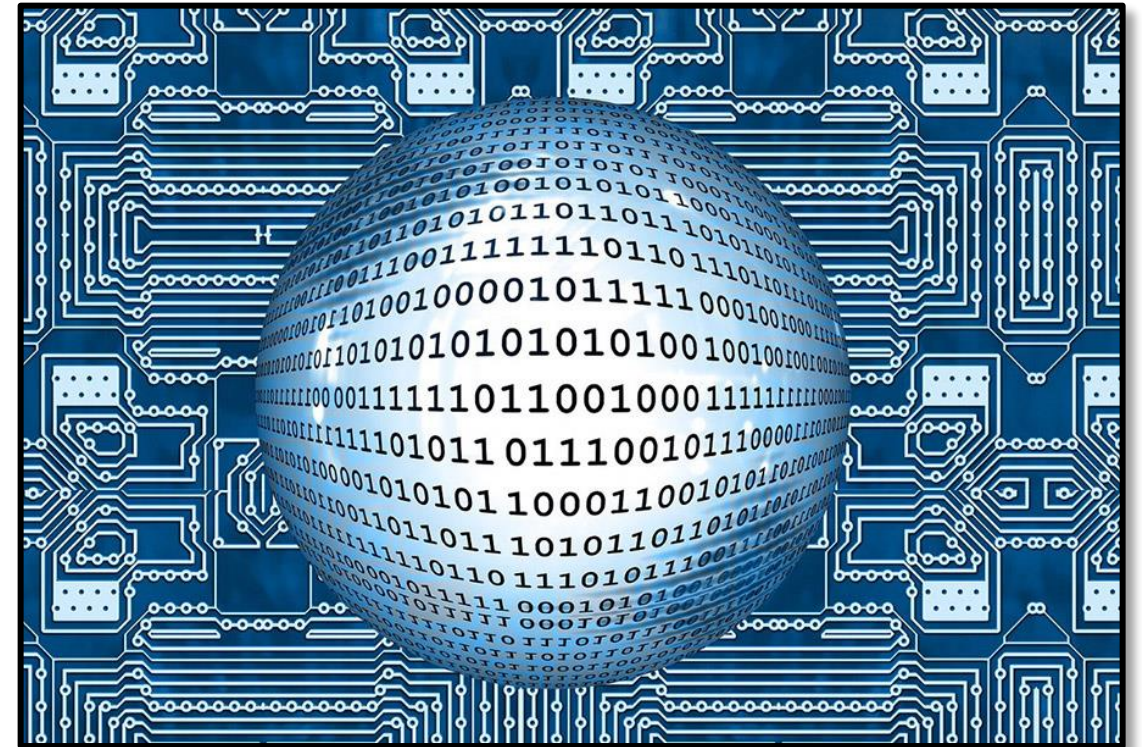


Traditional Bits

Before we can study quantum computing, it is essential to understand how traditional computers store and transform data. Current computers utilize transistors, various flip-flops, and latches to store data. The smallest memory component of a traditional computer is called a bit, which can either be 0 or 1. Multiple bits can represent various forms of information, from text to images and videos. With n bits, we can represent 2^n different combinations. It is important to note that at any given time, the n bits represent only one number, and all 2^n combinations cannot be represented simultaneously.

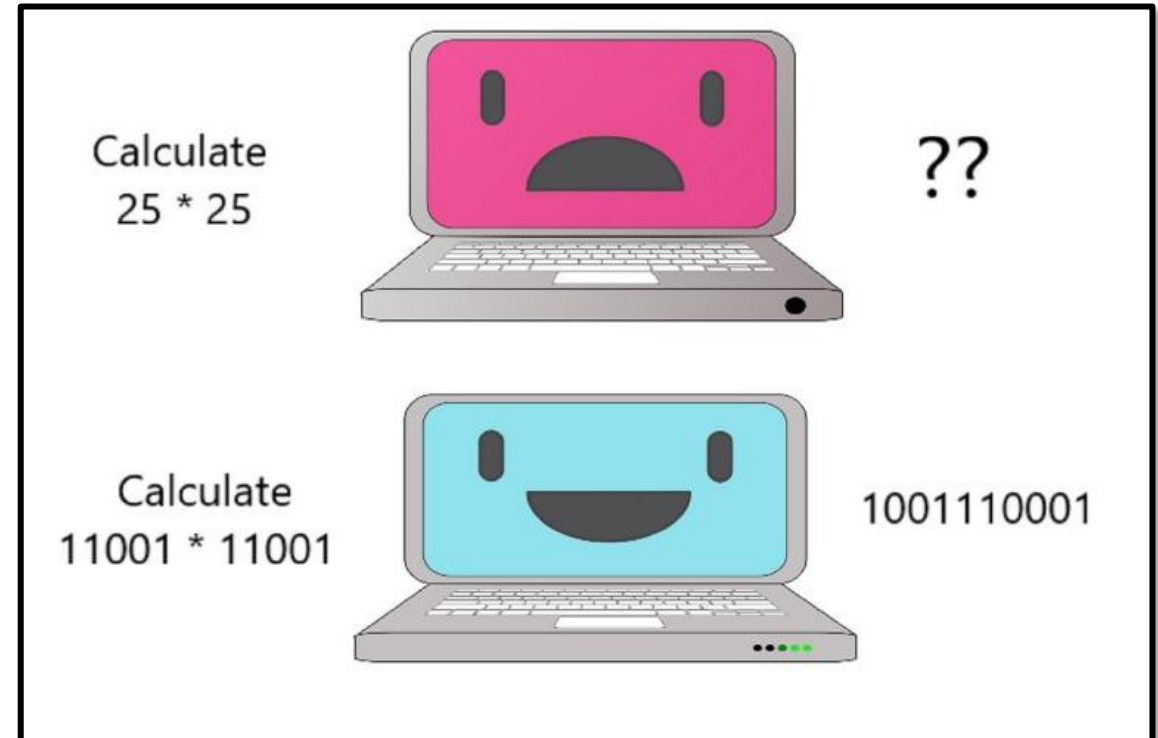


Traditional Bits

We can convert different numbers by calculating their binary representation. Instead of representing numbers in base 10, we will represent them in base 2.

To calculate this representation, we divide the number by 2 and record the remainders from left to right until we can no longer divide the result.

Additionally, we can convert text into binary form using the ASCII table. In this table, each character is represented by an 8-bit binary number.

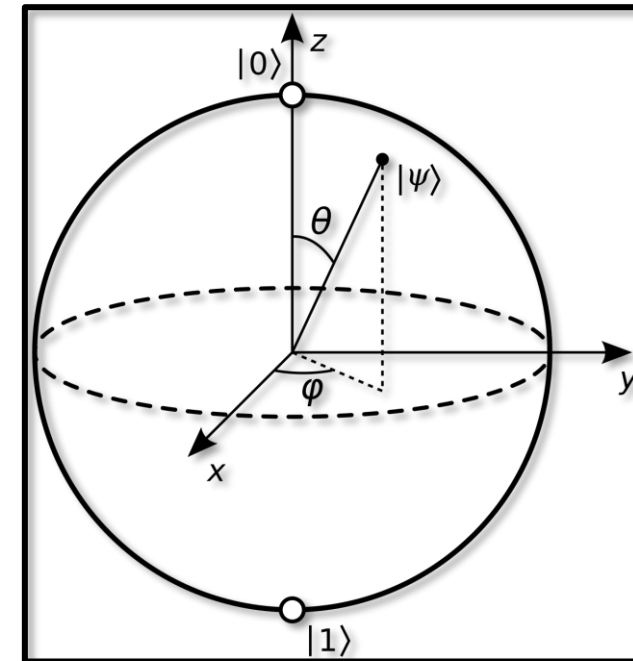


Source: www.medium.com

Quantum Qubit

A qubit is the fundamental difference between traditional computers and quantum computers. In quantum computing, a qubit, or quantum bit, is the basic unit of quantum information.

Qubits can exist in a 0 or 1 quantum state, or they can be in a superposition of the 0 and 1 states. With n qubits, we can represent 2^n possible states, and the qubits can be in all states simultaneously. This is the key concept behind quantum computers. We will explore what this means in the following slides.

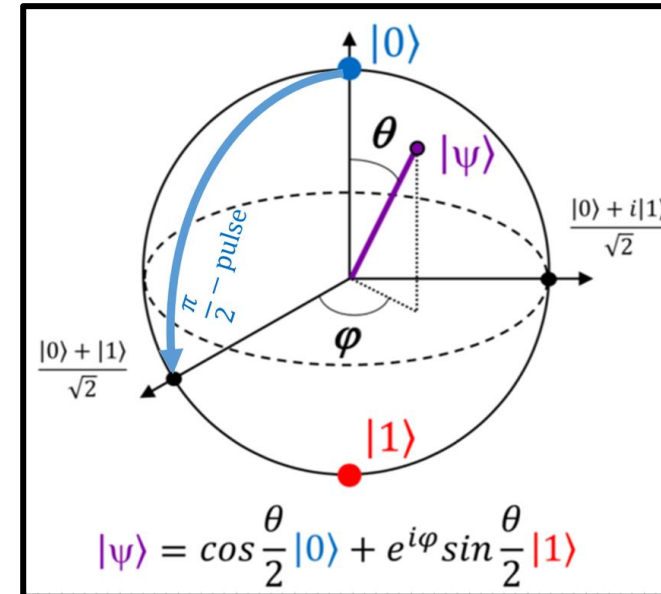


Bra-Ket notation

We denote a qubit's value by its position on the Bloch sphere using two variables. We use the Dirac bra-ket notation to describe the value of a qubit as a vector.

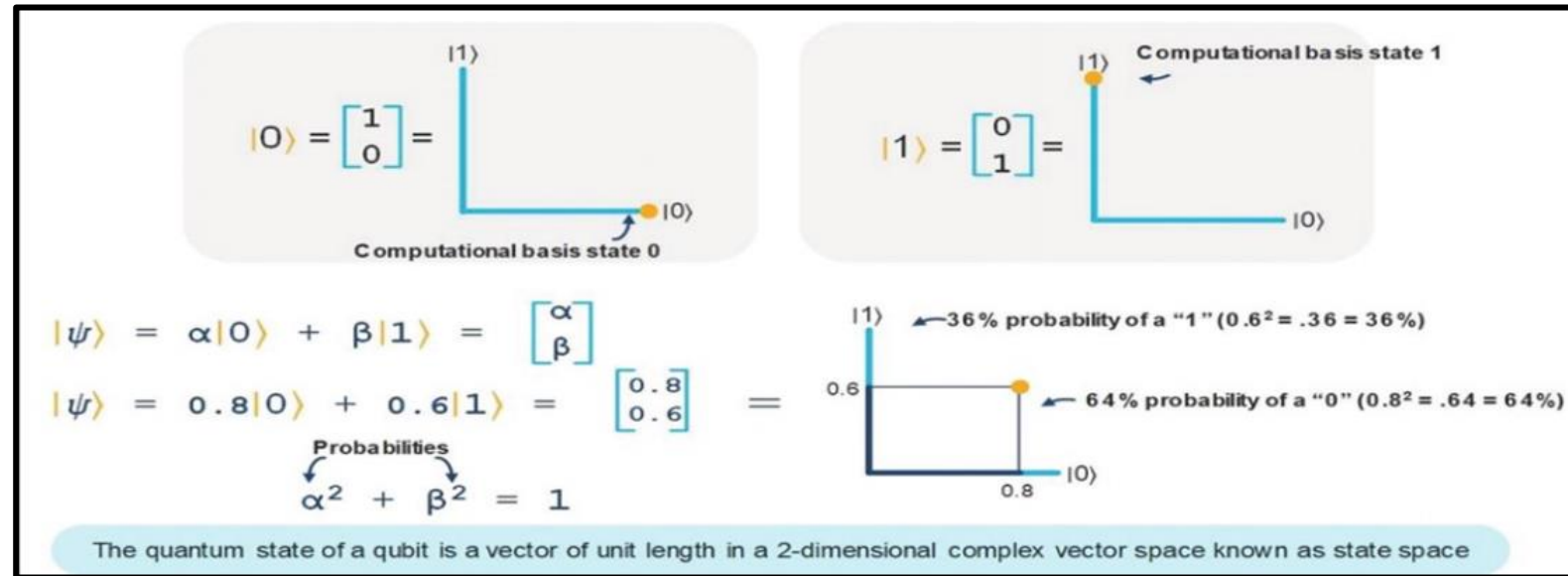
When the qubit is in the upper pole of the Bloch sphere, it is in the $|0\rangle$ state, which represents a value of 0. This means that the probability of measuring the qubit's value as 0 is 1. Conversely, if the qubit is in the lower pole of the Bloch sphere, it is in the $|1\rangle$ state, which represents a value of 1. In this case, the probability of measuring the qubit's value as 1 is 1.

We can represent any qubit's state anywhere on the Bloch sphere as a ket vector, which is the weighted sum of the $|1\rangle$ and $|0\rangle$ ket states. The weights are determined by the values of θ and ϕ , which define the qubit's position on the sphere.



Traditional Bits

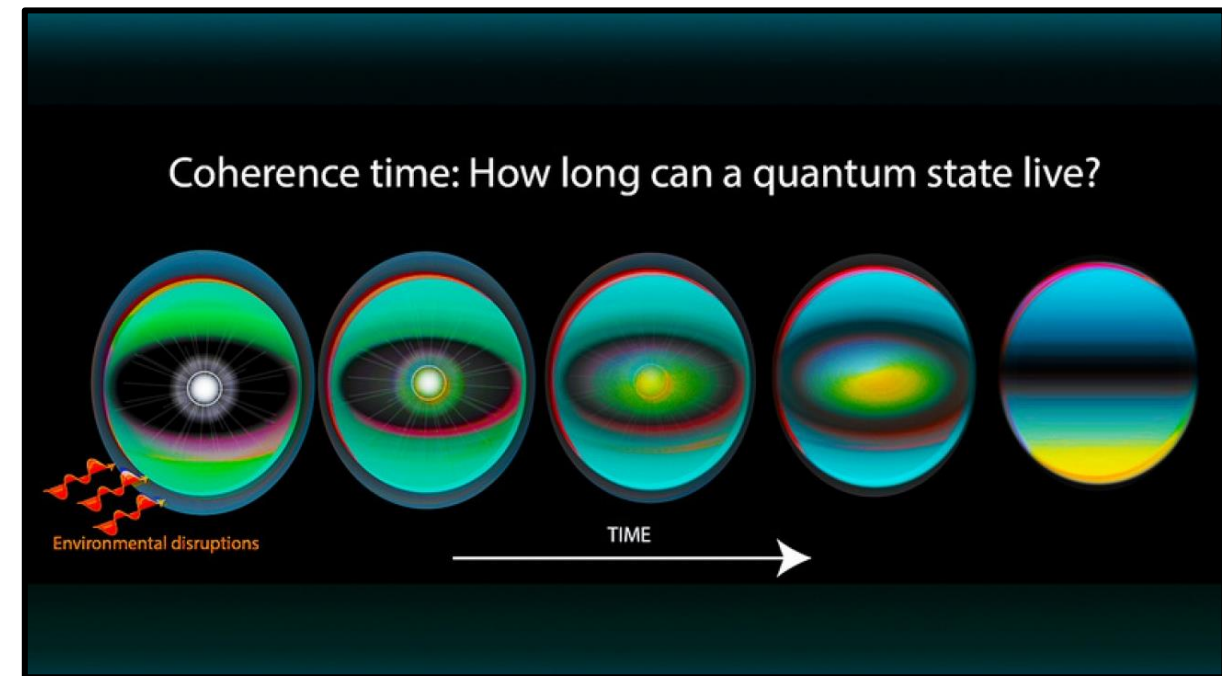
We can calculate the probabilities of the qubit's measurement results based on the weights of the $|0\rangle$ and $|1\rangle$ ket states. Specifically, if we take the square root of the weight associated with $|0\rangle$, it gives us the probability that the qubit's value is 0 after measurement. Similarly, we can calculate the probability of the qubit being measured as 1 by taking the square root of the weight associated with $|1\rangle$.



Santanu Ganguly, Quantum Machine Learning: An Applied Approach

Quantum decoherent

Keeping a qubit in its quantum state is a significant challenge. Quantum particles interact with their surroundings, which can cause them to return to their classical nature. This process is known as decoherence. The duration that a quantum state remains "alive" is referred to as coherence time. Decoherence poses a major obstacle in the development of stable and reliable quantum computers, as it limits the amount of time qubits can maintain their quantum properties before measurement.



Source: www.nist.gov

Classical information manipulation

In order to perform various calculations and transformations on classical information, we need gates to manipulate the values of the bits that store the data.

Many classical gates have been introduced, and complex structures such as adders, multipliers, and dividers have been designed by combining these gates.

Some of the most popular classical gates are the NOT gate, AND gate, OR gate, and the XOR gate. The operation of each gate can be represented by a truth table.

YES

INPUT		OUTPUT
A		
0		0
1		1

NOT

INPUT		OUTPUT
A		
0		1
1		0

AND

INPUT		OUTPUT
A	B	
0	0	0
1	0	0
0	1	0
1	1	1

OR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	1

XOR

INPUT		OUTPUT
A	B	
0	0	0
1	0	1
0	1	1
1	1	0

NAND

INPUT		OUTPUT
A	B	
0	0	1
1	0	1
0	1	1
1	1	0

NOR

INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	0

XNOR

INPUT		OUTPUT
A	B	
0	0	1
1	0	0
0	1	0
1	1	1

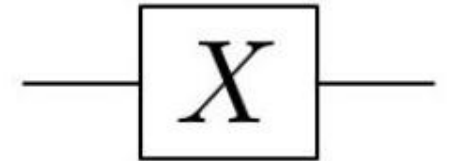
Pauli X gate

Just like classical computing, we need gates to manipulate the quantum data we have. These gates will be applied to qubits and change their positions on the Bloch sphere. The first gate we will study is the Pauli X gate. This gate is very similar to the classical NOT gate. It changes $|1\rangle$ to $|0\rangle$ and $|0\rangle$ to $|1\rangle$. This gate is often called the quantum NOT gate. The Pauli X gate can also be represented by the Pauli-X matrix.

Matrix representation

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

Circuit representation



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



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

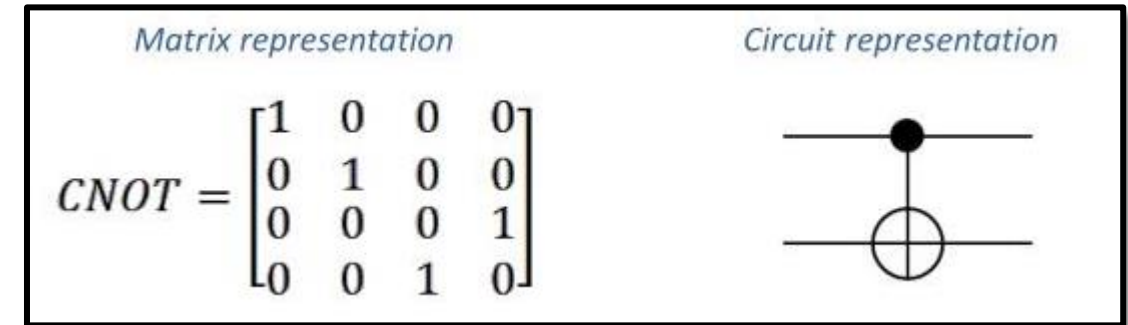
$$|1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$



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

CNOT gate

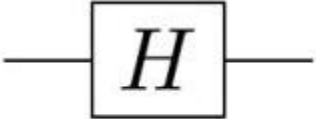
Another widely used gate in quantum computing is the controlled-NOT (CNOT) gate. This gate applies to two qubits: a control qubit and a target qubit. The value of the control qubit determines if the target qubit should be inverted or not. If the control qubit is in the $|1\rangle$ state, the CNOT gate applies the NOT operation to the target qubit, flipping its state from $|0\rangle$ to $|1\rangle$ or vice versa. If the control qubit is in the $|0\rangle$ state, the target qubit remains unchanged. The CNOT gate is very useful in quantum machine learning because it introduces non-linearity into quantum circuits. Non-linearity is crucial for fitting complex datasets, as it allows the circuit to learn and represent non-linear relationships between the input and output data.



$ 00\rangle$	\rightarrow	$ 00\rangle$
$ 01\rangle$	\rightarrow	$ 01\rangle$
$ 10\rangle$	\rightarrow	$ 11\rangle$
$ 11\rangle$	\rightarrow	$ 10\rangle$

Hadamard gate

One of the most important gates in quantum computing is the Hadamard gate. This gate is used to put a qubit into a superposition state, which is a key concept behind quantum computing. When we apply the Hadamard gate to $|0\rangle$ or $|1\rangle$, the qubit enters a superposition state where the probabilities of being measured as 0 or 1 are both 0.5. This means that after we apply Hadamard gate on $|0\rangle$ the qubit is now in a superposition of $|0\rangle$ and $|1\rangle$, with equal probabilities of being measured as either state.

Unitary matrix	Circuit representation
$H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	

$$\begin{aligned} H|0\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \\ H|1\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle \end{aligned}$$

Acting on pure states...

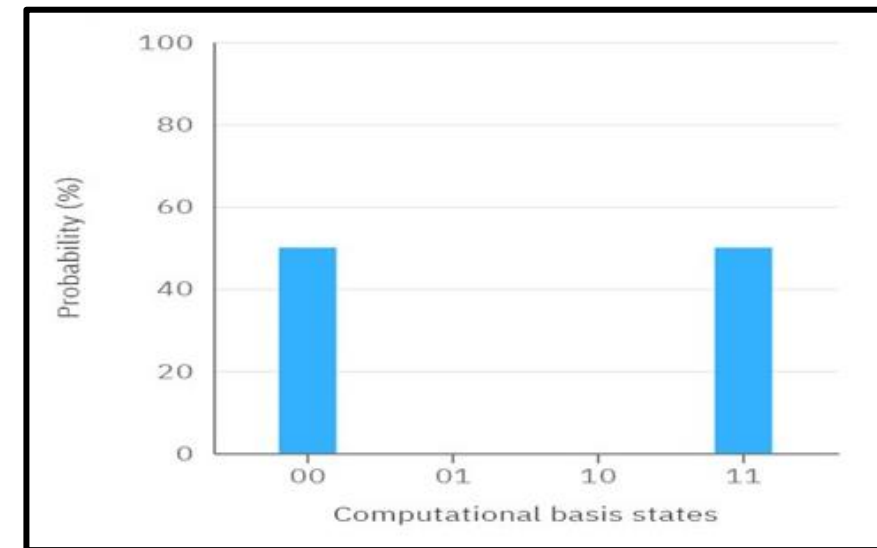
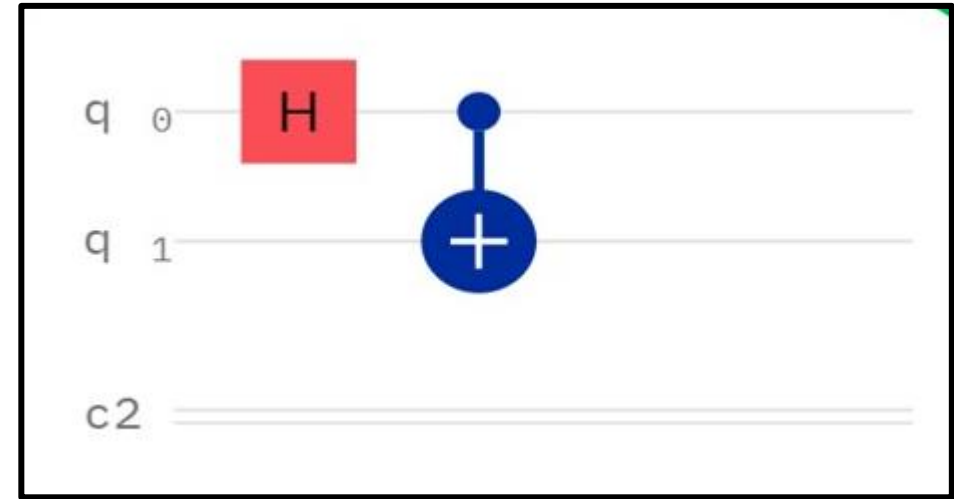
...gives a balanced superposition...

Quantum Circuit

There are many more quantum gates, and we will cover some of them in the next presentation. Others are out of the scope of this course.

Now that we have studied some of the most important quantum gates, we can design our first quantum circuit. In this circuit, we will have two qubits. We will apply the Hadamard gate to the first qubit, putting it into a superposition state. Then, we will add a CNOT gate, with the first qubit as the control qubit and the second qubit as the target qubit.

To get the output of the circuit, we will measure both qubits many times and observe the probabilities of each of the four possible states: $|00\rangle$, $|01\rangle$, $|10\rangle$, and $|11\rangle$. In this example, we can see that 50% of the time, both qubits are measured as 0, and the other 50% of the time, both qubits are measured as 1.



Summary

In this presentation, we explored the fundamental differences between traditional computing and quantum computing. We delved into the basics of quantum computing, including the concept of the qubit and various quantum gates.

We learned how to represent a qubit using the Dirac bra-ket notation and how its state can be described by its position on the Bloch sphere. We studied several key quantum gates, such as the Pauli X gate, the controlled-NOT (CNOT) gate, and the Hadamard gate, and how they manipulate the state of qubits.

To solidify our understanding, we designed a simple quantum circuit consisting of a Hadamard gate and a CNOT gate. By measuring the output of this circuit, we observed the probabilities of the possible qubit states, demonstrating the principles of quantum superposition and entanglement. In the next presentations, we will explore how quantum neural networks can be represented using these quantum gates and how we can train these models to solve complex problems. This will provide a deeper understanding of the potential applications of quantum computing in the field of machine learning.