

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

This is one of the greatest books for beginners in quantum computing, in my opinion, especially for those who might feel intimidated by the abundance of complicated online resources. The book's Analogies section, which simplified complicated topics for laypeople, was its strongest point. The majority of the book's issues are resolved below with little to no prior knowledge required, and repeated reads of the book made nearly every topic easier.

Note:- This book is solved according to my understanding. please email me for any comments or feedback.

Book link

<https://link.springer.com/book/10.1007/978-3-030-98339-0>

Chapter 1

Basic intro

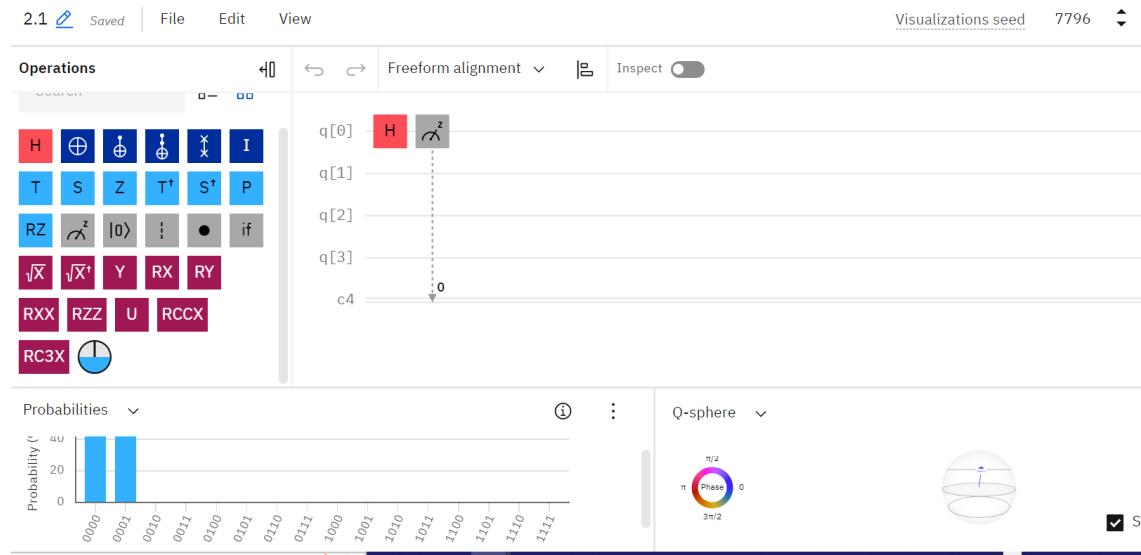
Code link

<https://github.com/nithingovindugari/IntrotoQC>

Chapter 2

2.1) Created IBM account

2.2)



2.3)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

### Details

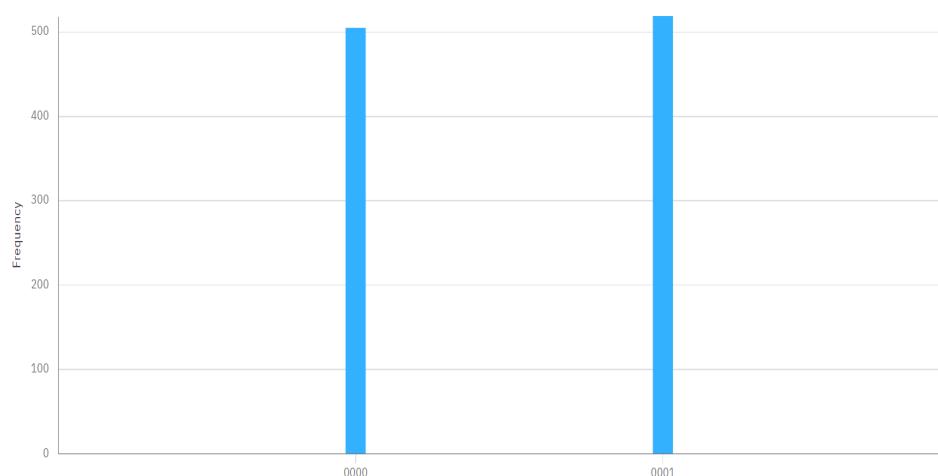
23m 57.8s

Total completion time

ibmq\_manila

Backend

Histogram



# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 3

### 3.1) Had a google collab account already

### 3.2 & 3.3 & 3.4)

```
import numpy as np
from sklearn.preprocessing import normalize

#problem3
v1 = np.array([[np.sqrt(3)],[1]])
v2 = np.array([[1],[1]])
inner_result = np.vdot(v1,v2)
print(inner_result)
print(v1.shape)

v3 = np.array([[1j],[1]])
v4 = np.array([[ -1j],[1]])
inner_result1 = np.vdot(v3,v4)
print(inner_result1)

v5 = np.array([[1+2j],[1]])
v6 = np.array([[1-1j],[1]])
inner_result2 = np.vdot(v5,v6)
print(inner_result2)
```

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 4

4.1)

From 4.18 we know

$$|\vec{v_1}\rangle = \frac{|\vec{v}\rangle}{\sqrt{\sum_{i=0}^{n-1} |a_i|^2}}$$

We can derive numerators of above co-efficients

$$|\vec{v_1}\rangle = 2|1\rangle + |1\rangle$$

$$= \left( \frac{2}{\sqrt{2^2+1}}, \frac{1}{\sqrt{2^2+1}} \right) = \left( \frac{2}{\sqrt{5}}, \frac{1}{\sqrt{5}} \right)$$

$$= (0.8944 \quad 0.4472)$$

4.3)

$$|\vec{v_1}\rangle = 2|1\rangle + (1+i)|\downarrow\rangle$$

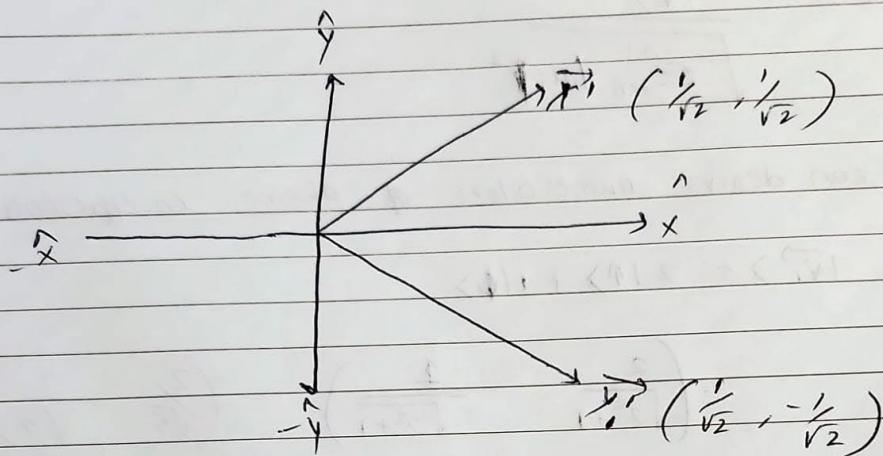
$$= \left( \frac{2}{\sqrt{2^2+(1+i)^2}}, \frac{1+i}{\sqrt{2^2+(1+i)^2}} \right) = \left( \frac{2}{\sqrt{4+2i}}, \frac{1+i}{\sqrt{4+2i}} \right)$$

$$= (0.816 + 0.4082i \quad 0.4082 + 0.4082i)$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

4.5) Given  $|\vec{x}\rangle = \frac{1}{\sqrt{2}}(|\hat{x}\rangle + |\hat{y}\rangle)$

$$|\vec{y}\rangle = \frac{1}{\sqrt{2}}(|\hat{x}\rangle - |\hat{y}\rangle)$$



$$\langle \vec{x} | \vec{x} \rangle = \left( \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}} \right) \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{2} + \frac{1}{2} = 1 \text{ (normalized)}$$

$$\langle \vec{y} | \vec{y} \rangle = \left( \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}} \right) \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{2} + \frac{1}{2} = 1 \text{ (normalized)}$$

$$\langle \vec{x} | \vec{y} \rangle = \left( \frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}} \right) \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{2} - \frac{1}{2} = 0 \text{ (orthogonal)}$$

$$\langle \vec{y} | \vec{x} \rangle = \left( \frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}} \right) \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} = \frac{1}{2} - \frac{1}{2} = 0 \text{ (orthogonal)}$$

It is satisfying both orthogonality and normalization

If satisfies orthonormality

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

```
#Problem 4
v1 = np.array([[2],[1]])
print(v1)
v2 = v1/np.linalg.norm(v1)
print(v2)

v3 = np.array([[2],[1+1j]])
print(v3)
v4 = v3/np.linalg.norm(v3)
print(v4)
```

```
[[2]
 [1]]
[[0.89442719]
 [0.4472136 ]]
[[2.+0.j]
 [1.+1.j]]
[[0.81649658+0.j          ]
 [0.40824829+0.40824829j]]
```

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 5

5.1) we know from 36 page.

$$|0\rangle = \frac{1}{\sqrt{2}} |+\rangle + \frac{1}{\sqrt{2}} |-\rangle \quad \& \quad |1\rangle = \frac{1}{\sqrt{2}} |+\rangle - \frac{1}{\sqrt{2}} |-\rangle \quad \xrightarrow{\textcircled{1}}$$

$$\text{we know } |+\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) \quad \& \quad |-\rangle = \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle) \quad \xrightarrow{\textcircled{2}}$$

Given  $|v_1\rangle = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$  in  $|+\rangle / |-\rangle$  basis

$$|\vec{v}_1\rangle = |+\rangle + 2|-\rangle$$

$$\text{we get } |0\rangle = \frac{1}{\sqrt{2}} (|+\rangle + 2|-\rangle) \quad |1\rangle = \frac{1}{\sqrt{2}} (|+\rangle - 2|-\rangle) \quad \xrightarrow{\textcircled{3}}$$

converting back by substituting ③ in ②

$$|+\rangle = \frac{1}{\sqrt{2}} \left( \frac{1}{\sqrt{2}} (2|+\rangle) \right) = 1 \quad \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$|-\rangle = \frac{1}{\sqrt{2}} \left( \frac{1}{\sqrt{2}} (4|-\rangle) \right) = 2$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

5.2)

$$\text{Given } |\vec{v}_1\rangle = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

Assumed it is in  $|0\rangle, |1\rangle$  basis

$$|\vec{v}_1\rangle = |10\rangle + |21\rangle$$

Converting it to  $|+\rangle, |-\rangle$  basis

$$= \left( \frac{1}{\sqrt{2}} |+\rangle + \frac{1}{\sqrt{2}} |-\rangle \right) + 2 \left( \frac{1}{\sqrt{2}} |+\rangle - \frac{1}{\sqrt{2}} |-\rangle \right)$$

$$= \frac{3}{\sqrt{2}} |+\rangle - \frac{1}{\sqrt{2}} |-\rangle$$

Normalized values are

$$= \begin{pmatrix} \frac{3}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{(3/\sqrt{2})^2 + (-1/\sqrt{2})^2}{\sqrt{2}} & \frac{(3/\sqrt{2})^2 + (-1/\sqrt{2})^2}{\sqrt{2}} \end{pmatrix}$$

$$= (0.948, -0.316)$$

Probability is positive as it can never be negative

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

5.3) Probability to find  $|1+\rangle$  is  $(0.948, 0.316)$  is 0.948

5.4) Probability to find  $|0\rangle$  in  $|0\rangle/|1\rangle$  basis

$$|\sqrt{1}\rangle = |10\rangle + 2|1-\rangle$$

$$= \left( \frac{1}{\sqrt{1+4}}, \frac{2}{\sqrt{1+4}} \right) = \left( \frac{1}{\sqrt{5}}, \frac{2}{\sqrt{5}} \right)$$

$$= (0.447, 0.894)$$

Probability of finding  $|0\rangle$  is 0.447

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

```
#Problem 5.2
v1 = np.array([[3/np.sqrt(2)],[1/np.sqrt(2)]])
print(v1)
v2 = v1/np.linalg.norm(v1)
print(v2)

# Problem 5.4
v3 = np.array([[1] , [(2 )]])
print(v3)
v4 = v3/np.linalg.norm(v3)
print(v4)
```

```
[[2.12132034]
 [0.70710678]]
[[0.9486833 ]
 [0.31622777]]
[[1]
 [2]]
[[0.4472136 ]
 [0.89442719]]
```

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 6

6.1  
sol.

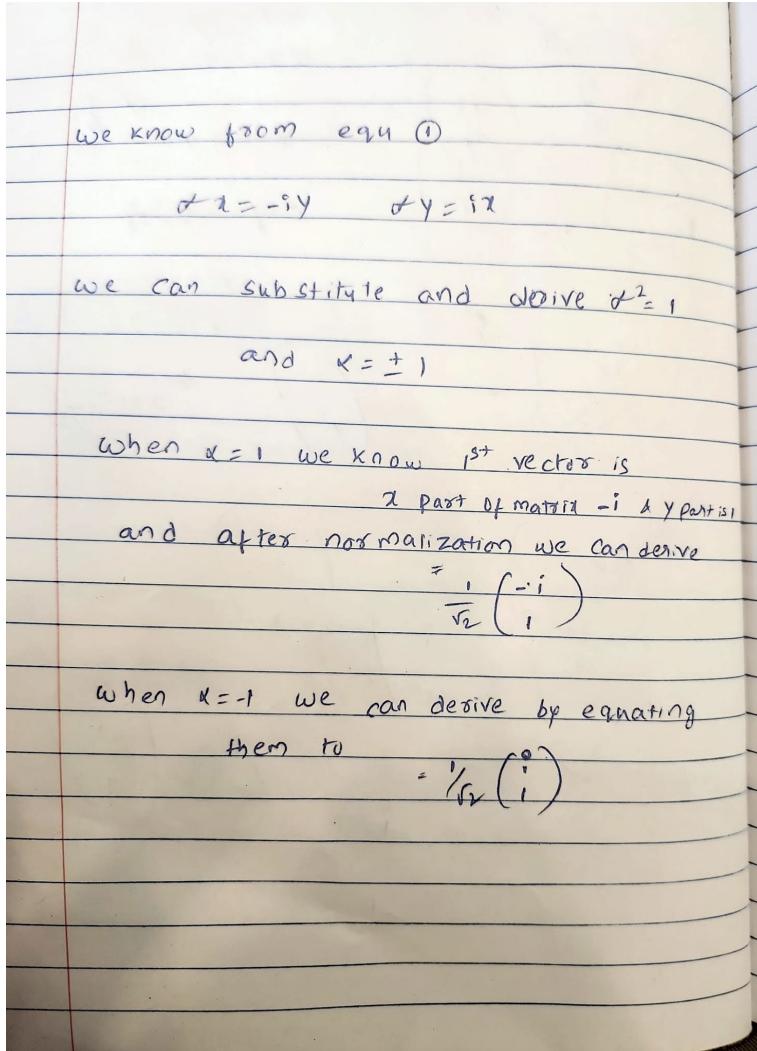
Given operator  $\sigma_y$  matrix

$$\sigma_y \text{ |eigen vector } \rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$
$$= \begin{pmatrix} -iy \\ ix \end{pmatrix}$$

we can assume eigen value as  $\alpha$

$$\begin{pmatrix} -iy \\ ix \end{pmatrix} = \alpha \begin{pmatrix} x \\ y \end{pmatrix} \quad \rightarrow \textcircled{1}$$
$$\alpha x = -iy \quad \alpha y = ix$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



```
#Lecture_6-Problems
import numpy as np
sigmay = np.array([[0, -1j], [1j, 0]])
print(sigmay)
eigenvalues,eigenvectors = np.linalg.eig(sigmay)
print(eigenvalues)
print(eigenvectors)

[[ 0.+0.j -0.-1.j]
 [ 0.+1.j  0.+0.j]
 [ 1.+0.j -1.+0.j]
 [[-0.         -0.70710678j  0.70710678+0.j
 [ 0.70710678+0.j       0.         -0.70710678j]]]
```

## Chapter 7

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

problems

$$7.1) \quad \sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\text{To find. } \sigma_y^2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 0 - i^2 & 0 + 0 \\ 0 + 0 & -i^2 + 0 \end{pmatrix} \quad (\because i^2 = -1)$$

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

$$\sigma_y^2 = I$$

$$7.2) \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\sigma_z^2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$= \begin{pmatrix} 1 * 1 + 0 & 0 + 0 \\ 0 + 0 & 0 + (-1) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I$$

$$\sigma_z^2 = I$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

7.3)  $\sigma_y$  is Hermitian

we know  $\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$

Firstly Conjugation

$$\sigma_y^* = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}$$

To transpose

$$\sigma_y^{*T} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = \sigma_y^{*T}$$

7.4)  $\langle 0 | \sigma_y$

$$(1 \ 0) \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = \begin{bmatrix} ix_0 + 0x_1 & ix_1 - i + 0x_0 \end{bmatrix}$$

$$\begin{pmatrix} 0 & -i \end{pmatrix}$$

It is in bra form and it is equal to  $\langle 0 | \sigma_y | 0 \rangle$  only if we do transpose and complex conjugate to  $(\langle 0 | \sigma_y)^*$  i.e.,  $(\langle 0 | \sigma_y)^T* = \langle 0 | \sigma_y | 0 \rangle$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

```
#Lecture_7-Problems
#7.1
sigmay=np.array([[0,-1j],[1j,0]])
print(sigmay)
print(np.matmul(sigmay, sigmay))

#7.2
sigmaz=np.array([[1,0],[0,-1]])
print(sigmaz)
print(np.matmul(sigmaz, sigmaz))

#7.3

sigmayh = np.conjugate(np.transpose(sigmay))
print(sigmayh)

#7.4
zerospin=np.array([[1],[0]])
print(zerospin.shape)
print(np.dot(sigmay,zerospin))
```

---

```
↳ [[ 0.+0.j -0.-1.j]
     [ 0.+1.j  0.+0.j]]
[[[1.+0.j 0.+0.j]
  [0.+0.j 1.+0.j]]
 [[ 1  0]
  [ 0 -1]]
[[[1 0]
  [0 1]]
[[ 0.-0.j  0.-1.j]
  [-0.+1.j  0.-0.j]]
(2, 1)
[[[0.+0.j]
  [0.+1.j]]]
```

## Chapter 8

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

### #Lecture\_8-Problems

```
import numpy as np
sigmax=np.array([[0,1],[1,0]])
print(sigmax)
eigenvalues, eigenvectors=np.linalg.eig(sigmax)
print(eigenvalues)
print(eigenvectors)

sigmaxp=np.array([1/2,1/2])
sigmaxn=np.array([1/2,-1/2])
eigenvalues, eigenvectors=np.linalg.eig(sigmax)
print(eigenvalues)
print(eigenvectors)
print(np.dot(sigmaxp,sigmaxp))
print(np.dot(sigmaxn,sigmaxn))
```

```
[[ 0.+0.j -0.-1.j]
 [ 0.+1.j  0.+0.j]]
[[1.+0.j 0.+0.j]
 [0.+0.j 1.+0.j]]
[[ 1  0]
 [ 0 -1]]
[[1 0]
 [0 1]]
[[ 0.-0.j  0.-1.j]
 [-0.+1.j  0.-0.j]]
(2, 1)
[[0.+0.j]
 [0.+1.j]]
```

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

8.2) Given  $\sigma_z$

we know  $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$  it is ket notation

and even if find adjoint to get bra it is same  
so it is Hermitian matrix

we know project operator is

$$P_{\sigma_z} = |\sigma_z\rangle \langle \sigma_z|$$

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

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

$$I = P_{\sigma_z}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

E.3)

Given

eigen vectors of  $\sigma_x$

$$|+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad |- \rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

we know for probability

$$P_i = |i\rangle \langle i|$$

probability of  $|+\rangle$  ~~is~~  $\Rightarrow$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \end{pmatrix}$$

$$= \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

so it is 50% & similarly for  $|-\rangle$

is

$$= \frac{1}{2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

```
#8.4
sigmaxp=np.array([1/np.sqrt(2),1/np.sqrt(2)])
sigmaxn=np.array([1/np.sqrt(2),-1/np.sqrt(2)])
eigenvalues, eigenvectors=np.linalg.eig(sigmax)
print(eigenvalues)
print(eigenvectors)
print(np.dot(sigmaxp,sigmaxp))
print(np.dot(sigmaxn,sigmaxn))
```

result

```
0.9999999999999998
0.9999999999999998
```

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

Q.1 sol

we know from eq 9.15

$$\text{that } U^\dagger U = I$$

we know from page no 24 we can use

$$\text{eq 4.1 } \hat{v}_i \cdot \hat{v}_j = \begin{cases} 1 & \text{if } i=j \\ 0 & \text{if } i \neq j \end{cases}$$

so when we consider a row in 9.15

we can say if it is in same row like

example  $b_{0,0}$  is 0<sup>th</sup> row it is 1 etc

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

q.2) Given q.2 used it in (i) (ii) (iii)

$$\begin{vmatrix} a_{0,0} - \lambda_i & a_{0,1} \\ a_{1,0} & a_{1,1} - \lambda_i \end{vmatrix}$$

i)  $\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

$$= \begin{vmatrix} 0 - \lambda_i & 1 \\ 1 & 0 - \lambda_i \end{vmatrix} = 0 \quad +\lambda_i^2 - 1 = 0$$

$\lambda_i^2 = \pm 1$

ii)  $\sigma_y = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix}$

$$= \begin{vmatrix} 0 - \lambda_i & i \\ -i & 0 - \lambda_i \end{vmatrix} = 0 \Rightarrow \lambda_i^2 - (i^2) = 0$$

$$\lambda^2 = -1$$

$\lambda = \pm 1$

iii)  $\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$

$$= \begin{vmatrix} 1 - \lambda_i & 0 \\ 0 & -1 - \lambda_i \end{vmatrix} = 0 \Rightarrow -(1 - \lambda^2) = 0$$

$$\lambda^2 = 1$$

$\lambda = \pm 1$

Eigen values are same for all

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



```
#Lecture_9-Problems
#9.3
# M=np.array([[1,0],[0,-1]])
sigmax=np.array([[0,1],[1,0]])
sigmay=np.array([[0,-1j],[1j,0]])
sigmaz=np.array([[1,0],[0,-1]])

print(np.linalg.det(sigmax))
print(np.linalg.det(sigmay))
print(np.linalg.det(sigmaz))
```

```
↳ -1.0
(-1+0j)
-1.0
```

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

9.4)

for example we can take

$$A \text{ as } \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \text{ i.e., } \sigma_x$$

We know pauli matrices are unitary  
we can prove it now

$$|v_i\rangle = \frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle$$

$$|v_i\rangle = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix}$$

$$A|v_i\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$$

We can prove orthonormality by

Adding squares of coefficient of transformed  
 $A|v_i\rangle$  and inner product with other matrices will be zero

$$\left(-\frac{1}{\sqrt{2}}\right)^2 + \left(\frac{1}{\sqrt{2}}\right)^2 = 1$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 10

(10.1)

Transformation matrix from basis  $\sigma_y$  to  $\sigma_z$

We know from problem 6.1

that eigen vectors of  $\sigma_y$  are  $\begin{pmatrix} 1 \\ -i \end{pmatrix}$ ,  $\begin{pmatrix} i \\ 1 \end{pmatrix}$

$$|+\rangle_{\sigma_y} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad |-\rangle_{\sigma_y} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$u = \langle \text{new}, y | \text{old}, x \rangle = \begin{bmatrix} \frac{1}{2} (-i, 1)^* \begin{pmatrix} 1 \\ 1 \end{pmatrix} & \frac{1}{2} (-i, 1)^* \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\ \frac{1}{2} (i, 1)^* \begin{pmatrix} 1 \\ 1 \end{pmatrix} & \frac{1}{2} (i, 1)^* \begin{pmatrix} 1 \\ -1 \end{pmatrix} \end{bmatrix}$$

$$= \left[ \frac{1}{2} \begin{bmatrix} i+1 & i-1 \\ -i+1 & -i-1 \end{bmatrix} \right]$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

10.2) Given in Page 87

complete basis is defined by outer product  
of each vectors sums to Identity matrix  
Idea outer product is nothing but Ket  $\times$  bra

$$\sum_{i=0}^{n-1} |i\rangle \langle i| = |-\rangle \langle -| + |+\rangle \langle +| \\ \text{By deriving, } \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I$$

10.3) For  $\sigma_z$  we know

$$|-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad |+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

We know from 10.19

$$\sigma_z = \sum_{i=0}^{2-1} \lambda_i |i\rangle \langle i| \\ = 1 \left( \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right) \left( \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right)^* + -1 \left( \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right) \left( \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right)^*$$

$$= \left( \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \right) - \left( \frac{1}{2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \right)$$

$$\sigma_z = \frac{1}{2} \begin{pmatrix} 0 & 2 \\ 2 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

## #Lecture\_10-Problems

### #10.4

```
import numpy as np
sigmaxn=np.array([1/np.sqrt(2),-1])
sigmaxp=np.array([1/np.sqrt(2),1])
sigmayp=np.array([1/np.sqrt(2),+1j])
sigmayn=np.array([1/np.sqrt(2),-1j])

print(np.matmul(sigmaxn, sigmayn))
print(np.matmul(sigmaxn, sigmayp))
print(np.matmul(sigmaxp, sigmayn))
print(np.matmul(sigmaxp, sigmayp))
```

```
(0.4999999999999999+1j)
(0.4999999999999999-1j)
(0.4999999999999999-1j)
(0.4999999999999999+1j)
```

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 11

11.1) Given  $|b\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$   $|g\rangle_2 = \begin{pmatrix} i/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$

From 11.11 & 11.14

we know

$$\begin{pmatrix} \alpha_3 \\ \beta_3 \\ \gamma_3 \\ \delta_3 \end{pmatrix} = \begin{pmatrix} \alpha_1 \alpha_2 \\ \alpha_1 \beta_2 \\ \beta_1 \alpha_2 \\ \beta_1 \beta_2 \end{pmatrix}$$

i)  $|f\rangle_1 \otimes |g\rangle_2$

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} i/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ i/\sqrt{2} \\ i/\sqrt{2} \end{pmatrix}$$

ii)  $|g\rangle_1 \otimes |f\rangle_2$

$$\begin{pmatrix} i/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ i/\sqrt{2} \\ 0 \\ i/\sqrt{2} \end{pmatrix}$$

Both are not same

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$11.2) \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ 0 \\ \frac{1}{\sqrt{2}} \\ \frac{i}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$\left( \begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \frac{1}{\sqrt{2}} \\ 0 \\ \frac{i}{\sqrt{2}} \end{array} \right)$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

11.3) we need to prove distribution rule

$$11.20 \quad (|f\rangle + |e\rangle) \otimes |g\rangle = |f\rangle \otimes |g\rangle + |e\rangle \otimes |g\rangle$$

$$\begin{aligned} & \left( \begin{pmatrix} 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right) \otimes \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \\ &= \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \end{aligned}$$

$$\begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} + \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \\ 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \quad L.H.S = R.H.S$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

### 11.4)

```
▶ #Lecture_11-Problems

#11.4
#proving 11.1
import numpy as np
g=np.array([[0],[1]])
h=np.array([[1/np.sqrt(2)],[1j/np.sqrt(2)]])
print(np.kron(g,h))
print(np.kron(h,g))

# proving 11.2
x = np.kron(h,g)
print(np.kron(g,x))

#proving 11.3
y=np.array([[1],[0]])
z = (g + y)
print(np.kron(z,h))
a = np.kron(z,h)
```

```
l = np.kron(g,h)
m = np.kron(y,h)
b = l+m
print(l+m)

if a.all() == b.all() :
    print(" L.H.S == R.H.S")
```

```
[[0.          +0.j        ]
 [0.          +0.j        ]
 [0.70710678+0.j      ]
 [0.          +0.70710678j]]
 [[0.          +0.j        ]
 [0.70710678+0.j      ]
 [0.          +0.j        ]
 [0.          +0.70710678j]]
 [[0.          +0.j        ]
 [0.          +0.j        ]
 [0.          +0.j        ]
 [0.          +0.j        ]
 [0.70710678+0.j      ]
 [0.          +0.j        ]
 [0.          +0.70710678j]]
 [[0.70710678+0.j      ]
 [0.          +0.70710678j]
 [0.70710678+0.j      ]
 [0.          +0.70710678j]]
 [[0.70710678+0.j      ]
 [0.          +0.70710678j]
 [0.70710678+0.j      ]
 [0.          +0.70710678j]]
 L.H.S == R.H.S
```

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

(1.5)

$$\text{Given } n_1 = n_2 = 1 \quad \& \quad |f\rangle = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$$|e\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad |g\rangle = \begin{pmatrix} i \\ 1 \end{pmatrix} \quad \text{and} \quad |k\rangle = \begin{pmatrix} i \\ 1 \end{pmatrix}$$

eqn 1.21

$$\langle h_1 | h_2 \rangle = \left\langle \sum_{j=1}^{n_1} |f_j\rangle \otimes |g_j\rangle \middle| \sum_{l=1}^{n_2} |e_l\rangle \otimes |k_l\rangle \right\rangle$$

$$= \sum_{j=1}^{n_1} \sum_{l=1}^{n_2} \langle f_j | e_l \rangle \langle g_j | k_l \rangle$$

$$\stackrel{\because n_1 = n_2 = 1}{=} \left\langle \begin{pmatrix} 1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} i \\ 1 \end{pmatrix} \middle| \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} i \\ 1 \end{pmatrix} \right\rangle$$

$$= (1|i; i|1) \begin{pmatrix} 0 \\ 0 \\ i \\ i \end{pmatrix} \stackrel{(\because \text{It is inner product})}{=} i - 1$$

R.H.S

$$= (1|1) \begin{pmatrix} 0 \\ 1 \end{pmatrix} \cdot (1|i) \begin{pmatrix} i \\ 1 \end{pmatrix}$$

$$= (0+1) (i-1)$$

$$= i - 1$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 12

12.1

so)

Given  $|f\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$  and  $|g\rangle = \begin{pmatrix} i/\sqrt{2} \\ i/\sqrt{2} \end{pmatrix}$

Initially

$\sigma_z |t_0\rangle,$

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

$$\sigma_z |g\rangle_2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} i/\sqrt{2} \\ -i/\sqrt{2} \end{pmatrix} = \begin{pmatrix} i/\sqrt{2} \\ i/\sqrt{2} \end{pmatrix}$$

$\sigma_z |f\rangle_1 \otimes \sigma_z |g\rangle_2$

$$= \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} i/\sqrt{2} \\ i/\sqrt{2} \end{pmatrix} = \begin{pmatrix} i/\sqrt{2} \\ i/\sqrt{2} \\ 0 \\ 0 \end{pmatrix}$$

Now to find  $|f\rangle_1 \otimes |g\rangle_2$

$$= \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} i/\sqrt{2} \\ i/\sqrt{2} \end{pmatrix} \Rightarrow \begin{pmatrix} 0 \\ 0 \\ i/\sqrt{2} \\ i/\sqrt{2} \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

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

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

$$\sigma_x \otimes \sigma_x (|1\rangle \otimes |g\rangle_2)$$

$$\textcircled{0} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1/\sqrt{2} \\ 1/\sqrt{2} \end{bmatrix}$$

$$= \begin{bmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \\ 0 \end{bmatrix}$$

So it is equal to  $\sigma_x |1\rangle_1 \otimes \sigma_x |g\rangle_2$

$$\sigma_x |1\rangle_1 \otimes \sigma_x |g\rangle_2 = \sigma_x \otimes \sigma_x (|1\rangle_1 \otimes |g\rangle_2)$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 13

### 13.1

As quantum computing is known for exponential speed and for quantum parallelism world can store up to  $2^{10^{21}}$  quantum states due to superposition.

### 13.2)

13.2)

Here  $|\phi^+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$  is proved already

$$|\phi^-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 0 \\ 0 \\ -1 \end{pmatrix}$$

$$= \frac{1}{\sqrt{2}} (|100\rangle + |010\rangle + |001\rangle - |111\rangle)$$

we can assume  $|\phi^+\rangle = |m\rangle \otimes |n\rangle$

$$|m\rangle = a_0|0\rangle + a_1|1\rangle \quad \& \quad |n\rangle = b_0|0\rangle + b_1|1\rangle$$

Given 13.11  $\rightarrow |\phi^+\rangle = a_0b_0|100\rangle + a_0b_1|101\rangle + a_1b_0|010\rangle + a_1b_1|111\rangle$

Similar to 13.12  $a_0b_0=1, a_0b_1=0, a_1b_0=0, a_1b_1=-1$

Similarly  $a_0b_0=0$  (or)  $a_1b_1=0$  should be here

So  $|\phi^+\rangle$  is entangled as our assumption is false

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$ \Psi^+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}$ $= \frac{1}{\sqrt{2}} ( 100\rangle +  101\rangle +  110\rangle +  111\rangle)$	$ \Psi^-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 0 \\ 1 \\ -1 \\ 0 \end{pmatrix}$ $= \frac{1}{\sqrt{2}} ( 0100\rangle +  101\rangle -  110\rangle +  011\rangle)$
<p style="color: blue; font-style: italic;">From 13.2</p> $a_0 b_0 = 0, a_0 b_1 = 1$ $a_1 b_0 = 0, a_1 b_1 = 0$	<p style="color: blue; font-style: italic;">From 13.2</p> $a_0 b_0 = 0, a_0 b_1 = 1, a_1 b_0 = 0$ $a_1 b_1 = 0$

we can assume  
 $a_0 b_0 = 0$  either of  
 them should be zero  
 it denies our above

we can assume  $a_0 b_0 = 0$   
 and one of them should be  
 zero it denies calculation

13.3) Yes, I believe entanglement is concept of distance we can expect the result.  
 It will be profitting for him

### 13.4)

Here as mentioned basis states are orthonormal and if we have the same state in both two spaces then the inner product will be 1

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 14

### 14.1

14.2

Given  $\frac{1}{2} (|00\rangle + |01\rangle + |10\rangle + |11\rangle)$

From 14.12  $\Rightarrow \frac{1}{2} |0\rangle\langle 0| + \frac{1}{2} |0\rangle\langle 1|$

$$+ \frac{1}{2} |1\rangle\langle 0| + \frac{1}{2} |1\rangle\langle 1|$$

$$\rho = \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} (|0\rangle\langle 0|) +$$

$$+ \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} (|0\rangle\langle 1|) + \frac{1}{2} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} (|1\rangle\langle 0|)$$

$$+ \frac{1}{2} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} (|1\rangle\langle 1|)$$

$$= \frac{1}{2} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \frac{1}{2} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \frac{1}{2} \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$$

$$+ \frac{1}{2} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\rho = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

As we know entanglement entropy is zero

from previous chapter that we know our given

state is not product of two subspaces

and it should be zero

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

14.2)

Given  $\rho = \frac{1}{2} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$   $\rightarrow 14.15$

- \* We know  $\rho_B = \text{Tr}_A(\rho)$

Figure 14.2 for tracing second qubit

$$= \begin{pmatrix} (0) & 0 & (0) & 0 \\ 0 & (1/2) & -1/2 & 0 \\ (0) & -1/2 & (1/2) & 0 \\ 0 & 0 & 0 & (0) \end{pmatrix}$$

Above same shape elements can be added

From above tracing B space

$$= \begin{bmatrix} 0 & x & 0 & x \\ x & 1/2 & x & 0 \\ 0 & x & 1/2 & x \\ x & 0 & x & 0 \end{bmatrix}$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 15

15.1)

we know

from 15.6

$$U_{NOT} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

To prove it as unitary we know

$$U^\dagger \cdot U = I$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \text{ should be } I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

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

Hence proved

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

15.2)

we know from 15.15

$$\text{CNot is } U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

from 9.18 we know

$$U^\dagger U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

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

done proved

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

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

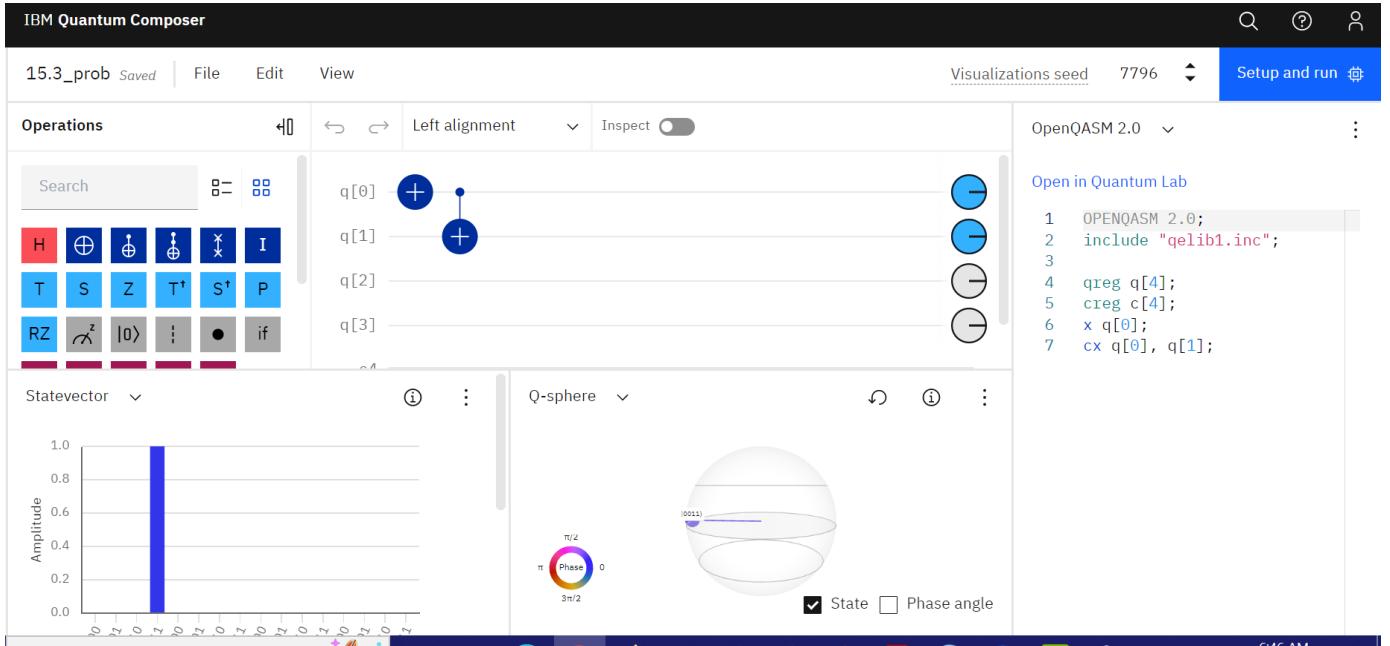
```
#Chapter 15

#15.1
import numpy as np
sigmay=np.array([[0,1],[1,0]])    #Not gate and transpose is same for this matrix
print(sigmay)
print(np.matmul(sigmay, sigmay))

#15.2
sigmac=np.array([[1,0,0,0],[0,1,0,0],[0,0,0,1],[0,0,1,0]])   #CNot gate and transpose is same for this matrix
print(sigmac)
print(np.matmul(sigmac, sigmac))
```

```
[[0 1]
 [1 0]
 [[1 0]
 [0 1]]
 [[1 0 0 0]
 [0 1 0 0]
 [0 0 0 1]
 [0 0 1 0]
 [[1 0 0 0]
 [0 1 0 0]
 [0 0 1 0]
 [0 0 0 1]]]
```

## 15.3)



## Chapter 16

### 16.1)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

16.1) we know Z-gate is  $\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

and  $|-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$  &  $|+\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

when we apply Z gate to  $|-\rangle$

$$Z|-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \times 1 + 0 \times -1 \\ 0 \times 1 - 1 \times -1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix} = |+\rangle$$

16.2)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

16.2)

We know control phase shift gate is

$$U_{CPS, \phi} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{bmatrix}$$

In Z-axis when angle is  $\pi$  we

$$\text{can assume } e^{i\phi} = \cos\phi + i\sin\phi$$

$$= \cos\pi + i\sin\pi$$

$$e^{i\phi} = -1$$

controlled Z gate is

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

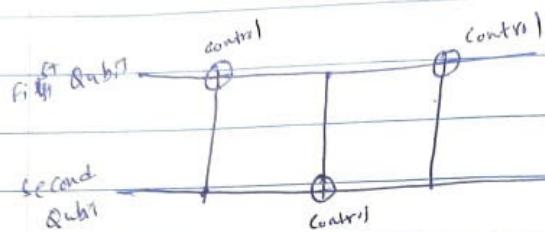
when Cz applied to  $|+\rangle|+\rangle = |+\rangle\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$  it becomes  $\frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

It is CNOT gate in  $|+\rangle|-\rangle$  basis

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

16.3) Here I have analyzed 3 CNOT gates usage building swap gate



Here 1<sup>st</sup> qubit converts (First Not gate)

1<sup>st</sup> CNot gate  $CNot_1 (|00\rangle = |00\rangle, |01\rangle = |10\rangle, |10\rangle = |11\rangle)$ ,

control is with 2<sup>nd</sup> qubit

2<sup>nd</sup> CNot gate  $CNot_2 (|00\rangle = |00\rangle, |01\rangle = |11\rangle, |10\rangle = |10\rangle)$ ,

control is with 1<sup>st</sup> qubit

3<sup>rd</sup> CNot gate  $CNot_3 (|00\rangle = |00\rangle, |11\rangle = |10\rangle, |10\rangle = |01\rangle)$ ,

we can see swap is done and it is proved

For CNot 2 using state representation we can derive

$$CNot_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \begin{array}{l} |00\rangle \\ |11\rangle \\ |10\rangle \\ |10\rangle \end{array}$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 17

### 17.1)

17.1

Given

$$H|\Psi\rangle = \alpha|+\rangle + \beta|-> \quad \text{From 17.4}$$

Normally we can directly understand that

co-efficient of  $|0\rangle$  &  $|1\rangle$  basis is from 17.4

$\alpha/\sqrt{2}(\alpha+\beta)$  &  $\beta/\sqrt{2}(\alpha-\beta)$  from understanding

we can even derive from 17.4

$$= \alpha|+\rangle + \beta|->$$

$$\text{we know } |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \quad |-> = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

$$= \alpha\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right) + \beta\left(\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\right)$$

$$= \frac{\alpha}{\sqrt{2}}|0\rangle + \frac{\alpha}{\sqrt{2}}|1\rangle + \frac{\beta}{\sqrt{2}}|0\rangle - \frac{\beta}{\sqrt{2}}|1\rangle$$

$$H|\Psi\rangle = \frac{1}{\sqrt{2}}(\alpha+\beta)|0\rangle + \frac{1}{\sqrt{2}}(\alpha-\beta)|1\rangle$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

17.2)

17.2)

$$\text{Given } H^{\otimes n}|y\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{x \cdot y} |x\rangle$$

Here  $n$  means  $n$  qubit Hadamard gate

$$H^{\otimes 2}|12\rangle = \frac{1}{\sqrt{2^2}} \sum_{x=0}^{2^2-1} (-1)^{x \cdot 2} |x\rangle$$

$$= \frac{1}{2} \left( (-1)^{0 \cdot 2} |00\rangle + (-1)^{1 \cdot 2} |10\rangle + (-1)^{2 \cdot 2} |12\rangle + (-1)^{3 \cdot 2} |13\rangle \right)$$

From 17.3 we have

$$0 \cdot 2 = (0 \text{AND} 1) \oplus (0 \text{AND} 0) = 0 \oplus 0 = 0$$

$$1 \cdot 2 = (0 \text{AND} 1) \oplus (1 \text{AND} 0) = 0 \oplus 0 = 0$$

$$2 \cdot 2 = (1 \text{AND} 1) \oplus (0 \text{AND} 0) = 1 \oplus 0 = 1$$

$$3 \cdot 2 = (1 \text{AND} 1) \oplus (1 \text{AND} 0) = 1 \oplus 0 = 1$$

$$= H^{\otimes 2}|12\rangle = \frac{1}{2} (|00\rangle + |10\rangle - |12\rangle - |13\rangle)$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

17.3)

17.3)

From 17.2 Example using 17.19

$$H^{\otimes 2} |12\rangle = H^{\otimes 2} |10\rangle$$

$$= (H|11\rangle) \otimes (H|10\rangle)$$

$$= \frac{1}{2^{2/2}} (|10\rangle + (-1)^1 |11\rangle) \otimes (|10\rangle + (-1)^0 |11\rangle)$$

$$= \frac{1}{2} (|10\rangle - |11\rangle) \otimes (|10\rangle + |11\rangle)$$

$$= \frac{1}{2} (|100\rangle + |101\rangle - |110\rangle - |111\rangle)$$

Here we proved same solution we achieved  
in previous problems

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

17.4)

17.4

From 17.10 we know

$$H^{\otimes 2} = H \otimes H$$

$$= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

We need to find  $H^{\otimes 2}|12\rangle$

We know  $|12\rangle = |10\rangle = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$

$$H^{\otimes 2}|10\rangle = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

$$= \frac{1}{2} \begin{pmatrix} 1 \times 0 + 1 \times 0 + 1 \times 1 + 1 \times 0 \\ 1 \times 0 - 1 \times 0 + 1 \times 1 + 1 \times 0 \\ 1 \times 0 + 1 \times 0 - 1 \times 1 - 1 \times 0 \\ 1 \times 0 - 1 \times 0 - 1 \times 1 + 1 \times 0 \end{pmatrix}$$

$$= \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ -1 \\ -1 \end{pmatrix}$$

$$= \frac{1}{2} (|100\rangle + |101\rangle - |110\rangle - |111\rangle)$$

Hence proved

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 18

### 18.1)

18.1)

$$|\psi\rangle = \alpha|100\rangle + \beta|101\rangle + \gamma|110\rangle + \delta|111\rangle$$

$$|\phi_1\rangle = \alpha|100\rangle + \beta|101\rangle + \gamma|110\rangle + \delta|101\rangle$$

$$|\phi_2\rangle = \alpha|100\rangle + \beta|110\rangle + \gamma|111\rangle + \delta|101\rangle$$

LSB

MSB

$$|\phi\rangle = \alpha|100\rangle + \beta|110\rangle + \gamma|101\rangle + \delta|111\rangle$$

So it does not have any effect even if we implement CNOT gates starting from LSB or MSB

$U_{x\oplus}$ , right,  $U_{x\oplus}$ , center,  $U_{x\oplus}$ , left

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

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

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

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

$U_{swap}$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

18.2)

18.2)

Given LSB as controlling bit from 15, 13  
we can derive

$$U_{XOR}|00\rangle = |0 \oplus 0, 0\rangle = |0, 0\rangle = |00\rangle$$

$$U_{XOR}|01\rangle = |0 \oplus 1, 0\rangle = |1, 0\rangle = |01\rangle$$

$$U_{XOR}|10\rangle = |1 \oplus 0, 0\rangle = |1, 0\rangle = |10\rangle$$

$$U_{XOR}|11\rangle = |1 \oplus 1, 1\rangle = |0, 1\rangle = |01\rangle$$

we can derive matrix as

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

It matches our assumed value in previous problem

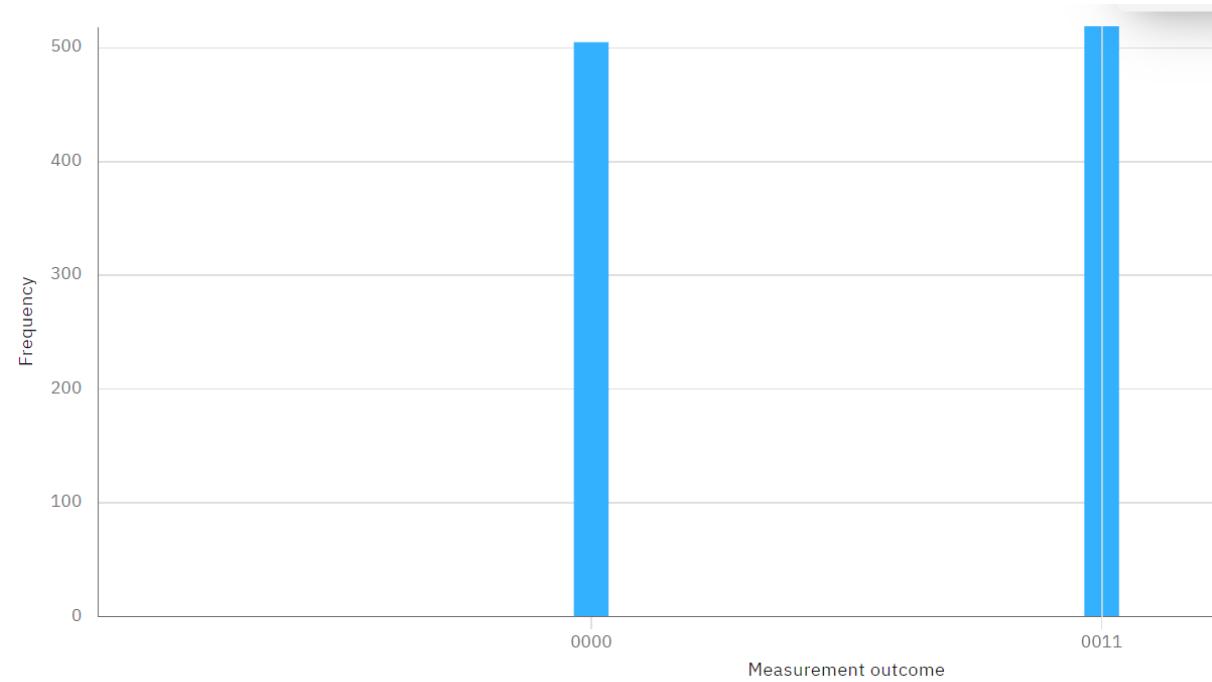
# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

18.3)

The screenshot shows the IBM Quantum Composer interface. At the top, it says "IBM Quantum Composer" and "18.3Exercise". Below the toolbar, there's a "Operations" panel with various quantum gates like RZ, RX, RY, RXX, RZZ, U, RCCX, and RC3X. The main workspace displays a quantum circuit with two qubits (q[0] and q[1]) and one classical register (c4). The circuit starts with an identity gate (I) on q[0], followed by a CNOT gate from q[0] to q[1], and ends with a measurement gate on q[1] to c4. To the right of the circuit, there are "Probabilities" and "Q-sphere" visualization tools. The "Probabilities" tool shows a bar chart of computational basis state frequencies. The "Q-sphere" tool shows a 3D sphere with points representing quantum states. On the right side of the interface, there's a "Visualizations seed" field (7796), a "Setup and run" button, and a "OpenQASM 2.0" code editor containing the following QASM code:

```
OPENQASM 2.0;
include "qelib1.inc";
qreg q[2];
creg c[4];
id q[0];
h q[1];
cx q[1], q[0];
measure q[0] -> c[0];
measure q[1] -> c[1];
```

## Simulation output



# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

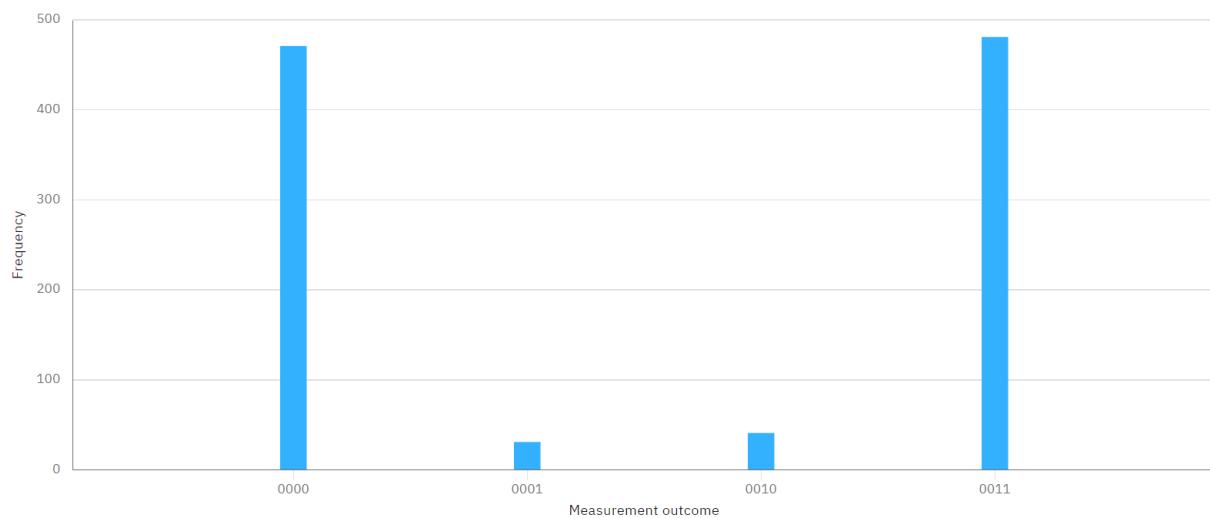
## Hardware execution result

### Details

36m 30.5s Total completion time	Sent from	<a href="#">Untitled circuit</a>
ibmq_manila Backend	Created on	Sep 07, 2022 11:50 AM
	Sent to queue	Sep 07, 2022 11:50 AM
	Provider	ibm-q/open/main
	Run mode	fairshare
	# of shots	1024
	# of circuits	1

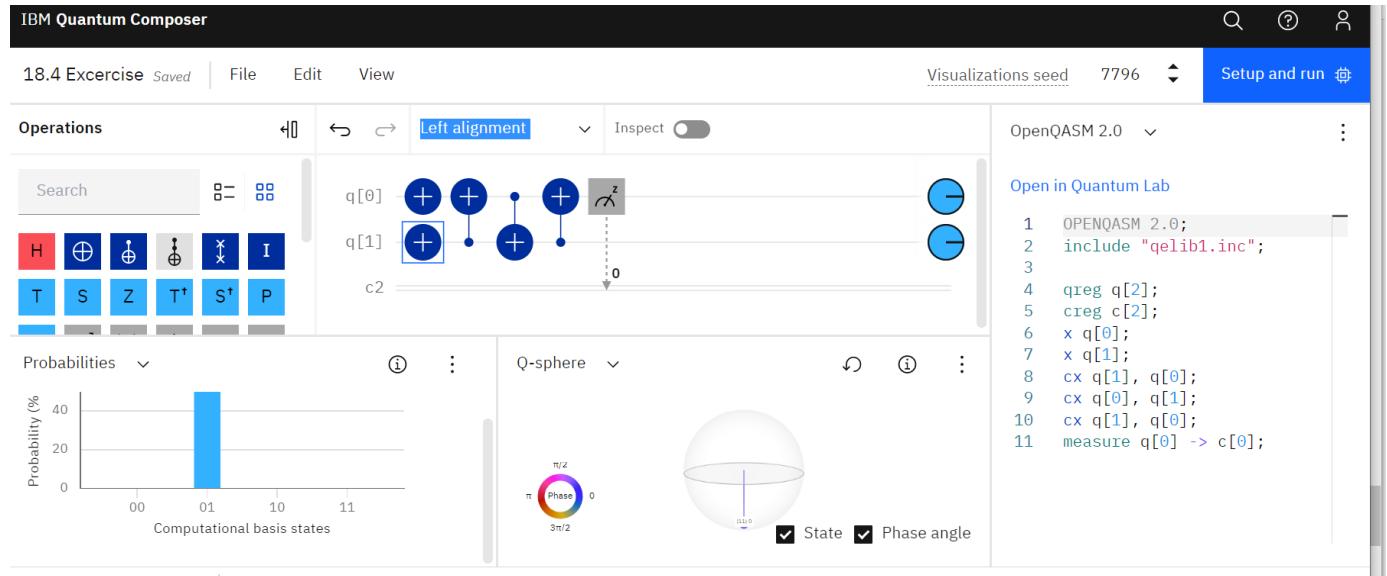
Status Timeline

- Created: Sep 07, 2022 11:50 AM
- Transpiling: 1.3s
- Validating: 1.1s
- In queue: 35m 51.6s
- Running: 4.9s  
time in system 4.9s
- Completed: Sep 07, 2022 12:26 PM



# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

18.4)



## Simulation



## Hardware execution result

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Details

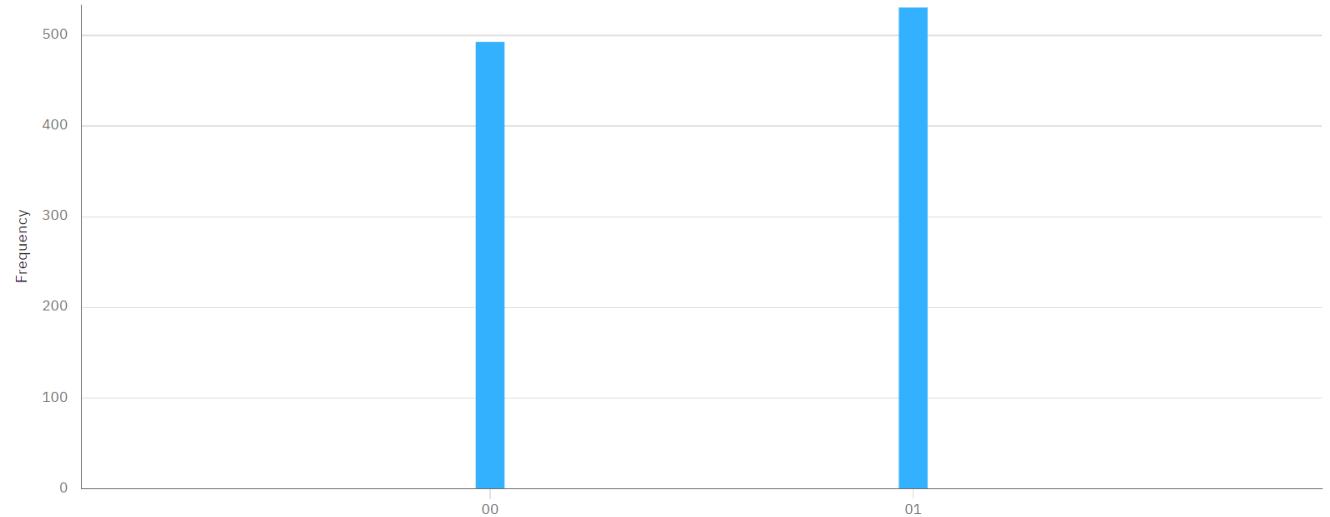
21h 2m 51.4s  
Total completion time

ibm\_oslo  
Backend

Sent from [18.4 Excercise](#)  
Created on Sep 07, 2022 12:14 PM  
Sent to queue Sep 07, 2022 12:14 PM  
Provider ibm-q/open/main  
Run mode fairshare  
# of shots 1024  
# of circuits 1

## Status Timeline

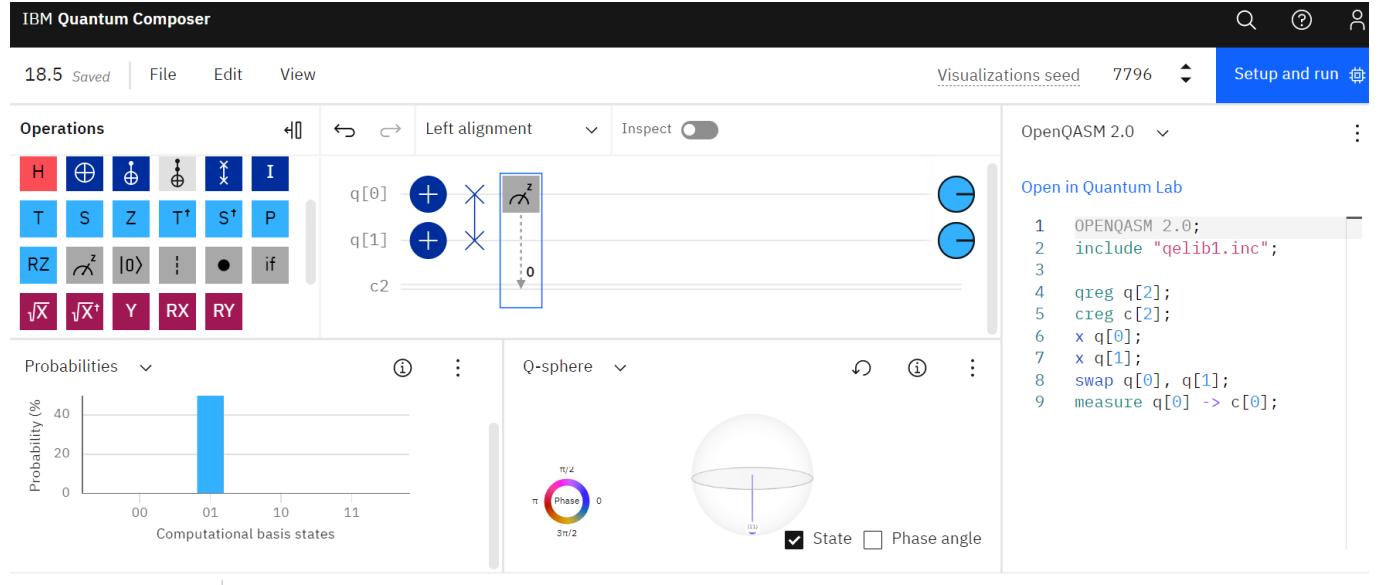
- ✓ Created: Sep 07, 2022 12:14 PM
- ✓ Transpiling: 1s
- ✓ Validating: 1.1s
- ✓ In queue: 21h 2m 41.1s
- ✓ Running: 4.4s  
time in system 4.4s
- ✓ Completed: Sep 08, 2022 9:16 AM



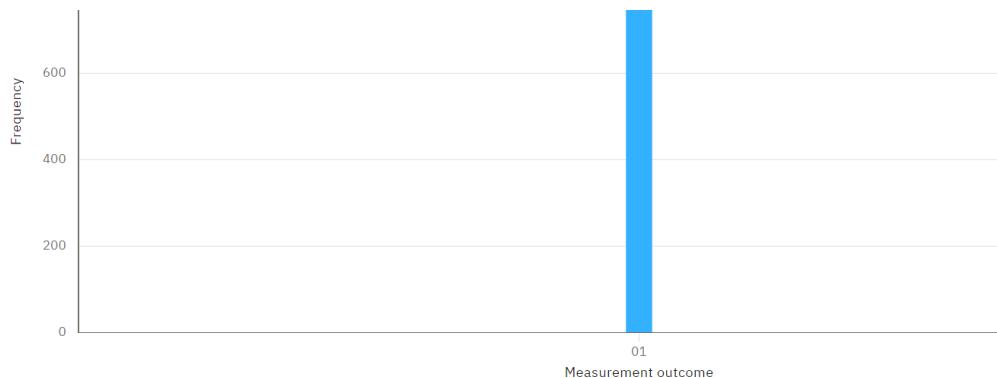
18.5)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

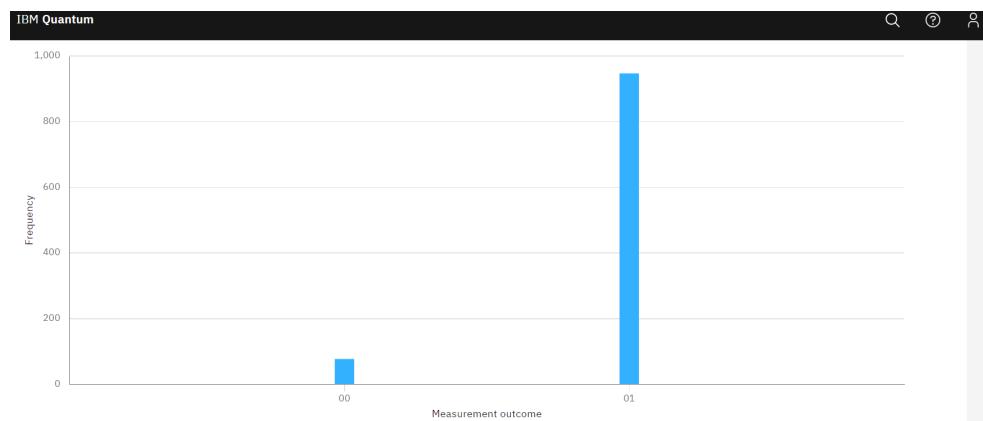
# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



## Simulation results



## Hardware execution result



## Chapter 19 19.1)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

19.1

Given

$$L \left( \frac{1}{\sqrt{2}} (|a\rangle_A + |b\rangle_A) |0\rangle_B \right) = \frac{1}{\sqrt{2}} (|a\rangle_A + |b\rangle_A) \\ + \frac{1}{\sqrt{2}} (|a\rangle_B + |b\rangle_B)$$

$$L(|a\rangle_A |0\rangle_B + |b\rangle_A |0\rangle_B)$$

$$\Rightarrow |a\rangle_A |a\rangle_B + |b\rangle_A |b\rangle_B \rightarrow L.H.S - \textcircled{1}$$

$\rightarrow R.H.S$  (Tensor product)

$$\Rightarrow |a\rangle_A |a\rangle_B + |a\rangle_A |b\rangle_B + |b\rangle_A |a\rangle_B + |b\rangle_A |b\rangle_B$$

$\textcircled{2}$

So here L.H.S  $\neq$  R.H.S because  $|a\rangle_A |b\rangle_B$   
 $\& |b\rangle_A |a\rangle_B$  probability is zero

Non cloning theorem survives on the concept  
of not able to copying linear combinations  
and linearity stands important

Ignored  $\frac{1}{\sqrt{2}}$  for simplification

19.2)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

Q. Here given  $|H\rangle$  as input we know  
 $|H\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \rightarrow D$

we know above from 17.1 problem that and as  
 we are supposed to be normalizing the  
 above it will work as general Quantum  
 teleportation algorithm.

$$b) \quad H_2 C_{10} H_1 = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$\text{from } 17.10 = \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{pmatrix}$$

3) AS it is  $|+\rangle$  base we are converting to general convention  
 and with linear combination of  $|0\rangle$  &  $|1\rangle$  is created with  
 first Hadamard gate

4) MSB and Hadamard gates are not cancelled

from  $|17.11\rangle$  we know it has equal probability of  
 all the four cases we can see same in output and  
 CNOT is for entanglement

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

19.3)

19.3) Yes, It is possible to clone basis state as Non-cloning theorem only limits copying of linear combination of Alice to Bob but when we measure Alice state we can clone as many times as we want as when we clone measure the value remains same

You can't clone arbitrary state due to linearity and we proved it 1<sup>st</sup> question

According to 19.7 we have to change basis and we can do teleportation but Alice won't be same after teleporting

Heisenberg uncertainty principle states that you can't calculate position of a things exactly at a time in two basis. we can't have exact basis states of two different basis at a time. we need to have linear combination of one state and base state of one state

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 20

### 20.1)

20.1)

$$= \frac{1}{\sqrt{2}} (\alpha |1000\rangle + \beta |101\rangle + \alpha |1011\rangle + \beta |110\rangle)$$

$$= \frac{1}{\sqrt{2}} \left( \alpha |10\rangle \otimes |10\rangle \otimes |10\rangle + \beta |1\rangle \otimes |0\rangle \otimes |11\rangle + \alpha |10\rangle \otimes |11\rangle \otimes |11\rangle \right. \\ \left. + \beta |11\rangle \otimes |11\rangle \otimes |10\rangle \right)$$

$$\frac{1}{\sqrt{2}} \left( \alpha \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right) + \beta \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$+ \alpha \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$= \frac{1}{\sqrt{2}} \left( \alpha \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right) + \beta \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

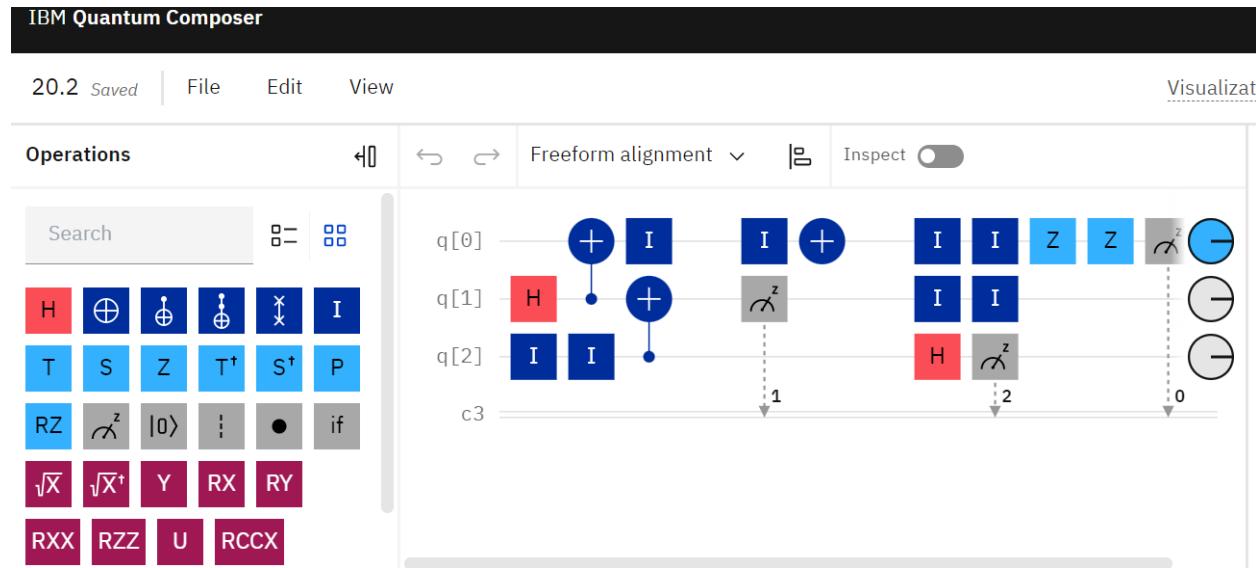
$$+\alpha \left( \left[ \begin{array}{c} 0 \\ 1 \\ 0 \\ 0 \end{array} \right] \otimes \left[ \begin{array}{c} 0 \\ 1 \end{array} \right] \right) + \beta \left( \left[ \begin{array}{c} 0 \\ 0 \\ 0 \\ 1 \end{array} \right] \otimes \left[ \begin{array}{c} 0 \\ 1 \end{array} \right] \right)$$

$$\frac{1}{\sqrt{2}} \left( \alpha \left[ \begin{array}{c} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \right] + \beta \left[ \begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{array} \right] + \alpha \left[ \begin{array}{c} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{array} \right] + \beta \left[ \begin{array}{c} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{array} \right] \right)$$

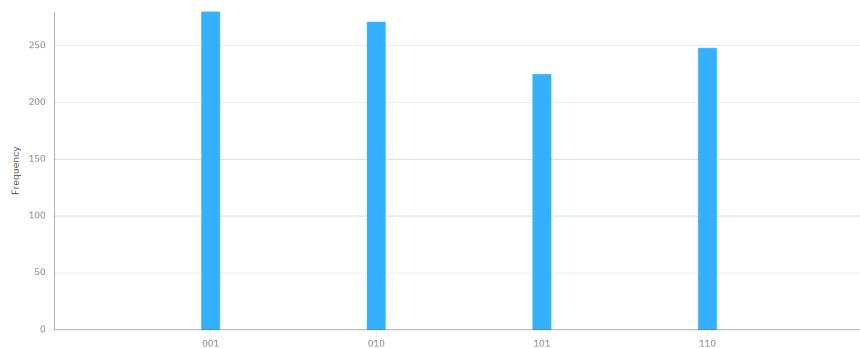
$$= \frac{1}{\sqrt{2}} \left[ \begin{array}{c} \alpha \\ 0 \\ 0 \\ \alpha \\ \beta \\ \beta \\ 0 \\ 0 \end{array} \right]$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

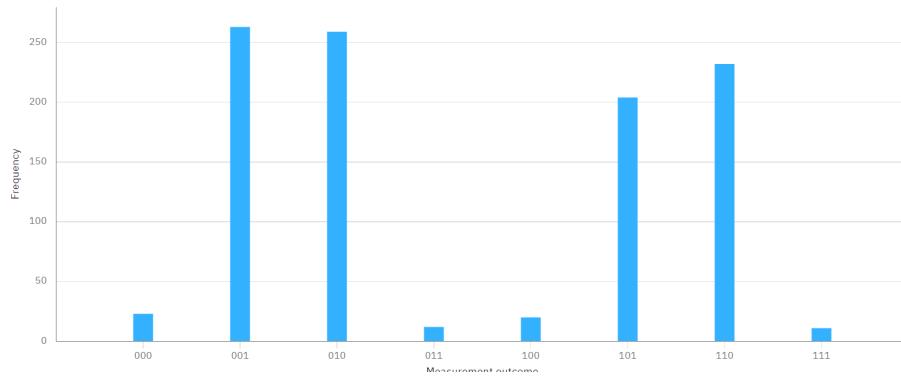
20.2)



## Simulation result



IBMM



20.3)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

20.3)

1) Due to no cloning theorem it won't allow cloning of linear arbitrary states so we need to measure Alice

2) Entanglement swapping (20.5) allows us to transport / teleport

3) Default circuit represents Alice to teleport to Bob. According to 20.4. so yes quantum teleportation happens by default

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

**21.1)**

$$\begin{aligned}
 & \text{i) } f_B(z) \\
 & U_f |0\rangle |0\rangle = |0\rangle |0 \oplus f(0)\rangle \\
 & U_f |0\rangle |1\rangle = |0\rangle |0 \oplus f(0)\rangle \\
 & U_f |1\rangle |0\rangle = |1\rangle |0 \oplus f(1)\rangle \\
 & U_f |1\rangle |1\rangle = |1\rangle |1 \oplus f(1)\rangle
 \end{aligned}$$
  

$$\begin{aligned}
 & \text{ii) } f_B(z) \Rightarrow f_B(0) = 0 \& f_B(1) = 1 \\
 & = |0\rangle |0 \oplus 0\rangle = |0\rangle |0\rangle \\
 & = |0\rangle |0 \oplus 0\rangle = |0\rangle |1\rangle \\
 & = |0\rangle |0 \oplus 1\rangle = |1\rangle |1\rangle \\
 & = |1\rangle |1 \oplus 1\rangle = |1\rangle |0\rangle
 \end{aligned}$$
  

$$\begin{aligned}
 & \text{iii) } f_C(z) \Rightarrow f_C(0) = 1 \& f_C(1) = 0 \\
 & = |0\rangle |0 \oplus 1\rangle = |0\rangle |1\rangle = |0\rangle \\
 & = |0\rangle |1 \oplus 1\rangle = |0\rangle |0\rangle \\
 & = |1\rangle |0 \oplus 0\rangle = |1\rangle |0\rangle \\
 & = |1\rangle |1 \oplus 0\rangle = |1\rangle |1\rangle
 \end{aligned}$$
  

$$\begin{aligned}
 & \text{iv) } f_D(z) \Rightarrow f_D(0) = 1 \& f_D(1) = 1 \\
 & = |0\rangle |0 \oplus 1\rangle = |0\rangle |1\rangle \\
 & = |0\rangle |1 \oplus 1\rangle = |0\rangle |0\rangle \\
 & = |1\rangle |0 \oplus 1\rangle = |0\rangle |1\rangle \\
 & = |1\rangle |1 \oplus 1\rangle = |1\rangle |0\rangle
 \end{aligned}$$

**21.2)**

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

21. 2)

We know from 17. 11 that

$$H^{\otimes 2} |100\rangle = \frac{1}{2} (|100\rangle + |101\rangle + |110\rangle + |111\rangle)$$

$$= \frac{1}{2} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$L.H.S = \frac{1}{2} (H \cdot H) (I \times) |10\rangle |10\rangle$$

$$= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

$$= \frac{1}{2} (H \otimes H) |0\rangle |1\rangle \rightarrow R.H.S$$

$$= \frac{1}{2} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix}$$

$$|+\rangle |-\rangle = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$= \frac{1}{2} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix}$$

$$\text{Hence } (H \otimes H) (I \otimes I) |0\rangle |0\rangle = (H \otimes H) |0\rangle |1\rangle = |+\rangle |-\rangle$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

21.3)

Given state  $|101\rangle$

from 17.21

$$H^{\otimes n} |y\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{x \cdot y} |x\rangle$$

we know  $n=2$   $|101\rangle = |1_0\rangle$

$$\begin{aligned} H^{\otimes 2} |1\rangle &= \frac{1}{\sqrt{2^2}} \left( (-1)^{0 \cdot 1} |0\rangle + (-1)^{1 \cdot 1} |1\rangle \right. \\ &\quad \left. + (-1)^{2 \cdot 1} |2\rangle + (-1)^{3 \cdot 1} |3\rangle \right) \end{aligned}$$

$$0 \cdot 1 = (0 \text{ AND } 0) \oplus (0 \text{ AND } 1) = 0 \oplus 0 = 0$$

$$1 \cdot 1 = (0 \text{ AND } 0) \oplus (1 \text{ AND } 1) = 0 \oplus 1 = 1$$

$$2 \cdot 1 = (1 \text{ AND } 0) \oplus (0 \text{ AND } 1) = 0 \oplus 0 = 0$$

$$3 \cdot 1 = (1 \text{ AND } 0) \oplus (1 \text{ AND } 1) = 0 \oplus 1 = 1$$

$$H^{\otimes 2} |1\rangle = H^{\otimes 2} |01\rangle = \frac{1}{2} (|0\rangle - |1\rangle + |2\rangle - |3\rangle)$$

$$= \frac{1}{2} (|100\rangle - |101\rangle + |110\rangle - |111\rangle) - i$$

from 21.6

$$|+\rangle_{1-2} = \frac{1}{2} (|100\rangle - |101\rangle + |110\rangle - |111\rangle) - i$$

Hence proved  $\textcircled{1} = \textcircled{2}$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

21.4)

From 21.3 let us try for balanced function

$f(x)$  and gives equal number of 0 & 1

$$\text{ex: } f_B(0) = 0, f_B(1) = 1$$

Q1

$$U_f |0\rangle |0\rangle = |0\rangle |0\rangle \oplus f(0)\rangle = |0\rangle |0\oplus 0\rangle = |0\rangle |0\rangle$$

$$U_f |0\rangle |1\rangle = |0\rangle |1\rangle \oplus f(0)\rangle = |0\rangle |1\oplus 0\rangle = |0\rangle |1\rangle$$

$$U_f |1\rangle |0\rangle = |1\rangle |0\rangle \oplus f(1)\rangle = |1\rangle |0\oplus 1\rangle = |1\rangle |1\rangle$$

$$U_f |1\rangle |1\rangle = |1\rangle |1\rangle \oplus f(1)\rangle = |1\rangle |1\oplus 1\rangle = |1\rangle |0\rangle$$

Wish you all the best

Chapter 22

22.1) & 22.2)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

22.1 Given  $x=5$ ,  $f(5)=3$ ,  $n=3$ ,  $m=3$

$y$  assumed as  $|0\rangle$

we know from 22.1

$$U_f |x\rangle_n |y\rangle_m = |x\rangle_n |y \oplus f(x)\rangle_m$$

$$U_f |15\rangle_3 |0\rangle_3 = |15\rangle_3 |0 \oplus f(5)\rangle_3$$

~~|100101>~~

$$= |101\rangle_3 |0 \oplus \text{PH}\rangle = |101\rangle_3 |000\rangle$$

$$= |101\rangle |001\rangle$$

$$= |1101011\rangle$$

22.2 If we are using same method as  
phase quantum oracle

Two basis vectors  $|a\rangle$  &  $|b\rangle$  and  
inner product is  $\langle a|b\rangle$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

i)  $|a\rangle$  transformed by XOR Quantum oracle  
we can assume  $y$  as 0

$$|a\rangle |0\rangle |f(a)\rangle$$

$$|a\rangle |f(a)\rangle$$

ii)  $|b\rangle \rightarrow |b\rangle |f(b)\rangle$

Inner product is (No matrix involved)

$$\langle a|b\rangle = \langle a\rangle |f$$

$$\langle f(a)|f(b)\rangle \langle a|b\rangle$$

for example if  $a = b$  then we know

inner product of same vectors is

$$\langle f(a)|f(b)\rangle \langle a|b\rangle$$

$$\langle f(a)|f(a)\rangle \langle a|a\rangle$$

$$= 1 \cdot 1 = 1$$

& If  $a \neq b$ ,  $\langle a|b\rangle = 0$  as we already know

Basic vector inner product is NOT changed

It is still unitary

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

22.3)

$$\begin{aligned}
 & \text{22.3} \quad U_{\text{swap}} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \langle j | U | i \rangle | ij \rangle \langle ij | \\
 & \qquad \qquad \qquad \boxed{N=4} \\
 & \begin{aligned}
 & \cdot \langle 001 | U_{\text{swap}} | 100 \rangle | 100 \rangle \langle 001 | + \langle 01 | U_{\text{swap}} | 100 \rangle \\
 & \qquad \qquad \qquad \langle 001 | \\
 & \qquad \qquad \qquad + \langle 101 | U_{\text{swap}} | 100 \rangle | 110 \rangle \langle 001 | \\
 & \qquad \qquad \qquad + \langle 111 | U_{\text{swap}} | 100 \rangle | 110 \rangle \langle 200 | \\
 & \qquad \qquad \qquad + \dots + \dots \\
 & \qquad \qquad \qquad \boxed{\cancel{\langle 001 | U_{\text{swap}} | 110 \rangle | 100 \rangle \langle 100 |} + \cancel{\langle 101 | U_{\text{swap}} | 110 \rangle | 100 \rangle \langle 100 |}} \\
 & \qquad \qquad \qquad \text{or} \\
 & \cdot \langle 001 | U_{\text{swap}} | 101 \rangle | 100 \rangle \langle 011 | + \langle 01 | U_{\text{swap}} | 101 \rangle | 100 \rangle \\
 & \qquad \qquad \qquad + \langle 101 | U_{\text{swap}} | 101 \rangle | 110 \rangle \langle 011 | + \\
 & \qquad \qquad \qquad \langle 111 | U_{\text{swap}} | 101 \rangle | 110 \rangle \langle 011 | \\
 & \qquad \qquad \qquad + \\
 & \cdot \langle 001 | U_{\text{swap}} | 110 \rangle | 100 \rangle \langle 101 | + \langle 01 | U_{\text{swap}} | 110 \rangle | 100 \rangle \\
 & \qquad \qquad \qquad + \langle 101 | U_{\text{swap}} | 110 \rangle | 110 \rangle \langle 101 | \\
 & \qquad \qquad \qquad + \langle 111 | U_{\text{swap}} | 110 \rangle | 110 \rangle \langle 101 |
 \end{aligned}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & i \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
$$v_{swap} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

from 16.3 It is proved

22.4)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

for  $f_G$   $f_G(0) = 1$ ,  $f_G(1) = 0$

$$22.4) \langle 01 \langle 11 | (U_f | 0 \rangle | 10 \rangle) = \langle 01 \langle 11 | 101 \rangle = 1$$

$$\langle 01 \langle 01 | (U_f | 0 \rangle | 11 \rangle) = \langle 01 \langle 01 | 100 \rangle = 1$$

$$\langle 11 \langle 01 | (U_f | 11 \rangle | 10 \rangle) = \langle 11 \langle 01 | 110 \rangle = 1$$

$$\langle 11 \langle 11 | (U_f | 11 \rangle | 11 \rangle) = \langle 11 \langle 11 | 111 \rangle = 1$$

$$= | 01 \rangle | 100 \rangle + | 100 \rangle | 01 \rangle + | 110 \rangle | 110 \rangle + | 111 \rangle | 111 \rangle$$

$$= \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} [1000] + \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} [0100] + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} [0010]$$

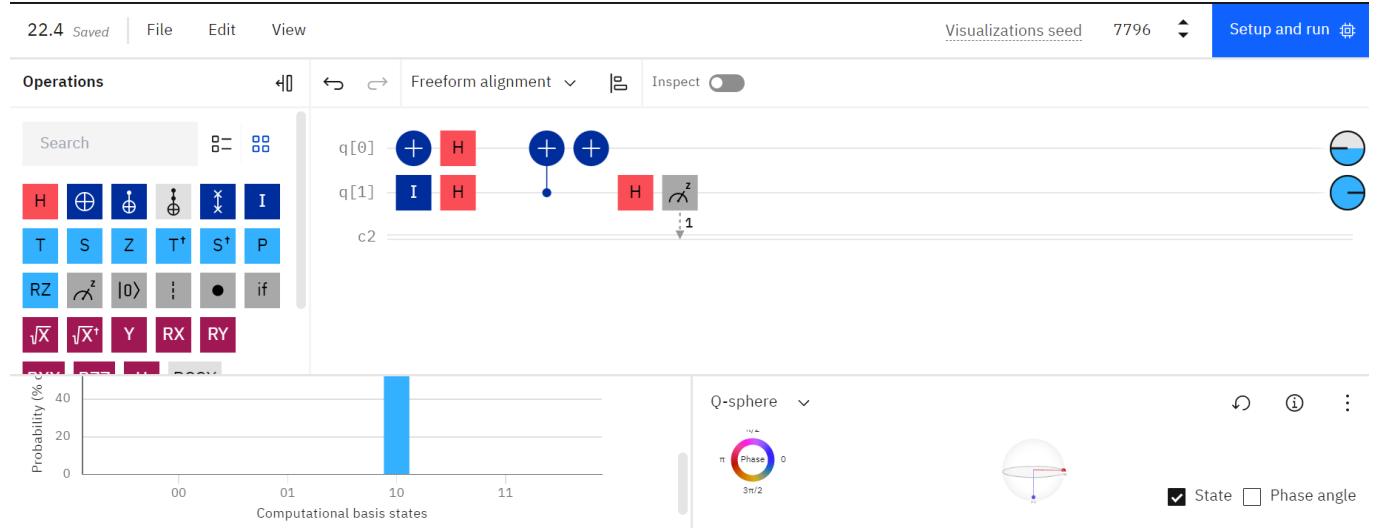
$$= \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} [0001]$$

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

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

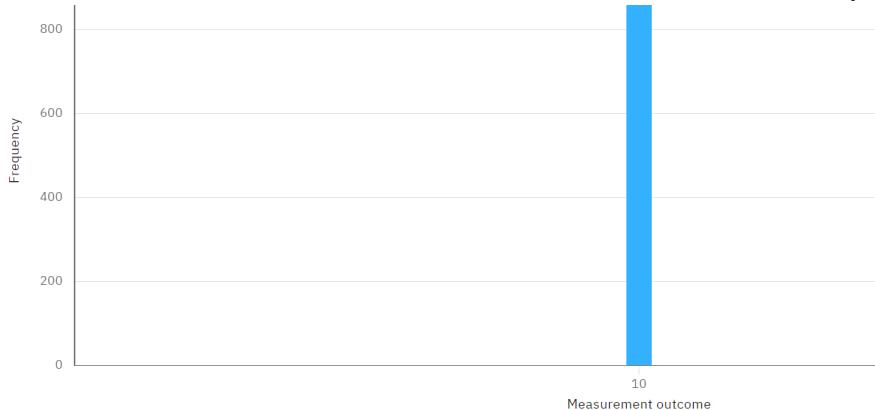
$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



## Simulator outcome

It is 10 in IBMQ instead of 01 because for balanced function MSB is always 1 ( page no 212)



## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$f_0, f_1(0) = 1 \quad f_0(1) = 1$$

$$\langle 01| \langle 11| (U_f |0\rangle |0\rangle) = \langle 01| \langle 10| = 1$$

$$\langle 01| \langle 01| (U_f |0\rangle |1\rangle) = \langle 00| \langle 10| = 1$$

$$\langle 11| \langle 11| (U_f |1\rangle |0\rangle) = \langle 11| \langle 11| = 1$$

$$\langle 11| \langle 01| (U_f |1\rangle |1\rangle) = \langle 10| \langle 11| = 1$$

$$= |01\rangle |00\rangle + |00\rangle |01\rangle + |11\rangle |10\rangle$$

$$+ |10\rangle |11\rangle$$

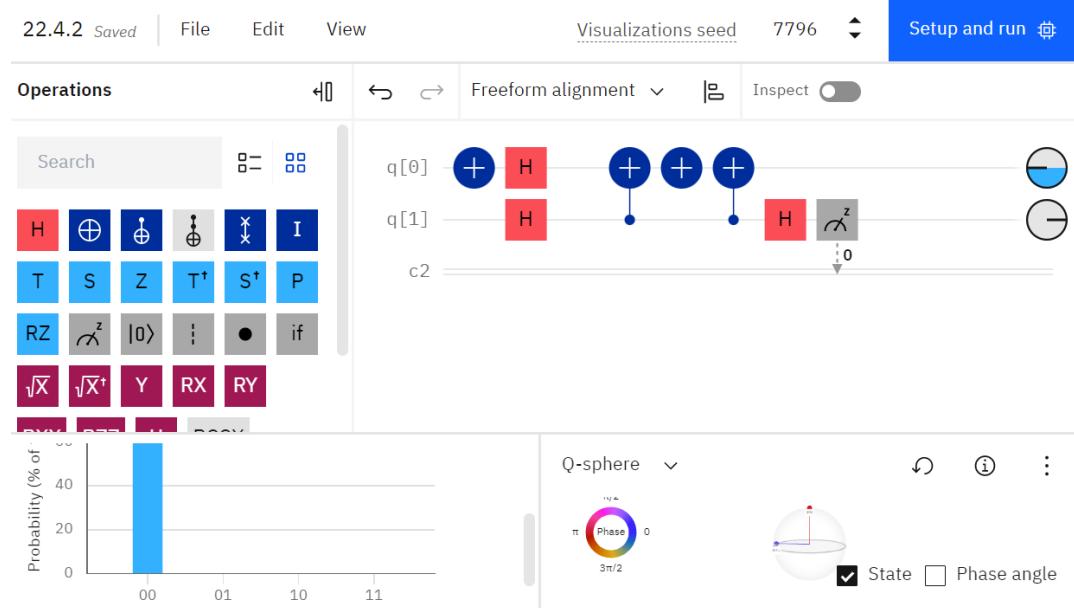
$$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

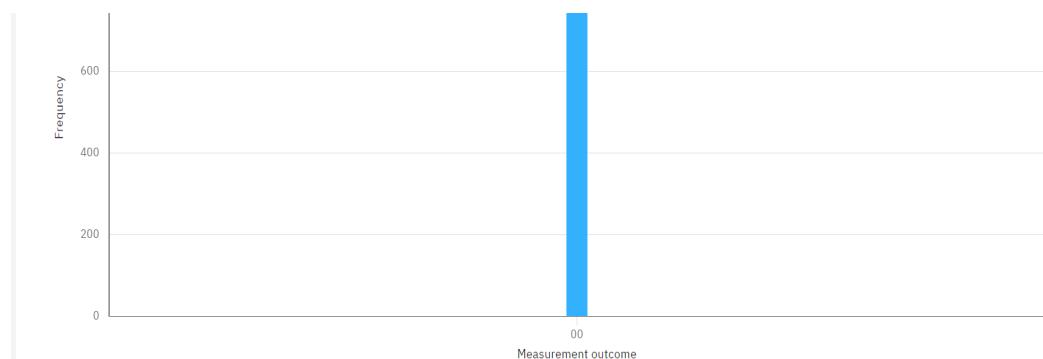
$$= \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

It is similar to above statement and as explained in page 200 it is always 0 for constant function here we can also remove both cnot gates as it wont make much difference. I have added it to describe the functionality



Simulator results



# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 23

13.1)  $\text{fig } 23.2$

Let us assume angle between  $|a\rangle$  &  $|b\rangle$  as  $\alpha$

$\sqrt{ b\rangle}$	$\alpha + 2\theta$	$-\theta$
$w\sqrt{ b\rangle}$	$\alpha - 2\theta$	$+3\theta$
$\sqrt{w}\sqrt{ b\rangle}$	$\alpha$	$+ \theta$
$w\sqrt{w}\sqrt{ b\rangle}$	$\alpha - 4\theta$	$+5\theta$

angle between  $|b\rangle$  &  $|a\rangle$ , between  $|b\rangle$  &  $|a\rangle$ ,

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

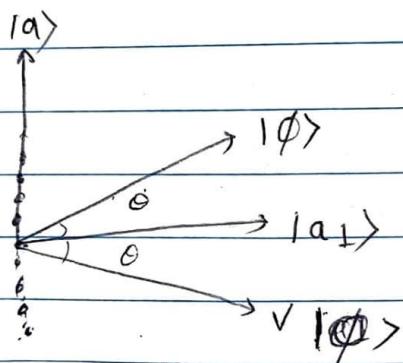
23.2)

23.2)

$$\text{Given } V = I - 2|a\rangle\langle a|$$

As we already know  $V$  is a quantum gate and it is linear combination of all basis vectors except our target entry  $|a\rangle$

we know phase shift is added to target entry and it is multiplied by  $-1$



$$\text{From } V|\phi\rangle = (I - 2|a\rangle\langle a|)|\phi\rangle$$

$$= \sum_{i=0, i \neq a}^{2-1} |\phi\rangle$$

Here when operator  $V$  is applied to  $|\phi\rangle$  then it reflects  $|\phi\rangle$  towards negation (phase shift)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

23.3)

Given  $n=2$ ,  $a=3$

$$|\phi\rangle = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle$$

We know  $n=2$ ,  $a=3$

$$= \frac{1}{\sqrt{2^2}} \sum_{x=0}^{2^2-1} |x\rangle$$
$$= \frac{1}{2} \sum_{x=0}^3 |x\rangle$$
$$= \frac{1}{2} (|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$
$$|a\rangle = |11\rangle$$
$$|a_+\rangle = \frac{1}{\sqrt{2^n - 1}} \sum_{x=0 (x \neq a)}^{2^n-1} |x\rangle$$
$$= \frac{1}{\sqrt{3}} \sum_{x=0 (x \neq 3)}^3 |x\rangle$$
$$= \frac{1}{\sqrt{3}} (|00\rangle + |01\rangle + |10\rangle)$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

23.4 & 23.5)

23.4 Sol)

we know

$$v|\psi\rangle = \sqrt{\frac{1}{2}} (|00\rangle + |01\rangle + |10\rangle + |11\rangle)$$

$$= \frac{1}{2} |00\rangle + |01\rangle + |10\rangle - |11\rangle$$

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

23.5)

Algorithm B has linear speed up  
over Algorithm A

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 24

### 24.1)

24.1)

$$U = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \langle j | U | i \rangle | j \rangle \langle i | \rightarrow 22.6$$

we know from 24.4 that

$$U | 1 \rangle_n = (-1)^{f(1)} | 1 \rangle_n \rightarrow 23.9$$

$U_f | 01 \rangle = - | 01 \rangle$  and for every  
other base it will be same

$$\langle 01 | \langle 01 | (U_f | 10 \rangle | 10 \rangle) = \langle 00 | | 00 \rangle = 1$$

$$\langle 01 | \langle 1 | (U_f | 10 \rangle | 10 \rangle) - \langle 00 | | 01 \rangle = -1 \quad [U_f | 01 \rangle = - | 01 \rangle]$$

$$\langle 1 | \langle 1 | (U_f | 11 \rangle | 11 \rangle) = \langle 11 | | 11 \rangle = 1$$

$$\langle 1 | \langle 0 | (U_f | 11 \rangle | 11 \rangle) = \langle 1 | | 01 \rangle = \langle 10 | | 0 \rangle = 1$$

$$U_f = \langle 01 | \langle 01 | (U_f | 10 \rangle | 10 \rangle) | 00 \rangle \langle 00 | + \langle 01 | \langle 1 | (U_f | 11 \rangle | 11 \rangle) | 01 \rangle \langle 01 | + \langle 1 | \langle 0 | (U_f | 11 \rangle | 10 \rangle) | 10 \rangle \langle 10 | + \langle 1 | \langle 0 | (U_f | 11 \rangle | 11 \rangle) | 11 \rangle \langle 11 |$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$= |00\rangle\langle 00| - |01\rangle\langle 01| + |11\rangle\langle 11| + |10\rangle\langle 10|$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = U_1 \text{ i.e., } 24.4$$

24.4)

we need to apply  $H^{\otimes 2}(2|0\rangle_2\langle 0|_2 - I)H^{\otimes 2}(|1\rangle_2)$

$$\text{Firstly } 2|0\rangle_2\langle 0|_2 - I = 2 \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$= 2 \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$2|0\rangle_2\langle 0|_2 - I = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\begin{aligned}
 &= \frac{1}{2} \begin{pmatrix} 1 & + & + & + \\ + & -1 & + & -1 \\ + & + & -1 & -1 \\ + & -1 & + & + \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \frac{1}{2} \begin{pmatrix} 1 & + & + & + \\ + & -1 & + & -1 \\ + & + & -1 & -1 \\ + & -1 & + & -1 \end{pmatrix} \\
 &\quad \frac{1}{2} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \\
 &= \frac{1}{4} \begin{pmatrix} 1 & + & + & + \\ + & -1 & + & -1 \\ + & + & -1 & -1 \\ + & -1 & + & + \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \end{pmatrix} \\
 &= \frac{1}{4} \begin{pmatrix} 1 & + & + & + \\ + & -1 & + & -1 \\ + & + & -1 & -1 \\ + & -1 & + & + \end{pmatrix} \begin{pmatrix} -1 \\ 1 \\ 1 \\ -1 \end{pmatrix} \\
 &= \frac{1}{4} \begin{pmatrix} +1 & +1 & +1 & -1 \\ -1 & -1 & +1 & +1 \\ +1 & +1 & -1 & +1 \\ -1 & -1 & -1 & -1 \end{pmatrix} \frac{1}{4} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \\
 &\quad \boxed{111} \text{ over rotab}!
 \end{aligned}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

24.3)

24.3)

We know from 23.9 for  $a=4$

$$U_f = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

$$\sqrt{1/\rho} = U_f \left( \frac{1}{2} (|000\rangle + |001\rangle + |100\rangle + |111\rangle) \right)$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

For  $a=3$

$$= H^{\otimes 2} (2|00\rangle_2 \langle 01_2 - I) H^{\otimes 2} (\sqrt{1/\rho})$$

$$= \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\frac{1}{\sqrt{4}} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1+1+1-1 \\ 1-1+1+1 \\ 1+1-1+1 \\ 1+1-1-1 \end{pmatrix}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 2 \\ 2 \\ 2 \\ -2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}$$

$$\frac{1}{\sqrt{4}} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix}$$

$$\frac{1}{\sqrt{4}} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \\ -1 \\ 1 \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\frac{1}{4} \begin{bmatrix} 1 & -1 & -1 & +1 \\ 1 & +1 & -1 & -1 \\ 1 & -1 & +1 & -1 \\ 1 & +1 & +1 & +1 \end{bmatrix} \cdot \frac{1}{4} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$\vdash |111\rangle$

24.4)

from 24.15

$$C_2 = (I \otimes H) \cup_{\text{xor}} (I \otimes H)$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 0 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ 0 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 1 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 0 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \\ 0 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} & 1 & \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \end{pmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$= \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

$$= \frac{1}{2} \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & -2 \end{bmatrix}$$

$$= \cancel{\frac{1}{2}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} = CZ$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

### 24.5

```
#24.5
#Keywords
#superposition is sup_pos
#amplitude is amp

#Grovers Algorithm is implemented using standard procedure.
#Steps followed in books are implemented and tried to create classical grovers algorithm in different

plots = []

# Putting all qubits in superposition using hadamard gate
def sup_pos(qubits):
    states = []
    total_states = int(math.pow(2,qubits))
    amp = 1/math.sqrt(total_states)
    for _ in range(0,total_states):
        states.append(amp)
    return states

# phase flip inverts the states amplitude
def grover_diffusion(states):
    average = sum(states)/len(states)
    for i in range(0,len(states)):
        states[i] = (average-states[i]) + average           #inversion about mean
    return states

#oracle to implement phase inversion
def oracle(states,datalist,key):
    for i in range(0,int(len(datalist))):
        if datalist[i] == key:
            states[i] *= -1                                #phase inversion
    return states
```

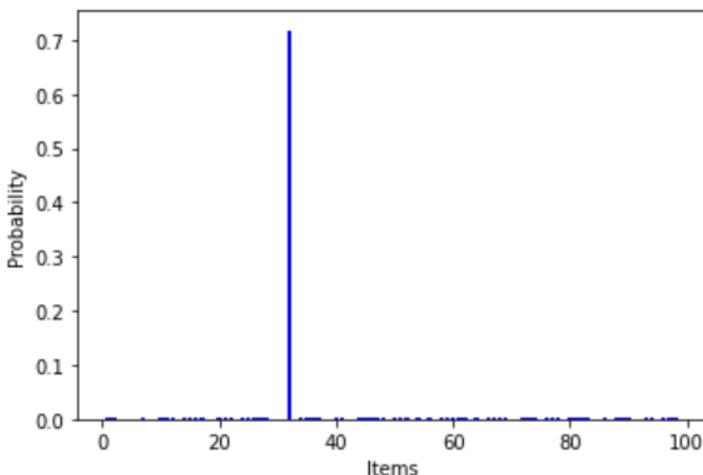
## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

```
#classical main grover search function
def grover_search(qubits,datalist,key):
    states = sup_pos(qubits)
    num_iterations = math.ceil(math.sqrt(math.pow(2,qubits)))           #iterations
    probability_states = []
    for _ in range(0,num_iterations):
        states = oracle(states,datalist,key)
        states = grover_diffusion(states)
        probability_states = [states[i]*states[i] for i in range(0,len(states))]
    plots.append(probability_states)
    return probability_states

def grover(datalist,key):
    size_datalist = len(datalist)
    qubits_needed = math.ceil(math.log(size_datalist,2))
    paddings_required = int(math.pow(2,qubits_needed)) - size_datalist
    #required if the number of data items is not a power of 2.
    for _ in range(0,paddings_required):
        datalist.append(0)
    grover_search(qubits_needed,datalist,key)
```

```
datalist = random.sample(range(1, 100), 64)
grover(datalist,32)
print(datalist)
print("plots showing the change in probabilities of items after every grover's iteration :")
result ={}
iteration = 1
for plot in plots:
    for i in range(len(plot)):
        result[datalist[i]] = plot[i]
    print("-----")
    print("Iteration ",iteration," :")
    plt.bar(result.keys(), result.values(),color='b')
    plt.ylabel("Probability")
    plt.xlabel("Items")
    plt.show()
    iteration +=1
    print("-----")
```

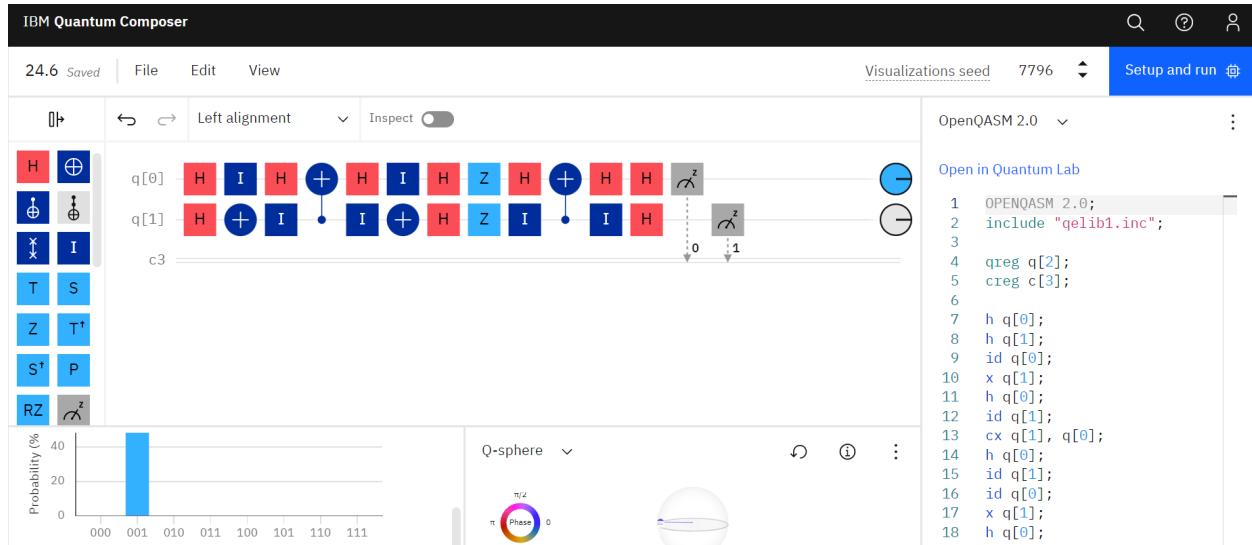
Iteration 8 :



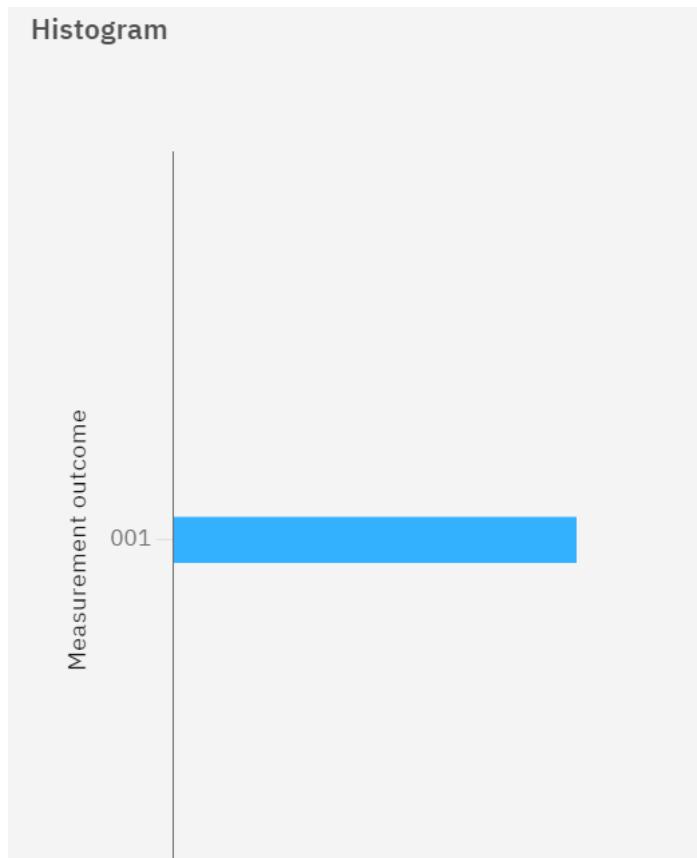
24.6)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



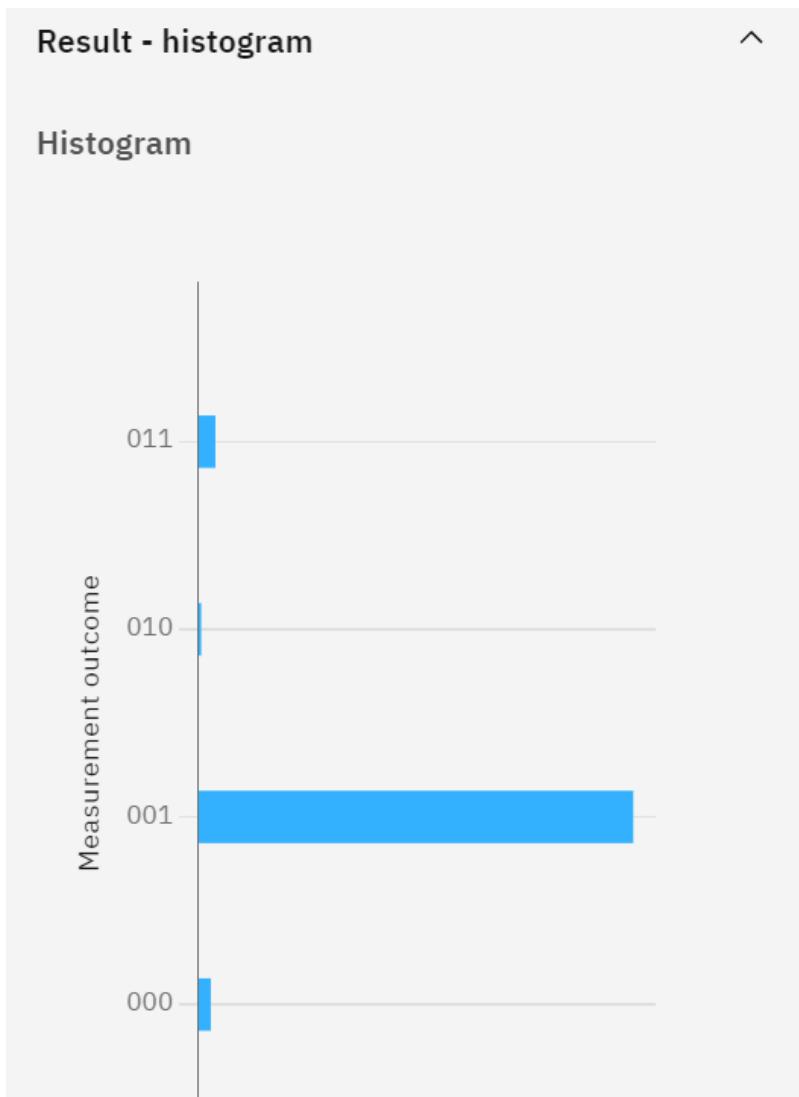
## Simulation



## Hardware circuit result

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 25

25.1)

Given that we have to prove

$$25.10 \Rightarrow 25.9$$

$$\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{N-1} \end{pmatrix} \xrightarrow{\frac{1}{\sqrt{N}}} \begin{pmatrix} w^{-0,0} & w^{-0,1} & \cdots & w^{-0,(N-1)} \\ w^{-1,0} & w^{-1,1} & \cdots & w^{-(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ w^{-(N-1),0} & w^{-(N-1),1} & \cdots & w^{(N-1)-(N-1)} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{pmatrix}$$

we know 25.9. is

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} e^{i 2\pi k j / N} x_j$$

$$= \frac{1}{\sqrt{N}} \begin{pmatrix} e^{-i 2\pi (0,0)/N} & e^{-i 2\pi (0,1)/N} & \cdots & e^{-i 2\pi (0,N-1)/N} \\ e^{-i 2\pi (1,0)/N} & e^{-i 2\pi (1,1)/N} & \cdots & e^{-i 2\pi (1,N-1)/N} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-i 2\pi ((N-1),0)/N} & e^{-i 2\pi ((N-1),1)/N} & \cdots & e^{-i 2\pi ((N-1),(N-1))/N} \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{pmatrix} \xrightarrow{\quad} ②$$

from 25.4 (First property we know)

$$e^{i2\pi m/N} = w^m$$

so it equals to ① after substitution

$$\begin{pmatrix} 1 \\ \sqrt{N} \\ w^{-0,0} & w^{-0,1} & \cdots & w^{-0,(N-1)} \\ w^{-1,0} & w^{-1,1} & \cdots & w^{-1,(N-1)} \\ w^{-(N-1),0} & w^{-(N-1),1} & \cdots & w^{-(N-1),(N-1)} \end{pmatrix}$$

$$\begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-1} \end{pmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

25.2)

25.2

Yes.  $U_{QFT}$  is Hermitian

From 25.20 we know

$$U_{QFT} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad \text{i.e., Hadamard gate}$$

$$U_{QFT}^T = U_{QFT} \quad (\text{Hermitian rule is satisfied})$$

(25.12)  
Here,  $U_{QFT}$  is symmetric & satisfies Hermitian requirements. I believe it should be Hermitian because according to Fourier transforms domains changes i.e., bases in QFT it should remain same

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

25.3)

U<sub>QFT</sub> is unitary (considering row vectors)

$$\langle x | = \frac{1}{\sqrt{N}} \begin{bmatrix} w^{-y_0}, w^{-y_1}, \dots, w^{-y_{(N-1)}} \end{bmatrix}^*$$

$$\langle y | = \frac{1}{\sqrt{N}} \begin{bmatrix} w^{-y_0}, w^{-y_1}, \dots, w^{-y_{(N-1)}} \end{bmatrix}^*$$

$$\langle x | y \rangle = \frac{1}{N} \sum_{i=0}^{N-1} (w^{-m \cdot L})^* \cdot w^{-n \cdot L}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} ((e^{-i2\pi m \cdot L/N}))^* e^{-i2\pi n \cdot L}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} w^{-m \cdot L} \cdot w^{-n \cdot L}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} (w^{-L(m-n)})$$

if  $m=n$  it is 1 else it is zero

$U_{QFT}$  is unitary

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

25.4)

25.3)

$U_{QFT}$  is unitary (considering row vectors)

$$|\psi\rangle = \frac{1}{\sqrt{N}} [w^{-y_0}, w^{-y_1}, \dots, w^{-y_{(N-1)}}]^T$$

$$|\psi\rangle = \frac{1}{\sqrt{N}} [w^{-y_0}, w^{-y_1}, \dots, w^{-y_{(N-1)}}]^T$$

$$\langle x|y\rangle = \frac{1}{N} \sum_{i=0}^{N-1} (w^{-m \cdot L})^+ w^{-n \cdot L}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} ((e^{i 2\pi m \cdot i / N})^+)^T e^{-i 2\pi n \cdot L}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} w^{-m \cdot L} \cdot w^{-n \cdot L}$$

$$= \frac{1}{N} \sum_{i=0}^{N-1} (w^{-L(m-n)})$$

if  $m=n$  it is 1 else it is zero

$U_{QFT}$  is unitary

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

25.5)

25.5)

$$\text{Given } U_{\text{IQFT}} U_{\text{QFT}} = U_{\text{QFT}} U_{\text{IQFT}} = I$$

we know

$$U_{\text{QFT}} = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^{-1} & \omega^{-2} & \dots & \omega^{-(N-1)} \\ 1 & \omega^{-2} & \omega^{-4} & \dots & \omega^{-2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{-(N-1)} & \omega^{-2(N-1)} & \dots & \omega^{-(N^2-N)} \end{pmatrix}$$

$$U_{\text{IQFT}} = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^1 & \omega^2 & \dots & \omega^{(N-1)} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \omega^{(N-1)(N-1)} & \dots & \omega^{(N^2-N)} \end{pmatrix}$$

$$U_{\text{IQFT}} * U_{\text{QFT}} = \frac{1}{N} \begin{pmatrix} N & N\omega^1 + \omega^2 + \omega^3 & N\omega^2 + \omega^4 + \omega^6 & \dots & N\omega^{(N-1)} + \omega^{(N^2-N)} \\ N\omega^1 + \omega^2 + \omega^3 & 1 + \omega^1 + \omega^2 + \dots + \omega^{(N-1)} & \dots & \dots & \dots \\ N\omega^2 + \omega^4 + \omega^6 & \dots & 1 + \omega^2 + \omega^4 + \dots + \omega^{(N-1)(N-1)} & \dots & \dots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ N\omega^{(N-1)} + \omega^{(N^2-N)} & \dots & \dots & \dots & N \end{pmatrix}$$

From 25.5 we know  $1 + \omega^1 + \omega^2 + \dots + \omega^{(N-1)} = 0$

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

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$U_{\text{QFT}} = \frac{1}{\sqrt{N}} \begin{bmatrix} N & 1 + \omega^{-1} & \omega & \dots & \omega^{(N-1)} \\ 1 + \omega & N & \dots & \dots & \omega^{(N-1)} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ & & & N & \end{bmatrix}$$

From 25.5 & 25.6

$$= \frac{1}{\sqrt{N}} \begin{bmatrix} N & 0 & \dots & 0 \\ 0 & N & \dots & \dots \\ \vdots & \vdots & \ddots & \vdots \\ & & & N \end{bmatrix}$$

25b)

We know

$$U_{\text{IQFT}} = \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & \omega^1 & \dots & \dots & & \omega^{(N-1)} \\ 1 & \omega^2 & \dots & \dots & & \omega^{(N-2)(N-1)} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 1 & \omega^{(N-1)} & \dots & \dots & \dots & \omega^{(N-1)(N-1)} \end{bmatrix}$$

For 2 qubit  $\omega = i$

$$= \frac{1}{\sqrt{N}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\begin{aligned}
 V_{\text{QFT}} &= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix} \\
 &= V_{\text{QFT}} \cdot V_{\text{I\&T}} \\
 &= \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \cdot \frac{1}{2} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{pmatrix} \\
 &= \frac{1}{4} \begin{pmatrix} 1 \times 1 + 1 \times 1 + 1 \times 1 + 1 \times 1 & 1 \times 1 + i - 1 + 1 & 1 - 1 + 1 - 1 & 1 - i + 1 - 1 \\ 1 \times 1 - i - 1 + 1 & 1 + 1 + 1 + 1 & 1 + i - 1 - i & 1 - 1 + i - 1 \\ 1 \times 1 + 1 - 1 & 1 + 1 - i - 1 & 1 + 1 + 1 + 1 & 1 + i - 1 - 1 \\ 1 \times 1 - 1 - i & 1 - 1 + 1 - 1 & 1 - i + 1 + 1 & 1 + 1 + i + 1 \end{pmatrix} \\
 &= \frac{1}{4} \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 0 & 0 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}
 \end{aligned}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



#25.7

```
from sympy.abc import w
N = int(input("Enter a value", ))
# w = 'w'

matrix=[]
row=[]
for i in range(N): #total row is 3
    row=[]
    for j in range(N): #total column is 3
        # row.append( a \N{SUPERSCRIPT} (i*j))
        row.append( (w)** (i*j) )

    matrix.append(row) #add fully defined column into the row
print( matrix )
```



Enter a value4

```
[[1, 1, 1, 1], [1, w, w**2, w**3], [1, w**2, w**4, w**6], [1, w**3, w**6, w**9]]
```

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 26

26.1)

26.1)

According to method show in book  
& 26.6) For 3 qubit matrix  $N = 8$

According to 25.22

$$U_{QFT} = \frac{1}{\sqrt{8}}$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & w^1 & w^2 & w^3 & w^4 & w^5 & w^6 & w^7 \\ 1 & w^{-2} & w^{-4} & w^6 & w^{-8} & w^{10} & w^{-12} & w^{-14} \\ 1 & w^{-3} & w^{-6} & w^{-9} & w^{-12} & w^{15} & w^{-18} & w^{-21} \\ 1 & w^{-4} & w^{-8} & w^{-12} & w^{16} & w^{20} & w^{-24} & w^{-1} \\ 1 & w^{-5} & w^{10} & w^{-15} & w^{20} & w^{25} & w^{-30} & w^{-35} \\ 1 & w^{-6} & w^{-12} & w^{-18} & w^{24} & w^{30} & w^{-36} & w^{-42} \\ 1 & w^{-7} & w^{-14} & w^{-21} & w^{-28} & w^{35} & w^{-44} & w \end{pmatrix}$$

In page 249 It is mentioned different so why  
has different interpretation and am following  
method used in this text book

$$\text{Given } w = e^{i \frac{2\pi}{N}} = e^{i \frac{2\pi}{8}}$$

If we convert it  $\cos \frac{\pi}{4} + i \sin \frac{\pi}{4} = \frac{1+i}{\sqrt{2}}$

$$\text{All even } w^8 = \left(e^{i \frac{2\pi}{8}}\right)^8 = \frac{1+i}{\sqrt{2}} = \frac{i}{\sqrt{2}}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$\begin{array}{cccccccccc}
 & | & | & | & | & | & | & | & | & \Psi \\
 \downarrow & \frac{1}{\sqrt{2}} & \frac{\sqrt{2}}{1+i} & \frac{(\sqrt{2})^3}{1+i} & \frac{(\sqrt{2})^4}{1+i} & \frac{(\sqrt{2})^5}{1+i} & \frac{(\sqrt{2})^6}{1+i} & \frac{(\sqrt{2})^7}{1+i} & \dots & \\
 \downarrow & (\frac{\sqrt{2}}{1+i})^2 & (\frac{\sqrt{2}}{1+i})^4 & (\frac{\sqrt{2}}{1+i})^6 & (\frac{\sqrt{2}}{1+i})^8 & (\frac{\sqrt{2}}{1+i})^{10} & (\frac{\sqrt{2}}{1+i})^{12} & (\frac{\sqrt{2}}{1+i})^{14} & \dots & \\
 \downarrow & (\frac{\sqrt{2}}{1+i})^3 & (\frac{\sqrt{2}}{1+i})^6 & (\frac{\sqrt{2}}{1+i})^9 & (\frac{\sqrt{2}}{1+i})^{12} & (\frac{\sqrt{2}}{1+i})^{15} & (\frac{\sqrt{2}}{1+i})^{18} & (\frac{\sqrt{2}}{1+i})^{21} & \dots & \\
 \downarrow & (\frac{\sqrt{2}}{1+i})^4 & (\frac{\sqrt{2}}{1+i})^8 & (\frac{\sqrt{2}}{1+i})^{16} & (\frac{\sqrt{2}}{1+i})^{32} & (\frac{\sqrt{2}}{1+i})^{64} & (\frac{\sqrt{2}}{1+i})^{128} & (\frac{\sqrt{2}}{1+i})^{256} & \dots & \\
 \downarrow & (\frac{\sqrt{2}}{1+i})^5 & (\frac{\sqrt{2}}{1+i})^{10} & (\frac{\sqrt{2}}{1+i})^{15} & (\frac{\sqrt{2}}{1+i})^{20} & (\frac{\sqrt{2}}{1+i})^{40} & (\frac{\sqrt{2}}{1+i})^{80} & (\frac{\sqrt{2}}{1+i})^{160} & \dots & \\
 \downarrow & (\frac{\sqrt{2}}{1+i})^6 & (\frac{\sqrt{2}}{1+i})^{12} & (\frac{\sqrt{2}}{1+i})^{18} & (\frac{\sqrt{2}}{1+i})^{24} & (\frac{\sqrt{2}}{1+i})^{48} & (\frac{\sqrt{2}}{1+i})^{96} & (\frac{\sqrt{2}}{1+i})^{192} & \dots & \\
 \downarrow & (\frac{\sqrt{2}}{1+i})^7 & (\frac{\sqrt{2}}{1+i})^{14} & (\frac{\sqrt{2}}{1+i})^{21} & (\frac{\sqrt{2}}{1+i})^{28} & (\frac{\sqrt{2}}{1+i})^{56} & (\frac{\sqrt{2}}{1+i})^{112} & (\frac{\sqrt{2}}{1+i})^{224} & \dots & \\
 \end{array}$$

here given input  $1000 \rightarrow$

As we know when apply Hadamard gate

$\rightarrow$  to individually it  $\oplus$  will be  $(1+i + 1-i)$

( $\therefore QFT = \text{Hadamard}^{\otimes n}$  for  $n$  bits)

and we know their will be no phase shift

if all bits is zero then we know their  
will equal probability in all states if all  
the given inputs are in starting state

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

26.2) Given that we have to prove

$$U_{CPS,\phi} U_{CPS,-\phi} = I$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{i\phi} \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-i\phi} \end{bmatrix}$$

$$\begin{bmatrix} 1 \otimes 1 + 0 \otimes 0 + 0 \otimes 0 + 0 \otimes 0 & 0 \otimes [0+1+0+i0] & 0 [0+0+1+i0] & 0 [0+0+0-i0] \\ 0 \otimes 1 + 0 \otimes 0 + 0 \otimes 0 + 0 \otimes 0 & 1 [0+1+0+i0] & 0 [0+0+1+i0] & 0 [0+0+0-i0] \\ 0 \otimes 1 + 0 \otimes 0 + 0 \otimes 0 + 0 \otimes 0 & 0 [0+1+0+i0] & 1 [0+0+1+i0] & 0 [0+0+0-i0] \\ 0 \otimes 1 + 0 \otimes 0 + 0 \otimes 0 + 0 \otimes 0 & 0 [0+0+0+i0] & 0 [0+0+1+i0] & e^{i\phi}, e^{-i\phi} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & e^{-i\phi} + i\phi \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

26.3)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

26.3) Given that we have to prove

$$U_{\text{swap}} = U_{\text{swap}}^{-1} \Rightarrow U_{\text{swap}} \cdot U_{\text{swap}} = I$$

$$U_{\text{swap}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$1^{\text{st}} \text{ method } U_{\text{swap}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

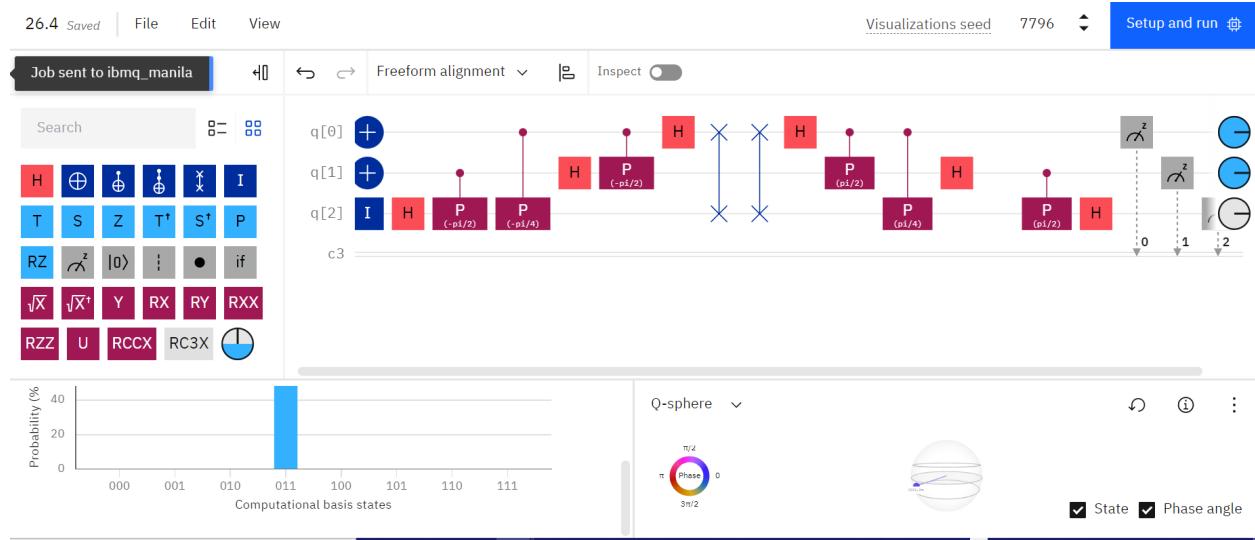
2nd method

$$U_{\text{swap}} \cdot U_{\text{swap}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

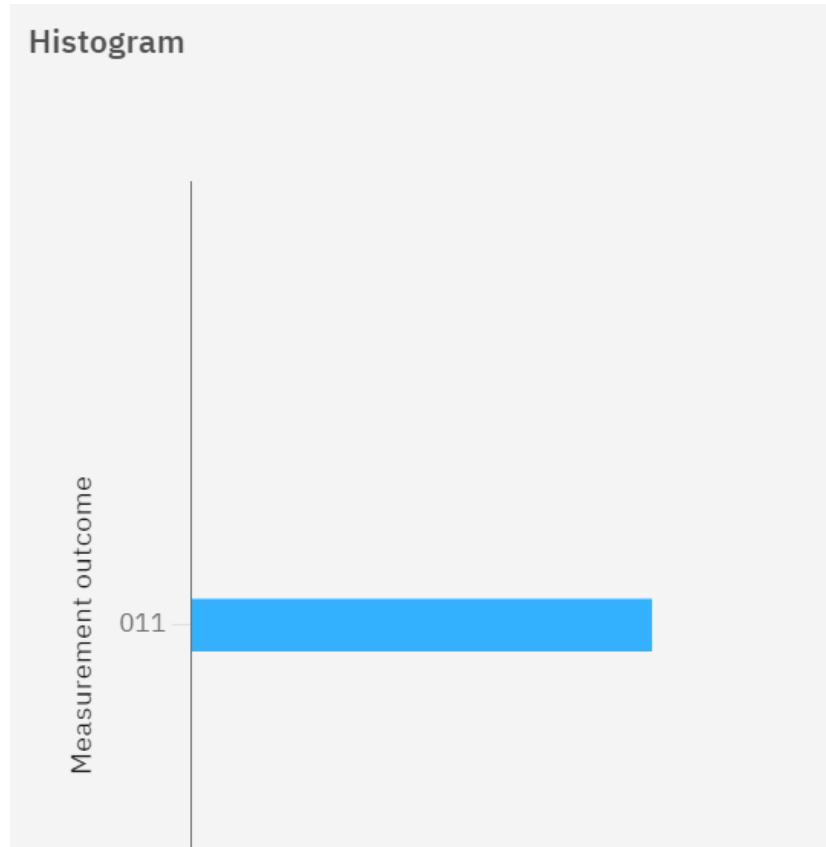
$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = I$$

26.4)

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



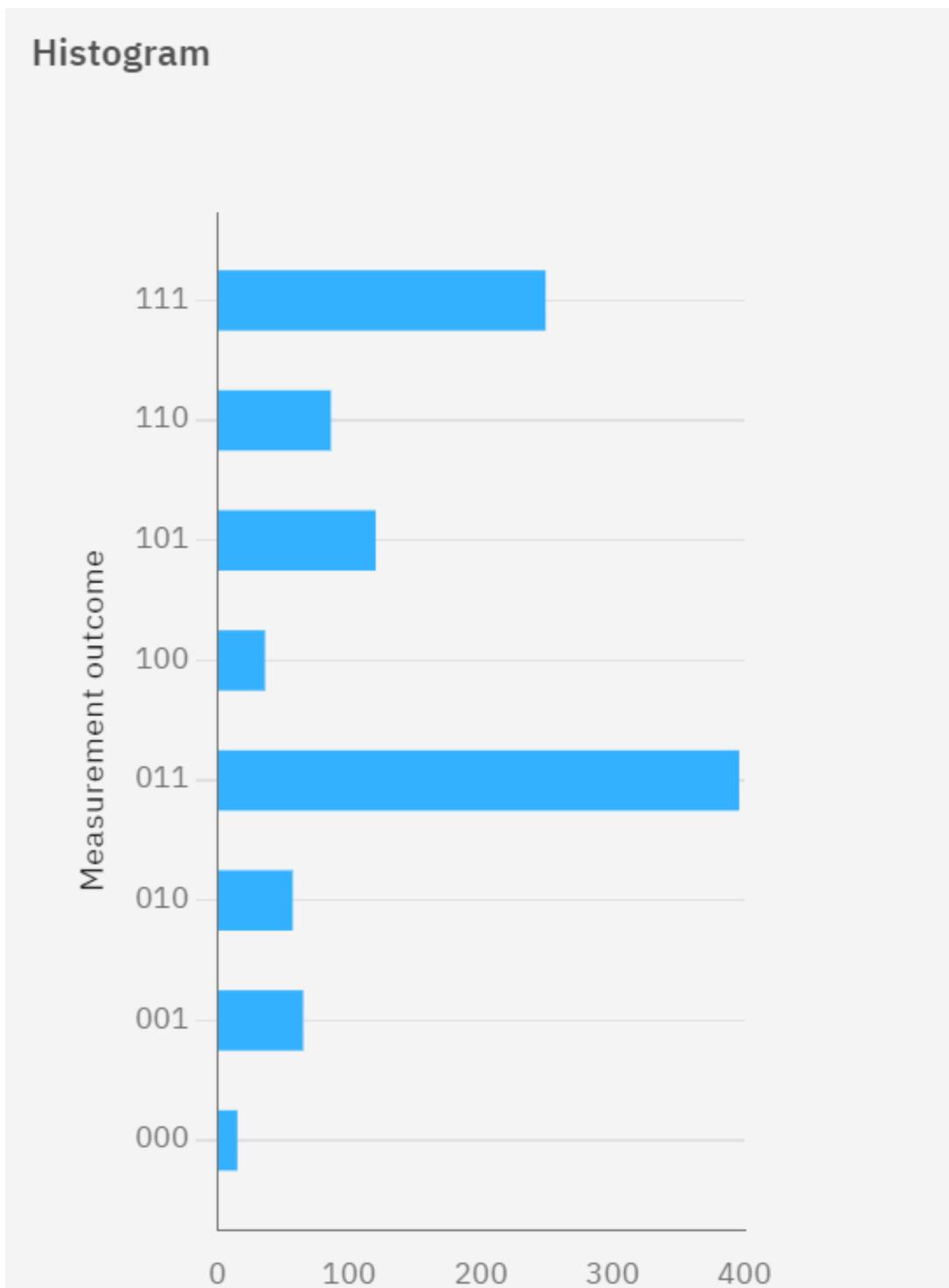
## Simulation result



## Hardware circuit result

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

26.5) according to figure 26.7 we need

$n-1$  U.c.p.s,  $\rho$  gates with  $\rho$  from  $\frac{-2\pi}{2^2}$  to  $\frac{-2\pi}{2^n}$

if the execution time is  $y = \frac{2^n}{2}$  seconds then

i believe as we know QF has  $O(n^2)$  speed

26.6) we have derived and elaborated matrix in 26.1

26.7)  $|11001100\rangle \xrightarrow{\text{swap}} |0011001\rangle$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

## Chapter 27

27.1)

27<sup>th</sup> Chapter

27.1) Given  $\sigma_z$  of  $| \psi \rangle$

From example 27.2 we know

$$| \psi \rangle = \cos\theta/2 e^{-i\phi/2} | 0 \rangle + \sin\theta/2 e^{i\phi/2} | 1 \rangle$$

If we include phase angle as well

$$| \psi \rangle = e^{i\chi} (\cos\theta/2 e^{-i\phi/2} | 0 \rangle + \sin\theta/2 e^{i\phi/2} | 1 \rangle)$$

↳ from [27.1]

$$\langle \psi | \sigma_z | \psi \rangle = e^{-i\chi} (\cos\theta/2 e^{i\phi/2} \sin\theta/2 e^{i\phi/2})$$

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} e^{i\chi} \begin{pmatrix} \cos\theta/2 e^{-i\phi/2} \\ \sin\theta/2 e^{i\phi/2} \end{pmatrix}$$

$$= e^{-i\chi+i\chi} \cdot \cos\theta \quad \text{from } \rightarrow 27.6$$

Global phase effect has no effect proved below  
 $= 1 \cdot \cos\theta = \cos\theta$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

27.2)

27.2) we have three cases proved and we need to prove remaining three cases.

i)  $z = -1, x = 0, y = 0$  i.e.,  $\theta = \pi, \phi = 0$

It is also called South pole

$$|\Psi_{\theta=\pi, \phi=0}\rangle = \cos \frac{\theta}{2} e^{-i\phi/2}|0\rangle + \sin \frac{\theta}{2} e^{i\phi/2}|1\rangle$$

$$= \cos \frac{\pi}{2} e^{-i\phi/2}|0\rangle + \sin \frac{\pi}{2} e^{i(\phi)/2}|1\rangle$$

$$= 0 \cdot |0\rangle + 1 \cdot |1\rangle$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$\frac{\pi}{2}$   
 $\frac{\pi}{4}$   
 $\frac{\theta}{8}$

$|11\rangle$        $\frac{3\pi}{4}$

ii) consider  $x=1, y=2=0$

$\theta = \frac{\pi}{2}, \phi = 0$

$$|\psi_{\theta=\frac{\pi}{2}, \phi=0}\rangle = \cos \frac{\theta}{2} e^{-i\phi/2} |0\rangle + \sin \frac{\theta}{2} e^{i\phi/2} |1\rangle$$

$$= \cos \frac{\pi}{4} e^{-i0} |0\rangle + \sin \frac{\pi}{4} e^{i0} |1\rangle$$

$$= \frac{1}{\sqrt{2}} (|0\rangle + \frac{i}{\sqrt{2}} |1\rangle)$$

$$= |+\rangle$$

iv) consider  $y=-1, x=2=0$

$\theta = \frac{\pi}{2}, \phi = -\frac{\pi}{2}$

$$|\psi_{\theta=\frac{\pi}{2}, \phi=\frac{\pi}{2}}\rangle = \cos \frac{\pi}{4} e^{i\pi/4} |0\rangle + \sin \frac{\pi}{4} e^{-i\pi/4} |1\rangle$$

$$= \frac{1}{\sqrt{2}} e^{i\pi/4} (|0\rangle + e^{-i\pi/2} |1\rangle)$$

$$= e^{i\pi/4} \frac{1}{\sqrt{2}} (|0\rangle - i|1\rangle)$$

Here we are ignoring global phase in all scenarios

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

27.3)

27.3)

$$|\psi\rangle = \cos\theta/2 e^{-i\phi/2} |0\rangle + \sin\theta/2 e^{i\phi/2} |1\rangle$$

$$= \begin{pmatrix} \cos\theta/2 e^{-i\phi/2} \\ \sin\theta/2 e^{i\phi/2} \end{pmatrix}$$

i)  ~~$\sigma_2$~~

$$\langle \psi | \sigma_2 | \psi \rangle = (\cos\theta/2 e^{i\phi/2} \sin\theta/2 e^{-i\phi/2})$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \cos\theta/2 e^{-i\phi/2} \\ \sin\theta/2 e^{i\phi/2} \end{pmatrix}$$

$$= (\cos\theta/2 e^{i\phi/2} \sin\theta/2 e^{-i\phi/2}) \begin{pmatrix} \sin\theta/2 e^{i\phi/2} \\ \cos\theta/2 e^{i\phi/2} \end{pmatrix}$$

$$= \cos\theta/2 e^{i\phi/2} \sin\theta/2 e^{-i\phi/2} + \sin\theta/2 e^{-i\phi/2} \cos\theta/2 e^{i\phi/2}$$

$$e^{i\phi} \quad \sin \frac{i\phi}{2} - \cos$$

$$= \cos\theta/2 \sin\theta/2 \cdot e^{i\phi} + \sin\theta/2 \cos\theta/2 e^{i\phi}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$= \cos\theta/2 \sin\theta/2 (e^{i\phi} + e^{-i\phi})$$

$$= \cos\theta/2 \sin\theta/2 (\cos\phi/2 + i\sin\phi/2 + \cos\phi/2 - i\sin\phi/2)$$

$$= \cos\theta/2 \sin\theta/2 (\cos\phi)$$

$$= \sin\theta \cos\phi$$

$$\text{ii) } |\psi\rangle = \cos\theta/2 e^{i\phi/2} \sin\theta/2 e^{-i\phi/2} \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\begin{bmatrix} \cos\theta/2 e^{-i\phi/2} \\ \sin\theta/2 e^{i\phi/2} \end{bmatrix}$$

$$= (\cos\theta/2 e^{i\phi/2} \sin\theta/2 e^{-i\phi/2})$$

$$\begin{bmatrix} -i\sin\theta/2 e^{i\phi/2} \\ i\cos\theta/2 e^{-i\phi/2} \end{bmatrix}$$

$$= \cos\theta/2 e^{i\phi/2} - i\sin\theta/2 e^{i\phi/2} + \sin\theta/2 e^{-i\phi/2} i\cos\theta/2 e^{-i\phi/2}$$

\*

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

$$i \sin\theta/2 \cos\phi/2 e^{-i\phi} - i \cos\theta/2 \sin\phi/2 e^{i\phi}$$

$$= i \sin\theta/2 \cos\phi/2 (e^{-i\phi} - e^{i\phi})$$

$$= i \sin\theta/2 \cos\phi/2 (\cancel{\cos\phi} - i \sin\phi/2 \cdot \cancel{\cos\theta} - i \sin\phi)$$

$$= i \sin\theta/2 \cos\phi/2 (-i \sin\phi)$$

$$= -i^2 \sin\theta \sin\phi$$

$$= \sin\theta \sin\phi$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

### 27.4

27.4) Given  $|\Psi\rangle = \left(\frac{3}{4} - i\frac{\sqrt{3}}{4}\right)|0\rangle + \left(\frac{\sqrt{3}}{4} + i\frac{1}{4}\right)|1\rangle$

$$= \frac{\sqrt{3}}{2} \left(\frac{\sqrt{3}-i}{2}\right)|0\rangle + \frac{1}{2} \left(\frac{\sqrt{3}+i}{2}\right)|1\rangle$$

(1)

We Know From (27.2)

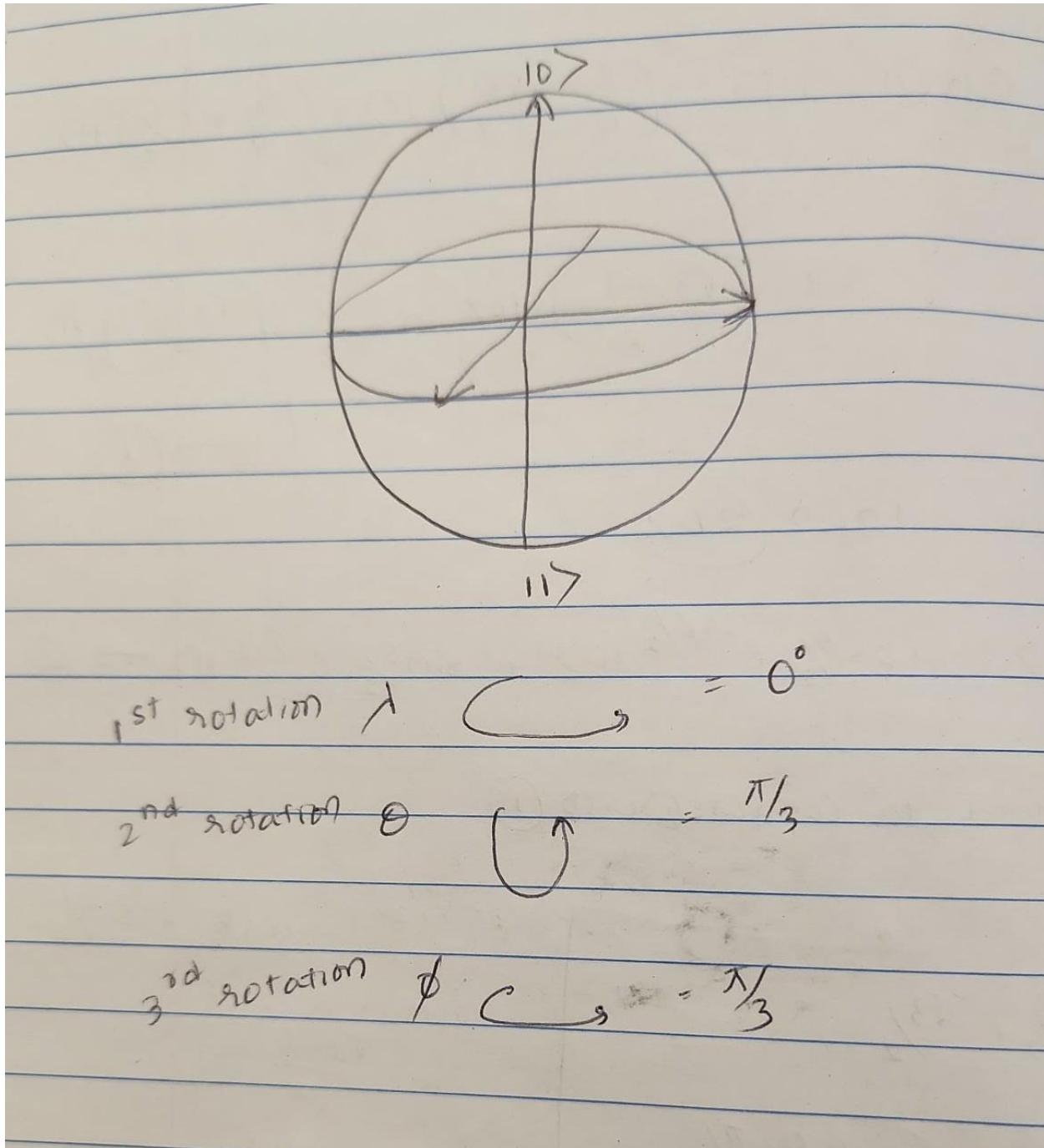
$$|\Psi\rangle = \cos\theta/2 e^{-i\phi/2}|0\rangle + \sin\theta/2 e^{i\phi/2}|1\rangle \rightarrow (2)$$

we need to compare (2) with (1)

$\cos\theta/2 = \frac{\sqrt{3}}{2}$ $\theta/2 = \pi/6 \Rightarrow \theta = \pi/3$ $\sin\theta/2 = 1/2$ $\theta/2 = \pi/6 \Rightarrow \theta = \pi/3$	$e^{-i\phi/2} = \cos\theta/2 - i\sin\theta/2$ $= \frac{\sqrt{3}-i}{2}$ $\frac{\sqrt{3}}{2} - i\frac{1}{2}$ $\cos\phi/2 = \frac{\sqrt{3}}{2} \quad \sin\phi/2 = 1/2$ $\phi/2 = \pi/3 \Rightarrow \phi = \pi/3$
---	---

$$|\Psi\rangle = \cos\pi/6 e^{-i\pi/6}|0\rangle + \sin\pi/6 e^{i\pi/6}|1\rangle$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

27.5)

27.5)

$$H(\psi) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$U_{\theta, \phi, \lambda} = U_{\text{Hadamard}}$

$$\begin{pmatrix} \cos\theta/2 & -e^{i\lambda} \sin\theta/2 \\ e^{i\phi} \sin\theta/2 & e^{i(\lambda+\phi)} \cos\theta/2 \end{pmatrix} = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

$$\cos\theta/2 = \frac{1}{\sqrt{2}}, \quad -e^{i\lambda} \sin\theta/2 = \frac{1}{\sqrt{2}}$$

$$e^{i\phi} \sin\theta/2 = \frac{1}{\sqrt{2}}, \quad e^{i(\lambda+\phi)} \cos\theta/2 = \frac{1}{\sqrt{2}}$$

$$\theta/2 = \frac{\pi}{4} \Rightarrow \theta = \frac{\pi}{2} \quad \text{we can derive } \lambda = \pi$$

$$\text{and } -e^{i\lambda} = 1 \Rightarrow \phi = 0$$

$$\Rightarrow U_{\theta, \phi, \lambda} = U_{\pi/2, 0, \pi}$$

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

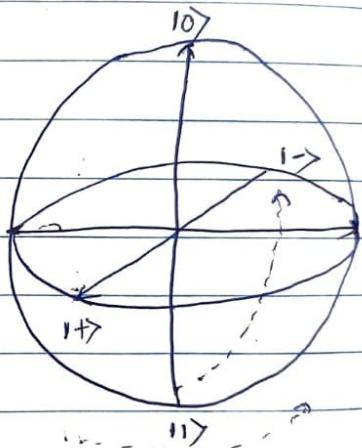
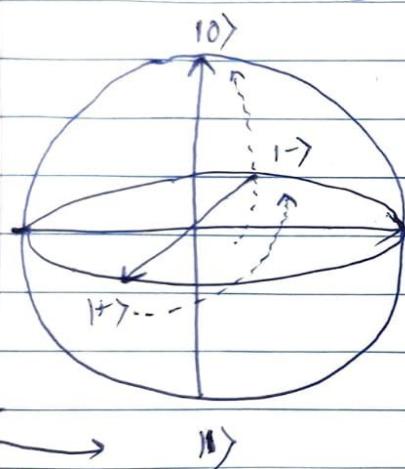
27.6)

27.6) Hadamard applied to  $|+\rangle$  to  $|1\rangle$

We know Hadamard's  $U\theta, \phi, \lambda$  is  $U_{\pi/2}, 0, \pi$

$$H \rightarrow |+\rangle \rightarrow |0\rangle$$

$$H \rightarrow |+\rangle \rightarrow |-\rangle$$



1<sup>st</sup> rotation  $\lambda = \pi$

2<sup>nd</sup> rotation  $\theta = \pi/2$

3<sup>rd</sup> rotation  $\phi = 0$

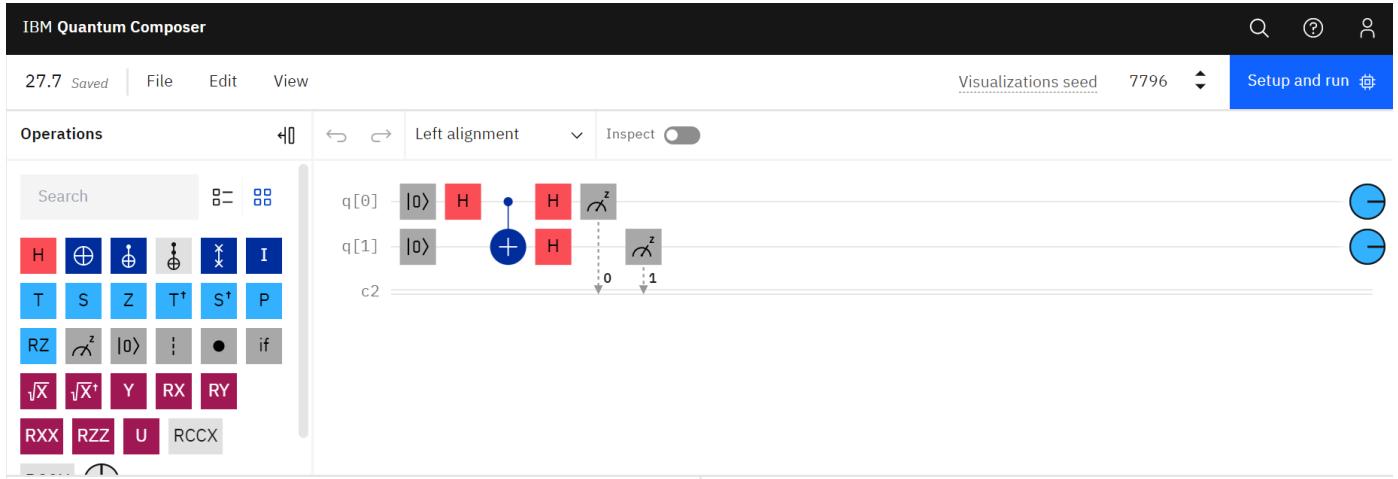
1<sup>st</sup> rotation  $\lambda = \pi$  (No use)

2<sup>nd</sup> rotation  $\theta = \pi/2$

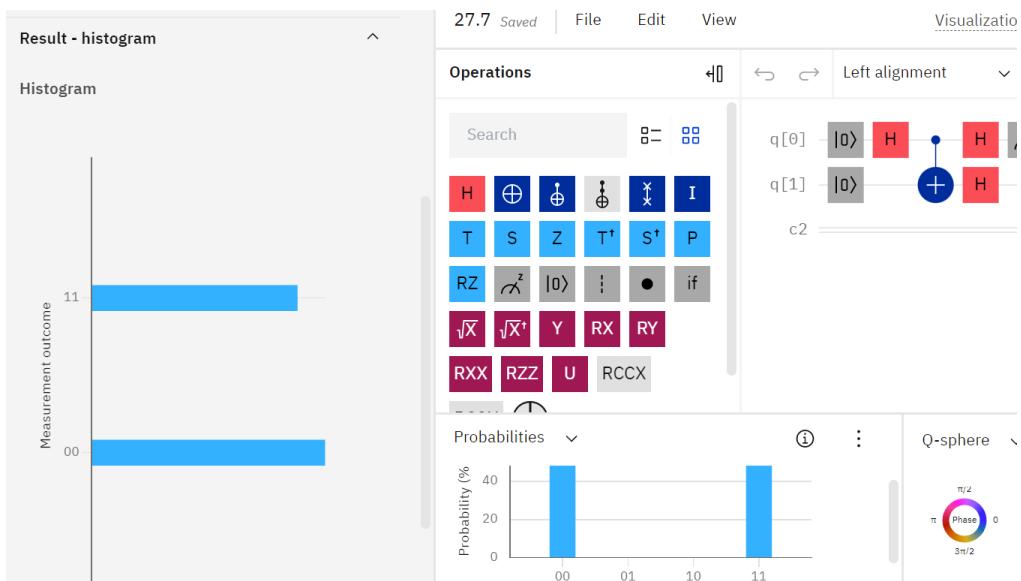
3<sup>rd</sup> rotation  $\phi = 0$

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

27.7



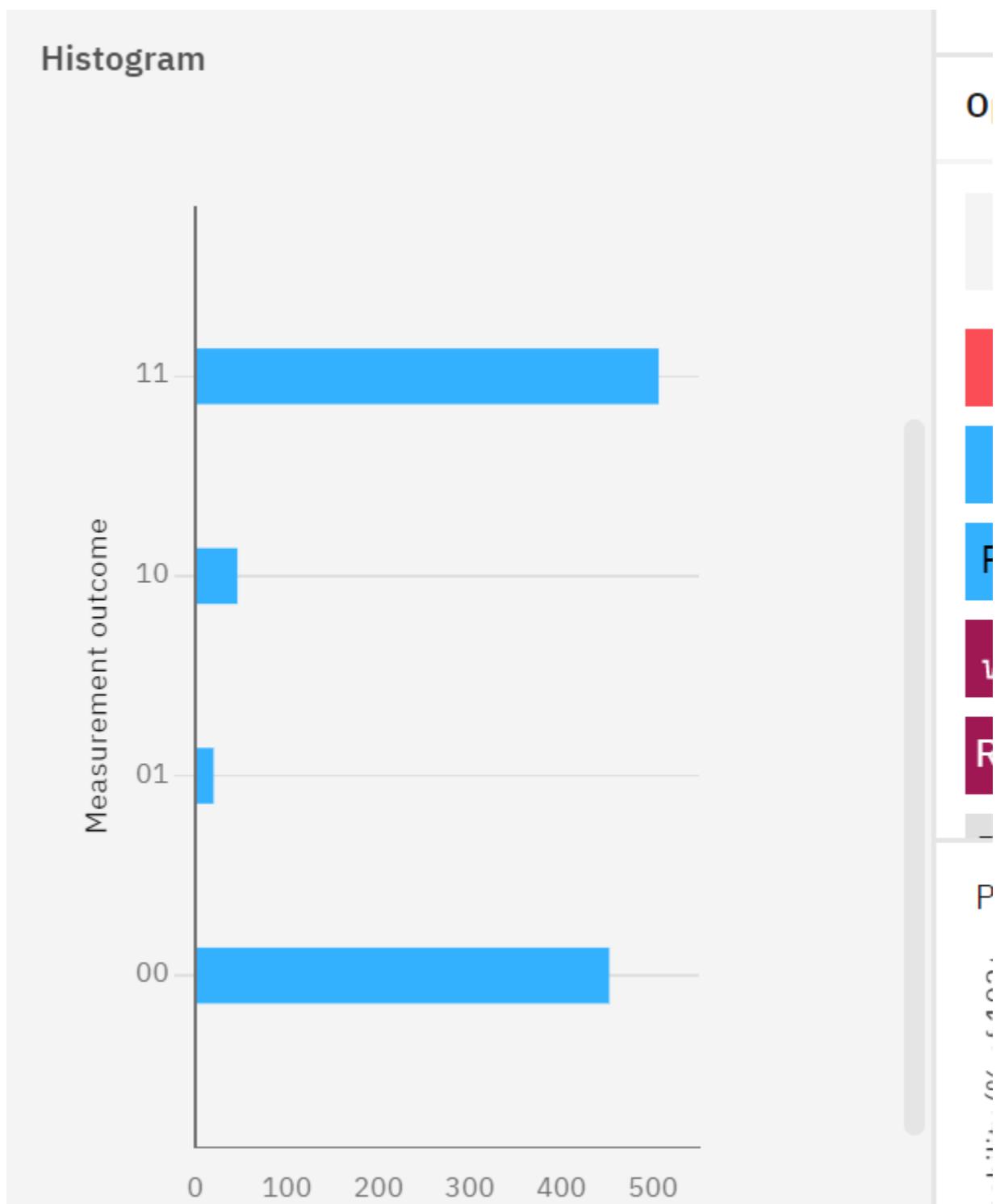
## Result



## Hardware execution result

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



Chapter 28

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

28.1)

Given  $\frac{1}{\sqrt{2}} \begin{pmatrix} -i & -i \\ -i & i \end{pmatrix}$

we can extract  $-i$  from the above gate it becomes hadamard gate

$$-i \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$$-i \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} = e^{i\lambda} \begin{pmatrix} \cos \theta/2 & -e^{i\lambda} \sin \theta/2 \\ e^{i\lambda} \sin \theta/2 & e^{i(\lambda+\phi)} \cos \theta/2 \end{pmatrix}$$

$$\cos \theta/2 = \frac{1}{\sqrt{2}} \Rightarrow \theta/2 = \frac{\pi}{4} \quad [\theta = \frac{\pi}{2}]$$

$$= e^{-i\lambda} \sin \theta/2 \Rightarrow \boxed{\sin \theta/2 = \frac{1}{\sqrt{2}}} \Rightarrow -e^{-i\lambda} \Rightarrow \lambda = \pi$$

$$e^{i\phi} \sin \theta/2 = e^{i\phi} \Rightarrow \phi = 0$$

$$e^{i\gamma} = -i \Rightarrow -(\cos \gamma + i \sin \gamma) = \gamma = \frac{\pi}{2}$$

$$\boxed{U_{\theta, \phi, \lambda, \gamma} = U_{\frac{\pi}{2}, 0, \pi, \frac{\pi}{2}}}$$

28.2)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

28.2

Given  $C - U_{\theta, \phi, \lambda, \gamma}$

$$\begin{matrix} & & 00 & 01 & 10 & 11 \\ 00 & 1 & 0 & 0 & 0 & 0 \\ 01 & 0 & 1 & e^{i\theta} \cos \gamma & e^{-i(\lambda+\gamma)} \sin \theta / 2 & 0 \\ 10 & 0 & 0 & e^{i(\phi+\gamma)} \sin \theta / 2 & e^{i(\lambda+\phi+\gamma)} \cos \theta / 2 & 0 \\ 11 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

From 28.7 we can derive

We know :-

$$c_{2,3} = \langle 111 | 101 | C - U_{\theta, \phi, \lambda, \gamma} | 111 \rangle = -e^{i(\lambda+\gamma)} \sin \theta / 2$$

which is in  $|110\rangle$   
basis  
after inner product

$$c_{3,3} = \langle 111 | 111 | C - U_{\theta, \phi, \lambda, \gamma} | 111 \rangle = e^{i(\lambda+\phi+\gamma)} \cos \theta / 2$$

$$c_{2,2} = \langle 101 | 101 | C - U_{\theta, \phi, \lambda, \gamma} | 101 \rangle = e^{i\gamma} \cos \theta / 2$$

$$c_{3,2} = \langle 101 | 111 | C - U_{\theta, \phi, \lambda, \gamma} | 101 \rangle = e^{i(\phi+\lambda)} \sin \theta / 2$$

28.3)

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

28.3) Given

$$C = U_{\theta, \phi, \lambda} V = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U_{\theta, \phi, \lambda}$$

$$|0\rangle\langle 0| \Rightarrow |0\rangle\langle 0| \Rightarrow \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$|1\rangle\langle 1| \Rightarrow |1\rangle\langle 1| \Rightarrow \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \cdot e^{i\gamma} \begin{bmatrix} \cos \frac{\theta}{2}, e^{i\frac{\phi}{2}} \\ e^{i(\frac{\theta+\phi}{2})} \sin \frac{\theta}{2}, e^{i(\frac{\lambda+\theta+\phi}{2})} \cos \frac{\theta}{2} \end{bmatrix}$$

Anything multiplied with Identity is same

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & e^{i\gamma} \cos \frac{\theta}{2} & e^{i(\lambda\pi)} \sin \frac{\theta}{2} \\ 0 & 0 & e^{i(\phi+\gamma)} \sin \frac{\theta}{2} & e^{i(\lambda+\theta+\phi)} \cos \frac{\theta}{2} \end{bmatrix}$$

$$e^{i\gamma} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & e^{i\gamma} \cos \frac{\theta}{2} & e^{i(\lambda\pi)} \sin \frac{\theta}{2} \\ 0 & 0 & e^{i(\phi+\gamma)} \sin \frac{\theta}{2} & e^{i(\lambda+\theta+\phi)} \cos \frac{\theta}{2} \end{bmatrix}$$

28.4)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

28.4)

We know

$$U_{ps, \pi} |e_1\rangle = +1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} = +1 |e_0\rangle$$

$$Z |e_0\rangle = e^{i2\pi \cdot 0} |e_0\rangle = |e_0\rangle$$

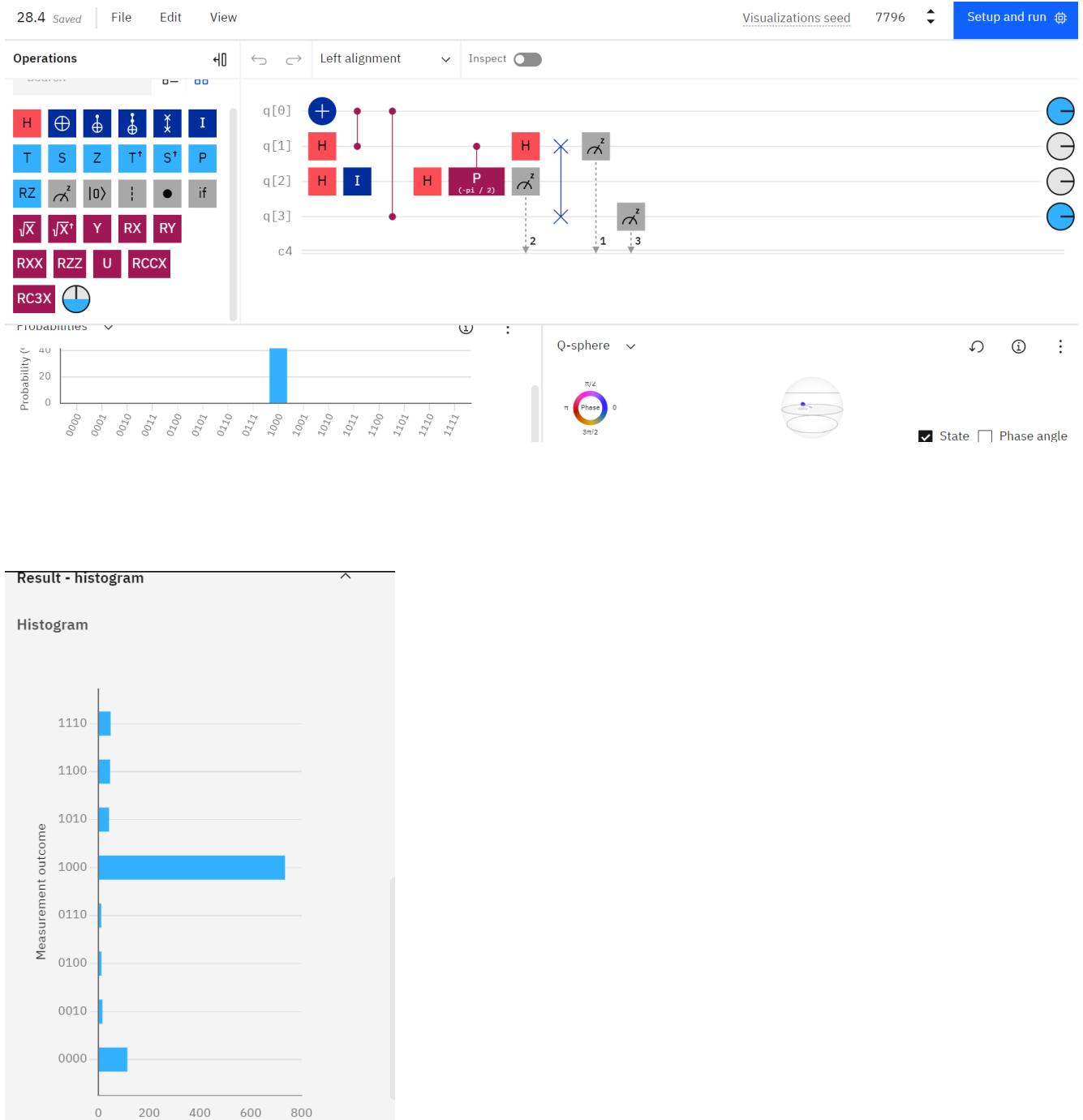
$$Z^2 |e_0\rangle = (e^{i2\pi \cdot 0})^2 |e_0\rangle = |e_0\rangle$$

$$Z = e^{i \cdot 2\pi \cdot 0} \quad \text{i.e., } e^0 = \cos 0 + i \sin 0 = 1$$

If control Qubit is 0  $\otimes |e_0\rangle$  it does nothing

28.5)

# Introduction to Quantum computing ( From a layperson to programmer in 30 steps)



Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

28.6)

Equation in  $2\pi\psi = 2\pi k / 2^n$  is proved  
in 28.3.2

$$2\pi\psi = 2\pi k / 2^n$$

$$\text{Here } |k\rangle = |10\rangle = |2\rangle$$

$$2\pi\psi = 2\pi k / 2^n = 2\pi \cdot 2 / 4^2 = \pi$$

which has <sup>been</sup> proved

Chapter 29

Nithin Reddy Govindugari(nithinreddy1747@gmail.com)

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

29.1) By using method in (29.2)  
 $177 \bmod 25$

$$177 = 25 \times 7 + 2$$

i.e., 2

29.2 we can find GCD by euclidean method

firstly we have  $1242 \times 242$

i) Divide 1242 by 242 and we get  
remainder 32

ii) Divide 242 by 32 (until last value)  
 $242 \div 32 \rightarrow$  remainder is 18

iii) Divide 32 by 18  
 $32 \div 18 \rightarrow$  remainder 14

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

iv) Divide 18 by 14

$18 \div 14 \rightarrow$  remainder is 4  $\neq 0$

v) Divide 18 by 4

$18 \div 4 \rightarrow$  remainder is 2  $\neq 0$

vi) Divide 4 by 2  $\rightarrow$  remainder is 0



This will become GCD as remainder is zero ( value of GCD  $\rightarrow 2$  )

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

29.3)

```
# Shors simplified algorithm in python but cant handle big prime numbers
#29.3
# gcf is calculated
def get_gcf(a, b):
    while a != b:
        if a > b:
            a -= b
        elif b > a:
            b -= a
    return a

# I have used below integers as a in our text book we can add any number of numbers

# 12 is not suitable a so i have used incrementation to get better a

Randomint = [8, 10, 12]

for g in Randomint:
    N = 63
    done = False
    p = 0
    p_values = []
    while not done:
        p += 1
        r = g**p % N
        if g**p % N == 1:
            print(g, p)
            p_values.append(p)
        if len(p_values) >= 1:
            p = p_values[0]
            if p_values[0] % 2 != 0:
                g += 1
                p = 0
                p_values = []
            else:
                x = g**(p/2)+1
                y = g**((p/2)-1)
                a = int(get_gcf(N, x))
                b = int(get_gcf(N, y))
                if a == 1 or b == 1:
                    g += 1
                    p = 0
                    p_values = []
                else:
                    print(g, p, a, b)
                    done = True
```

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

```
if p > 500:  
    g += 1  
    p = 0  
    p_values = []  
  
print(a, "x", b, "=", N)
```

) 8 2  
8 2 9 7  
9 x 7 = 63  
10 6  
10 6 7 9  
7 x 9 = 63  
13 6  
13 6 7 9  
7 x 9 = 63

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

29.4)

29.4) Here we have four steps to do before measurement and as informed it is practice for substitution

For the last question where it is asked that if  $n=1$  will be possible and according to my understanding it is not possible because as explained in textbook 29.3 [page No 282] we need have  $n$  where  $2^n \geq n$  if  $n=1$   $2^1 \geq 1$  so we need to have  $N$  which is less than  $\frac{2^n}{2}$  which is not appropriate and i estimate we need to have atleast  $2^{n+2}$  qubits  $\rightarrow$  means  $n$  should be atleast 2

Step 1

$$|\psi\rangle = (H^{\otimes 2n} I^{\otimes n}) (|0\rangle^{\otimes 2n} |0\rangle^{\otimes n})$$

$$= \frac{1}{2^n} \sum_{x=0}^{2^n-1} |x\rangle_{2n} |0\rangle_n$$

If  $n=1$

$$= \frac{1}{2} \sum_{x=0}^3 |x\rangle_2 |0\rangle \rightarrow ①$$

we can expand it to

$$= \frac{1}{2} [ |00\rangle + |01\rangle + |10\rangle + |11\rangle ]$$

But we are keeping it as ① for simplification

= 1

## Introduction to Quantum computing ( From a layperson to programmer in 30 steps)

Step 2 Applying Quantum oracle & evaluating function  
 $|E_2\rangle = v_f |E_1\rangle$

From 29.11

$$|E_2\rangle = \frac{1}{\sqrt{2}} \sum_{x=0}^3 |13\rangle_2 |f(x)\rangle_3$$

Step 3

$$U_{\text{Aff}} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \omega^{-kx} |k\rangle \quad (k \text{ is row}) \quad (x \text{ is column})$$

$$|E_3\rangle = (U_{\text{Aff}} \otimes I^{10}) |E_2\rangle$$

From 29.13

$$= \frac{1}{\sqrt{2^n}} \sum_{x=0}^3 \sum_{k=0}^3 \omega^{-kx} |k\rangle_2 |f(x)\rangle_3$$

Step 4

It is measurement on the least significant bit

which we can ignore according to the question asked

It is not possible to have 1 in my opinion as we need to have at least 5 qubits to execute Shors Algorithm

# **Introduction to Quantum computing ( From a layperson to programmer in 30 steps)**