Microsoft Research

Each year Microsoft Research hosts hundreds of influential speakers from around the world including leading scientists, renowned experts in technology, book authors, and leading academics, and makes videos of these lectures freely available.

Quantum Computing for Computer Scientists

The gate quantum computation model

Why learn quantum computing?

- Quantum supremacy expected this year
  - Microsoft, Google, Intel, IBM all investing in quantum computer development
- Several exciting applications already known
  - Efficiently factor large composite numbers, breaking RSA encryption (Shor's algorithm, 1994)
  - Search an unordered list in $O(\sqrt{n})$ time (Grover's algorithm, 1996)
  - Believed exponential speedup in simulating quantum mechanical systems
- Intellectually interesting – quantum mechanics is outside your intuition!
  - Get a small glimpse of what you don't know you don't know

Learning objectives

- Representing computation with basic linear algebra (vectors and matrices)
- Qbits, superposition, and quantum logic gates
- The simplest problem where a quantum computer beats a classical computer
- Bonus topics: quantum entanglement and teleportation

Representing classical bits as a vector

One bit with the value 0, also written as $|0\rangle$ (Dirac vector notation)

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

One bit with the value 1, also written as $|1\rangle$

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

Review: matrix multiplication

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} ax + by \\ cx + dy \end{pmatrix}$$

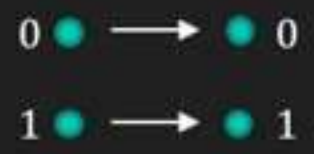
$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} ax + by + cz \\ dx + ey + fz \\ gx + hy + iz \end{pmatrix}$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} w & x \\ y & z \end{pmatrix} = \begin{pmatrix} aw + by & ax + bz \\ cw + dy & cx + dz \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

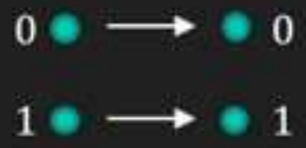
Operations on one classical bit (cbit)

Identity	$f(x) = x$		$\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}1 & 0\\0 & 1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}$
Negation	$f(x) = \neg x$		$\begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}0 & 1\\1 & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}$
Constant-0	$f(x) = 0$		$\begin{pmatrix}1 & 1\\0 & 0\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}1 & 1\\0 & 0\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}$
Constant-1	$f(x) = 1$		$\begin{pmatrix}0 & 0\\1 & 1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}0 & 0\\1 & 1\end{pmatrix}\begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}$

Reversible computing

- Reversible means given the operation and output value, you can find the input value
  - For $Ax = b$, given b and A, you can uniquely find x
- Operations which permute are reversible; operations which erase & overwrite are not
  - Identity and Negation are reversible
  - Constant-0 and Constant-1 are not reversible
- Quantum computers use only reversible operations, so we will only care about those
  - In fact, all quantum operators *are their own inverses*

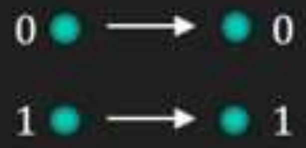
Operations on one classical bit (cbit)

Identity	$f(x) = x$		$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
Negation	$f(x) = \neg x$		$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
Constant-0	$f(x) = 0$		$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
Constant-1	$f(x) = 1$		$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

Reversible computing

- Reversible means given the operation and output value, you can find the input value
  - For $Ax = b$, given b and A, you can uniquely find x
- Operations which permute are reversible; operations which erase & overwrite are not
  - Identity and Negation are reversible
  - Constant-0 and Constant-1 are not reversible
- Quantum computers use only reversible operations, so we will only care about those
  - In fact, all quantum operators *are their own inverses*

Operations on one classical bit (cbit)

Identity	$f(x) = x$		$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
Negation	$f(x) = \neg x$		$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
Constant-0	$f(x) = 0$		$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
Constant-1	$f(x) = 1$		$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

Reversible computing

- Reversible means given the operation and output value, you can find the input value
  - For $Ax = b$, given b and A, you can uniquely find x
- Operations which permute are reversible; operations which erase & overwrite are not
  - Identity and Negation are reversible
  - Constant-0 and Constant-1 are not reversible
- Quantum computers use only reversible operations, so we will only care about those
  - In fact, all quantum operators *are their own inverses*

Review: tensor product of vectors

$$\begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} x_0 \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \\ x_1 \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} x_0 y_0 \\ x_0 y_1 \\ x_1 y_0 \\ x_1 y_1 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 2 \end{pmatrix} \otimes \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \\ 6 \\ 8 \end{pmatrix}$$

$$\begin{pmatrix} x_0 \\ x_1 \end{pmatrix} \otimes \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \otimes \begin{pmatrix} z_0 \\ z_1 \end{pmatrix} = \begin{pmatrix} x_0 y_0 z_0 \\ x_0 y_0 z_1 \\ x_0 y_1 z_0 \\ x_0 y_1 z_1 \\ x_1 y_0 z_0 \\ x_1 y_0 z_1 \\ x_1 y_1 z_0 \\ x_1 y_1 z_1 \end{pmatrix}$$

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

Representing multiple cbits

$$|00\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} \qquad |01\rangle = \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}$$

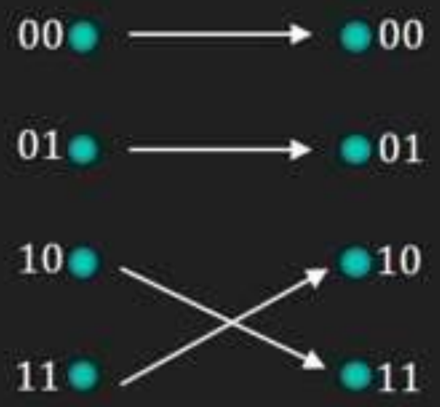
$$|10\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} \qquad |11\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

$$|4\rangle = |100\rangle = \begin{pmatrix}0\\1\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} \otimes \begin{pmatrix}1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\0\\1\\0\\0\\0\end{pmatrix}$$

- We call this tensored representation the **product state**
- We can **factor** the product state back into the **individual state** representation
- The product state of n bits is a vector of size 2^n

Operations on multiple cbits: CNOT

- Operates on pairs of bits, one of which is the "control" bit and the other the "target" bit
- If the control bit is 1, then the target bit is flipped
- If the control bit is 0, then the target bit is unchanged
- The control bit is always unchanged
- With most-significant bit as control and least-significant bit as target, action is as follows:



$$C = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Operations on multiple cbits: CNOT

$$C|10\rangle = C\left(\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix} = |11\rangle$$

$$C|11\rangle = C\left(\begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\0\\0\\1\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix} = \begin{pmatrix}0\\1\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix} = |10\rangle$$

Operations on multiple cbits: CNOT

$$C|00\rangle = C\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix} = |00\rangle$$

$$C|01\rangle = C\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\\1\\0\\0\end{pmatrix} = \begin{pmatrix}0\\1\\0\\0\end{pmatrix} = \begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}0\\1\end{pmatrix} = |01\rangle$$

Recap

- We represent classical bits in vector form as $\binom{1}{0}$ for 0 and $\binom{0}{1}$ for 1
- Operations on bits are represented by matrix multiplication on bit vectors
- Quantum computers only use reversible operations
- Multi-bit states are written as the tensor product of single-bit vectors
- The CNOT gate is a fundamental building block of reversible computing

Qbits and superposition

- Surprise! We've actually been using qbits all along!
- The cbit vectors we've been using are just special cases of qbit vectors
- A qbit is represented by $\binom{a}{b}$ where a and b are Complex numbers and $||a||^2 + ||b||^2 = 1$
  - The cbit vectors $\binom{1}{0}$ and $\binom{0}{1}$ fit within this definition
  - Don't worry! For this presentation, we'll only use familiar Real numbers.
- Example qbit values:

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \quad \begin{pmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} \end{pmatrix} \quad \begin{pmatrix} -1 \\ 0 \end{pmatrix} \quad \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{pmatrix}$$

Qbits and superposition

- How can a qbit to have a value which is not 0 or 1? This is called superposition.
- Superposition means the qbit is both 0 and 1 and the same time
- When we **measure** the qbit, it **collapses** to an actual value of 0 or 1
  - We usually do this at the end of a quantum computation to get the result
- If a qbit has value $\binom{a}{b}$ then it collapses to 0 with probability $\|a\|^2$ and 1 with probability $\|b\|^2$
  - For example, qbit $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$ has a $\left\|\frac{1}{\sqrt{2}}\right\|^2 = \frac{1}{2}$ chance of collapsing to 0 or 1 (coin flip)
  - The qbit $\binom{1}{0}$ has a 100% chance of collapsing to 0, and $\binom{0}{1}$ has a 100% chance of collapsing to 1

Qbits and superposition

- Multiple qbits are similarly represented by the tensor product $\binom{a}{b} \otimes \binom{c}{d} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix}$
  - Note that $\|ac\|^2 + \|ad\|^2 + \|bc\|^2 + \|bd\|^2 = 1$
- For example, the system $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$ (note that $\left\|\frac{1}{2}\right\|^2 = \frac{1}{4}$, and $\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = 1$)
  - There's a ¼ chance each of collapsing to $|00\rangle, |01\rangle, |10\rangle$, or $|11\rangle$

Operations on qbits

- How do we operate on qbits? The same way we operate on cbits: with matrices!
- All the matrix operators we've seen also work on qbits (bit flip, CNOT, etc.)
- Matrix operators model the effect of some device which manipulates qbit spin/polarization without measuring and collapsing it

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{pmatrix}$$

- There are several important matrix operators which only make sense in a quantum context

Qbits and superposition

- Multiple qbits are similarly represented by the tensor product $\binom{a}{b} \otimes \binom{c}{d} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix}$
  - Note that $\|ac\|^2 + \|ad\|^2 + \|bc\|^2 + \|bd\|^2 = 1$
- For example, the system $\begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \\ \frac{1}{2} \end{pmatrix}$ (note that $\left\|\frac{1}{2}\right\|^2 = \frac{1}{4}$, and $\frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} = 1$)
  - There's a ¼ chance each of collapsing to $|00\rangle, |01\rangle, |10\rangle$, or $|11\rangle$

Operations on qbits

- How do we operate on qbits? The same way we operate on cbits: with matrices!
- All the matrix operators we've seen also work on qbits (bit flip, CNOT, etc.)
- Matrix operators model the effect of some device which manipulates qbit spin/polarization without measuring and collapsing it

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{2} \\ \frac{\sqrt{3}}{2} \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{3}}{2} \\ \frac{1}{2} \end{pmatrix}$$

- There are several important matrix operators which only make sense in a quantum context

The Hadamard gate

- The Hadamard gate takes a 0- or 1-bit and puts it into exactly equal superposition

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

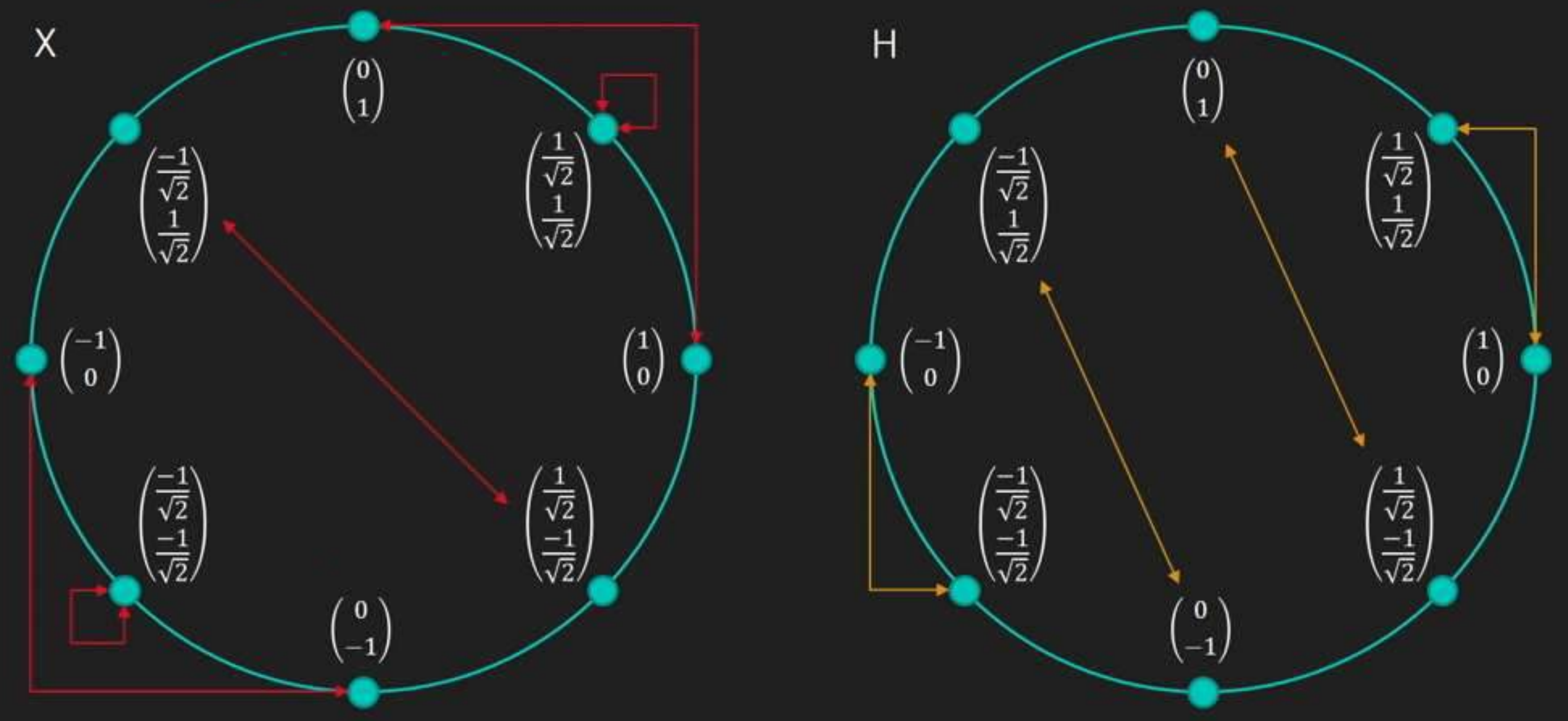
The Hadamard gate

- The Hadamard gate also takes a qbit in exactly-equal superposition, and transforms it into a 0- or 1-bit! (This should be unsurprising – remember operations are their own inverse!)

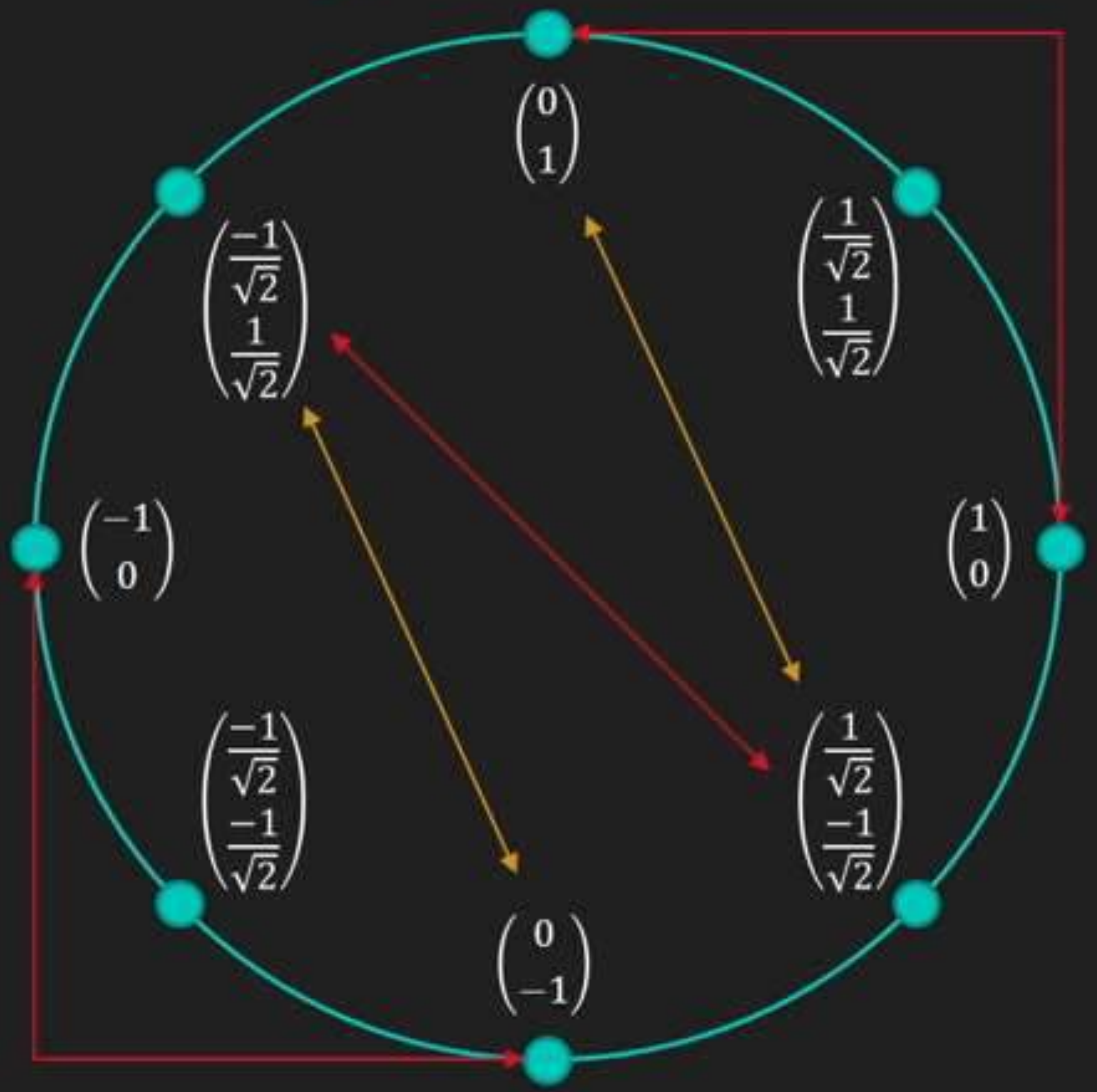
$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{-1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

- We can transition out of superposition without measurement!
- We can thus structure quantum computation deterministically instead of probabilistically

The unit circle state machine



The unit circle state machine



Recap

- Cbits are just a special case of qbits, which are 2-vectors of Complex numbers
- Qbits can be in superposition, and are probabilistically collapsed to cbits by measurement
- Multi-qbit systems are tensor products of single-qbit systems, like with cbits
- Matrices represent operations on qbits, same as with cbits
- The Hadamard gate takes 0- and 1-bits to equal superposition, and back
- We can think of qbits and their operations as forming a state machine on the unit circle
  - Actually the unit sphere if we use complex numbers

The Deutsch oracle

- Imagine someone gives you a black box containing a function on one bit
  - Recall! What are the four possible functions on one bit?
- You don't know which function is inside the box, but can try inputs and see outputs
- How many queries would it take to determine the function on a classical computer?
- How many on a quantum computer?

The Deutsch oracle

- What if you want to check whether the unknown function is constant, or variable?
  - Constant-0 & constant-1 are constant, identity & negation are variable
- How many queries would it take on a classical computer?
- How many on a quantum computer?

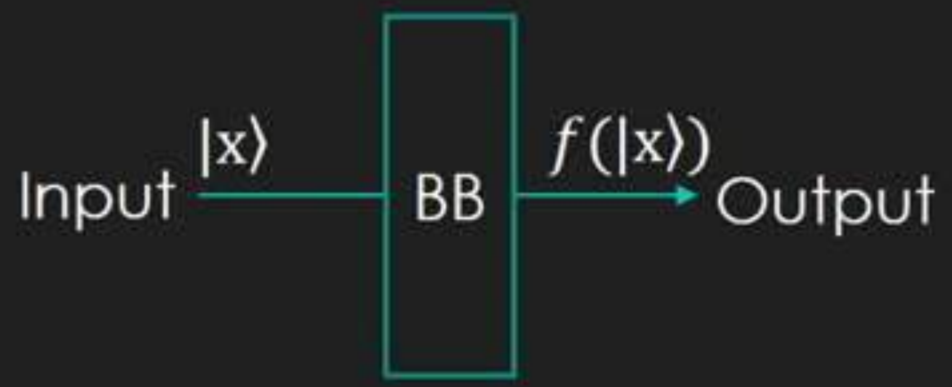
The Deutsch oracle

- How can it be done in a single query!?
- We can do it with the magic of superposition!
- First, we have to define what each of the four functions look like on a quantum computer
  - We have an immediate problem with the constant functions

The Deutsch oracle

- How do we write nonreversible functions in a reversible way?
- Common hack: add an additional **output qbit** to which the function action is applied
- We thus have to rewire our black box:

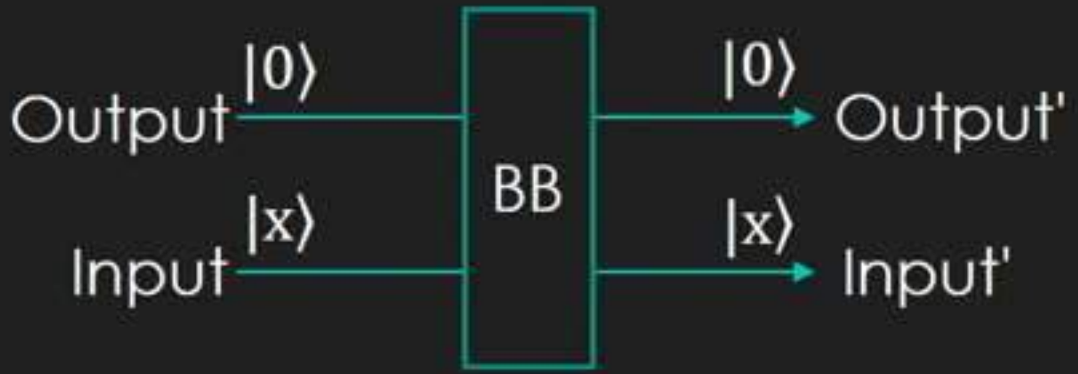
Before:

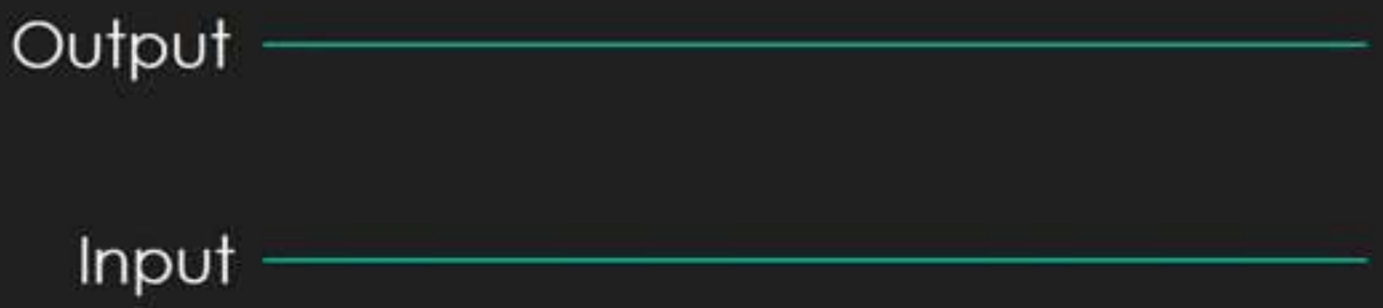


After:

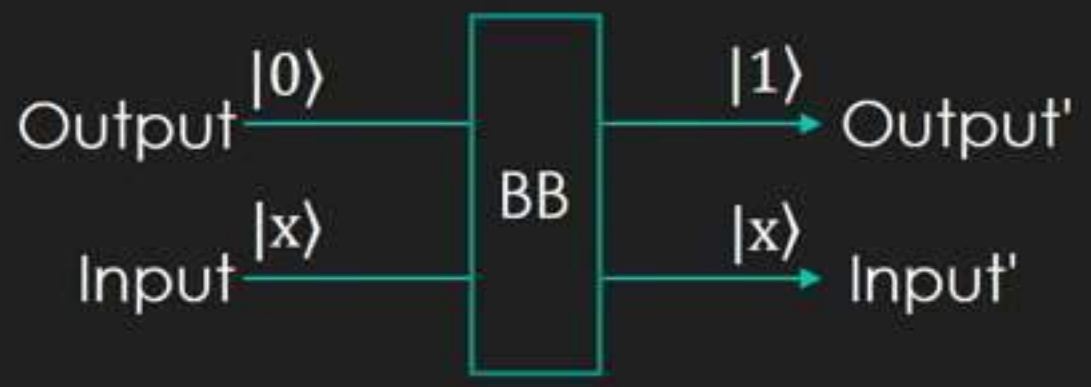
- The black box leaves the **input qbit** unchanged, writing function output to **output qbit**

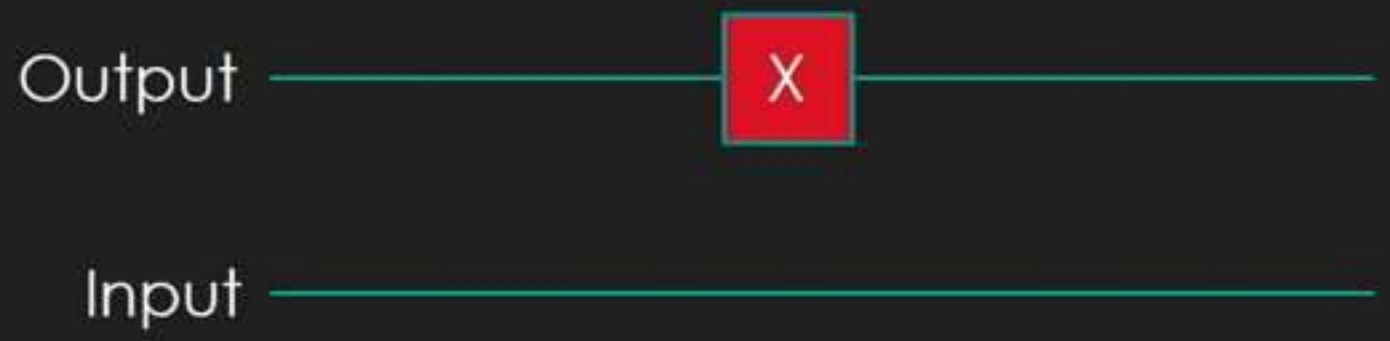
The Deutsch oracle: constant-0





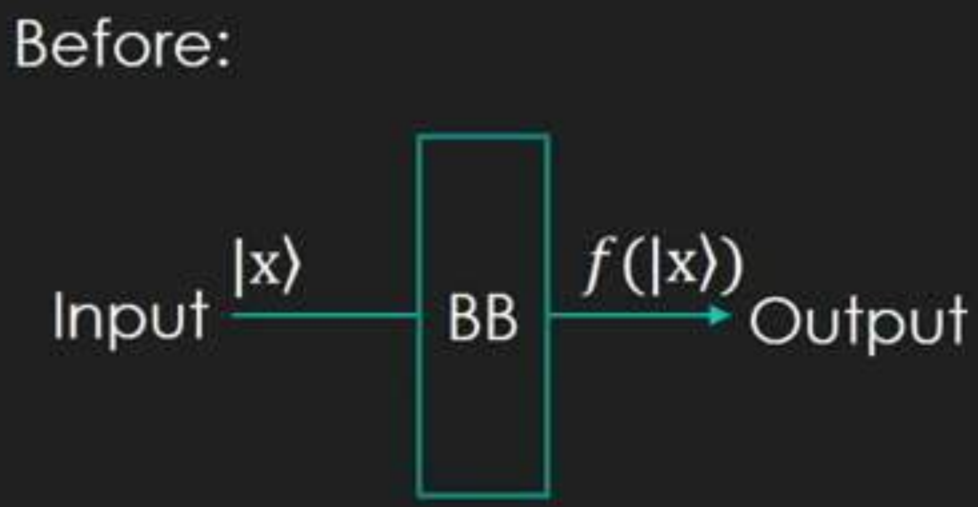
The Deutsch oracle: constant-1

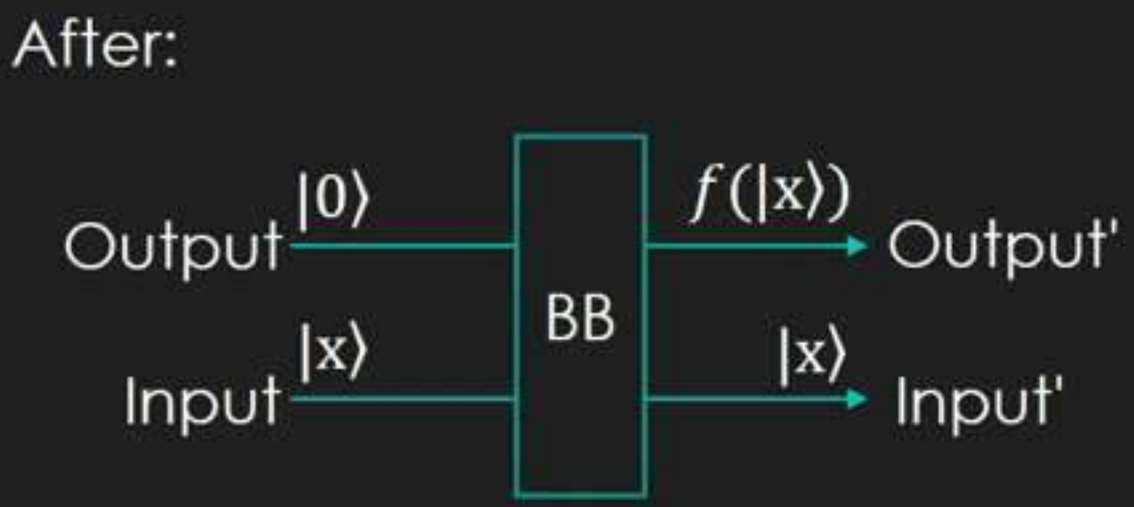




The Deutsch oracle

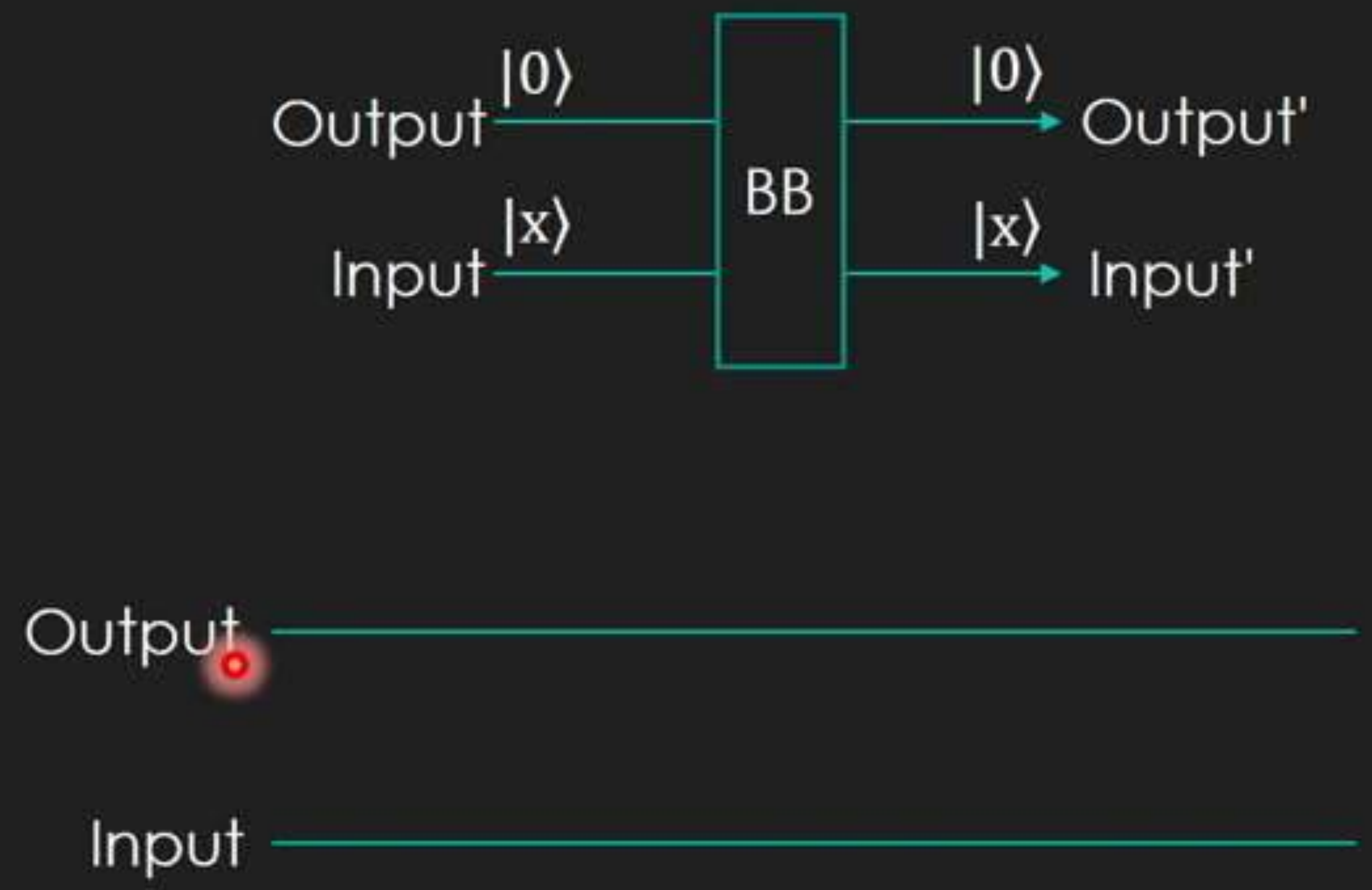
- How do we write nonreversible functions in a reversible way?
- Common hack: add an additional **output qbit** to which the function action is applied
- We thus have to rewire our black box:



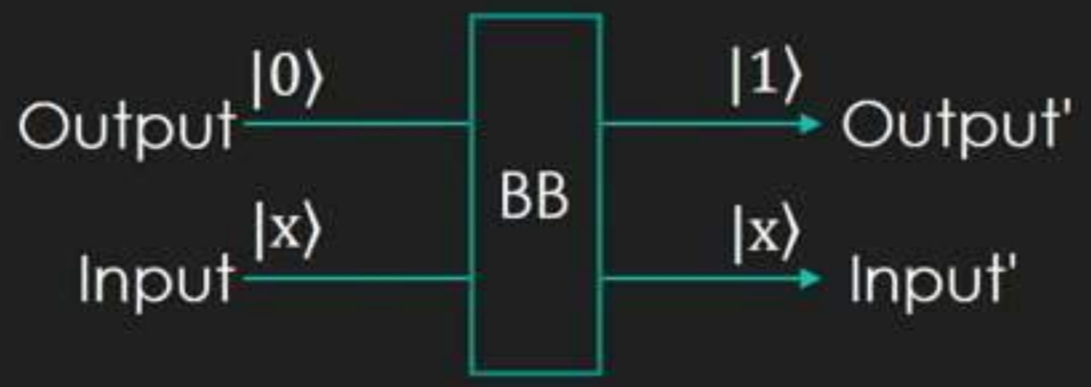


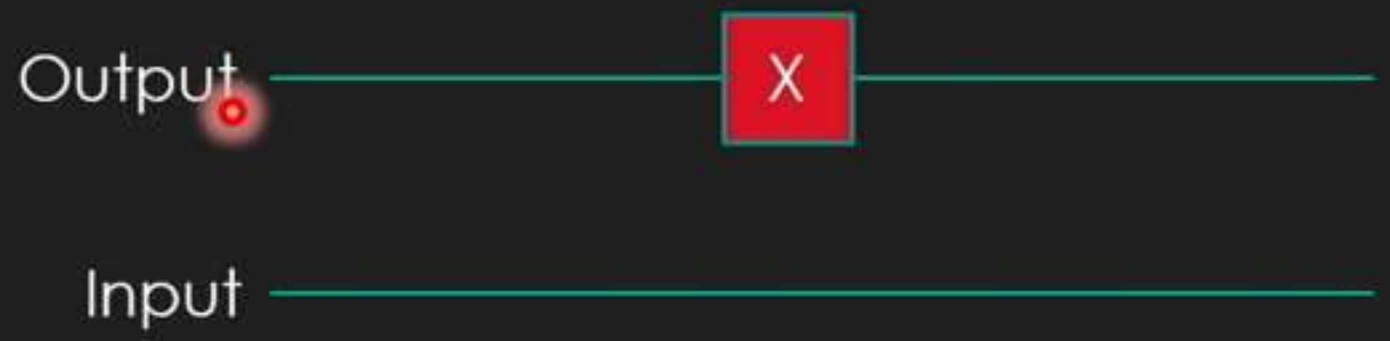
- The black box leaves the **input qbit** unchanged, writing function output to **output qbit**

The Deutsch oracle: constant-0

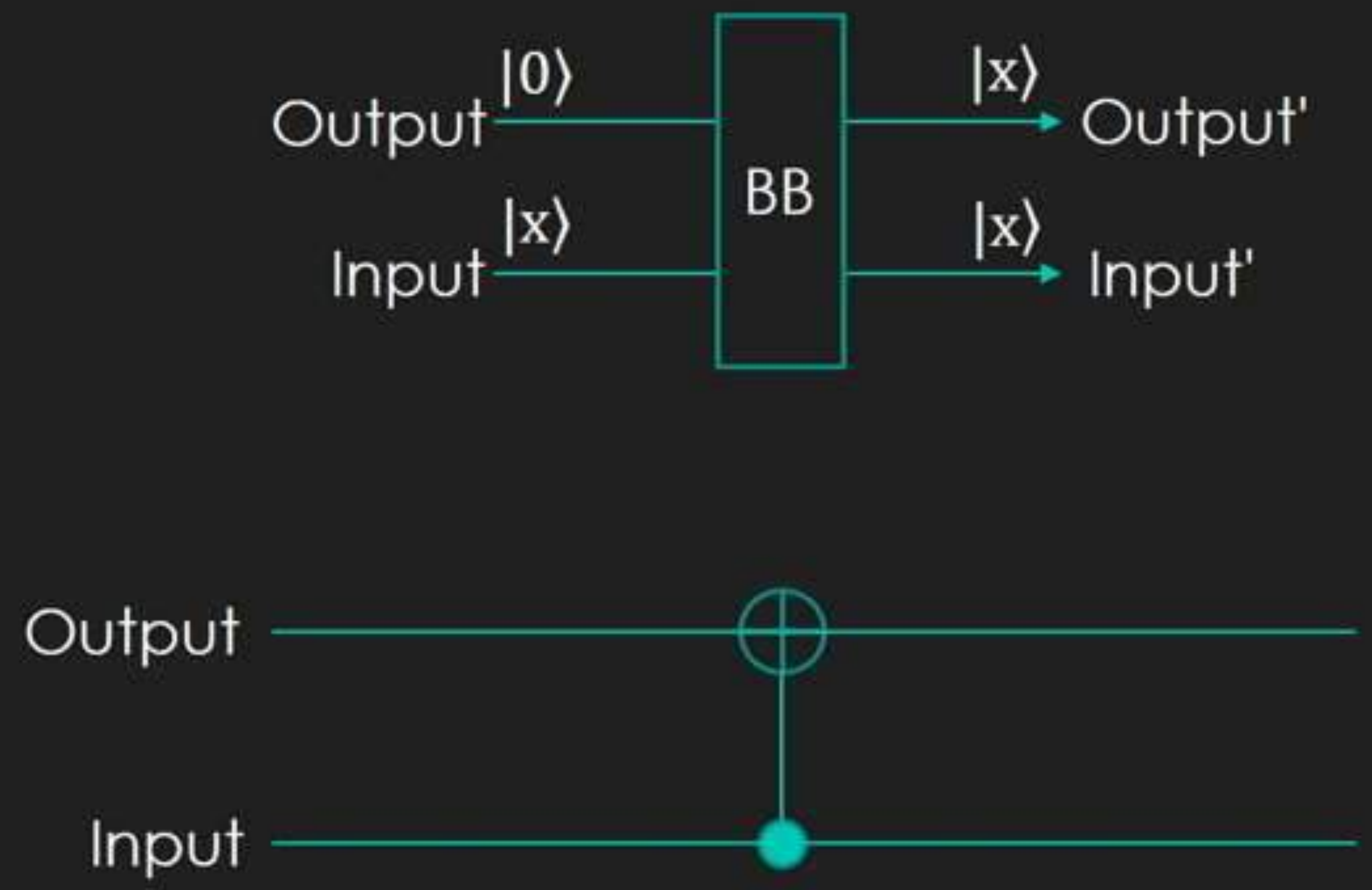


The Deutsch oracle: constant-1

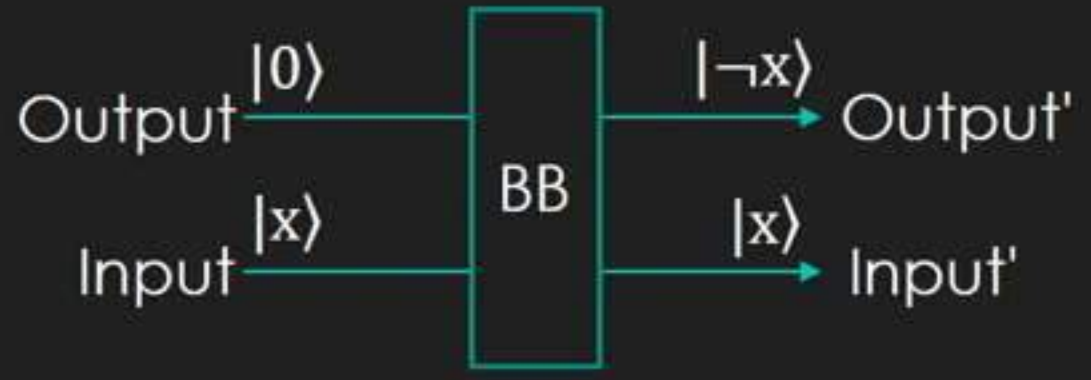




The Deutsch oracle: identity

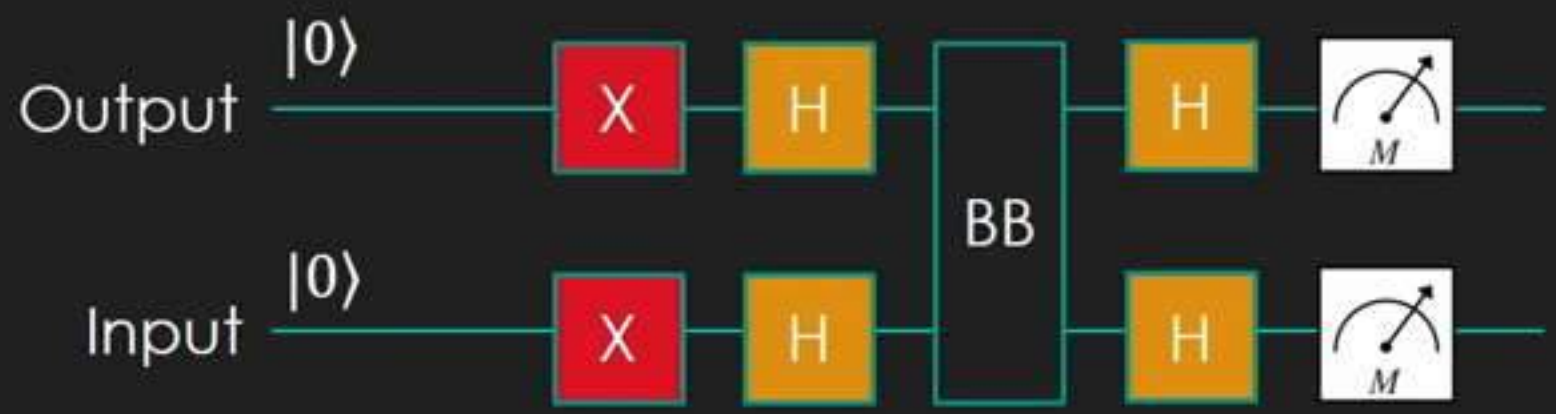


The Deutsch oracle: negation



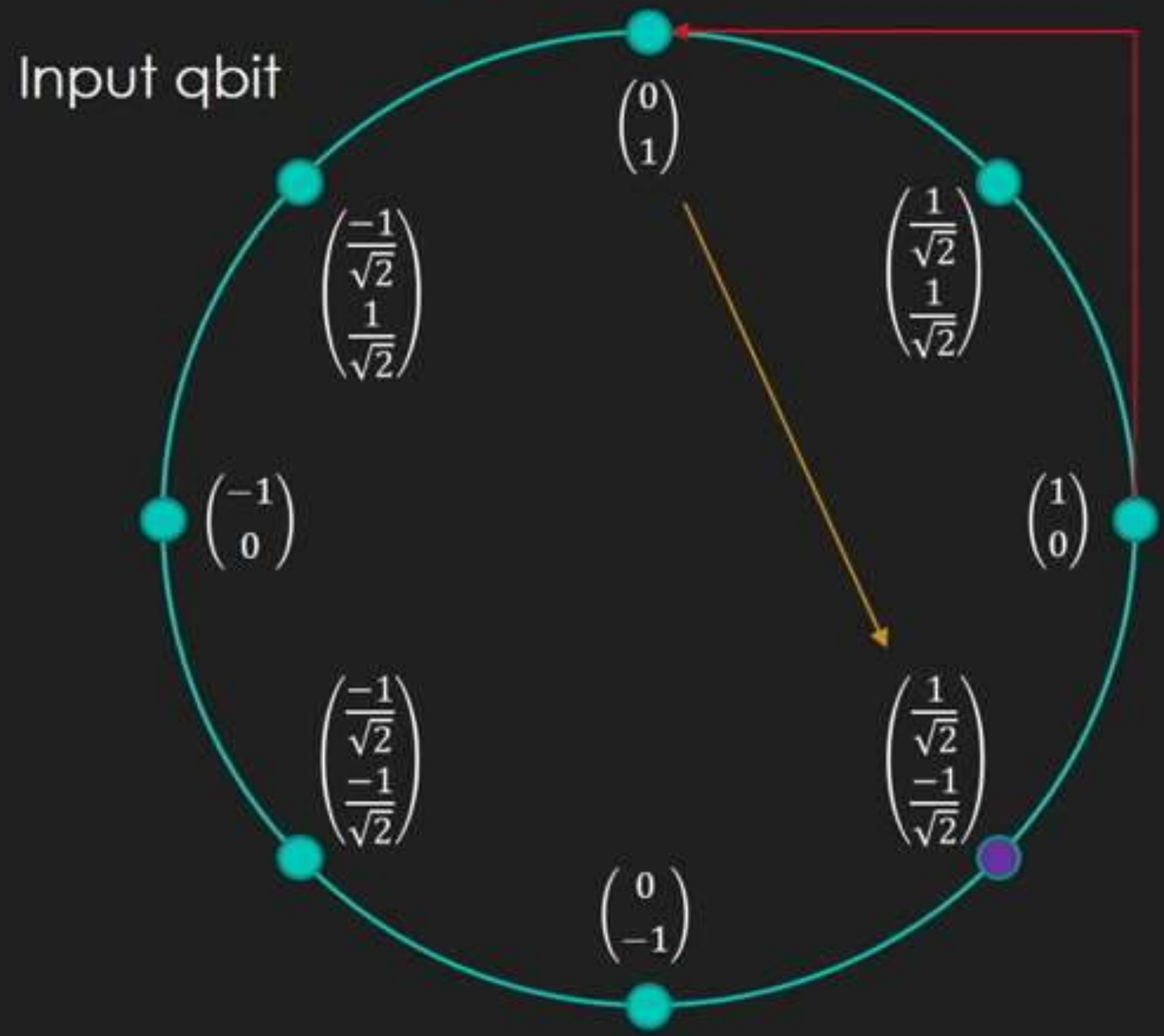
The Deutsch oracle

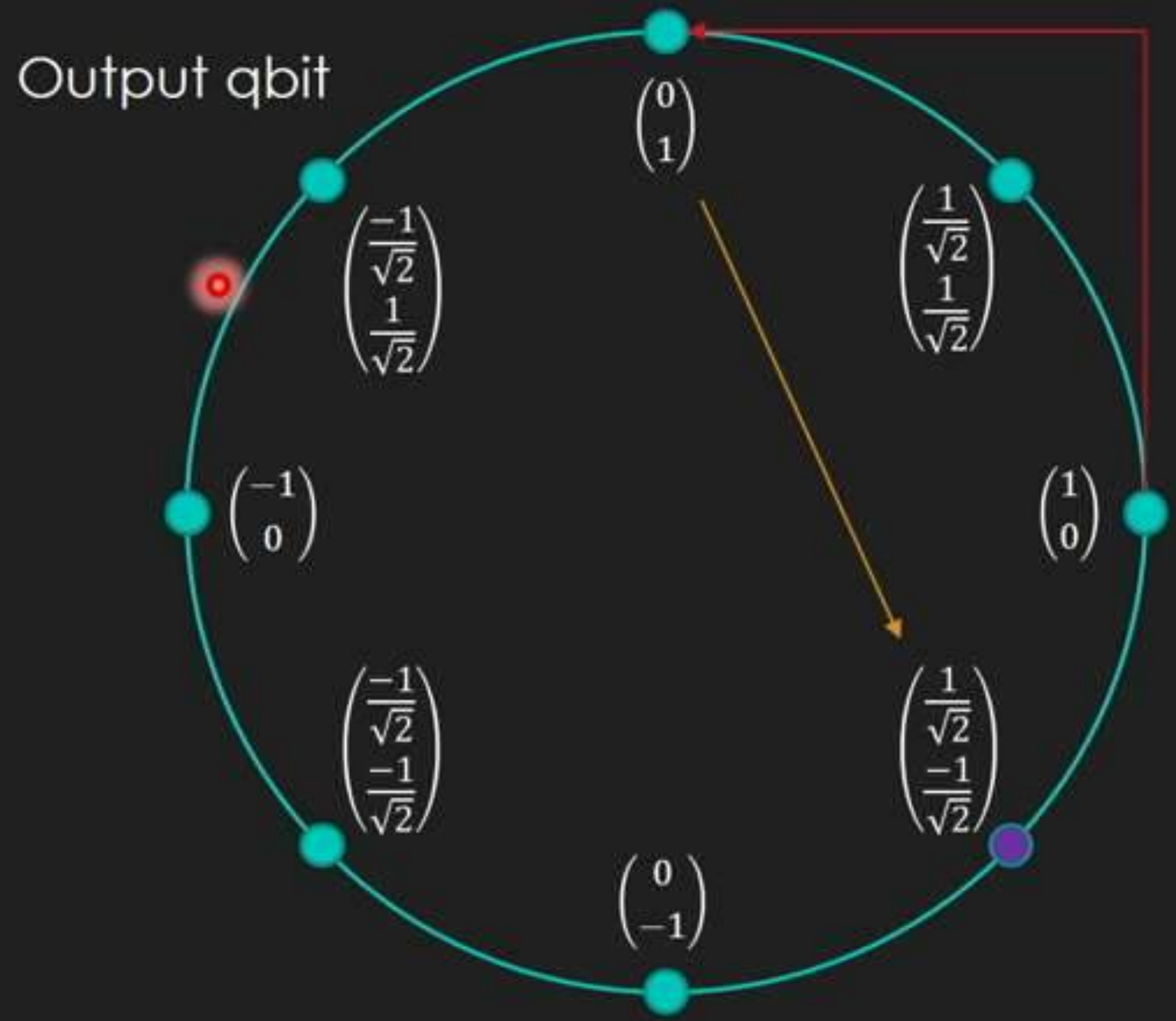
- How do we solve it on a quantum computer in one query?



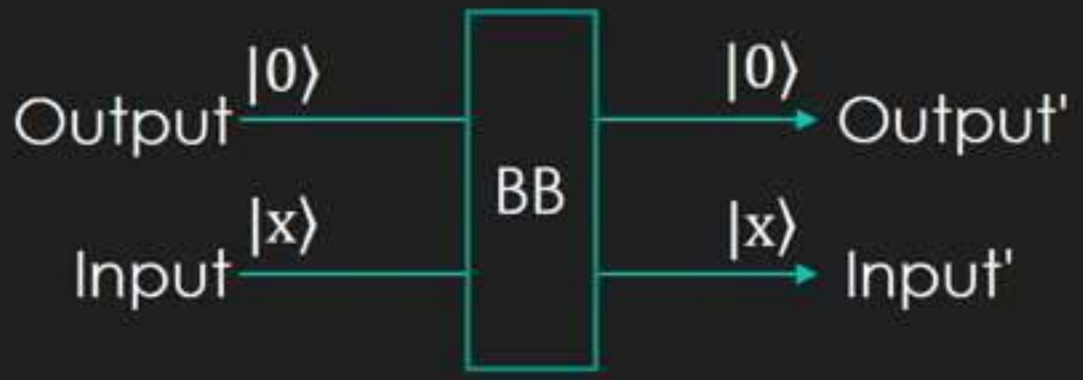
- If the black-box function is constant, system will be in state $|11\rangle$ after measurement
- If the black-box function is variable, system will be in state $|01\rangle$ after measurement

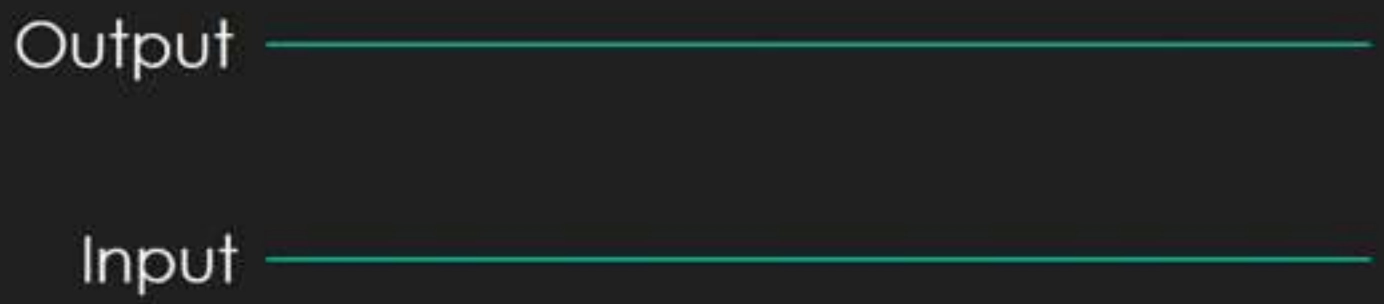
The Deutsch oracle: preprocessing



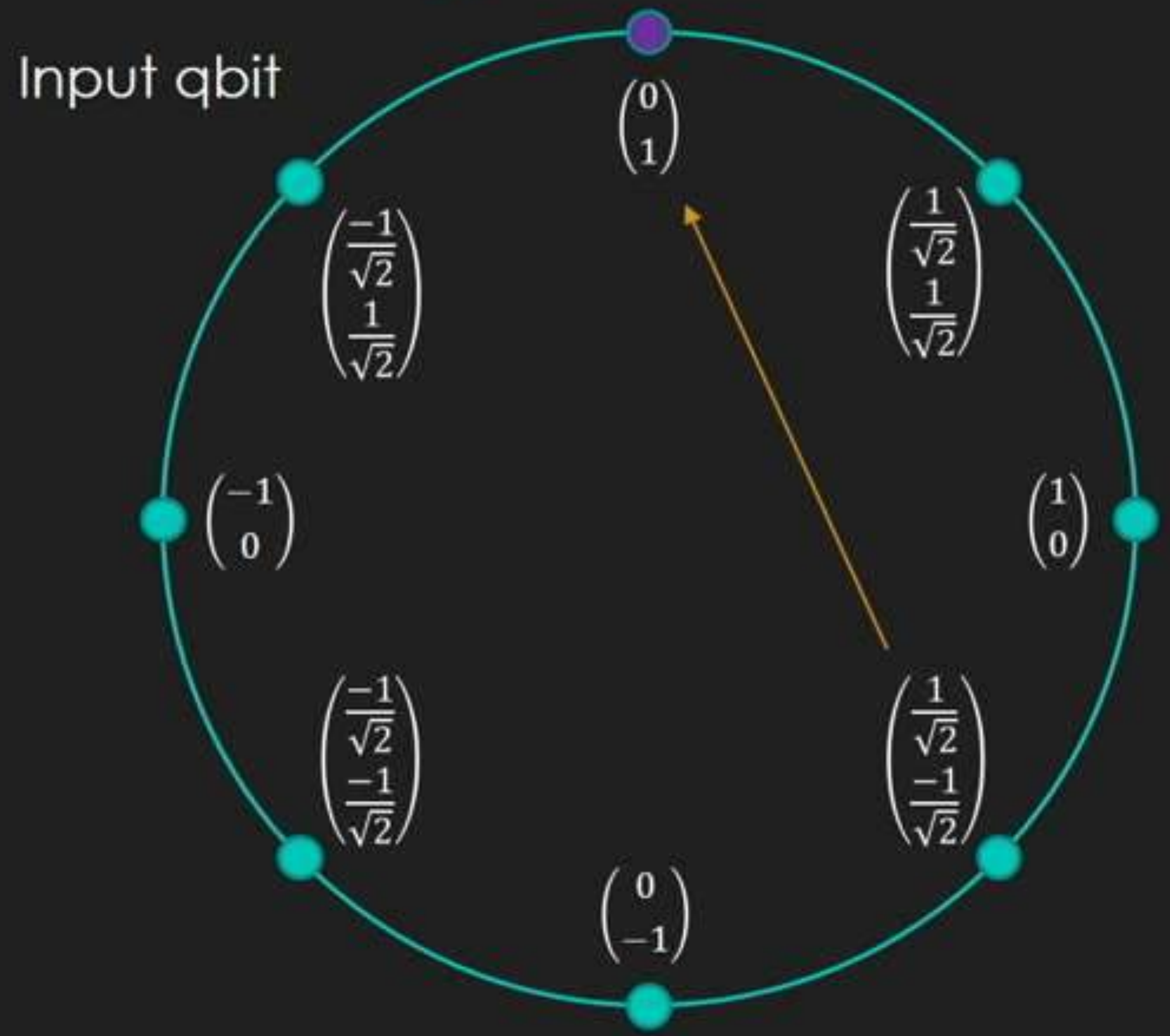


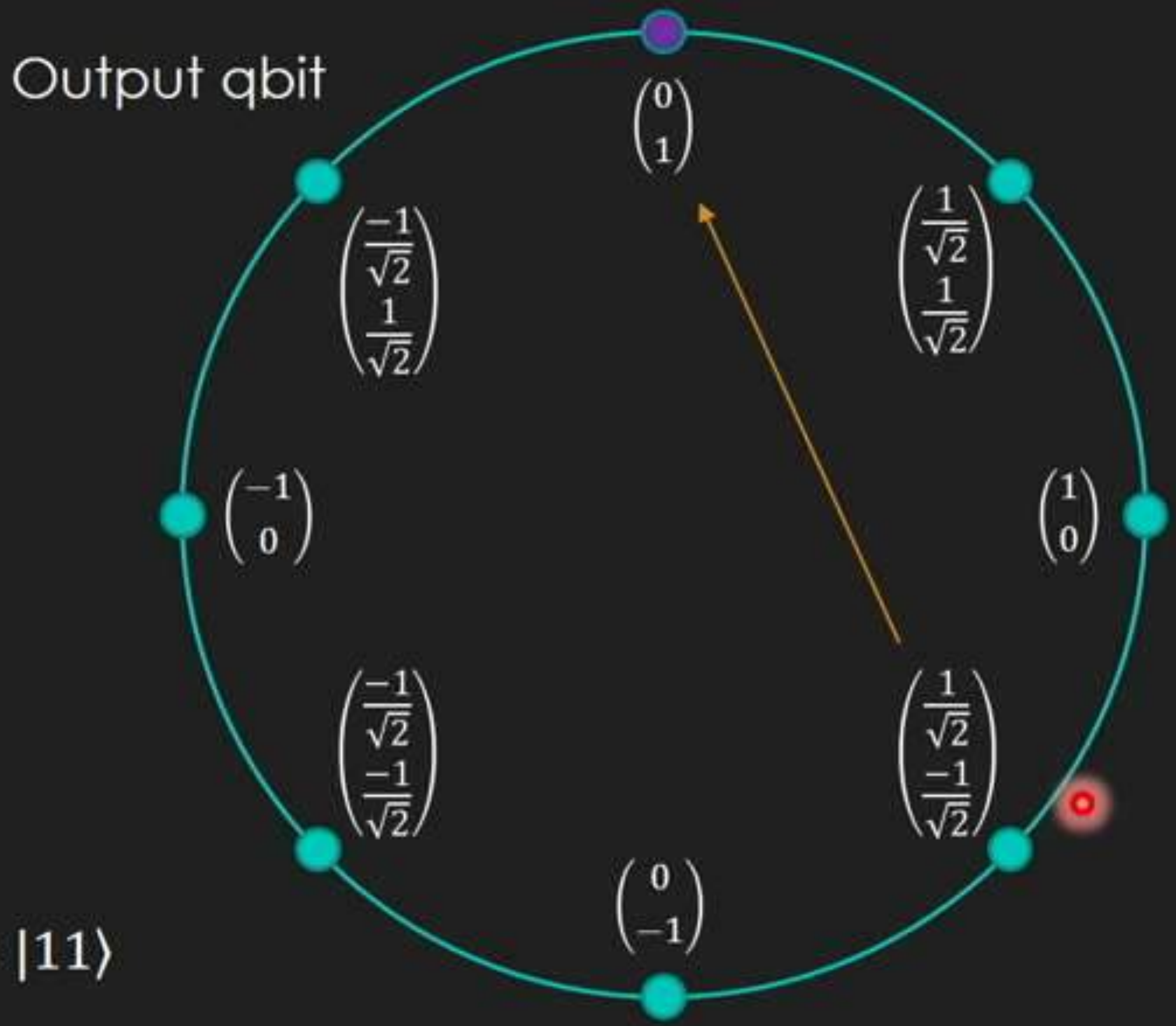
The Deutsch oracle: constant-0





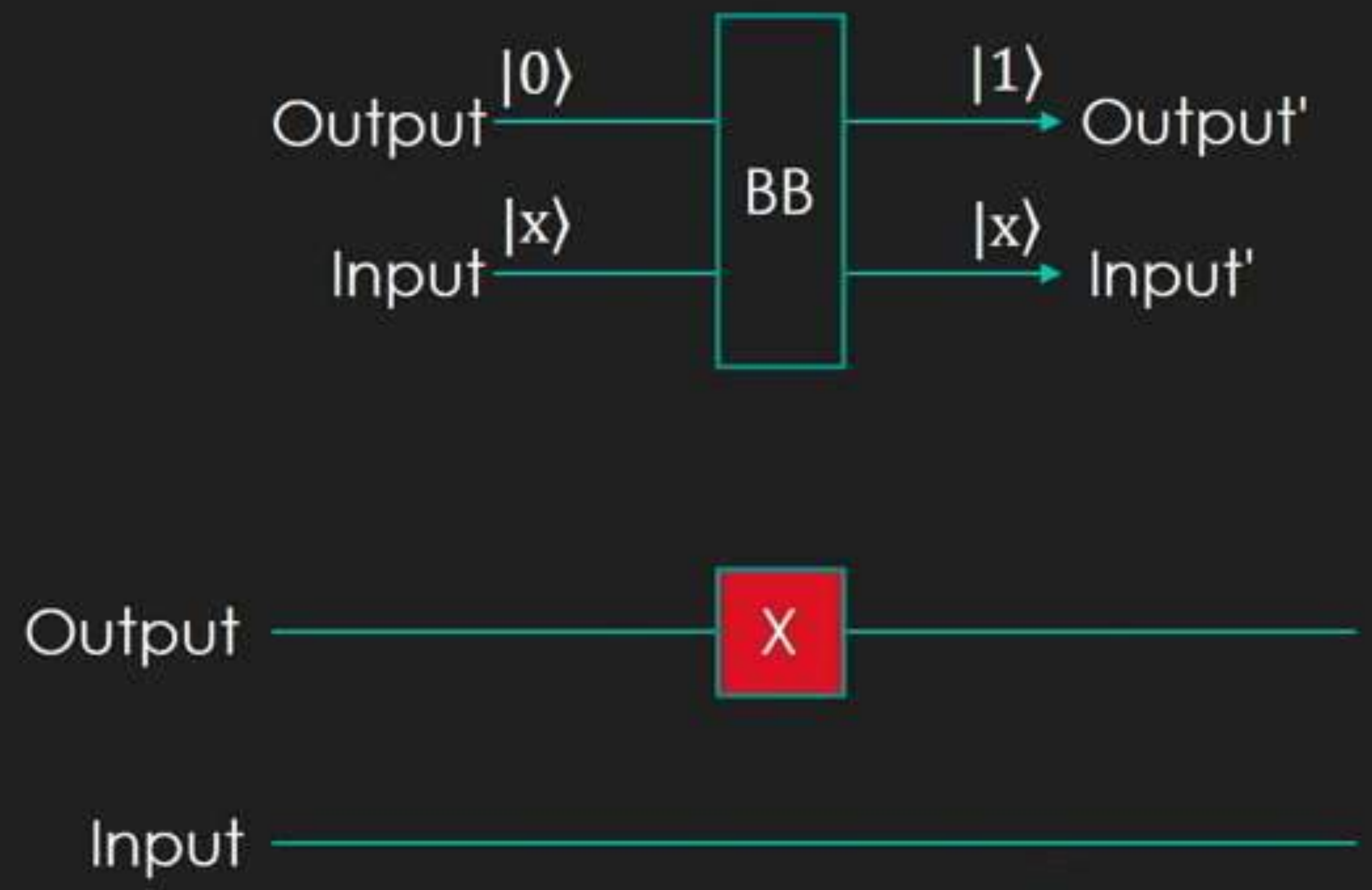
The Deutsch oracle: constant-0



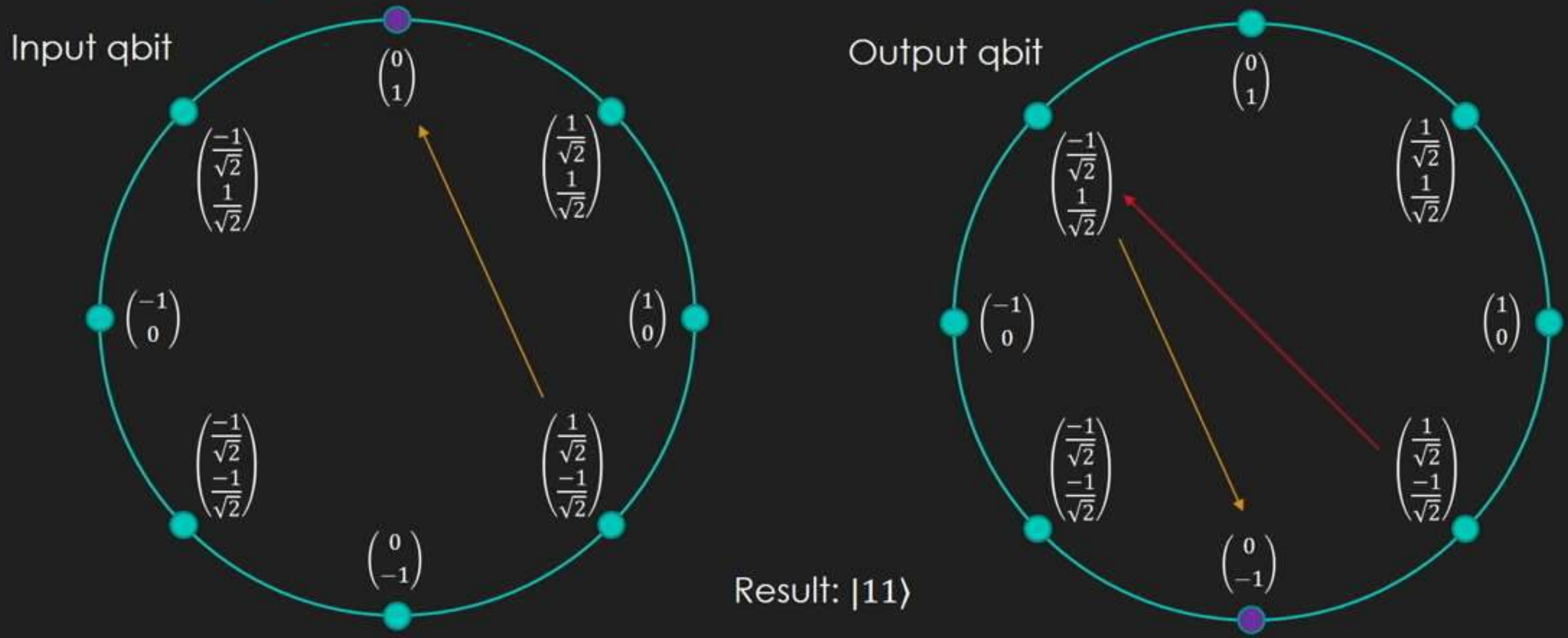


Result: $|11\rangle$

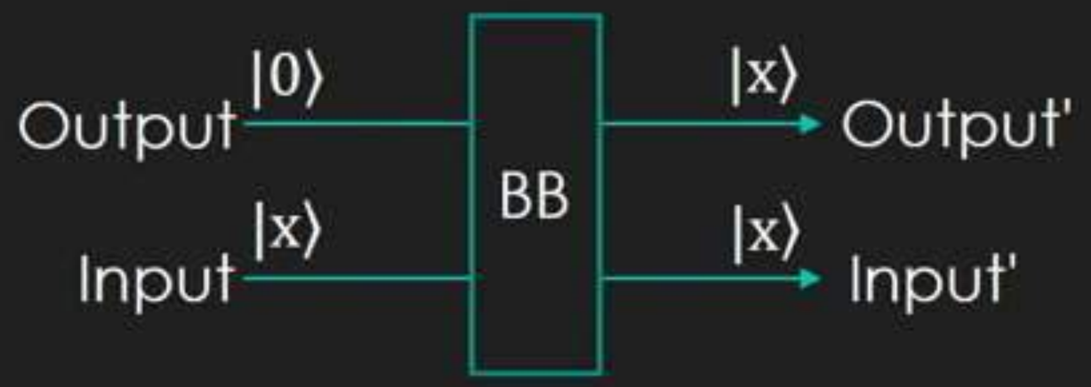
The Deutsch oracle: constant-1



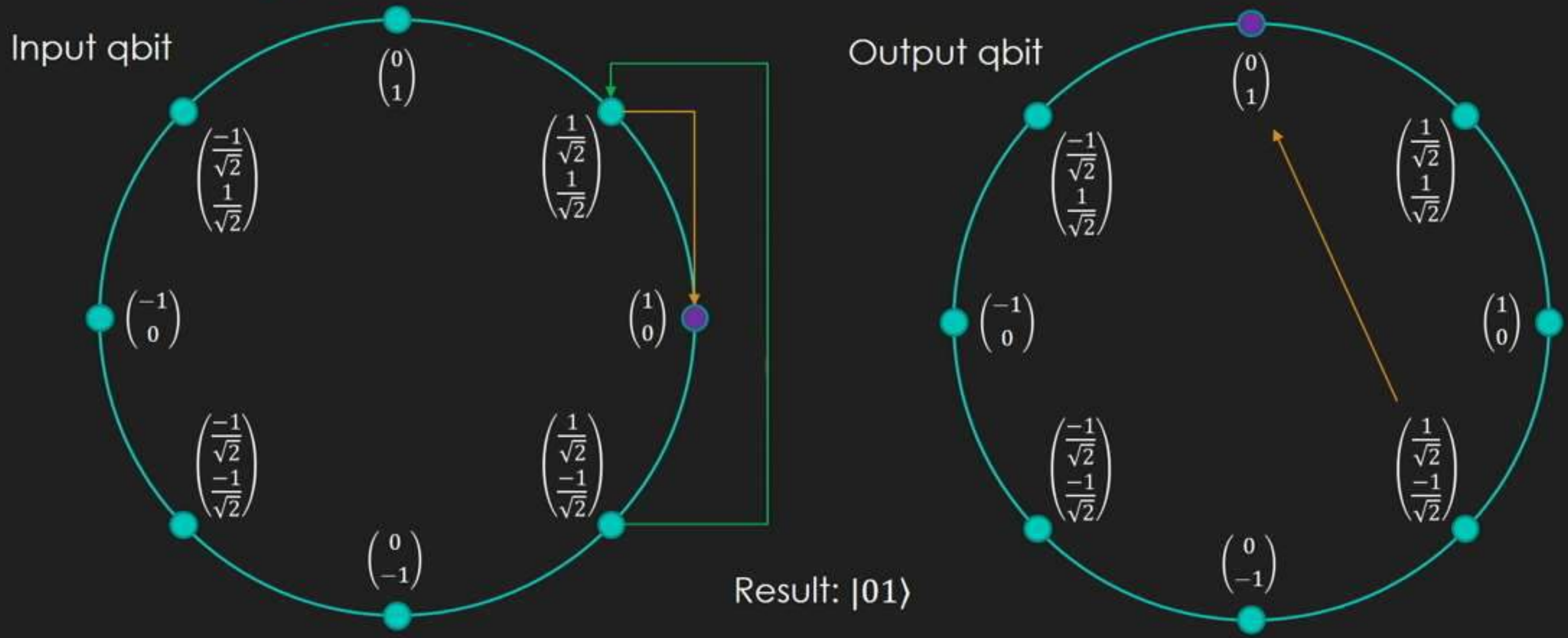
The Deutsch oracle: constant-1



The Deutsch oracle: identity



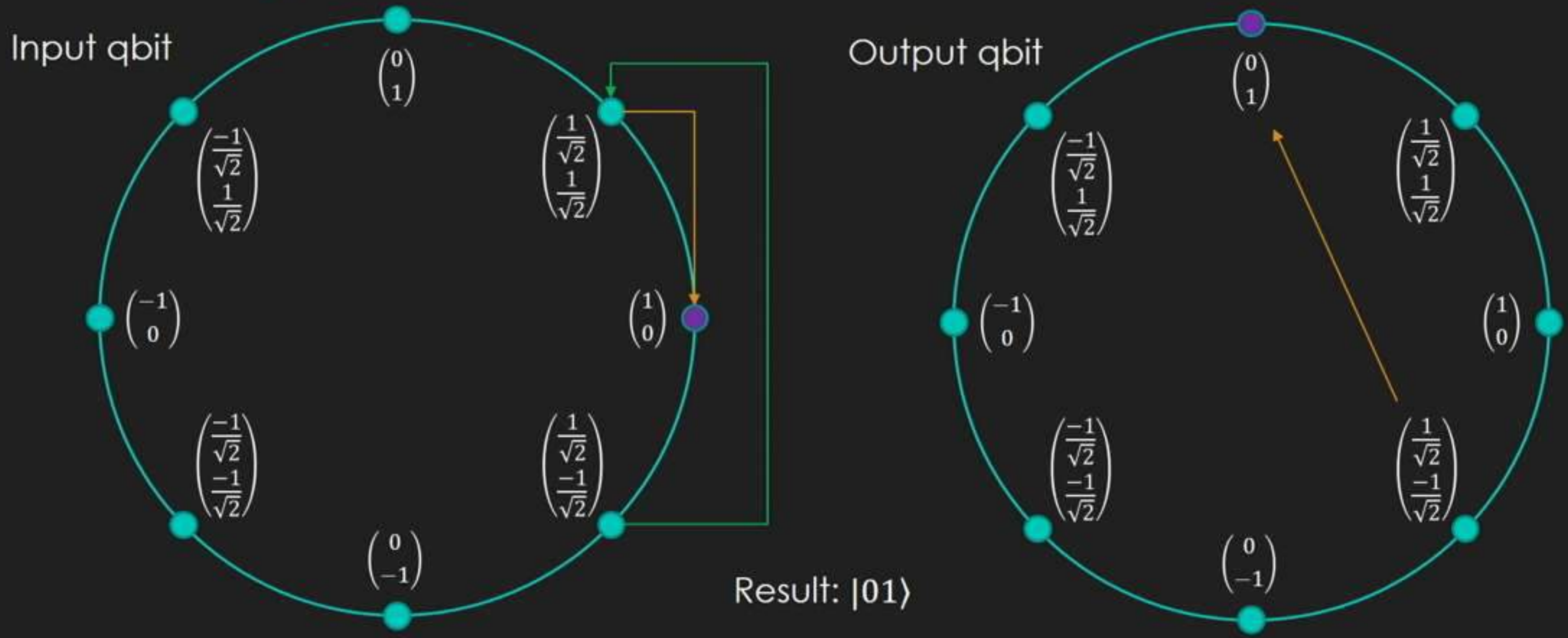
The Deutsch oracle: identity



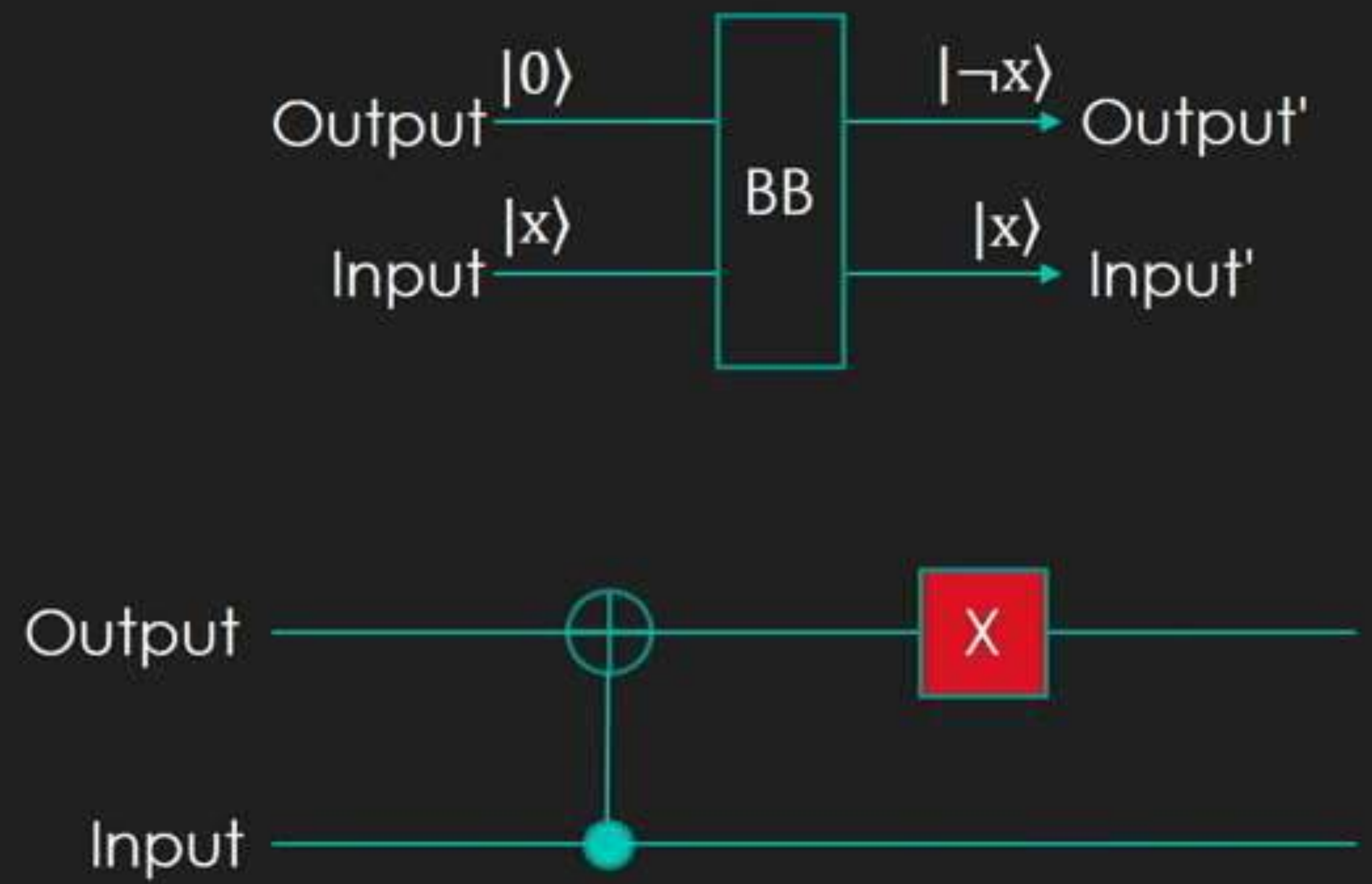
The Deutsch oracle: identity

$$C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{-1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{-1}{\sqrt{2}}\end{pmatrix}\right)=C\begin{pmatrix}\frac{1}{2}\\ \frac{-1}{2}\\ \frac{-1}{2}\\ \frac{1}{2}\end{pmatrix}=\frac{1}{2}\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}1\\-1\\-1\\1\end{pmatrix}=\frac{1}{2}\begin{pmatrix}1\\-1\\1\\-1\end{pmatrix}=\begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{-1}{\sqrt{2}}\end{pmatrix}$$

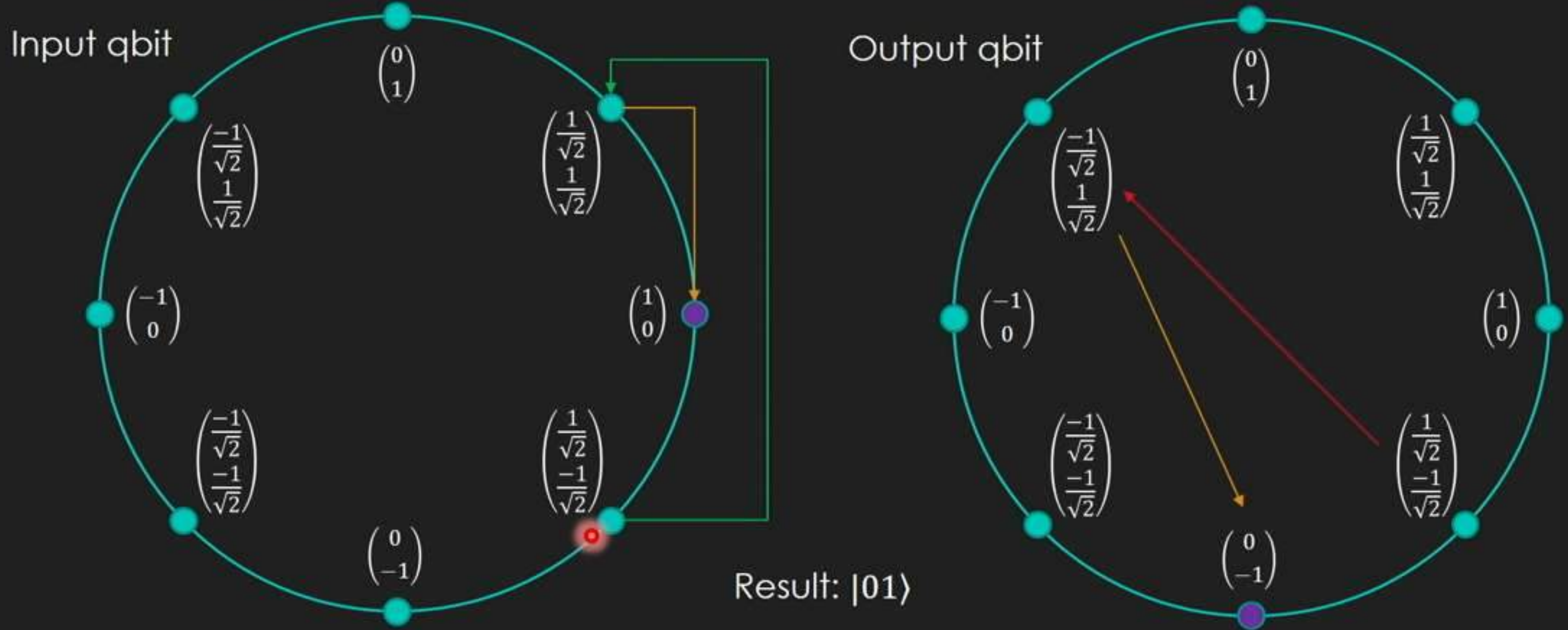
The Deutsch oracle: identity



The Deutsch oracle: negation



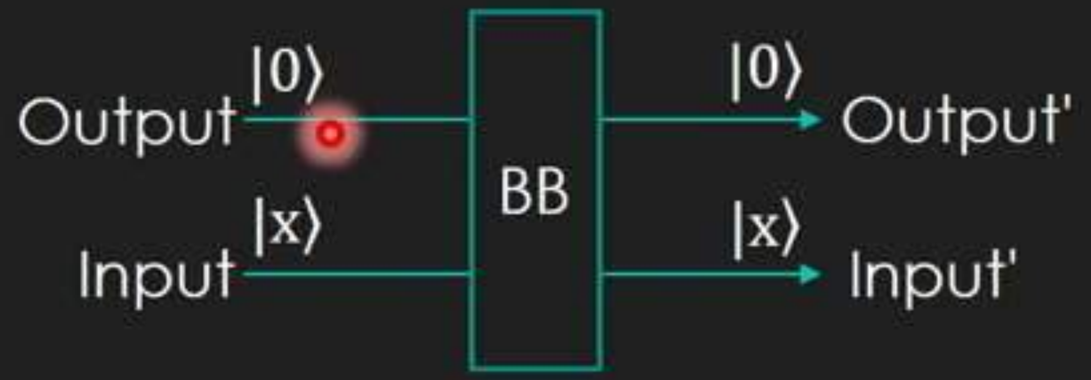
The Deutsch oracle: negation

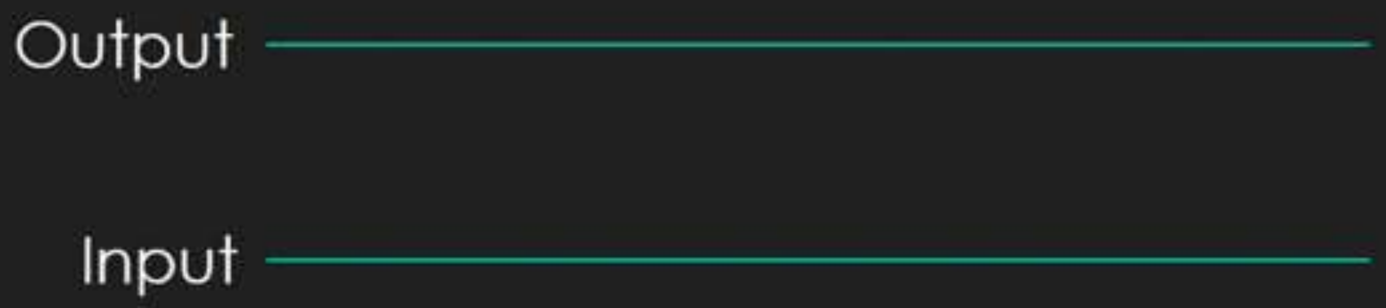


The Deutsch oracle

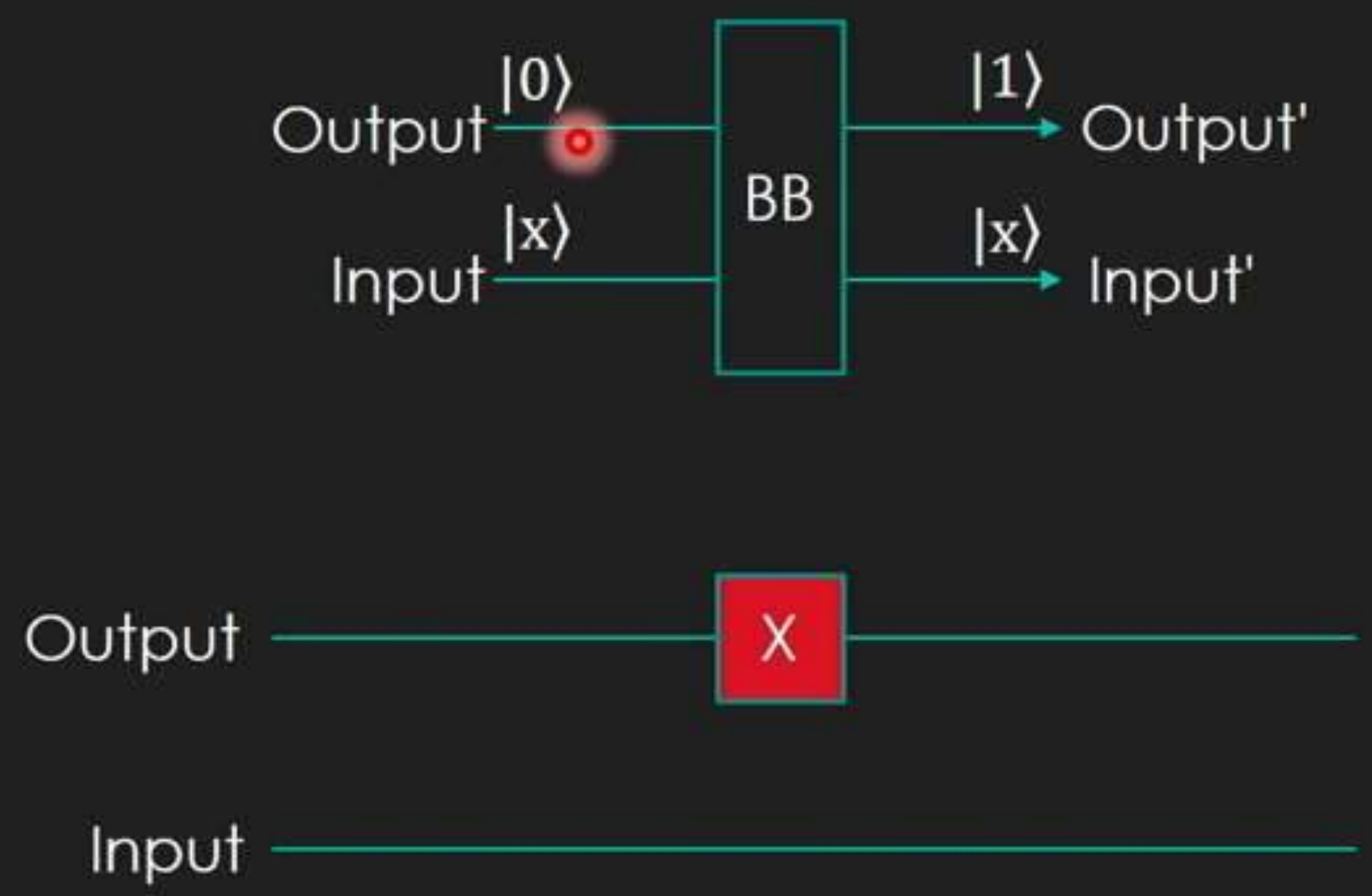
- We did it! We determined whether the function was constant or variable in a single query!
- Intuition: the difference *within* the categories (negation) was neutralized, while the difference *between* the categories (CNOT) was magnified
- This problem seems pretty contrived (and it was, when it was published)
- A generalized version with an n-bit black box also exists (Deutsch-Josza problem)
  - Determine whether the function returns the same value for all 2^n inputs (i.e. is constant)
- A variant of the generalized version was an inspiration for Shor's algorithm!

The Deutsch oracle: constant-0

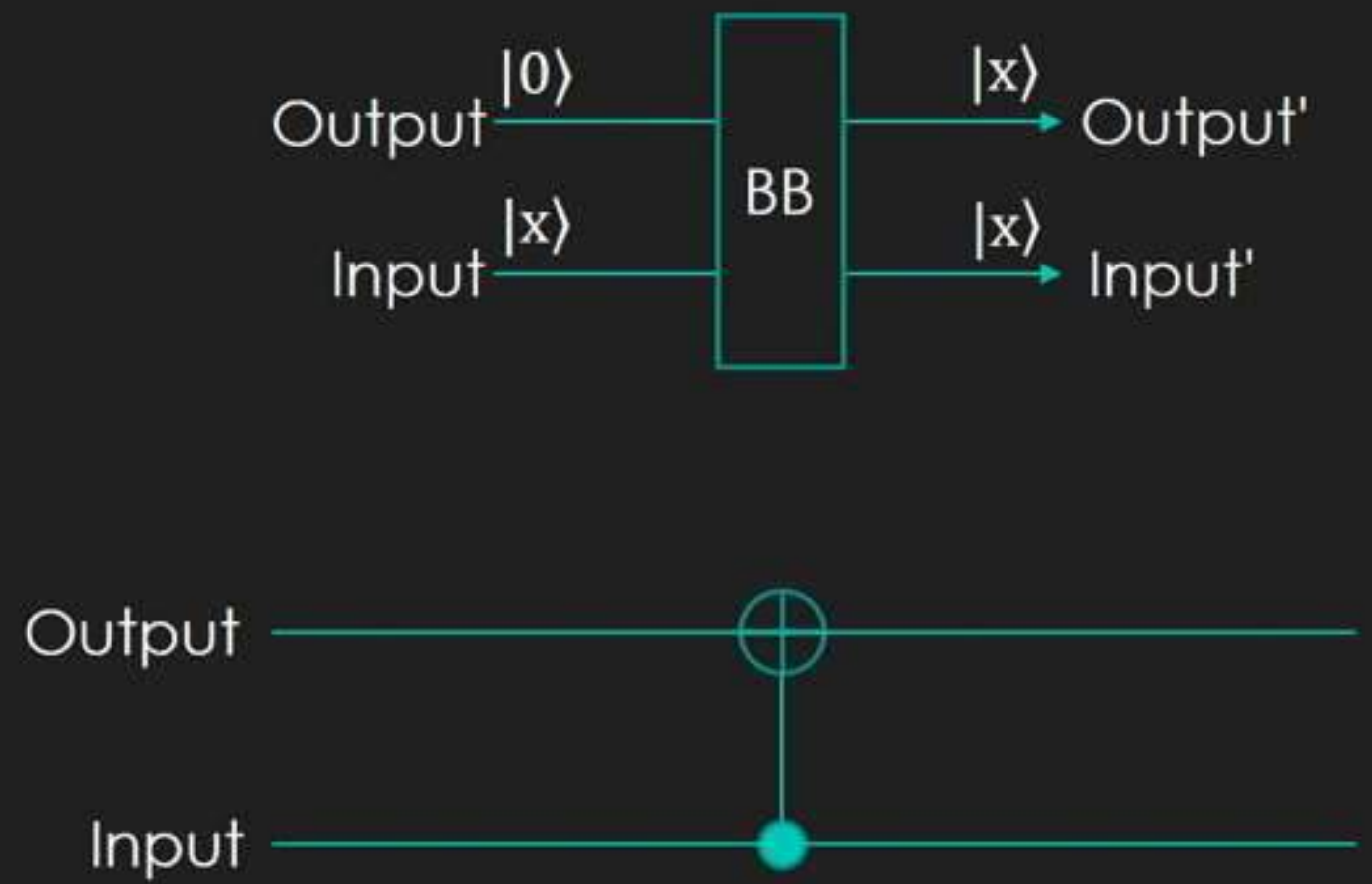




The Deutsch oracle: constant-1



The Deutsch oracle: identity



The Deutsch oracle

- We did it! We determined whether the function was constant or variable in a single query!
- Intuition: the difference *within* the categories (negation) was neutralized, while the difference *between* the categories (CNOT) was magnified
- This problem seems pretty contrived (and it was, when it was published)
- A generalized version with an n-bit black box also exists (Deutsch-Josza problem)
  - Determine whether the function returns the same value for all 2^n inputs (i.e. is constant)
- A variant of the generalized version was an inspiration for Shor's algorithm!

Full recap

- We learned how to model classical computation with basic linear algebra
- We learned about qbits, superposition, and the Hadamard gate
- We learned the Deutsch Oracle problem, where quantum outperforms classical

Bonus topics

- Quantum entanglement
- Quantum teleportation

Entanglement

- If the product state of two qbits cannot be factored, they are said to be **entangled**

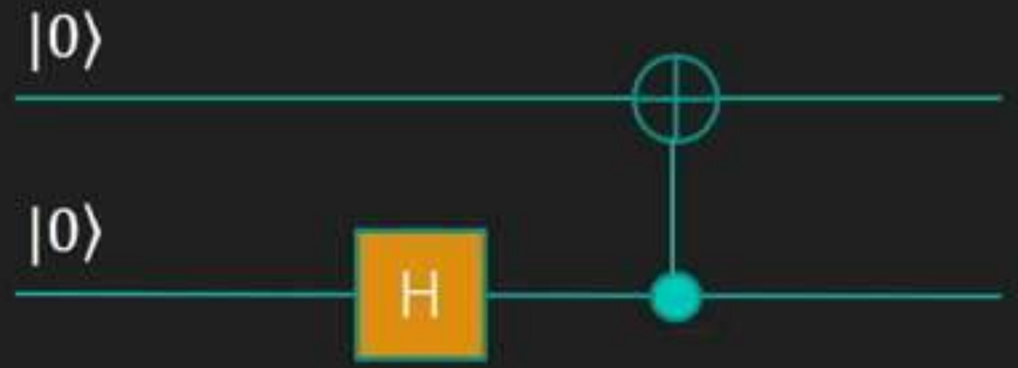
$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix}$$

$$\begin{aligned} ac &= \frac{1}{\sqrt{2}} \\ ad &= 0 \\ bc &= 0 \\ bd &= \frac{1}{\sqrt{2}} \end{aligned}$$

- The system of equations has no solution, so we cannot factor the quantum state!
- This has a 50% chance of collapsing to $|00\rangle$ and 50% chance of collapsing to $|11\rangle$

Entanglement

How can we reach an entangled state? Easy!



$$CH_1\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\ \frac{1}{\sqrt{2}}\\0\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\0\\ \frac{1}{\sqrt{2}}\end{pmatrix}$$

Entanglement

- If the product state of two qbits cannot be factored, they are said to be **entangled**

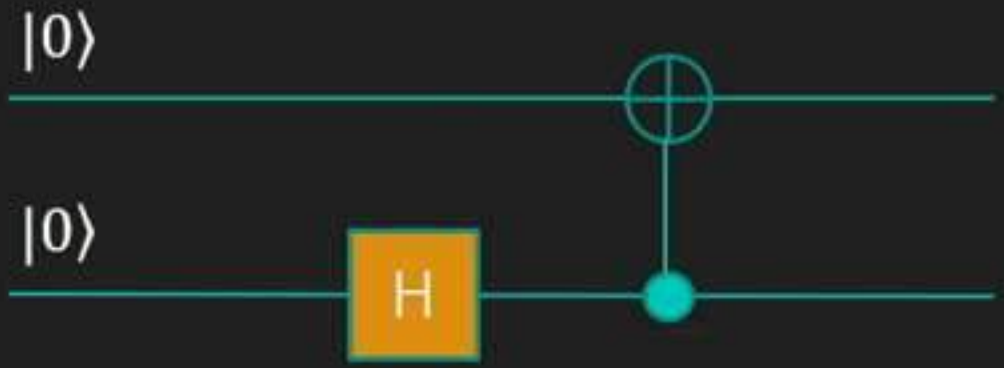
$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \binom{a}{b} \otimes \binom{c}{d}$$

$$ac = \frac{1}{\sqrt{2}}$$
$$ad = 0$$
$$bc = 0$$
$$bd = \frac{1}{\sqrt{2}}$$

- The system of equations has no solution, so we cannot factor the quantum state!
- This has a 50% chance of collapsing to $|00\rangle$ and 50% chance of collapsing to $|11\rangle$

Entanglement

How can we reach an entangled state? Easy!



$$CH_1\left(\begin{pmatrix}1\\0\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = C\left(\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}\otimes\begin{pmatrix}1\\0\end{pmatrix}\right) = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\\frac{1}{\sqrt{2}}\\0\end{pmatrix} = \begin{pmatrix}\frac{1}{\sqrt{2}}\\0\\0\\\frac{1}{\sqrt{2}}\end{pmatrix}$$

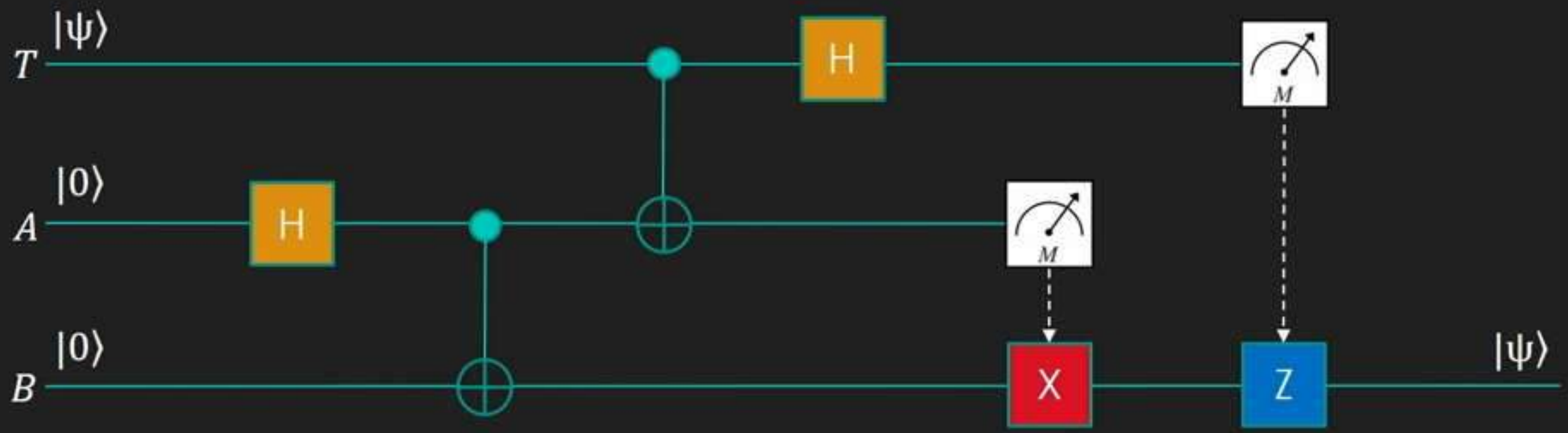
Entanglement

- What's going on here? The qbits seem to be coordinating in some way
  - Measuring one qbit also collapses the other in a correlated state
- This coordination happens even across vast stretches of space
- The coordination even happens faster than the speed of light! It is instantaneous.
  - A 2013 experiment measured particles within 0.01% of the travel time of light between them
- Surely the qbits "decided" at the time of entanglement what they would do?
  - No! This is called "hidden variable" theory and was disproved by John Bell in 1964
- This does indeed break locality through faster-than-light coordination
  - However – and this is the critical part – *no information can be communicated*

Teleportation

- **Quantum teleportation** is the process by which the state of an arbitrary qbit is transferred from one location to another by way of two other entangled qbits
- You can transfer qbit states (cut & paste) but you cannot clone them (copy & paste)
  - This is called the **No-cloning theorem**
- The teleportation is not faster-than-light, because some classical information must be sent

Teleportation



$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Further learning goals

- Deutsch-Jozsa algorithm and Simon's periodicity problem
  - Former yields oracle separation between EQP and P, latter between BQP and BPP
- Shor's algorithm and Grover's algorithm
- Quantum cryptographic key exchange
- How qbits, gates, and measurement are actually implemented
- Quantum error correction
- Quantum programming language design

Entanglement

- If the product state of two qbits cannot be factored, they are said to be **entangled**

$$\begin{pmatrix} \frac{1}{\sqrt{2}} \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix}$$

$$\begin{aligned} ac &= \frac{1}{\sqrt{2}} \\ ad &= 0 \\ bc &= 0 \\ bd &= \frac{1}{\sqrt{2}} \end{aligned}$$

- The system of equations has no solution, so we cannot factor the quantum state!
- This has a 50% chance of collapsing to $|00\rangle$ and 50% chance of collapsing to $|11\rangle$

Further learning goals

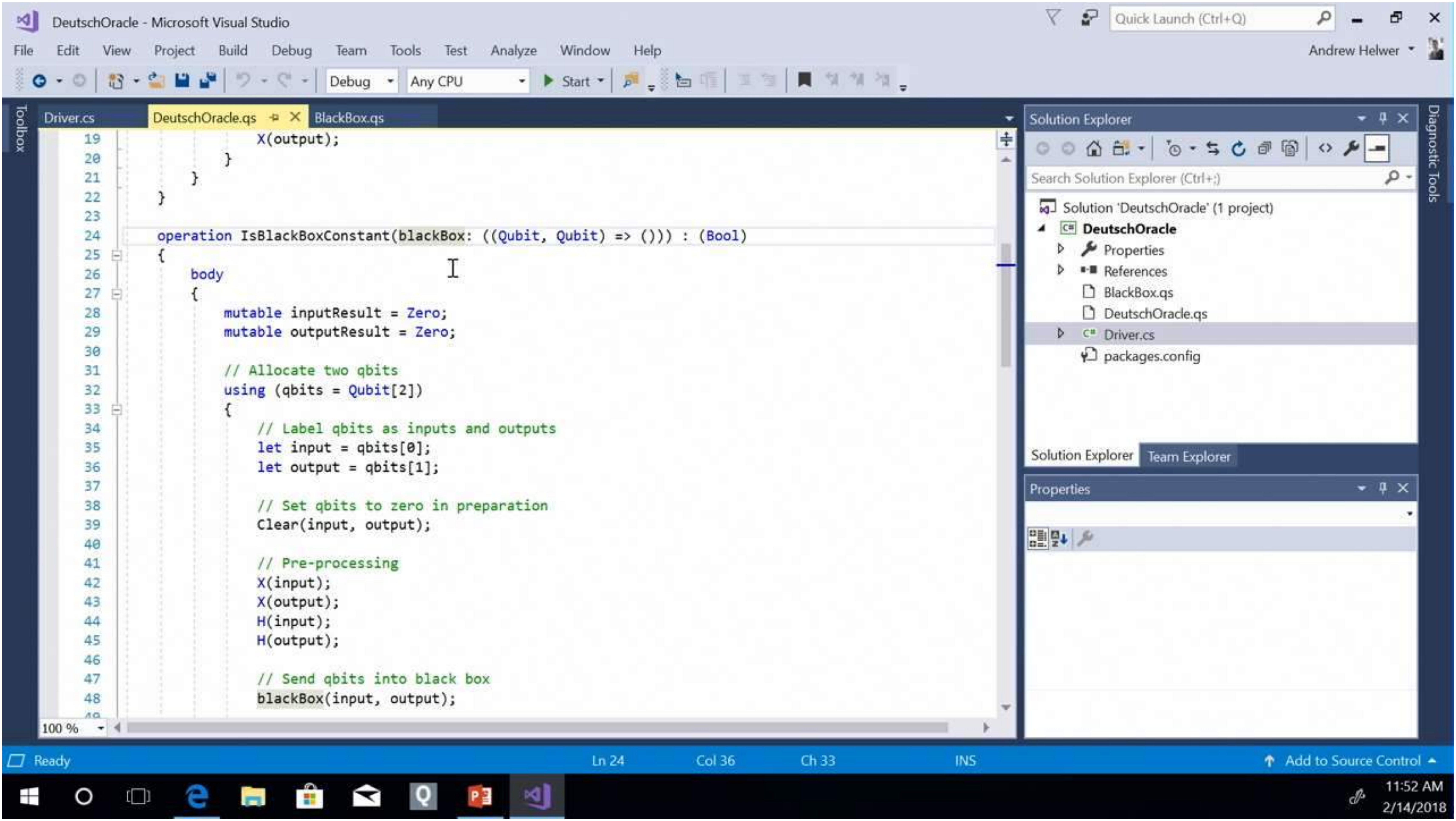
- Deutsch-Jozsa algorithm and Simon's periodicity problem
  - Former yields oracle separation between EQP and P, latter between BQP and BPP
- Shor's algorithm and Grover's algorithm
- Quantum cryptographic key exchange
- How qbits, gates, and measurement are actually implemented
- Quantum error correction
- Quantum programming language design

Further reading

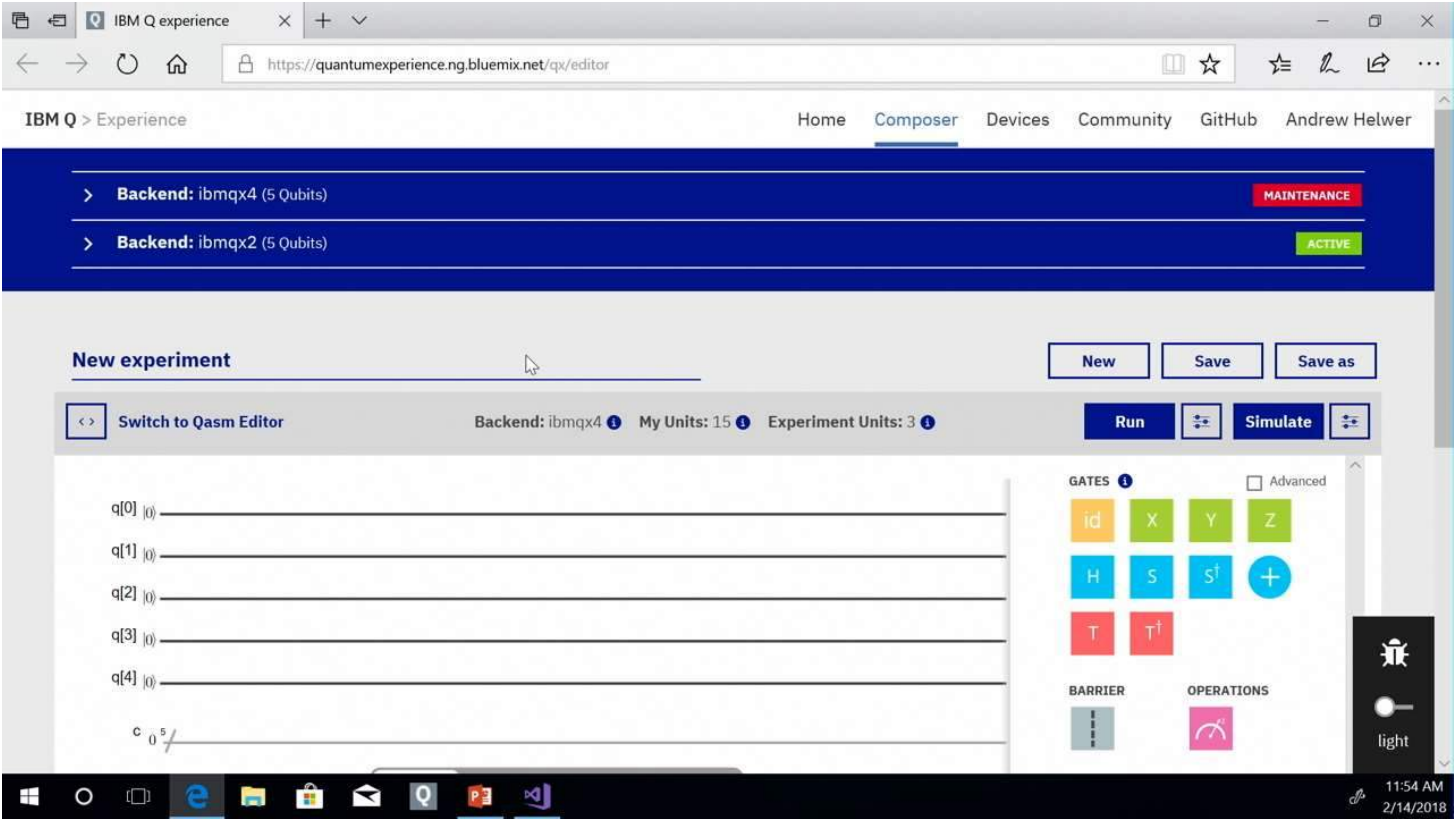
- Recommended textbook: *Quantum Computing for Computer Scientists*
  - Others have recommended *Quantum Computing: A Gentle Introduction*
  - For those with heavier math backgrounds, *Quantum Computer Science: An Introduction*
- The Microsoft Quantum Development Kit docs are nice [link]
  - The development kit contains a quantum computer simulator!
  - Exercise: implement the Deutsch Oracle tester in Q#
- Some skepticism about physically-realizable quantum computers [link]
  - Noise might increase exponentially with the number of physical qbits

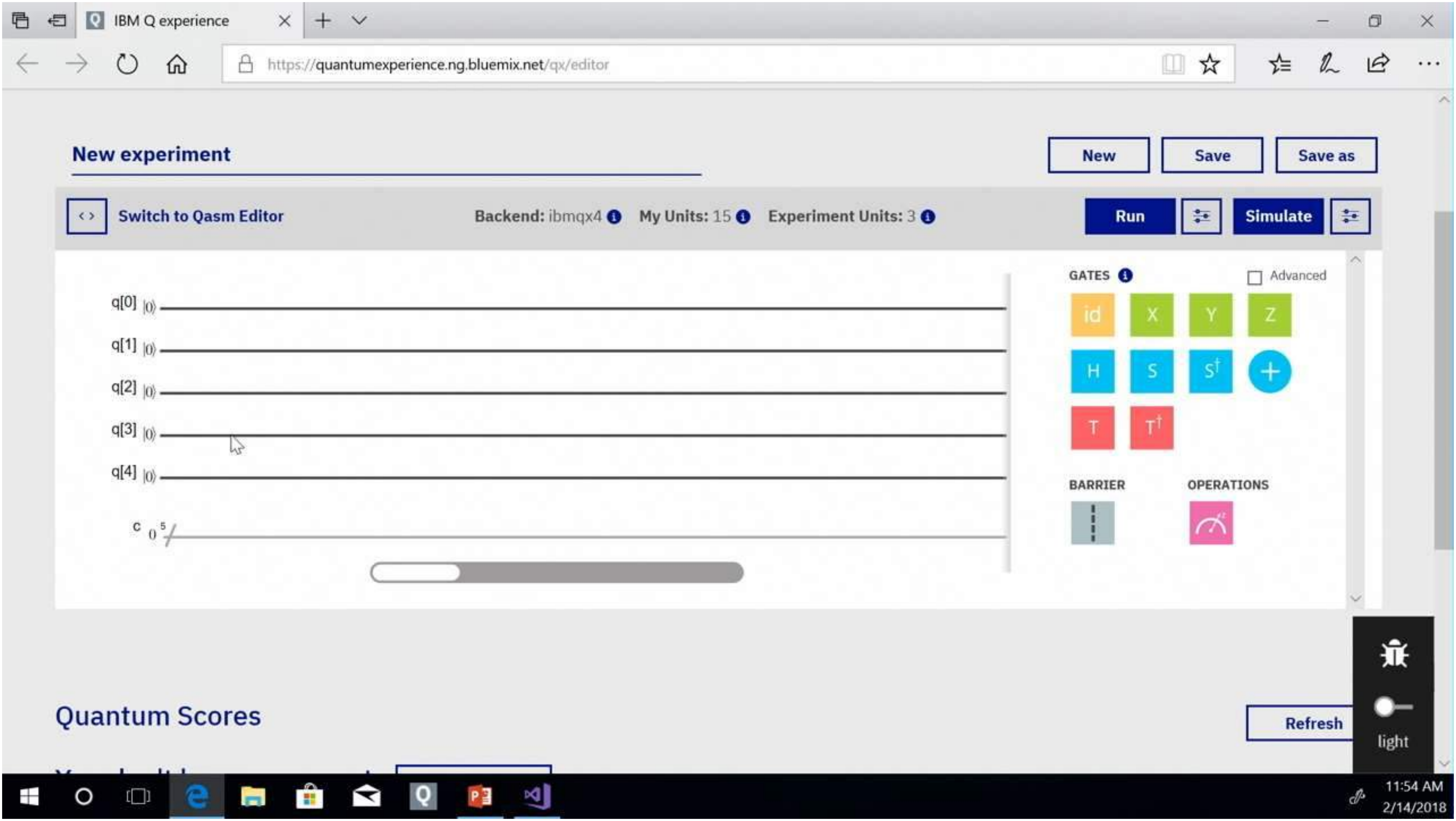
Appendices

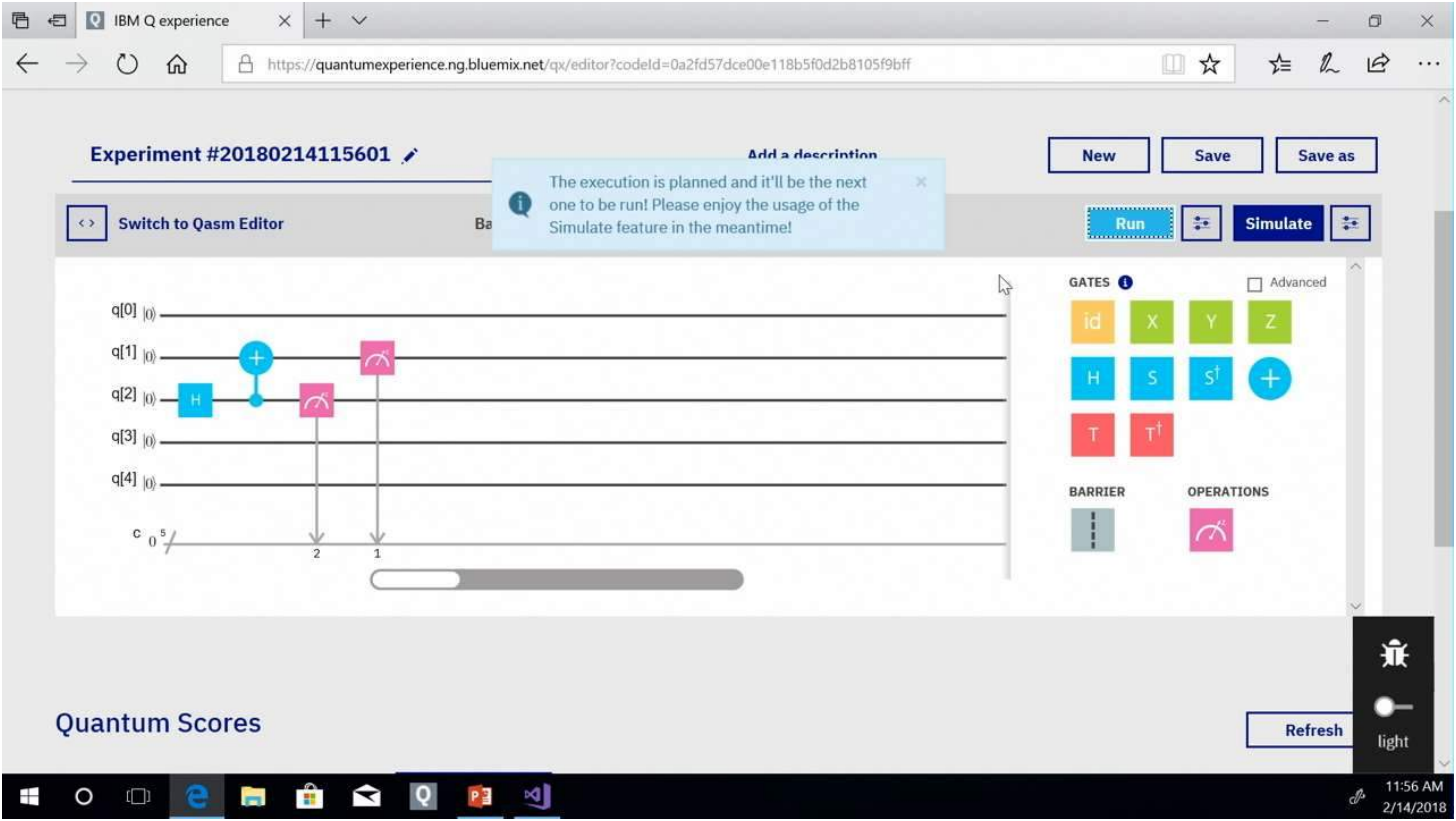
- Single-bit operations on multi-bit states
- Quantum teleportation math

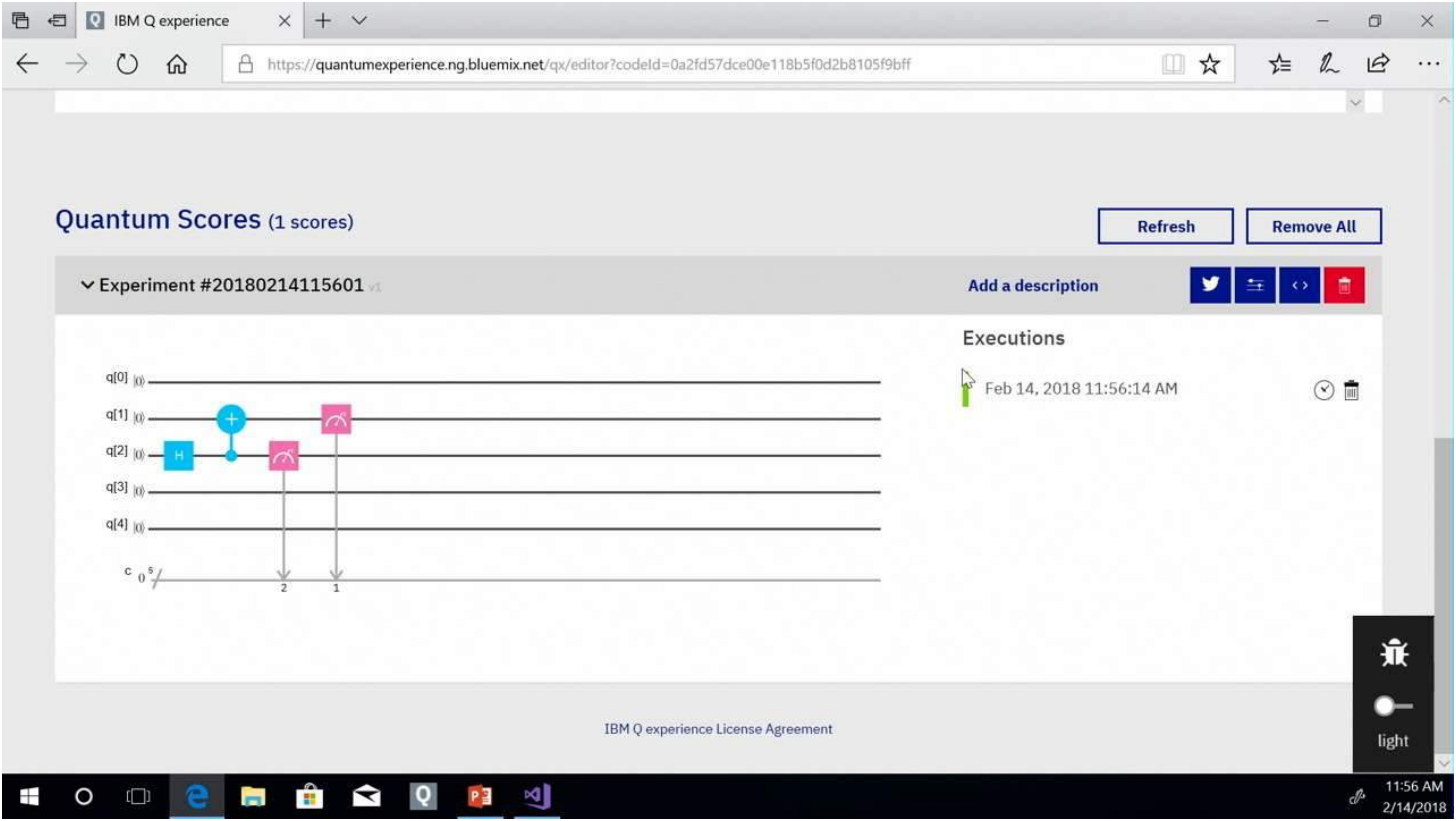


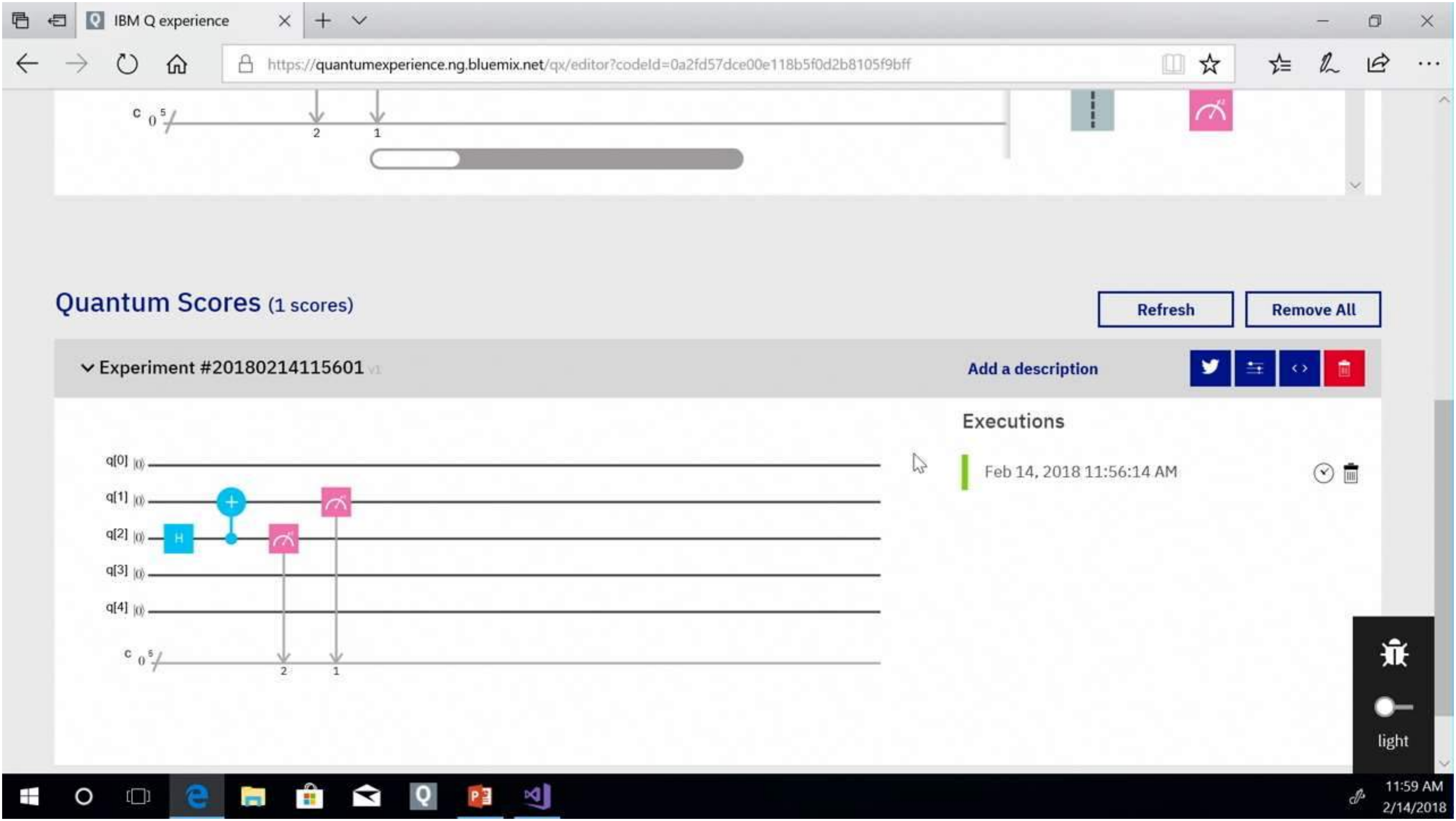
DeutschOracle - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
File
Edit
View
Project
Build
Debug
Team
Tools
Test
Analyze
Window
Help
Andrew Helwer
Debug
Any CPU
Start
Toolbox
Driver.cs
DeutschOracle.qs
BlackBox.qs
X(output);
}
}
}
operation IsBlackBoxConstant(blackBox: ((Qubit, Qubit) => ())) : (Bool)
{
body
{
mutable inputResult = Zero;
mutable outputResult = Zero;
// Allocate two qbits
using (qbits = Qubit[2])
{
// Label qbits as inputs and outputs
let input = qbits[0];
let output = qbits[1];
// Set qbits to zero in preparation
Clear(input, output);
// Pre-processing
X(input);
X(output);
H(input);
H(output);
// Send qbits into black box
blackBox(input, output);
100 %
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'DeutschOracle' (1 project)
DeutschOracle
Properties
References
BlackBox.qs
DeutschOracle.qs
Driver.cs
packages.config
Solution Explorer
Team Explorer
Properties
Diagnostic Tools
Ready
Ln 24
Col 36
Ch 33
INS
Add to Source Control
11:52 AM
2/14/2018

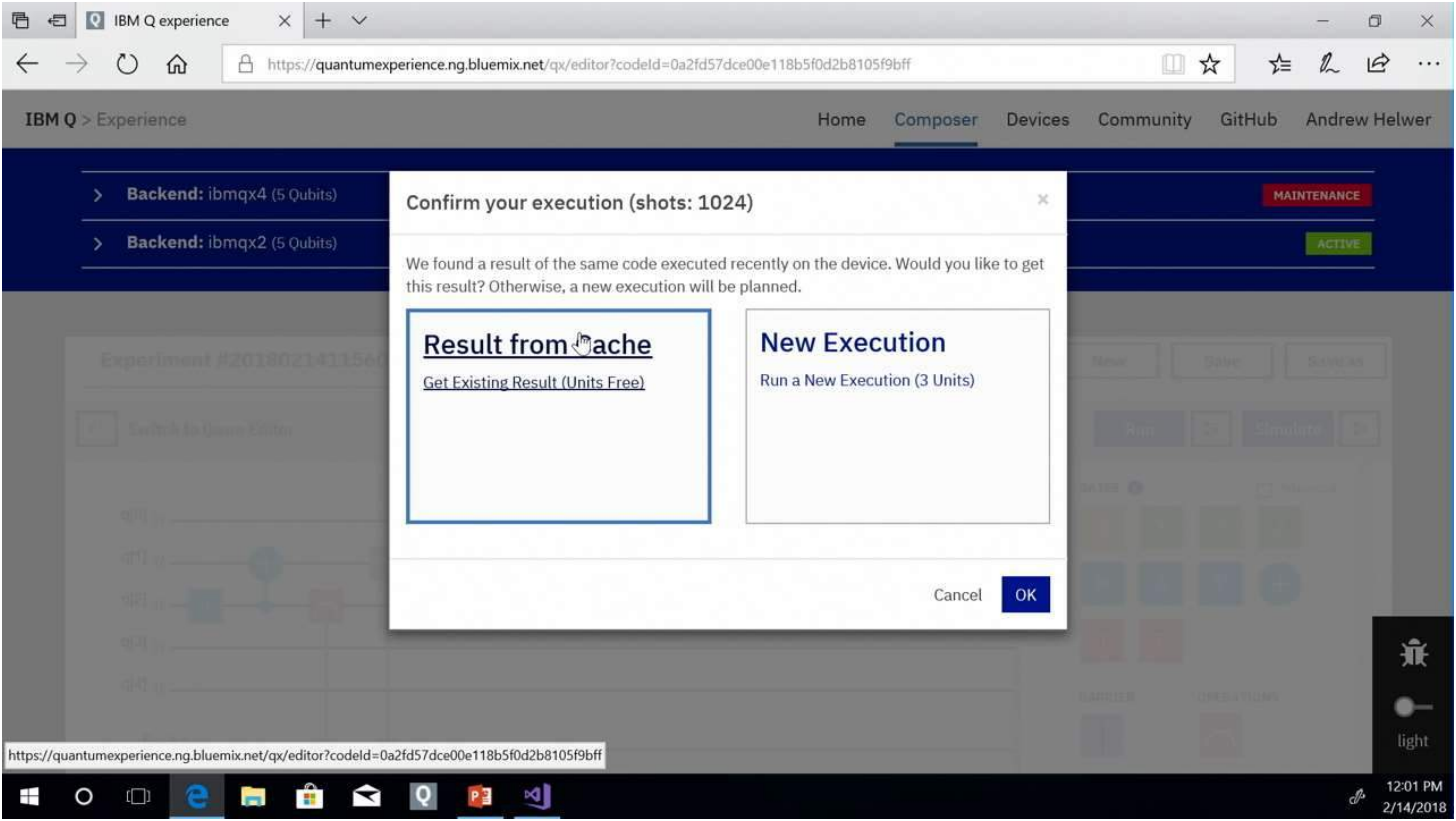

IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor
IBM Q > Experience
Home
Composer
Devices
Community
GitHub
Andrew Helwer
Backend: ibmqx4 (5 Qubits)
MAINTENANCE
Backend: ibmqx2 (5 Qubits)
ACTIVE
New experiment
New
Save
Save as
Switch to Qasm Editor
Backend: ibmqx4
My Units: 15
Experiment Units: 3
Run
Simulate
q[0] |0⟩
q[1] |0⟩
q[2] |0⟩
q[3] |0⟩
q[4] |0⟩
c 0 5
GATES
Advanced
id
X
Y
Z
H
S
S†
T
T†
BARRIER
OPERATIONS
light
11:54 AM
2/14/2018

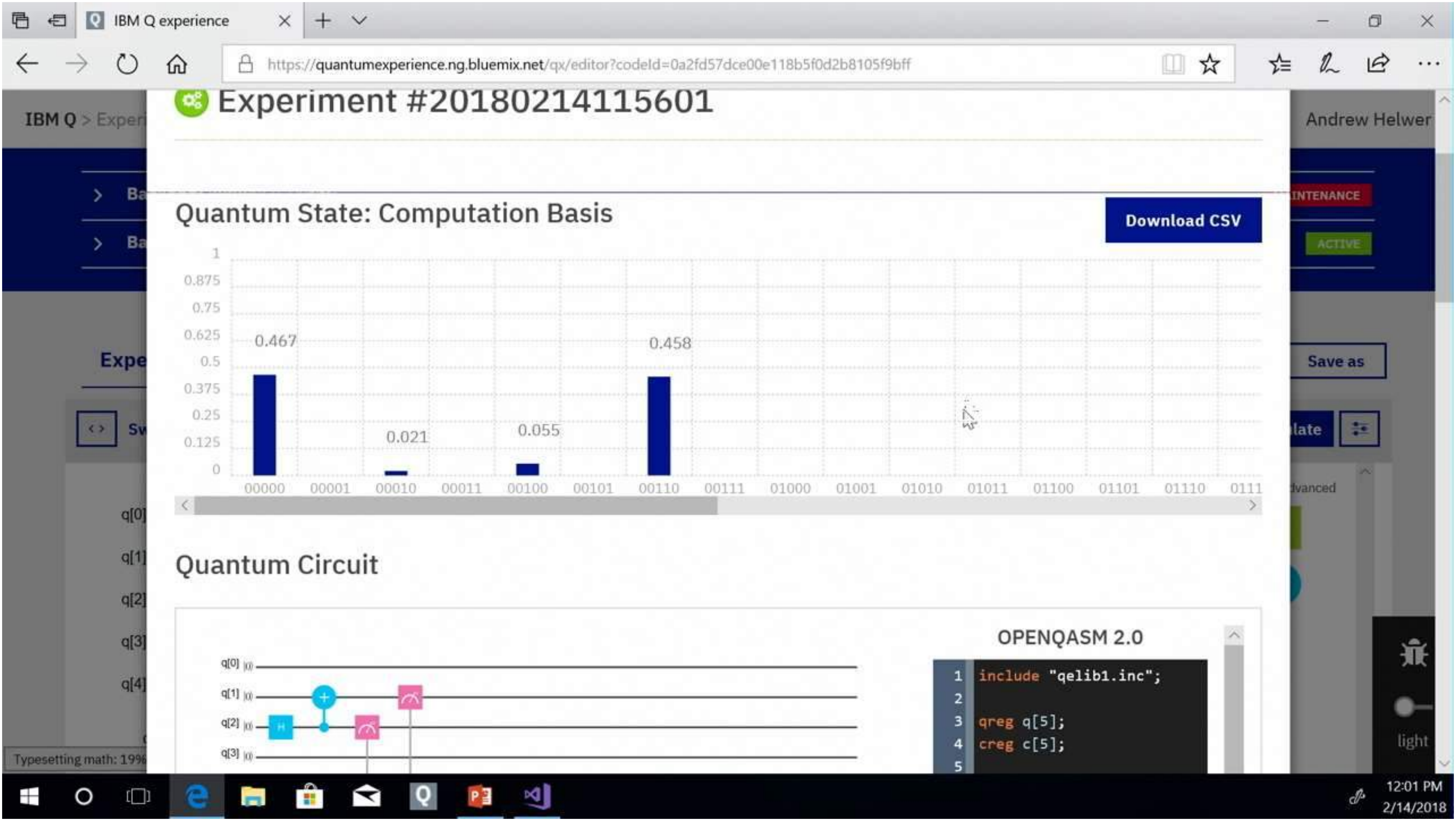

IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor
New experiment
New
Save
Save as
Switch to Qasm Editor
Backend: ibmqx4
My Units: 15
Experiment Units: 3
Run
Simulate
q[0] |0⟩
q[1] |0⟩
q[2] |0⟩
q[3] |0⟩
q[4] |0⟩
c 0 5
GATES
Advanced
id
X
Y
Z
H
S
S†
T
T†
BARRIER
OPERATIONS
light
Quantum Scores
Refresh
11:54 AM
2/14/2018


IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor?codeId=0a2fd57dce00e118b5f0d2b8105f9bff
Experiment #20180214115601
Add a description
New
Save
Save as
Switch to Qasm Editor
The execution is planned and it'll be the next one to be run! Please enjoy the usage of the Simulate feature in the meantime!
Run
Simulate
q[0] |0⟩
q[1] |0⟩
q[2] |0⟩
q[3] |0⟩
q[4] |0⟩
c 0 5
H
2
1
GATES
Advanced
id
X
Y
Z
H
S
S†
T
T†
BARRIER
OPERATIONS
Quantum Scores
Refresh
light
11:56 AM
2/14/2018


IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor?codeId=0a2fd57dce00e118b5f0d2b8105f9bff
Quantum Scores (1 scores)
Refresh
Remove All
Experiment #20180214115601
Add a description
Executions
Feb 14, 2018 11:56:14 AM
q[0]
q[1]
q[2]
q[3]
q[4]
H
c
2
1
IBM Q experience License Agreement
light
11:56 AM
2/14/2018


IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor?codeId=0a2fd57dce00e118b5f0d2b8105f9bff
Quantum Scores (1 scores)
Refresh
Remove All
Experiment #20180214115601
Add a description
Executions
Feb 14, 2018 11:56:14 AM
q[0]
q[1]
q[2]
q[3]
q[4]
H
c
light
11:59 AM
2/14/2018


IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor?codeId=0a2fd57dce00e118b5f0d2b8105f9bff
IBM Q > Experience
Home
Composer
Devices
Community
GitHub
Andrew Helwer
Backend: ibmqx4 (5 Qubits)
MAINTENANCE
Backend: ibmqx2 (5 Qubits)
ACTIVE
Confirm your execution (shots: 1024)
We found a result of the same code executed recently on the device. Would you like to get this result? Otherwise, a new execution will be planned.
Result from Cache
Get Existing Result (Units Free)
New Execution
Run a New Execution (3 Units)
Cancel
OK
light
https://quantumexperience.ng.bluemix.net/qx/editor?codeId=0a2fd57dce00e118b5f0d2b8105f9bff
12:01 PM
2/14/2018


IBM Q experience
https://quantumexperience.ng.bluemix.net/qx/editor?codeId=0a2fd57dce00e118b5f0d2b8105f9bff
Experiment #20180214115601
IBM Q
Andrew Helwer
Quantum State: Computation Basis
Download CSV
1
0.875
0.75
0.625
0.5
0.375
0.25
0.125
0
0.467
0.021
0.055
0.458
00000
00001
00010
00011
00100
00101
00110
00111
01000
01001
01010
01011
01100
01101
01110
0111
Quantum Circuit
q[0]
q[1]
q[2]
q[3]
H
OPENQASM 2.0
1 include "qelib1.inc";
2
3 qreg q[5];
4 creg c[5];
5
Save as
ACTIVE
Typesetting math: 19%
light
12:01 PM
2/14/2018