**1. Single Qubit Gates**

### 1. What is a qubit’s state?

* A qubit can be **|0⟩** (like classical 0), **|1⟩** (like classical 1), or a **superposition** (mix of both).
* We write a qubit state as

∣ψ⟩=α∣0⟩+β∣1⟩

where α and β are complex numbers, and |α|² + |β|² = 1.

**(a) Pauli-X Gate (NOT Gate)**

* **Matrix:**

X = [ [0 , 1],

[1 , 0] ]

* **Action:** Flips the state of a qubit.
  + X|0⟩ = |1⟩
  + X|1⟩ = |0⟩
* **Use:** Equivalent to classical NOT gate.

**(b) Pauli-Y Gate**

* **Matrix:**

Y = [ [0 , -i],

[i , 0] ]

* **Action:** Performs both a bit-flip and a phase-flip.
  + Y|0⟩ = i|1⟩
  + Y|1⟩ = -i|0⟩
* **Use:** Used in quantum rotations.

**(c) Pauli-Z Gate (Phase Flip)**

* **Matrix:**

Z = [ [1 , 0],

[0 , -1] ]

* **Action:** Leaves |0⟩ unchanged, flips the phase of |1⟩.
  + Z|0⟩ = |0⟩
  + Z|1⟩ = -|1⟩
* **Use:** Essential for creating relative phase changes.

### (d) Hadamard Gate (H Gate)

* **Matrix:**

H = (1/sqrt(2)) \* [ [1 , 1],

[1 , -1] ]

* **Action:** Creates superposition.
  + H|0⟩ = (|0⟩ + |1⟩)/√2
  + H|1⟩ = (|0⟩ - |1⟩)/√2
* **Use:** Foundation of quantum parallelism and quantum algorithms.

**2. Multiple Qubit Gates**

**(a) CNOT Gate (Controlled-NOT)**

A **multi-qubit gate** is an operation that acts on **two or more qubits at the same time**.

* Unlike single qubit gates (which only rotate one qubit), multi-qubit gates can **create entanglement** — a special quantum link between qubits.
* These gates are represented by **4×4 matrices** for 2 qubits, **8×8** for 3 qubits, etc.
* **Matrix:**

CNOT = [ [1 , 0 , 0 , 0],

[0 , 1 , 0 , 0],

[0 , 0 , 0 , 1],

[0 , 0 , 1 , 0] ]

* **Action:**
  + If control qubit = |0⟩, target is unchanged.
  + If control qubit = |1⟩, target flips.
* **Example:**
  + Input |10⟩ → |11⟩
  + Input |11⟩ → |10⟩
* **Use:** Creates entanglement, crucial for quantum error correction.

**(b) SWAP Gate**

* **Matrix:**

SWAP = [ [1 , 0 , 0 , 0],

[0 , 0 , 1 , 0],

[0 , 1 , 0 , 0],

[0 , 0 , 0 , 1] ]

* **Action:** Swaps two qubits.
  + Input |01⟩ → |10⟩
  + Input |10⟩ → |01⟩
* **Use:** Rearranges qubits in circuits, useful in hardware with limited connectivity.

**(c) Toffoli Gate (CCNOT Gate)**

* **Matrix:** 8x8 (for 3 qubits).
* **Action:**
  + Two qubits act as controls, the third is target.
  + Target flips only if both controls are 1.
* **Example:**
  + Input |110⟩ → |111⟩
  + Input |111⟩ → |110⟩
* **Use:** Universal for classical reversible computing, used in arithmetic circuits.

**(d) Controlled-Z (CZ Gate)**

* **Matrix:**

CZ = [ [1 , 0 , 0 , 0],

[0 , 1 , 0 , 0],

[0 , 0 , 1 , 0],

[0 , 0 , 0 , -1] ]

* **Action:**
  + If control = |0⟩, no effect.
  + If control = |1⟩, applies Z to target (adds phase -1).
* **Example:**
  + Input |11⟩ → -|11⟩
* **Use:** Creates entanglement, especially for Bell states.

**Measurements in Bases vs Computational Basis**

**1. Computational Basis Measurement (Z-Basis)**

* In quantum computing, the **computational basis** is the most commonly used basis.
* It consists of the two states:  
  |0⟩ = [1, 0]ᵀ and |1⟩ = [0, 1]ᵀ.
* Every quantum state can be written as a linear combination of these basis states:  
  |ψ⟩ = α|0⟩ + β|1⟩, where |α|² + |β|² = 1.
* When we perform a measurement in the computational basis:
  + The qubit collapses to |0⟩ with probability |α|².
  + The qubit collapses to |1⟩ with probability |β|².
* This means that although the qubit may exist in superposition before measurement, the act of measuring destroys the superposition and forces the qubit into a definite classical state (0 or 1).
* **Example:** If |ψ⟩ = (1/√2)(|0⟩ + |1⟩), measurement results in 0 or 1, each with probability 0.5.

**2. Measurement in Other Bases**

* A qubit can also be measured in bases other than the computational basis.
* A basis is defined as a pair of orthogonal states. For example:  
  { |u⟩ , |v⟩ } such that ⟨u|v⟩ = 0.
* Measuring in a different basis changes the outcomes that are observed.
* The measurement result will collapse the qubit to one of the basis states with a probability equal to the squared magnitude of its amplitude in that basis.
* **Example – Hadamard (X-Basis):**
  + The X-basis consists of the states:  
    |+⟩ = (|0⟩ + |1⟩)/√2, and |−⟩ = (|0⟩ − |1⟩)/√2.
  + If we measure a qubit in this basis, the outcomes are |+⟩ or |−⟩ instead of |0⟩ or |1⟩.
  + This type of measurement is useful when we want to detect quantum interference or relative phase information between |0⟩ and |1⟩.
* Other bases (like the Y-basis) are also used, for example:  
  |i+⟩ = (|0⟩ + i|1⟩)/√2 and |i−⟩ = (|0⟩ − i|1⟩)/√2.  
  These are useful when studying phase properties of qubits.

# ****Quantum Circuits****

* A **quantum circuit** is a model for quantum computation in which computation is represented as a sequence of **quantum gates** applied to qubits.
* It is similar to a classical logic circuit, but instead of bits and logic gates, it uses **qubits** and **unitary gates**.
* Quantum circuits can perform operations like **superposition, entanglement, interference**.

### ****2. Components of a Quantum Circuit****

1. **Qubits** – Wires in a circuit represent qubits, the basic unit of quantum information.
2. **Quantum Gates** – Unitary operations applied to qubits (e.g., X, H, CNOT).
3. **Measurements** – At the end of the circuit, qubits are measured, collapsing them into classical values (0 or 1).
4. **Classical Control** – Sometimes classical bits are used to conditionally apply quantum gates.

### ****3. Working Principle****

* A quantum algorithm starts with all qubits initialized to a state, usually |0⟩.
* Quantum gates are applied in sequence, transforming the state of the qubits.
* Qubits may become **entangled**, meaning their states are correlated.
* Finally, measurement is performed to extract classical results.
* The probability of each outcome depends on the **quantum amplitudes** created by the gates.

### ****4. Example**** (a) ****Superposition Circuit****

* Apply Hadamard gate (H) on |0⟩.
* Circuit: |0⟩ ──H──●
* Resulting state: (|0⟩ + |1⟩)/√2.
* Measurement gives 0 or 1 with equal probability.

### ****5. Advantages of Quantum Circuits****

* Can simulate problems intractable for classical computers.
* Allow parallelism through superposition.
* Use entanglement for correlations beyond classical limits.
* Basis for implementing quantum algorithms such as **Shor’s Algorithm** and **Grover’s Algorithm**.

**Qubit Copying Circuit**

 In quantum mechanics, **you cannot perfectly copy an unknown qubit** (this is the **No-Cloning Theorem**).

 But we can **partially "copy" information** using a circuit that entangles the qubit with another blank qubit.

**1. No-Cloning Theorem**

* In quantum mechanics, it is **impossible to perfectly copy (clone)** an unknown qubit state.
* This is called the **No-Cloning Theorem**.
* If we have a qubit in state:

∣ψ⟩=α∣0⟩+β∣1⟩

there is no quantum operation that can produce two identical copies of this state.

* This is because quantum states are continuous and cloning would violate linearity of quantum mechanics.

**2. Approximate Copying**

* Even though we cannot perfectly copy a qubit, we can **transfer its information** into another qubit using entanglement and classical communication.
* One simple circuit that *looks like copying* is the **CNOT-based circuit**.

**3. Qubit Copying Using CNOT**

* Suppose we want to copy the state of a qubit |ψ⟩ into another qubit initially in |0⟩.
* We use a **CNOT gate** with:
  + Control qubit = |ψ⟩
  + Target qubit = |0⟩

**Action:**

* If |ψ⟩ = |0⟩, then output is |00⟩.
* If |ψ⟩ = |1⟩, then output is |11⟩.
* If |ψ⟩ is in superposition (α|0⟩ + β|1⟩), the output becomes entangled:

α∣00⟩+β∣11⟩

* This is not a true copy, but an **entangled state** where the target qubit contains correlated information.

**4. Applications**

* Used in **entanglement generation** (e.g., creating Bell states).
* Basis for **quantum teleportation** (where qubit information is transferred, not copied).
* Useful for **error correction codes**.

**1. Bell States**

* Bell states are a special set of **two-qubit entangled states**.
* They represent the simplest and most important examples of quantum entanglement.
* They are also called **EPR pairs** (Einstein–Podolsky–Rosen states).

**The Four Bell States**

The four Bell states are:

1. **Φ⁺** = (|00⟩ + |11⟩) / √2
2. **Φ⁻** = (|00⟩ − |11⟩) / √2
3. **Ψ⁺** = (|01⟩ + |10⟩) / √2
4. **Ψ⁻** = (|01⟩ − |10⟩) / √2

**Creation of Bell State (Φ⁺)**

* Start with two qubits in |00⟩.
* Apply **Hadamard gate (H)** on the first qubit.
* Apply **CNOT gate** with the first qubit as control, second as target.

Circuit:

|0⟩ ──H──■──

│

|0⟩ ─────X──

Output: (|00⟩ + |11⟩) / √2

**Applications of Bell States**

* Fundamental resource for quantum teleportation.
* Basis for quantum cryptography (Ekert protocol).
* Used to demonstrate nonlocality and violation of Bell’s inequalities.

**2. Quantum Teleportation**

* Quantum teleportation is a protocol that **transmits an unknown qubit state** from one location (Alice) to another (Bob), using:
  + A pair of entangled qubits (Bell state)
  + Classical communication (2 classical bits)

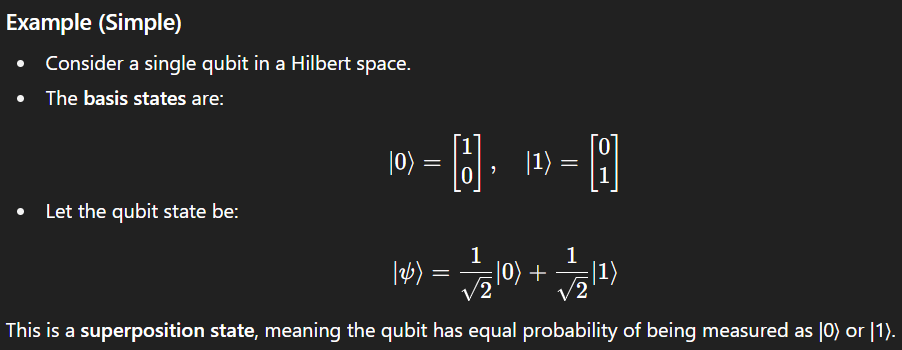
**Steps of Quantum Teleportation**

1. **Initial Setup**
   * Alice has qubit |ψ⟩ = α|0⟩ + β|1⟩ (unknown state).
   * Alice and Bob share a Bell state (Φ⁺ = (|00⟩ + |11⟩)/√2).
2. **Entangling Alice’s Qubits**
   * Alice applies CNOT between her unknown qubit and her part of Bell pair.
   * Then applies a Hadamard gate on the unknown qubit.
3. **Measurement**
   * Alice measures her two qubits in the computational basis.
   * She gets one of four possible results (00, 01, 10, 11).
   * She sends these **two classical bits** to Bob.
4. **Bob’s Correction**
   * Depending on Alice’s result, Bob applies a correction gate on his qubit:
     + If result = 00 → apply I (do nothing)
     + If result = 01 → apply X
     + If result = 10 → apply Z
     + If result = 11 → apply XZ
   * After correction, Bob’s qubit becomes **|ψ⟩ = α|0⟩ + β|1⟩** (the original state).

# Hilbert Spaces in Quantum Computation

A Hilbert space is a mathematical space used in quantum mechanics and quantum computation to describe quantum states. It is a complete vector space with an inner product, which allows us to represent superposition, entanglement, and measurement. Every qubit and quantum system exists in a Hilbert space.

1. **Definition**: A Hilbert space is a complete inner product space where quantum states are represented as vectors.
2. **Basis States**:
   * A single qubit lives in a 2-dimensional Hilbert space, spanned by basis states |0⟩ and |1⟩.
   * For n qubits, the Hilbert space has dimension 2^n.
3. **Superposition**: Any quantum state is a linear combination of basis states:  
   |ψ⟩ = α|0⟩ + β|1⟩, where α, β ∈ C and |α|² + |β|² = 1.
4. **Inner Product**: The inner product ⟨φ|ψ⟩ gives probability amplitudes.
5. **Normalization**: All quantum states are normalized so that the total probability equals 1.
6. **Operators**: Quantum gates are represented as unitary matrices that act on vectors in Hilbert space.
7. **Importance**: Hilbert spaces provide the mathematical foundation for superposition, interference, entanglement, and measurement in quantum computation.

****

# Products and Tensor Products

In quantum computation, when we have more than one qubit, we need a way to represent their combined state.  
This is done using the **tensor product**. It joins smaller Hilbert spaces to form a bigger space.

1. A **product state** is just two qubits written together, like |0⟩|1⟩ = |01⟩.
2. The **tensor product** is the mathematical rule for combining quantum states.
3. If qubit A = [a, b]ᵀ and qubit B = [c, d]ᵀ, then  
   A ⊗ B = [a·c, a·d, b·c, b·d]ᵀ
4. With more qubits, the Hilbert space size doubles each time:
   * 1 qubit → 2 states
   * 2 qubits → 4 states
   * 3 qubits → 8 states, and so on.
5. Tensor products also explain **entanglement**, which cannot be written as a simple product.

**Example 1: Tensor product**

* Qubit 1: |0⟩ = [1, 0]ᵀ
* Qubit 2: |1⟩ = [0, 1]ᵀ
* Tensor product: |0⟩⊗|1⟩ = [0, 1, 0, 0]ᵀ = |01⟩

# Matrices in Quantum Computation

* In quantum computation, **matrices** are the mathematical tools used to represent quantum states and operations.
* Qubits are expressed as **column vectors**, and quantum gates are represented as **unitary matrices**.
* The action of a quantum gate on a qubit is expressed as a **matrix-vector multiplication**.
* Matrices form the **backbone of quantum mechanics and quantum computing**.

## 3. Example (Matrix Operation)

* Apply the **Pauli-X gate** on |0⟩:
  + Pauli-X matrix:  
    X = [[0, 1],  
    [1, 0]]
  + Qubit state:  
    |0⟩ = [1, 0]ᵀ
  + Matrix multiplication:  
    X × |0⟩ = [[0, 1],  
    [1, 0]] × [1, 0]ᵀ = [0, 1]ᵀ = |1⟩
* **Result:** The Pauli-X gate flips |0⟩ to |1⟩.

## 2. Graphs in Quantum Computation

* A **graph** is a set of vertices (nodes) connected by edges (links).
* In quantum computation, graphs are used to model **circuits, entanglement, and algorithms**.

### Key Theory Points

1. Graphs represent **quantum circuits**: each line is a qubit, and each symbol is a gate.
2. **Graph states** are entangled states defined by vertices (qubits) and edges (entanglement).
3. Graphs are used in **quantum error correction codes** (e.g., surface codes, stabilizer codes).
4. **Quantum walks**, the quantum version of random walks, are performed on graphs.
5. Graphs are useful for **optimization problems** in quantum computing.
6. **Entanglement structure** between qubits can be visualized using graphs.
7. Graphs also represent **quantum networks** for communication.
8. Many **quantum algorithms** (Grover, Shor, etc.) use graph representations.
9. Graph theory connects quantum computing with **classical CS and mathematics**.
10. Hence, graphs are both a **visual tool** and a **theoretical framework**.

### Example (Graph State)

* Consider **two qubits connected by an edge**.
* Each qubit is a vertex, and the edge represents entanglement (e.g., a Bell state).
* This forms a simple **graph state** with entangled qubits.

# Sums Over Paths in Quantum Computation

* In classical physics, a particle follows only one definite path. In quantum circuits, each possible sequence of operations is like a path.
* The principle forms the **foundation of quantum algorithms like shor’s Algorithm , Grovers Algorithms uses this technique**.
* In quantum mechanics, particles explore **all possible paths simultaneously**.
* This principle is called the **Sum-Over-Paths**, introduced by Richard Feynman.
* It explains the unusual behaviors of quantum systems such as **interference and superposition**.
* The **Sum-Over-Paths** principle states that:
  + A quantum system does not move through a single path.
  + Instead, it considers **every possible path** from the starting state to the ending state.
  + Each path gives a contribution, called an **amplitude**.
  + The final result depends on the **sum of all amplitudes** from different paths.

### Example 1: Double Slit Experiment

* A photon can pass through **slit A** or **slit B**.
* Instead of choosing only one, it takes **both paths at once**.
* The two contributions combine and create an **interference pattern** on the screen.
* This interference comes from the **sum of paths**.