots0} \\ \alpha_{00\ldots1} \\ \vdots \\ \alpha_{11\ldots1} \end{bmatrix}, \qquad \sum_{\vec{b}} \alpha_{\vec{b}}^2 = 1.$$

- A superposition is exactly such a vector — like a probability distribution, but amplitudes may be negative (in general: complex).
- $2^n$ amplitudes for $n$ qubits: the exponential cost Feynman warned about.
- Measuring all qubits returns $\vec{b}$ with probability $|\alpha_{\vec{b}}|^2$ (Born rule).

This is the whole state space — and gates are just linear maps on these vectors.

# From circuits to linear algebra (1): a gate is a matrix

A state is a basis vector (one entry $= 1$):

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

# From circuits to linear algebra (1): a gate is a matrix

A state is a basis vector (one entry $= 1$):

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

A reversible gate is a permutation matrix. NOT ($X$) swaps $|0\rangle \leftrightarrow |1\rangle$:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

# From circuits to linear algebra (1): a gate is a matrix

A state is a basis vector (one entry $= 1$):

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

one-bit circuit:

$$|0\rangle \;\; -\!\!\oplus\!\!- \quad |1\rangle$$

A reversible gate is a permutation matrix. NOT ($X$) swaps $|0\rangle \leftrightarrow |1\rangle$:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Running the gate = matrix $\times$ vector:

$$X\,|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle\,.$$

# From circuits to linear algebra (1): a gate is a matrix

A state is a basis vector (one entry $= 1$):

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

A reversible gate is a permutation matrix. NOT ($X$) swaps $|0\rangle \leftrightarrow |1\rangle$:

$$X = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}.$$

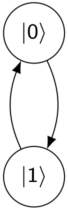
Running the gate $=$ matrix $\times$ vector:

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

one-bit circuit:

$$|0\rangle \;\; -\!\oplus\!- \;\; |1\rangle$$

the matrix is an automaton (which state goes where):

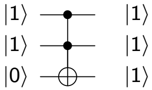



column $|b\rangle$ of $X$ $=$ where $|b\rangle$ goes.

# From circuits to linear algebra (2): bigger gate, bigger matrix

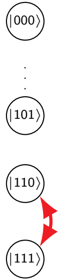
$n$ qubits $\Rightarrow 2^n$ basis states. Toffoli flips bit 3 only when bits $1, 2$ are both 1: it swaps $|110\rangle \leftrightarrow |111\rangle$ and fixes the other six.

three-bit circuit:




$CCX\,|110\rangle = |111\rangle$

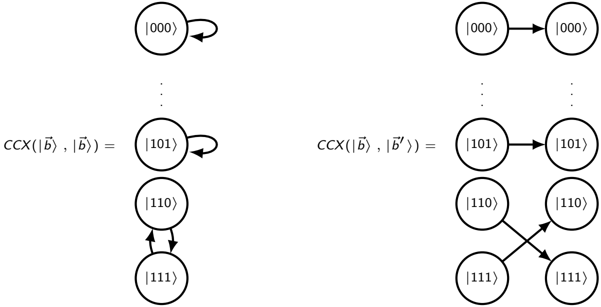
$$CCX = \begin{bmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&1\\ 0&0&0&0&0&0&1&0 \end{bmatrix}$$




the other six
map to themselves

# The Toffoli permutation, as an automaton

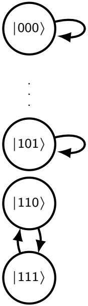
TOFFOLI only swaps $|110\rangle \leftrightarrow |111\rangle$ and fixes the other six. The *same* permutation, drawn two ways:



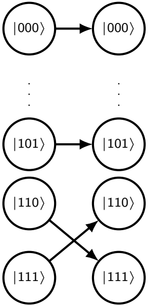
# The Toffoli permutation, as an automaton

TOFFOLI only swaps $|110\rangle \leftrightarrow |111\rangle$ and fixes the other six. The *same* permutation, drawn two ways:

$CCX(|\vec{b}\rangle\ ,\ |\vec{b}\rangle) =$



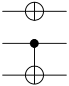

$CCX(|\vec{b}\rangle\ ,\ |\vec{b}'\rangle) =$

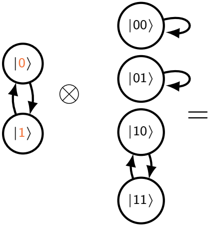



self-loops $CCX(|\vec{b}\rangle\ ,\ |\vec{b}\rangle)$ read off tensor products; transitions $CCX(|\vec{b}\rangle\ ,\ |\vec{b}'\rangle)$ read off matrix products.

# Wires side by side = tensor product: $X \otimes CX$

Top wire $X$, bottom two wires $CX$ act independently, so the gate is $X \otimes CX$. Its automaton is the product of the two:

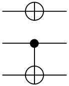


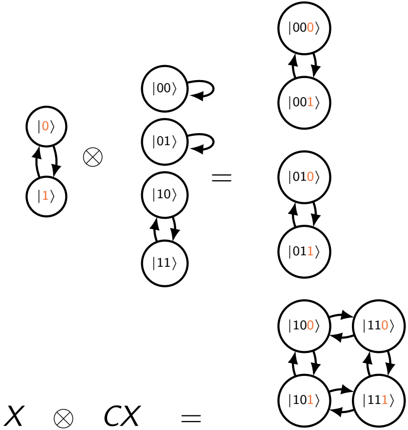



$$X \otimes CX =$$

# Wires side by side = tensor product: $X \otimes CX$

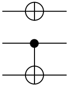
Top wire $X$, bottom two wires $CX$ act independently, so the gate is $X \otimes CX$. Its automaton is the product of the two:

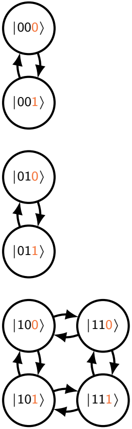




# Wires side by side = tensor product: $X \otimes CX$

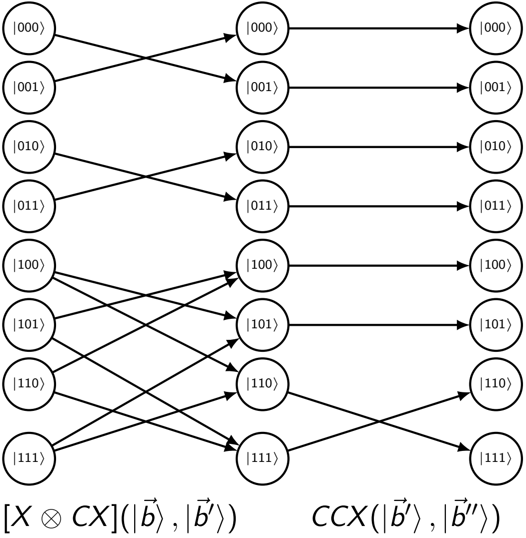
Top wire $X$, bottom two wires $CX$ act independently, so the gate is $X \otimes CX$. Its automaton is the product of the two:





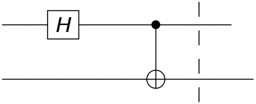
# Gates in sequence = matrix product: $CCX \times (X \otimes CX)$

Do $X \otimes CX$ first, then $CCX$. Follow the arrows: state $\vec{b} \rightarrow \vec{b}' \rightarrow \vec{b}''$. Circuit order $\Rightarrow$ matrix order is reversed: $CCX \times (X \otimes CX)$.



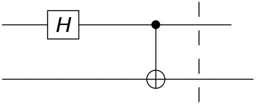
# Entanglement: The Bell state / EPR pair

$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$



# Entanglement: The Bell state / EPR pair

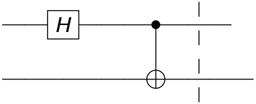
$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$




$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot H \otimes I \cdot \begin{bmatrix}1\\0\\0\\0\end{bmatrix} \;=$$

# Entanglement: The Bell state / EPR pair

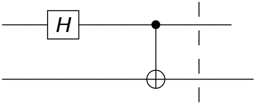
$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$




$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot H \otimes I \cdot \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}} \cdot \underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}} =$$

# Entanglement: The Bell state / EPR pair

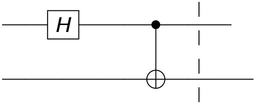
$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$




$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot H \otimes I \cdot \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}} \cdot \underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}} = \frac{1}{\sqrt{2}} \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

# Entanglement: The Bell state / EPR pair

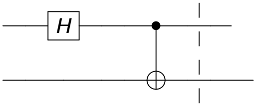
$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$




$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot H \otimes I \cdot \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}} \cdot \underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}} = \frac{1}{\sqrt{2}} \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

# Entanglement: The Bell state / EPR pair

$$\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$

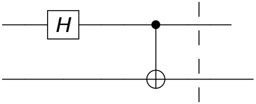



$$\frac{1}{\sqrt{2}}\,|00\rangle+\frac{1}{\sqrt{2}}\,|11\rangle=\frac{1}{\sqrt{2}}\cdot\begin{bmatrix}1\\0\\0\\1\end{bmatrix}\begin{matrix}=00\\ \\ \\ =11\end{matrix}$$

$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot H\otimes I\cdot\begin{bmatrix}1\\0\\0\\0\end{bmatrix}=\frac{1}{\sqrt{2}}\cdot\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot\underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}}\cdot\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}}=\frac{1}{\sqrt{2}}\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot\begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

# Entanglement: The Bell state / EPR pair

$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$

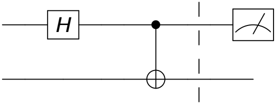



$$\frac{1}{\sqrt{2}}\left|00\right\rangle + \frac{1}{\sqrt{2}}\left|11\right\rangle = \frac{1}{\sqrt{2}} \cdot \begin{bmatrix}1\\0\\0\\1\end{bmatrix} \begin{array}{l} = 00 \\ \\ \\ = 11 \end{array}$$

$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot H \otimes I \cdot \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}} \cdot \underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}} = \frac{1}{\sqrt{2}} \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

# Entanglement: The Bell state / EPR pair

$$\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$

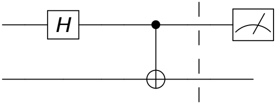



$$\frac{1}{\sqrt{2}}\left|00\right\rangle+\frac{1}{\sqrt{2}}\left|11\right\rangle=\frac{1}{\sqrt{2}}\cdot\begin{bmatrix}1\\0\\0\\1\end{bmatrix}\begin{matrix}=00\\ \\ \\ =11\end{matrix}$$

$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot H\otimes I\cdot\begin{bmatrix}1\\0\\0\\0\end{bmatrix}=\frac{1}{\sqrt{2}}\cdot\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot\underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}}\cdot\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}}=\frac{1}{\sqrt{2}}\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot\begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

# Entanglement: The Bell state / EPR pair

$$\begin{bmatrix}1\\0\end{bmatrix}\otimes\begin{bmatrix}1\\0\end{bmatrix}=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$



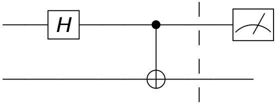

collapses to $\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = 00$ *or* $\begin{bmatrix}0\\0\\0\\1\end{bmatrix} = 11$

$$\frac{1}{\sqrt{2}}\left|00\right\rangle + \frac{1}{\sqrt{2}}\left|11\right\rangle = \frac{1}{\sqrt{2}}\cdot\begin{bmatrix}1\\0\\0\\1\end{bmatrix}\begin{matrix}=00\\ \\ \\ =11\end{matrix}$$

$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot H\otimes I\cdot\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}}\cdot\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot\underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}}\cdot\underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}} = \frac{1}{\sqrt{2}}\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT}\cdot\begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

# Entanglement: The Bell state / EPR pair

$$\begin{bmatrix}1\\0\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}1\\0\\0\\0\end{bmatrix}$$




collapses to $\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = 00$ *or* $\begin{bmatrix}0\\0\\0\\1\end{bmatrix} = 11$

$$\frac{1}{\sqrt{2}}\,|00\rangle + \frac{1}{\sqrt{2}}\,|11\rangle = \frac{1}{\sqrt{2}} \cdot \begin{bmatrix}1\\0\\0\\1\end{bmatrix} \begin{array}{l} = 00 \\ \\ \\ = 11 \end{array}$$

$$\underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot H \otimes I \cdot \begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \underbrace{\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix}}_{H\otimes\mathbb{I}} \cdot \underbrace{\begin{bmatrix}1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}} = \frac{1}{\sqrt{2}} \underbrace{\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}}_{CNOT} \cdot \begin{bmatrix}1\\0\\1\\0\end{bmatrix}$$

## Entanglement

Entanglement states cannot be expressed as a tensor product of single-qubit states:

$$\frac{1}{\sqrt{2}} \cdot \begin{bmatrix}1\\0\\0\\1\end{bmatrix} = \frac{1}{\sqrt{2}} \cdot \left( \begin{bmatrix}1\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\1\end{bmatrix} \right) \neq \begin{bmatrix}\alpha_1\\\beta_1\end{bmatrix} \otimes \begin{bmatrix}\alpha_2\\\beta_2\end{bmatrix} \text{ for } \alpha_1, \beta_1, \alpha_2, \beta_2 \in \mathbb{R}$$

We read circuits from left to right, but the linear algebra works the other way around!

$$\begin{bmatrix}1\\0\end{bmatrix} \; -\boxed{A}-\boxed{B}-\boxed{C}- \quad \equiv C \cdot B \cdot A \cdot \begin{bmatrix}1\\0\end{bmatrix}$$

We read circuits from left to right, but the linear algebra works the other way around!

$$\begin{bmatrix}1\\0\end{bmatrix} \;-\!\boxed{A}\!-\!\boxed{B}\!-\!\boxed{C}\!- \quad \equiv C \cdot B \cdot A \cdot \begin{bmatrix}1\\0\end{bmatrix}$$

## Definition (Tensor product)

A tensor product of matrices $A, B$, written $A \otimes B$, is defined as:

$$A \otimes B = \begin{bmatrix} a_{00} \cdot B & a_{01} \cdot B & \dots & a_{0l} \cdot B \\ a_{10} \cdot B & a_{11} \cdot B & \dots & a_{1l} \cdot B \\ \vdots & \vdots & \ddots & \vdots \\ a_{k0} \cdot B & a_{k1} \cdot B & \dots & a_{kl} \cdot B \end{bmatrix}, \text{ where } A = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0l} \\ a_{10} & a_{11} & \dots & a_{1l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k0} & a_{k1} & \dots & a_{kl} \end{bmatrix}$$

We read circuits from left to right, but the linear algebra works the other way around!

$$\begin{bmatrix}1\\0\end{bmatrix} \; - \boxed{A} - \boxed{B} - \boxed{C} - \quad \equiv C \cdot B \cdot A \cdot \begin{bmatrix}1\\0\end{bmatrix}$$

## Definition (Tensor product)

A tensor product of matrices $A, B$, written $A \otimes B$, is defined as:

$$A \otimes B = \begin{bmatrix} a_{00} \cdot B & a_{01} \cdot B & \dots & a_{0l} \cdot B \\ a_{10} \cdot B & a_{11} \cdot B & \dots & a_{1l} \cdot B \\ \vdots & \vdots & \ddots & \vdots \\ a_{k0} \cdot B & a_{k1} \cdot B & \dots & a_{kl} \cdot B \end{bmatrix}, \text{ where } A = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0l} \\ a_{10} & a_{11} & \dots & a_{1l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k0} & a_{k1} & \dots & a_{kl} \end{bmatrix}$$

$$|100\rangle = \underbrace{\begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}0\\1\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}}$$

We read circuits from left to right, but the linear algebra works the other way around!

$$\begin{bmatrix}1\\0\end{bmatrix} \;-\boxed{A}-\boxed{B}-\boxed{C}-\; \equiv C \cdot B \cdot A \cdot \begin{bmatrix}1\\0\end{bmatrix}$$

## Definition (Tensor product)

A tensor product of matrices $A, B$, written $A \otimes B$, is defined as:

$$A \otimes B = \begin{bmatrix} a_{00} \cdot B & a_{01} \cdot B & \dots & a_{0l} \cdot B \\ a_{10} \cdot B & a_{11} \cdot B & \dots & a_{1l} \cdot B \\ \vdots & \vdots & \ddots & \vdots \\ a_{k0} \cdot B & a_{k1} \cdot B & \dots & a_{kl} \cdot B \end{bmatrix}, \text{ where } A = \begin{bmatrix} a_{00} & a_{01} & \dots & a_{0l} \\ a_{10} & a_{11} & \dots & a_{1l} \\ \vdots & \vdots & \ddots & \vdots \\ a_{k0} & a_{k1} & \dots & a_{kl} \end{bmatrix}$$

$$|100\rangle = \underbrace{\begin{bmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{bmatrix}}_{\begin{bmatrix}0\\1\end{bmatrix} \otimes \begin{bmatrix}1\\0\end{bmatrix}^{\otimes 2}}$$

An $n$-qubit state is a $2^n$ vector and an $n$-qubit circuit represents a $2^n \times 2^n$ unitary matrix.

# Universal gate-sets

### Three-qubit gate set

The set $\{ H, CCX \}$ is universal for approximating any unitary! [1]

# Universal gate-sets

## Three-qubit gate set

The set $\{ H, CCX \}$ is universal for approximating any unitary! [1]

## Single qubit-gates

$$H = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \qquad T = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{i} \end{bmatrix}$$

[1] D. Aharonov, "A simple proof that toffoli and hadamard are quantum universal," *arXiv preprint quant-ph/0301040*, 2003.

# Universal gate-sets

## Three-qubit gate set

The set $\{ H, CCX \}$ is universal for approximating any unitary! [1]

## Single qubit-gates

$$H = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad S = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} \qquad T = \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{i} \end{bmatrix}$$

## Other universal gate sets

The set $\{ H, T, \mathrm{CNOT} \}$ is universal for approximating any unitary...
The Clifford gate set $\{ H, S, \mathrm{CNOT} \}$ is not universal and classically simulatable...

[1] D. Aharonov, "A simple proof that toffoli and hadamard are quantum universal," *arXiv preprint quant-ph/0301040*, 2003.

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| | |
| | |
| | |
| | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | |
| | |
| | |
| | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
| --- | --- |
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| | |
| | |
| | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | |
| | |
| | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| | |
| | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | |
| | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| | |
| | |
| | |

## Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| **Circuit picture** | **Linear algebra picture** |
| --- | --- |
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | |

# Summary & step to quantum

For a circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix |

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits | vector $\vec{v}$ of length $2^n$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix |

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| **Circuit picture** | **Linear algebra picture** |
|---|---|
| state of $n$ bits (prob. distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of probabilities: $\vec{v}_j \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix |

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits (prob. distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of probabilities: $\vec{v}_j \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are probability vectors: $M_{jk} \in [0,1]$ and $\sum_{k=1}^{2^n} M_{jk} = 1$; |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix |

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits (prob. distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of probabilities: $\vec{v}_j \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are probability vectors: $M_{jk} \in [0,1]$ and $\sum_{k=1}^{2^n} M_{jk} = 1$; whose columns are orthogonal |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix |

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits (prob. distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of probabilities: $\vec{v}_j \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are probability vectors: $M_{jk} \in [0,1]$ and $\sum_{k=1}^{2^n} M_{jk} = 1$; whose columns are orthogonal |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix with the same properties as a gate |

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ bits (prob. distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of probabilities: $\vec{v}_j \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are probability vectors: $M_{jk} \in [0,1]$ and $\sum_{k=1}^{2^n} M_{jk} = 1$; whose columns are orthogonal |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix with the same properties as a gate |

Now let's make the step to **quantum**!

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ qubits (~~prob.~~ amplitude distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of ~~probabilities~~ amplitudes: $\vec{v}_j{}^2 \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j{}^2 = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are ~~probability~~ amplitude vectors: $M_{jk}{}^2 \in [0,1]$ and $\sum_{k=1}^{2^n} M_{jk}{}^2 = 1$; whose columns are orthogonal |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix with the same properties as a gate |

Now let's make the step to **quantum**!

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ qubits (~~prob.~~ amplitude distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of ~~probabilities~~ amplitudes: $\vec{v}_j^{\,2} \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j^{\,2} = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are ~~probability~~ amplitude vectors: ${M_{jk}}^2 \in [0,1]$ and $\sum_{k=1}^{2^n} {M_{jk}}^2 = 1$; whose columns are orthogonal |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix with the same properties as a gate |

Now let's make the step to **quantum**!

**Note:** generally, quantum has complex numbers ($\vec{v}_j^2 \to |\vec{v}_j|^2$, etc.), but we will only use real numbers today.

# Summary & step to quantum

For a probabilistic / reversible circuit mapping $n$ to $n$ bits:

| Circuit picture | Linear algebra picture |
|---|---|
| state of $n$ qubits (~~prob.~~ amplitude distribution over bitstrings) | vector $\vec{v}$ of length $2^n$ of ~~probabilities~~ amplitudes: $\vec{v}_j{}^2 \in [0,1]$ and $\sum_{j=1}^{2^n} \vec{v}_j{}^2 = 1$ |
| gate | $2^n \times 2^n$ matrix $M$ whose columns are ~~probability~~ amplitude vectors: $M_{jk}{}^2 \in [0,1]$ and $\sum_{k=1}^{2^n} M_{jk}{}^2 = 1$; whose columns are orthogonal |
| evaluate the circuit on some input | matrix-vector product |
| sequential composition of gates | matrix product |
| parallel composition of gates | Kronecker product |
| $n$-bit circuit | $2^n \times 2^n$ matrix with the same properties as a gate |

Now let's make the step to **quantum**!

**Note:** generally, quantum has complex numbers ($\vec{v}_j^2 \rightarrow |\vec{v}_j|^2$, etc.), but we will only use real numbers today.

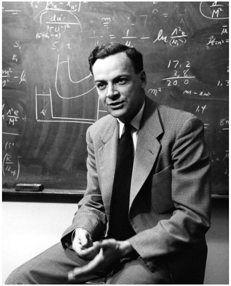
**Note:** We can use automata instead!

*"If you were shot with a poison arrow, would you waste time and breath by asking who shot the arrow, how tall was he and of what complexion?"*

Siddhartha Gautama

*"If you were shot with a poison arrow, would you waste time and breath by asking who shot the arrow, how tall was he and of what complexion?"*

Siddhartha Gautama



Feynman (maybe)

*"Shut up and calculate"*

# Takeaways & Further Reading

## Takeaways

- "Quantum computing becomes easy once one takes the physics out" — Aaronson

## Further reading: see books [1–3]


[1] R. J. Lipton and K. W. Regan, *Introduction to quantum algorithms via linear algebra*. MIT Press, 2021.

[2] J. Watrous, *The theory of quantum information*. Cambridge university press, 2018.

[3] M. A. Nielsen and I. L. Chuang, "Quantum information and quantum computation," *Cambridge: Cambridge University Press*, vol. 2, 2000.

[4] A.-J. Quist, J. Mei, T. Coopmans, and A. Laarman, "Advancing quantum computing with formal methods," *arXiv preprint arXiv:2407.11675*, 2024.

[5] D. Thanos, A. Villoria, S. Brand, A.-J. Quist, J. Mei, et al., "Automated reasoning in quantum circuit compilation," in *Model Checking Software (SPIN)*, 2024.

# Takeaways & Further Reading

## Takeaways

- "Quantum computing becomes easy once one takes the physics out" — Aaronson
- We can use the circuit, linear algebra and automata representations.

## Further reading: see books [1–3]


[1] R. J. Lipton and K. W. Regan, *Introduction to quantum algorithms via linear algebra*. MIT Press, 2021.

[2] J. Watrous, *The theory of quantum information*. Cambridge university press, 2018.

[3] M. A. Nielsen and I. L. Chuang, "Quantum information and quantum computation," *Cambridge: Cambridge University Press*, vol. 2, 2000.

[4] A.-J. Quist, J. Mei, T. Coopmans, and A. Laarman, "Advancing quantum computing with formal methods," *arXiv preprint arXiv:2407.11675*, 2024.

[5] D. Thanos, A. Villoria, S. Brand, A.-J. Quist, J. Mei, et al., "Automated reasoning in quantum circuit compilation," in *Model Checking Software (SPIN)*, 2024.

# Takeaways & Further Reading

## Takeaways

- "Quantum computing becomes easy once one takes the physics out" — Aaronson
- We can use the circuit, linear algebra and automata representations.

## Further reading: see books [1–3]

- This tutorial was published at FM'24, see (extended) Arxiv version [4]


[1] R. J. Lipton and K. W. Regan, *Introduction to quantum algorithms via linear algebra*. MIT Press, 2021.

[2] J. Watrous, *The theory of quantum information*. Cambridge university press, 2018.

[3] M. A. Nielsen and I. L. Chuang, "Quantum information and quantum computation," *Cambridge: Cambridge University Press*, vol. 2, 2000.

[4] A.-J. Quist, J. Mei, T. Coopmans, and A. Laarman, "Advancing quantum computing with formal methods," *arXiv preprint arXiv:2407.11675*, 2024.

[5] D. Thanos, A. Villoria, S. Brand, A.-J. Quist, J. Mei, et al., "Automated reasoning in quantum circuit compilation," in *Model Checking Software (SPIN)*, 2024.

# Takeaways & Further Reading

## Takeaways

- "Quantum computing becomes easy once one takes the physics out" — Aaronson
- We can use the circuit, linear algebra and automata representations.

## Further reading: see books [1–3]

- This tutorial was published at FM'24, see (extended) Arxiv version [4]
- Survey on formal methods for quantum computing: [5]


[1] R. J. Lipton and K. W. Regan, *Introduction to quantum algorithms via linear algebra*. MIT Press, 2021.

[2] J. Watrous, *The theory of quantum information*. Cambridge university press, 2018.

[3] M. A. Nielsen and I. L. Chuang, "Quantum information and quantum computation," *Cambridge: Cambridge University Press*, vol. 2, 2000.

[4] A.-J. Quist, J. Mei, T. Coopmans, and A. Laarman, "Advancing quantum computing with formal methods," *arXiv preprint arXiv:2407.11675*, 2024.

[5] D. Thanos, A. Villoria, S. Brand, A.-J. Quist, J. Mei, et al., "Automated reasoning in quantum circuit compilation," in *Model Checking Software (SPIN)*, 2024.

# Takeaways & Further Reading

## Takeaways

- "Quantum computing becomes easy once one takes the physics out" — Aaronson
- We can use the circuit, linear algebra and automata representations.

## Further reading: see books [1–3]

- This tutorial was published at FM'24, see (extended) Arxiv version [4]
- Survey on formal methods for quantum computing: [5]

Lectures 2 & 3: Automated Reasoning for Quantum Computing.

[1] R. J. Lipton and K. W. Regan, *Introduction to quantum algorithms via linear algebra*. MIT Press, 2021.

[2] J. Watrous, *The theory of quantum information*. Cambridge university press, 2018.

[3] M. A. Nielsen and I. L. Chuang, "Quantum information and quantum computation," *Cambridge: Cambridge University Press*, vol. 2, 2000.

[4] A.-J. Quist, J. Mei, T. Coopmans, and A. Laarman, "Advancing quantum computing with formal methods," *arXiv preprint arXiv:2407.11675*, 2024.

[5] D. Thanos, A. Villoria, S. Brand, A.-J. Quist, J. Mei, et al., "Automated reasoning in quantum circuit compilation," in *Model Checking Software (SPIN)*, 2024.