Quantum Information and Computation

Qiskit Coding Term Paper on:

Bit-Flip and Phase-Flip Error Correction Coding using Three Qubit Code & Inner Products

written by:
Aniket
M.Tech. Optoelectronics and Optical Communication
Indian Institute of Technology Delhi

I. ABSTRACT & MOTIVATION

Three-qubit quantum error correction codes, such as the bit-flip code and phase-flip codes, represent foundational strategies for detecting and correcting errors in small-scale quantum systems. These codes, utilizing an ancilla qubit entangled with the original qubit, enable the identification and correction of specific errors without direct state measurement. While simple, three-qubit codes serve as essential building blocks in the development of more intricate quantum error correction schemes, contributing to the understanding and implementation of fault-tolerant quantum computation.

I was motivated to work on this topic because I am already working on Error-Correction Coding using Low-Density-Parity-Check (LDPC) Codes and Hybrid-Automatic-Repeat-reQuest (HARQ) protocol for Free Space Optical Communication (FSO) as my M.Tech. Major Project. Consequently, I have naturally had a keen interest in Quantum Error-Correction Coding as well. Hence, we are going to discuss **Bit-Flip and Phase-Flip Error Correction Coding using Three Qubit Code** in this assignment.

Finally, to analyse the error-correction, **we calculate the inner product of the corrected state with the original state** to find out the similarity between them.

II. THEORY

A) BIT-FLIP ERROR CORRECTION CODING

In three qubit bit-flip error-correction coding, we initially create an arbitrary state $|\Psi\rangle$ which is to be transmitted and is given as,

$$|\Psi\rangle_0 = \alpha|0\rangle + \beta|1\rangle$$

A single qubit is encoded using the circuit in Fig. 1 in three carrier qubits. As a result, $|\Psi\rangle$ at time, t, (after red block which denotes the encoding circuit) becomes,

$$|\Psi\rangle = \alpha|000\rangle + \beta|111\rangle$$

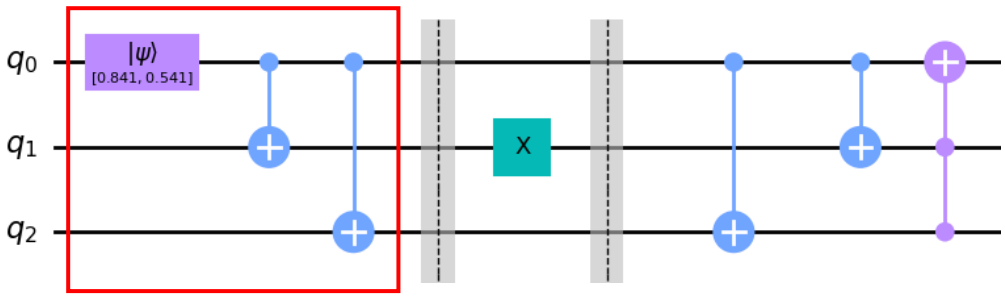


Fig. 1 Quantum Circuit highlighting the Encoder Block

Three-qubit encoding and decoding circuit corrects an X error (or Bit-Flip error) on a single carrier if it occurs at a time between the barriers (shown as grey dashed lines).

Let us now suppose that between the barriers, a bit-flip error might occur on the first or second or third carrier, but not on more than one carrier. That is, there is at most one (possible) corrupted qubit, but we do not know which one has been corrupted. The result will be one of the four possibilities,

$$|\Psi\rangle = \alpha|000\rangle + \beta|111\rangle \qquad |\Psi\rangle = \alpha|100\rangle + \beta|011\rangle$$
$$|\Psi\rangle = \alpha|010\rangle + \beta|101\rangle \qquad |\Psi\rangle = \alpha|001\rangle + \beta|110\rangle$$

Note that these four possibilities correspond, as α and β are varied, to four mutually orthogonal subspaces. Thus, the information of interest to us, the ratio β/α, has not really disappeared. It is just hiding. So, we need to locate its hiding place and extract it.
To simulate a random bit-flip error in one of the qubits, we make the following circuit in which X-gate is applied randomly between one of the three qubits.

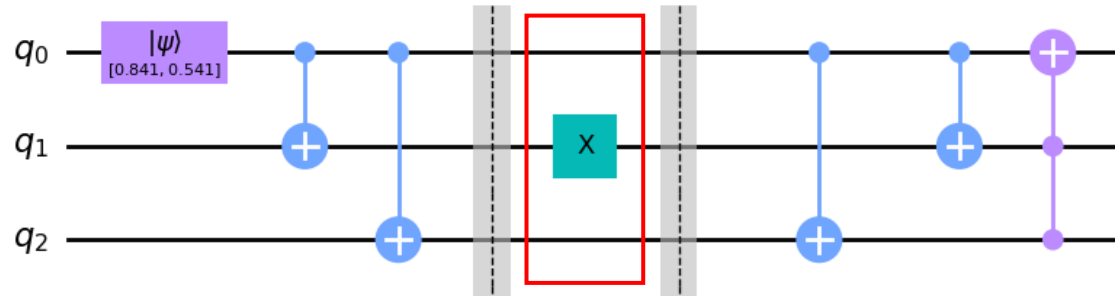


Fig. 2 Quantum Circuit highlighting the Bit-Flip Block

This position of X-gate is decided randomly using the code which is given in the next section. Thus, after the 2nd barrier, we get one of the four possible states as mentioned above.

Now, in the decoder section, these following operations are done as part of the decoding circuit, which is shown in the red block as follows,

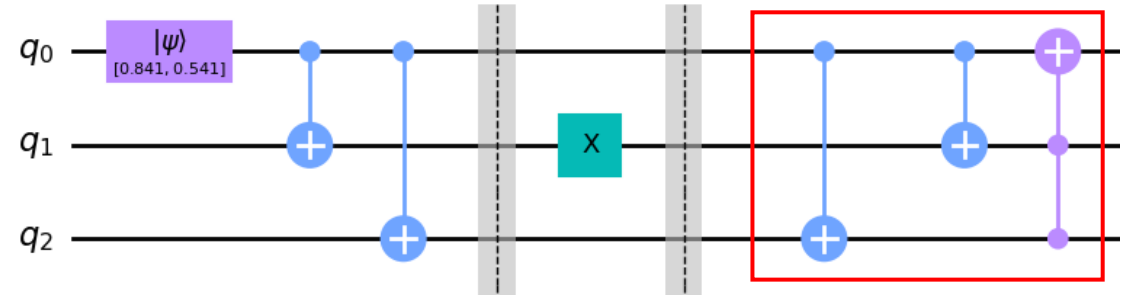


Fig. 3 Quantum Circuit highlighting the Decoder Block

After this, theoretically, we get the original state back as,

$$|\Psi\rangle_C = \alpha|0\rangle + \beta|1\rangle$$

But **on a real quantum computer**, we do not get the perfect state back as there is quantum noise in the circuit. Hence, as we do not get the original state ($|\Psi\rangle_O$) back perfectly. We need to use controlled-swap (CSWAP) circuit (shown below) to calculate the inner product of the corrected state ($|\Psi\rangle_C$) and the original state ($|\Psi\rangle_O$), which tells us how close the corrected state ($|\Psi\rangle_C$) to the original state ($|\Psi\rangle_O$) is. Hence, here we have to add two extra bits, **q3 and q4 as Ancilla Bits**.

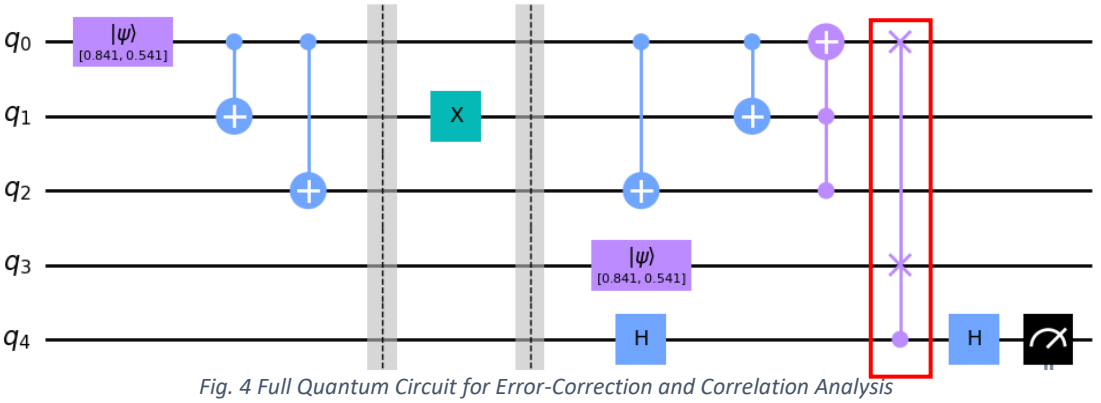


Fig. 4 Full Quantum Circuit for Error-Correction and Correlation Analysis

Moreover, on real quantum computer, we do not have to use random X-gate application on one of the three qubit carrier because we anyway get quantum noise on application of various gates. So, while running the circuit on real quantum computer, we use the following circuit.

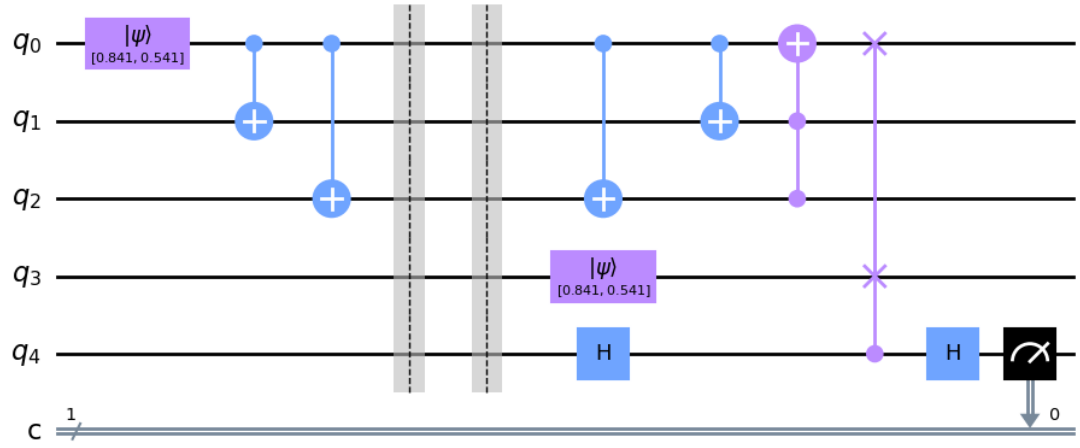


Fig. 5 Circuit used for Real Quantum Computer

More about Controlled SWAP Gate:

We can use the SWAP test to determine the inner product of 2 states $|\phi\rangle$ and $|\psi\rangle$. The circuit is shown below,

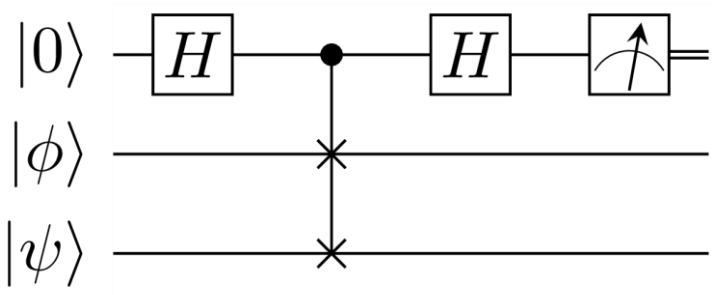


Fig. 6 Controlled SWAP Gate to compute any inner product
https://quantumcomputing.stackexchange.com/questions/12797/quantum-circuit-to-compute-any-inner-product

The state of the system at the beginning of the protocol is $|0\rangle\otimes|\phi\rangle\otimes|\psi\rangle$.

After the Hadamard gate, the state of the system is $|+\rangle\otimes|\phi\rangle\otimes|\psi\rangle$.

The controlled SWAP gate transforms the state into, $1/\sqrt{2}\ (\ |0\rangle\otimes|\phi\rangle\otimes|\psi\rangle + |1\rangle\otimes|\psi\rangle\otimes|\phi\rangle\ )$.

The second Hadamard gate results in the following,

$$= 1/2\ (\ |0\rangle|\phi\rangle|\psi\rangle + |1\rangle|\phi\rangle|\psi\rangle + |0\rangle|\psi\rangle|\phi\rangle - |1\rangle|\psi\rangle|\phi\rangle)$$

$$= 1/2\ (|\phi\rangle|\psi\rangle+|\psi\rangle|\phi\rangle)\ |0\rangle + 1/2\ (|\phi\rangle|\psi\rangle - |\psi\rangle|\phi\rangle)\ |1\rangle$$

Now, the measurement gate on the first qubit ensures that it's 0 with a probability of,

$$P\ (1^{st}\ \text{qubit} = 0) = 1/2\ \ (\langle\phi|\langle\psi| + \langle\psi|\langle\phi|)\ 1/2\ (|\phi\rangle|\psi\rangle + |\psi\rangle|\phi\rangle) = 1/2 + 1/2\ |\langle\psi|\phi\rangle|^2$$

When measured, if the inner product of the two state is 1 (which means the two states are the same), the probability of 1^{st} qubit = 0, comes out to 1 (100%). However, if the inner product is zero, then

we get the probability as 1/2, which denotes orthogonal states. In any other case, we get the probability as somewhere between 1/2 and 1 which in our circuit shows that the similarity of the corrected and original states.

B) PHASE-FLIP ERROR CORRECTION CODING

Phase-flip Error corresponds to errors against Z-gate. We use the following form to represent errors in phase in a given state, which is initially given as,

$$|\Psi\rangle_0 = \alpha|+\rangle + \beta|-\rangle$$

After the encoding process, we get the following state,

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

Now, in case of phase flip errors, we interchange the role of X gate with Z-gate which introduces a phase-flip when it operates on either of the states $|+\rangle$ or $|-\rangle$ and flips one state into the other one. Thus, A simple way of constructing the coding and decoding circuit in this case is shown in Fig. 7, obtained by adding Hadamard Gates at strategic points to the circuit as below,

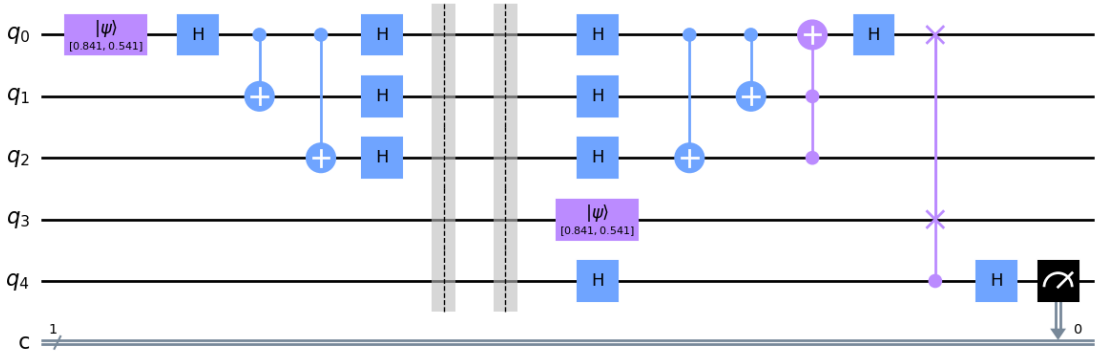


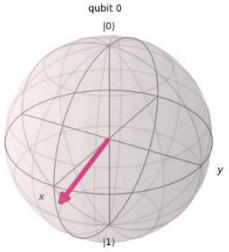
Fig. 7 Quantum Circuit for Correcting Phase-Flip Errors

The rest of the process is similar to the Bit-flip Error Correction circuit. Here as well, when we run this circuit on a real Quantum Computer, we do not get back the original state ($|\Psi\rangle_0$) back perfectly due to the presence of Quantum Noise and therefore, here as well, we use the Quantum-SWAP gate/circuit to calculate the inner product of the corrected state ($|\Psi\rangle_C$) with the original state ($|\Psi\rangle_0$). Thus, using these two circuits, we can correct both Bit-Flip and Phase-Flip errors which are common errors in Quantum Communication Circuits.

III. CODE, RESULTS & DISCUSSION

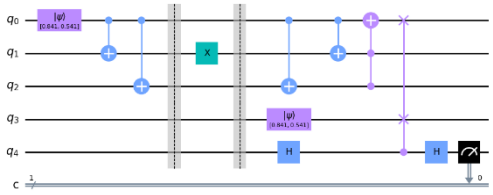
a) Preparing an arbitrary State $|\Psi\rangle$ in Computational Basis

```
from qiskit import QuantumCircuit, transpile, assemble, Aer, IBMQ, execute
from qiskit.visualization import plot_histogram, circuit_drawer,plot_bloch_multivector
from qiskit.quantum_info import Statevector
from qiskit.providers.aer.noise import NoiseModel, depolarizing_error
import matplotlib.pyplot as plt
import numpy as np

alpha = np.random.rand()
beta = np.sqrt(1-alpha**2)

psi = Statevector([alpha, beta])

plot_bloch_multivector(psi)
```



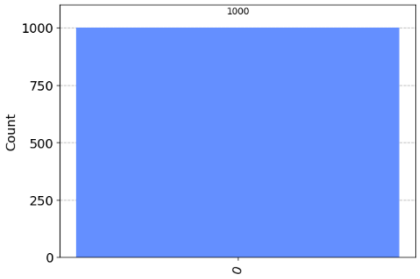
b) Building the Quantum Circuit for Random Bit-Flip Error Correction

```
qc =  QuantumCircuit(5,1)
qc.initialize(psi,0)
qc.cx(0,1)
qc.cx(0,2)
qc.barrier()
qbit = np.random.randint(0,3)
print(qbit)
qc.x(qbit)
qc.barrier()
qc.cx(0,2)
qc.cx(0,1)
qc.ccx(2,1,0)
qc.initialize(psi,3)
qc.h(4)
qc.cswap(4,3,0)
qc.h(4)
qc.measure(4,0)
qc.draw('mpl')
```



c) Running Simulation for Bit-Flip Error Correction Circuit

```
backend_sim = Aer.get_backend('qasm_simulator')

# We've set the number of repeats of the circuit to be 1000.
job_sim = backend_sim.run(transpile(qc, backend_sim), shots=1000)

# Grab the results from the job.
result_sim = job_sim.result()
counts = result_sim.get_counts(qc)
print(counts)
from qiskit.visualization import plot_histogram
plot_histogram(counts)
```



Here, as explained in the Controlled swap gate section, this information is coming after measuring the last qubit which gives the squared inner product of the corrected state ($|\Psi\rangle_C$) with the original state ($|\Psi\rangle_0$). The above result shows that at the end of the circuit, the original state ($|\Psi\rangle_0$) has been reproduced as theoretically we expect no errors. (explained in detail in the next section)

d) Running Bit-Flip Error Correction Circuit on Real Quantum Computer

```
from qiskit.providers.ibmq import least_busy
# Load IBM Quantum account
provider = IBMQ.load_account()
provider = IBMQ.get_provider(hub= 'ibm-q')
backend = provider.get_backend('ibm_nairobi')
# Transpiling the quantum circuit for the chosen backend
tqc = transpile(qc, backend, optimization_level=3)
# Run the quantum circuit on the chosen backend
job = backend.run(tqc, shots = 1000)
```

```
# Monitor the job status
from qiskit.tools.monitor import job_monitor
job_monitor(job)

# Retrieve and plot the results
result = job.result()
counts = result.get_counts()
print(counts)
plot_histogram(counts)
```

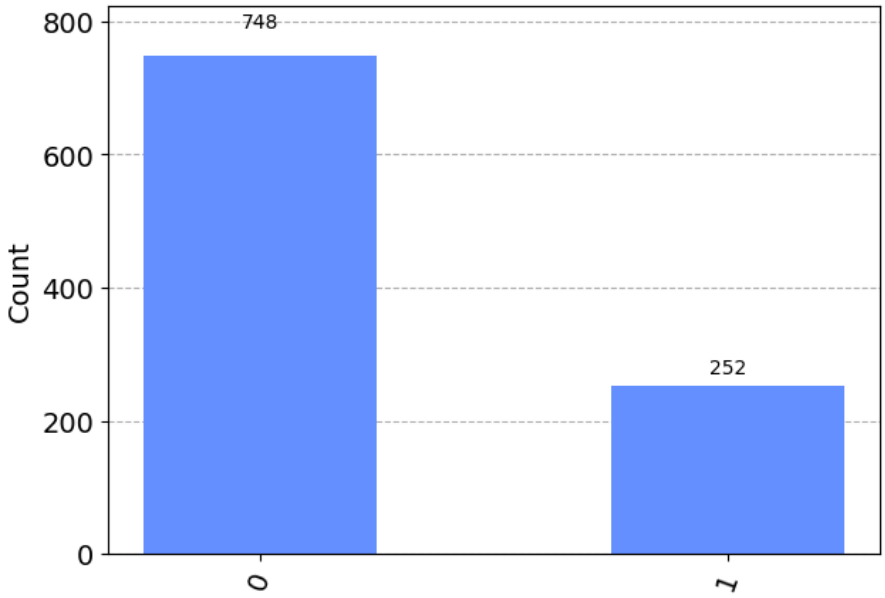


Fig. 8 Results for Bit-Flip Error Correction Circuit (Real Quantum Computer)

Here, we can see that due to Quantum Errors, we are not able to get the initial state back in its original form and instead get a state which has some finite errors as shown from the results of the squared inner product.

From the histogram, we can see that,

$$P\ (5^{th}\ \text{qubit} = 0) = 748/1000 = \mathbf{0.748}$$

Calculating the inner product,

If we use the back calculations from the Controlled SWAP Gate, we can calculate the inner product of the two corrected and the original state as follows,

$$1/2 + 1/2\ |\langle\psi_c|\ \psi_o\rangle|^2 = 0.748$$

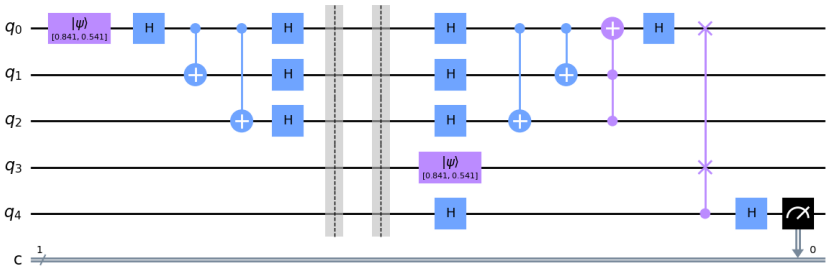
$$|\langle\psi_c|\ \psi_o\rangle|^2 = 0.496$$

$$|\langle\psi_c|\ \psi_o\rangle| = 0.704$$

which is the inner product of $|\Psi\rangle_{\text{Corrected}}$ and $|\Psi\rangle_{\text{Original}}$ as **0.704**, which in a way, represents the similarity of the corrected state with the original state. Thus, due to quantum errors, we are not able to correct the errored state back into its original form.

e) Building the Quantum Circuit for Random Phase-Flip Error Correction

```
qc =  QuantumCircuit(5,1)
qc.initialize(psi,0)
qc.h(0)
qc.cx(0,1)
qc.cx(0,2)
qc.h(0)
qc.h(1)
qc.h(2)
qc.barrier()
qc.barrier()
qc.h(0)
qc.h(1)
qc.h(2)
qc.cx(0,2)
qc.cx(0,1)
qc.ccx(2,1,0)
qc.h(0)
qc.initialize(psi,3)
qc.h(4)
qc.cswap(4,3,0)
qc.h(4)
qc.measure(4,0)
qc.draw('mpl')
```



f) Running Simulation for Phase-Flip Error Correction Circuit

```
backend_sim = Aer.get_backend('qasm_simulator')

# We've set the number of repeats of the circuit to be 1000.
job_sim = backend_sim.run(transpile(qc, backend_sim), shots=1000)

# Grab the results from the job.
result_sim = job_sim.result()
counts = result_sim.get_counts(qc)
print(counts)
from qiskit.visualization import plot_histogram
plot_histogram(counts)
```

Similar to the Bit-Flip Correction Circuit, this result shows that after applying the entire circuits the initial state is reproduced in its original form without any errors because on simulation, we do not expect any presence of quantum errors and we get a perfectly error-corrected state ($|\Psi\rangle_C$) which is same as the original state ($|\Psi\rangle_0$).

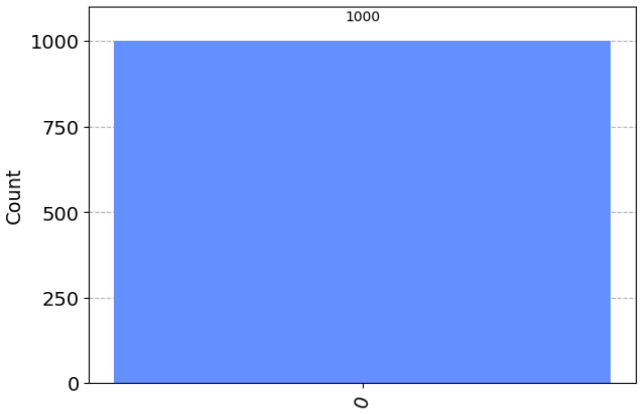


Fig. 9 Simulation Results

g) Running Phase-Flip Error Correction Circuit on Real Quantum Computer

```
from qiskit.providers.ibmq import least_busy

# Load IBM Quantum account
provider = IBMQ.load_account()

provider = IBMQ.get_provider(hub= 'ibm-q')
backend = provider.get_backend('ibm_nairobi')

# Transpile the quantum circuit for the chosen backend
tqc = transpile(qc, backend, optimization_level=3)

# Run the quantum circuit on the chosen backend
job = backend.run(tqc, shots = 1000)

# Monitor the job status
from qiskit.tools.monitor import job_monitor
job_monitor(job)

# Retrieve and plot the results
result = job.result()
counts = result.get_counts()
print(counts)
plot_histogram(counts)
```

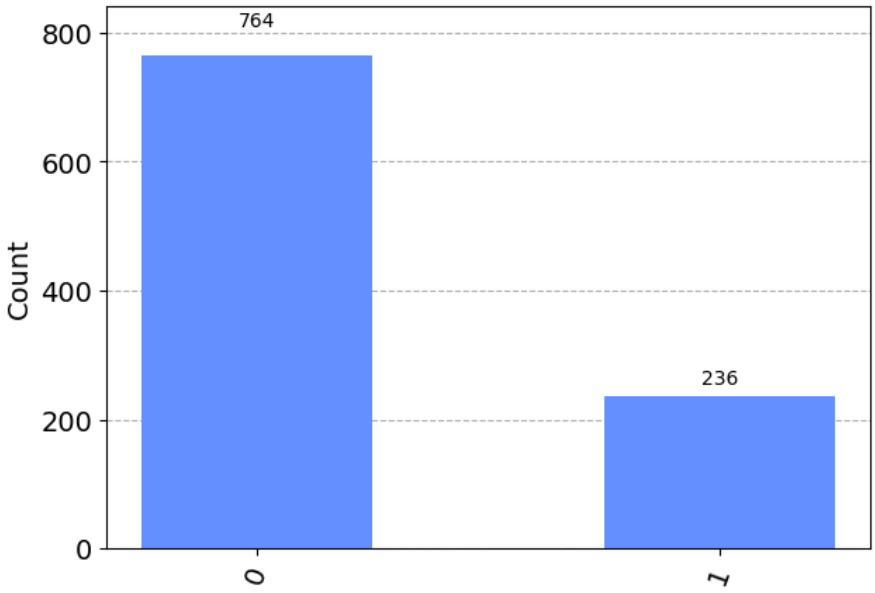


Fig. 10 Results for Phase-Flip Error Correction Circuit (Real Quantum Computer)

From the histogram, we can see that,

$$P\ (5^{th}\ \text{qubit} = 0) = 764/1000 = \mathbf{0.764}$$

Now, similar to Bit-Error Correction Case, we calculate the inner product of the corrected and the original state as, $|\langle\psi_c|\ \psi_o\rangle| = \mathbf{0.727}$.

IV. DETAILS OF THE QUANTUM COMPUTER/SERVER USED

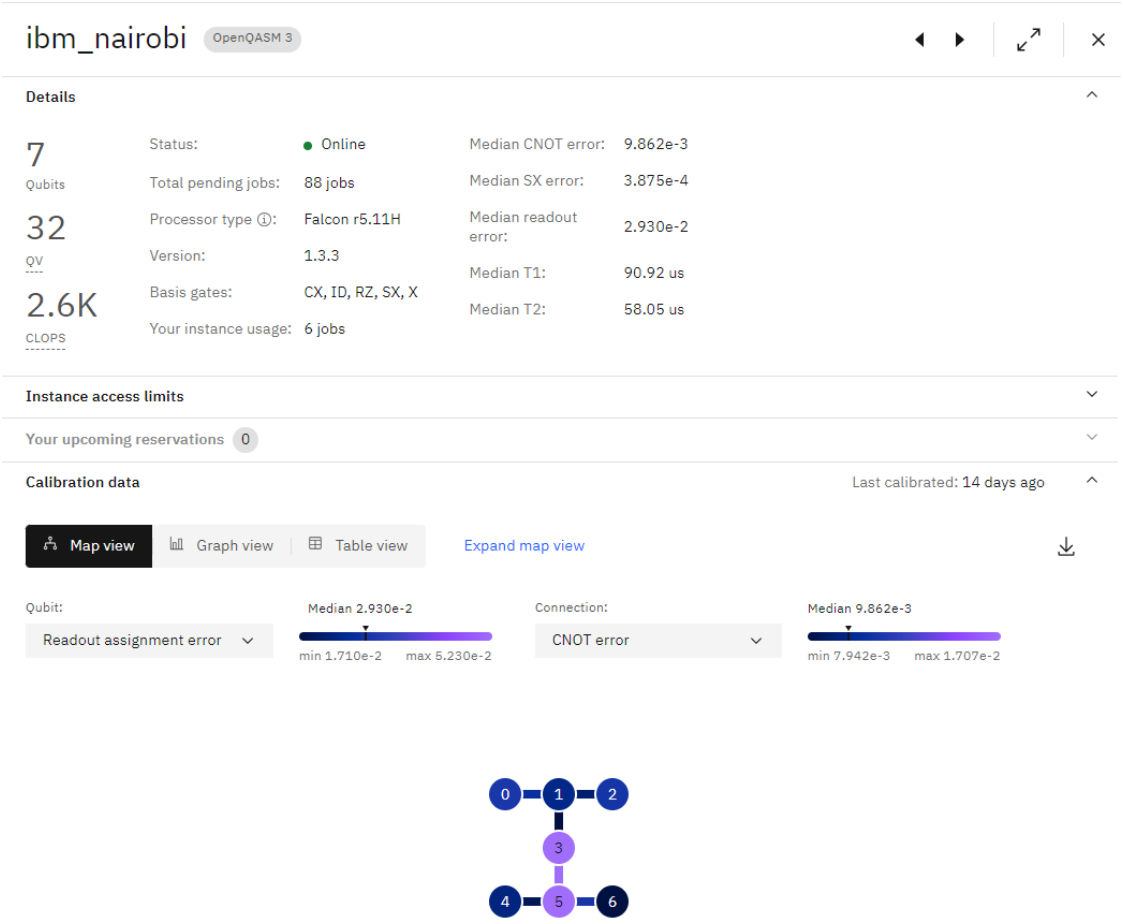


Fig. 11 Details of the IBM Quantum Computer used

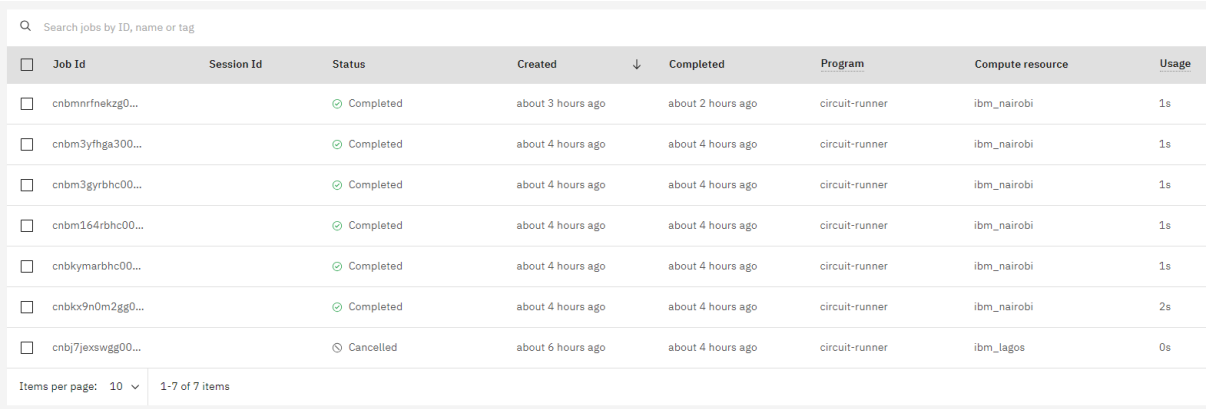


Fig. 12 Details of the Real Hardware Run Trials