Lab 1:

quantum circuits, multiqubit statevector manipulation and evaluation, the Quantum Information Qiskit package and visualization tools

Week 4 : 19/10/2020

TA: Hisham Ashraf Amer

Email: s-hisham.amer@zewailcity.edu.eg

Overview of Lab 1:

1. Build our first quantum circuits using Qiskit.

2. Introduce the Quantum information Module.

3. Building Multiqubit Operators to manipulate statevectors using the module.

4. Build a Bell state and visualize its measurement and elaborate on maximally entangled states.

5. Introduce the GHZ state, and build a quantum circuit for a 4-Qubit GHZ state and run it on IBM's quantum Computer Simulator.

6. Assess the Density matrices of GHZ states on Qiskit from an actual run on an IBM quantum computer, so we will see the effect of noise and decoherence on the density matrix of states resulting from real circuits.

7. Very quickly discuss the concept behind one very promising application of quantum computers, Variational Quantum Eigen-solvers, which we will build in a later lab session.

NB:
Remember to keep the Notebook with the code open while u study since not all the code is included here, I only post a few snippets of the code here

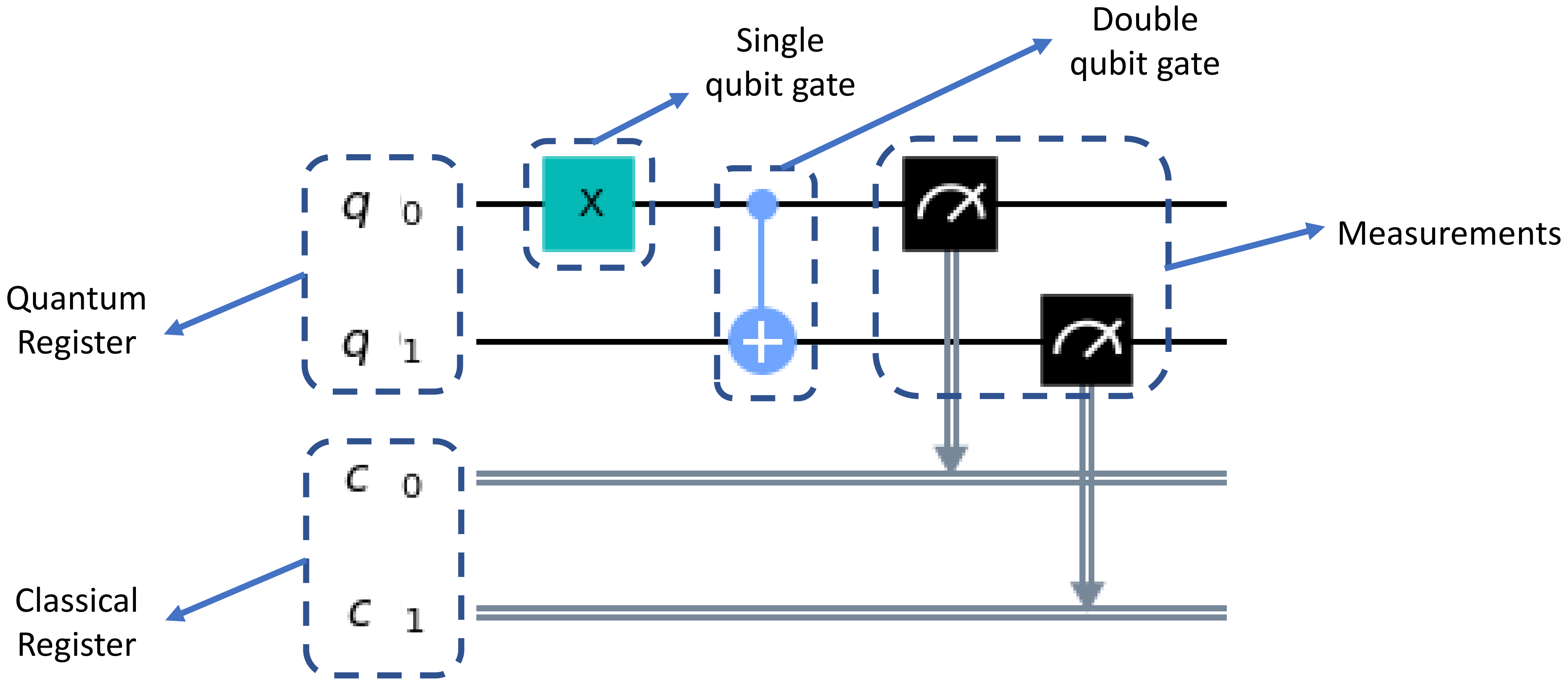
The nature of quantum mechanics and Quantum Hardware limit the type of procedures we can run. We will be mostly discussing the most popular kind, the quantum circuit model.

That's why we have an entire field dedicated to finding ways in which we can translate quantum circuit results into meaningful output
Quantum Algorithms

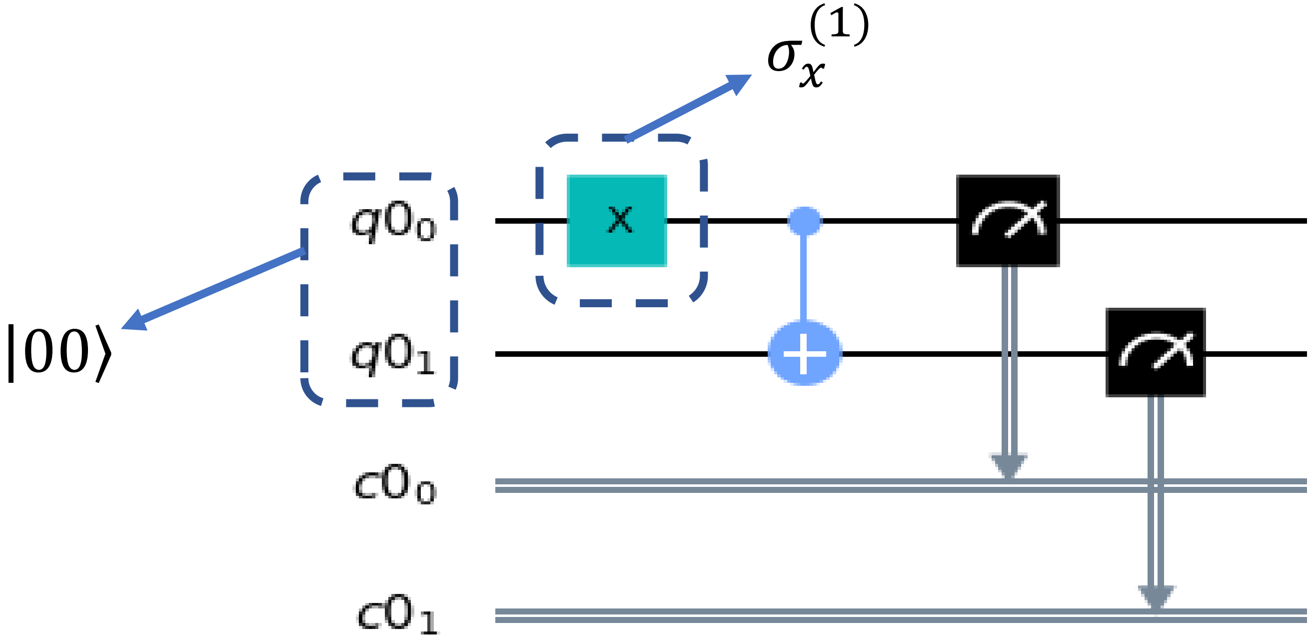
Every function we want to achieve needs its own algorithm and circuit design.

Quantum computers, based on the circuit model, are NOT universal yet.

Quantum circuit constituents



Write down the mathematical equivalent of this circuit



$$\mathrm{CNOT}\left(\sigma_x^{(1)} \otimes \mathbb{I}_{2\times 2}^{(2)}\right)|00\rangle$$

2 minute task :
derive the CNOT gate's matrix form

The CNOT gate flips the TARGET qubit iff the CONTROL qubit is $|1\rangle$

2 minute task: HINT:

CNOT$|00\rangle$ = $|00\rangle$

CNOT$|01\rangle$ = $|01\rangle$

CNOT$|10\rangle$ = $|11\rangle$

CNOT$|11\rangle$ = $|10\rangle$

Input basis

output basis

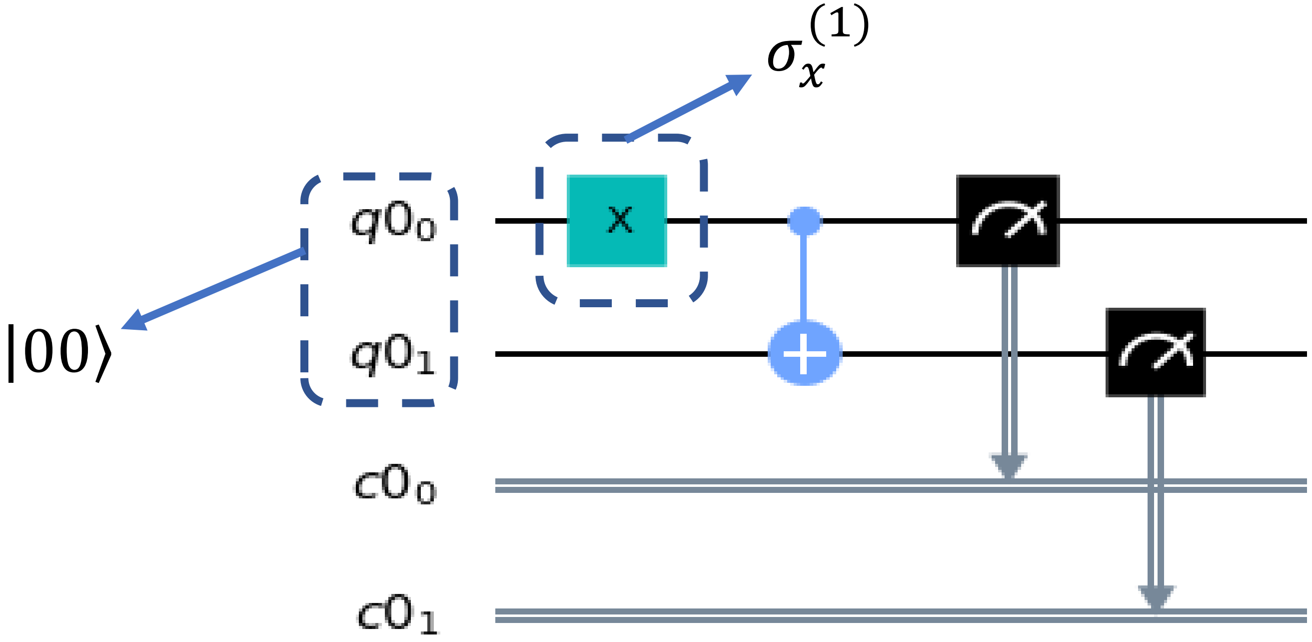
$$\begin{array}{cc} & \begin{array}{cccc} \quad|00\rangle\quad & \quad|01\rangle\quad & \quad|10\rangle\quad & \quad|11\rangle\quad \end{array} \\ \begin{array}{c} |00\rangle \\ |01\rangle \\ |10\rangle \\ |11\rangle \end{array} & \begin{bmatrix} \boxed{\langle 00|CNOT|00\rangle} & \langle 00|CNOT|01\rangle & \langle 00|CNOT|10\rangle & \langle 00|CNOT|11\rangle \\ \langle 01|CNOT|00\rangle & \boxed{\langle 01|CNOT|01\rangle} & \langle 01|CNOT|10\rangle & \langle 01|CNOT|11\rangle \\ \langle 10|CNOT|00\rangle & \langle 10|CNOT|01\rangle & \langle 10|CNOT|10\rangle & \boxed{\langle 10|CNOT|11\rangle} \\ \langle 11|CNOT|00\rangle & \langle 11|CNOT|01\rangle & \boxed{\langle 11|CNOT|10\rangle} & \langle 11|CNOT|11\rangle \end{bmatrix} \end{array} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

WHAT IF :

$$\mathrm{CNOT}|00\rangle = |00\rangle$$
$$\mathrm{CNOT}|01\rangle = |11\rangle$$
$$\mathrm{CNOT}|10\rangle = |10\rangle$$
$$\mathrm{CNOT}|11\rangle = |01\rangle$$

CNOT in this basis would be $= \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$

Write down the mathematical equivalent of this circuit



$$\text{CNOT}\left(\sigma_x^{(1)} \otimes \mathbb{I}_{2\times 2}^{(2)}\right)|00\rangle$$

Write down the mathematical equivalent of this circuit

$$\mathrm{CNOT}\left(\sigma_x^{(1)} \otimes \mathbb{I}_{2\times2}^{(2)}\right)|00\rangle = \mathrm{CNOT}\,|10\rangle = |11\rangle$$

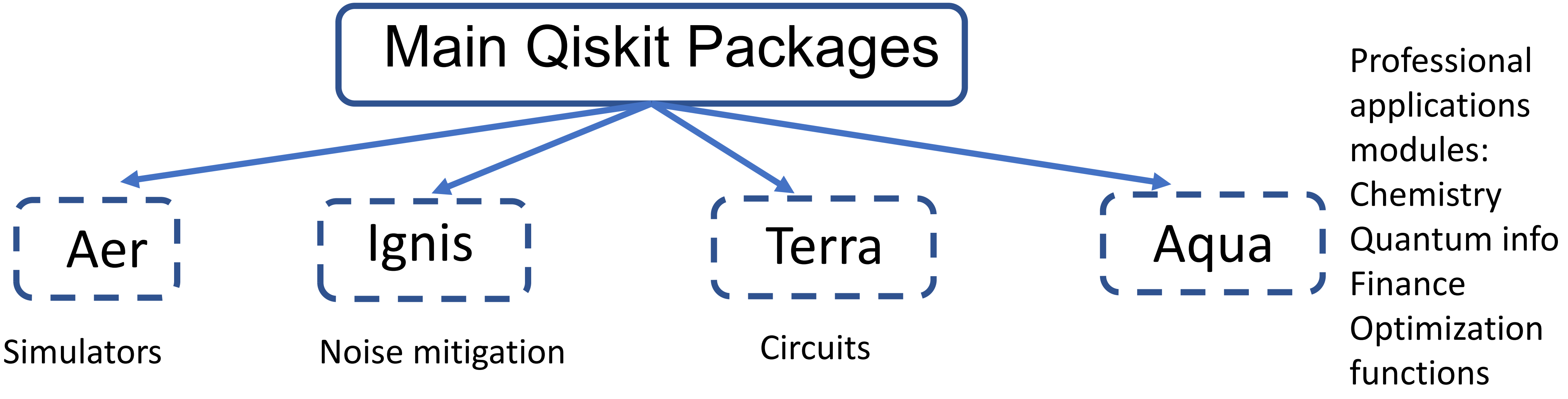
$$\begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatrix}0\begin{pmatrix}1&0\\0&1\end{pmatrix} & 1\begin{pmatrix}1&0\\0&1\end{pmatrix}\\ 1\begin{pmatrix}1&0\\0&1\end{pmatrix} & 0\begin{pmatrix}1&0\\0&1\end{pmatrix}\end{pmatrix}\begin{pmatrix}1\\0\\0\\0\end{pmatrix} = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}\begin{pmatr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

We will be using

The quantum circuit model

Through

Qiskit
(built on Python) to run IBM Q quantum processors



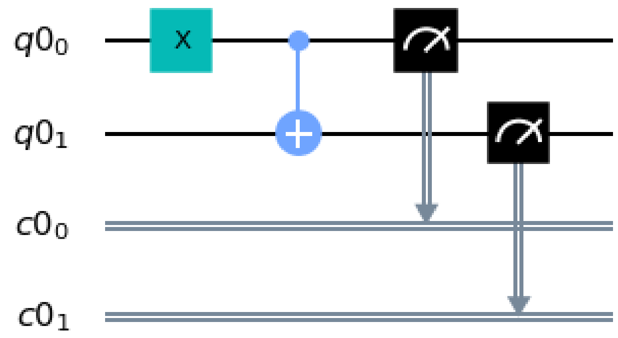
$$\text{CNOT}\left(\sigma_x^{(1)} \otimes \mathbb{I}_{2\times 2}^{(2)}\right)|00\rangle = |11\rangle \longrightarrow$$

```
print(statevector)
```

```
[0.+0.j 0.+0.j 0.+0.j 1.+0.j]
```

```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit

'''initializing the circuit with 2 qubits in the quantum register and 2 in
the classical register and their quantum circuit'''

qr = QuantumRegister(2, 'q')
cr = ClassicalRegister(2, 'c')
circ = QuantumCircuit(qr, cr)

## add gates to the quantum circuit

circ.x(qr[0])
circ.cx(qr[0], qr[1])
circ.measure(qr[0], cr[0])
circ.measure(qr[1], cr[1])
```

```
## printing out the circuit in the matplotlib format

circ.draw(output='mpl',cregbundle=False)
```



5 min task :

Find a suitable circuit to generate the **Singlet state**:

$$\frac{1}{\sqrt{2}}(|01\rangle - |10\rangle)$$

UP TO A PHASE

In terms of Qiskit it would be :

$$\frac{1}{\sqrt{2}}(|10\rangle - |01\rangle)$$

Qasm Simulator and Visualization tools (Bloch sphere and Histograms)

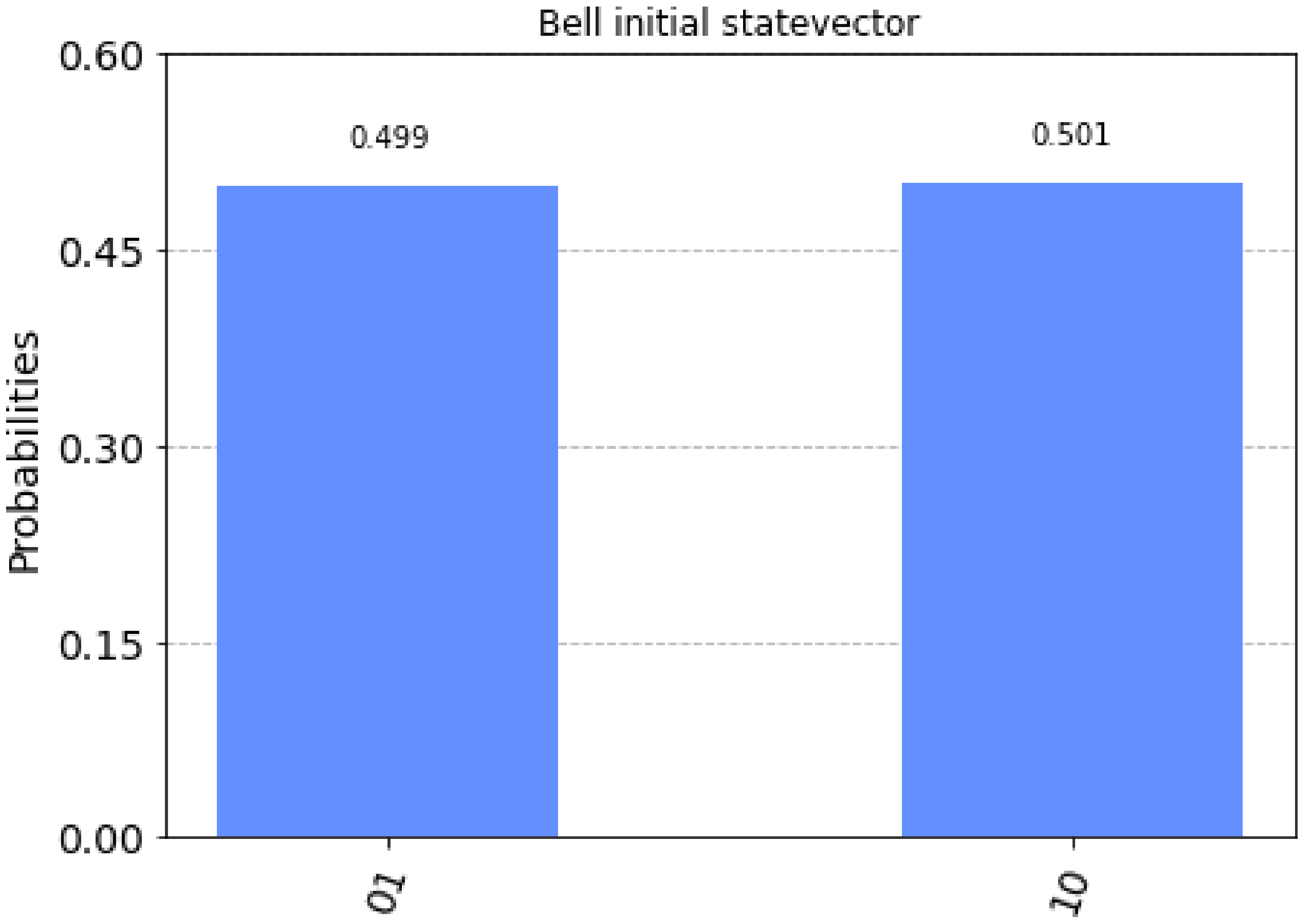
```
## specifying the simulator, we will use the Qasm simulator form the Aer Qiskit package

sim = Aer.get_backend('qasm_simulator')

#############################################
```

```
## run the circuit 'circ' on the Qasm simulator 'sim' storing the results in the "rslt" object
rslt = execute(circ, backend = sim).result()
```

```
#for informative visualization of results we can use qiskit visualization tools but first need
# to import the required tool:

from qiskit.tools.visualization import plot_histogram
```

```
## plotting a histogram of the measurements made by the circuit

plot_histogram(rslt.get_counts(circ),title="Bell initial statevector")
```

Qasm Simulator and Visualization tools (Bloch sphere and Histograms)



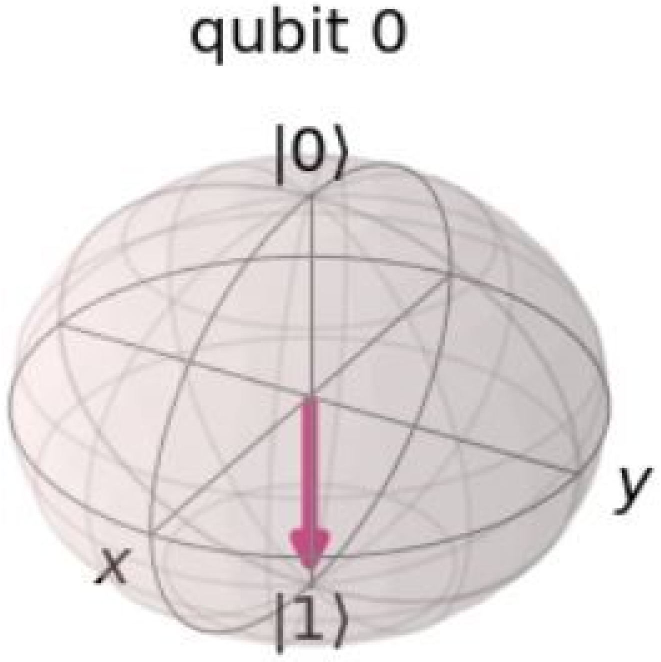
Bloch sphere of a given state

```
## import the plot bloch vector function from Qiksit

from qiskit.tools.visualization import plot_bloch_multivector

# setup a circuit to generate a single qubit state

circ = QuantumCircuit(1,1)
circ.x(0)
S_simulator = Aer.get_backend('statevector_simulator')
rslt = execute(circ, backend = S_simulator).result()
statevector = rslt.get_statevector()

print(statevector)
```

```
[0.+0.j 1.+0.j]
```

Bloch sphere of a general state

```
## given a statevector we can plot it on a bloch sphere
plot_bloch_multivector(statevector)
```

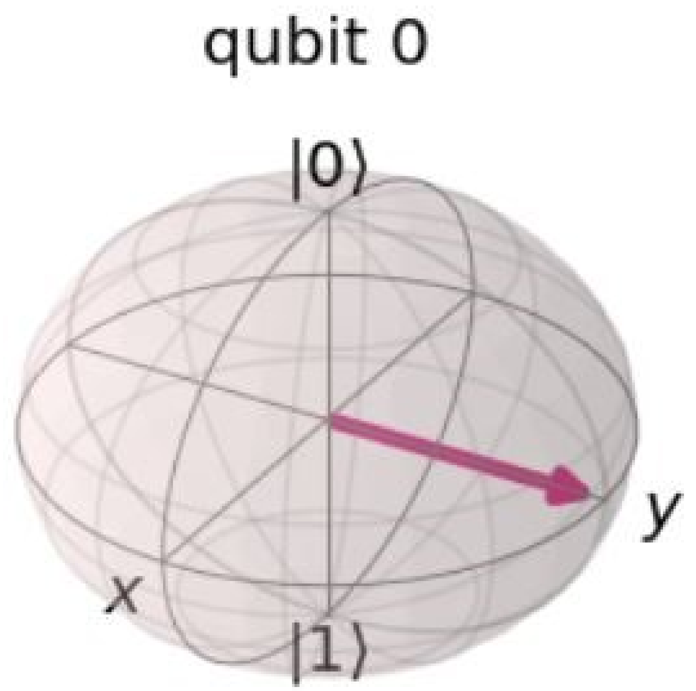


2 Minute Task:

Generate the Bloch sphere of the statevector $|r\rangle$

Visualizing the Bloch sphere of a general state

```
my_statevector = np.array([0.707, 0.0+0.707j])
plot_bloch_multivector(my_statevector)
```



What about General circuits with many qubit gates ?

In 1994, David P. DiVincenzo

"proved that quantum gates operating on just two bits at a time are sufficient to construct a general quantum circuit"

Paper : https://arxiv.org/abs/cond-mat/9407022v1