

Quantum Computing: Don't Panic

Anita Ramanan | Frances Tibble
Software Development Engineers @ Microsoft

<https://aka.ms/quantumadventures>

@whywontitbuild | @frances_tibble
anraman@microsoft.com | frtibble@microsoft.com



QUANTUM COMPUTING

- What is quantum computing? Why is it important?
- How do we build a quantum computer?
- What kinds of physical systems are already in development?
- What is Q#/the QDK and why would you use it?
- Can we use Q# to teleport a quantum state?

Quantum mechanics is weird



...and we don't know why!



Nitrogen
Fixation

cheap fertilizer
everywhere

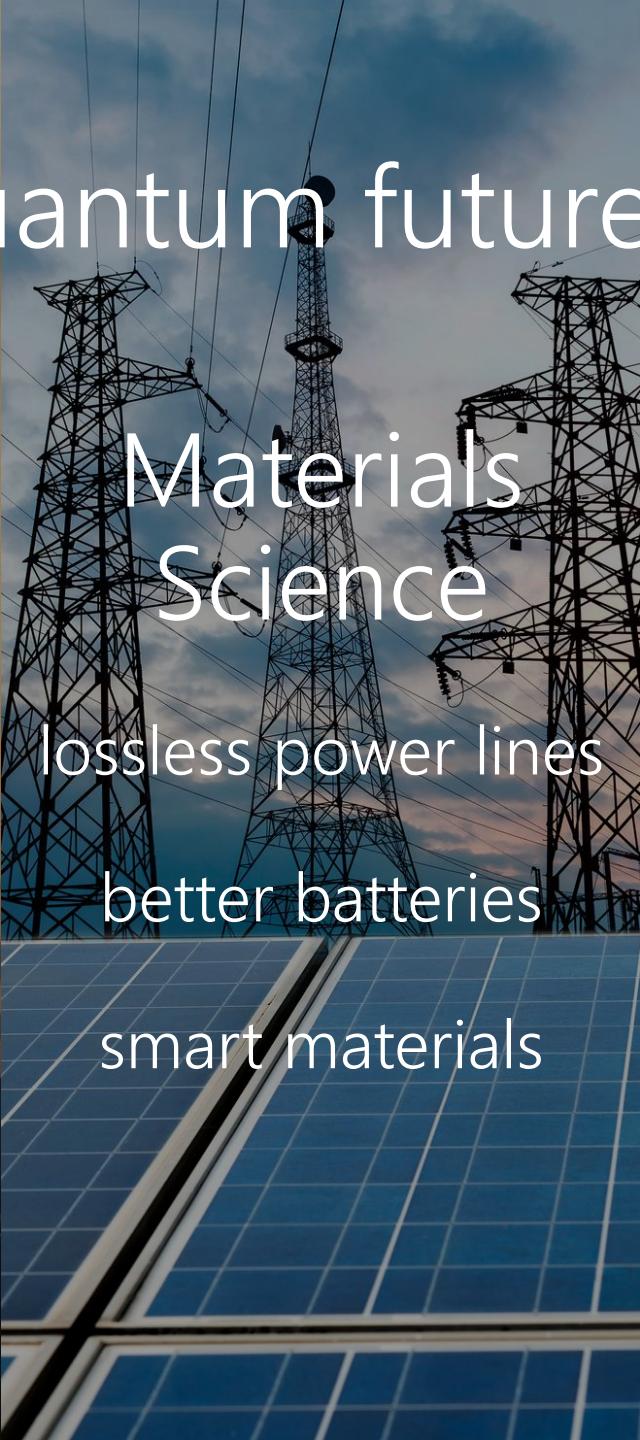


Imagine the quantum future...



Carbon
Capture

mitigate global
warming



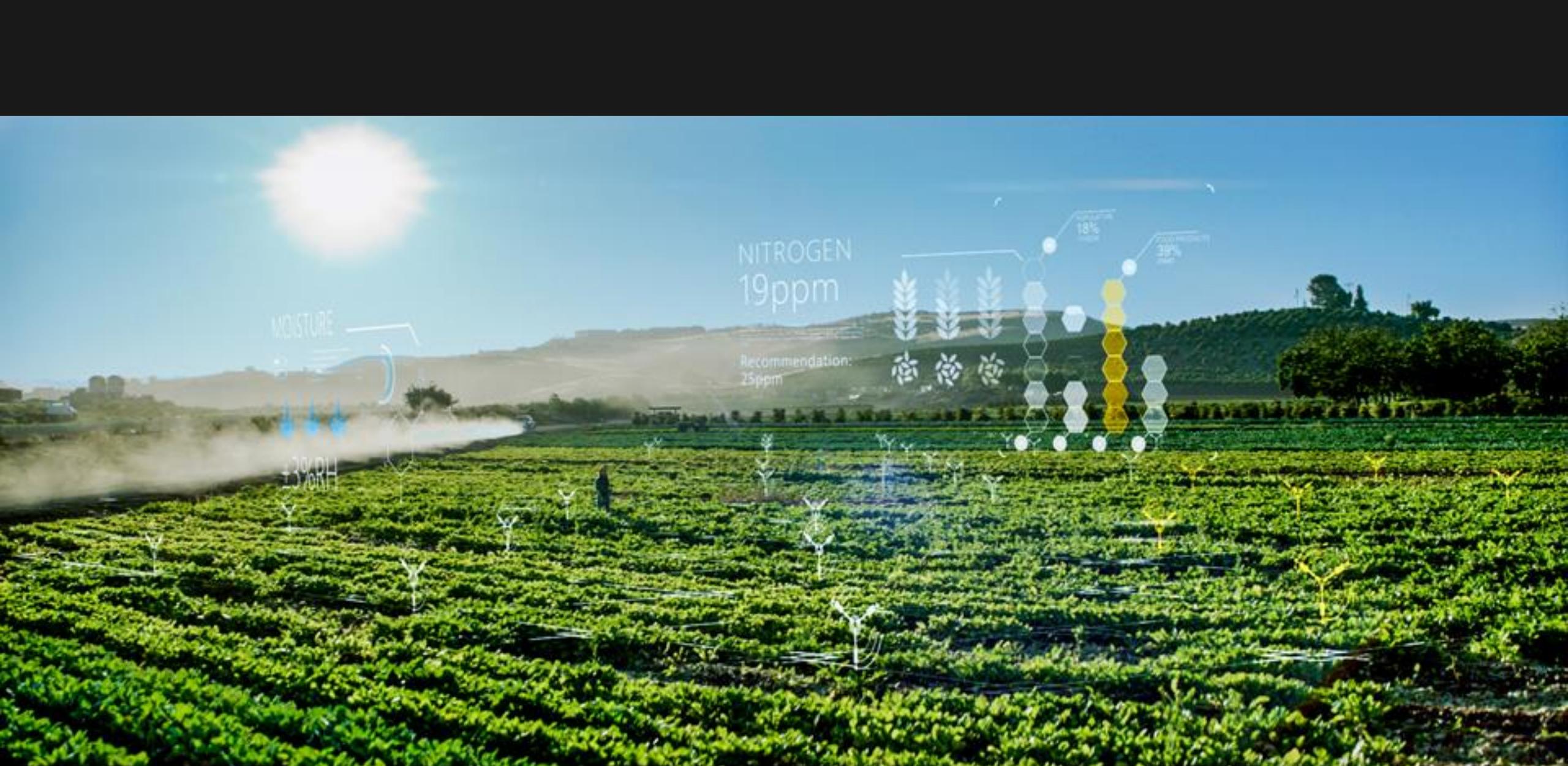
Materials
Science

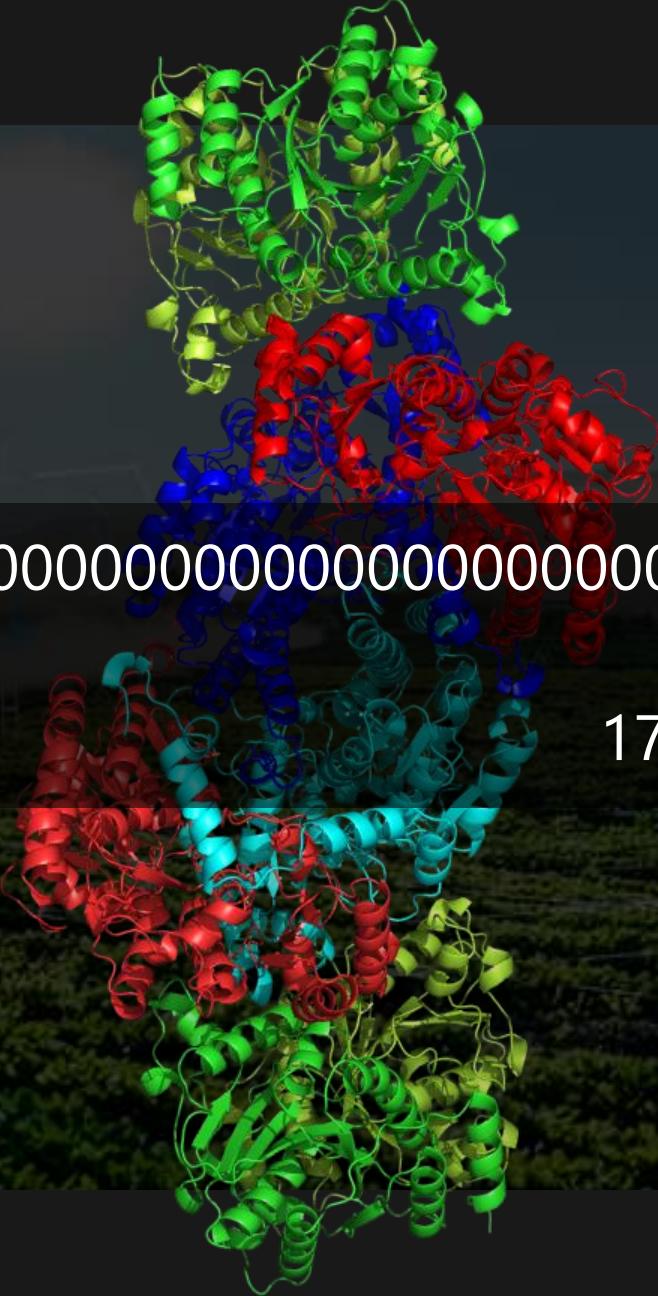
lossless power lines
better batteries

smart materials



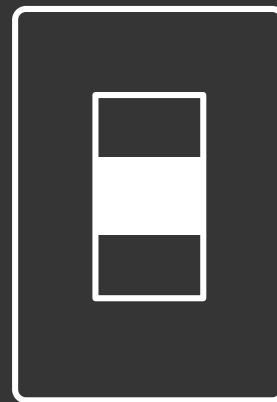
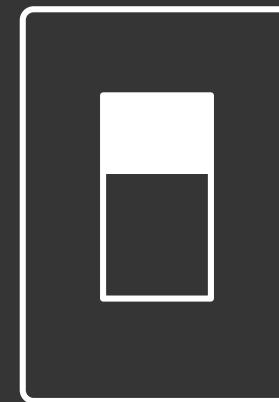
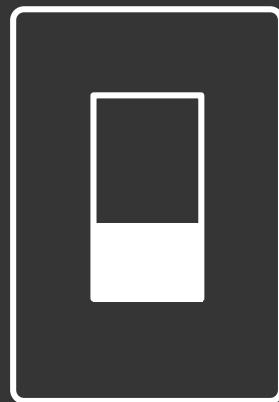
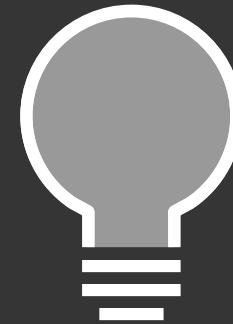
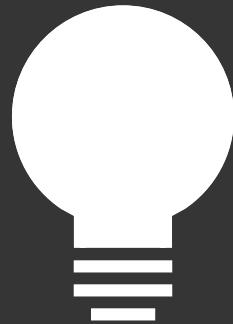
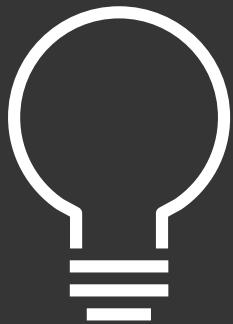
Machine
learning

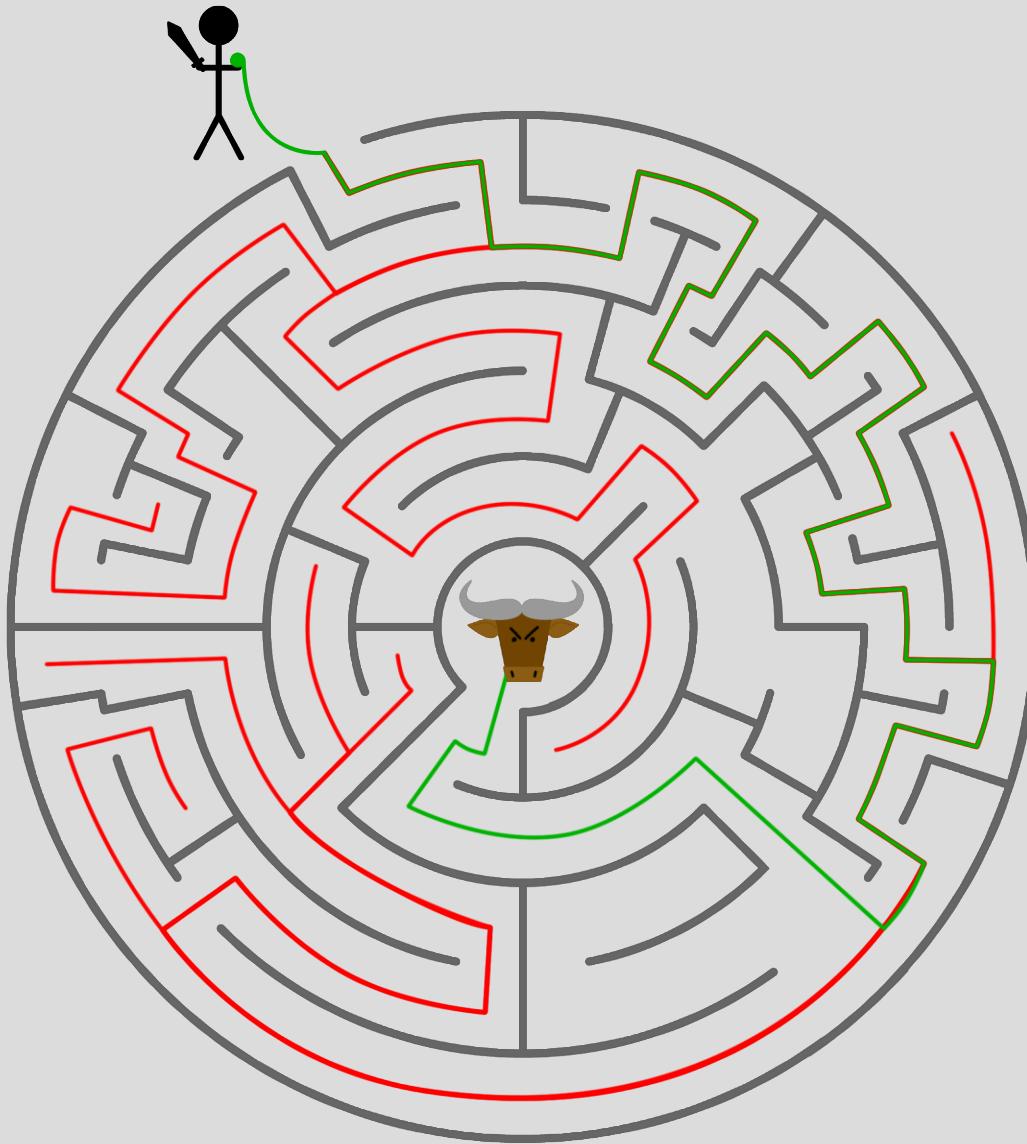


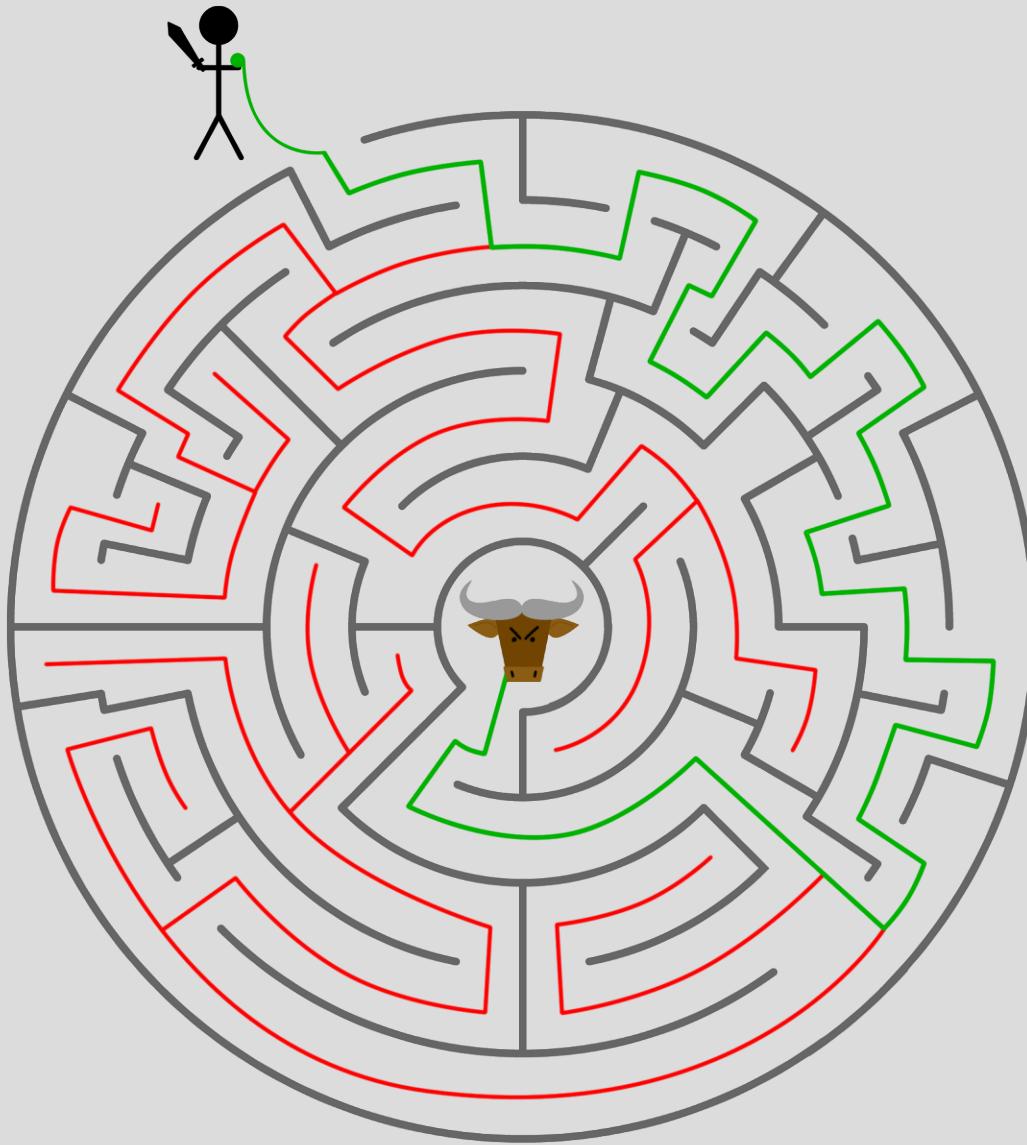


$1500 = 1.5 \times 10^{51}$ bits

OR
170 qubits







overall (superposed) state $|\psi\rangle$

possible qubit states $|0\rangle$ and $|1\rangle$

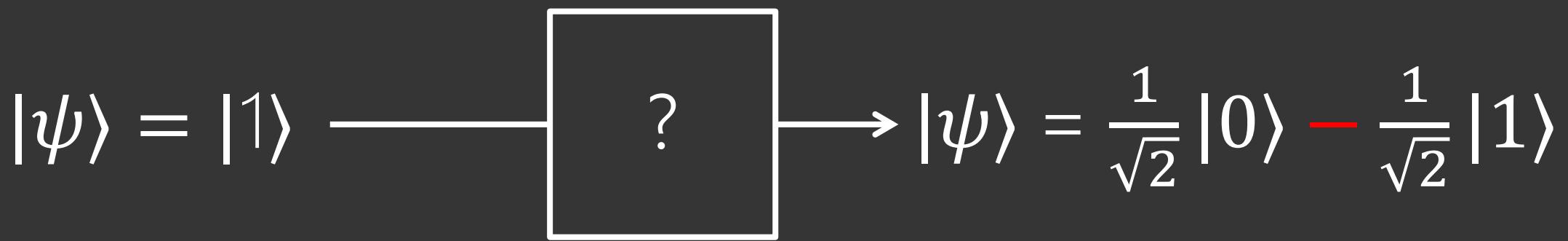
$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

probability amplitudes α and β

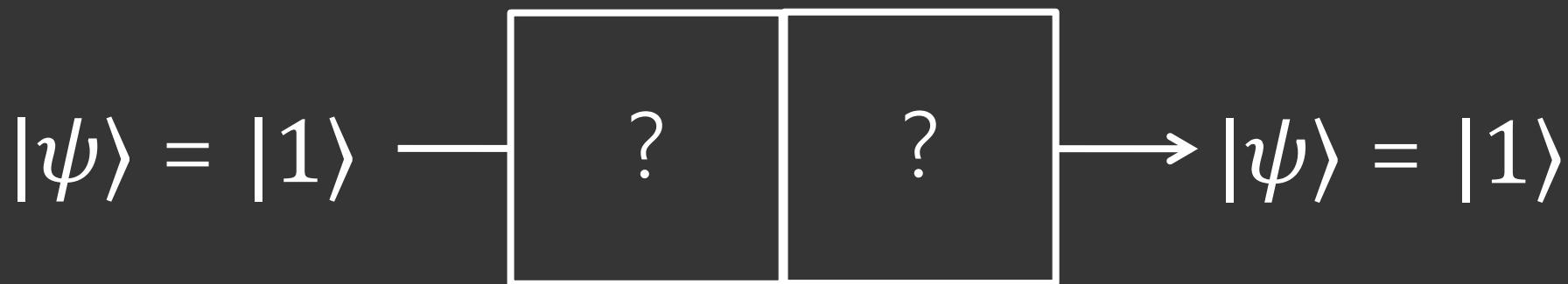
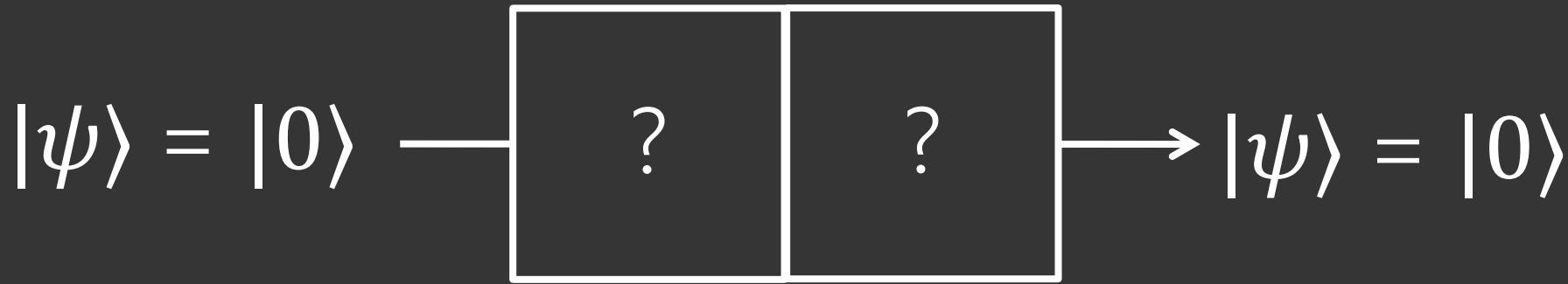
Probability of seeing $|0\rangle$ when measured: $|\alpha|^2$

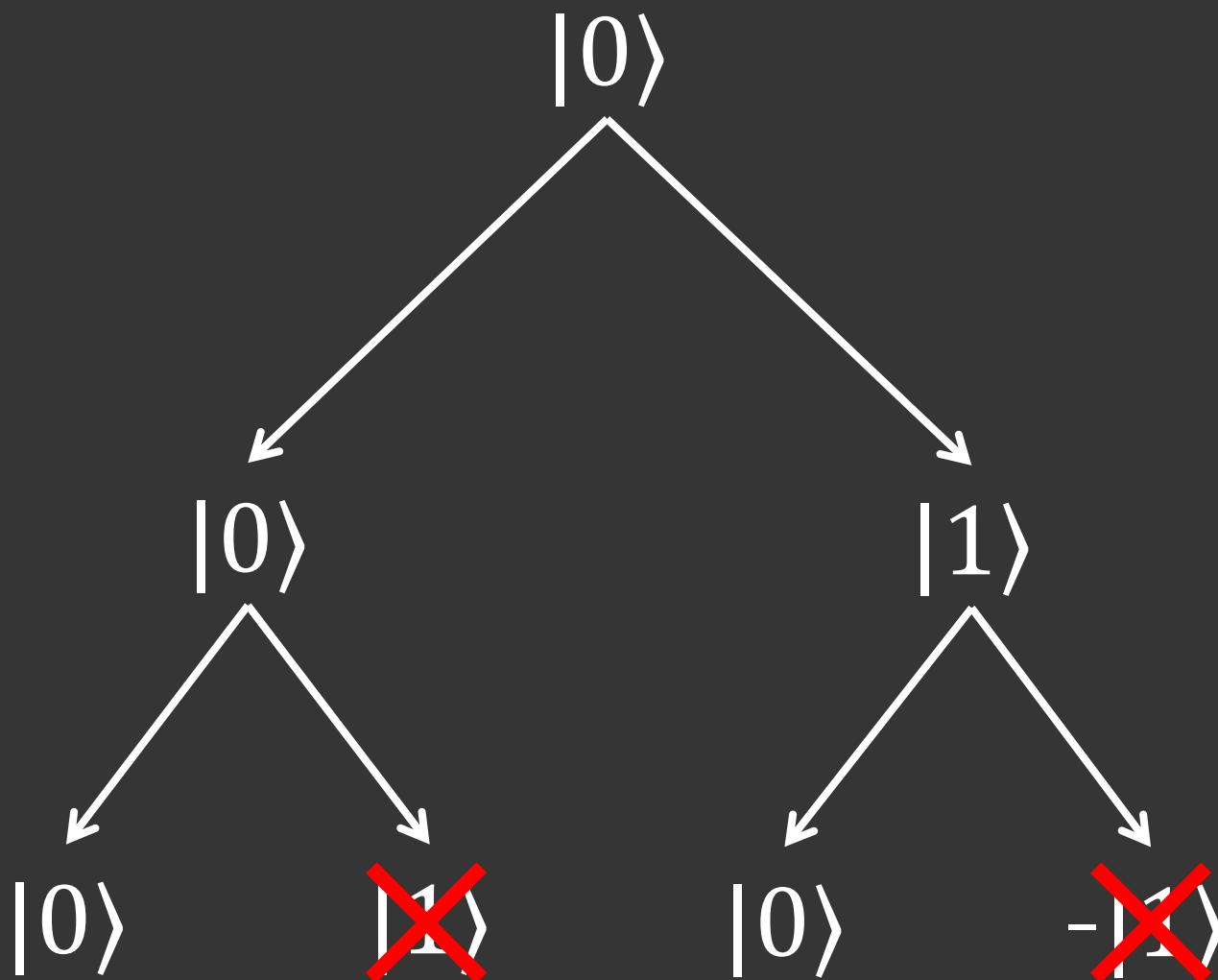
Probability of seeing $|1\rangle$ when measured: $|\beta|^2$

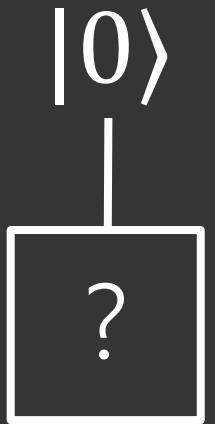
Total probability of seeing either $|0\rangle$ or $|1\rangle$: $|\alpha|^2 + |\beta|^2 = 1$



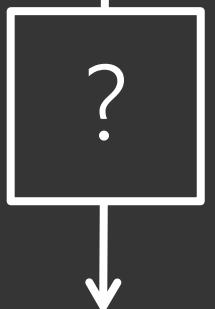
$$|\alpha|^2 = |\beta|^2 = \left| \frac{\pm 1}{\sqrt{2}} \right|^2 = \frac{1}{2}$$







$$\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \longleftarrow \text{Not a measurement!}$$



$$\frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right) + \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{2}|0\rangle + \frac{1}{2}|1\rangle + \frac{1}{2}|0\rangle - \frac{1}{2}|1\rangle = |0\rangle$$



Entanglement

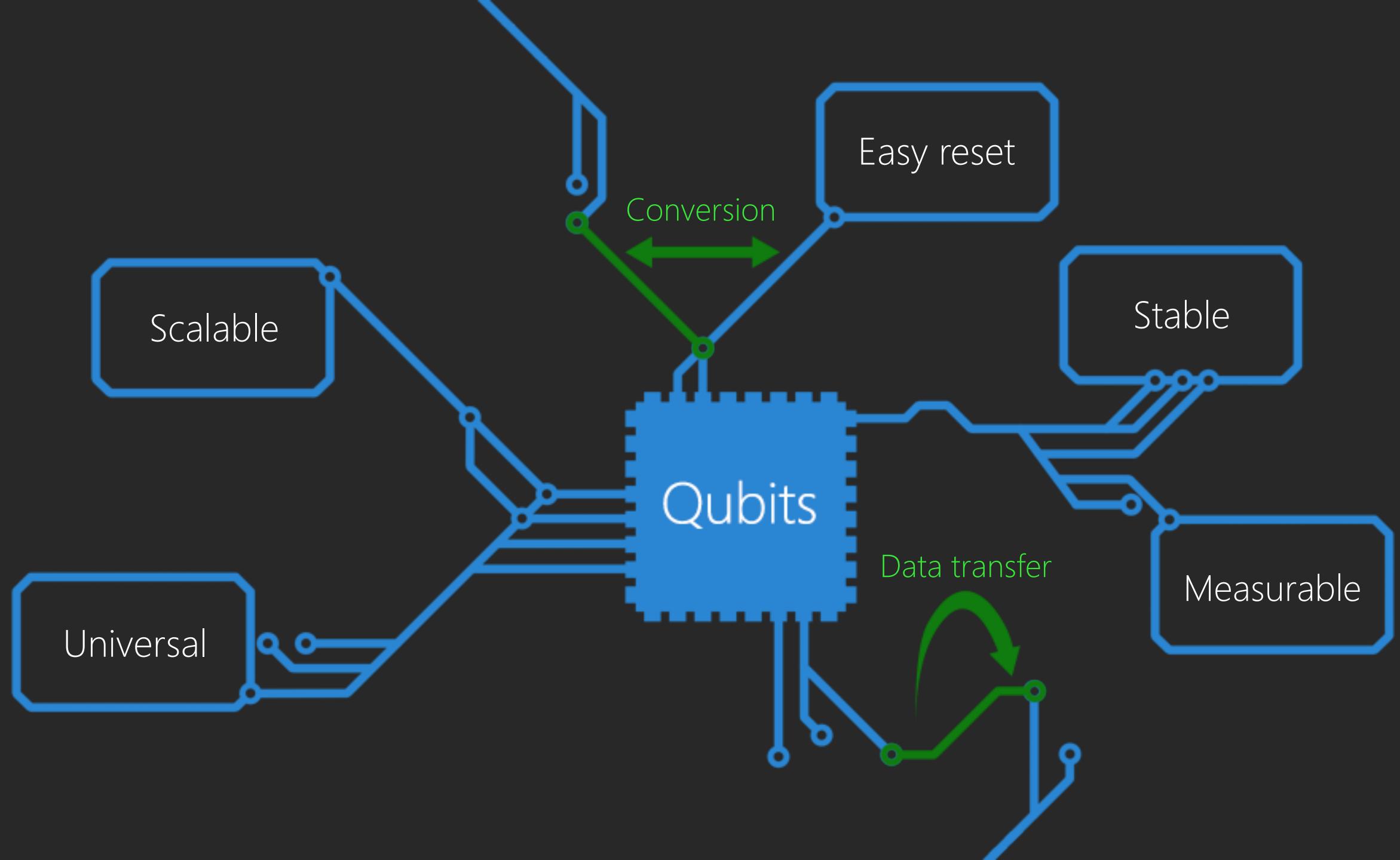
Image credit: https://commons.wikimedia.org/wiki/File:Fils_électriques.JPG

License: <https://creativecommons.org/licenses/by-sa/4.0/>

$$\frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle$$

Image credit: https://commons.wikimedia.org/wiki/File:Fils_électriques.JPG

License: <https://creativecommons.org/licenses/by-sa/4.0/>



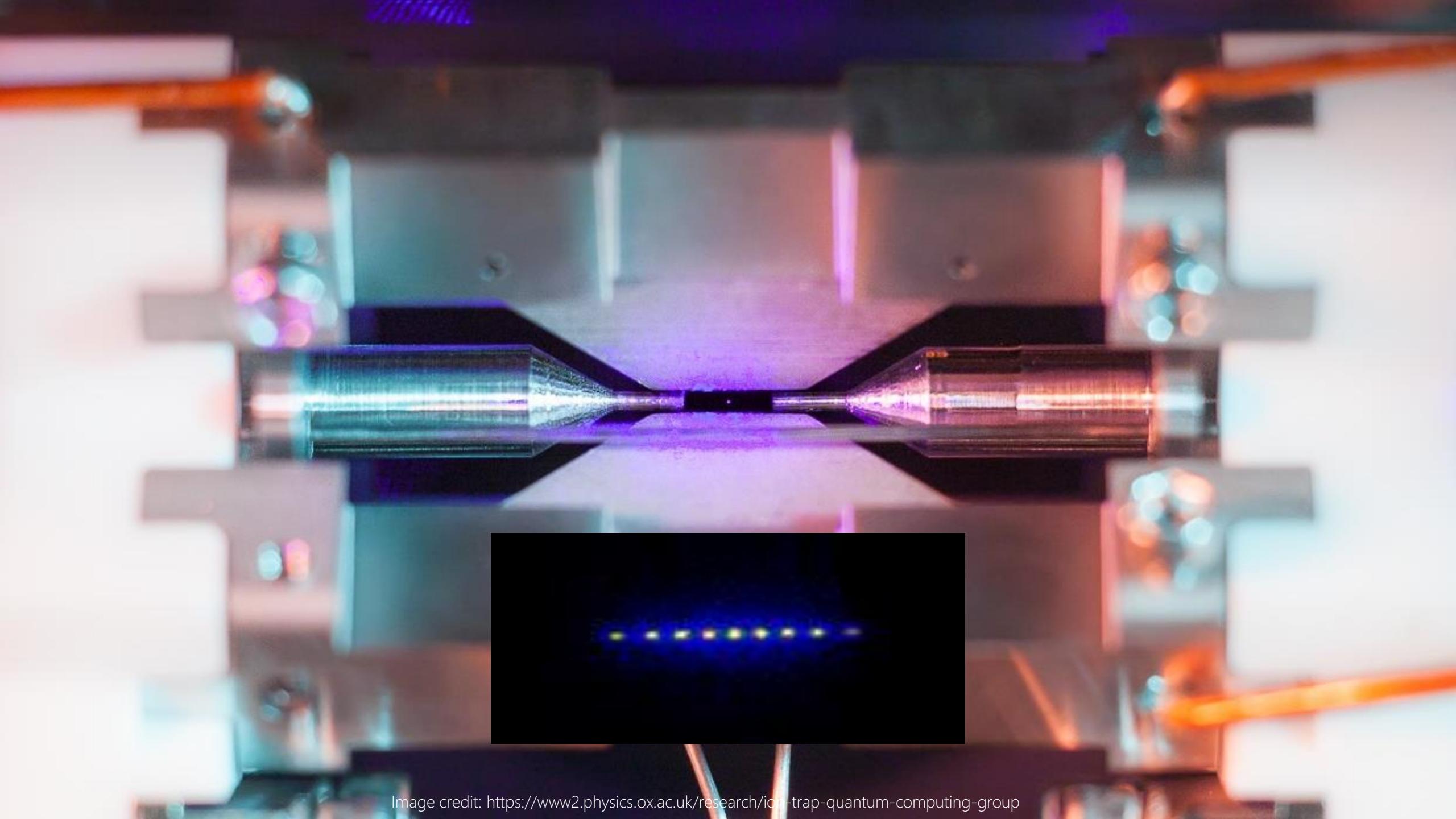
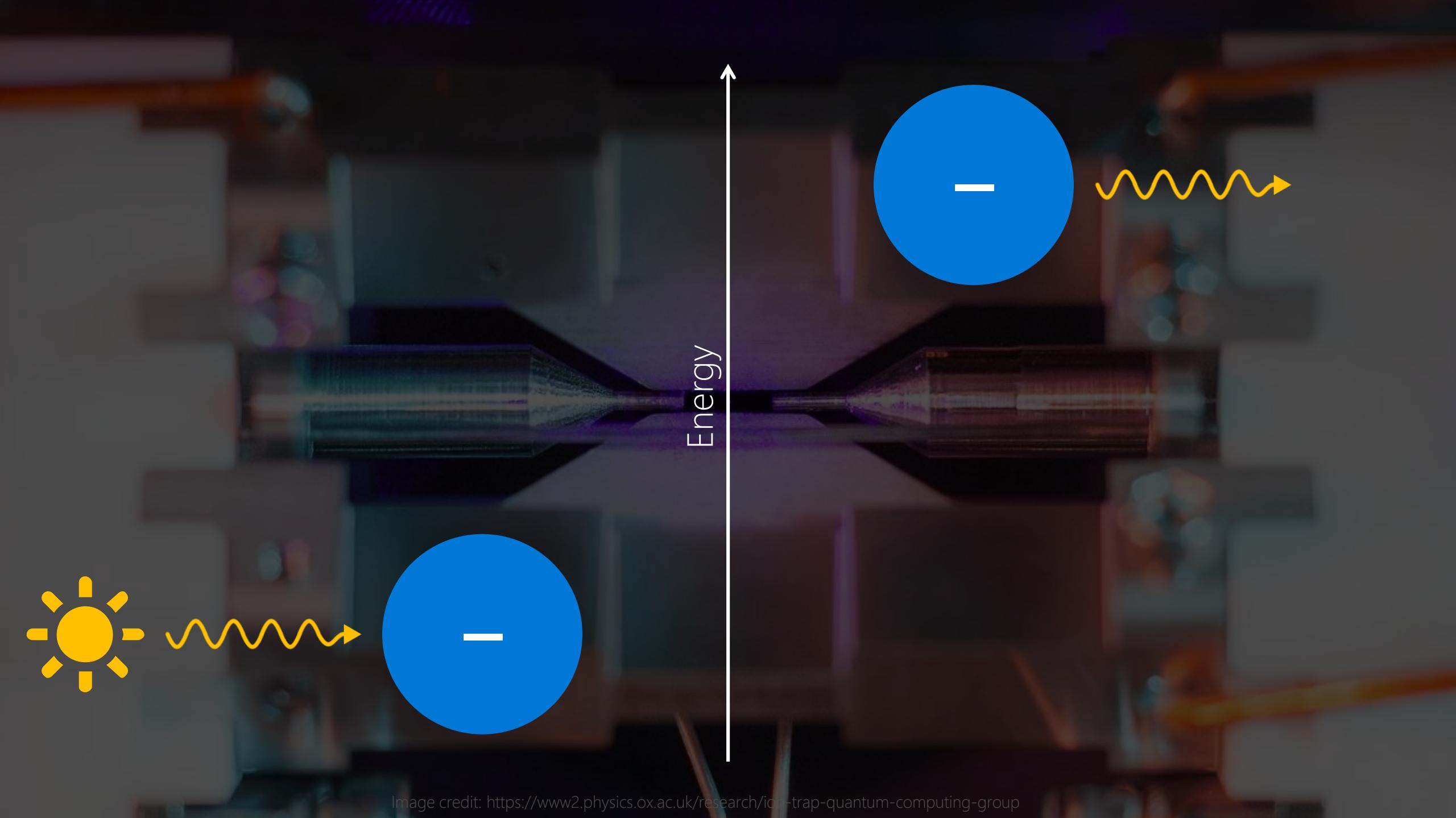


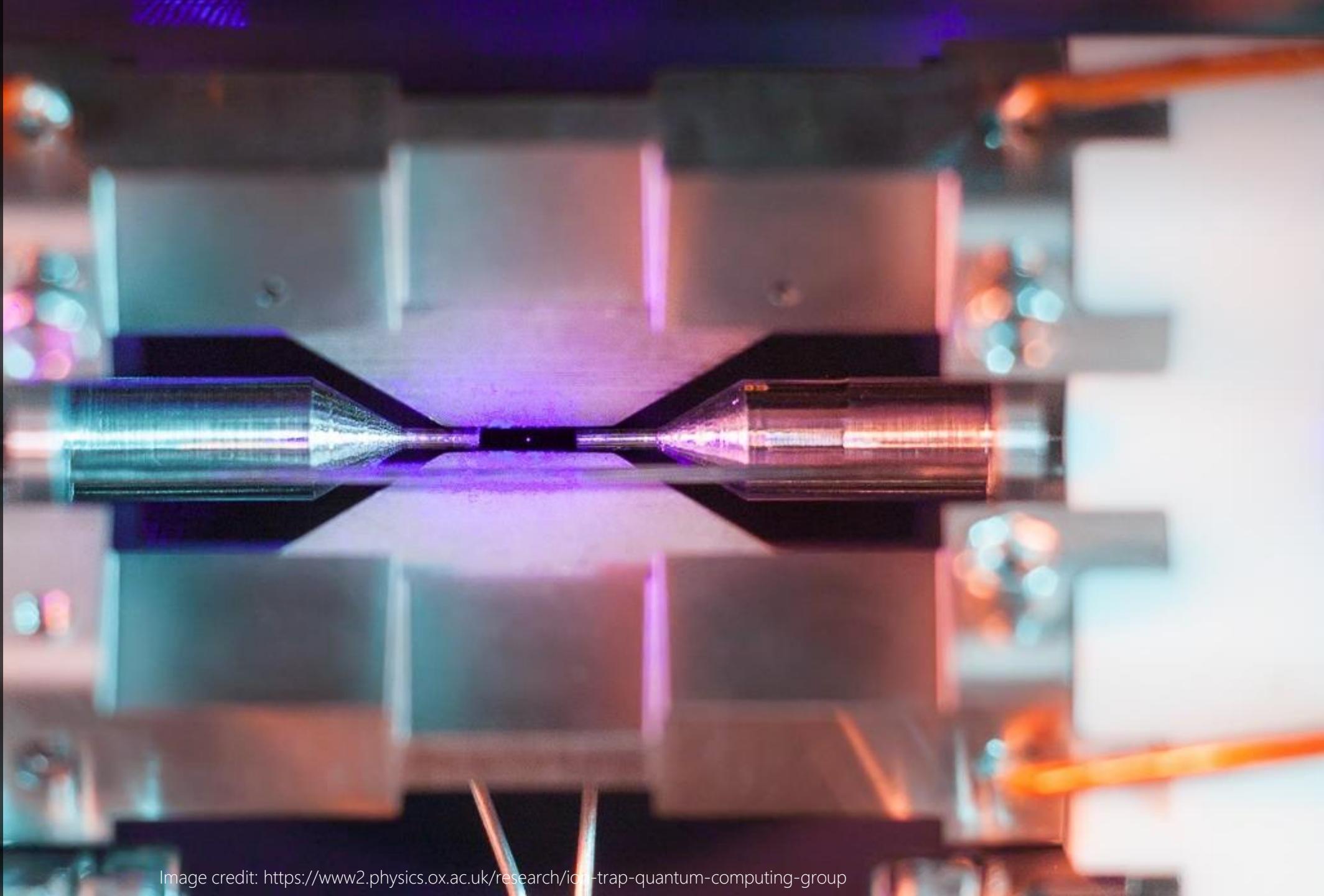
Image credit: <https://www2.physics.ox.ac.uk/research/ion-trap-quantum-computing-group>



Easy reset

Measurable

Universal



Conversion Data transfer

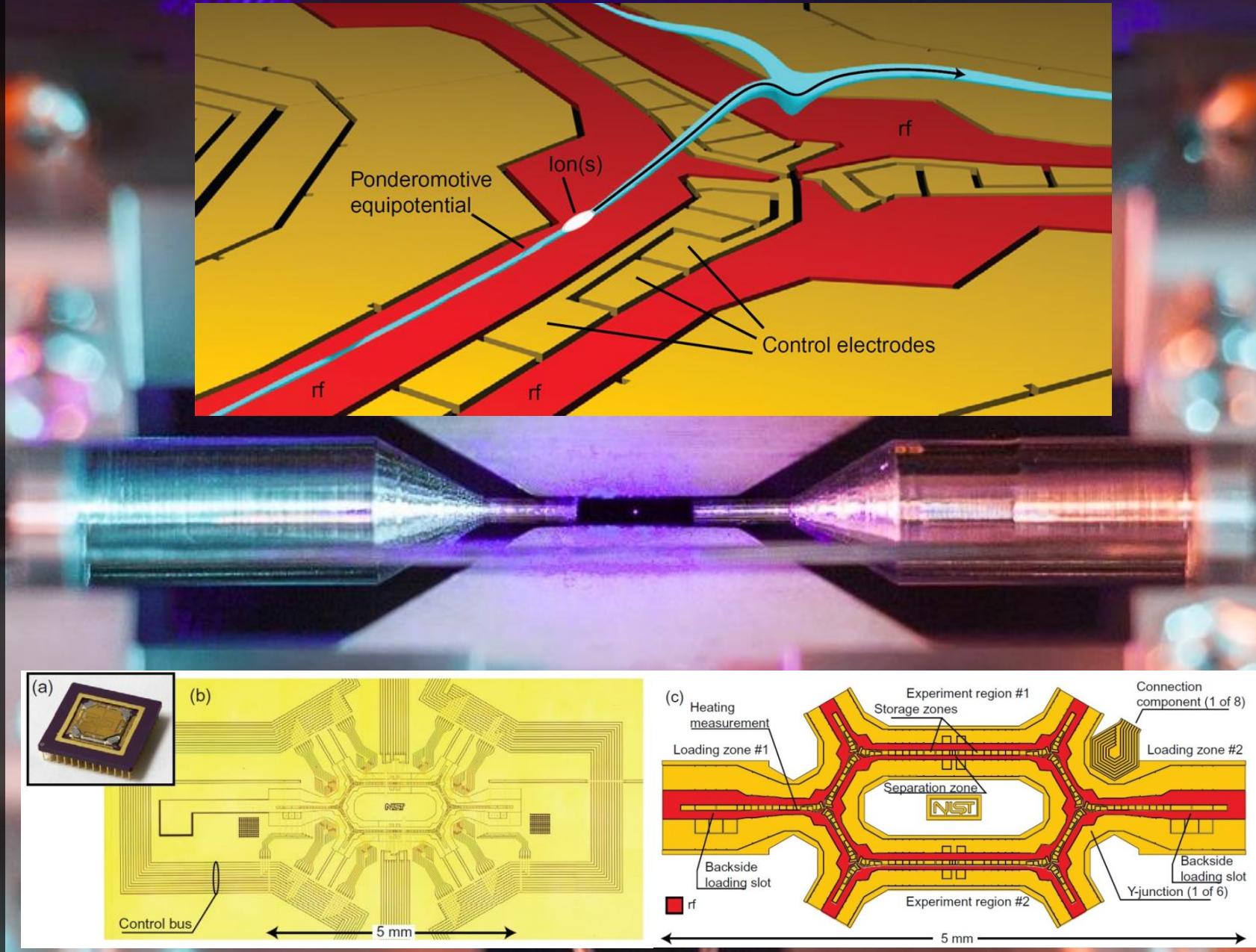
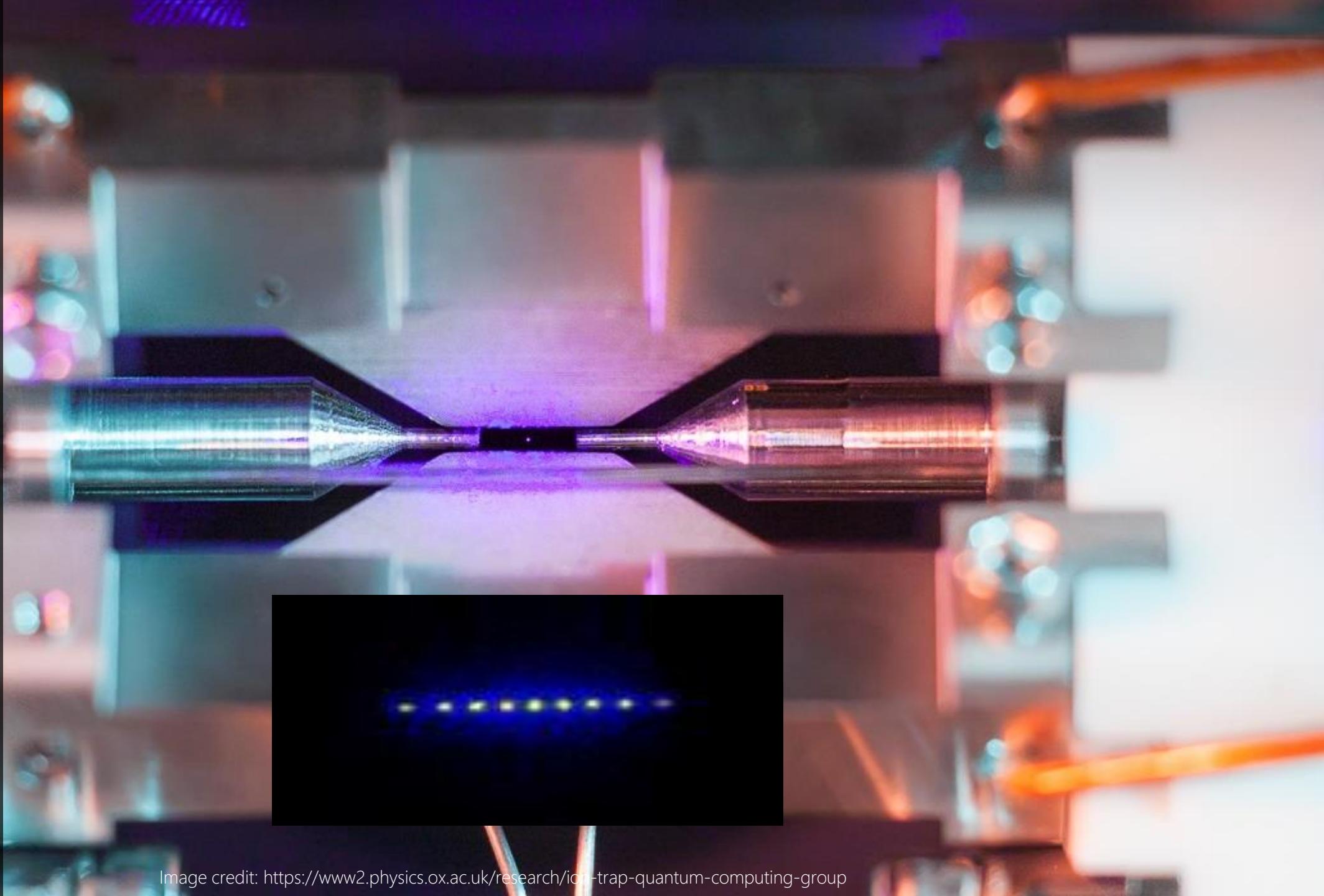


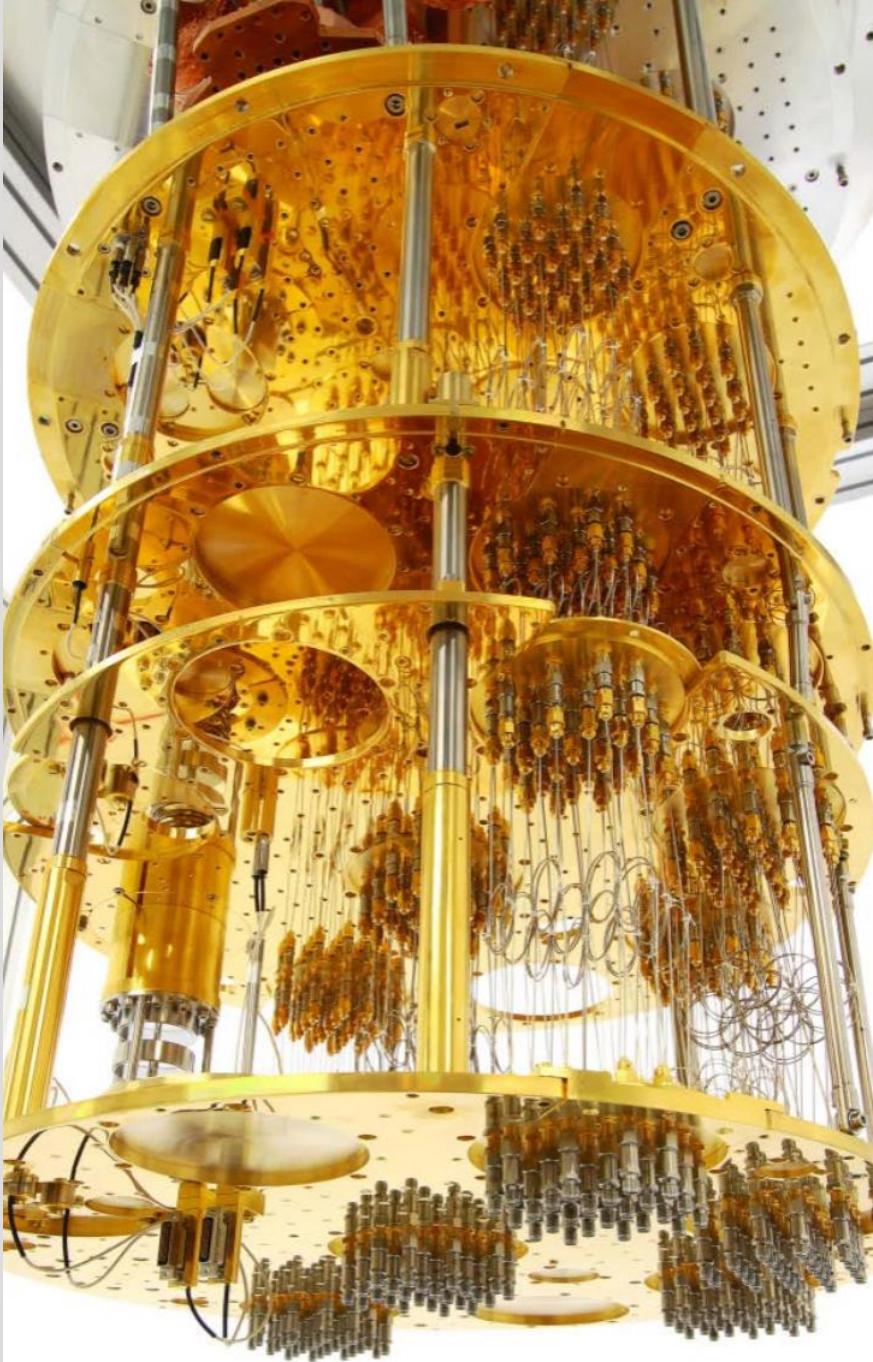
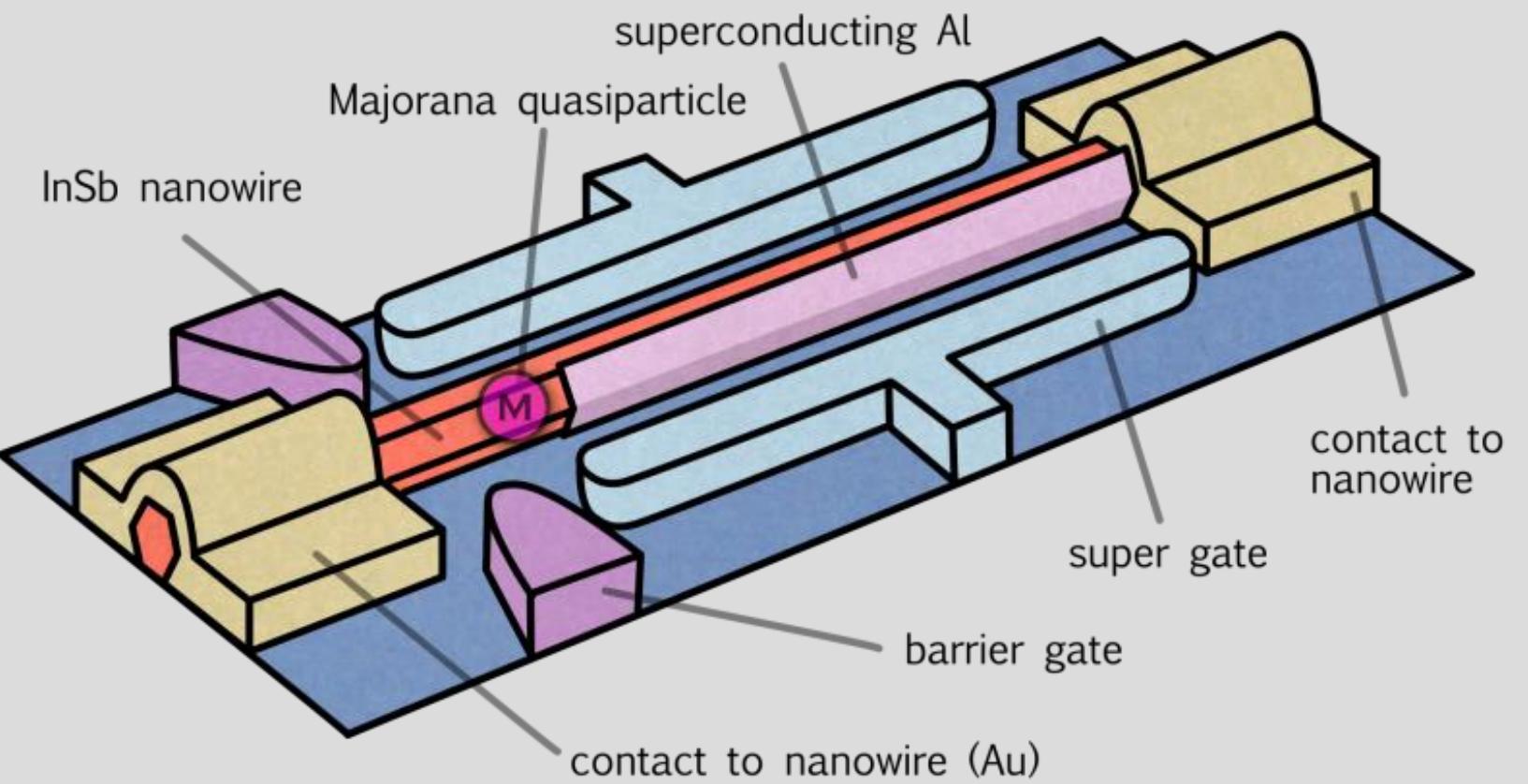
Image credit: NIST, 2010

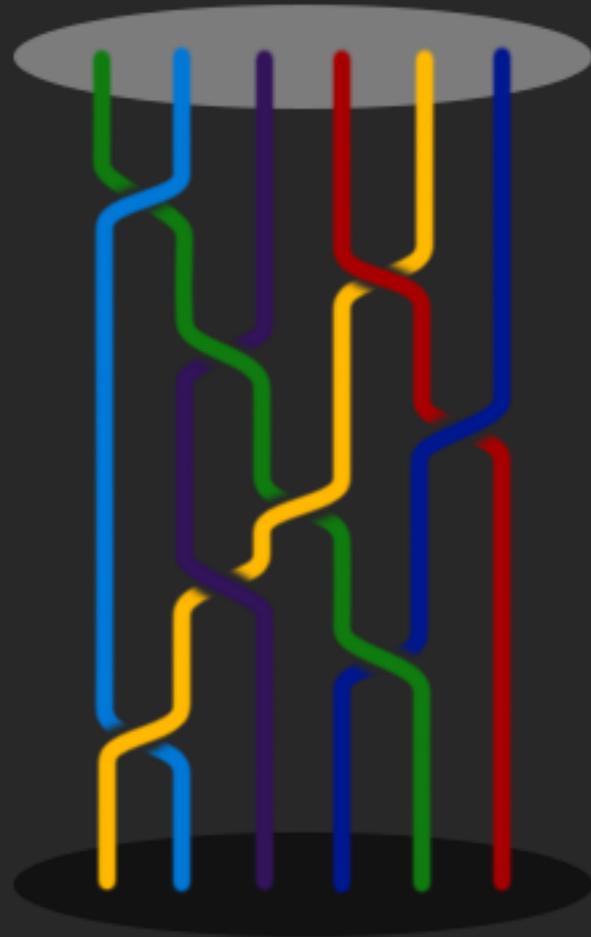
Image credit: <https://www2.physics.ox.ac.uk/research/ion-trap-quantum-computing-group>

Stable

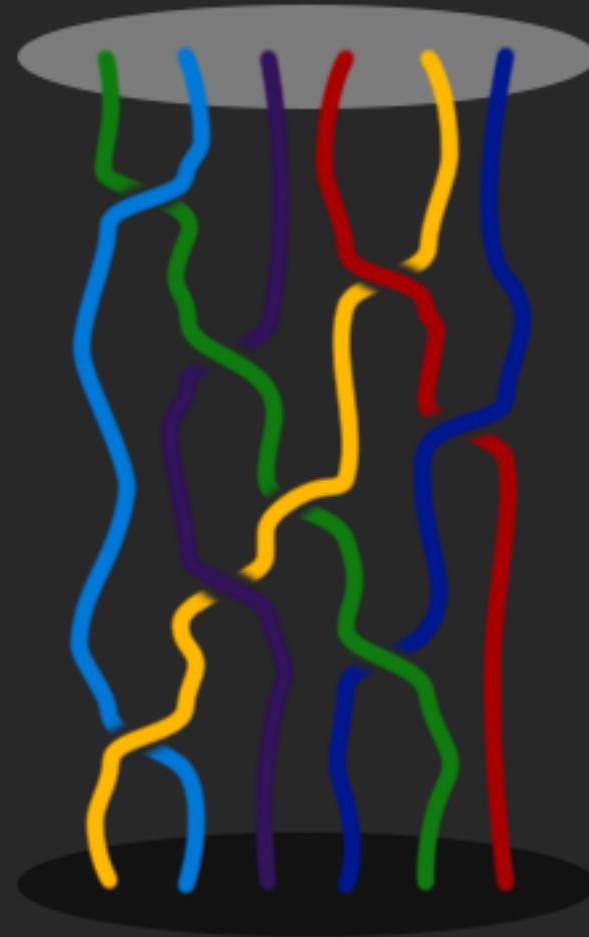
Scalable







=

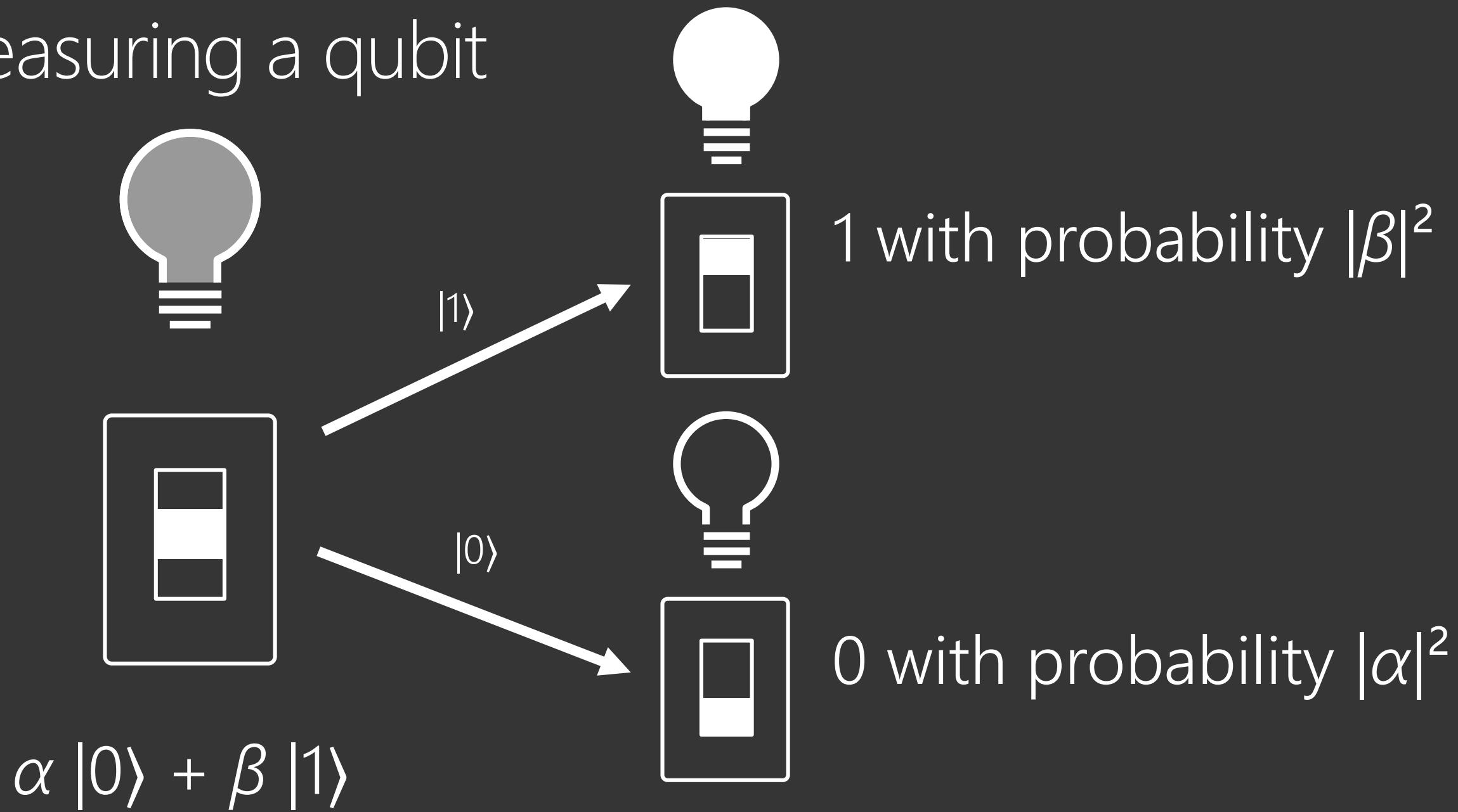


Quantum Development – Microsoft's QDK

OPEN EDITORS
TeleportationSample.qs Teleportation
SAMPLES
vscode
AdiabaticIsing
BitFlipCode
DatabaseSearch
H2SimulationCmdLine
H2SimulationGUI
HubbardSimulation
IntegerFactorization
IsingGenerators
IsingPhaseEstimation
IsingTrotterEvolution
Measurement
PhaseEstimation
PythonInterop
SimpleAlgorithms
SimpleIsing
Teleportation
bin
obj
App.config
Program.cs
README.md
TeleportationSample.csproj
TeleportationSample.qs
UnitTesting
README.md

```
32 // is unimportant.
33 // Summary
34 // Sends the state of one qubit to a target qubit by using
35 // teleportation.
36 /**
37 /**
38 /**
39 /**
40 /**
41 /**
42 /**
43 /**
44 /**
45 /**
46 /**
47 /**
48 /**
49 /**
50 /**
51 /**
52 /**
53 /**
54 /**
55 /**
56 /**
57 /**
58 /**
59 /**
60 /**
61 /**
62 /**
63 /**
64 /**
65 /**
32 // is unimportant.
33 // Summary
34 // Sends the state of one qubit to a target qubit by using
35 // teleportation.
36 /**
37 /**
38 /**
39 /**
40 /**
41 /**
42 /**
43 /**
44 /**
45 /**
46 /**
47 /**
48 /**
49 /**
50 /**
51 /**
52 /**
53 /**
54 /**
55 /**
56 /**
57 /**
58 /**
59 /**
60 /**
61 /**
62 /**
63 /**
64 /**
65 /**
```

Measuring a qubit

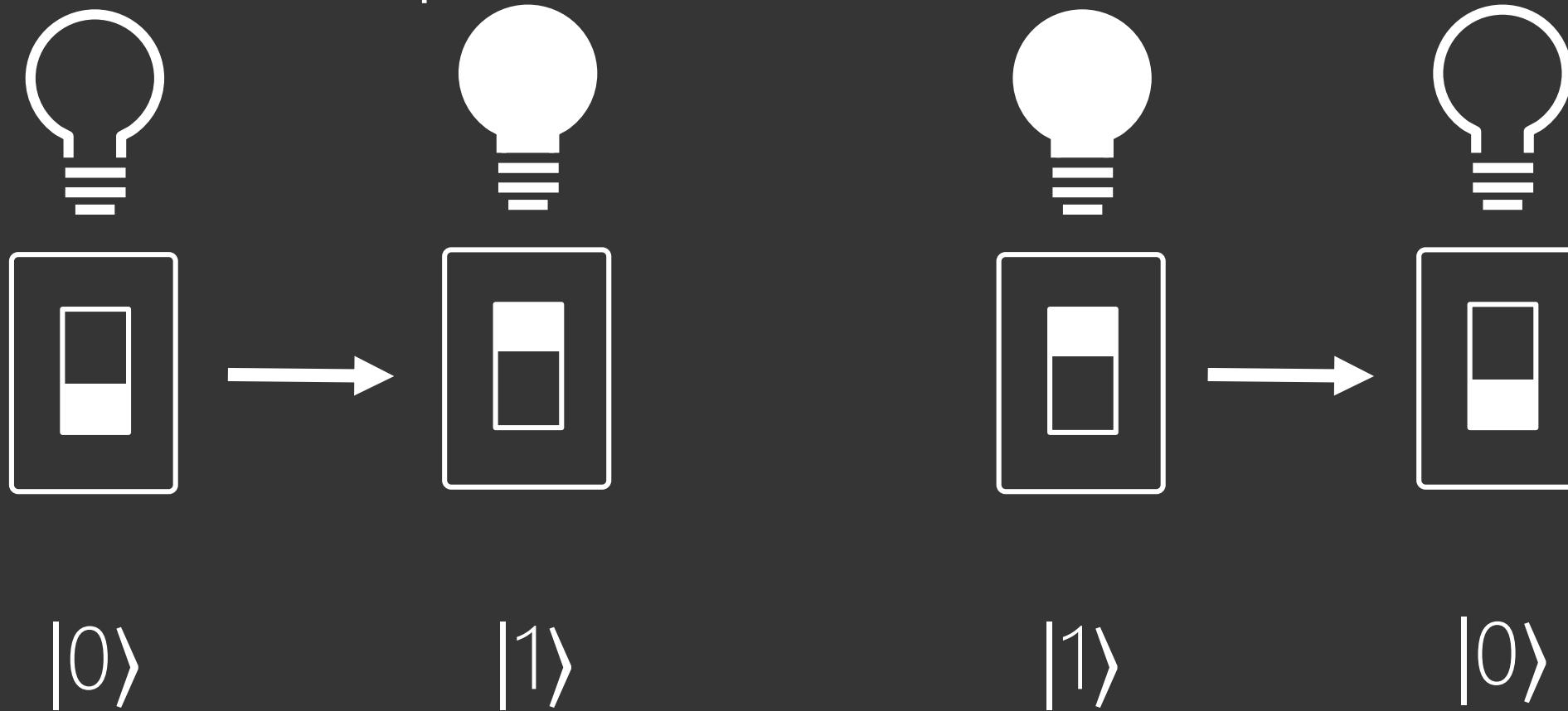


Measuring a qubit

```
operation M (qubit : Qubit) : Result
```

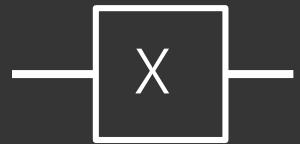


Quantum Operations – NOT

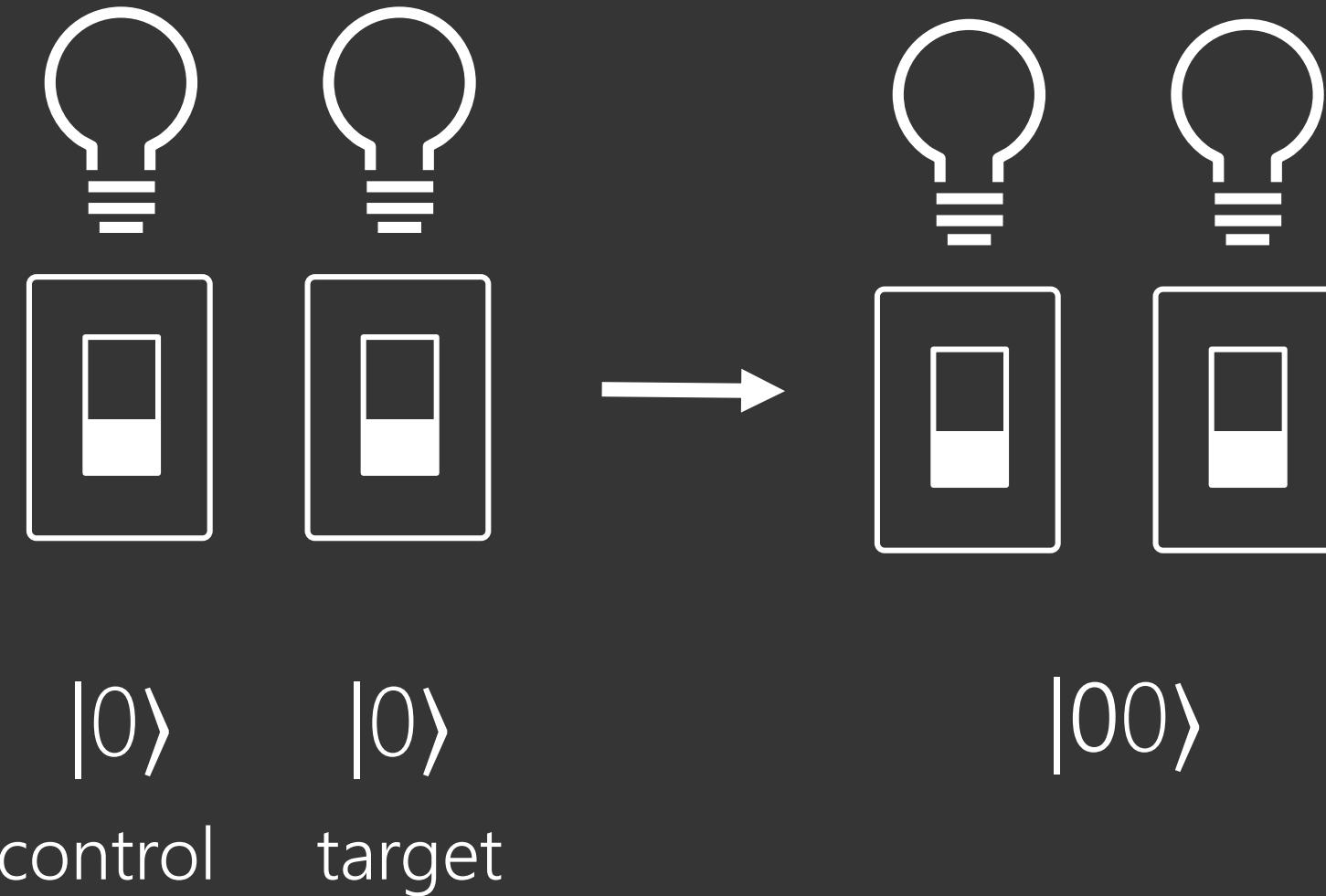


Quantum Operations – NOT

```
operation X (qubit : Qubit) : ()
```

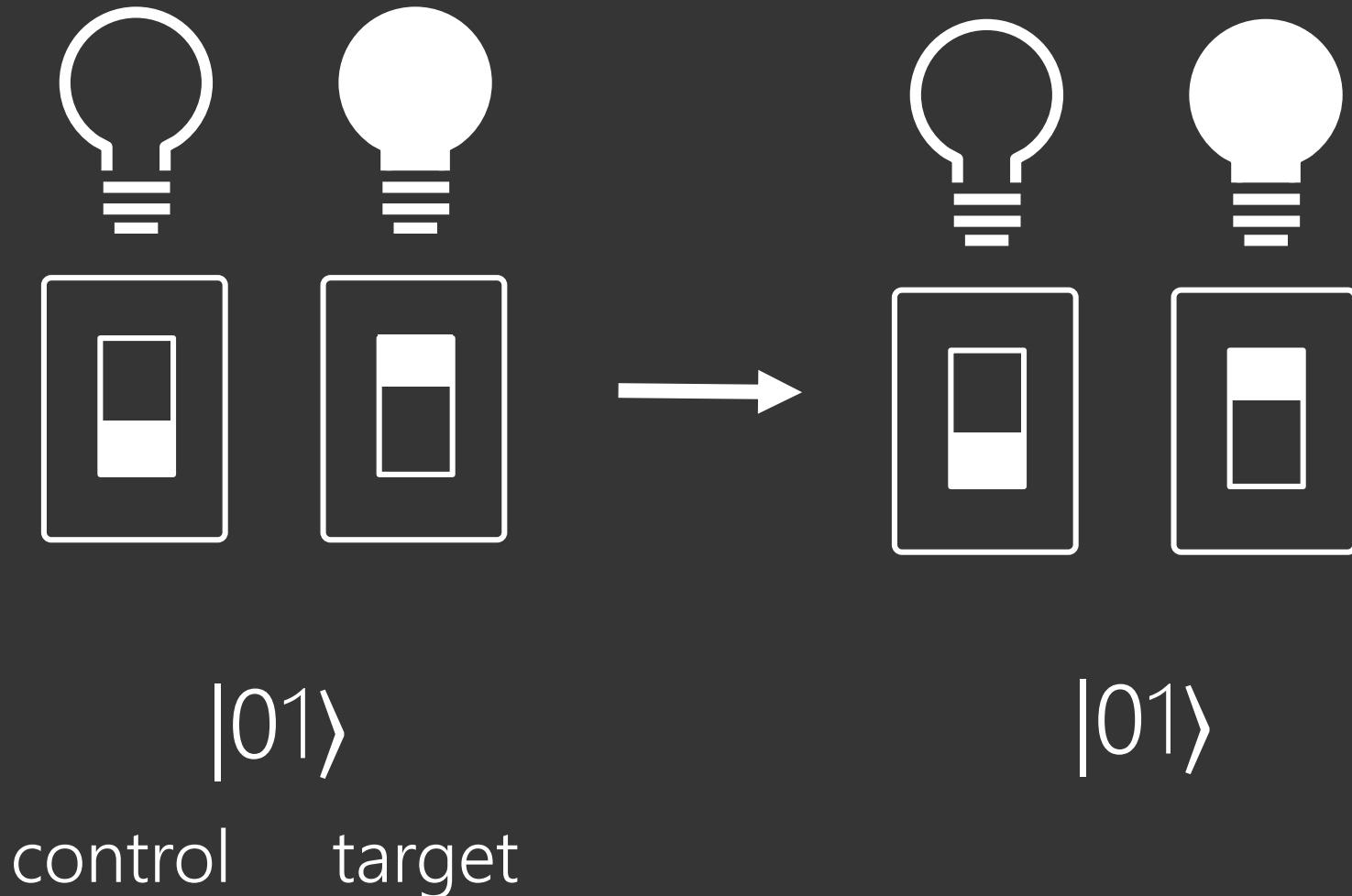


Quantum Operations - CNOT



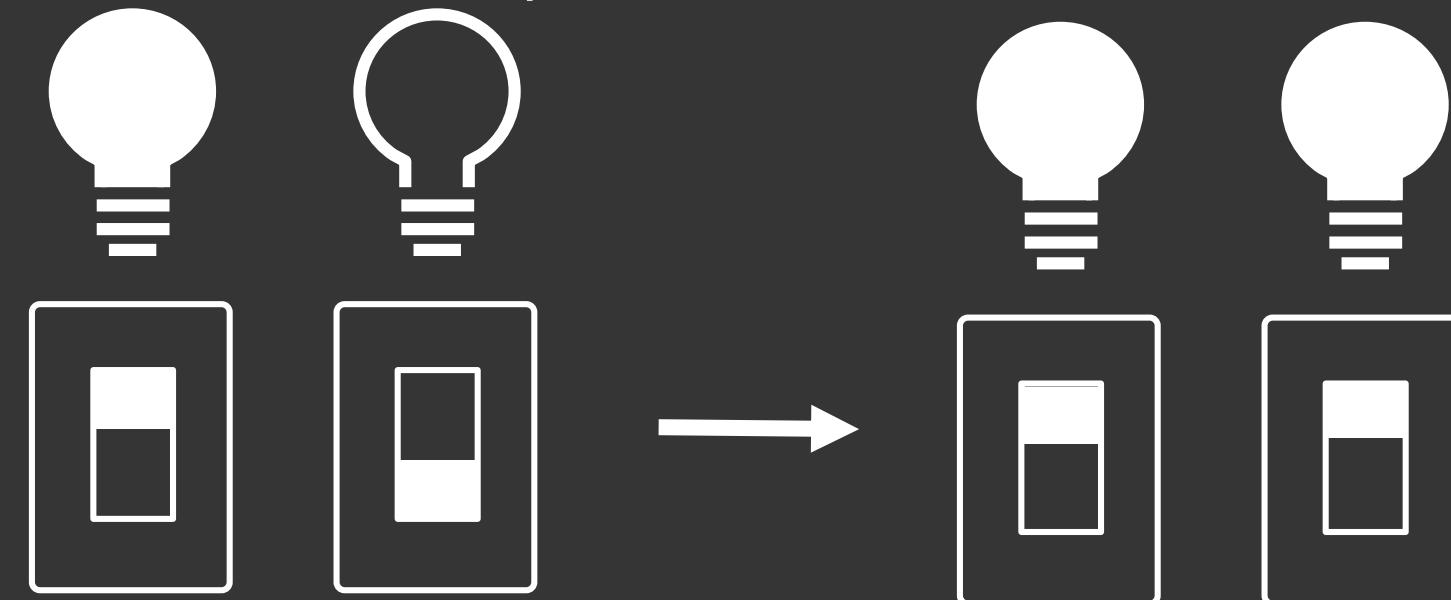
Input	Output
$ 00\rangle$	$ 00\rangle$

Quantum Operations - CNOT



Input	Output
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$

Quantum Operations - CNOT

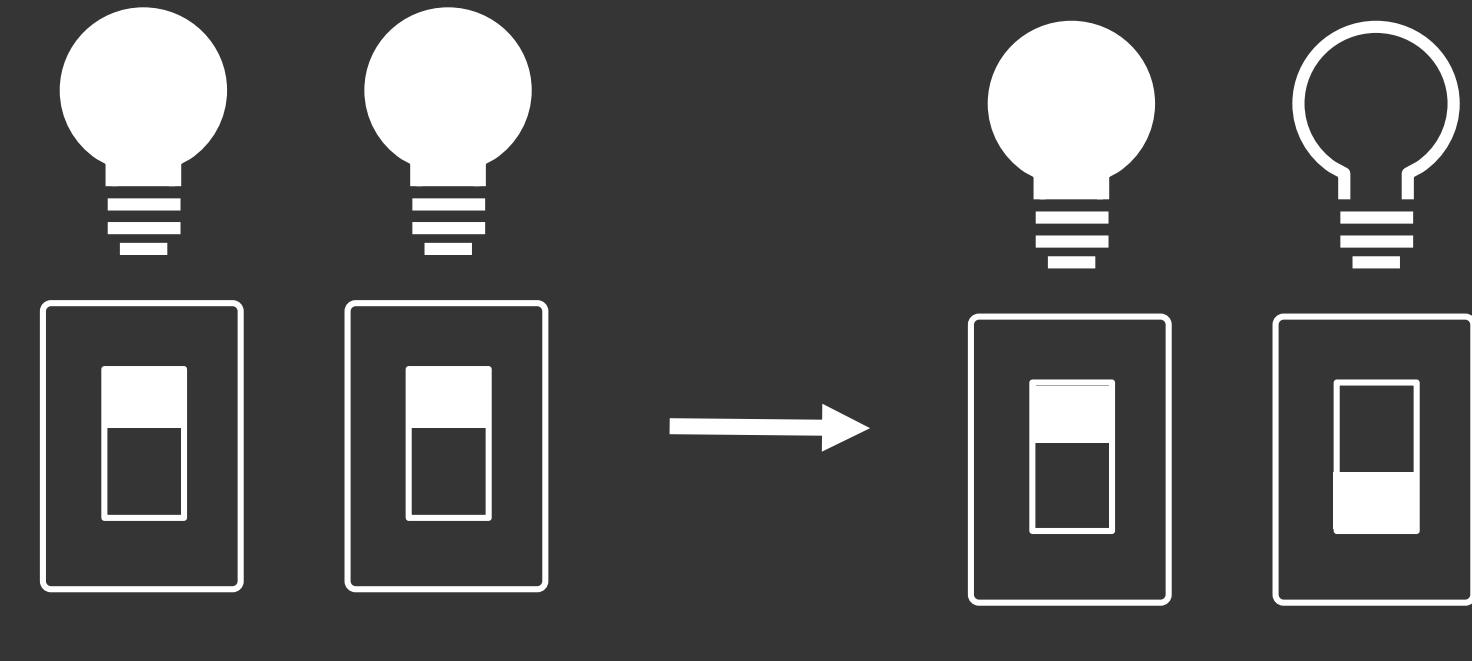


$|10\rangle$
control

$|11\rangle$

Input	Output
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$
$ 10\rangle$	$ 11\rangle$

Quantum Operations - CNOT



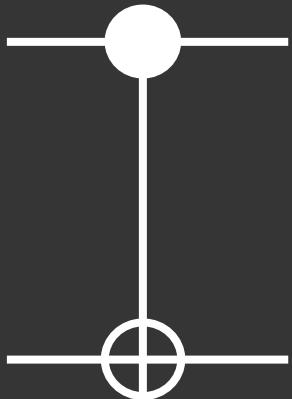
$|11\rangle$
control

$|10\rangle$

Input	Output
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$
$ 10\rangle$	$ 11\rangle$
$ 11\rangle$	$ 10\rangle$

Quantum Operations – CNOT

```
operation CNOT (control : Qubit, target : Qubit) : ()
```



Quantum Operations - Z

Input	Output
$ 0\rangle$	$ 0\rangle$
$ 1\rangle$	$- 1\rangle$

$$|0\rangle \longrightarrow |0\rangle$$

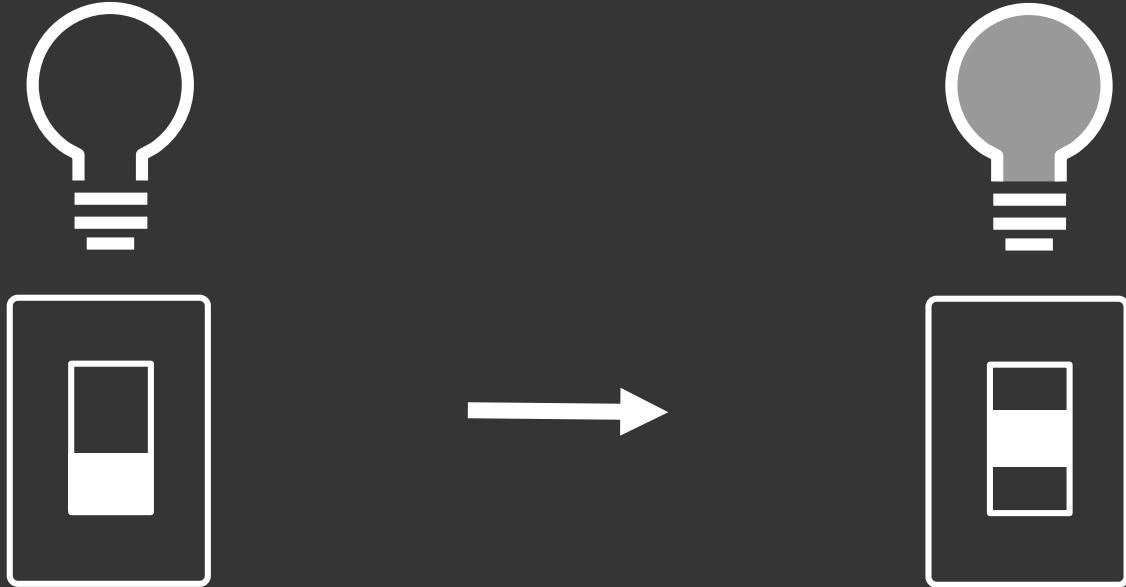
$$|1\rangle \longrightarrow -|1\rangle$$

Quantum Operations – Z

```
operation Z (qubit : Qubit) : ()
```

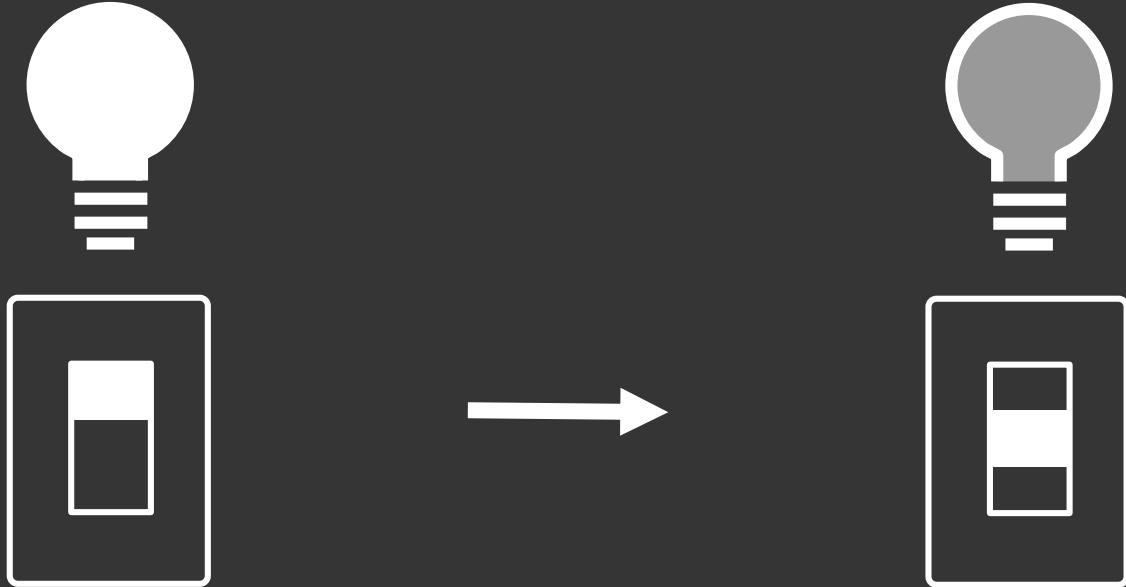


Quantum Operations - Hadamard

 $|0\rangle$

$$\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle$$

Quantum Operations - Hadamard



$|1\rangle$

$$\frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle$$

Quantum Operations – Hadamard

```
operation H (qubit : Qubit) : ()
```



Our Toolbox:

NOT

Input	Output
$ 0\rangle$	$ 1\rangle$
$ 1\rangle$	$ 0\rangle$

CNOT

Input	Output
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$
$ 10\rangle$	$ 11\rangle$
$ 11\rangle$	$ 10\rangle$

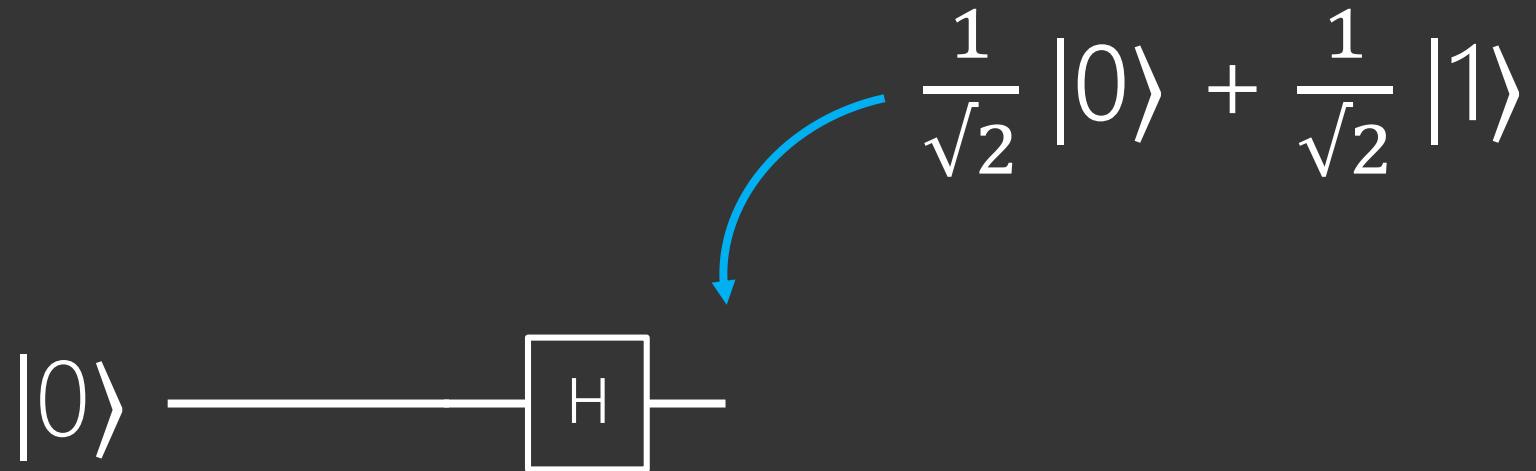
Z

Input	Output
$ 0\rangle$	$ 0\rangle$
$ 1\rangle$	$- 1\rangle$

Hadamard

Input	Output
$ 0\rangle$	$\frac{1}{\sqrt{2}} 0\rangle + \frac{1}{\sqrt{2}} 1\rangle$
$ 1\rangle$	$\frac{1}{\sqrt{2}} 0\rangle - \frac{1}{\sqrt{2}} 1\rangle$

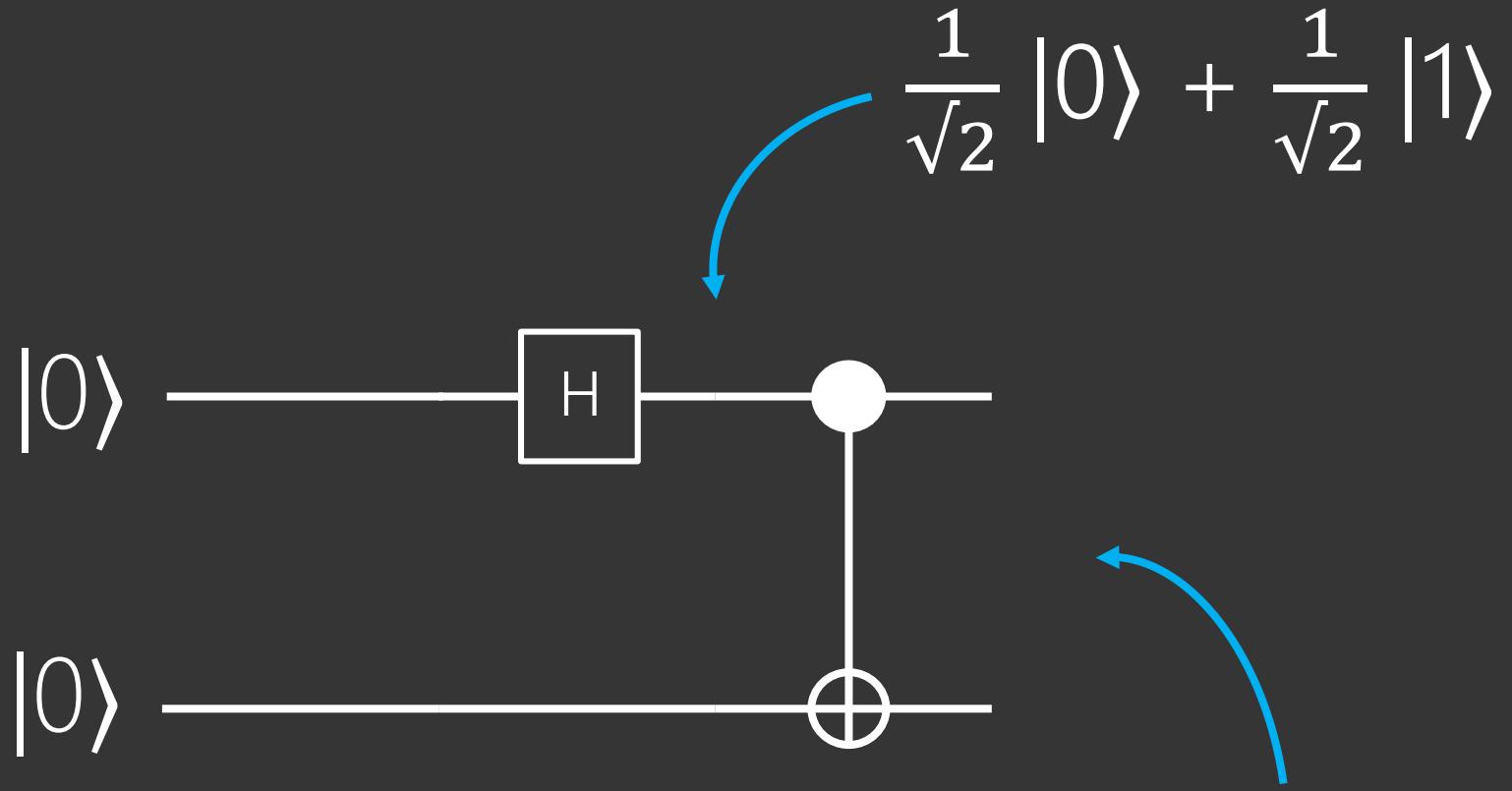
Entanglement



$|0\rangle$

Input	Output
$ 0\rangle$	$\frac{1}{\sqrt{2}} 0\rangle + \frac{1}{\sqrt{2}} 1\rangle$
$ 1\rangle$	$\frac{1}{\sqrt{2}} 0\rangle - \frac{1}{\sqrt{2}} 1\rangle$

Entanglement



$$(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle)|0\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$$

Input	Output
$ 00\rangle$	$ 00\rangle$
$ 01\rangle$	$ 01\rangle$
$ 10\rangle$	$ 11\rangle$
$ 11\rangle$	$ 10\rangle$

Entanglement - Summary

$$|00\rangle \longrightarrow \frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle$$

Quantum Teleportation - Motivation

$|\psi\rangle$

Physically?

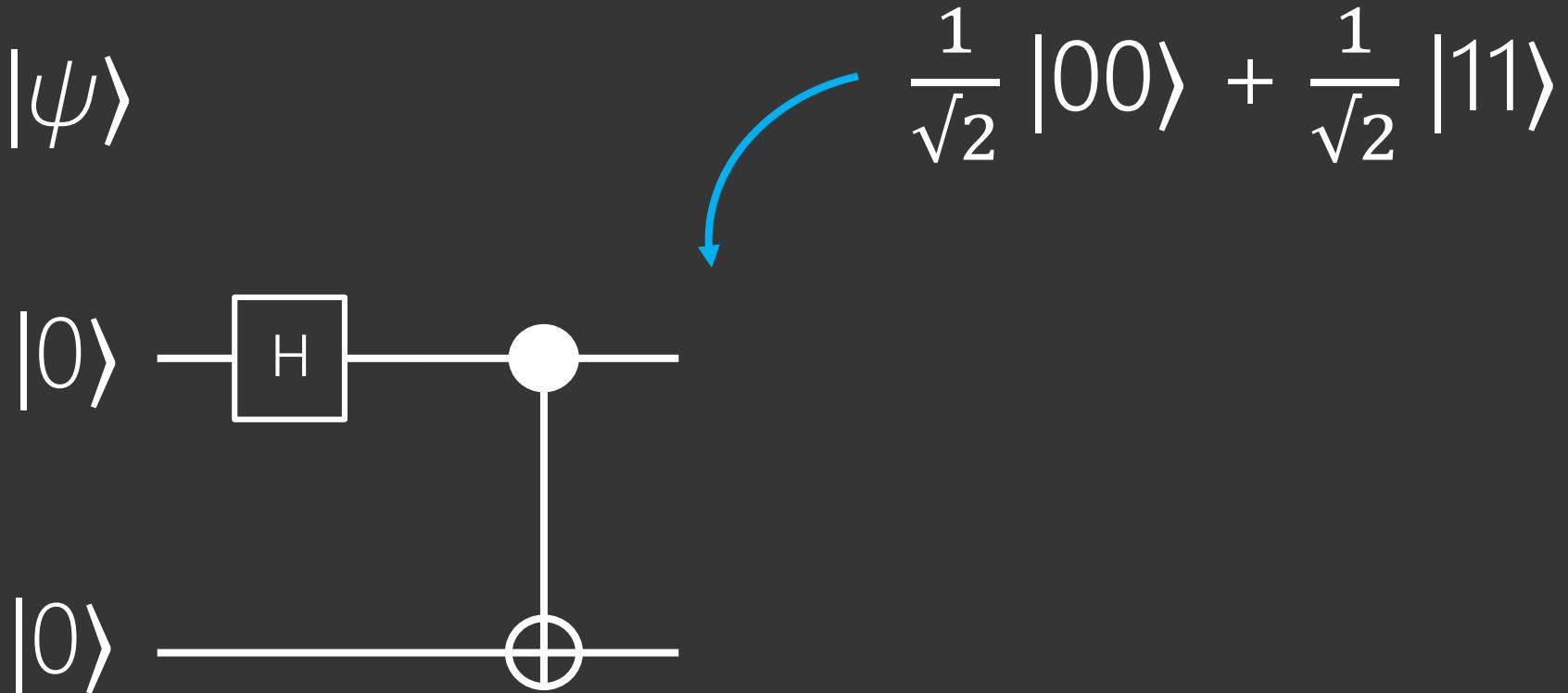
$|0\rangle$

Classically?

$|0\rangle$

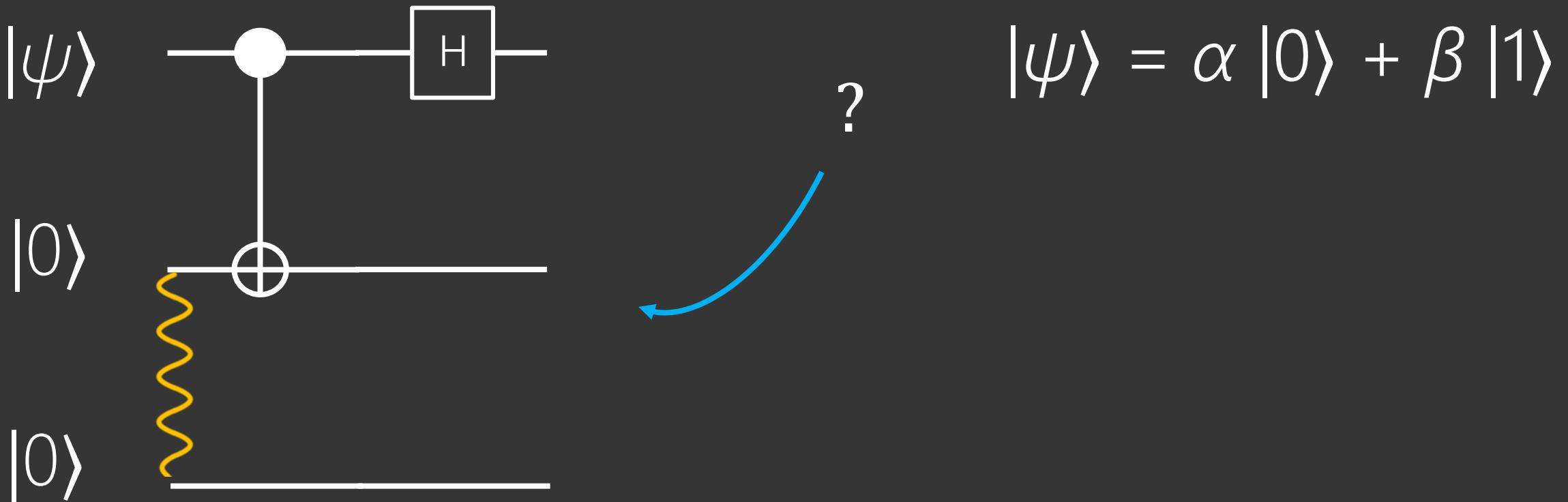
Quantum teleportation!

Quantum Teleportation - Explained



Step 0: Entangle our qubits

Quantum Teleportation - Explained

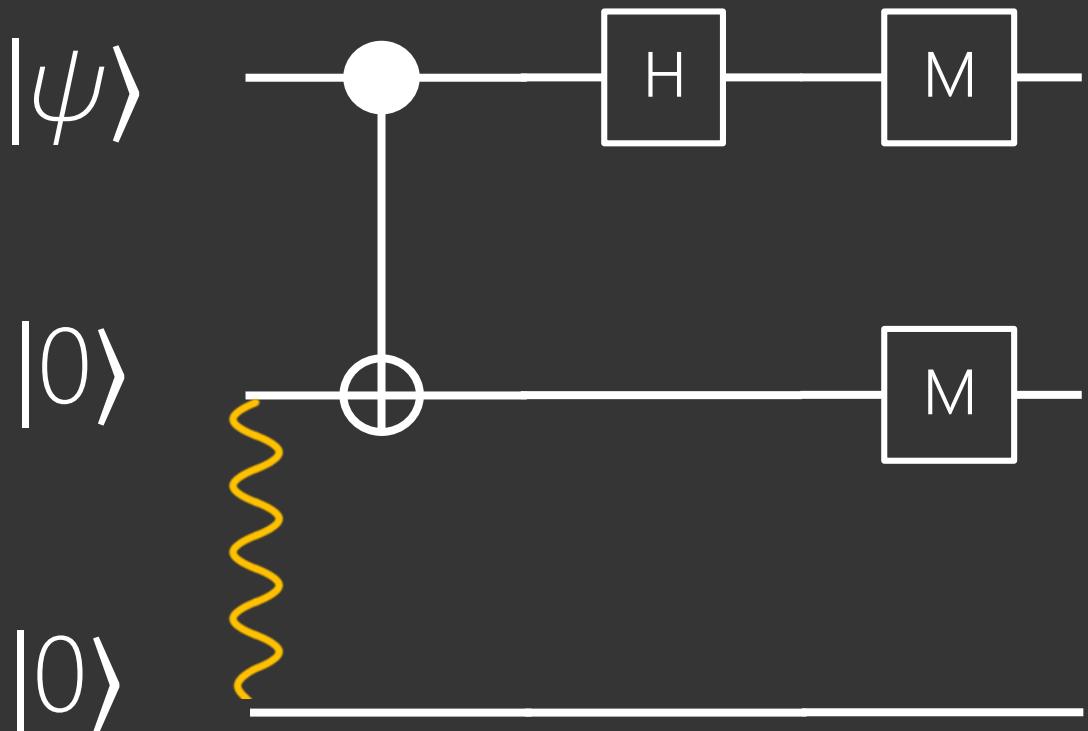


Step 1: Entangle my qubit with the state to be sent

Quantum Teleportation - Explained

$$\frac{1}{2} [|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle) + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)]$$

Quantum Teleportation - Explained



Step 2: Measure the first two qubits (mine, message)

Quantum Teleportation - Explained

$$\frac{1}{2} \underbrace{[|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle) + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)]}_{\text{Blue bracket and arrow}} \rightarrow |00\rangle(\alpha|0\rangle + \beta|1\rangle)$$
$$|00\rangle(\alpha|0\rangle + \beta|1\rangle) \underbrace{\qquad\qquad\qquad}_{\text{Yellow bracket and double arrows}} |\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Step 3: Interpret the result

Quantum Teleportation - Explained

$$\frac{1}{2} [|00\rangle(\alpha|0\rangle + \beta|1\rangle) + \underline{|01\rangle(\alpha|1\rangle + \beta|0\rangle)} + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + |11\rangle(\alpha|1\rangle - \beta|0\rangle)]$$

Apply a NOT gate

$$|01\rangle(\alpha|1\rangle + \beta|0\rangle)$$

\uparrow \uparrow

$$\underline{\hspace{10em}}$$
$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

Step 3: Interpret the result

Quantum Teleportation - Explained

$$\frac{1}{2} [|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle) + \underline{|10\rangle(\alpha|0\rangle - \beta|1\rangle)} + |11\rangle(\alpha|1\rangle - \beta|0\rangle)]$$



Apply a Z gate

$$|10\rangle(\alpha|0\rangle - \beta|1\rangle)$$



$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

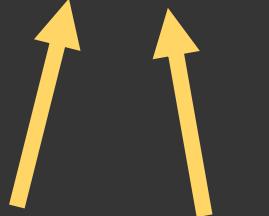
Step 3: Interpret the result

Quantum Teleportation - Explained

$$\frac{1}{2} [|00\rangle(\alpha|0\rangle + \beta|1\rangle) + |01\rangle(\alpha|1\rangle + \beta|0\rangle) + |10\rangle(\alpha|0\rangle - \beta|1\rangle) + \underline{|11\rangle(\alpha|1\rangle - \beta|0\rangle)}$$

Apply a Z gate and NOT gate

$$|11\rangle(\alpha|1\rangle - \beta|0\rangle)$$



$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

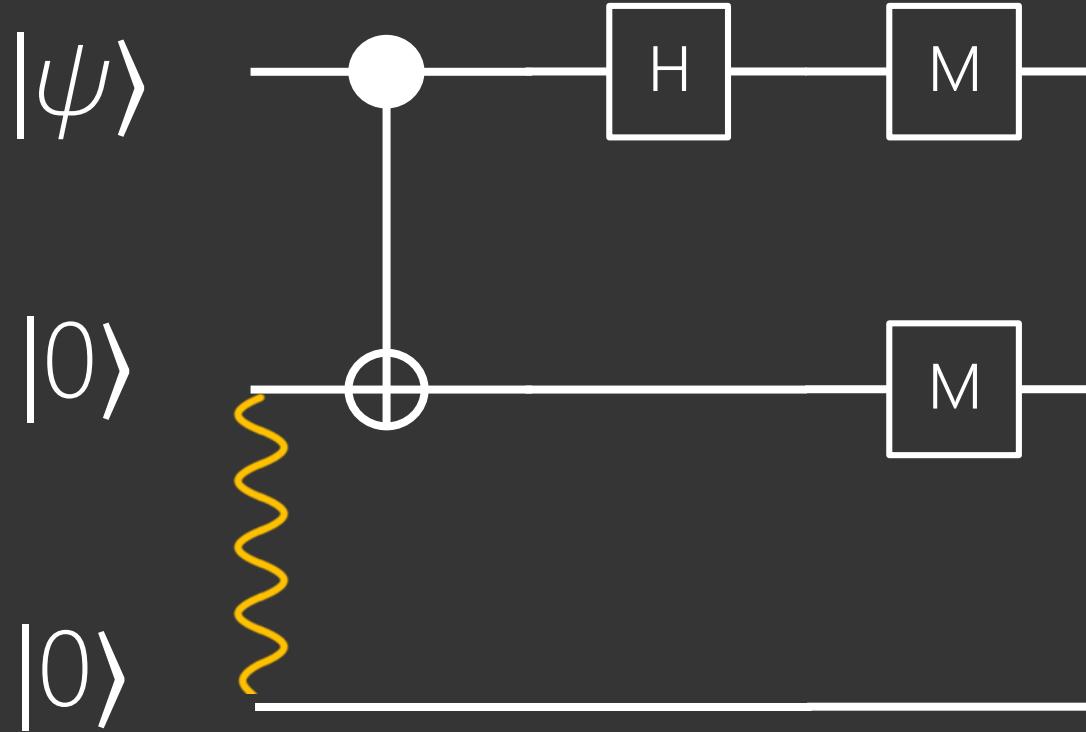
Step 3: Interpret the result

Quantum Teleportation - Summary

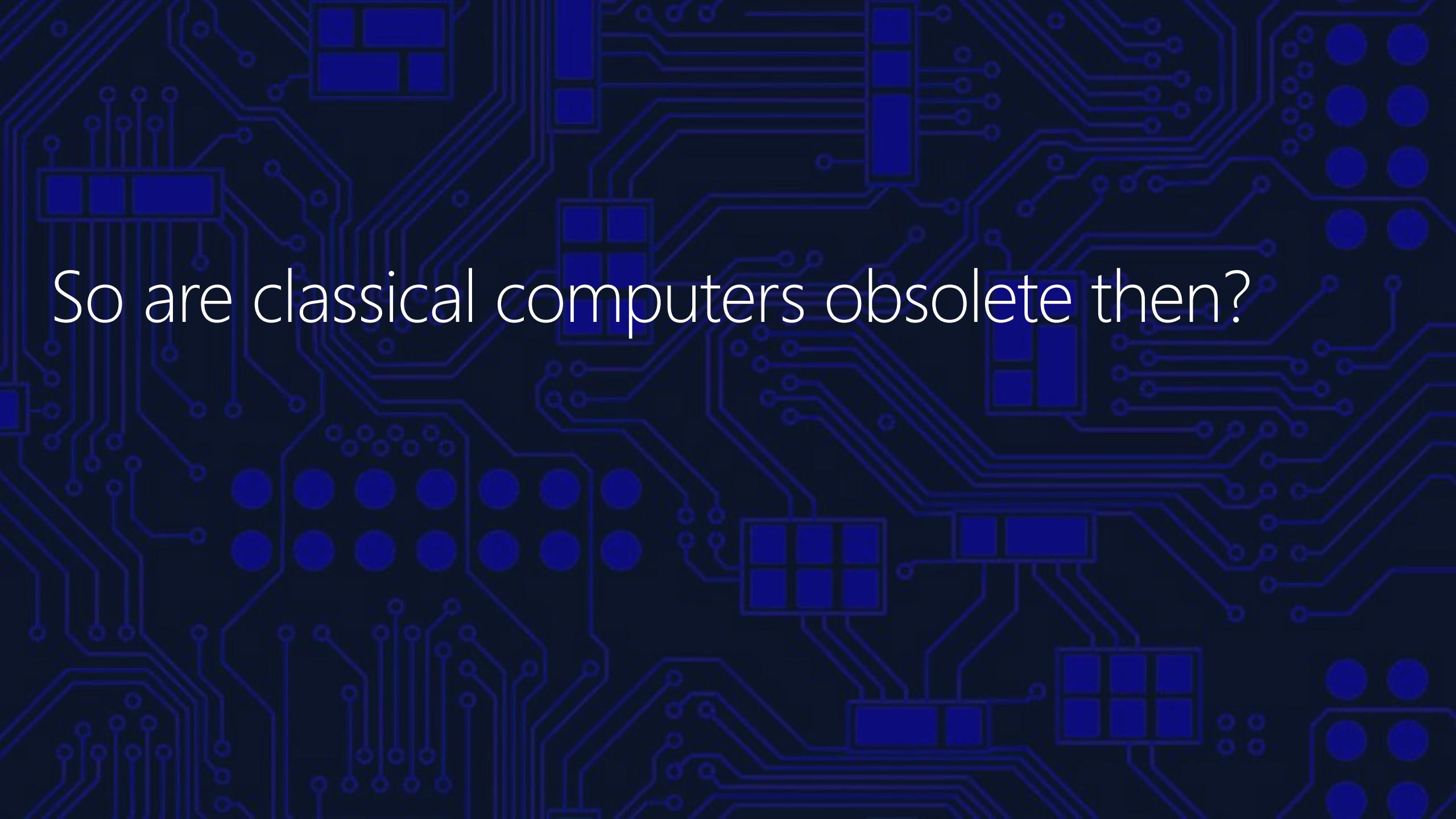
Measurement	Operation
$ 00\rangle$	Do nothing
$ 01\rangle$	Apply NOT
$ 10\rangle$	Apply Z
$ 11\rangle$	Apply NOT, Z

Step 4: Apply the gates

Quantum Teleportation – Code!



aka.ms/QuantumComp



So are classical computers obsolete then?

EXPLORER

OPEN EDITORS

Shor.qs IntegerFactorization

SAMPLES

- › .vscode
- › AdiabaticIsing
- › BitFlipCode
- › DatabaseSearch
- › H2SimulationCmdLine
- › H2SimulationGUI
- › IntegerFactorization
- › bin

- › obj
- ⚙ App.config
- ❼ IntegerFactorization.csproj
- ❷ Program.cs
- ❶ README.md

Shor.qs

- › IsingGenerators
- › IsingPhaseEstimation
- › IsingTrotterEvolution
- › Measurement
- › PhaseEstimation
- › PythonInterop
- › ReversibleLogicSynthesis
- › SimpleAlgorithms
- › SimpleIsing
- › Teleportation
- › UnitTesting
- ❶ README.md

Shor.qs x

```
27     /// # Input
28     /// ## number
29     /// An integer to be factored
30     /// ## useRobustPhaseEstimation
31     /// If set to true, we use Microsoft.Quantum.Canon.RobustPhaseEstimation and
32     /// Microsoft.Quantum.Canon.QuantumPhaseEstimation otherwise
33     ///
34     /// # Output
35     /// Pair of numbers p > 1 and q > 1 such that p·q = `number`
36 operation Shor ( number : Int, useRobustPhaseEstimation : Bool ) : (Int,Int) {
37     body
38     {
39         /> First check the most trivial case, if the provided number is even
40         if ( number % 2 == 0 ) {
41             Message("An even number has been passed; 2 is the factor.");
42             return (number / 2, 2);
43         }
44         /> Next try to guess a number co-prime to `number`
45         // Get a random integer in the interval [1,number-1]
46         let coprimeCandidate = RandomInt(number - 2) + 1;
47
48         // Check if the random integer indeed co-prime using
49         // Microsoft.Quantum.Canon.IsCoprime.
50         // If true use Quantum algorithm for Period finding.
51         if( IsCoprime(coprimeCandidate, number) ) {
52
53             // Print a message using Microsoft.Quantum.Primitive.Message
54             // indicating that we are doing something quantum.
55             Message($"Estimating period of {coprimeCandidate}");
56
57             // Call Quantum Period finding algorithm for
58             // `coprimeCandidate` mod `number`.
59             // Here we have a choice which Phase Estimation algorithm to use.
60             let period = EstimatePeriod(coprimeCandidate, number, useRobustPhaseEstimation);
61         }
62     }
63 }
```

What's in it for me?

>The developer's perspective

EXPLORER

OPEN EDITORS

Shor.qs IntegerFactorization

SAMPLES

- › .vscode
- › AdiabaticIsing
- › BitFlipCode
- › DatabaseSearch
- › H2SimulationCmdLine
- › H2SimulationGUI
- › HubbardSimulation

IntegerFactorization

- › bin
- › obj
- ⚙ App.config
- RSS IntegerFactorization.csproj
- CS Program.cs

README.md

Shor.qs

- › IsingGenerators
- › IsingPhaseEstimation
- › IsingTrotterEvolution
- › Measurement
- › PhaseEstimation

- › PythonInterop
- › ReversibleLogicSynthesis

- › SimpleAlgorithms

- › SimpleIsing

- › Teleportation

- › UnitTesting

README.md

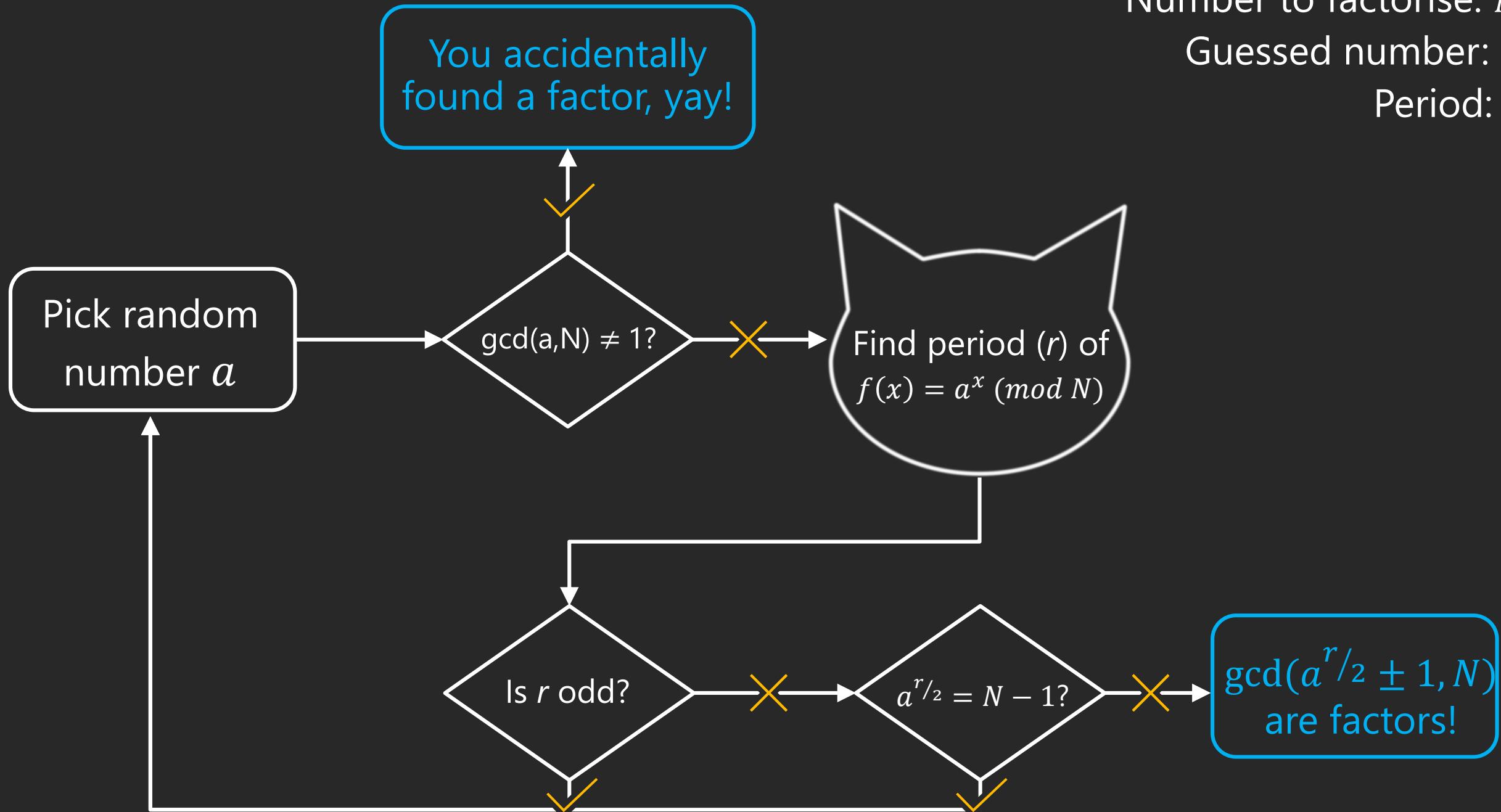
Shor.qs

```
27    /// # Input
28    /// ## number
29    /// An integer to be factored
30    /// ## useRobustPhaseEstimation
31    /// If set to true, we use Microsoft.Quantum.Canon.RobustPhaseEstimation and
32    /// Microsoft.Quantum.Canon.QuantumPhaseEstimation otherwise
```

1. Combine large prime numbers.
2. Use composite number as public key.
3. Share public key.
4. Encrypt data using public key.
5. Send encrypted data.
6. Decrypt data with private key.
7. ???
8. Profit.

```
33    ///
34    //## Output
35    //## Returns a tuple of two integers, each being a factor of 'number'.
36    operation Shor( number : Int, useRobustPhaseEstimation : Bool ) : (Int,Int) {
37        {
38            // First check the most trivial case, if the provided number is even
39            if( number % 2 == 0 ) {
40                Message("An even number has been passed; 2 is the factor.");
41            }
42            // Next try to guess a number co-prime to `number`
43            // Get a random integer in the interval [1,number-1]
44            let coprimeCandidate = Random.GetInt(1, number - 1);
45            let period = EstimatePeriod(coprimeCandidate, number);
46            if( IsCoprime(coprimeCandidate, number) ) {
47                // Check if the random integer indeed co-prime using
48                // Microsoft.Quantum.Canon.IsCoprime.
49                // If true use Quantum algorithm for Period finding.
50                if( IsCoprime(coprimeCandidate, number) ) {
51                    // Print a message using Microsoft.Quantum.Primitive.Message
52                    // indicating that we are doing something quantum.
53                    Message($"Estimating period of {coprimeCandidate}");
54
55
56
57
58                    // Call Quantum Period finding algorithm for
59                    // `coprimeCandidate` mod `number`.
60                    // Here we have a choice which Phase Estimation algorithm to use.
61                    let period = EstimatePeriod(coprimeCandidate, number, useRobustPhaseEstimation);
```

Number to factorise: N
Guessed number: a
Period: r



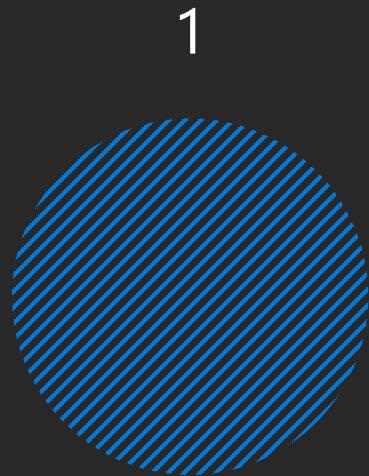
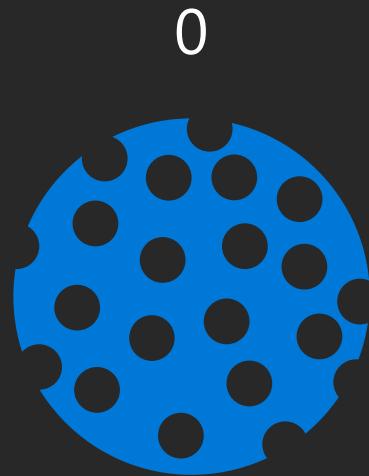
Satya Nadella (CEO)



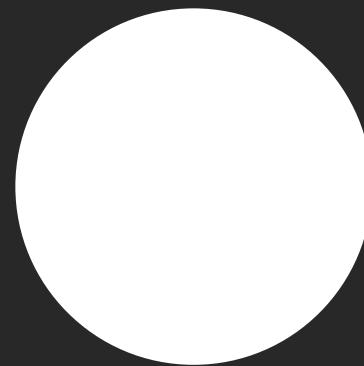
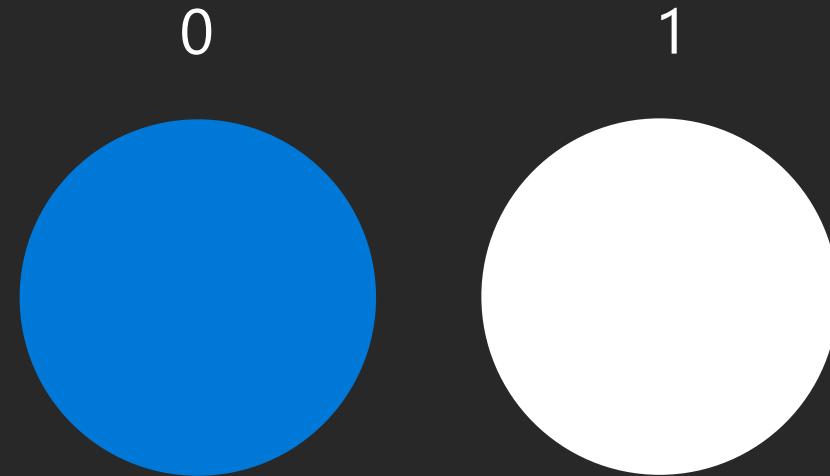
Frances



Anita



Spotty/stripy mode (basis)



Blue/white mode (basis)

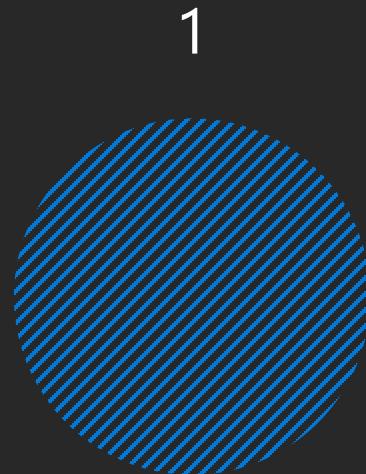
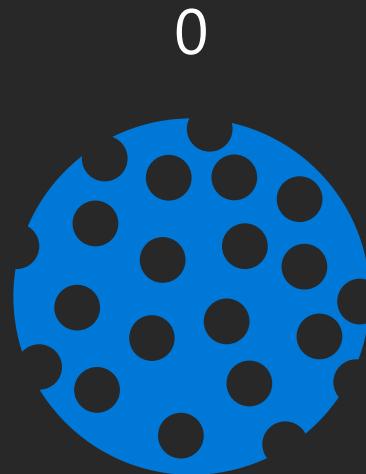
Frances



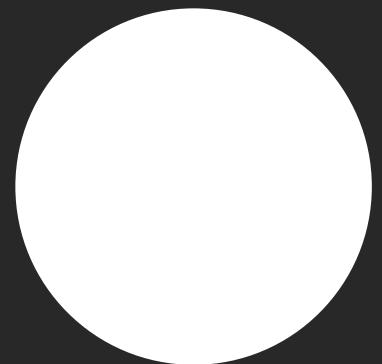
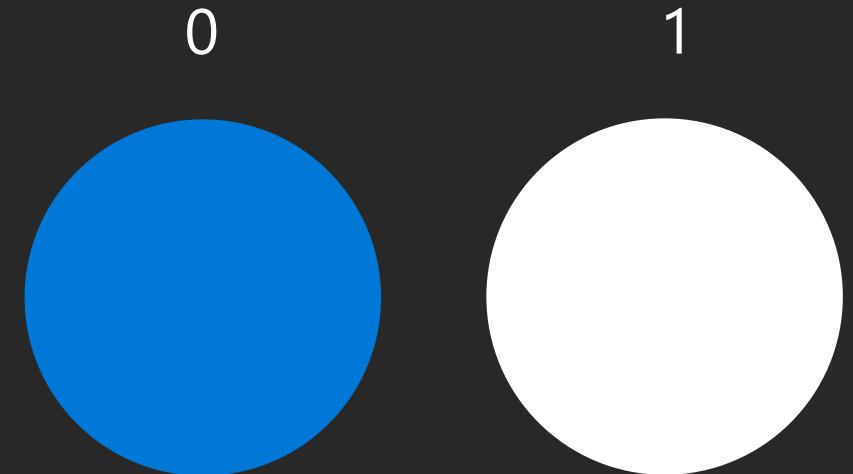
Key

Anita





Spotty/stripy mode (basis)

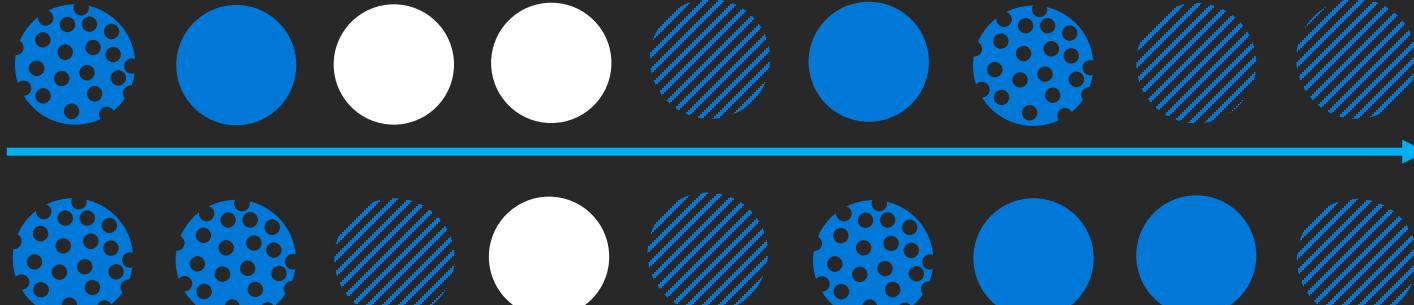


Blue/white mode (basis)

Frances

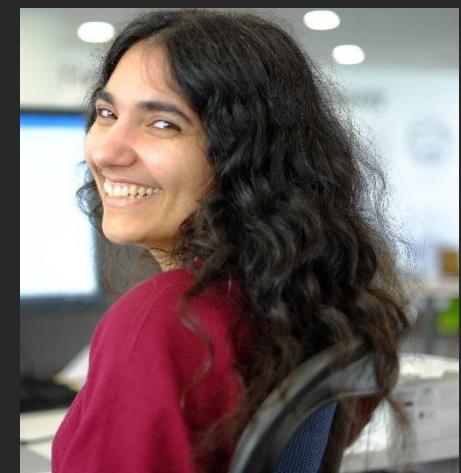


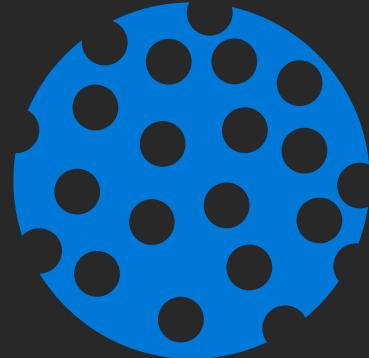
Frances sends



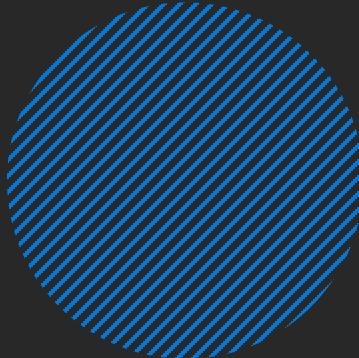
Anita measures

Anita

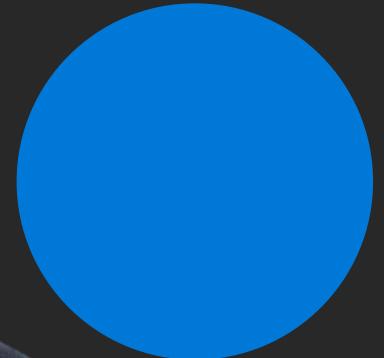




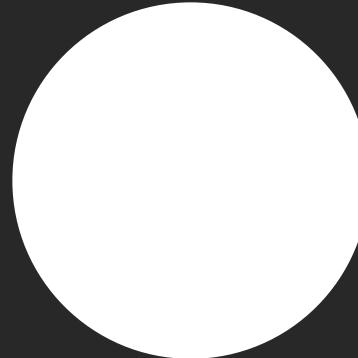
0



1



0



1

Spotty/stripy mode (basis)

Blue/white mode (basis)

Frances



Anita



Grover's search:



$N/2$ on average, N at worst – hence $O(N)$



Using Grover's: $O(\sqrt{N})$

QUANTUM COMPUTING

- What is quantum computing? Why is it important?
- How do we build a quantum computer?
- What kinds of physical systems are already in development?
- What is Q#/the QDK and why would you use it?
- Can we use Q# to teleport a quantum state?

aka.ms/QuantumComp

Q&A?

Q. Do you follow Microsoft Quantum on Twitter?

A. No? Go to aka.ms/QuantumTwitter

Q. Do you receive the Microsoft Quantum newsletter?

A. No? Go to aka.ms/QuantumNewsletter

Q. Interested in learning more about quantum computing from the ground up?

A. Yes? Go to aka.ms/QuantumAdventures

Bonus! github.com/frtibble/QuantumWorkshop



Anita Ramanan | Frances Tibble
<https://aka.ms/quantumadventures>
@whywontitbuild | @frances_tibble
anraman@microsoft.com | frtibble@microsoft.com