

# Qubit Measurement Visualization in Default, Single and Multiple-Qubit System using Qiskit

Richa Dixit Pandey<sup>1\*</sup>, Rabins Porwal<sup>2</sup>

<sup>1</sup>Department of Computer Application, School of Engineering and Technology (UIET), Chhatrapati Shahu Ji Maharaj University(CSJMU), India , richadpandey@csjmu.ac.in or 0009-0002-0651-2392

<sup>2</sup>Department of Computer Application, School of Engineering and Technology (UIET), Chhatrapati Shahu Ji Maharaj University(CSJMU), India, rabins@csjmu.ac.in or 0000-0003-4646-4962

**Abstract.** Quantum Computing is solving problems across various domain like Quantum state representation, measurement visualization, and performance characterization together describe how quantum information is encoded, observed, interpreted, and evaluated within a quantum computing system. This research explores how qubit measurements collapse quantum states and how visualization tools help interpret these behaviors.

**Keywords :** Qiskit, Visualization, Measurement, Qubit, tomography.

## 1 Introduction

Qubit measurement visualization area generally refers to the quantum state representation and measurement, visualization and interpretability, characterization and performance analysis. Quantum computing anticipate on qubits, which unlike classical bits can exist in superposition and entangled states. Understanding and visualization of qubit states and their measurement outcomes is critical to bridging the gap between abstract quantum formalism and intuitive understanding and crucial for both research and teaching. The Bloch sphere serves as the most common tool for single qubit visualization (states) providing an intuitive geometric representation of superposition and phase, while measurement histograms are used to represent multi-qubit probability distributions, for multi-qubit systems, visualization becomes more complex due to exponential state space growth. This paper presents a detailed methodology for qubit visualization using Qiskit simulations, covering basis states, superpositions, and entangled states such as Bell and GHZ. [4]

In this work, a methodical framework for using Qiskit to visualize qubit measurement results across default, single-qubit, and multi-qubit systems is presented. It makes a contribution by:

- illustrating how measurement converts quantum states into classical information.
- Offering straightforward visualization techniques, such as statevectors, Bloch spheres, histograms, and multi-qubit outcome distributions.
- Contrasting measurement behavior with increasing system sizes to show the emergence of complexity, entanglement, and correlation patterns. When taken as a whole, these contributions promote better comprehension of quantum measurement dynamics and their useful application in Qiskit.

This paper is organized as follows:\newline

Section 2 covers keypoints of all algorithms (2.1-2.5). Section 3 covers related literature review. Section 4 discuss methodology under 4.1-4.3. In section 5 keeps result and discussion, 5.1 and 5.2 shows the figure of Bloch sphere and histogram, in 5.3 shows the "Table 1" with expected outcomes of qubit visualization (default, single, multi

---

\* Corresponding author: [author@email.org](mailto:author@email.org)

qubits), "Table 2" shows frequency measurement to create a histogram and in figure 4 shows the circuit to create the Bell state, in figure 5 to generate the GHZ state, in figure 6 add one more qubit, one more CX and in figure 7 add extra qubit and more CX. In section 6, summarized the main findings. In section 7 gives the acknowledgments.

## 2 Key Points

This work can be indicating as follows Please note sections 2.1-2.5 below for more information. The Qiskit flow offers a structured approach to the design and analysis of quantum circuits through the initialization of qubits (commonly in the  $|0\rangle$  state), the application of quantum gates to implement superposition and entanglement, measurement, the execution of the circuit multiple times (shots) to provide statistical results, and the visualization of the outcome using tools such as Bloch spheres, statevectors, and histograms. A single-qubit circuit flow showcases superposition through the application of a Hadamard gate to the  $|0\rangle$  state to produce  $|+\rangle = 1/\sqrt{2}(|0\rangle + |1\rangle)$ , which is represented on the X-axis of the Bloch sphere with a 50/50 split in measurement probability. Building on this, a two-qubit circuit flow generates a Bell state through the application of a Hadamard gate followed by a CNOT gate, resulting in the entangled state  $1/\sqrt{2}(|00\rangle + |11\rangle)$  with maximally entangled and correlated results, while a three-qubit circuit flow generates a GHZ state  $1/\sqrt{2}(|000\rangle + |111\rangle)$ , which showcases multipartite entanglement where all qubits are correlated.

### 2.1 Workflow and Flowchart of the Qiskit:

**qubits initialized → quantum gates applied → measure → repeat over shots → visualize results (Bloch sphere / histogram / statevector)**

Explains how flowchart summarizes the Qiskit workflow, showing how qubits are initialized, transformed by quantum gates, measured to obtain classical outcomes, repeated over multiple shots to gather probabilities, and finally visualized using Bloch spheres, histograms, or statevector plots depending on the system.

### 2.2 Flow: $|0\rangle \rightarrow$ Hadamard $\rightarrow |+\rangle \rightarrow$ simulate with Qiskit $\rightarrow$ plot statevector on Bloch sphere

Explains the Algorithms 1 that all steps briefly show how a single-qubit state is visualized on the Bloch sphere: a one-qubit circuit is initialized in the  $|0\rangle$  state, a Hadamard gate is applied to create the  $|+\rangle$  superposition, the resulting statevector is generated using Qiskit's simulator, and finally this state is plotted on the Bloch sphere to display its position in quantum state space.

**2.1 Flow:  $|0\rangle \rightarrow$  Hadamard  $\rightarrow$  superposition  $\rightarrow$  measure over many shots  $\rightarrow$  plot histogram of 0 "and" 1 approx 50/50**

Explains the Algorithms 2 that the steps outline how Qiskit generates a histogram of single- qubit measurement outcomes: the circuit is initialized in  $|0\rangle$ , a Hadamard gate creates an equal superposition, the qubit is measured and executed over many shots, and the collected results are plotted as a histogram showing equal probabilities for 0 and 1.

**2.2 Flow:  $|00\rangle \rightarrow$  Hadamard on qubit-1  $\rightarrow$  CNOT  $\rightarrow$  entangled Bell state, then (optional) statevector/measurement  $\rightarrow$  correlated outcomes 00 "and" 11**

Explains the Algorithms 3 that steps show how a Bell state is created: two qubits start in  $|00\rangle$ , a Hadamard gate places the first qubit in superposition, a CNOT gate entangles both qubits, and the resulting Bell state can be confirmed through the statevector or optional measurement, which yields correlated outcomes like 00 and 11.

**2.3 Flow:  $|000\rangle \rightarrow$  Hadamard on qubit-1  $\rightarrow$  CNOT, CNOT  $\rightarrow$  GHZ state  $\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \rightarrow$  (optional) measure  $\rightarrow$  correlated outcomes 000 "and" 111**

Explains the Algorithms 4 that steps show how a GHZ state is created: the three qubits start in  $|000\rangle$ , a Hadamard gate puts the first qubit into superposition, and two CNOT gates entangle all three qubits, forming the GHZ state  $\frac{1}{\sqrt{2}}(|000\rangle + |111\rangle)$ . Optional measurement then reveals perfectly correlated results such as 000 and 111, confirming full three-qubit entanglement.

### 3. Literature Review

Visualization of qubit measurement is an essential aspect of understanding quantum state evolution and measurement processes in quantum computing. The Bloch sphere representation has historically been one of the most fundamental tools for visualizing single-qubit states, allowing users to understand superposition and phase relations. For multi-qubit systems, the challenge increases due to entanglement and exponential state space growth, which has driven the development of computational visualization tools like Qiskit, IBM's open-source quantum computing SDK [5].

Qiskit provides integrated functions such as `plot_bloch_multivector()`, `plot_histogram()`, and `plot_state_city()` for representing statevectors and measurement distributions [7]. These tools help visualize both simulated and real hardware results, enabling researchers and learners to interpret probabilistic quantum outcomes effectively. The Aer simulator backend within Qiskit allows for measurement result sampling, which can be compared with ideal statevector visualizations to study noise effects [8].

Previous studies have emphasized the importance of visualization in education and debugging quantum algorithms. For instance, Nielsen and Chuang demonstrated that

conceptual understanding of quantum state measurement is enhanced through geometric representation on the Bloch sphere [1]. Similarly, interactive platforms such as IBM Quantum Experience leverage Qiskit's visualization modules to teach quantum measurement and circuit execution, significantly aiding comprehension of quantum principles [14].

For multi-qubit systems, visualization becomes complex due to correlations among qubits. Research by Preskill [6] and Gambetta et al [17] has shown that statistical visualization methods, such as histograms of measurement outcomes and density matrix plots, are more practical for higher-dimensional systems than Bloch spheres. These tools allow researchers to infer entanglement, coherence, and fidelity from observed data distributions.

Despite these advances, limitations exist. Bloch sphere visualization is restricted to single-qubit representations and cannot efficiently depict systems beyond two qubits. Furthermore, decoherence and quantum noise can distort measurement outcomes, complicating direct visualization of hardware states [9]. To overcome these, Qiskit includes noise modeling and error mitigation tools—like Zero Noise Extrapolation (ZNE) and Measurement Error Mitigation (MEM)—that help align simulation results with experimental observations [19]. In the future, visualization in quantum computing is expected to evolve toward interactive 3D interfaces, AI-assisted analysis, and tensor-network-based visualizations for multi-qubit correlations [13]. Such developments could make complex quantum behavior more intuitive and accessible for both researchers and students. This study demonstrates key quantum principles using Qiskit simulator, validates measurement visualization tools for default, single and multi-qubit systems, highlights current visualization challenges including decoherence and scalability, and calls for collaborative efforts to advance quantum education and research.

## 4. Methodology

It separates data generation (ensuring reproducible, controlled quantum experiments/simulations) from visualization & analysis (how you interpret, present, and reason about those results).

This allows for the possibility of flexibility: depending on goals, you may focus only on measurement statistics, or also on underlying quantumstate structure.

It reflects scientific workflow-design → execute → observe → analyze → conclude-but adapted for a quantum computation and visualization context.

Methodology divided into 3 parts-

- Implementation & Data Generation

Build quantum circuits with Qiskit: single-qubit and multi-qubit, apply the gates, and then add measurements by calling the `measure()` or `measure_all()` methods to collapse the qubit states into classical outcomes.

Run the circuits in a simulator (or real backend), for many shots, and collect the measurement results/counts (or statevectors/ density matrices if you want pre-measurement state).

Store the results - counts or state representations - in structured form: Python dict for counts, Qiskit Statevector/DensityMatrix for states.

- Visualization & Analysis

Qiskit has built-in visualization tools in `qiskit.visualization`, including for example: `plot_histogram()` that shows measurement outcome distributions (counts), and for pre-measurement states use e.g. `plot_state_qsphere()`, `plot_state_city()`, `plot_bloch_multivector()` etc.

Analyze and compare: for example, compare pre-measurement quantum states (superposition / entanglement) with post-measurement outcome distributions, study how measurement collapses states, look into correlations in multi-qubit measurements, etc.

Optional: For multi-qubit / complex states, when default plots are insufficient, extend with custom plots or more advanced visual / analytic methods.

- Experiments

Both single and multi-qubit regimes are covered, thereby giving breadth to your study. They combine state-based visualization before the measurement with measurement-based visualization after the measurement, so that you can explore collapse, entanglement, randomness, statistical effects.

They include quantitative analysis-probabilities, shot-count dependence, subsystem effects-which can yield publishable or reportable results even for small-scale experiments.

They are feasible on a classical simulator via Qiskit: no real quantum hardware is required, thus good for reproducibility and easy prototyping.

## 4.1 Background

In quantum mechanics, the state of a single qubit is described as a superposition of basis states. This is written as  $\alpha|0\rangle + \beta|1\rangle$ , with  $\alpha$  and  $\beta$  being complex numbers. When the qubit is measured in the computational basis, it collapses to  $|0\rangle$  with probability  $|\alpha|^2$ , or to  $|1\rangle$  with probability  $|\beta|^2$ . By repeating the experiment multiple times (shots), we obtain a probability distribution of outcomes. Multi-qubit states are written by combining single-qubit states using the tensor product. For example, the Bell state  $(|00\rangle + |11\rangle)/\sqrt{2}$  demonstrates quantum entanglement[15].

Qiskit's application in Qubit measurement visualization include simulating how qubits collapse under measurement, displaying single-qubit states on the Bloch sphere, and visualizing multi-qubit probability distributions and entanglement through histograms and statevector plots. These tools help analyze superposition, entanglement, and measurement behavior in both simple and complex quantum system.

In this research, Qiskit an open- source tool for quantum computing is used to model and show how qubit states behave and what results we get when measuring them. The process begins with qubit initialization in the state  $|0\rangle^{\wedge n}$ , followed by the application of quantum gates to prepare desired states. For single qubits, final states are represented on the Bloch sphere, while for multi-qubit systems, measurement histograms are used to capture probability distributions.

The algorithm (visualization process) is structured as follows :

1. Initialize qubits in  $|0\rangle^{\wedge n}$
2. Apply unitary gates, quantum gates (Hadamard, CNOT, etc.)
3. Compute final statevector
4. If  $n=1$ : map to Bloch sphere
5. If  $n \geq 2$ : run measurements and generate histogram
6. Export plots(Compare results with theoretical predictions)

Figure 1 shows the flowchart of the standard workflow for qubit measurement visualization in Qiskit.

## 4.2 Flowchart for Qubit Measurement Visualization

Start Initialize Qubit(s) Define quantum register (single or multiple qubits). Apply Quantum Gates (e.g., H, X, CNOT) to prepare state or entanglement. Quantum State Evolution System evolves to superposition/entangled state. Measurement Operation Apply measurement operator (collapse superposition  $\rightarrow$  basis states). Classical Register Storage Store measured results (0s and 1s). Execute Shots (Multiple Runs)

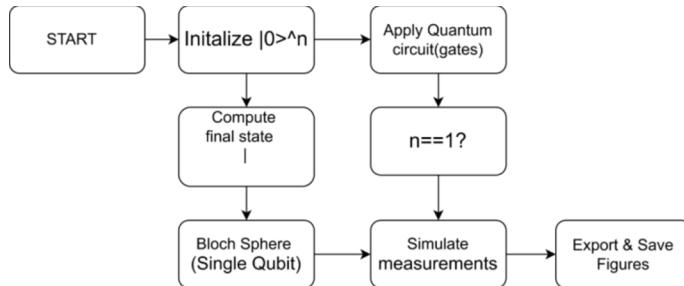


Fig. 1. Workflow for qubit measurement visualization.

Repeat circuit execution for probability distribution. Collect Results  
Count frequency of measurement outcomes. Visualization Histogram / Bar chart for probabilities. Bloch sphere representation (for single qubit). Statevector visualization (if no measurement yet). End

- (1) Start
- (2) Initialize Qubit(s)  
Define quantum register (single or multiple qubits).
- (3) Apply Quantum Gates  
(e.g., H, X, CNOT) to prepare state or entanglement.
- (4) Quantum State Evolution  
System evolves to superposition/entangled state.
- (5) Measurement Operation  
Apply measurement operator (collapse superposition  $\rightarrow$  basis states).
- (6) Classical Register Storage  
Store measured results (0s and 1s).
- (7) Execute Shots (Multiple Runs)  
Repeat circuit execution for probability distribution.
- (8) Collect Results  
Count frequency of measurement outcomes.
- (9) Visualization  
Histogram / Bar chart for probabilities.  
Bloch sphere representation (for single qubit).  
Statevector visualization (if no measurement yet).
- (10) End

## 4.3 Qubit's Visualizing & Measurement Quantum Computing Algorithms (Qiskit)

### Combined Algorithm: Quantum State Visualization and Entanglement Simulation

Input: Number of qubits ( n, like 1, 2, 3Q(s) )

Output: Visual representations (Bloch sphere, measurement histogram, and entangled state visualization) [11]

#### Step 1: Initialize the Quantum System

1. Create a quantum circuit with n qubits and n classical bits.
2. All qubits start in the computational basis state  $(|0\rangle^n)$ .

#### Step 2: Create Superposition

- If ( n=1): Apply a Hadamard (H) gate to the qubit to produce the superposition state  $(|+\rangle = 1/\sqrt{2})(|0\rangle + |1\rangle)$ .
- If(n=2) :Apply an H gate to the first qubit. This step prepares the first qubit in a superposition, forming the foundation for entanglement.

#### Step 3: Generate Entanglement (if applicable)

- If ( n = 2 ): Apply CNOT(control = qubit 0, target = qubit 1)→ Produces a Bell state:  $(|\Phi^+\rangle = 1/\sqrt{2})(|00\rangle + |11\rangle)$ .
- If ( n = 3 ): Apply CNOT(control = qubit 0, target = qubit 1).Apply CNOT(control = qubit 0, target = qubit 2)→ Produces a GHZ state:  $(|\text{GHZ}\rangle = 1/\sqrt{2})(|000\rangle + |111\rangle)$ .

#### Step 4: Visualization of Quantum State

- **Single Qubit ( n = 1):** Simulate the circuit to obtain the state vector. Plot on the Bloch sphere to visualize the qubit's position in quantum state [10] space.
- **Multiple Qubits ( n ≥ 2):** Use statevector simulation to extract the system's full quantum state. If entangled, visualize correlations or partial traces for each qubit using density matrix visualizations or Statevector plots in Qiskit[12].

#### Step 5: Measurement

1. Measure all qubits and store the results in classical bits.
2. Execute the circuit on a simulator (e.g., 1000 shots).
3. Collect measurement outcomes to analyze probabilities or correlations:

For a single qubit, histogram shows ~50% for each of  $|0\rangle$  and  $|1\rangle$ . For Bell or GHZ states, correlated outcomes (e.g., 00 & 11 for Bell, 000 & 111 for GHZ) dominate. [18]

#### Step 6: Visualize Results

- Plot measurement histograms to show probability distributions. Confirm quantum behavior: Equal probabilities (superposition), Correlated outcomes (entanglement)

## 5. Results and Discussion

Code snippets (Qiskit-style) reproduce the figures. Run them in a Jupyter notebook or Google Colab or Cirq [2]. This table summarizes the expected outcomes of qubit visualization experiments using Qiskit, covering default, single, and multiple- qubit systems.

Code snippets in Qiskit style run all the visualization figures if executed in Jupyter Notebook, Google Colab, or Cirq-supported environments.

These results confirm the expected behavior in default, single-qubit, and multi-qubit systems: state vectors and Bloch spheres agree with theoretical predictions, measurement histograms reflect the appropriate probability distributions, and multi-qubit circuits exhibit the appropriate entanglement signatures and correlated outcomes.

### 5.1 Experimental work

Measurements of quantum circuits on the Qiskit simulator demonstrate that quantum circuits with no qubits and no measurements produce no results. while qubits with initial values but no gates produce deterministic  $|0\rangle$  results. The application of gates introduces superposition and entanglement. Leading to probabilistic and correlated measurement outcomes. Measurement operations allow for the collapse of the state and the visualization of results, validating predictions. Related work shown in table 3.

### 5.2 Bloch sphere

The Bloch sphere helps us see qubit states as points on a sphere. The states  $|0\rangle$  and  $|1\rangle$  sit on the top and bottom, while  $|+\rangle$  and  $|-\rangle$  are on the sides. This makes it easier to understand how qubits can exist in different states and combinations. Figure 2 shows the Bloch vector for  $|0\rangle, |1\rangle, |+\rangle$  and  $|-\rangle$  using experiment work in Google Colab.

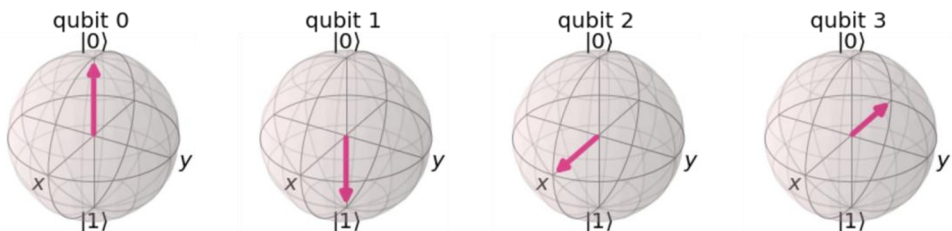


Fig. 2: Bloch sphere grid for  $|0\rangle, |1\rangle, |+\rangle$  and  $|-\rangle$ .

### 5.3 Qiskit: Histogram

Histogram is a preferred, intuitive and easy way to understand and visualize the results of a Quantum Circuit. Qiskit provides inbuilt function to create histogram visualizations. In this chapter of the Qiskit Tutorial, you will learn about how to create histogram visualization by making use of the visualization module of qiskit.

A histogram plots the frequencies of measurements of Qubits. This makes histogram an excellent choice for visualizing the results from QASM Simulator.

- Importing histogram visualization function. The `plot_histogram()` function of `qiskit.visualization` creates Histogram visualization. from `qiskit.visualization` import `plot_histogram()`.
- Plotting Histogram. The `plot_histogram()` method accepts data for plotting histogram in the form of a Python dictionary. The dictionary has the various measurement values as keys and their corresponding frequencies as values.

To creating a histogram is depend upon State → a possible value or category your variable can take, such as “00”, “01”, “10”, “11”.

Frequency→ how many times that value (state) appears in your dataset.out of all your observations, state “11” occurred 1000 times, and the other states (“00”, “01”, “10”) never occurred (frequency = 0).

Example In the following example, we create a histogram with the following measurement frequencies-

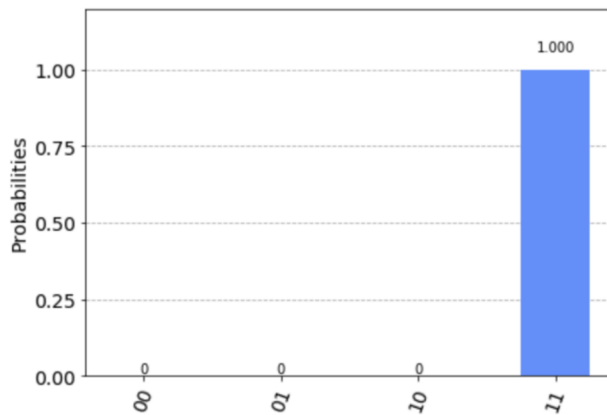


Fig. 3: Plot histogram ‘00’:0, ‘01’:0, ‘10’:0, ‘11’:1000

#### 5.4 Bell- and –GHZ

The Bell state [3] represents the most fundamental form of quantum entanglement, involving two qubits whose measurement outcomes are perfectly correlated. When one qubit is observed in the  $|0\rangle$  state, the other collapses to  $|0\rangle$ ; likewise, if one is found in  $|1\rangle$ . However, the specific outcome remains indeterminate until measurement. In Qiskit, this entangled state-mathematically expressed as  $(|0\rangle + |1\rangle)/\sqrt{2}$  can be generated using a simple quantum circuit that applies a Hadamard gate followed by a controlled-NOT(CNOT) gate.

The figure 4 presents a 2-qubit Bell-state circuit generating quantum entanglement. The top qubit  $q_0$  undergoes a Hadamard (H) gate first, placing it into a superposition of  $|0\rangle$  and  $|1\rangle$ . A subsequent CNOT gate is applied with  $q_0$  acting as the control and  $q_1$  as the target, flipping  $q_1$  only if  $q_0$  is in state  $|1\rangle$ . This operation entangles the two qubits, thereby yielding the Bell state  $(|00\rangle + |11\rangle)/\sqrt{2}$ . The entanglement is followed by measuring both qubits, as represented by the black measurement boxes that are connected to classical bits.

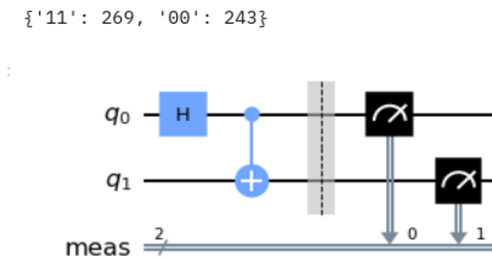
**Table 1: expected outcomes of qubit visualization (default, single and multi-qubits)**

Qubit Configuration	Visualization Tool	Expected Output	Phenomenon Demonstrated
Default	plot_bloch_multivector()	'0':1.0, '1':0.0	Ground state
Single(Superposition)	plot_bloch_multivector(), plot_histogram()	Approx50/50 each	Superposition
Two (Bell State)	plot_histogram()	'00':0.5, '11':0.5	Entanglement
Two(GHZ)	plot_state_city()	Real bars for  00> and  11>	Coherence visualization

**Table 2: Frequency measurement to create a histogram**

State	Frequency
00	0
01	0
10	0
11	1000

The counts in the result counts indeed confirm that only 00 and 11 appear nearly equally (243 and 269 times out of 512 shots), as one would anticipate from a perfectly correlated entangled Bell pair.



**Fig. 4: Shows the circuit to create the Bell state.**

Figure 5 shows the circuit to create using one more qubit, and one more CX.

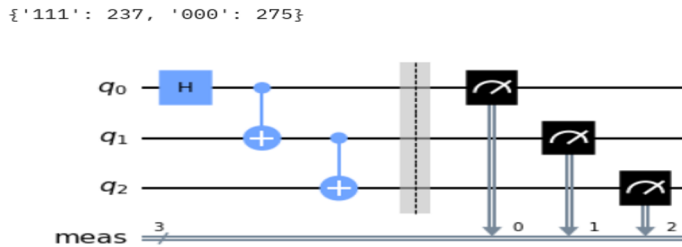
The GHZ state is similar, for three qubits:  $(|000\rangle + |111\rangle)/\sqrt{2}$ . In terms of circuit, it has one more qubit, and one more CX.

This figure presents a 3-qubit circuit used to generate a GHZ state, an entangled state involving all three qubits. The circuit starts off with the action of Hadamard gate on qubit q0 in order to put it into a superposition of  $|0\rangle$  and  $|1\rangle$ . This is followed by two CNOT gates - the first one from q0 to q1 and the second one from q0 to q2. This procedure copies the state of q0 onto q1 and q2, thus generating the GHZ state.

$$\frac{(|000\rangle + |111\rangle)}{\sqrt{2}}$$

After this entanglement, all three qubits are measured, represented by the three measurement blocks connected to classical bits. Only the results 000 and 111 are observed,

with near equal probabilities of 275 and 237 out of 512 shots. This is definitive proof that the three qubits are perfectly correlated, hence confirming this is a GHZ state.

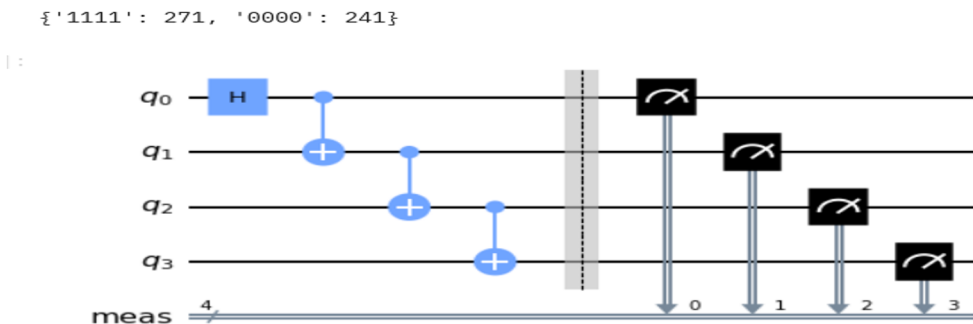


**Fig. 5: Add one more qubit, and more CX.**

The figure 6 shows a 4-qubit circuit for creating a 4-qubit GHZ state, Then, to generalize, just add a loop, which is an entangled state for which all the qubits exhibit correlated outcomes. The beginning of the circuit applies a Hadamard gate to qubit q0. This puts that qubit into the superposition of  $|0\rangle$  and  $|1\rangle$ . That then gets copied, via a chain of CNOTs—first from  $q_0 \rightarrow q_1$ , then  $q_1 \rightarrow q_2$ , then  $q_2 \rightarrow q_3$ —to the other qubits, spreading  $q_0$ 's value to all the qubits, yielding the GHZ state.

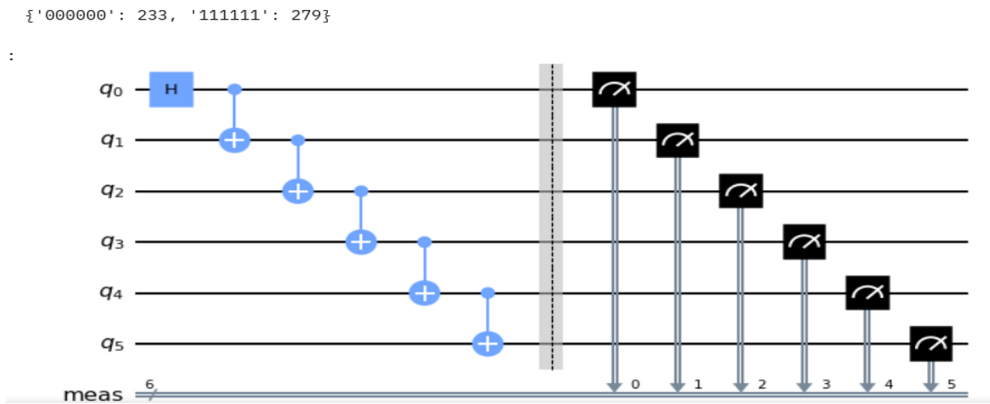
$$\frac{(|0000\rangle + |1111\rangle)}{\sqrt{2}}$$

Finally, all four qubits are measured, represented by the measurement boxes followed by classical bits. The outcome distribution consists of solely 0000 and 1111, appearing almost half each (241 and 271 times, respectively), proving strong multi-qubit correlation and thereby successful creation of GHZ states.



**Fig. 6: Add one more qubit, and more CX.**

The figure 7 shows an n-qubit GHZ state circuit, here illustrated with six qubits. First, there is a Hadamard gate on  $q_0$  that puts it into the superposition of  $|0\rangle$  and  $|1\rangle$ . Then there is a loop over the qubits with a chain of CNOT gates from each qubit to the next - namely,  $q_0 \rightarrow q_1$ ,  $q_1 \rightarrow q_2$ , ...,  $q_4 \rightarrow q_5$ . This copies the state of  $q_0$  across all qubits and entangles them: the GHZ state  $(|000000\rangle + |111111\rangle)/\sqrt{2}$ .



**Fig. 7: Add one more qubit, and one more CX.**

All qubits are measured as represented by the six boxes labeled Measurement. The outcomes only include 000000 and 111111, with close to equal counts, 233 and 279 out of 512 shots, indicating strong multi-qubit correlations. This proves that the circuit successfully prepared an extended GHZ state on all qubits.


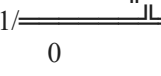



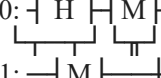

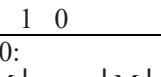
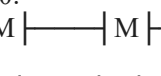
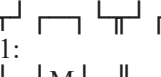
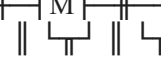
The results we obtained using Qiskit agree with the basic ideas of quantum computing explained by Nielsen and Chuang (2010). The visualizations clearly show how qubits behave when they are placed in superposition and when they become entangled, just as theory predicts.

The Bloch sphere and measurement histograms confirm that the quantum states were created and measured correctly. These outcomes match the expected results from quantum theory, showing that the algorithms we used are accurate and consistent with the standard principles of quantum computation [16].

In Table 3 shows quantum circuit measurement on the simulator and feels the difference between qubits, bits, gate with measurement and gate without measurement. after apply these on simulator, it will display in result (Table 3).

The main contribution of this paper is to implement coding part of qubit measurement visualization using Qiskit (Quantum circuit measurement on the simulator) as following-

- With qubit(s) and bit(s), without Gates and measurement.
- With 1024 shots(Q) and Bit, with Gates and measurement.
- With 2 qubits and 2 bits, without Gates and measurement.
- With 2 qubits and 2 bits, without Gates and with measurement.
- With 2 qubits and 2 bits, with Gates and measurement.
- With 2,2048 shots qubits and 2 bits, without Gates and with measurement.

Quantum Circuit Measurement on the Simulator					
S. N.	Qubits	Bits	Gates, Measurement	output	Result
1.	1	1	-, -	Empty	q: c: 1/
2.	1, 1024 shots	1	Hadamard, Measurement	{'1':512, '0':512}	q:  c:  0
3.	2	2	-, -	Empty	q_0: q_1: c: 2/
4.	2	2	-, Measurement	{'00':1024}	q 0:  q 1:  c:  0 1
5.	2	2	Hadamard, Measurement	{'00':495, '01':529}	q 0:  q 1:  c:  1 0
6.	2, 2048 shots	2	-, Measurement	{'00', 2048}	q_0:  q_1:  c:  0 1 0 1

## 6 Conclusion and Future Work

The emergence of Quantum computing is not only reshaping technological frontiers but also challenging hardware and software systems to keep pace with an increasingly complex scientific paradigm. This study has highlighted two key dimensions i.e. measurement and visualization.

Our Qiskit simulations, the core principles of quantum mechanics, in this paper were trying to demonstrate superposition, entanglement. The Bloch sphere representation illustrates superposition and the construction of Bell and GHZ states demonstrates perfect correlation. These results reinforce that single-qubit behavior measurement entanglement gives confidence that quantum circuits are correct that Qiskit's tools faithfully reflect quantum computational phenomena.

## 7 Acknowledgments

I would like to thank Chhatrapati Shahu Ji Maharaj University for providing the necessary facilities and support to carry out this research work. This work was supported in part by the Government of India, under Centre for Development of Advanced Computing (Hyderabad and IIT Roorkee), NPTEL online courses. Also thanks to IIT Kanpur provide us workshop on Quantum Computing. Thank Dr. Rabins Porwal (Professor and Head) for valuable discussions and suggestions that improved the quality of this work. I am very thank full to VIT Chennai Quick26 team organise this valuable Conference.

## References

1. Barthe, A., Grossi, M., Tura, J., & Dunjko, V. (2023). *Bloch sphere binary trees: A method for the visualization of sets of multi-qubit pure states*. arXiv. <https://arxiv.org/abs/2302.02957>
2. Bloch, F. (1946). Nuclear induction. *Physical Review*, 70(7–8), 460–474. <https://doi.org/10.1103/PhysRev.70.460>
3. Qiskit Community. (2019). *Qiskit: An open-source framework for quantum computing*. Zenodo. <https://doi.org/10.5281/zenodo.2562111>
4. Cross, A. W., Bishop, L. S., Smolin, J. A., & Gambetta, J. M. (2017). *Open quantum assembly language*. arXiv. <https://arxiv.org/abs/1707.03429>
5. Aleksandrowicz, G., et al. (2019). *QiskitAer: A high-performance simulator framework for quantum circuits*. arXiv. <https://arxiv.org/abs/1904.10397>
6. Nielsen, M. A., & Chuang, I. L. (2000). *Quantum computation and quantum information*. Cambridge University Press.
7. Chen, Z., et al. (2021). Teaching quantum computing with Qiskit and IBM Quantum Experience. *IEEE Transactions on Education*, 64(3), 223–231.
8. Preskill, J. (2018). Quantum computing in the NISQ era and beyond. *Quantum*, 2, 79. <https://doi.org/10.22331/q-2018-08-06-79>
9. Gambetta, J. M., Chow, J. M., & Steffen, M. (2017). Building logical qubits in a superconducting quantum computing system. *npj Quantum Information*, 3(1),

2. <https://doi.org/10.1038/s41534-016-0004-0M>. Ben Rabha, M.F. Boujmil, M. Saadoun, B. Bessaïs, Eur. Phys. J. Appl. Phys. (to be published)

#### Books

10. Temme, K., Bravyi, S., & Gambetta, J. M. (2017). Error mitigation for short-depth quantum circuits. *Physical Review Letters*, *119*(18), 180509. <https://doi.org/10.1103/PhysRevLett.119.180509>. Couturier, Y.H. Abou and E. Grolleau, Element of nuclear safety, (EDP Sciences, Les Ulis, 2019)
11. Nation, P. D., Kang, H., Sundaresan, N., & Gambetta, J. M. (2021). Scalable mitigation of measurement errors on quantum computers. *PRX Quantum*, *2*(4), 040326. <https://doi.org/10.1103/PRXQuantum.2.040326>. N. Ozisik, Radiative transfer and interactions with conduction and convection (John Wiley and Sons, New York, 1973)

#### Proceedings

12. Nielsen, M. A., & Chuang, I. L. (2010). *Quantum computation and quantum information* (10th anniversary ed.). Cambridge University Press.
13. IBM Quantum Team. (2025). *Qiskit visualization—plot histogram function*. [https://docs.quantum.ibm.com/api/qiskit/visualization/plot\\_histogram](https://docs.quantum.ibm.com/api/qiskit/visualization/plot_histogram)
14. IBM Quantum Team. (2025). *Visualize results in Qiskit: plot histogram and state visualization*. <https://quantum.cloud.ibm.com/docs/api/qiskit/visualization>
15. IBM Quantum Team. (2023). *Qiskit visualization module: plot\_bloch\_vector, plot\_bloch\_multivector, plot\_histogram*. <https://docs.quantum.ibm.com/api/qiskit/visualization>
16. Qiskit Tutorials Team. (2022). *Plotting data in Qiskit*. GitHub. [https://github.com/Qiskit/qiskit-tutorials/blob/master/tutorials/circuits/2\\_plotting\\_data\\_in\\_qiskit.ipynb](https://github.com/Qiskit/qiskit-tutorials/blob/master/tutorials/circuits/2_plotting_data_in_qiskit.ipynb)
17. asgunzi. (2025). *Bell-and-GHZ: Implementation of Bell and GHZ states using Qiskit*. GitHub. <https://github.com/asgunzi/Bell-and-GHZ>
18. Asgunzi. *Bell-and-GHZ: Implementation of Bell and GHZ States Using Qiskit*. GitHub, 2025, <https://github.com/asgunzi/Bell-and-GHZ>. Accessed Day Month Year.
19. Nielsen, Michael A., and Isaac L. Chuang. *Quantum Computation and Quantum Information*. 10th Anniversary ed., Cambridge University Press, 2010.

## A Online Resources

The official Qiskit documentation is a great place to start. In Qiskit, the “Visualizations” module page provides a complete list of built-in plotting and visualization functions, such as `plot_histogram()` (for measurement outcome counts), `plot_bloch_multivector()`, `plot_state_qsphere()`, `plot_state_city()`, `plot_state_hinton()`, etc. — covering both measurement results and quantum-state visualizations.

For example, `plot_histogram()` is explicitly documented for visualizing output counts from measurements, which is useful when you run circuits on a simulator or real back-end and want a quick bar-chart view of the resulting distributions.

It contains a set of state-visualization functions - Bloch sphere per qubit, Q-sphere, density-matrix cityscape, or Hinton diagrams-to visualize the pre-measurement quantum state of one or more qubits. That makes it easier to develop intuition about superposition, entanglement, and the structure of multi-qubit states.

In addition to the official documentation, teaching guides and tutorials help you make use of these tools in practice. Such is the case with the "Visualize results" guide on the Qiskit documentation site: after explaining how to use `plot_histogram` after you run a circuit with measurement, it shows how to extract counts and display them. This is particularly useful if you are starting out with quantum computing, because it provides step-by-step methods to go from circuit design to the visualization of measurement statistics.

Also, more extensive resources like "Qiskit Pocket Guide" — book/tutorial, showcase typical workflows; for example, preparation of a Bell-state circuit, measurement, and plotting of histograms of the outcome counts.

If you ever intend to do more than simple single- and two-qubit examples—especially if you are looking at multi-qubit systems, entanglement, or sets of states—there are some research papers on advanced visualization frameworks. As just one example, there is a preprint entitled "Bloch Sphere Binary Trees: A method for the visualization of sets of multi-qubit systems pure states" which describes how an entire set of arbitrary multi-qubit pure states can be represented in a "binary tree of Bloch spheres" by applying Schmidt decomposition and then representing the result on the Bloch sphere to make high-dimensional quantum state spaces more accessible and intuitive.

Similarly, the published work "Visualizing Quantum Circuits: State Vector Difference Highlighting and the Half-Matrix" offers novel visualization approaches to show how quantum states evolve under gates: highlighting amplitude changes across layers and giving more complete insight than per-qubit Bloch-sphere views. Finally, community resources such as forums, blogs, and online tutorials often provide practical examples of common pitfalls and user experiences. While these are a bit less formal than documentation or papers, they can help you troubleshoot real-world issues—how to extract measurement counts correctly, differences between simulator and real back-end, and caveats when visualizing multi-qubit states.