

JOIN LIVE



Quantum Machine Learning - Intro & DEMO

LIVE ONLINE SESSION

10-Sep @ 11.00 AM IST

BY PRASANNA VENKATESH J

AI & CLOUD EVANGELIST



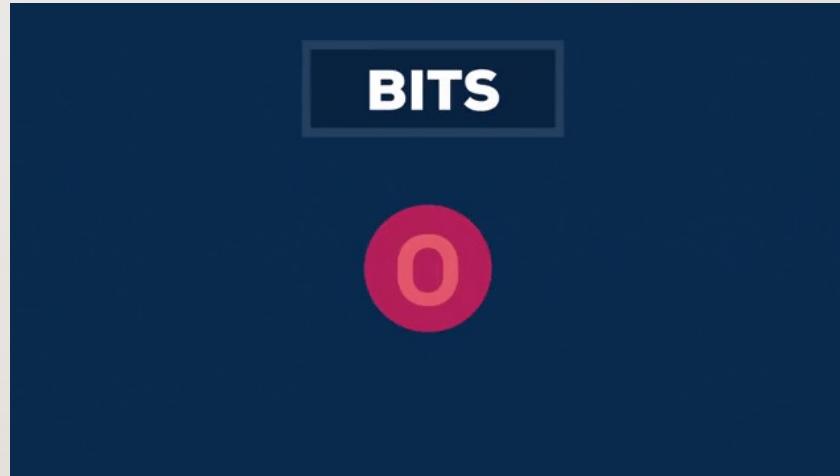
[HTTPS://WWW.LINKEDIN.COM
/IN/PRASANNA-VENKATESH-
JAYAPRAKASH/](https://www.linkedin.com/in/prasanna-venkatesh-jayaprakash/)

PRASANNAVJ@GMAIL.COM

+91-9840337443

CLASSICAL COMPUTERS

- Basis of classical computers
- BITS
 - Either Zero or One
 - Only one state at a time



SIMPLE COIN FLIP

- Is it Head? Or Tails?
- What is the value when it is in REST?
- What is the value when it is tossed?

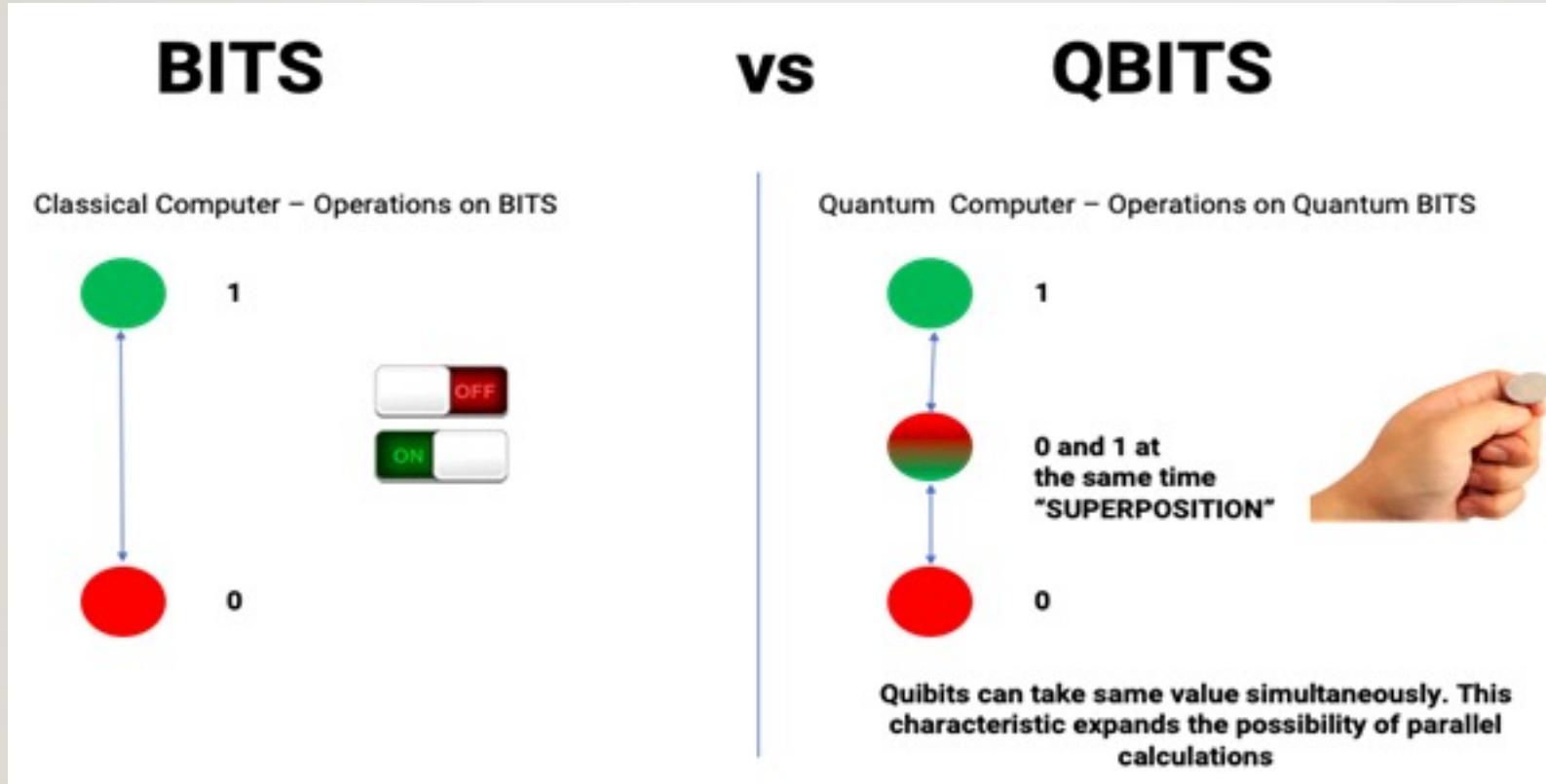




SIMPLE COMPARISON

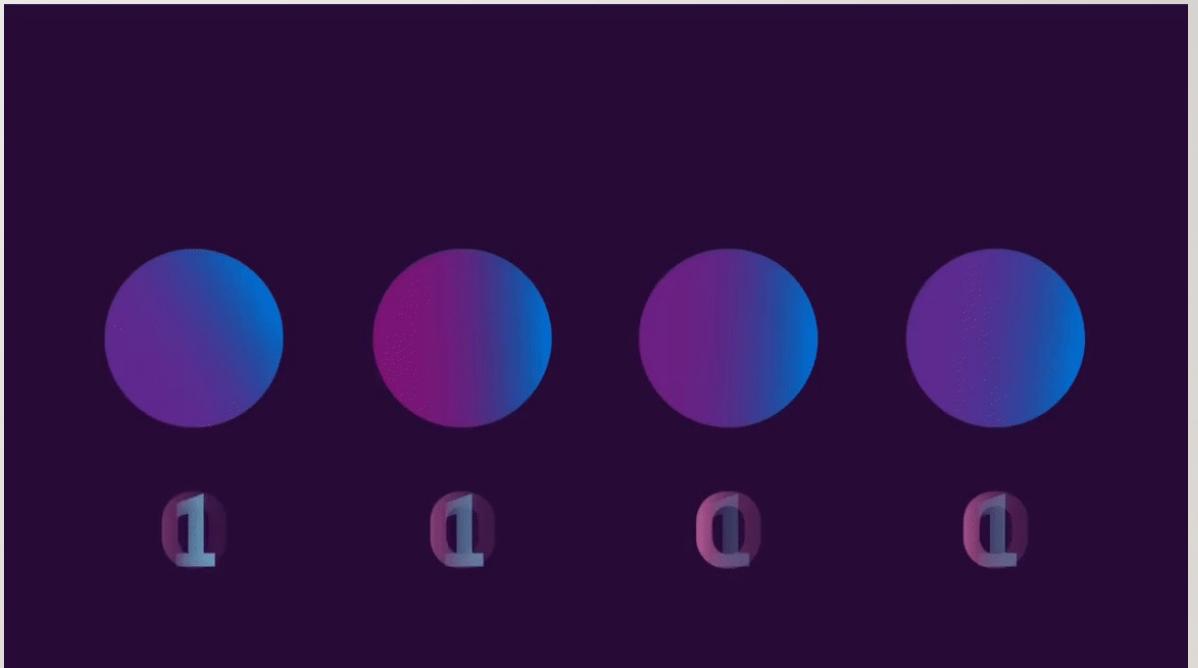
- **Coin at stable stage on the hand – compared to classical computers BITS.**
 - At any point only one state is possible.
- **Coin at the tossed stage – quantum computing super position stage.**
 - At any point two possible values at the same time for a bit
 - Quantum Bit
 - QuBIT

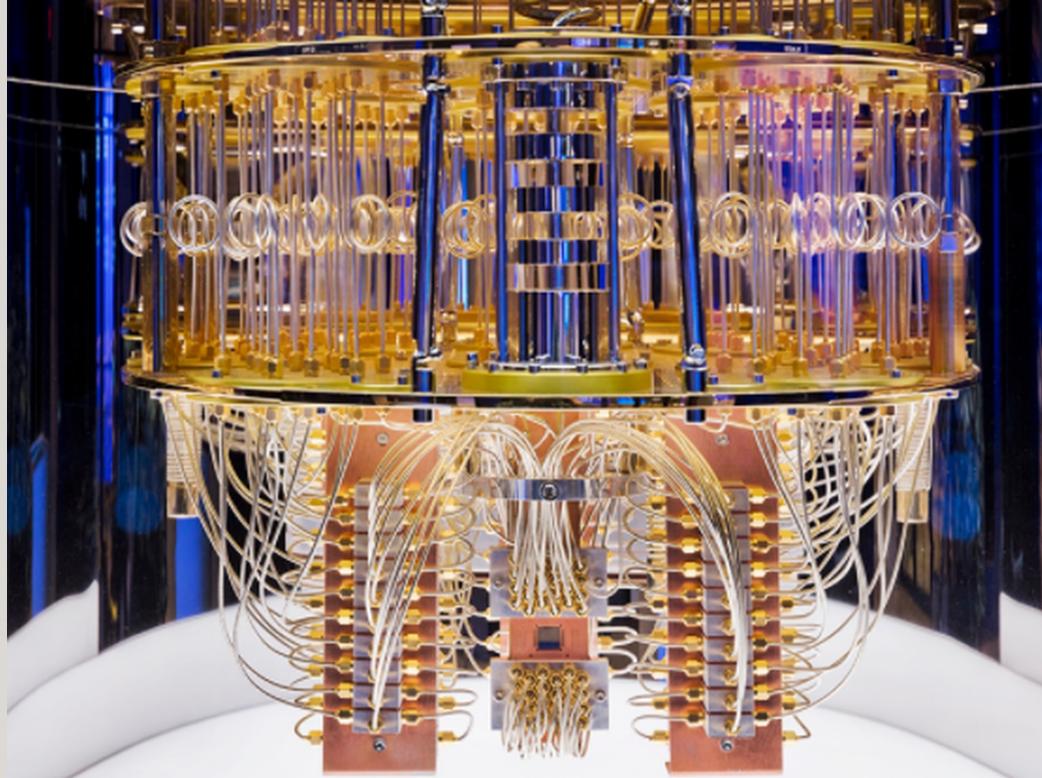
BITS VS QBITS



QUBIT – SUPER POSITION STATE

- At the same time has both zero and one.
- Image shows 4 qubits and the value changes both 1 and 0 at the same time.
- This state is called super position state





SAMPLE QUANTUM COMPUTERS

QUANTUM COMPUTING PLAYERS



GOOGLE BRISTLEcone

- 72 qubit quantum computer in a chip.
- Mainly used for research and innovations
- For creating algorithm building



GOOGLE SYCAMORE CHIPS



2^{52}

Times of processing.

Google showcased the
Quantum Supremacy???

- New computational
capability that is not
feasible by typical super
computers of the world.



IBM QUANTUM COMPUTING PROGRESS



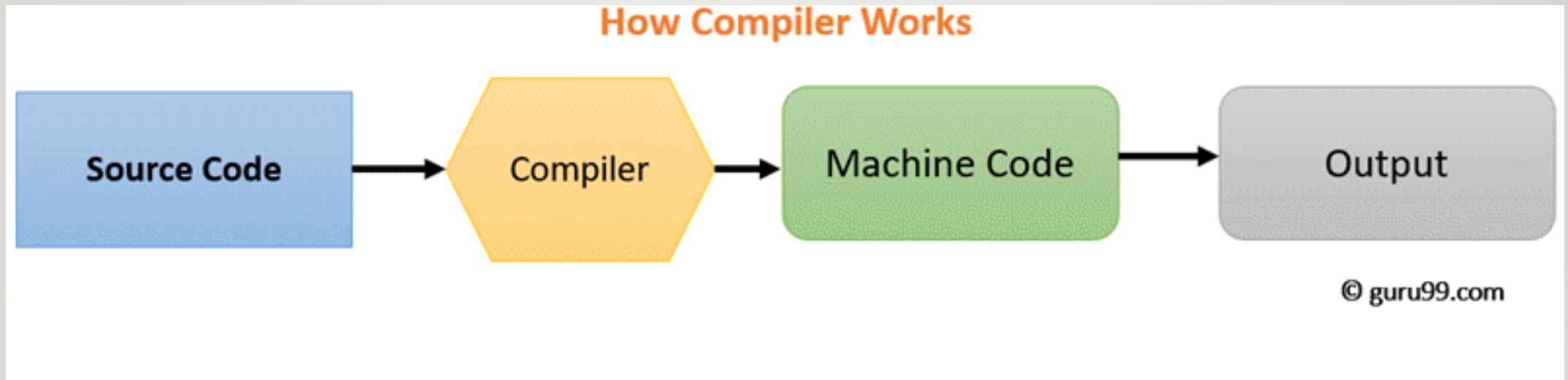
June-2022 release by IBM

1386-qubit processor – Kookaburra

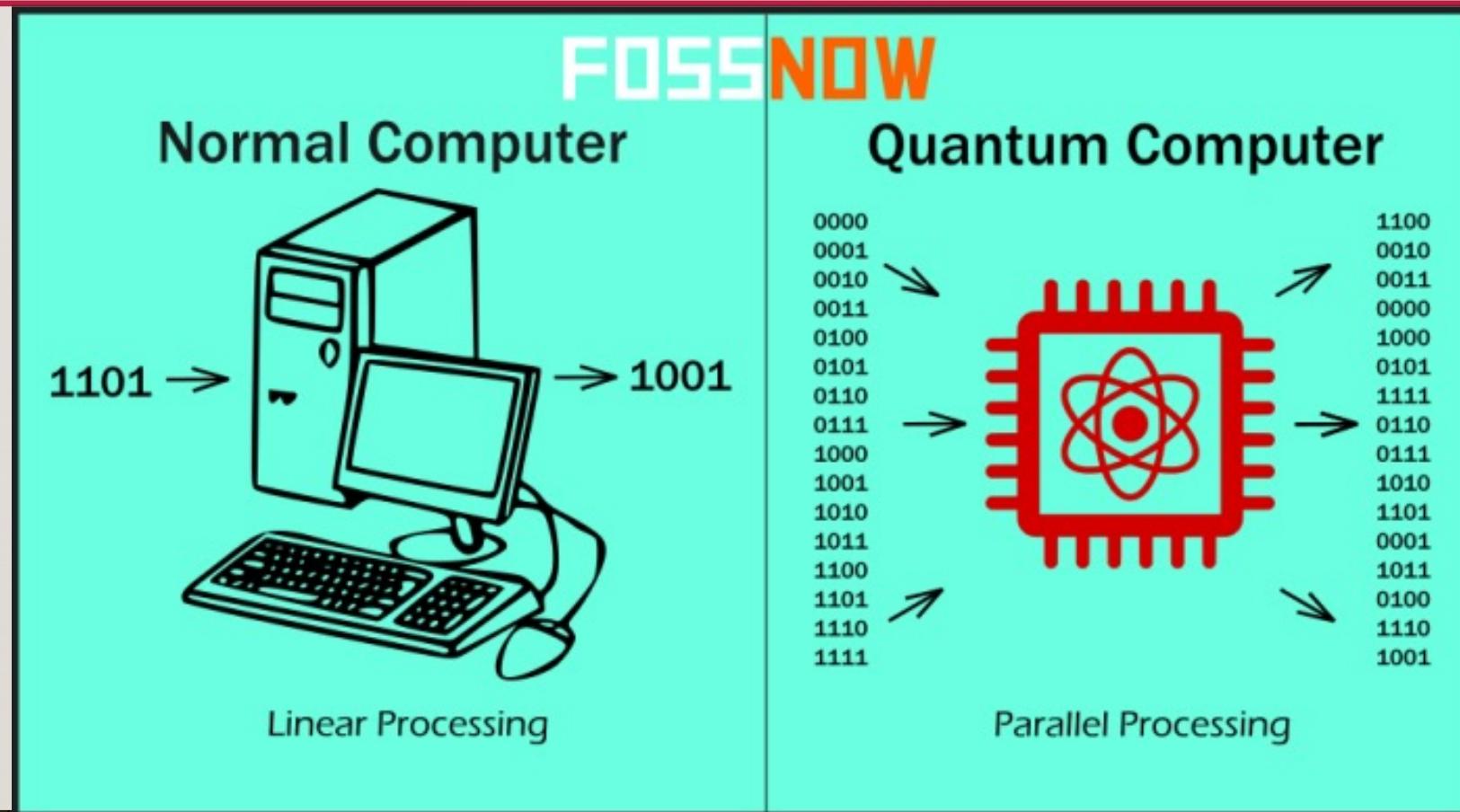
4,158-qubit POC system built using three connected Kookaburra processors by 2025



CLASSICAL COMPUTER EXECUTION METHOD



QUANTUM COMPUTER

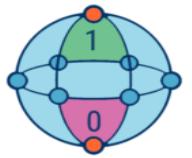


<https://www.linkedin.com/in/prasanna-venkatesh-jayaprakash/>

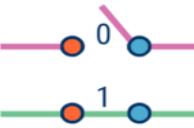
Quantum Computing

Vs.

Classical Computing

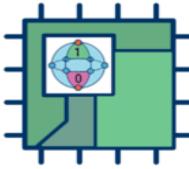


Calculates with qubits, which can represent 0 and 1 at the same time

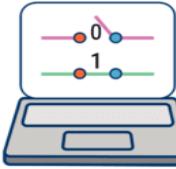


Power increases exponentially in proportion to the number of qubits

Calculates with transistors, which can represent either 0 or 1



Quantum computers have high error rates and need to be kept ultracold



Classical computers have low error rates and can operate at room temp



Well suited for tasks like optimization problems, data analysis, and simulations



Most everyday processing is best handled by classical computers

COMPARING QUANTUM AND CLASSICAL COMPUTING

**Quantum Universal
Languages**
Full-stack libraries
Quantum algorithms
Quantum circuits
Assembly language
Hardware
XACC
ProjectQ
CirqProjectQ

IBM	Rigetti	DWave	Xanadu	Google	Microsoft*	Qilimanjaro*	
QISKit	Forest		Strawberry Fields	Cirq	Quantum Development Kit		
QISKit Aqua	Grove	QSage ToQ		OpenFermion -Cirq	Q#		
QISKit Terra	pyquil	qbsolv		Cirq		Qibo	
Open QASM	Quil	QMASM	Blackbird	Other Quantum Machine Instruction Languages			
Quantum device							

* Hardware under development. Quantum programs are run on their own simulators.

VARIOUS QUANTUM PROGRAMMING FRAMEWORKS

a 1950s computing

Assembly language
(low-level) programs

Relay circuits and
discrete wires

b Classical computing today

Algorithms

High-level languages

Compiler

Classical architecture
(memory, arithmetic
operations, control
operations, communication)

Hardware building
blocks: gates, bits

VLSI circuits

Semiconductor
transistors

c Quantum computing

Algorithms

High-level languages

Classical compiler

Quantum compiler

Classical
architecture
(control operations)

Hardware building
blocks (gates, bits)

VLSI circuits

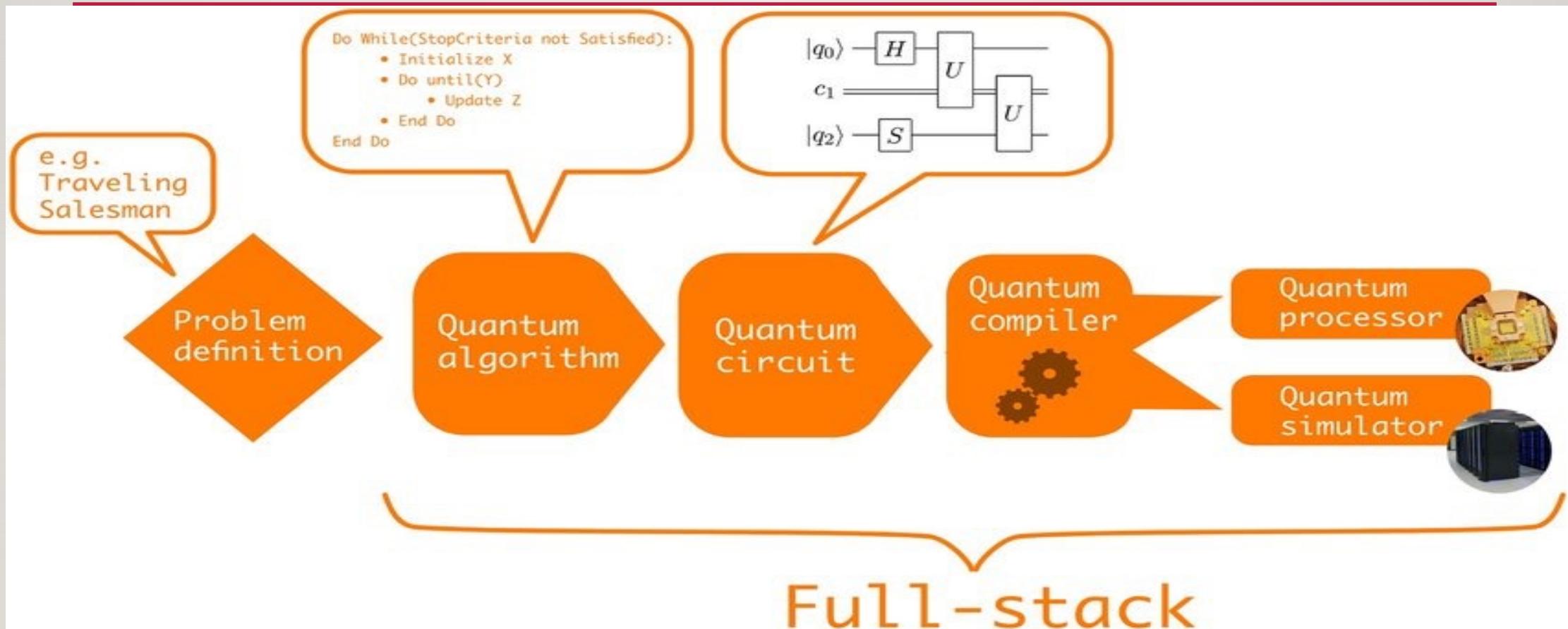
Semiconductor
transistors

Quantum
architecture
(QC gates, qubits,
communication)

Error-correction
and control pulses

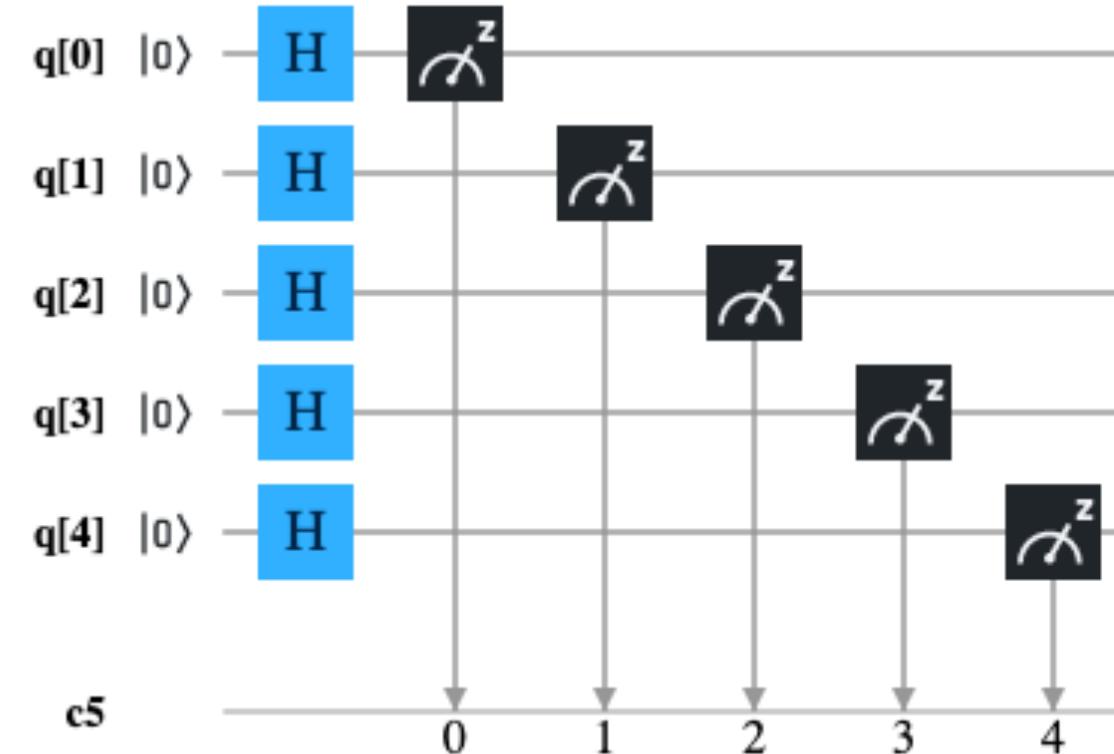
Underlying technology
(semiconductors,
trapped ions)

QUANTUM PROGRAMMING FLOW

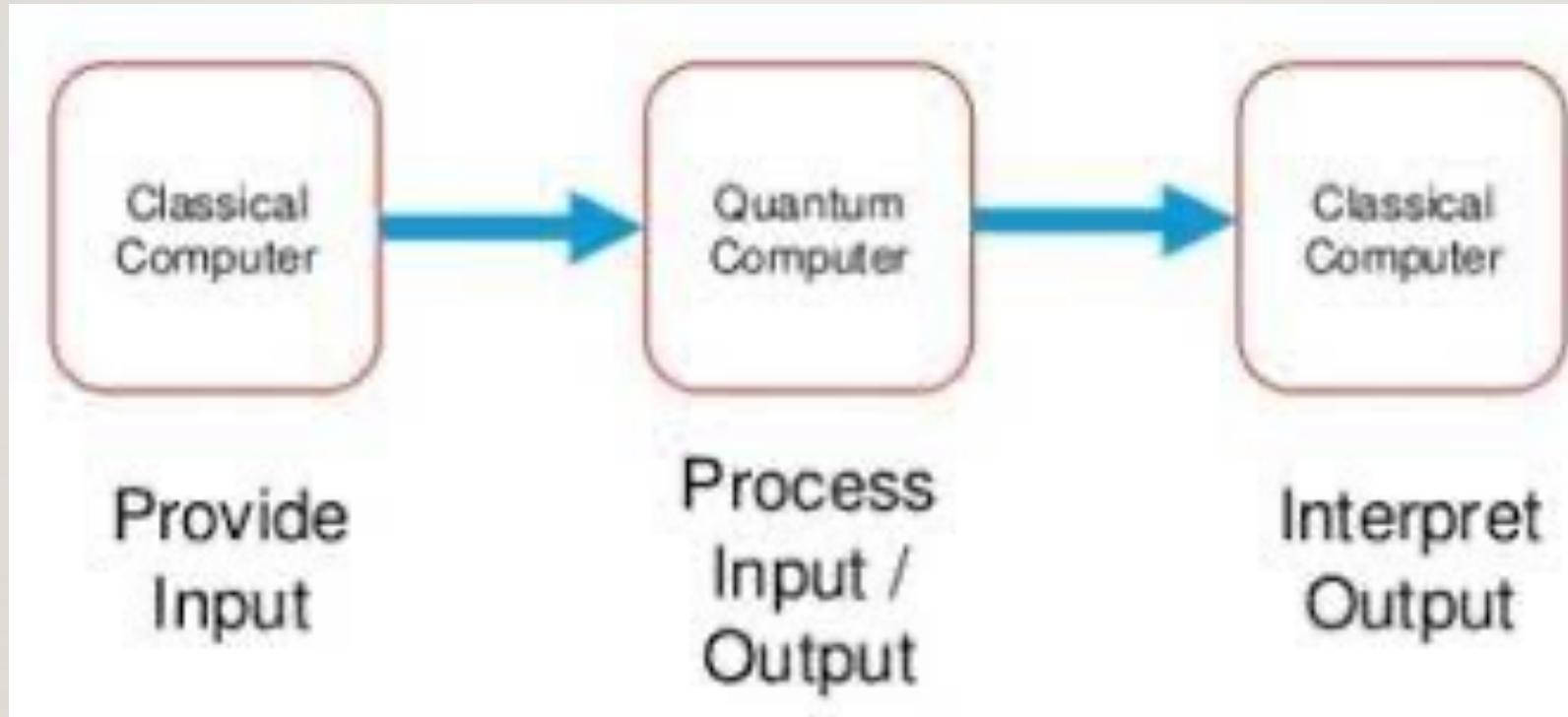


QUANTUM CIRCUIT – EXAMPLE

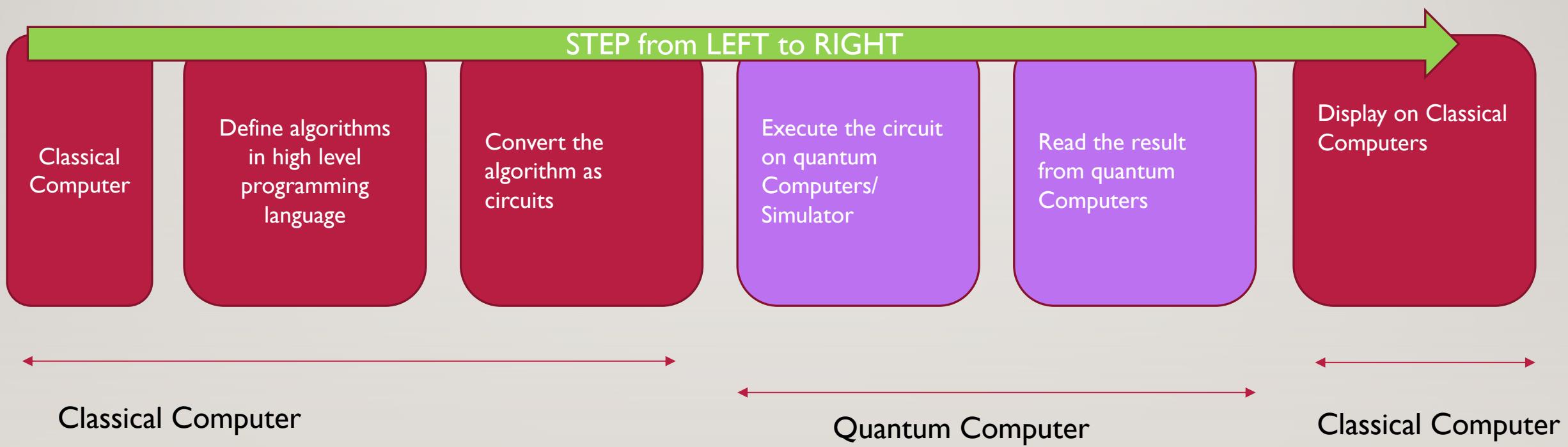
RANDOM NUMBER GENERATION ALGO



INTERACTING WITH QUANTUM COMPUTER



CLASSICAL COMPUTER TO QUANTUM COMPUTER INTERACTION



GOOGLE CIRQ – INTRODUCTION

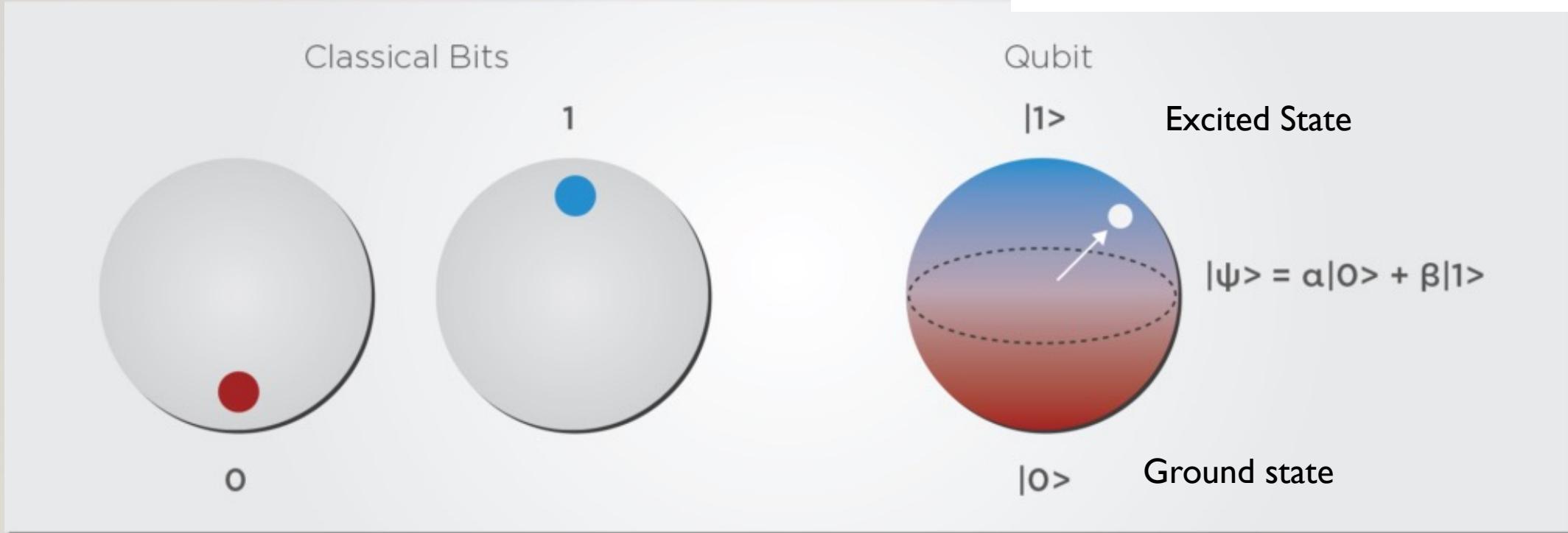
- Open Source Programming language for Quantum Algorithms and Circuit Building.
- Available on GitHub - <https://quantumai.google/cirq> \$ <https://github.com/quantumlib/cirq>
- Python library for writing, manipulating, and optimizing quantum circuits and running them against quantum computers and simulators.
- Enable users to create Qubits, Gates, Operations, Circuits, Moments, Apply simulation or real device execution and visualize the result.



QUBIT INTRO

$|0\rangle$ = Ket Notation to represent the Qubit.

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



QUANTUM QUBIT IN BLOCH SPHERE

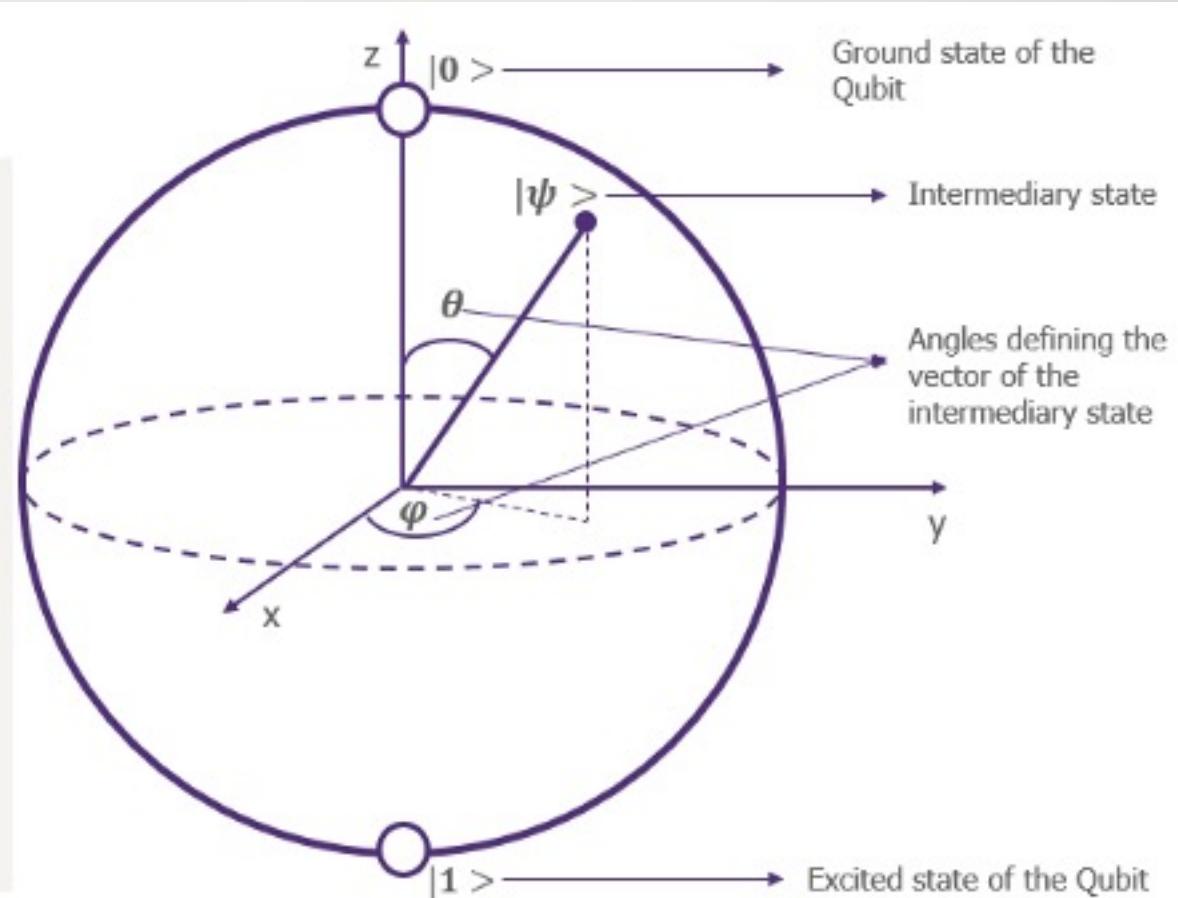
State of the qubit Probability of the **0** state Probability of the **1** state

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$
$$|\alpha|^2 + |\beta|^2 = 1$$

The relationship between α and β according to the **Max Born rule**, related to the **Schrodinger** wave function that defines the states $|0\rangle$ and $|1\rangle$

$$|\psi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \rightarrow \begin{bmatrix} \cos(\frac{\theta}{2}) \\ \exp(i\varphi)\sin(\frac{\theta}{2}) \end{bmatrix}$$

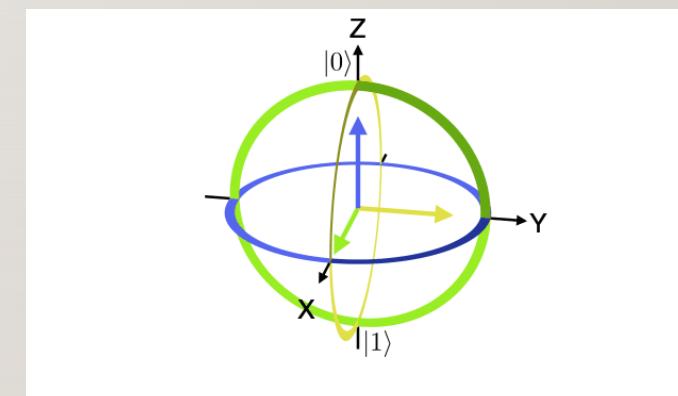
Euler formula



BASICS OF QUANTUM COMPUTING

- State Vector Representation of Qubit
 - state of a physical system in quantum mechanics is represented by **a vector belonging to a complex vector space**. This vector is known as the state space of the system.
 - Such a physical state of a **quantum system** is represented by a symbol $| \rangle$, known as a **ket**.

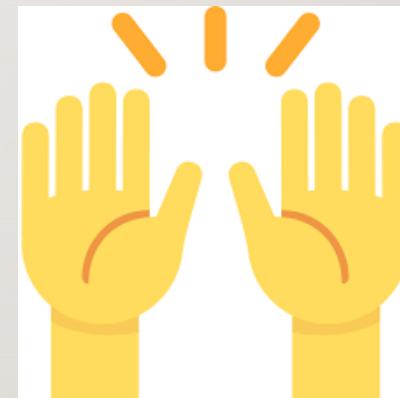
$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$



BASICS OF QUANTUM COMPUTING

- **Quantum Entanglement**

- If two qubits are in quantum entanglement then if one qubit value is observed other qubit value is automatically identified.
- i.e you have two gloves in a box
 - If you take out left hand glove
 - You are sure and know that the other glove is right hand one!
- CNOT Gate is used for quantum entanglement



PRACTICAL- I

- Please open the file - **Quantum ML Workshop - Intro to Cirq - I.ipynb** in Google COLAB.
- Download the above file from Github Link:
<https://github.com/prasannavj/QuantumMachineLearning>
- Google COLAB:
 - Search for Google COLAB
 - Click on the result
 - File -> Upload Notebook

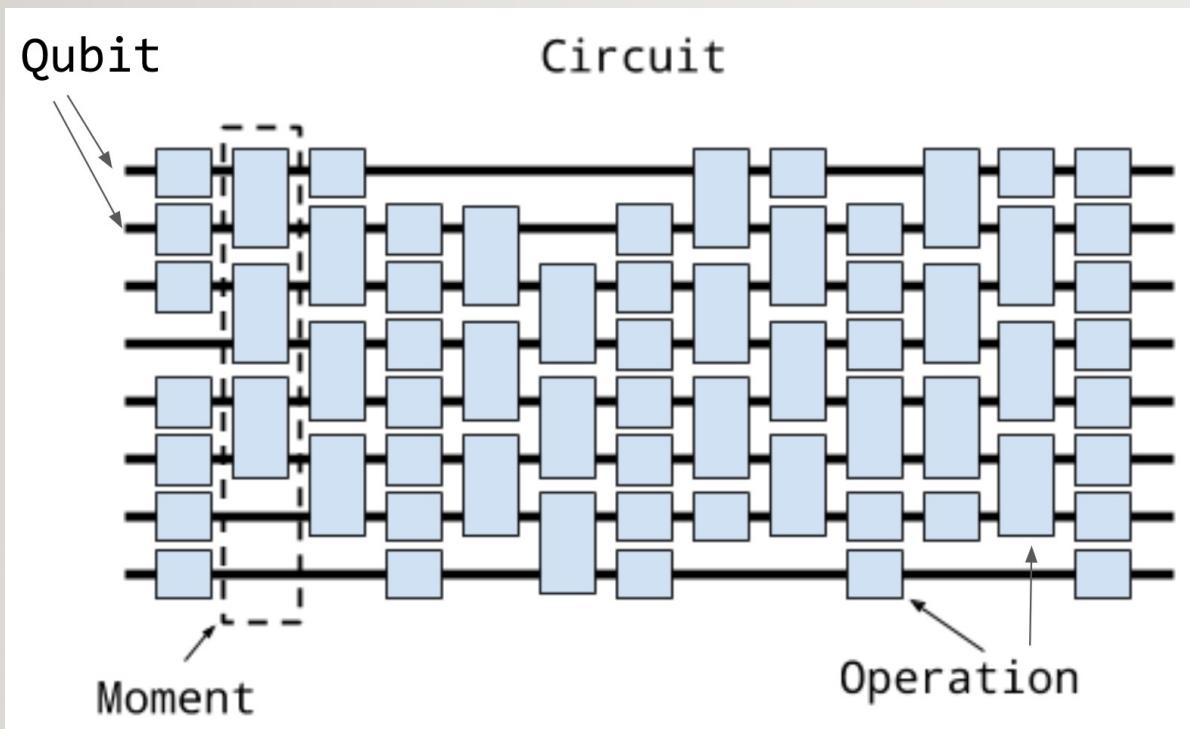


REPRESENTING QUBIT IN CIRQ

- 3 type of Qubits
 - Named Qubit
 - `cirq.NamedQubit('target')`
 - Line Qubit
 - `cirq.LineQubit.range(2)`
 - Grid Qubit
 - `cirq.GridQubit(4, 5)`



QUBITS, OPERATIONS, MOMENTS AND CIRCUITS

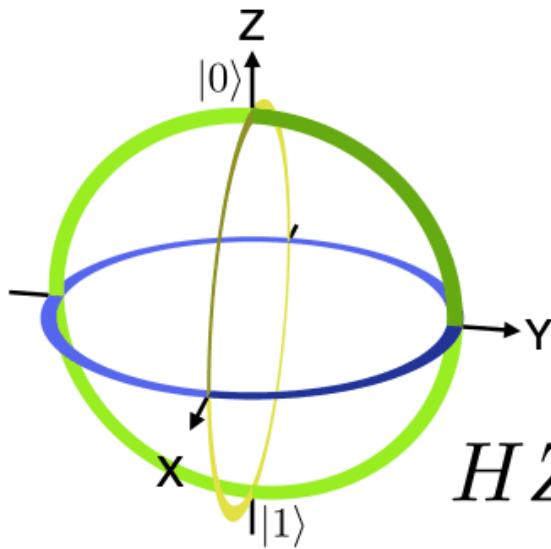


- A **Circuit** is a collection of **Moments**.
- A **Moment** is a collection of **Operations** that all act during the same abstract time slice.
- An **Operation** is an effect that operates on a specific subset of **Qubits**.
 - The most common type of Operation is a Gate applied to several qubits (a "GateOperation").



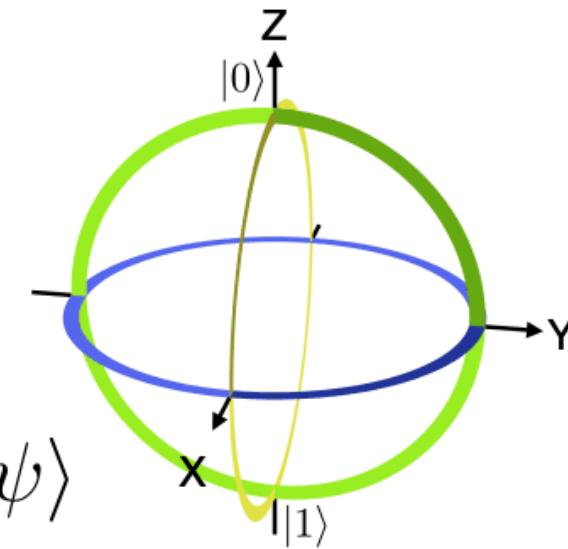
SAMPLE CIRCUIT FLOW

$$\begin{array}{c} \text{---} \boxed{H} \text{---} \boxed{Z} \text{---} \boxed{H} \text{---} = \text{---} \boxed{X} \text{---} \\ \end{array}$$



=

$$HZH |\psi\rangle$$



GATES AND OPERATIONS

- Gates Introduction
 - CNOT gate – to Entangle two qubits.
 - `cirq.CNOT`
 - Hadamar Gate - Take the qubit to the superposition.
 - `Cirq.H`
 - X Gate – to rotate the X axis
 - `Cirq.X`
 - Y gate – to rotate on Y axis
 - `Cirq.Y`
 - Z Gate – to rorate on Z axis
 - `Cirq.Z`



GATES AND OPERATIONS

- Operations
 - Square Root calculation
 - `cirq.X**0.5`
 - Perform Square calculation
 - `cirq.YPowGate(exponent=0.25)`



CIRCUIT AND SIMULATION

- Create circuit
 - `circuit = cirq.Circuit()`
 - `circuit.append()`
- Simulating the result
 - `cirq.Simulator()`
 - `simulator.run()`



ADVANCED SIMULATOR

- Qsim – Quantum simulator – dedicated advanced simulator python open source library for the simulation of the quantum computing
 - !pip install qsimcirq
 - qsimcirq.QSimSimulator()
 - qsim_simulator.simulate(circuit)



PRACTICAL-2

- Please open the file - **Quantum ML Workshop - Advanced Cirq - 2.ipynb** in Google COLAB.
- Download the above file from Github Link:
<https://github.com/prasannavj/QuantumMachineLearning>
- Google COLAB:
 - Search for Google COLAB
 - Click on the result
 - File -> Upload Notebook



ADVANCED CIRCUITS AND OPERATIONS

- Creating custom gates
 - Python class inherit the details from `cirq.Gate`
 - Define Number of Qubit details.
 - Define Unitary Matrix details
 - Define Circuit diagram information
- Importing and exporting the circuits
 - `cirq.to_json`
 - `cirq.read_json`

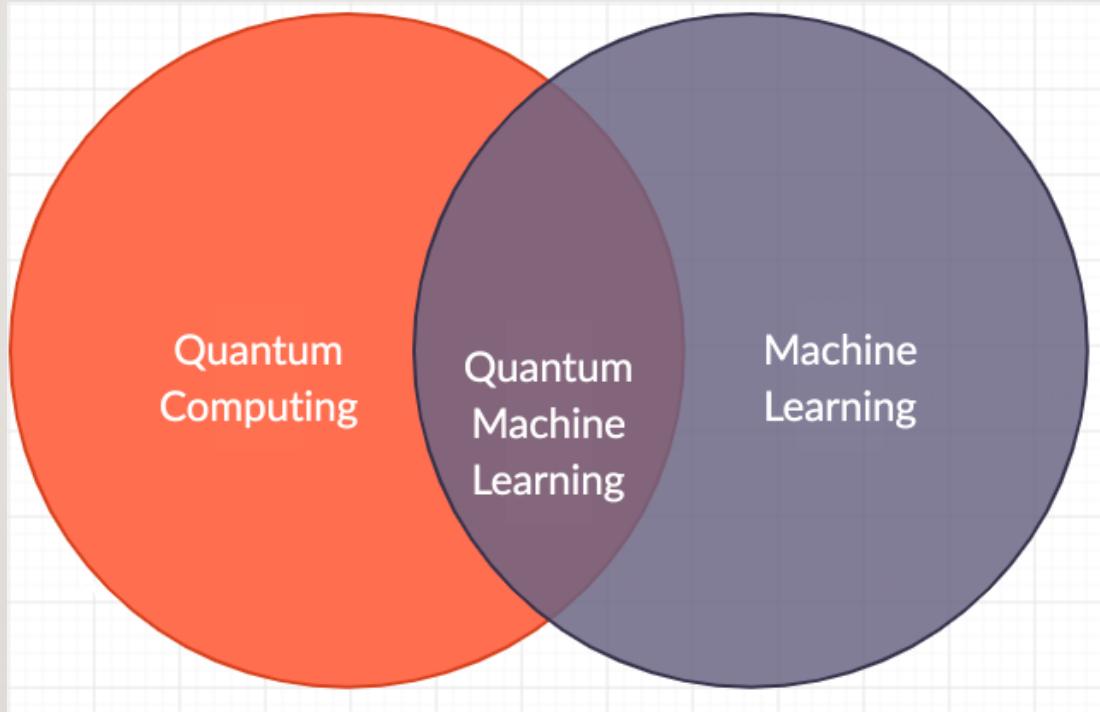


INTERACTING WITH THE QUANTUM DEVICE

- Importing device access details
 - `cirq_google.Sycamore` – to get the Sycamore device for our execution.
 - `cirq_google.Sycamore.metadata.gate_durations` – to get duration of an operations
 - `cirq_google.Sycamore.validate_operation`
 - Quantum devices has many device restrictions and the `validate_operation` used to validate whether these operations are valid or not before executing.
 - One of the basic validation is Operation on non-adjacent qubits will raise an error.



QUANTUM MACHINE LEARNING



GOOGLE TENSORFLOW QUANTUM

- Quantum machine learning library from google
- Open source library
- Released in 2020
- In collaboration with univ of waterloo and Voswagon team
- TensorflowQuantum(TFQ)
- Under the hood “Cirq” integrate “Tensorflow”



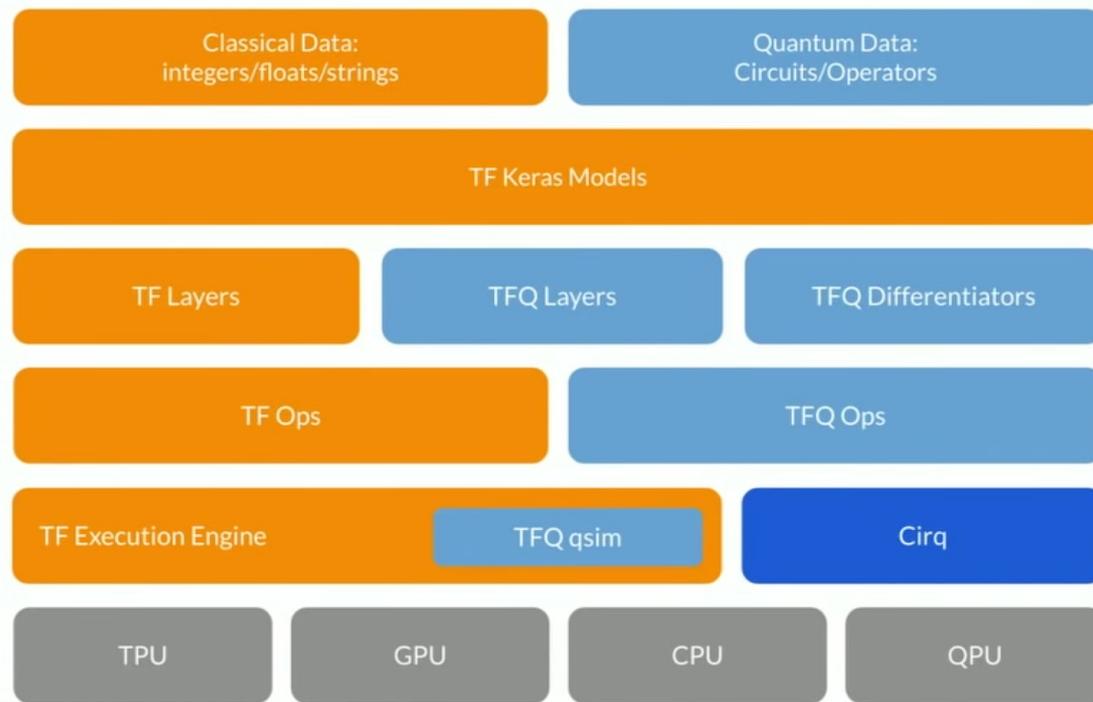
UNDERSTAND TENSORFLOW QUANTUM(TFQ)

- Quantum Data
 - *Quantum data* exhibits superposition and entanglement, leading to joint probability distributions that could require an exponential amount of classical computational resources to represent or store.
- Hybrid Quantum classical models
 - Tensorflow extended to support the quantum hybrid quantum classical algorithms

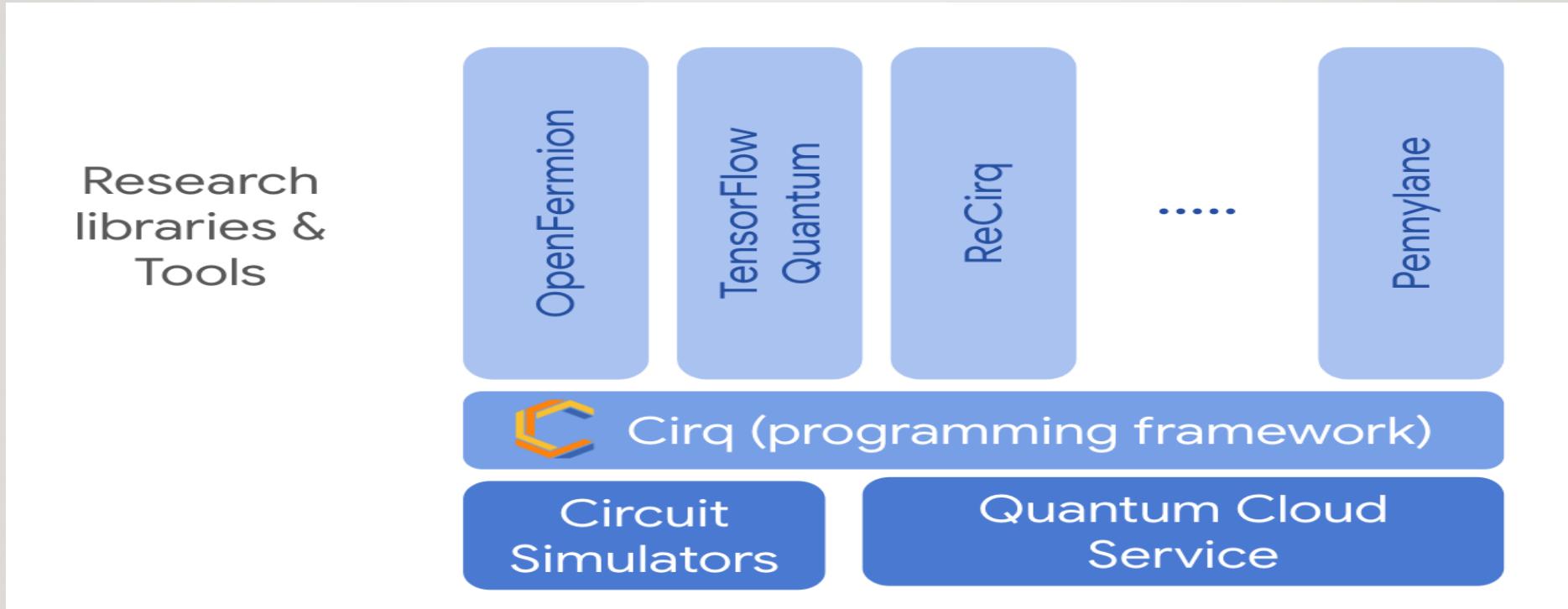


TENSORFLOW QUANTUM

Software stack



TENSORFLOW QUANTUM



SUPPORTED QUANTUM CLOUD SERVICES

Company	Type of Quantum Computer
<u>Alpine Quantum Technologies</u>	Trapped ions
<u>IonQ</u>	Trapped ions
<u>Microsoft Azure Quantum</u>	Trapped ions (Honeywell and IonQ)
<u>Pasqal</u>	Neutral atoms
<u>Rigetti</u>	Superconducting qubits



TENSORFLOW QUANTUM – DESIGN PRINCIPLES

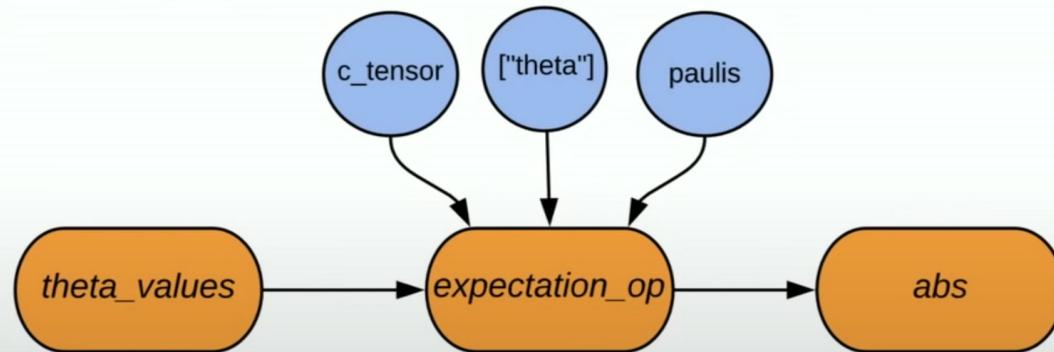
1. **Differentiability:** Must support differentiation of quantum circuits and hybrid backpropagation.
2. **Circuit Batching:** Quantum data loaded as quantum circuits, training over many different quantum circuits in parallel.
3. **Execution Backend Agnostic:** Switch from a simulator to a real device easily with few changes.
4. **Minimalism:** A bridge between Cirq and TF; does not require users to re-learn how interface with quantum computers or solve problems using machine learning.



TENSORFLOW QUANTUM

Software architecture

- Circuits are **TENSORS**, use Cirq constructs to generate these tensors
- Converting circuits to classical data (aka running or simulating them) can be done by **OPs**

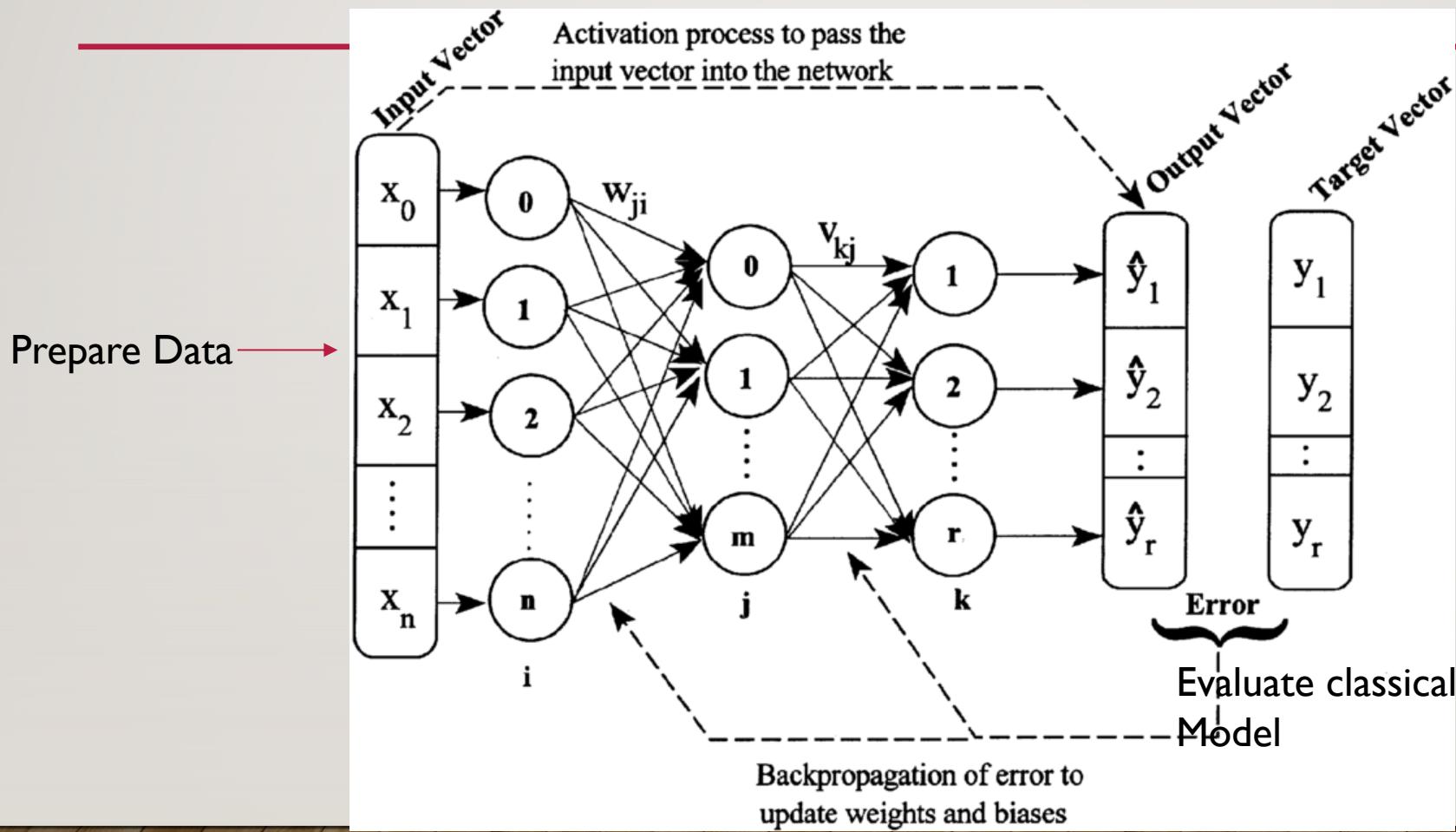


PRACTICAL-3

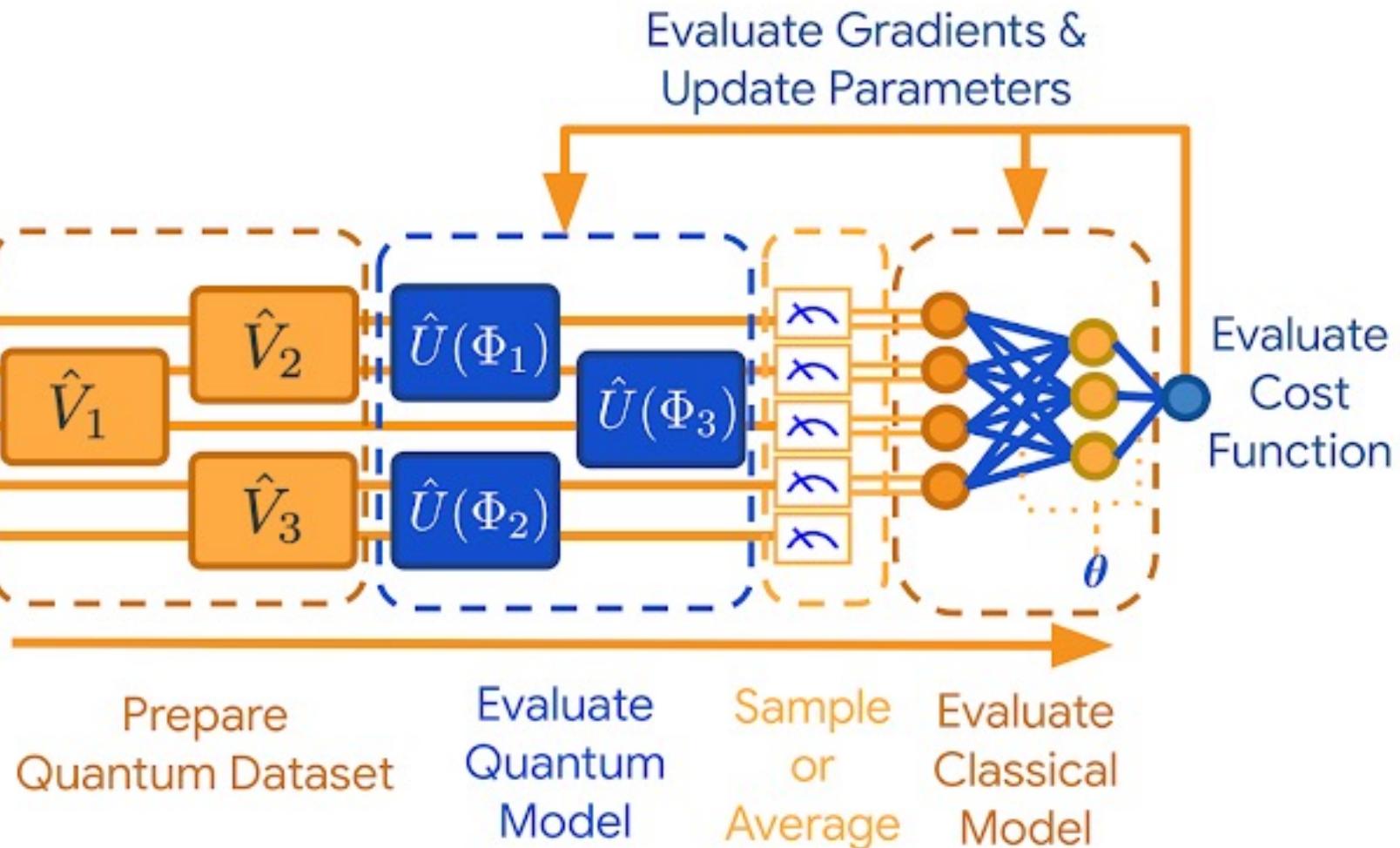
- Please open the file - **Quantum ML Workshop - Quantum Machine Learning Basics - 3.ipynb** in Google COLAB.
- Download the above file from Github Link:
<https://github.com/prasannavj/QuantumMachineLearning>
- Google COLAB:
 - Search for Google COLAB
 - Click on the result
 - File -> Upload Notebook



TYPICAL NEURAL NETWORK

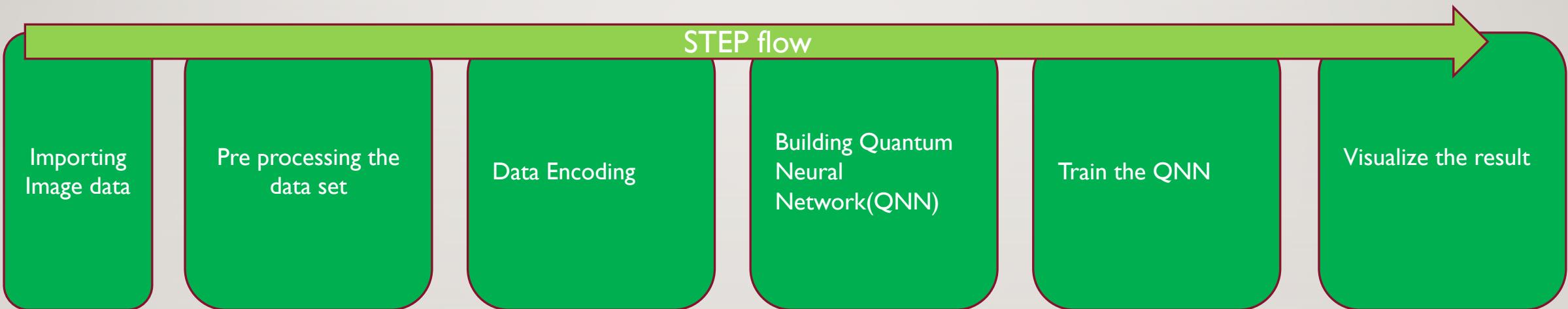


USING TENSORFLOW QUANTUM



HYBRID QUANTUM MACHINE LEARNING APPROACH

- Image Classification on Fashion MNIST with TensorFlow-Quantum and Cirq



SYMPY AND CIRQ

- SYMPY – Symbolic parameters to the circuit can be added using the SYMPY library
- SVGCircuit - for better visual representation of the Circuits (A Custom Circuit Visualizer)
- QNN – Quantum Neural Network
- Cirq.xx – tensor product of two x gate
- Cirq.yy – tensor product of two y gate



PRACTICALS - 4

- Please Download the file from GitHub Repo:
 - <https://github.com/prasannavj/QuantumMachineLearning>
- Quantum ML Workshop - Quantum Machine Learning Advanced - 4 -
Image_Classification_on_Fashion_MNIST_with_TensorFlow_Quantum_and_Cirq.ipynb

