Quantum Computing

- Lecture 3 (April 30, 2025)

- Today:
  - Quantum unitary operations

Qubit

- Sigle-qubit state: The numbers α and β are **complex number** and $|\alpha|^2 + |\beta|^2 = 1$

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

- An $\boldsymbol{n}$**-qubit states** (in the computational basis)

$$|\boldsymbol{\phi}\rangle = \sum_{i=0}^{2^n-1} \alpha_i\,|i\rangle, \text{ where } \alpha_i \in \mathbb{C} \text{ and } \sum_{i=0}^{n-1} |\alpha_i|^2 = 1$$

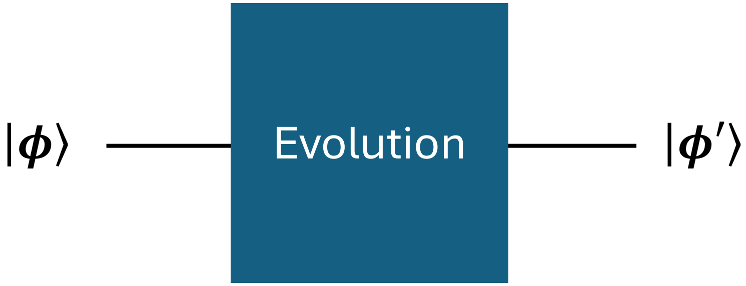
- General description: Let $\{|\phi_0\rangle, |\phi_1\rangle, |\phi_2\rangle, \ldots, |\phi_{N-1}\rangle\}$ be an orthonormal basis

$$|\boldsymbol{\phi}\rangle = \sum_{i=0}^{N-1} \alpha_i\,|\phi_i\rangle, \text{ where } \alpha_i \in \mathbb{C} \text{ and } \sum_{i=0}^{N-1} |\alpha_i|^2 = 1$$

Qubit

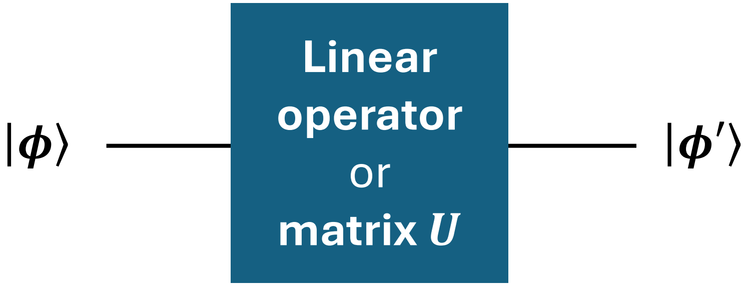
- Some operations introduced last week:
  - **Adjoint:** $\boldsymbol{U}^{\dagger} = (\boldsymbol{U}^{*})^{\boldsymbol{T}} = \left(\boldsymbol{U}^{\boldsymbol{T}}\right)^{*}$
  - **Inner product/Outer product:** $\langle\psi|\phi\rangle$, $|\psi\rangle\langle\phi|$
  - **Tensor product:** $|\phi\rangle \otimes |\phi\rangle = |\phi\phi\rangle$, $\boldsymbol{U_1} \otimes \boldsymbol{U_2}$

Unitary Operation



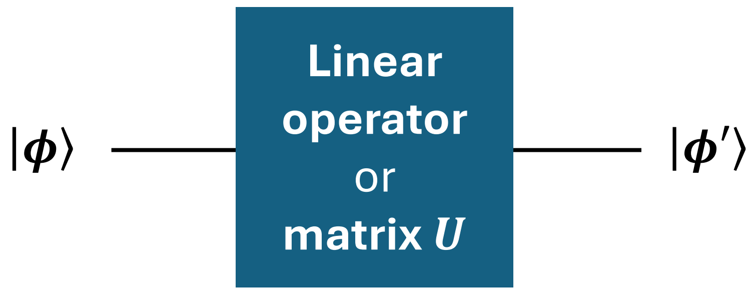
- The **Schrödinger equation** describes the evolution of the quantum state of an isolated system
  - The equation is **linear** (i.e., any linear combination of solutions is a solution)
- $\Rightarrow$ The evolution of quantum states is also linear
  - Always keep in mind: **linear operations $\Leftrightarrow$ matrices!**
- We use **linear operators (or matrices) to describe evolutions of quantum states**.

Unitary Operation



- We use **linear operators** (or matrices) to describe evolutions of quantum states.
- Observations:
  - (1) A quantum state – (evolution) → another quantum state
  - (2) By definition, a quantum state is a unit vector (normalized condition)

Unitary Operation



- We use **linear operators** (or matrices) to describe evolutions of quantum states.
- Observations:
  - (1) A quantum state – (evolution) → another quantum state
  - (2) By definition, a quantum state is a unit vector (normalized condition)
- **Quantum evolutions preserve the norm!**
  - Let $\boldsymbol{U}$ denote such a linear operation. **For any quantum state** $|\boldsymbol{\phi}\rangle$**,** $\||\boldsymbol{\phi}\rangle\| = \|\boldsymbol{U}|\boldsymbol{\phi}\rangle\| = \mathbf{1}$

Unitary Operation

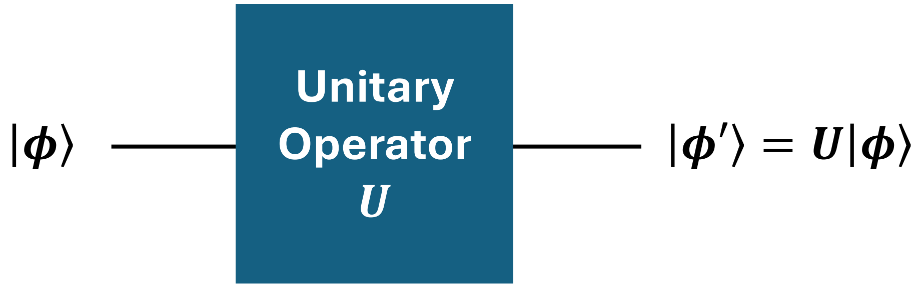
Some Linear Algebra – **Unitary**:

- **Unitary matrices** (unitary operators, or simply unitaries)
- A square matrix $\boldsymbol{U}$ is a unitary if **one of the following conditions holds**:
  - (1) **For any** $|\boldsymbol{\phi}\rangle$, $\||\boldsymbol{\phi}\rangle\| = \|\boldsymbol{U}|\boldsymbol{\phi}\rangle\|$
  - (2) $\boldsymbol{U}^{\dagger} = \boldsymbol{U}^{-1}$ (or $\boldsymbol{U}^{\dagger}\boldsymbol{U} = \boldsymbol{I}$)
  - …
- Exercise: (1) $\Longleftrightarrow$ (2)

- **Hermitian**: A matrix (or linear operator) $\boldsymbol{U}$ is *Hermitian* or *self-adjoint* if $\boldsymbol{U} = \boldsymbol{U}^{\dagger}$
- **Normal operator/matrix:** $\boldsymbol{U}\boldsymbol{U}^{\dagger} = \boldsymbol{U}^{\dagger}\boldsymbol{U}$ (but not necessarily $= \boldsymbol{I}$)
- Quick thought: Unitary $\Longrightarrow$ Normal

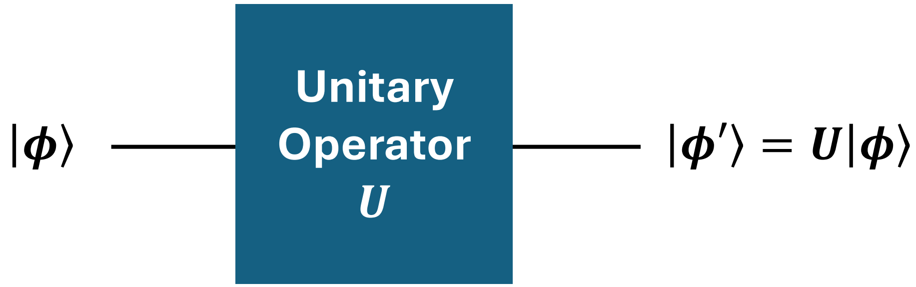
- **Quantum evolutions (linear operators or matrices) preserve the norm**
  - Let $\boldsymbol{U}$ denote such an operation. **For any quantum state** $|\boldsymbol{\phi}\rangle$, $\||\boldsymbol{\phi}\rangle\| = \|\boldsymbol{U}|\boldsymbol{\phi}\rangle\| = \mathbf{1}$

Unitary Operation



- We use a **unitary** to describe the evolution of a quantum state.
- In quantum computing, we use **unitary operations** to operate qubit(s)

Unitary Operation



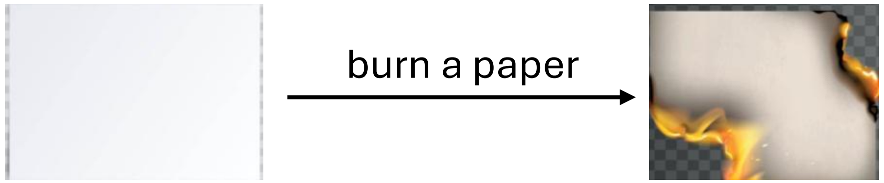
- We use a **unitary** to describe the evolution of a quantum state.
- In quantum computing, we use **unitary operations** to operate qubit(s)
    - Unitaries are invertible $\Longrightarrow$ Unitary operations are always **reversible**

- In contrast to classical computing, quantum computing relies on **reversible computation**

Unitary Operation

Some physics (or philosophy?):

- In the real world, there are some operations that are **believed to be irreversible**:

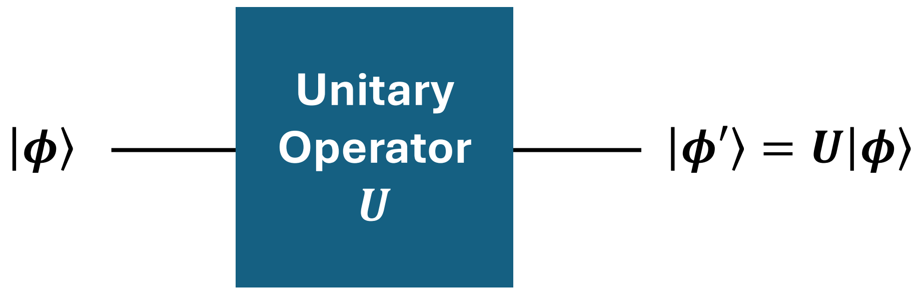


- **Quantum physics: Information must be preserved and cannot be erased** (unless you are dealing with a black hole) – There must exists some unitary $\boldsymbol{U}$ (**in theory**) such that you can...

- ...if you can **isolate the system** (pure state vs mixed state, will be introduced in the future)
- and **find the right unitary operator** (very hard)!

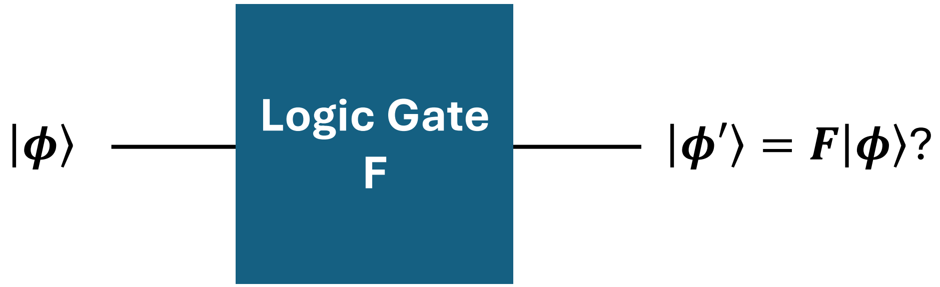
(source of images: Vector)

Quantum Logic Gates



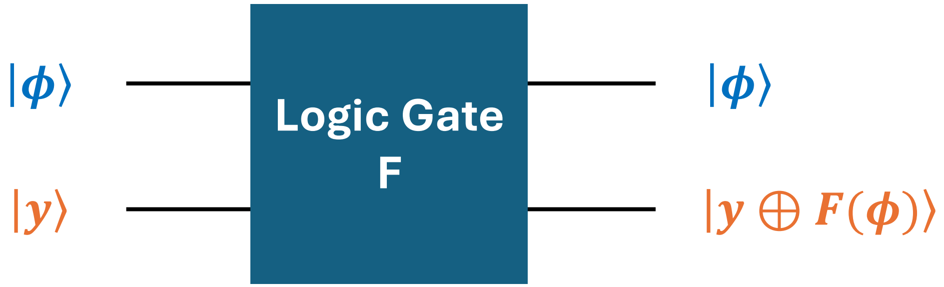
- Quantum computing relies on **unitary operations**
  - Any unitary matrix specifies a valid quantum gate/operation/algorithm
- Similar to classical computing, we use **logic gates** as the basic building blocks in quantum computing

Quantum Logic Gates



- Quantum computing relies on **unitary operations**
  - Any unitary matrix specifies a valid quantum gate/operation/algorithm
- Similar to classical computing, we use **logic gates** as the basic building blocks in quantum computing
  - The quantum logic gates should obey the rules in quantum mechanics
  - NOT gate is reversible
  - AND, NAND, OR, and XOR gates are irreversible

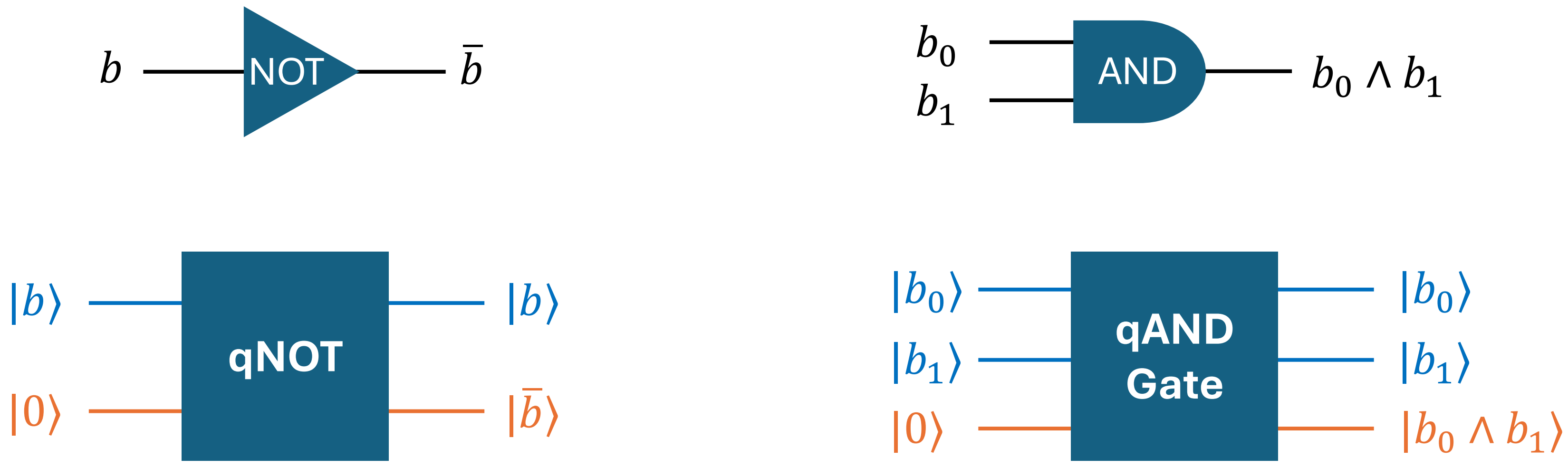
Quantum Logic Gates



- Quantum computing relies on **unitary operations**
  - Any unitary matrix specifies a valid quantum gate/operation/algorithm
- Similar to classical computing, we use **logic gates** as the basic building blocks in quantum computing
  - The quantum logic gates should obey the rules in quantum mechanics
  - NOT gate is reversible
  - AND, NAND, OR, and XOR gates are irreversible – We **preserve the input** to make them reversible…
  - …and **store the result** using ancilla qubit(s) (or auxiliary, temporary workspace) which are usually set as 0

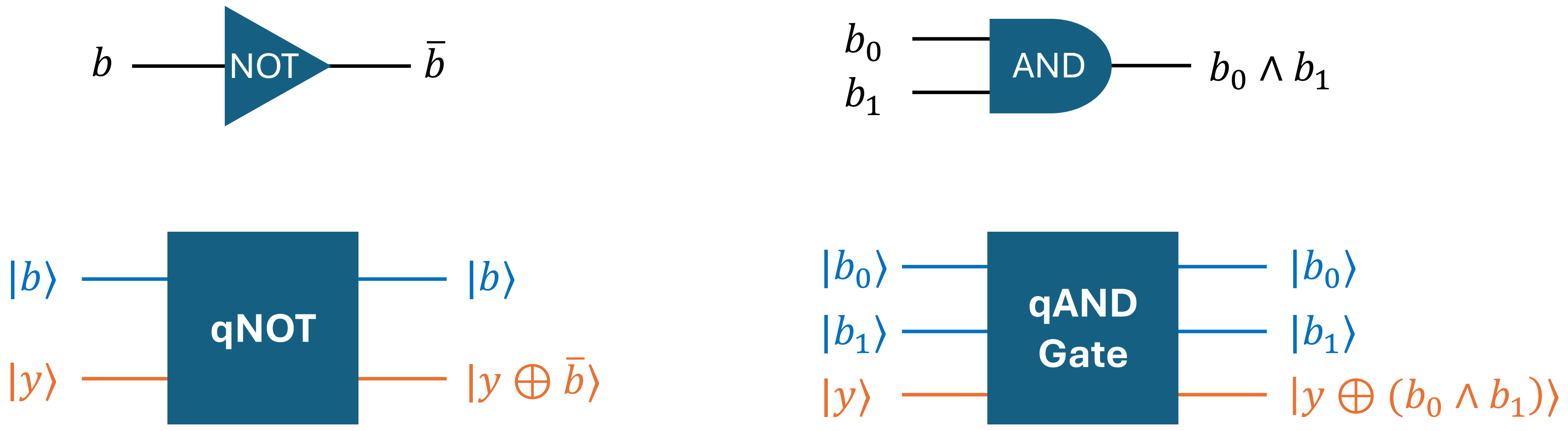
Quantum Logic Gates

- Examples (let's focus on the computational basis):



Quantum Logic Gates

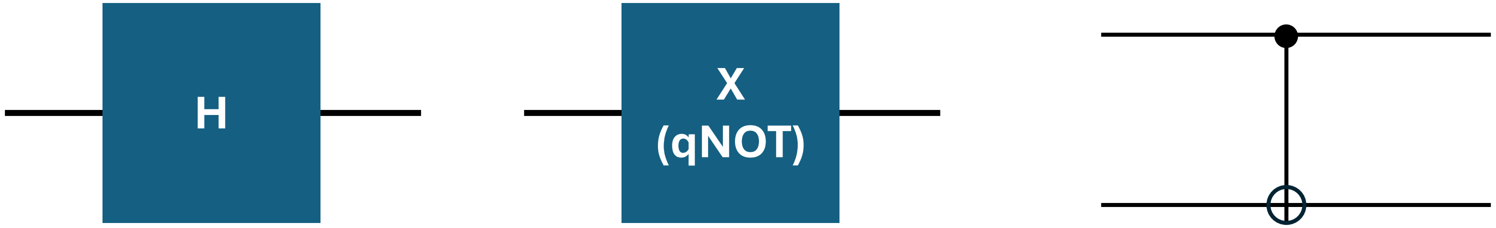
- Examples (let's focus on the computational basis):



- How can we define the qOR and qNAND gates?

Quantum Gates

- More basic quantum gates:



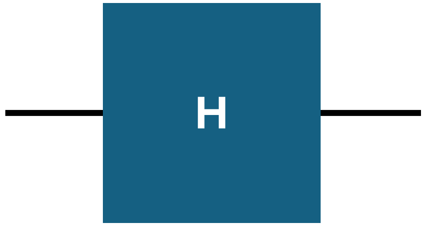
Hadamard Matrix | Pauli-X (The qNOT gate) | CNOT (The Controlled NOT/X gate) | …

- Their matrix representations (in the computational basis):

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

Quantum Gates

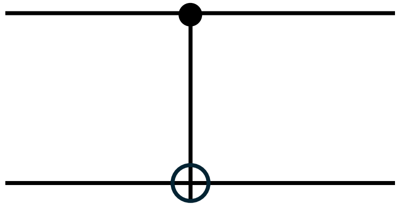
- Hadamard Matrix:



$$:= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

- $\boldsymbol{H}|0\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{|\mathbf{0}\rangle + |\mathbf{1}\rangle}{\sqrt{2}}$
- $\boldsymbol{H}|1\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \frac{|\mathbf{0}\rangle - |\mathbf{1}\rangle}{\sqrt{2}}$
- By Exercise 5 in Week 1, $\boldsymbol{H}^{\mathbf{2}} = \boldsymbol{I}$
- Turns a qubit to "halfway" between $|0\rangle$ and $|1\rangle$.

- CNOT:



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

- CNOT$|\mathbf{0}\rangle|\boldsymbol{b}\rangle \to |\mathbf{0}\rangle|\boldsymbol{b}\rangle$
- CNOT$|\mathbf{1}\rangle|\boldsymbol{b}\rangle \to |\mathbf{1}\rangle|\mathbf{1} \oplus \boldsymbol{b}\rangle = |\mathbf{1}\rangle|\overline{\boldsymbol{b}}\rangle$
- Classical counterpart:
  - If the first bit = 0: do nothing;
  - Else: Flip the second bit

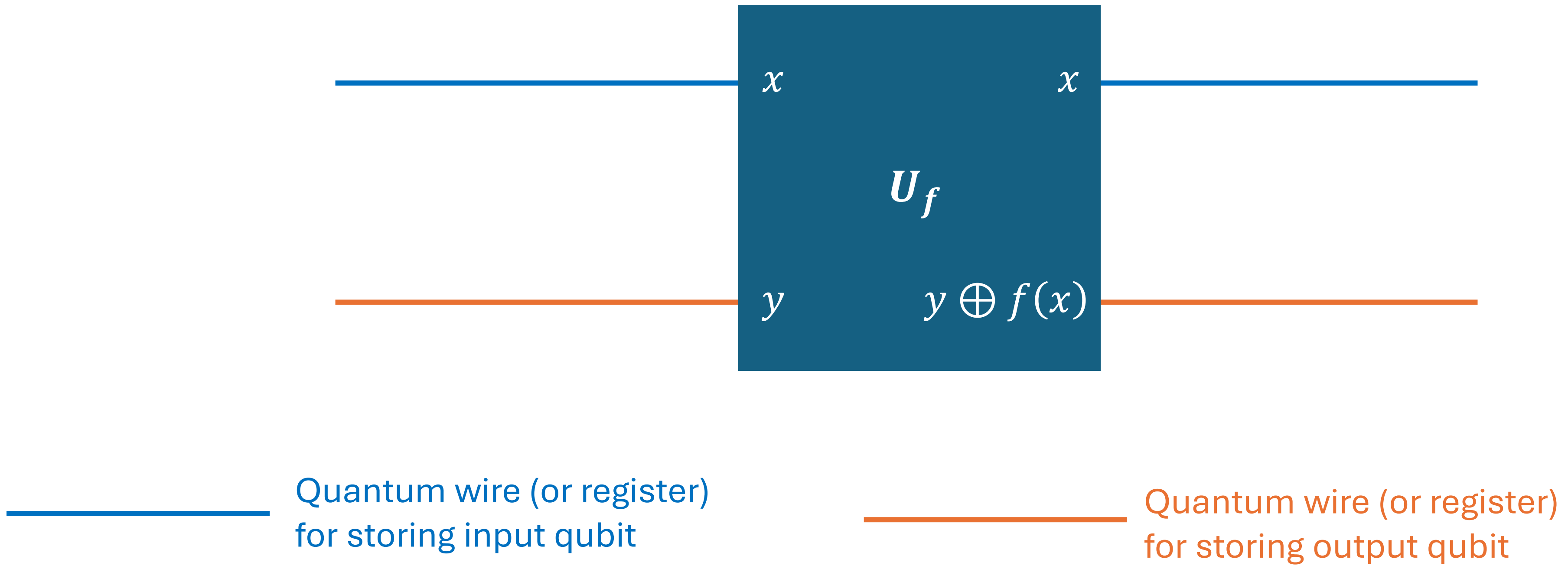
Quantum Gates

- Let f be a finite *computable* function.
    - There exists a circuit that implements f
    - Construct circuits using logic gates

- In Quantum Computing:
    - Construct a quantum circuit to compute f (using quantum logic gates)
    - Require reversible computation, while f may not be reversible

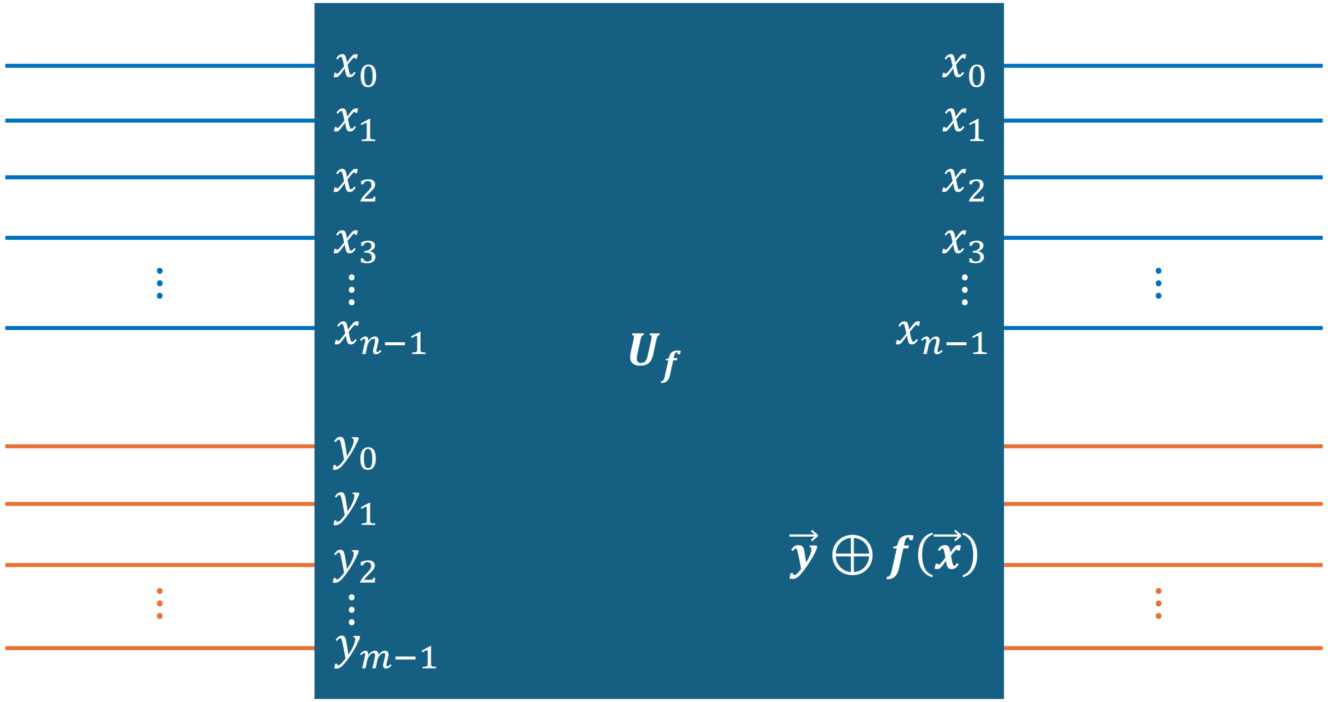
Quantum Circuits

- Generally, let $f: \{0,1\} \to \{0,1\}$ be a computable bit function.
- Define the quantum version of f as:



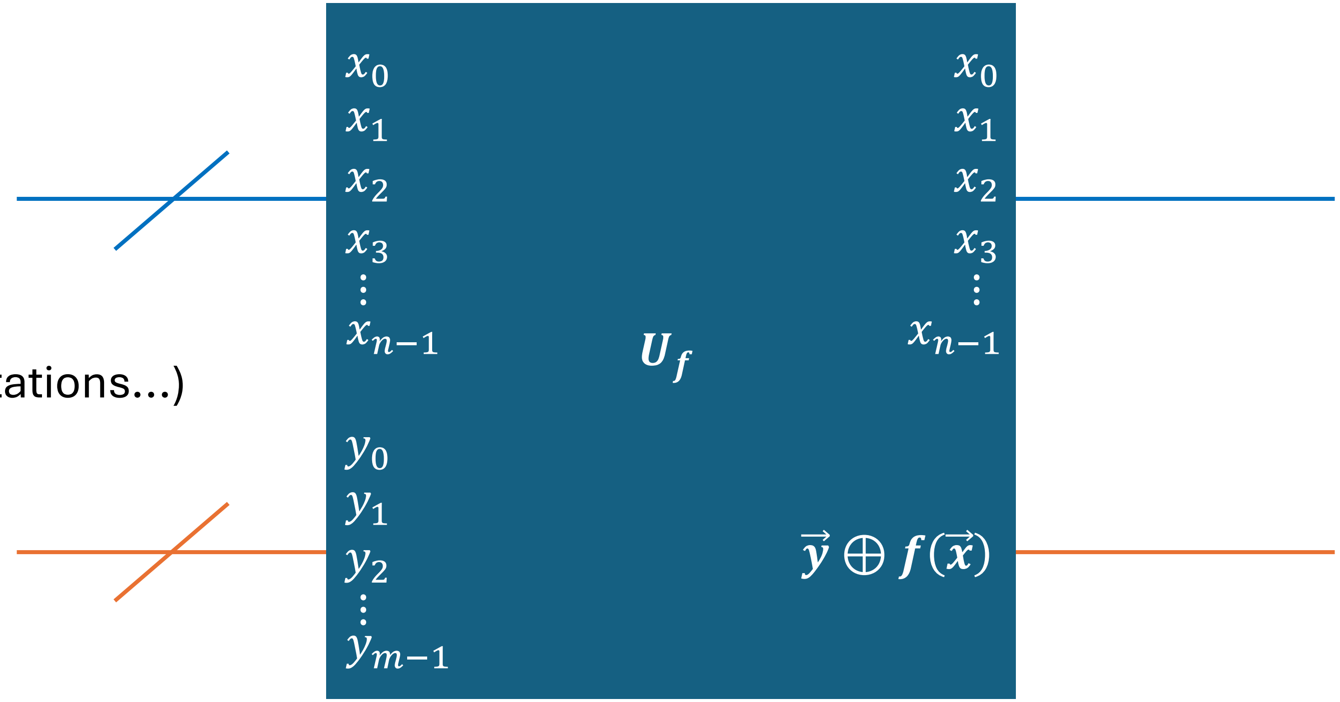
Quantum Circuits

- More generally, let $f: \{0,1\}^n \rightarrow \{0,1\}^m$ be a computable function



Quantum Circuits

- More generally, let $f: \{0,1\}^n \rightarrow \{0,1\}^m$ be a computable function
  - $\boldsymbol{U_f}$ **is also a unitary**



(Simplify the notations...)

Quantum Circuits

- Let f be a finite *computable* function.
  - There exists a circuit that implements f
  - Construct circuits using logic gates

- In Quantum Computing:
  - Construct a quantum circuit to compute f (using quantum logic gates)
  - Require reversible computation, while f may not be reversible
  - **Generic transformation:** $f \rightarrow \boldsymbol{U_f}$ (make it unitary using ancilla qubits)

Quantum Circuits

- Let f be a finite *computable* function.
  - There exists a circuit that implements f
  - Construct circuits using logic gates

- In Quantum Computing:
  - Construct a quantum circuit to compute f (using quantum logic gates)
  - Require reversible computation, while f may not be reversible
  - **Generic transformation:** $f \rightarrow \boldsymbol{U_f}$ (make it unitary using ancilla qubits)

- **Any classical algorithm (circuit) can be simulated by a quantum algorithm (circuit)**
  - Classical algorithms/circuits are built from classical logic gates
  - Classical logic gates can be simulated using reversible quantum logic gates
  - Quantum logic gates can be composed into quantum algorithms/circuits

Evaluation on Superposition

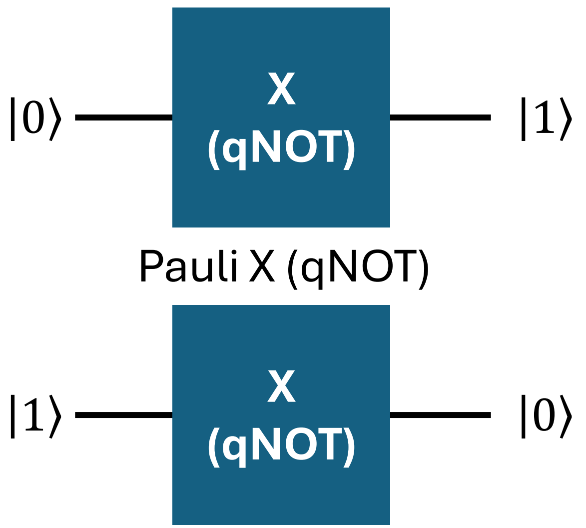
- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{Uv_1} + c_2\boldsymbol{Uv_2}$

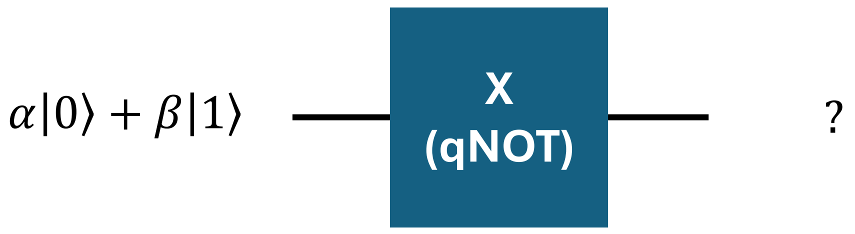
- Quantum gates (Unitaries) operate on superposition: **Linearity**

Evaluation on Superposition

- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{Uv_1} + c_2\boldsymbol{Uv_2}$

- Quantum gates (Unitaries) operate on superposition: **Linearity**

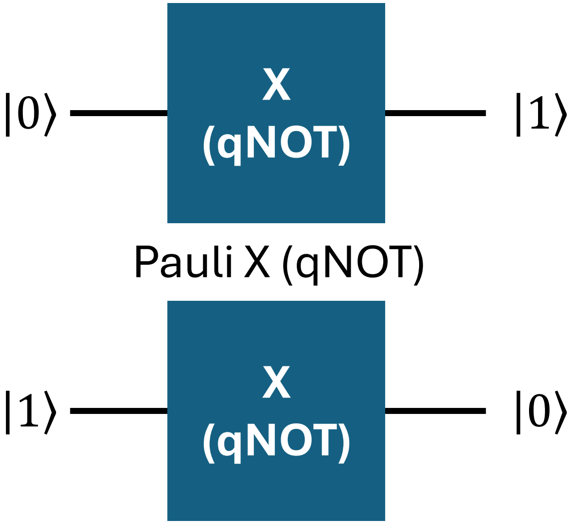


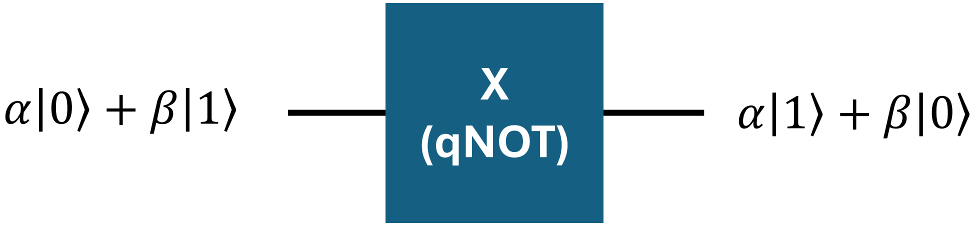


Evaluation on Superposition

- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{U}\boldsymbol{v_1} + c_2\boldsymbol{U}\boldsymbol{v_2}$

- Quantum gates (Unitaries) operate on superposition: **Linearity**





Evaluation on Superposition

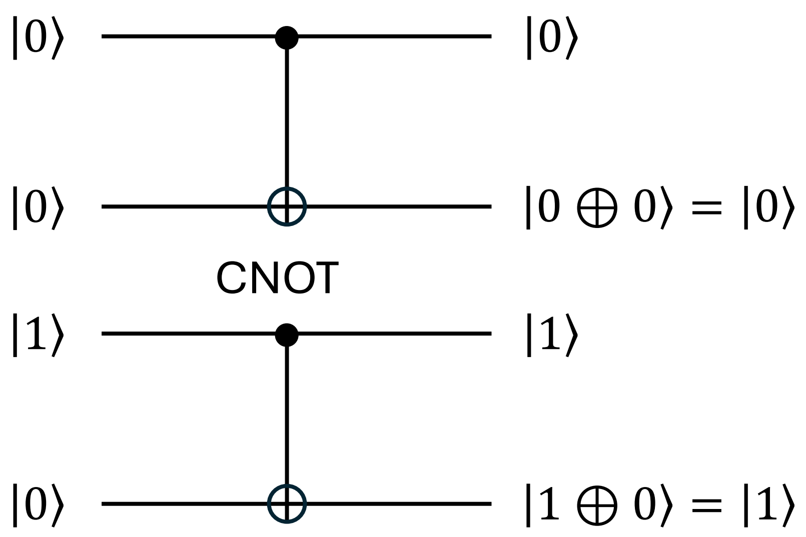
- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{U}\boldsymbol{v_1} + c_2\boldsymbol{U}\boldsymbol{v_2}$

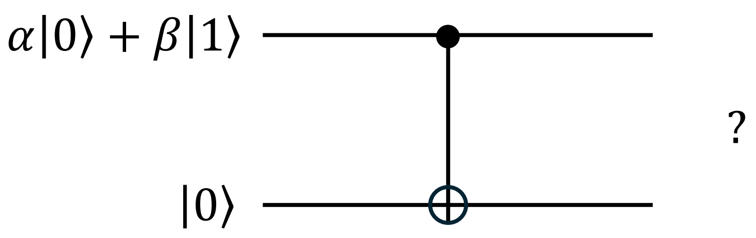
- Quantum gates (Unitaries) operate on superposition: **Linearity**

Evaluation on Superposition

- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{Uv_1} + c_2\boldsymbol{Uv_2}$

- Quantum gates (Unitaries) operate on superposition: **Linearity**

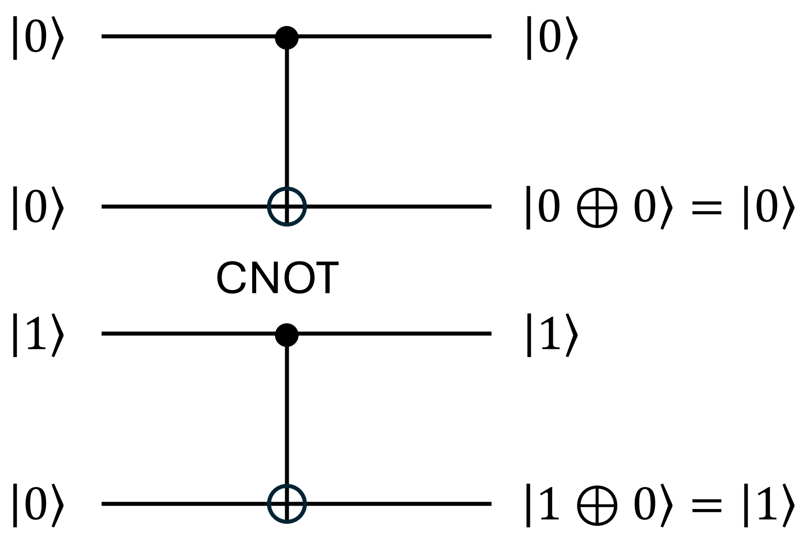


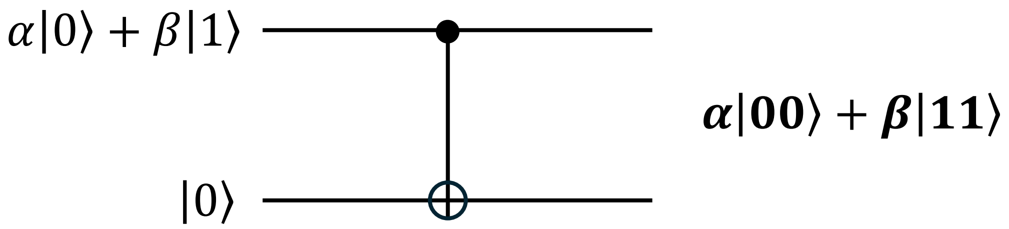


Evaluation on Superposition

- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{U}\boldsymbol{v_1} + c_2\boldsymbol{U}\boldsymbol{v_2}$

- Quantum gates (Unitaries) operate on superposition: **Linearity**

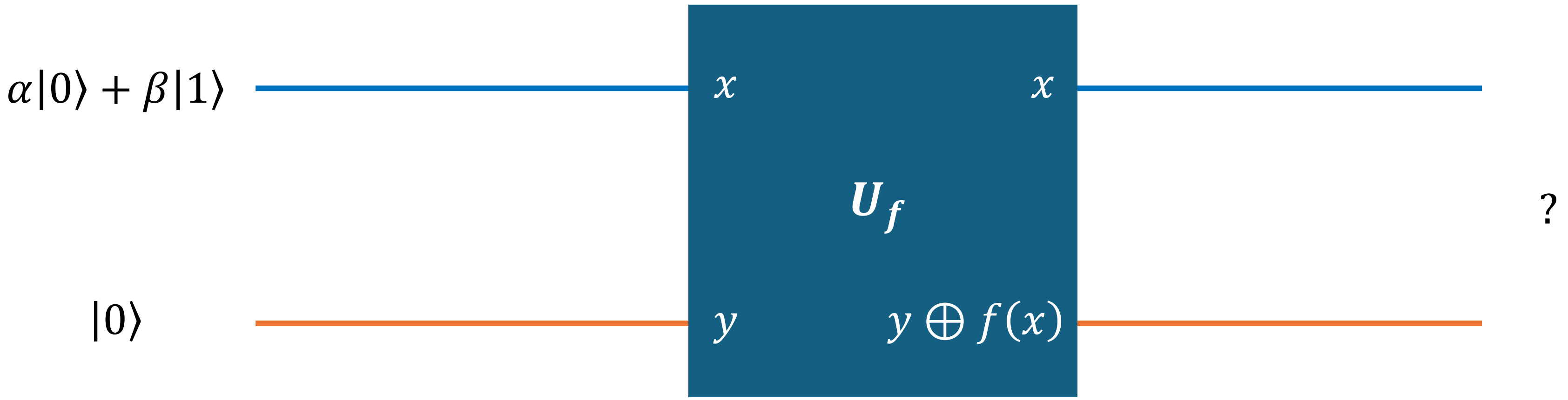




Evaluation on Superposition

- Any quantum gate is a unitary operator
  - A unitary operator has **linearity:** $\boldsymbol{U}(c_1\boldsymbol{v_1} + c_2\boldsymbol{v_2}) = c_1\boldsymbol{U}\boldsymbol{v_1} + c_2\boldsymbol{U}\boldsymbol{v_2}$

- Quantum gates (Unitaries) operate on superposition: **Linearity**



Summary

- Quantum gates are described by unitaries
  - Any unitary also specifies a valid quantum gate

- Basic quantum gates: Hadamard, Pauli-X (NOT), CNOT, …

- Make a classical computable function unitary $f \rightarrow \boldsymbol{U_f}$
  - Any classical algorithm can be simulated by quantum computers

- Evaluation on superposition
  - View any quantum gate as a unitary linear operator (matrix)
  - Quantum gates act on superpositions according to linearity

Topics for Next Week

- Deutsch’s algorithm
- More linear algebra on unitary operations
- The Deutsch-Jozsa algorithm
- Simple measurement and superdense coding

References

- **[NC00]:** Sections 1.3.1 – 1.3.5 (no-cloning theorem), 1.4.1 – 1.4.2