ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

Scuola di Scienze
Dipartimento di Fisica e Astronomia
Corso di Laurea in Fisica

Experimental analysis of quantum error correction through IBM Quantum

Relatrice:
Prof.ssa Elisa Ercolessi

Presentata da:
Pietro Malagoli

Anno Accademico 2022/2023

Abstract

L'informatica quantistica è un campo di studio nuovo e promettente, ma la fragilità intrinseca dei qubit pone grandi difficoltà alle sue implementazioni. La *Quantum Error Correction* (QEC) offre protocolli per mitigare questi effetti e aprire la strada a una computazione quantistica più affidabile. Questa tesi si propone di descrivere i principi di base dell'informatica quantistica e della QEC, con particolare attenzione ai codici di correzione degli errori di *bit* e *phase flip* a tre qubit e alla loro combinazione nel codice Shor. In seguito, il codice di correzione degli errori a tre qubit *bit flip* viene testato, prima su un simulatore per presentare lo scenario ideale, i.e. quello in cui non si verifica alcun errore, poi su un dispositivo quantistico reale, IBM Nairobi, per verificare se l'implementazione di un codice di correzione degli errori a tre qubit *bit flip* su uno stato a singolo qubit aumenta l'accuratezza della trasmissione di un singolo qubit attraverso un canale non unitaio. I risultati mostrano una risposta positiva a questa domanda, dimostrando che l'implementazione del codice QEC abbassa il tasso di errore *bit flip* dallo 0,98% allo 0,12%.

Abstract

Quantum computing is a new and promising field of study but the inherent fragility of qubits poses great difficulties to its implementations. Quantum Error Correction (QEC) offers protocols to mitigate these effects and pave the way for a more reliable quantum computation. This thesis aims to describe the basic principles of quantum computing and QEC, with a focus on three-qubit bit and phase flip error correction codes and their combination in the Shor code. Afterwards, the three-qubit bit flip error correction code is tested, first on a simulator to present the ideal scenario, i.e. where no error occurs, then on a real quantum device, IBM Nairobi, to verify if the implementation of a three-qubit bit flip error correction code on a single-qubit state raises the accuracy of transmitting a single qubit through a noisy channel. The results show a positive answer to this question, exhibiting that the implementation of the QEC code lowers the bit flip error rate from 0.98% to 0.12%.

Contents

List of Figures

Introduction

Noise represents a significant challenge in information processing systems and any technological advance is inextricably linked with the advance in error protection. And a new and promising field as quantum computing is no exception. It is of fundamental importance the implementation of more efficient protocols to shield quantum circuits from noise. This is what quantum error correction (QEC) promises to achieve.

In the field of quantum computing, the qubit stands as the fundamental building block that promises to revolutionize the way we process information and solve complex problems. At the core of this new paradigm lies the concept of quantum bits or "qubits," which defy classical notions of computation and promise to unlock unprecedented computational power.
While the potential of qubits is promising, it comes with a significant challenge: the inherent fragility of quantum states. Quantum systems are highly susceptible to environmental noise and errors, leading to the degradation of quantum information. The delicate quantum coherence that underpins the power of qubits is easily disrupted, threatening the reliability of quantum computations.
This vulnerability underscores the critical need for quantum error correction – a field of study that seeks to develop techniques and algorithms to protect quantum information from errors. Without effective QEC, the great promises of quantum computing could remain out of reach, as errors accumulate and erode the integrity of quantum computations.

The objective of this thesis is to investigate, analyze, and experimentally evaluate the efficacy of quantum error correction techniques, particularly in addressing bit flip errors. This examination is conducted utilizing the advanced tools provided by IBM Quantum.

The initial chapter lays the foundational knowledge by presenting the fundamental principles of quantum mechanics, introducing the concept of qubits, and delving into the vector state formalism and density operator formalism. This chapter provides the essential context for the following study of quantum error correction.
Chapter Two describes the core principles of quantum error correction. It introduces common quantum gates, such as the Hadamard gate, the Controlled-NOT (CNOT) gate, and the Pauli basis. The attention is then shifted to the most common types of errors that can affect a quantum circuit, limiting to single-qubit flip errors, bit flip, and phase flip errors. Lastly, this chapter concludes with a section on the Shor code, a 9-qubit error correction code that combines the two precedently mentioned errors.
Chapter Three concludes the thesis with the empirical validation of the three-qubit bit flip error, where IBM Quantum resources are employed to assess the impact of QEC on quantum computing accuracy, both utilizing a simulator and a real quantum device. It is firstly portrayed the ideal scenario by studying a superposition state through the

simulator and then the potential of QEC is tested on the real quantum device IBM Nairobi.

Chapter 1

Fundamentals of quantum computation

Quantum computation stands as a paradigm that promises to redefine the boundaries of computational power and information processing. This first chapter lays down the essential building blocks that create the foundation of this field. This chapter begins by introducing the qubit, the quantum analogue of the classical bit. Subsequently, a summary of the required quantum formalism, a mathematical framework that describes quantum systems and their evolution, is provided. With these preliminary sections completed, a descriptive analysis of some of the fundamental quantum circuits, the quantum counterpart of classical electronic circuits, which enable the manipulation of quantum information, is then carried out.

1.1 The qubit

The basic unit of classical computation is the *bit*. The basic unit of quantum computation is the *quantum bit* or *qubit*.
In terms of mathematical formalism, the *qubit* is an object belonging to a two-dimensional Hilbert space and its state is a two-dimensional vector in that complex space. However, the best way to understand the nature of this quantum object is by comparing it to the classical bit.
The bit has a *state* - either 0 or 1. In the same way, two possible states for the qubit are $|0\rangle$ or $|1\rangle$, which are called *computational basis states* and form an orthonormal basis in the Hilbert space. An intuitive way to represent these two states is by writing them as column vectors:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \tag{1.1}$$

Where *qubits* and *bits* differ is that the former can take other states apart from $|0\rangle$ and $|1\rangle$. A state where the *qubit* is both $|0\rangle$ and $|1\rangle$ at the same time is called *superposition*.

Definition 1.1.1. Superposition: Any linear combination $\sum_i \alpha_i|\psi_i\rangle$, where $\alpha_i \in \mathbb{C}$ is known as *amplitude*, is called a superposition of the states $|\psi_i\rangle$,

Some common examples of superposition are the states $|+\rangle$, $|-\rangle$, $|+\rangle_y$ and $|-\rangle_y$. These are defined as:

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}}, \quad |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \tag{1.2}$$

$$|+\rangle_y = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad |-\rangle_y = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \tag{1.3}$$

At a more general level, a qubit is defined as the superposition of both states $|0\rangle$ and $|1\rangle$:

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1.4}$$

where $|\Psi\rangle$ is the state of the qubit and α and β are the complex coefficients of respectively states $|0\rangle$ and $|1\rangle$.
These two complex numbers are fundamental because their magnitude's square expresses the probability of the associated state to occur after a measurement:

$$p_0 = |\alpha|^2, \quad p_1 = (1 - p_0) = |\beta|^2 \tag{1.5}$$

where p_0 and p_1 are, respectively, the probabilities for the state $|0\rangle$ and $|1\rangle$.
A direct consequence of (1.5) is that α and β have to satisfy this relation:

$$|\alpha|^2 + |\beta|^2 = 1. \tag{1.6}$$

This result comes also from the fact that as in Postulate 1.2.1, $|\Psi\rangle$ has to be a unit vector on the Hilbert space, which translates to the condition:

$$\langle\Psi|\Psi\rangle = 1 \tag{1.7}$$

which then directly brings to (1.6). The condition (1.7) is usually known as the *normalization condition* for the state vector [1].
The state of a qubit can also be visualized geometrically by using the unit three-dimensional sphere, called the *Bloch sphere* (Figure 1.1).
A state is identified on the sphere by two complex numbers, θ and φ, through the expression:

$$|\Psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle \tag{1.8}$$

This representation is widely used in quantum computation since the majority of operations on a single qubit can be visualized as rotations on the Bloch sphere. This could lead one to the conclusion that infinite information can be stored in a qubit since there

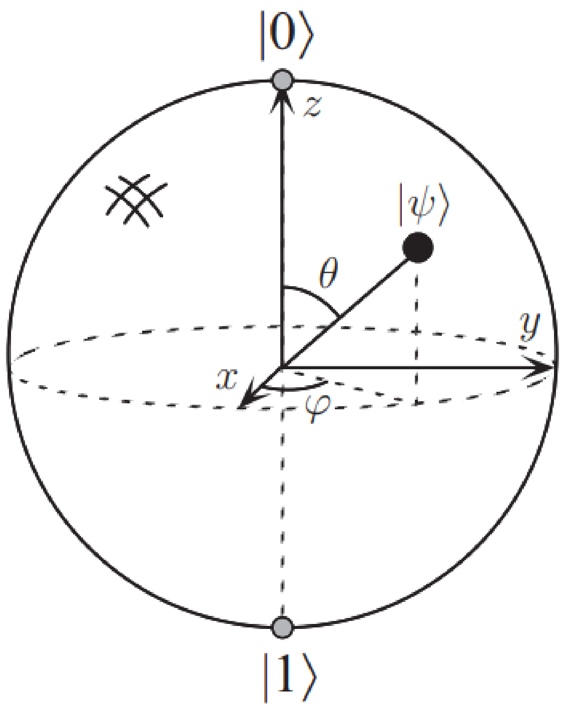


Figure 1.1: The Bloch sphere[2]

are infinite points on the surface of a sphere. However, this is not a real possibility since the qubit collapses when measured, turning the superposition of $|0\rangle$ and $|1\rangle$ to the actual measurement result. The reason for this collapse is unknown; it's one of the fundamental postulates of quantum mechanics [2].

1.2 Quantum formalism

In the pursuit of understanding and mitigating errors in quantum computation, a fundamental grasp of quantum formalism is indispensable. By exploring quantum states, operators, and the unique principles of quantum mechanics, we lay the groundwork for comprehending the challenges of quantum error correction and the strategies aimed at preserving quantum information integrity. Naturally, the content provided in this paper will be constrained to the bare minimum required for discussing the basics of QEC.

1.2.1 Vector state formalism

First and foremost, it is compulsory to set the stage for the study of quantum mechanics:

Postulate 1.2.1. *Associated with any isolated physical system is a Hilbert space (a complex vector space with an inner product, in $\mathbb{C}$) referred to as the state space (of the system). The system is fully described by its state vector, which is a unit vector in the system's state space* [2].

Quantum mechanics does not provide information about the state space or the state vector of a given physical system. Determining these aspects for a particular system presents a challenging problem for which many rules have been developed.

Definition 1.2.1. Linear operator: Given two vector space U and V over a field K, a transformation $A : U \to V$ is linear if and only if: [1]

$$A(\alpha x + \beta y) = \alpha A x + \beta A y. \tag{1.9}$$

Definition 1.2.2. Hermitian conjugate: Each linear operator U on an inner product space defines a Hermitian conjugate (or adjoint) $U^\dagger$ on that space such that:

$$\langle Ux, y\rangle = \langle x, U^\dagger y\rangle \tag{1.10}$$

where $\langle\cdot,\cdot\rangle$ is the inner product on the vector space [1].

Definition 1.2.3. Hermitian operator: Given a linear operator H, it is said to be Hermitian (or self-adjoint) if and only if it satisfies the following condition:

$$H^\dagger = H \tag{1.11}$$

In other words, an operator is Hermitian if and only if it is equal to its Hermitian conjugate[1].

Definition 1.2.4. Unitary operator: An operator U is said to be *unitary* if and only if satisfies the following condition[1]:

$$U^\dagger U = U U^\dagger = \mathbb{I} \tag{1.12}$$

where $U^\dagger$ is the Hermitian conjugate of operator U, a in Definition 1.2.2.

Postulate 1.2.2. *The evolution of a closed quantum system is described by an unitary transformation. Given a state $|\Psi\rangle$ at time t and a state $|\Psi'\rangle$ at time t', that is expressed as:*

$$|\Psi'\rangle = U|\Psi\rangle \tag{1.13}$$

where U is an unitary operator and $U = U(t, t')$[2].

Postulate 1.2.3. ***Schrödinger equation**: The evolution in time of a closed quantum system is described by the Schrödinger equation:*

$$i\hbar \frac{d|\psi\rangle}{dt} = H|\psi\rangle \tag{1.14}$$

where $\hbar$ is the Planck's constant and H is a a Hermitian operator referred to as the Hamiltonian of the system. The time evolution is represented by a unitary operator U, of which H is the generator. The time evolution law is then:

$$i\hbar\frac{dU}{dt} = HU. \tag{1.15}$$

In the case of a time-independent Hamiltonian the time evolution operator U between the times t_1 and t_2 can be written as[2]:

$$U(t_2, t_1) = e^{\frac{i(t_2-t_1)H}{\hbar}} \tag{1.16}$$

In a similar manner to how quantum mechanics doesn't reveal the state space or quantum state of a specific quantum system, it also does not specify which unitary operators U characterize the quantum dynamics observed in the real world.

Postulate 1.2.4. ***Quantum measurement:*** *In quantum mechanics, measurement is described by a collection $\{M_m\}$ of measurement operators, which act on the state space. The index m refers to the possible measurement's outcomes: for example, if a system is in state $|\psi\rangle$ just before being measured then the probability of outcome m is:*

$$P(m) = \langle\psi|M_m^\dagger M_m|\psi\rangle. \tag{1.17}$$

The state of the system after the measurement is given by[2]:

$$\frac{M_m|\psi\rangle}{\sqrt{\langle\psi|M_m^\dagger M_m|\psi\rangle}}. \tag{1.18}$$

The measurement operators previously presented have to satisfy the *completeness equation*[2]:

$$\sum_m M_m^\dagger M_m = \mathbb{I}. \tag{1.19}$$

This relation also verifies that the probabilities sum to one:

$$1 = \sum_m P(m) = \sum_m \langle\psi|M_m^\dagger M_m|\psi\rangle. \tag{1.20}$$

Postulate 1.2.5. *The state space of a composite physical system results from the tensor product of the state spaces of the individual component physical systems. Moreover, when dealing with systems labeled from 1 to n, if the system number i is initially set in the state $|\psi_i\rangle$, then the combined state of the entire system is: $|\psi_1\rangle \otimes |\psi_2\rangle \otimes ...|\psi_n\rangle$ [2].*

1.2.2 Density operator formalism

Given a quantum system in one of a number of states $|\psi_i\rangle$ with corresponding probabilities p_i, then $\{p_i, |\psi_i\rangle\}$ is an *ensemble of pure states.* Naturally, $\sum_i p_i = 1$. This concept is preliminary to the definition of the density operator system, which is now shown:

Definition 1.2.5. Density operator: Given a quantum system as characterized above, the corresponding density operator (or density matrix) is defined as[2]:

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|. \tag{1.21}$$

This formalism is equivalent to the vector state one since it is possible to demonstrate that it can be used to formulate every postulate of quantum mechanics.
The density operator language can be utilized to describe the evolution of a quantum system:

Postulate 1.2.6. ***Evolution of a system:*** *Given s closed quantum system whose evolution is described by a unitary operator U and its initial state is $|\psi_i\rangle$ with probability p_i. After the evolution has occurred, the system will be in state $U|\psi_i\rangle$ with probability p_i[2]:*

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \xrightarrow{U} \sum_i p_i U|\psi_i\rangle\langle\psi_i|U^\dagger = U\rho U^\dagger \tag{1.22}$$

Definition 1.2.6. Pure and mixed states: A quantum system is said to be in a *pure state* if its state $|\psi\rangle$ is known exactly. Its density operator is expressed as $\rho = |\psi\rangle\langle\psi|$. Alternatively, ρ is said to be in a *mixed state*, which is a mixture of the different pure states in the ensemble for ρ[2].

An immediate way to discern mixed and pure states is by using their trace: in fact, pure states satisfy the condition:

$$Tr(\rho^2) = 1 \tag{1.23}$$

while mixed states:

$$Tr(\rho^2) < 1. \tag{1.24}$$

Measurements too can be expressed through this formalism:

Postulate 1.2.7. ***Measurement:*** *Suppose a measurement described by the measurement operator M_m. If the initial state was $|\psi_i|$, subsequently the probability of a result m is[2]:*

$$p(m|i) = \langle\psi_i|M_m^\dagger M_m|\psi_i\rangle = Tr(M_m^\dagger M_m|\psi_i\rangle\langle\psi_i|), \tag{1.25}$$

where the following result has been used:

$$Tr(A|\psi\rangle\langle\psi|) = \sum_i \langle i|A|\psi\rangle\langle\psi|i\rangle \tag{1.26}$$

where A is an arbitrary matrix.
By the law of total probability:

$$p(m) = \sum_i p(m|i)p_i = \sum_i p_i Tr(M_m^\dagger M_m |\psi_i\rangle\langle\psi_i|) = Tr(m_m^\dagger M_m \rho). \tag{1.27}$$

The state after obtaining the measurement result m is:

$$|\psi_i^m\rangle = \frac{M_m|\psi_i\rangle}{\sqrt{\langle\psi_i|M_m^\dagger M_m|\psi_i\rangle}}. \tag{1.28}$$

Therefore, after a measurement of result m the system can be described as an ensemble of states $|\psi_i^m\rangle$ with respective probabilities $p(i|m)$. The density operator associated with this system is therefore:

$$\rho_m = \sum_i p(i|m)|\psi_i^m\rangle\langle\psi_i^m| = \sum_i p(i|m)\frac{M_m|\psi_i\rangle\langle\psi_i|M_m^\dagger}{\langle\psi_i|M_m^\dagger M_m|\psi_i\rangle}. \tag{1.29}$$

Lastly, by applying (1.25) and (1.27), the same result obtained through the vector state formalism is reached:

$$\rho_m = \sum_i p_i \frac{M_m|\psi_i\rangle\langle|M_m^\dagger}{Tr(M_m^\dagger M_m\rho)} = \frac{M_m\rho M_m^\dagger}{Tr(M_m^\dagger M_m\rho)}. \tag{1.30}$$

Theorem 1.2.1. ***Characterization of density operators****: An operator ρ is the density operator associated to an ensemble $\{p_i, |\psi_i\rangle\}$ if and only if the following conditions are verified[2]:*

1) Trace: $Tr(\rho) = 1$.

2) Positivity: ρ is a positive operator.

3) ρ is limited.

4) self-adjoint: $\rho = \rho^\dagger$.

This theorem offers an intrinsic characterization of density operators, which are defined as positive, self-adjoint operators ρ with a trace equal to one. By adopting this definition, we can rephrase the postulates of quantum mechanics within the framework of density operators. Here are the reformulated postulates[2]:

Postulate 1.2.8. *Connected to any isolated physical system exists a complex vector space equipped with an inner product, forming a Hilbert space recognized as the system's state space. The complete description of the system relies on its density operator, denoted as a self-adjoint and positive operator ρ with a unit trace, operating within the state space of the system. When a quantum system assumes the state ρ_i with a probability p_i, the density operator for the system is defined as $\sum_i p_i\rho_i$.*

This is the reformulated version of Postulate 1.2.1.

Postulate 1.2.9. *The evolution of a closed quantum system can be characterized by a unitary transformation. In other words, the state ρ of the system at time t_1 is connected to the state ρ', which exists at time t_2, through the action of a unitary operator U. Importantly, this unitary operator depends solely on the times t_1 and t_2.*

$$\rho' = U\rho U^\dagger \tag{1.31}$$

This is the reformulated version of Postulate 1.2.2.

Postulate 1.2.10. *Quantum measurements are characterized by a set $\{M_m\}$ of measurement operators, which are operators that operate on the state space of the system under measurement. The index m signifies the potential measurement outcomes in the experiment. When the quantum system is in the state ρ just before the measurement, the probability of obtaining result m is determined by:*

$$p(m) = Tr(M_m^\dagger M_m \rho), \tag{1.32}$$

and after the measurement has been executed the state of the quantum system is:

$$\frac{M_m \rho M_m^\dagger}{Tr(M_m^\dagger M_m \rho}. \tag{1.33}$$

Moreover, the measurement operators verify the completeness condition:

$$\sum_i M_m^\dagger M_m = \mathbb{I}. \tag{1.34}$$

This is the reformulated version of Postulate 1.2.4.

Postulate 1.2.11. *The state space of a composite physical system is the tensor product of the state spaces of the component physical systems. Furthermore, in the scenario where there are n systems denoted as 1 through n, and if the system indexed as i is initially prepared in the state ρ_i, then the collective state of the entire system is given by $\rho_1 \otimes \rho_2 \otimes ...\rho_n$.*

This is the reformulated version of Postulate 1.2.5.

1.3 Quantum circuits

Classical computers use electronic circuits to transmit bits of information, while quantum computers use quantum circuits to transmit information through qubits. While electronic

circuits are formed by wires and logic gates, quantum circuits are formed by wires and elementary quantum gates [2].
To introduce this last concept we first have to present some preliminary results, starting with the Pauli basis for the two-dimensional Hilbert space:

$$\mathbb{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{1.35}$$

These matrices are the most used operators in quantum computation since the first one is the identity $\mathbb{I}$ operator and the remaining three are π rotations on one of the three axes of the Bloch sphere. Moreover, the eigenkets of these three matrices are the three bases previously defined: the *X-basis* $\{|+\rangle, |-\rangle\}$, the *Y-basis* $\{|+\rangle_y, |-\rangle_y\}$ and the *Z-basis* $\{|0\rangle, |1\rangle\}$.

1.3.1 Single-qubit gates

Considering single-qubit logical gates for classical computation, the only (non-trivial) one is the NOT gate, which turns 0 into 1 and vice versa [2].
Is it possible to define the same operation (and so the same gate) for quantum computing? The answer is yes, but this gate is obviously more elaborated than the classical one since the qubit can be in states other than $|0\rangle$ and $|1\rangle$. So the question that arises from this is: what is the effect of this gate on a superposition state? The quantum NOT gate acts *linearly* onto the state

$$|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1.36}$$

taking it to the state where the roles of the computational basis states are interchanged [2]:

$$X|\Psi\rangle = \beta|0\rangle + \alpha|1\rangle. \tag{1.37}$$

It's useful to highlight the gate's matrix form, which coincides with the Pauli's X basis:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}. \tag{1.38}$$

It's easy to make sense of this result: remembering (1.1), we can see that the vector representation for the state $|\Psi\rangle$ is:

$$|\Psi\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \tag{1.39}$$

and if we apply X to this vector we get the final result as defined in (1.37):

$$X \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \beta \\ \alpha \end{bmatrix}. \tag{1.40}$$

This is only one example but it turns out that all the single-qubit quantum gates are, in fact, 2x2 matrices. However, the condition (1.6), which has to be satisfied for any state in the form $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$ and so also by the state $X|\Psi\rangle = \beta|0\rangle + \alpha|1\rangle$, dictates that all the single-qubit gates U have to be *unitary*, as in Definition 1.2.4.
This result expresses in terms of vector formalism on a two by two Hilbert space what was already pointed out in Postulate 1.2.2.

Hadamard gate

One of the most commonly used gates in quantum computing is the Hadamard gate or H gate. It is a single-qubit gate that consists in a π rotation around the X+Z axis and its effect is to change computational basis from $\{|0\rangle, |1\rangle\}$ to $\{|+\rangle, |-\rangle\}$ and vice versa.[3] It is defined as:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1.41}$$

where the matrix in written in the computational basis $\{|0\rangle, |1\rangle\}$. This gate is particularly frequent in quantum circuits since it provides the opportunity to create a superposition state, such as $|+\rangle$ or $|-\rangle$.

From the combination of the H gate and Pauli's gates, we obtain three useful identities, whose proof comes from direct computation:

$$HXH = Z \quad HYH = -Y \quad HZH = X. \tag{1.42}$$

Phase shift gate

The phase shift is a family of single-qubit gates that map the basis states $|0\rangle \rightarrow |0\rangle$ and $|1\rangle \rightarrow e^{i\varphi}|1\rangle$, where $\varphi \in [0, 2\pi]$. The only value changed by this gate is the phase of the quantum state, while the probability of measuring $|0\rangle$ or $|1\rangle$ stays the same. This equates to a rotation of φ radians around the Z-axis of the Bloch sphere.
This gate is formally defined as follows:

$$P(\varphi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\varphi} \end{bmatrix}. \tag{1.43}$$

Some notorious examples of this gate are the T gate ($\varphi = \frac{\pi}{4}$), the S gate, or phase gate, ($\varphi = \frac{\pi}{2}$) and the Pauli-Z gate ($\varphi = \pi$):

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = P(\pi) \tag{1.44}$$

$$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix} = P(\frac{\pi}{2}) = \sqrt{Z} \tag{1.45}$$

$$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{\pi}{4}} \end{bmatrix} = P(\frac{\pi}{4}) = \sqrt{S} = \sqrt[4]{Z}. \tag{1.46}$$

It's worth pointing out that the phase shift gate $P(\varphi)$ is not Hermitian, as in Definition 1.2.3, except for $\varphi = n\pi$ with $n \in \mathbb{Z}$. However, they do have the following property [3]:

$$P^{\dagger}(\varphi) = P(-\varphi) \tag{1.47}$$

1.3.2 Multiple qubits gates

More complex circuits include gates that act on multiple qubits at once. For example, the circuit that implements the operation $U = X_1 \otimes X_2$ on the state $|00\rangle$ is visualized as [4]:

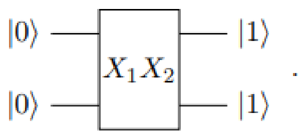


Figure 1.2: Schematic of the operation $U = X_1 \otimes X_2$ on the state $|00\rangle$. In the picture, the tensor product symbol ($\otimes$) has been omitted for clarity [4].

CNOT gate

The Controlled-NOT gate, or CNOT gate, is one of the foundation blocks for creating quantum circuits. As hinted by its name, this gate performs a controlled-NOT operation, meaning that a NOT gate (as defined in Section 1.3.1) is executed on a qubit, the target, and that this operation is controlled by the state of a second qubit, the control. In particular, the NOT is executed on the target if and only if the control is in state $|1\rangle$. Its usual circuital representation is:
As can be seen in Figure 1.3, two scenarios can unfold [5]:

A: $|x\rangle = |1\rangle \rightarrow X|y\rangle$

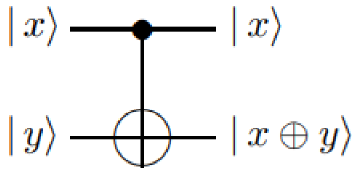


Figure 1.3: CNOT gate schematics where the control qubit is in state $|x\rangle$ and the target in state $|y\rangle$ ($\oplus$ is the symbol for the XOR, which is CNOT's logical analog).

B: $|x\rangle \neq |1\rangle \rightarrow |y\rangle = |y\rangle$

It is defined as:

$$CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \tag{1.48}$$

This gate holds extreme relevance in quantum computing since, when it's combined with the Hadamard gate, it permits the creation of *entanglement* between the control and the target [6].

Definition 1.3.1. Entenglement: An *entangled state* of a composite system is a state that cannot be written as a product state of the composite system [7].

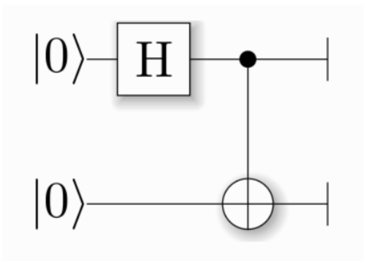


Figure 1.4: Simplest circuit to form an entangled state with two qubits. It includes a Hadamard gate and a CNOT gate [8].

The entangled state created is the following:

$$|\psi\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}} \tag{1.49}$$

SWAP gate

The quantum SWAP gate, a fundamental operation in quantum computing, plays a pivotal role in reshuffling the quantum states of two qubits. As its name suggests, it enables the exchange of information between qubits, effectively swapping their quantum states. This gate holds significance in various quantum algorithms and quantum circuit designs, serving as a cornerstone for tasks like sorting and data rearrangement.
In the computational basis, it is formally defined as:

$$SWAP = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.50}$$

but it can also be decomposed through the Pauli's basis:

$$SWAP = \frac{\mathbb{I} \otimes \mathbb{I} + X \otimes X + Y \otimes Y + Z \otimes Z}{2}. \tag{1.51}$$

Toffoli gate

Lastly, the Toffoli gate, or CCNOT gate: It's a 3-bit gate which is universal[1] in classical computation but not in quantum computing, although the quantum version of this gate is analogue to the classical one [9]. In this gate, if the two qubits are in state $|1\rangle$ a Pauli-X gate is applied on the third qubit, if not, nothing happens. It is defined as:

$$CCNOT = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{1.52}$$

and it can be reduced using the Pauli basis:

$$CCNOT = e^{i\frac{\pi}{8}(\mathbb{I}-Z_1)\otimes(\mathbb{I}-Z_2)\otimes(\mathbb{I}-X_3)}. \tag{1.53}$$

[1]When a gate is said to be *universal*, it means that it can be used to construct any arbitrary computation or perform any desired logical operation. In other words, a universal gate can serve as a building block for creating complex computations or algorithms.

Chapter 2

Quantum error correction

We commence our exploration by constructing the foundational framework of quantum error-correcting codes, designed to safeguard quantum information from the disruptive influence of noise. These codes employ a distinctive encoding method, rendering quantum states robust in the face of noise-induced disturbances, and facilitating decoding when the desire arises to restore the original state.

The fundamental concept underpinning noise protection techniques revolves around the principle that when protecting a message from the influence of noise, it is prudent to augment the message with redundant information. This redundancy serves as a protective measure. Consequently, even if certain portions of the encoded message are subject to corruption due to noise, the presence of redundancy ensures the possibility of message recovery or decoding. This restoration process results in the retrieval of all the original information contained within the message.

When developing quantum error correction codes, we have to face three quite formidable difficulties that arise from the very nature of the quantum world [2]:

- *No-cloning*: The classical approach of protecting a bit of information by duplicating it various times is no longer a possibility in the quantum realm since it is forbidden by the no-cloning theorem, which is stated below [10].
- *Errors are continuous*: Quantum circuits suffer from inherently continuous errors caused by noise. This raises a fundamental issue since correcting those continuous errors would require infinite precision [11].
- *Measurement destroys quantum information*: We have to observe the output of a channel to be able to decide which recovery method to use; although, since observing a quantum system, such as the qubit, collapses it, this makes recovery impossible.

Theorem 2.0.1. ***No-cloning***: *Given a system, it is impossible to construct a unitary operator U_{clone} which performs the following operation:*

$$U_{clone}(|\psi\rangle \otimes |0\rangle) \rightarrow |\psi\rangle \otimes |\psi\rangle, \tag{2.1}$$

where $|\psi\rangle$ is the state to be cloned.

This result is in clear contrast with classical codes, where one of the fundamental assumptions is that data can be duplicated at will [4].

2.1 Errors in quantum circuits

In the field of quantum computing, quantum circuits serve as the backbone for performing complex operations on quantum information, holding the promise of revolutionizing computational capabilities. However, this potential is accompanied by a significant challenge: susceptibility to errors. Quantum circuits are highly sensitive, and deviations from ideal conditions can lead to erroneous outcomes. Errors in quantum circuits can originate from various sources, such as environmental influences, imperfect gate operations, and the inherent behavior of quantum systems.
In this section, only bit flip and phase flip errors are going to be presented since they are the most common one-qubit errors. However, various other errors can affect the output of a quantum circuit, but for conciseness, they are excluded from the content of this thesis.

2.1.1 Bit flip error

The concept of bit-flip error is present also in classical computation, where it consists of an error causing the value of a bit to switch from 0 to 1 or vice versa. It is the same operation as applying a logical NOT gate.
In quantum computing, a bit flip error, often denoted as an X-error, is a type of quantum error that occurs when the state of a qubit is unintentionally altered by the application of a Pauli-X gate (also known as a bit flip gate). This gate, when applied in error, flips the state of the qubit from $|0\rangle$ to $|1\rangle$ or vice versa, thereby introducing a change in the qubit's quantum information.
The formal definition of this error is the following:

Definition 2.1.1. Bit flip error: Given a qubit in state $|\Psi\rangle = \alpha|0\rangle + |1\rangle$, a bit flip error consists of an (erroneous) application of a Pauli-X gate to the qubit, resulting in state $|\Psi'\rangle$:

$$|\Psi'\rangle = X|\Psi\rangle = \beta|0\rangle + \alpha|1\rangle \tag{2.2}$$

which can be visualized geometrically as a π rotation around the x-axis of the Bloch sphere, up to a global phase.

2.1.2 Phase flip error

In quantum computing, a phase flip error, often denoted as a Z-error, is a type of quantum error that occurs when the phase (or relative sign) of a qubit's quantum state is unintentionally inverted. This inversion typically arises due to the application of a Pauli-Z gate (also known as a phase flip gate) in error. The Pauli-Z gate changes the sign of the $|1\rangle$ state while leaving the $|0\rangle$ state unaffected.
The formal definition is the following:

Definition 2.1.2. Phase flip error: Given a qubit in state $|\Psi\rangle = \alpha|0\rangle + \beta|1\rangle$, a phase flip error consists of an (erroneous) application of a Pauli-Z gate to the qubit, resulting in the state $|\Psi'\rangle$:

$$|\Psi'\rangle = Z|\Psi\rangle = \alpha|0\rangle - \beta|1\rangle \tag{2.3}$$

which can be visualized geometrically as a π rotation around the z-axis of the Bloch sphere.

2.2 Repetition codes and stabilizer measurement

The fundamental idea behind QEC codes is that, if we want to safeguard a message from noise, then we should *encode* it, by adding some redundant information to the message. This redundancy serves as a protective measure, allowing for potential recovery or *decoding* of the message even in cases where certain information within the encoded message has been corrupted by noise. Ultimately, the goal is to ensure the retrieval of all the original information present in the initial message.[2] So, how is redundancy added to a quantum system to allow errors to be detected in real time? Classical repetition codes work by increasing the resources used to encode the data beyond the theoretical minimum. In the same way, in quantum codes, redundancy is added by expanding the Hilbert space in which the qubits are encoded. [12]

As an example of this process, a prototypical two-qubit code is presented, designed to (only) detect a single-bit flip error on the state $|\psi\rangle$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \xrightarrow{encoder} |\psi\rangle_L = \alpha|00\rangle + \beta|11\rangle = \alpha|0\rangle_L + \beta|1\rangle_L, \tag{2.4}$$

where $|0\rangle_L = |00\rangle$ and $|1\rangle_L = |11\rangle$ are called *logical qubits.*
It is important to notice that this process does not correspond to cloning the state as:

$$|\psi\rangle_L = \alpha|00\rangle + \beta|11\rangle \neq |\psi\rangle \otimes |\psi\rangle. \tag{2.5}$$

What has been achieved through this operation of encoding is to distribute the information from the original state $|\psi\rangle$ to the two-qubit entangled state $|\psi\rangle_L$.

Let's consider the state space *before* and *after* the encoding process: Prior to it, the single qubit is parametrized into a two-dimensional Hilbert space :

$$|\psi\rangle \in \mathcal{H}_2 = span\{|0\rangle, |1\rangle\}. \tag{2.6}$$

while, after the encoding operation, the logical qubit is located in a four-dimensional Hilbert space $|\psi\rangle_L \in \mathcal{H}_4$:

$$|\psi\rangle_L \in \mathcal{H}_4 = span\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}. \tag{2.7}$$

In particular, the logical qubit (just after the encoding process and before any error happens) is in a subspace of $\mathcal{H}_4$ known as the *codespace*:

$$|\psi\rangle_L \in \mathcal{C} = span\{|00\rangle, |11\rangle\} \subset \mathcal{H}_4. \tag{2.8}$$

Now, the hypothesized scenario is the one where a bit-flip error occurred on the first qubit:

$$X_1|\psi\rangle_L = \alpha|10\rangle + \beta|01\rangle \tag{2.9}$$

where X_1 is the Pauli X-gate acting on the first qubit. The resulting state belongs to a particular subspace of the state space:

$$X_1|\psi\rangle_L \in \mathcal{F} \subset \mathcal{H}_4, \tag{2.10}$$

where $\mathcal{F}$ is the error subspace and it's the same subspace where the logical state would have been rotated to in the case of an X_2 error. If no error occurred, then the logical state still belongs in the $\mathcal{C}$ codespace.
Now, since the $\mathcal{F}$ and $\mathcal{C}$ subspaces are mutually orthogonal, it's possible to identify the subspace where the logical qubit resides through a projective measurement without compromising the integrity of the encoded quantum information. In quantum coding scenarios, these kinds of measurements are referred to as stabilizer measurements. [4]

Now a method to differentiate between the two subspaces $\mathcal{F}$ and $\mathcal{C}$ is necessary: it is the projective measurement Z_1Z_2 operator, which yields (+1) eigenvalue when applied to $|\psi\rangle_L$:

$$Z_1Z_2|\psi\rangle_L = Z_1Z_2(\alpha|00\rangle + \beta|11\rangle) = (+1)|\psi\rangle_L. \tag{2.11}$$

Since this operator leaves the logical state unchanged, it is said to *stabilize* it and it is the building block for stabilizer codes. On the contrary, this operator projects both error spaces $X_1|\psi\rangle_L$ and $X_2|\psi\rangle_L$ onto the (-1) eigenvalue space. It is of important notice that for both cases, the Z_1Z_2 operator does not disturb the information encoded in the two coefficients α and β. [13]

2.3 3-qubit codes

However, using 3-qubit codes rather than two-qubit codes offers several advantages:

- **Increased Redundancy**: 3-qubit codes provide more redundancy than 2-qubit codes. Redundancy is essential in error correction because it allows for the detection and correction of a greater number of errors. With more redundancy, it becomes possible to correct more errors per encoded qubit.
- **Higher Error Tolerance**: The additional qubit in 3-qubit codes enhances their error tolerance. They can detect and correct a wider range of errors, including those that would be undetectable or uncorrectable with 2-qubit codes. This is especially important in practical quantum computing where noise and errors are prevalent.
- **Reduced Logical Error Rate**: In many cases, 3-qubit codes have a lower logical error rate compared to 2-qubit codes. This means that the error-corrected qubit is more reliable when using 3-qubit codes, leading to better overall performance of quantum circuits.
- **Simplified Error Detection**: 3-qubit codes often simplify error detection and correction procedures. They allow for the detection of specific types of errors (such as bit-flip and phase-flip errors) through measurements on the ancilla qubit, making it easier to identify and correct errors.

Above all this, a 2-qubit code is not sufficient since it only *detects* the error and it does not provide enough information to allow one to infer which qubit the error occurred on, while both detection and correction are possible with a 3-qubit code.
For all these reasons, 3-qubit codes are the standard and they are what is going to be analyzed in the following sections.

A simple example is now presented for a better understanding of the topic.
Suppose we want to send a message through a *binary symmetric channel*, which is a channel where noise acting on it has only two possible effects. For example, if the effect of the noise on such a channel is to flip the bit being transmitted, then there will be a probability $p > 0$ that the bit is flipped and a probability $1-p$ that the bit is transmitted through the channel without error. [2] In this type of channel, a simple way to safeguard the information from this type of error is to insert some redundancy in the code, i.e. replacing the single bit with three copies of itself.

$$0 \to 000 \quad 1 \to 111 \tag{2.12}$$

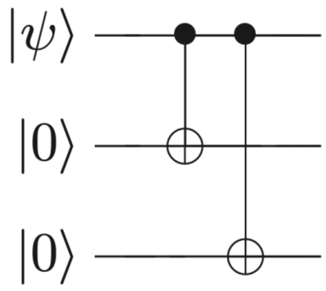


Figure 2.1: In this circuit the original qubit is encoded by entangling it with two qubits through two CNOT gates

where the bits strings 000 and 111 are usually respectively referred to as *logical* 0 ($|0\rangle_L$) and *logical* 1 ($|1\rangle_L$). Now the final logical state is in the form:

$$|\psi\rangle_L = \alpha|000\rangle + \beta|111\rangle. \tag{2.13}$$

On the other end of the channel a certain output, for example 001, is collected by the receiver. With the assumption that the probability p is not too high, it is reasonable to think that only the third bit has been flipped and that 0 was the original bit that was sent.
This decoding process is referred to as *majority voting* since the original input is what appears the most times in the output. This fails if more than one bit is flipped. However, the probability that two or more bits are flipped is given by:

$$p_e = 3p^2(1-p) + p^3 = 3p^2 - 2p^3 \tag{2.14}$$

and if $p < \frac{1}{2} \to p_e < p$ meaning that, without encoding, the process is more susceptible to bit flip errors.

2.3.1 3-qubit bit flip code

The case of bit flip error is going to be used as a testing ground to introduce concepts fundamental to the process of quantum error detection and correction.

Given the usual general state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, it is first encoded through the logical bits:

$$\alpha|0\rangle \to \alpha|000\rangle = \alpha|0\rangle_L \tag{2.15}$$

$$\beta|1\rangle \to \beta|111\rangle = \beta|1\rangle_L. \tag{2.16}$$

A circuit that executes this encoding process is shown in Figure 2.1. Subsequently, every qubit undergoes an autonomous pass through a dedicated instance of the bit flip channel.

The fundamental supposition to this model is that the error occurs only on one or fewer qubits, as previously discussed.
Now a simple two-stage procedure for detecting and correcting a bit flip error related to this situation is going to be presented:

(1) **Syndrome measurement**: this step allows us to detect if an error has occurred on one qubit. The result is known as *error syndrome* and for this channel, there are four possible outcomes, corresponding to four projection operators:

$$P_0 \equiv |000\rangle\langle 000| + |111\rangle\langle 111| \to \text{no error} \tag{2.17}$$
$$P_1 \equiv |100\rangle\langle 100| + |011\rangle\langle 011\rangle \to \text{bit flip error on qubit one} \tag{2.18}$$
$$P_2 \equiv |010\rangle\langle 010| + |101\rangle\langle 101| \to \text{bit flip error on qubit two} \tag{2.19}$$
$$P_3 \equiv |001\rangle\langle 001| + |110\rangle\langle 110| \to \text{bit flip error on qubit three} \tag{2.20}$$

Suppose that a bit flip error occurred on the second qubit and the state is then $|\psi_1\rangle = \alpha|010\rangle + \beta|101\rangle$. By direct computation, one can verify that $\langle\psi_1|P_2|\psi_1\rangle \neq 0$ while $\langle\psi_1|P_0|\psi_1\rangle = \langle\psi_1|P_1|\psi_1\rangle = \langle\psi_1|P_3|\psi_1\rangle = 0$. This implies that the error occurred on the second qubit. An important feature of this method is that, as can be easily verified, the state is conserved by the syndrome measurement through the application of the projection operator. This is crucial to work around the issue of the state collapse, as discussed in Section 2.3.

(2) **Recovery**: The value of the error syndrome before obtained is now used to recover the original state; since that value was 2, it indicates that the error occurred on the second qubit and so the correction, which in this case corresponds to the application of a Pauli X-gate, is going to be applied to that qubit. Hence, this simple process has made it possible to recover the initial state with perfect accuracy.

Definition 2.3.1. Fidelity: Given two states ρ_1 and ρ_2, the fidelity F expresses their distinguishability:

$$F(\rho_1, \rho_2) \equiv |\langle\psi_1|\psi_2\rangle|^2 \tag{2.21}$$

where $|\psi_1\rangle$ and $|\psi_2\rangle$ are the corresponding vector states [14].

In particular, the fidelity between a pure ($|\psi\rangle$) and a mixed (ρ) state, as in Definition 1.2.6 is given by:

$$F(|\psi\rangle, \rho) = \sqrt{\langle\psi|\rho|\psi\rangle}. \tag{2.22}$$

As one can easily imagine, the goal of quantum error correction is to enhance the accuracy with which quantum information is stored or transmitted, approaching the highest achievable fidelity, which is equal to one.
Consequently, fidelity is the perfect tool to verify if a QEC implementation of a code, such as the previously discussed three-qubit bit flip code, has actually increased the accuracy of the original code. Hence, the *minimum* fidelity of the three-qubit bit flip code

is going to be compared with the fidelity when no error correction is executed.
Given that the quantum system is in state $|\psi\rangle$, in the case that the error correction is performed, the state of the qubit, after it has been sent through the channel, is:

$$\rho = (1-p)|\psi\rangle\langle\psi| + pX|\psi\rangle\langle\psi|X \tag{2.23}$$

and the fidelity is:

$$F = \sqrt{\langle\psi|\rho|\psi\rangle} = \sqrt{(1-p) + p\langle\psi|X|\psi\rangle\langle\psi|X|\psi\rangle}. \tag{2.24}$$

Since the second term under the square root is non-negative and equal to zero when $|\psi\rangle = |0\rangle$, then the minimum value for the fidelity is $F = \sqrt{1-p}$.
If the encoded state is expressed as $|\psi\rangle = \alpha|0_L\rangle + \beta|1_L\rangle$, then the state after passing through the noisy channel and the error correction implementation is:

$$\rho = [(1-p)^3 + 3p(1-p)^2]|\psi\rangle\langle\psi| + \cdots \tag{2.25}$$

where the terms regarding the bit flips of two or three qubits have been omitted and they are all positive terms. Hence, the fidelity F is:

$$F = \sqrt{\langle\psi|\rho|\psi\rangle} \geq \sqrt{(1-p)^3 + 3p(1-p)^2} = \sqrt{1-3p^2+2p^3} \tag{2.26}$$

and therefore the fidelity is improved, under the condition that $p < \frac{1}{2}$, as already discussed earlier [2].

Now an example of a three-qubit bit flip code is shown in Figure 2.2. The image in Figure 2.2 can also be found in my GitHub repository.

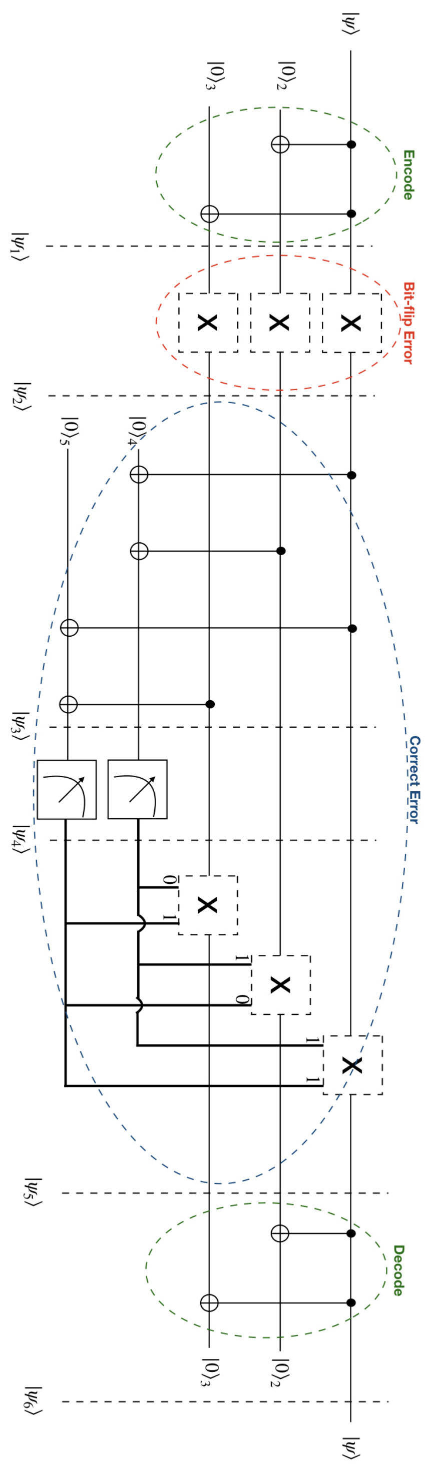


Figure 2.2: Example of a bit flip code on a one qubit system. In the figure, the encoding, bit flip error, error correction, and decoding phases are highlighted [15].

First and foremost, some assumptions are ought to be made:

- If more than one qubit is transferred through the noisy bit flip channel, then it's assumed that errors occur independently on each qubit.
- Only X errors (single-bit flip errors) are considered.
- Errors occur on one or fewer qubits.

The state that is analyzed is the following:

$$|\psi\rangle = \alpha|0\rangle + |1\rangle. \tag{2.27}$$

First of all, the qubit has to be encoded in a three-qubit entangled state and this is done by adding two *ancilla* qubits, which are appropriately initialized to $|0\rangle$. The entanglement is performed using two CNOT gates, one from qubit 1 to qubit 2 (CX_{12}) and one from qubit 1 to qubit 3 (CX_{13}).
The state of the entangled system is now:

$$|\psi_1\rangle = \alpha|000\rangle + \beta|111\rangle \tag{2.28}$$

The next step is to send the three qubits through independent single-qubit bit flip channels, which make the noisy communication channel in this case. During the communication, a bit flip error could have occurred on one of the three qubits (as assumed before), and therefore four possible states are presented:

1) $|\psi_2\rangle = \alpha|000\rangle + \beta|111\rangle \rightarrow$ no error.

2) $|\psi_2\rangle = \alpha|100\rangle + \beta|011\rangle \rightarrow$ error on the first qubit.

3) $|\psi_2\rangle = \alpha|010\rangle + \beta|101\rangle \rightarrow$ error on the second qubit.

4) $|\psi_2\rangle = \alpha|001\rangle + \beta|110\rangle \rightarrow$ error on the third qubit.

Now the error detection phase is next: in order to perform the syndrome measurement, two more ancilla qubits are needed, also initialized to $|0\rangle$. These two extra qubits are then entangled with the three qubits exiting from the bit flip channel using four CNOT gates: $CX_{14}, CX_{24}, CX_{15}, CX_{35}$.
At this point, the state of the quantum system is the following:

1) $|\psi_3\rangle = \alpha|000\rangle|00\rangle + \beta|111\rangle|00\rangle$

2) $|\psi_3\rangle = \alpha|001\rangle|01\rangle + \beta|110\rangle|01\rangle$

3) $|\psi_3\rangle = \alpha|010\rangle|10\rangle + \beta|101\rangle|10\rangle$

4) $|\psi_3\rangle = \alpha|100\rangle|11\rangle + \beta|011\rangle|11\rangle$

Now the error syndrome is measured through two measurement gates acting on each one of the last two qubits. The result is stored in a classical 2 bit-register.
There are four possible outcomes for the error syndrome:

1) **00** : $|\psi_4\rangle = \alpha|000\rangle + \beta|111\rangle$
2) **01** : $|\psi_4\rangle = \alpha|001\rangle + \beta|110\rangle$
3) **10** : $|\psi_4\rangle = \alpha|010\rangle + \beta|101\rangle$
4) **11** : $|\psi_4\rangle = \alpha|100\rangle + \beta|011\rangle$

Knowing this, the corresponding gate can be applied to correct the bit flip error:

1) **00**: no correction is needed.
2) **01**: Apply a Pauli X gate on the **third** qubit.
3) **10**: Apply a Pauli X gate on the **second** qubit.
4) **11**: Apply a Pauli X gate on the **first** qubit.

After the correction is applied, the state of the system is:

$$|\psi_5\rangle = \alpha|000\rangle + \beta|111\rangle \tag{2.29}$$

as it was initially sent. Lastly, the decoding phase is executed with the usage of two CNOT gates (CX_{12}, CX_{13}) on the first and second ancilla qubits so as to disentangle them from the first qubit.
The final state is:

$$|\psi_6\rangle = \alpha|0\rangle + \beta|1\rangle = |\psi\rangle. \tag{2.30}$$

Therefore, the original state has been successfully recovered, protecting the communication through a noisy channel from a bit flip error.

2.3.2 3-qubit phase flip code

Now the case of a phase error is considered, which error transforms the basis state as:

$$|0\rangle \to U|0\rangle = e^{i\phi}|0\rangle \tag{2.31}$$

$$|1\rangle \to U|1\rangle = e^{-i\phi}|1\rangle. \tag{2.32}$$

The same assumptions as the previous section are applied.
The generic state analyzed is:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle. \tag{2.33}$$

An example of a three-qubit phase flip code is shown in Figure 2.3. The image in Figure 2.3 can also be found in my GitHub repository.

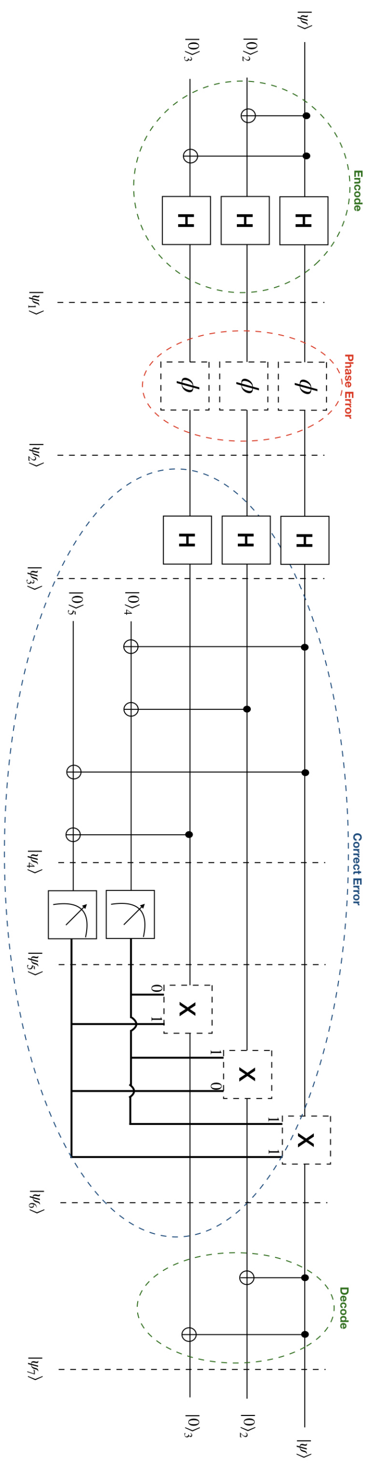


Figure 2.3: Example of a phase flip code on a one-qubit system. The figure highlights the encoding, phase flip error, error correction, and decoding phases.

Before sending the qubit through the noisy channel, it is encoded in an entangled state similar to the one applied previously by using two ancilla qubits initialized to $|0\rangle$. To do so CX_{12} and CX_{13} gates and then a Hadamard gate on each qubit are applied. This process leads to the encoded state:

$$|\psi\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)] + \beta[(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)]\} \quad (2.34)$$

The three qubits are then sent through the noisy communication channel and retrieved on the receiver's side. During the communication, a phase error could have occurred on any of the three qubits giving one of the following states:

1) $|\psi_2\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)] + \beta[(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)]\}$ $\rightarrow$ no error.

2) $|\psi_2\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[(e^{i\phi}|0\rangle + e^{-i\phi}|1\rangle)(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)] + \beta[(e^{i\phi}|0\rangle - e^{-i\phi}|1\rangle)(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)]\}$ $\rightarrow$ error on the first qubit.

3) $|\psi_2\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[(|0\rangle + |1\rangle)(e^{i\phi}|0\rangle + e^{-i\phi}|1\rangle)(|0\rangle + |1\rangle)] + \beta[(|0\rangle - |1\rangle)(e^{i\phi}|0\rangle - e^{-i\phi}|1\rangle)(|0\rangle - |1\rangle)]\}$ $\rightarrow$ error on the second qubit.

4) $|\psi_2\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[(|0\rangle + |1\rangle)(|0\rangle + |1\rangle)(e^{i\phi}|0\rangle + e^{-i\phi}|1\rangle)] + \beta[(|0\rangle - |1\rangle)(|0\rangle - |1\rangle)(e^{i\phi}|0\rangle - e^{-i\phi}|1\rangle)]\}$ $\rightarrow$ error on the third qubit.

In order to correct the state received at the other end of the quantum communication channel from possible errors, the Hadamard gate is applied on all three qubits again. Correspondingly, the resulting possible states of the system are:

1) $|\psi_3\rangle = \alpha|000\rangle + \beta|111\rangle$

2) $|\psi_3\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[(e^{i\phi}|0\rangle + e^{i\phi}|1\rangle + e^{-i\phi}|0\rangle - e^{-i\phi}|1\rangle)|00\rangle] + \beta[(e^{i\phi}|0\rangle + e^{i\phi}|1\rangle - e^{-i\phi}|0\rangle + e^{-i\phi}|1\rangle)|11\rangle]\}$

3) $|\psi_3\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[|0\rangle(e^{i\phi}|0\rangle + e^{i\phi}|1\rangle + e^{-i\phi}|0\rangle - e^{-i\phi}|1\rangle)|0\rangle] + \beta[|1\rangle(e^{i\phi}|0\rangle + e^{i\phi}|1\rangle + e^{-i\phi}|0\rangle - e^{-i\phi}|1\rangle)|1\rangle]\}$

4) $|\psi_3\rangle = \frac{1}{\sqrt{2^3}}\{\alpha[|00\rangle(e^{i\phi}|0\rangle + e^{i\phi}|1\rangle + e^{-i\phi}|0\rangle - e^{-i\phi}|1\rangle)] + \beta[|11\rangle(e^{i\phi}|0\rangle + e^{i\phi}|1\rangle - e^{-i\phi}|0\rangle + e^{-i\phi}|1\rangle)]\}$

Expanding the phase as $e^{i\phi} = \cos\phi + i\sin\phi$ (Euler's formula) and simplifying terms the result obtained is the following:

1) $|\psi_3\rangle = \alpha|000\rangle + \beta|111\rangle$

2) $|\psi_3\rangle = \alpha(\cos\phi|000\rangle + i\sin\phi|100\rangle) + \beta(\cos\phi|111\rangle + i\sin\phi|011\rangle)$

3) $|\psi_3\rangle = \alpha(\cos\phi|000\rangle + i\sin\phi|010\rangle) + \beta(\cos\phi|111\rangle + i\sin\phi|101\rangle)$

4) $|\psi_3\rangle = \alpha(\cos\phi|000\rangle + i\sin\phi|001\rangle) + \beta(\cos\phi|111\rangle + i\sin\phi|110\rangle)$

Two more ancilla qubits in the state $|0\rangle$ are added to the system. Moreover, they are entangled with the three bits received by using the following gates: $CX_{14}, CX_{24}, CX_{15}, CX_{35}$. The state of the system now is:

1) $|\psi_4\rangle = \alpha|000\rangle|00\rangle + \beta|111\rangle|00\rangle$

2) $|\psi_4\rangle = \alpha(\cos\phi|000\rangle|00\rangle + i\sin\phi|100\rangle)|11\rangle + \beta(\cos\phi|111\rangle|00\rangle + i\sin\phi|011\rangle|11\rangle)$

3) $|\psi_4\rangle = \alpha(\cos\phi|000\rangle|00\rangle + i\sin\phi|010\rangle|10\rangle) + \beta(\cos\phi|111\rangle|00\rangle + i\sin\phi|101\rangle|10\rangle)$

4) $|\psi_4\rangle = \alpha(\cos\phi|000\rangle|00\rangle + i\sin\phi|001\rangle|01\rangle) + \beta(\cos\phi|111\rangle|00\rangle + i\sin\phi|110\rangle|01\rangle)$

The two ancilla qubits just added are then measured. They are used to measure the syndrome, which can be used to diagnose and then correct any error that occurred on the three qubits. There are four possible outcomes for the error syndrome:

1) **00**: $|\psi_5\rangle = \alpha|000\rangle + \beta|111\rangle$

2) **11**: $|\psi_5\rangle = \alpha|100\rangle + \beta|011\rangle$

3) **10**: $|\psi_5\rangle = \alpha|010\rangle + \beta|101\rangle$

4) **01**: $|\psi_5\rangle = \alpha|001\rangle + \beta|110\rangle$

Therefore, the state of the three qubits can be corrected by applying the corresponding gate:

1) **00**: No correction is needed.

2) **11**: Apply a Pauli X gate to the **first** qubit.

3) **10**: Apply a Pauli X gate to the **second** qubit.

4) **01**: Apply a Pauli X gate to the **third** qubit.

After which the state of the system will be:

$$|\psi_6\rangle = \alpha|000\rangle + \beta|111\rangle, \tag{2.35}$$

as originally sent. Now, in order to find the state of the qubit that was intended to be transferred, the system is decoded from the three-qubit state by disentangling the qubit from the two ancillas. Thus the gates CX_{12} and CX_{13} are applied and the result is:

$$|\psi_7\rangle = \alpha|0\rangle + \beta|1\rangle = |\psi\rangle. \tag{2.36}$$

The state of the qubit $|\psi\rangle$ has been successfully communicated through a noisy channel, protecting it from any single phase error.

2.4 Shor code

The Shor code is fundamental in quantum error correction since it is a fairly simple code that can protect a single qubit against the effect of an *arbitrary* error. It is a combination of the previously seen three-qubit bit flip and phase flip codes.

"One of the main difficulties of quantum computation is that decoherence destroys the information in a superposition of states contained in a quantum computer, thus making long computations impossible. It is shown how to reduce the effects of decoherence for information stored in quantum memory, assuming that the decoherence process acts independently on each of the bits stored in memory. This involves the uses of a quantum analog of error-correcting codes."

This was an extract from the abstract of "*Scheme for reducing decoherence in quantum computer memory*"(1995) which is the paper where Peter W. Shor introduced its homonymous code [12].

Definition 2.4.1. Decoherence: In quantum mechanics, decoherence is a unitary process that entangles a qubit with the environment.

The fundamental assumption of this procedure is that decoherence only affects one qubit of the superposition, while the other qubits are unaffected. This assumption corresponds to the one in classical information theory of the independence of noise and the validity of it will be discussed later.
The Shor code is a useful procedure to store an arbitrary state of n qubits in $9n$ qubits, protecting the information from decoherence.
Firstly, the qubit is encoded using the three-qubit phase flip code:

$$|0\rangle \rightarrow |+++\rangle \tag{2.37}$$

$$|1\rangle \rightarrow |---\rangle \tag{2.38}$$

After that, each one of the three resulting qubit is encoded using the three-qubit bit flip code, which results in:

$$|0\rangle \rightarrow |0_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \tag{2.39}$$

$$|1\rangle \rightarrow |1_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle) \tag{2.40}$$

so that the encoded state is:

$$|\psi_L\rangle = \alpha|0_L\rangle + \beta|1_L\rangle. \tag{2.41}$$

The circuit implementing this procedure is shown in Figure 2.4.
This method of combining multiple types of codes in a cascade is called *concatenation.*

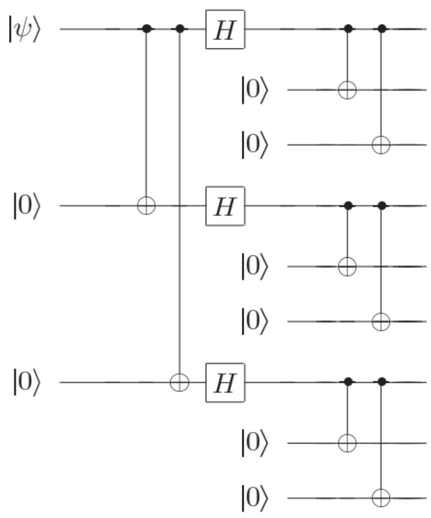


Figure 2.4: Circuit for the encoding phase of the Shor code

The important effect of this code is to protect both from bit flip and phase flip errors *on any qubit.*
After the encoding phase and the error has (potentially) occurred, the encoded state is read. This is executed by using an ancilla qubit, i.e. by entangling the encoded qubits with other qubits and then measuring these other qubits instead of directly measuring the encoded qubits, to avoid the collapse of the state.
Suppose that no decoherence has occurred so that the first three qubits are in state $|000\rangle + |111\rangle$. Consequently, the other two sets are in state $|000\rangle + |111\rangle$ too. Therefore, even if decoherence has occurred on one qubit when we measure the nine qubits (in the Bell basis $|000\rangle \pm |111\rangle, |001\rangle \pm |110\rangle, |010\rangle \pm |101\rangle, |100\rangle \pm |011\rangle$) we can infer the expected measurement outcome by examining the majority result among the three sets of measurements (*majority voting*). Consequently, we can determine whether the encoded bit corresponds to 0 or 1. Nonetheless, this approach does not enable us to recover the initial qubit's superposition or entanglement, as measuring the nine qubits essentially projects all of them onto a subspace. Hence, to protect the superposition state of the encoded qubit, our approach essentially involves measuring the decoherence while avoiding the measurement of the qubits' states. The result of this procedure is in effect reversing the decoherence.
Let's first suppose that decoherence occurs on the first qubit. This supposition is totally arbitrary since the process would be exactly the same if any of the other qubits decoheres. Since $|0\rangle$ and $|1\rangle$ form a basis for the first qubits, only these two states are to be considered.

The decoherence process is sketched as:

$$|e_0\rangle|0\rangle \rightarrow |a_0\rangle|0\rangle + |a_1\rangle|1\rangle \tag{2.42}$$
$$|e_0\rangle|1\rangle \rightarrow |a_2\rangle|0\rangle + |a_3\rangle|1\rangle \tag{2.43}$$

where $|e_0\rangle$ are the basis vector and $|a_i\rangle$ are state of the environment.
Now it is shown what happens to $|0_L\rangle$ when it decoheres. It goes to the superposition:

$$(\frac{1}{\sqrt{2}}[(|a_0\rangle|0\rangle + |a_1\rangle|1\rangle)|00\rangle + (|a_2\rangle|0\rangle + |a_3\rangle|1\rangle)|11\rangle] \tag{2.44}$$

It has to be noted that $|0_L\rangle$ is a nine-qubit state but above only one block of three qubits is analyzed for conciseness.
Now it is written in the Bell basis:

$$\begin{aligned}
&\frac{1}{2\sqrt{2}}(|a_0\rangle + |a_3\rangle)(|000\rangle + |111\rangle)\\
+&\frac{1}{2\sqrt{2}}(|a_0\rangle - |a_3\rangle)(|000\rangle - |111\rangle)\\
+&\frac{1}{2\sqrt{2}}(|a_1\rangle + |a_2\rangle)(|100\rangle + |011\rangle)\\
+&\frac{1}{2\sqrt{2}}(|a_1\rangle - |a_2\rangle)(|100\rangle - |011\rangle).
\end{aligned} \tag{2.45}$$

The same goes for the vector $|000\rangle - |111\rangle$:

$$\begin{aligned}
&\frac{1}{2\sqrt{2}}(|a_0\rangle + |a_3\rangle)(|000\rangle - |111\rangle)\\
+&\frac{1}{2\sqrt{2}}(|a_0\rangle - |a_3\rangle)(|000\rangle + |111\rangle)\\
+&\frac{1}{2\sqrt{2}}(|a_1\rangle + |a_2\rangle)(|100\rangle - |011\rangle)\\
+&\frac{1}{2\sqrt{2}}(|a_1\rangle - |a_2\rangle)(|100\rangle + |011\rangle).
\end{aligned} \tag{2.46}$$

The takeaway from this is that the environmental state remains identical for corresponding vectors resulting from the decoherence of the two quantum states encoding 0 and 1 ($|0_L\rangle$ and $|1_L\rangle$). Hence, it becomes feasible to recover the initial state of the encoded vector while preserving unitary evolution. This is achieved by introducing a small number of ancillary qubits that provide information regarding which qubit experienced decoherence and whether there was a change in the sign of the Bell superposition. Through the measurement of these ancillary qubits, it is possible to successfully recover the original

state. Moreover, any pre-existing superposition of basic Bell states remains unaffected since the coefficients remain consistent, regardless of whether the initial vector experienced decoherence from state $|000\rangle + |111\rangle$ or $|000\rangle - |111\rangle$.
By measuring the ancillary qubits that tell which qubit was decohered, the original state was in effect restored.

Now a more detailed description of this code is provided. After the state has been encoded, the (potential) error must be detected and subsequently corrected. By performing the correction part of the circuit seen for the bit flip code (Figure 2.2) for each block of three qubits, one can detect and correct single X errors. On the other hand, phase flip errors (Z) are corrected by comparing the signs between the three blocks. This is achieved through a circuit such as the one shown in Figure 2.5. The initial group of six CNOT gates assesses the phase comparison between blocks one and two, while the subsequent set of CNOT gates between blocks two and three. It's worth emphasizing that a phase flip occurring on any single qubit within a three-qubit block produces identical effects. This explains why the 9-qubit code is a degenerate code[1]. In this type of code, the mapping between correctable errors and unique states is not unique. Therefore, as long as we are aware of the block where the error occurs, the specific qubit to which we apply the correction operator becomes inconsequential.

If more than one qubit within a nine-tuple experience decoherence, the encoding scheme becomes nonfunctional. However, the likelihood of this scenario is directly proportional to the square of the probability that a single qubit experiences decoherence. In other words, if each qubit undergoes decoherence with a probability p, then the probability that k qubits remain unaffected is represented as $(1-p)$. In our scheme, we replace each qubit with nine qubits. The probability that at least two qubits within a specific nine-tuple encounter decoherence is calculated as $1-(1+8p)(1-p)^8$, approximately equal to $36p^2$. Consequently, the probability that our set of $9k$ qubits can be decoded to restore the original quantum state is approximately $(1-36p^2)^k$. Hence, for a probability of decoherence less than $\frac{1}{36}$, we have an enhanced storage method for preserving quantum-coherent states involving a large number of qubits.

[1]Quantum codes are said to be degenerate when different types of errors have the same effect on the codestates.

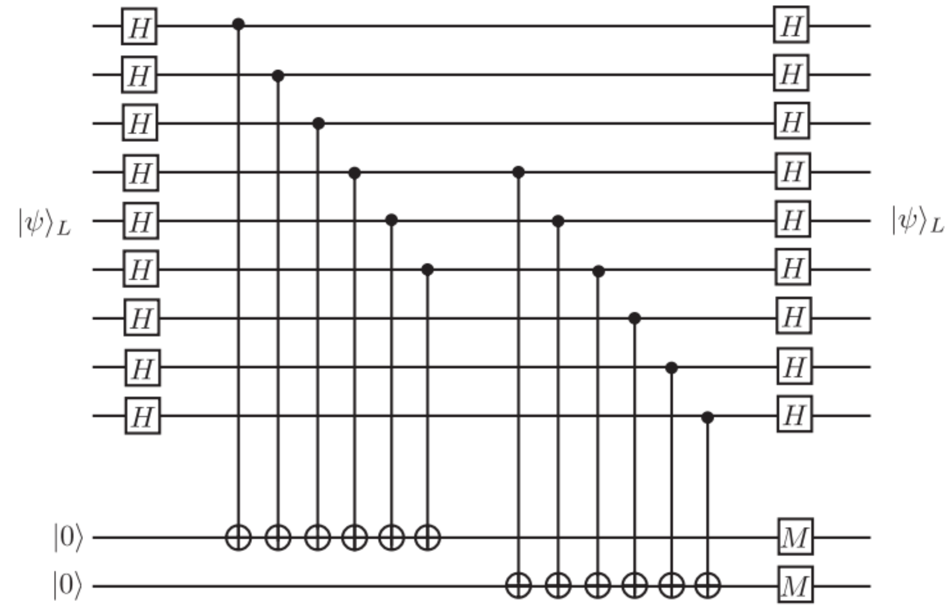


Figure 2.5: 9-qubit circuit correction section required to deal with a phase flip error (Z). $|\psi_L\rangle$ is the encoded state as in (2.41).

Chapter 3

Validation of Quantum Error Correction codes with IBM Quantum

This last chapter represents the central objective of this thesis, which is to empirically evaluate the effectiveness of quantum error correction (QEC) implementations in enhancing the accuracy of quantum computing systems, particularly in mitigating bit flip errors. This evaluation is conducted using the comprehensive toolset provided by IBM Quantum [16].

Within this chapter, a practical assessment is conducted to verify the impact of quantum error correction on quantum computing accuracy. IBM Quantum Lab is utilized as a robust platform to construct and simulate a quantum error correction circuit specifically designed to rectify bit flip errors in a qubit initially prepared in the $|+\rangle$ state.

This verification process unfolds through distinct phases:

1) **Measurement of the $|+\rangle$ state**: Building upon the foundational knowledge, IBM Quantum Lab's capabilities are employed to precisely measure and analyze the $|+\rangle$ state, establishing a baseline for subsequent evaluations.

2) **Encoding and decoding parts are added**: Following the initial measurement, the encoding and decoding phase is added to the previous circuit and simulated to perform both those two phases, followed by outcome measurement.

3) **Complete QEC circuit**: The same procedure is repeated but with the implementation of the complete quantum error correction code designed to protect from bit flip errors.

4) **Error hardware implementation**: Leveraging IBM Quantum Lab's specialized simulation capabilities, the aim is to emulate error hardware intricacies closely,

effectively bridging the gap between theoretical principles and practical quantum environments.

5) **Real Quantum Device Testing**: The ultimate validation of this research involves the practical application of error correction on an IBM-provided real quantum device, serving as a pivotal step in assessing the actual impact of quantum error correction techniques.

Throughout this chapter, data is collected, analyses are conducted, and results are presented graphically. The objective is to empirically verify whether quantum error correction, as implemented using IBM Quantum Lab, significantly elevates the accuracy of quantum computing by providing protection against bit flip errors, thus contributing to the advancement of quantum computing technology.

3.1 Simulation of Error Correction Circuits

This section focuses on the simulation of error correction circuits in ideal conditions, without the implementation of hardware error. The primary objective here is to verify the efficacy of these circuits in mitigating bit flip errors within quantum computations. IBM Quantum Lab's versatile capabilities are employed to construct and analyze these circuits. The state examined is the superposition state $|+\rangle$. Subsequently, circuits encompassing both encoding and decoding phases are simulated, followed by the implementation of the complete quantum error correction (QEC) code designed to address bit flip errors. This simulation phase establishes a foundational baseline for subsequent analyses, offering insights into the theoretical potential of quantum error correction under ideal circumstances.

3.1.1 Constructing and measuring the superposition state

The first circuit presented is the most basic version and it only includes the state preparation and subsequent measurement. From now on this circuit will be referred to as the "*basic circuit*". This simplest circuit is generated and analyzed so that later the results from its study can be compared to the ones from the analysis of the complete circuit, which will include a QEC implementation.

Firstly, some preliminary code is needed to load up the necessary libraries:

```
from qiskit import QuantumCircuit
import matplotlib.pyplot as plt
```

The first library provides the methods and tools for defining and creating a quantum circuit, while the second one is for plotting the data and printing out the circuit schematics

(as seen in Figure 3.1).
Then the quantum circuit has to be built by using the following commands:

```
# Define the circuit
bit_flip = QuantumCircuit(1,1)
```

With the last command, a quantum circuit was created: it's a circuit formed by 1 qubit, as specified by the first parameter of the function, and a 1-bit classical register, as specified by the second parameter.
The next step would be to initialize the qubit state to $|0\rangle$, but Qiskit already does this automatically.
The system is now described as follows:

$$|\psi\rangle = |0\rangle \tag{3.1}$$

At this point, the superposition state ($|+\rangle$) is set by applying a Hadamard gate (or H gate) to the qubit, implementing the following operation:

$$H|\psi\rangle = H|0\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = |+\rangle. \tag{3.2}$$

This is achieved through the following command, which applies a Hadamard gate on the first qubit, which here is numbered as the qubit 0:

```
# Prepares qubit in the desired initial |+> state
bit_flip.h(0)
```

Next, the qubit is measured through the apposite Qiskit command:

```
# Measure
bit_flip.measure(0,0)
```

The two parameters of the `QuantumCircuit.measure()` method expresses, respectively, which qubit to measure and in which classical register store the result of the measurement.
The result of this simple code is the circuit (which is obtained through the apposite Qiskit `print()` method) shown in Figure 3.1.

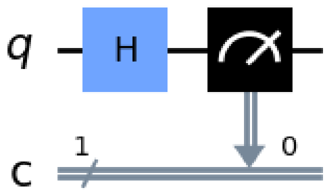


Figure 3.1: Qiskit circuit preparing the superposition state ($|+\rangle$) and measuring it.

3.1.2 Adding the encoding and decoding sections

In this subsection, the focus shifts to the examination of a quantum circuit that integrates encoding and decoding phases. IBM Quantum's versatile simulation capabilities are employed to analyze how this circuit functions within the context of mitigating bit flip errors. From now on this circuit will be referred to as the "*encoding circuit*".
As in the previous subsection, the (main) qubit is prepared in the superposition state $|+\rangle$, and then measurements are carried out. However, the additional encoding and decoding phases demand the usage of two ancilla qubits to encode the state.
The same libraries as before are required and loaded but two more are imported for convenience:

```python
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister
import matplotlib.pyplot as plt
```

The additional two libraries (`QuantumRegister, ClassicalRegister` to better manage qubits and bits in the circuit. `QuantumRegister()` creates a quantum register, which is a collection of qubits, and permits the management of them, by calling and acting on them separately. On the other hand, `ClassicalRegister()` implements a classical register, which is a collection of bits, which is used to record and store the measurement outcomes.
Then the circuit is created:

```python
# Define the quantum and classical registers
q = QuantumRegister(3, 'q')
c2 = ClassicalRegister(1, 'c')

# Define the circuit
bit_flip = QuantumCircuit(q,c)
```

The first parameter of `QuantumRegister()` and `ClassicalRegister()` expresses the size of the register, i.e. how many qubits or bits (respectively) are stored in the register, while the second parameter indicates the name of the register, which is used to call the register's content.
The system is now in the state:

$$|\psi\rangle = |000\rangle \tag{3.3}$$

since, as already highlighted in the previous section, Qiskit initializes by default each qubit to the state $|0\rangle$.
The main qubit (which is chosen to be the first qubit, numbered as 0) is now prepared in the superposition state $|+\rangle$, implementing the operation already seen in (3.2):

```
# Prepares the first qubit (main) in the desired initial state (|+>)
bit_flip.h(q[0])
```

Now the state of the system is as follows:

$$|\psi\rangle = |+\rangle \otimes |00\rangle \tag{3.4}$$

Next, the encoding has to be carried out. This is executed by applying two CNOT (CX) gates as shown in Figure 3.2. This is achieved through the following Qiskit command lines:

```
# Encodes the qubit in a three-qubit entangled state
bit_flip.cx(q[0], q[1])
bit_flip.cx(q[0], q[2])
```

Now the system state is as follows:

$$|\psi\rangle = \frac{|000\rangle + |111\rangle}{\sqrt{2}}. \tag{3.5}$$

The system is now decoded from the encoded state ((3.5)) to the original state ((3.4)). This is done by reapplying the same CNOT gates as in the encoding phase. This process disentangles the state, returning the system to the original state $|\psi\rangle = |+\rangle \otimes |00\rangle$.
Lastly, only the first qubit (q_0) is measured, through the command:

```
# Check the state of the initial qubit
bit_flip.measure(q[0], c[0])
```

The result of this code is the circuit (which is obtained through the apposite Qiskit `print()` method) shown in Figure 3.2.

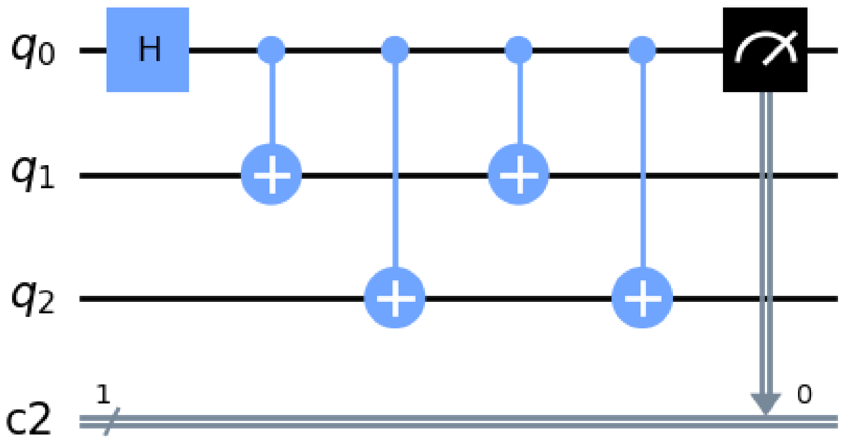


Figure 3.2: Qiskit circuit executing a three-qubit encoding and decoding on a qubit in superposition state $|+\rangle$ and measuring it.

3.1.3 QEC implementation

In this section, an examination of the complete QEC circuit designed for the three-qubit bit flip error correction is presented. From now on this circuit will be referred to as the "*complete circuit*". As in the previous sections, the initial qubit is initialized in a superposition state $|+\rangle$. Then it is encoded following the three-qubit bit flip model and the error correction phase is executed. Lastly, the system is decoded and measurement is carried out.

The same libraries as the previous section are imported:

```
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister
```

Then the circuit is generated using the specific Qiskit methods:

```
# Define the quantum and classical registers
q = QuantumRegister(5, 'q')
c0 = ClassicalRegister(2, 'c0')
c2 = ClassicalRegister(1, 'c2')

# Define the circuit
bit_flip = QuantumCircuit(q, c0,c2)
```

Now the system's state is:

$$|\psi\rangle = |00000\rangle. \tag{3.6}$$

The main qubit (which is chosen to be the first qubit, numbered as 0) is now prepared in the superposition state $|+\rangle$, implementing the operation already seen in (3.2):

```
# Prepares qubit in the desired initial state
bit_flip.h(q[0])
```

Now the state of the system is as follows:

$$|\psi\rangle = |+\rangle \otimes |0000\rangle \tag{3.7}$$

Now the encoding phase is carried out as done in the previous section.

The following code section is what discerns this circuit from the one in the previous section. In this section, the two additional ancilla qubits (q_3 and q_4) are entangled with the three other qubits.

```
# Adds additional two qubits for error-correction
bit_flip.cx(q[0], q[3])
bit_flip.cx(q[1], q[3])
bit_flip.cx(q[0], q[4])
bit_flip.cx(q[2], q[4])
```

This entanglement is fundamental since it's the first step for the *syndrome measurement*. The syndrome measurement is "stored" in these two additional ancilla qubits, which are then measured to extract the syndrome. The result of this measurement is then saved into a two-bit classical register, which was defined at the start of this code.

```
# Measure the two additional qubits
bit_flip.measure(q[3], c0[0])
bit_flip.measure(q[4], c0[1])
```

After the state of the last two ancilla qubits, which gives out the syndrome, is measured, based on the result of this measurement the proper correction is applied. This correction is executed through a particular type of gate, the *conditional* X gate, which is a modified version of the Pauli X gate (already seen in (1.35)). This gate applies an X gate on the qubit only if the associated classical register has a particular value.
In the case under study, the syndrome measurement is read as already discussed in Section 2.3.1 which is proposed again here for convenience:

1) **00**: No correction is needed.

2) **11**: Apply a Pauli X gate to the **first** qubit.

3) **10**: Apply a Pauli X gate to the **second** qubit.

4) **01**: Apply a Pauli X gate to the **third** qubit.

This is implemented in code as:

```
    # Do error correction
    bit_flip.x(q[2]).c_if(c0, 1)
    bit_flip.x(q[1]).c_if(c0, 2)
    bit_flip.x(q[0]).c_if(c0,3)
```

Naturally, if no error has occurred, then the syndrome measurement gives out 0, and therefore, as displayed in the list above, no correction is needed, and no X gate has to be applied. At this point, the decoding process is carried out. As already previously seen, this is done through the application of two CNOT gates, the same as in the encoding process. The command lines for executing this are the following:

```
    # Decodes the qubit from the three-qubit entangled state
bit_flip.cx(q[0], q[1])
bit_flip.cx(q[0], q[2])
```

Lastly, the first initial qubit is measured:

```
    # Check the state of the initial qubit
    bit_flip.measure(q[0], c2[0])
```

The result of this code is the circuit (which is obtained through the apposite Qiskit `print()` method) shown in Figure 3.3.

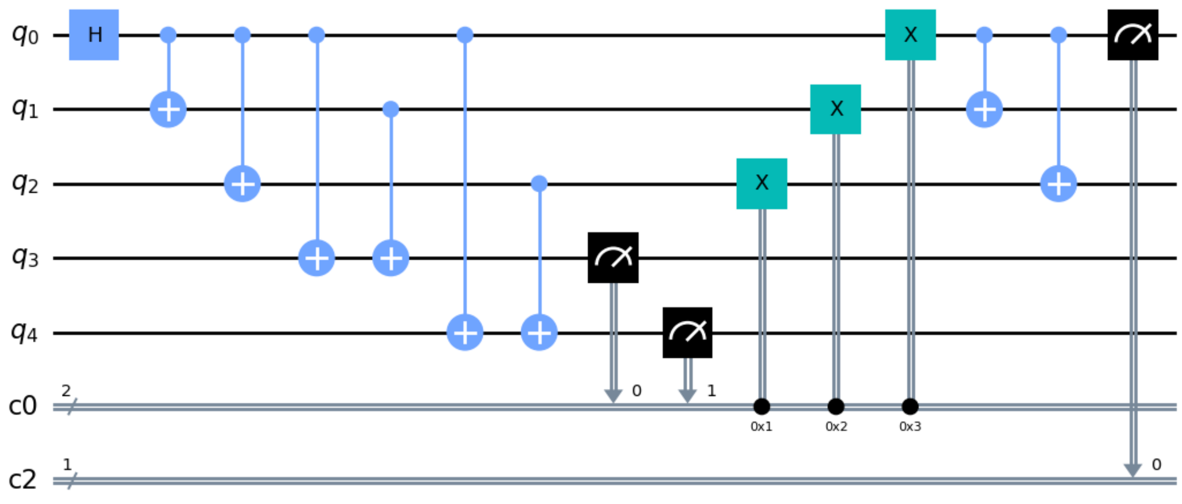


Figure 3.3: Qiskit circuit implementing a full three-qubit bit flip error correction on a $|+\rangle$ state.

3.1.4 Executing the circuit, extracting and plotting the data

The circuits earlier defined have to be run by taking advantage of the simulator provided by IBM. In this section, the basics of this procedure, tailored to this particular case, are going to be discussed.
After the circuit is executed, the data has to be extracted, stored, and then plotted and analyzed. This section will also address these tasks.

Each instance of the simulator runs the circuit a number of times equal to the value of the `shots` variable, which, according to Qiskit default settings, is set to 1024 (`shots = 1024`). Additionally, a `for` loop is added so that `n = 10000` instances of the simulator are executed. This precise value of `n` has been chosen for two reasons: it provides a relevant statistical sample while maintaining computation times at a reasonable level.
Now this process is shown in more detail by explicating the corresponding command lines.
First, some additional libraries are to be loaded:

```
import numpy as np
from qiskit import transpile, assemble, Aer, execute
from qiskit.visualization import plot_histogram
from qiskit.providers.aer import AerSimulator
```

The library `qiskit.visualization` is for plotting the data and the circuit, while `numpy`

is required for analyzing the data and the other libraries are necessary for running the circuit on an IBM simulator.
The next step is to define the `n` variable, which is required to loop the circuit:

```
# Number of times to repeat the circuit
n = 10000
```

Now, the simulator framework is selected. For this study, it has been chosen the `Aer Simulator` framework, which includes various IBM simulators:

- **Statevector Simulator**: This simulator calculates the quantum state vector of the quantum circuit, which represents the probability amplitudes of all possible states. It's used for ideal, noise-free simulations.
- **QASM Simulator**: This simulator simulates the measurement outcomes of a quantum circuit. It returns the counts of different measurement results (0s and 1s) and can be used also to simulate noisy quantum devices.
- **Unitary Simulator**: This simulator calculates the unitary matrix representation of a quantum circuit, which can be useful for analyzing the quantum gates in your circuit.
- **Density Matrix Simulator**: This simulator calculates the density matrix of a quantum circuit, which provides a more detailed description of the quantum state, including information about quantum entanglement and mixed states.

Since the goal of this thesis is to study QEC on noisy channels, the QASM simulator is chosen, since, as seen in its description above, it is the default for noisy channels, although in the simulated circuit just described, all the gates are perfectly unitary, i.e. not noisy. This decision was made to ensure some continuity with the real quantum device.
Then two lists are created to store the outcomes of values 0 and 1 from each instance's measurement:

```
# Lists to store the '0' and '1' measurement outcomes
results_0 = []
results_1 = []
```

Subsequently, the for loop is created:

```
# Create a file to write measurements
with open('measurements_0_h.txt', 'w') as file0,
 ↪  open('measurements_1_h.txt', 'w') as file1:
```

```
for i in range(n):
    # Compile the circuit
    compiled_circuit = transpile(bit_flip, simulator)

    # Run the circuit
    job = simulator.run(compiled_circuit)

    # Get the result
    result = job.result()

    # Get the counts for '00' and '11' outcomes
    counts = result.get_counts()
    count_0 = counts.get('0 00', 0)
    count_1 = counts.get('1 00', 0)

    # Write the measurement outcomes to the file in columns
    file0.write(f"{count_0}")

    # Return for readability
    file0.write("\n")

    # Write the measurement outcomes to the file in columns
    file1.write(f"{count_1}")

    # Return for readability
    file1.write("\n")

    # Append the outcomes to the lists
    results_0.append(count_0)
    results_1.append(count_1)
```

In this section of code, two .txt files are open and the results from the measurements are stored in those two files. Also, the circuit, using the previously selected simulator, is run inside a `for loop`, going from 1 to the previously defined `n`. Lastly, the 0 and 1 outcomes are appended to the lists previously created.
Then the standard deviation and the mean are calculated for the 0 and 1 lists of outcomes. Lastly, the results are plotted and fitted with a normal distribution. The code implementing these last tasks is here omitted for conciseness, but it can be found in my

GitHub repository [17].

3.1.5 Simulation result and analysis

This section presents the outcomes of the measurements and analysis conducted with the simulation framework. The comprehensive simulations cover various scenarios, including the measurement of the $|+\rangle$ state and the execution of a comprehensive QEC circuit designed for bit flip error mitigation. Taking advantage of the simulator's capabilities, the performance and efficacy of error correction strategies are examined. The disclosed results hold significance in evaluating the theoretical foundations of QEC in preserving quantum information.
For each one of the three circuits, the mean probability of both '0' and '1' outcomes and the standard deviation on those probabilities are given and then two graphs are produced, one for the '0' occurrences and one for the '1' occurrences.
In Figures 3.4,3.5 and 3.6 on the X-axis is displayed the "Measurement Outcome" which is how many times the value '0' or '1' occurred in each one of the `n = 10000` instances. This value ranges from 0 to 1024 but here the limits of the X-axis are set to 440 and 580 since values outside of this range are negligible. On the other hand, on the Y-axis is displayed the "Probability Density" which is a value that ranges from 0 to 1 and expresses the probability of a certain occurrence value to arise, calculated on how many times that value occurred in the total of the `n` instances. The total sum of the probability density on the two graphs is 1.

State	Expected value	$\overline{p}$	$\sigma_{\overline{p}}$
'0'	0.5000	0.4997	0.0155
'1'	0.5000	0.5003	0.0155

Table 3.1: Results for the basic circuit

State	Expected value	$\overline{p}$	$\sigma_{\overline{p}}$
'0'	0.5000	0.4997	0.0158
'1'	0.5000	0.5003	0.0158

Table 3.2: Results for the encoding circuit

State	Expected value	$\overline{p}$	$\sigma_{\overline{p}}$
'0'	0.500	0.4999	0.0156
'1'	0.500	0.5000	0.0156

Table 3.3: Results for the complete circuit

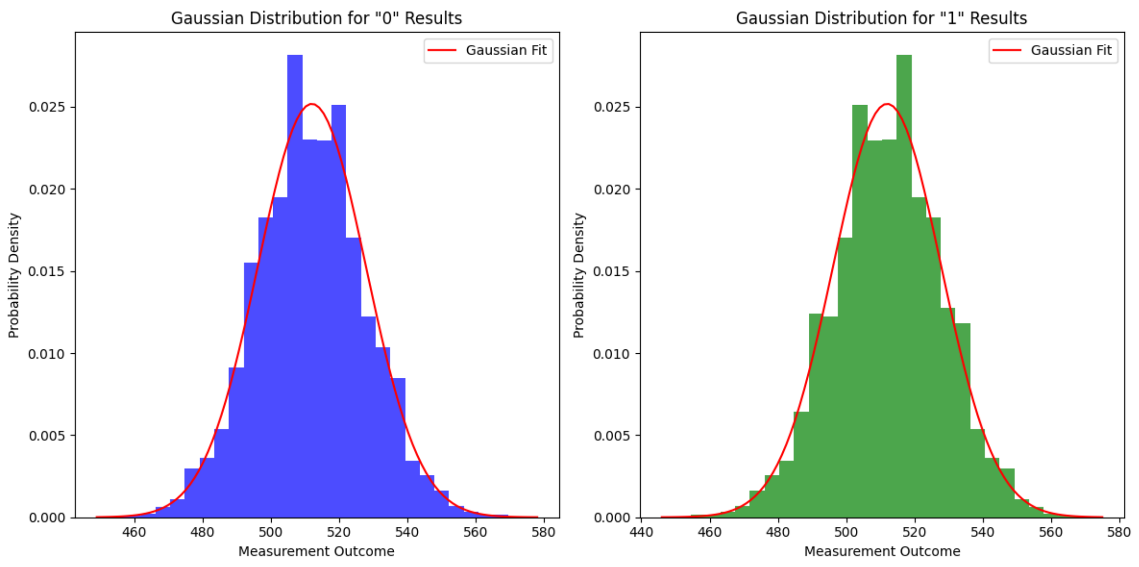


Figure 3.4: Plotted result of the '0' and '1' occurrences for the basic circuit. The histograms (blue and green) are fitted with a Gaussian distribution (red).

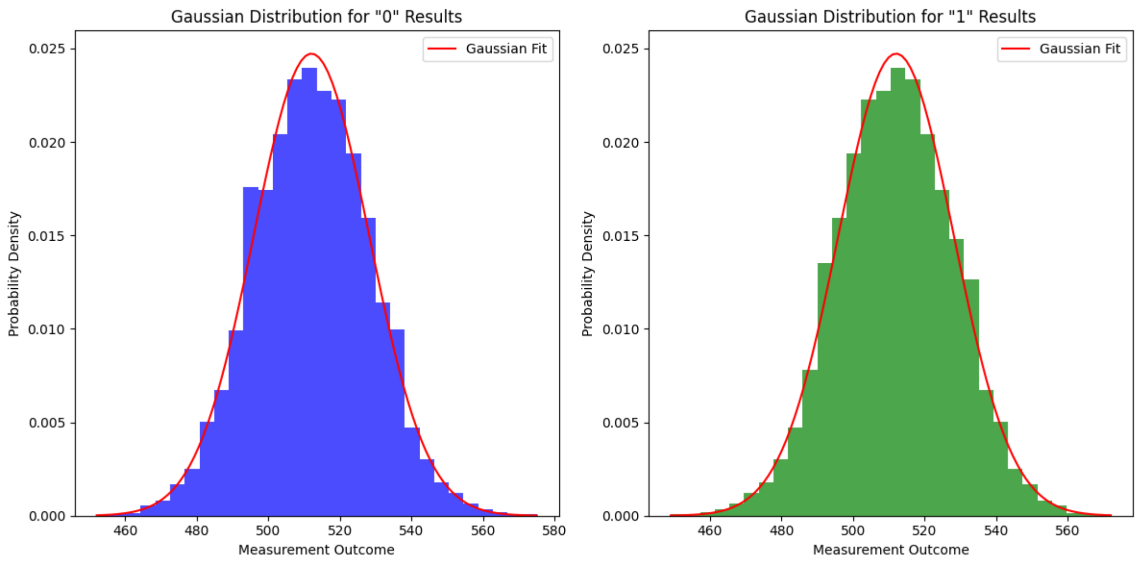


Figure 3.5: Plotted result of the '0' and '1' occurrences for the encoding circuit. The histograms (blue and green) are fitted with a Gaussian distribution (red).

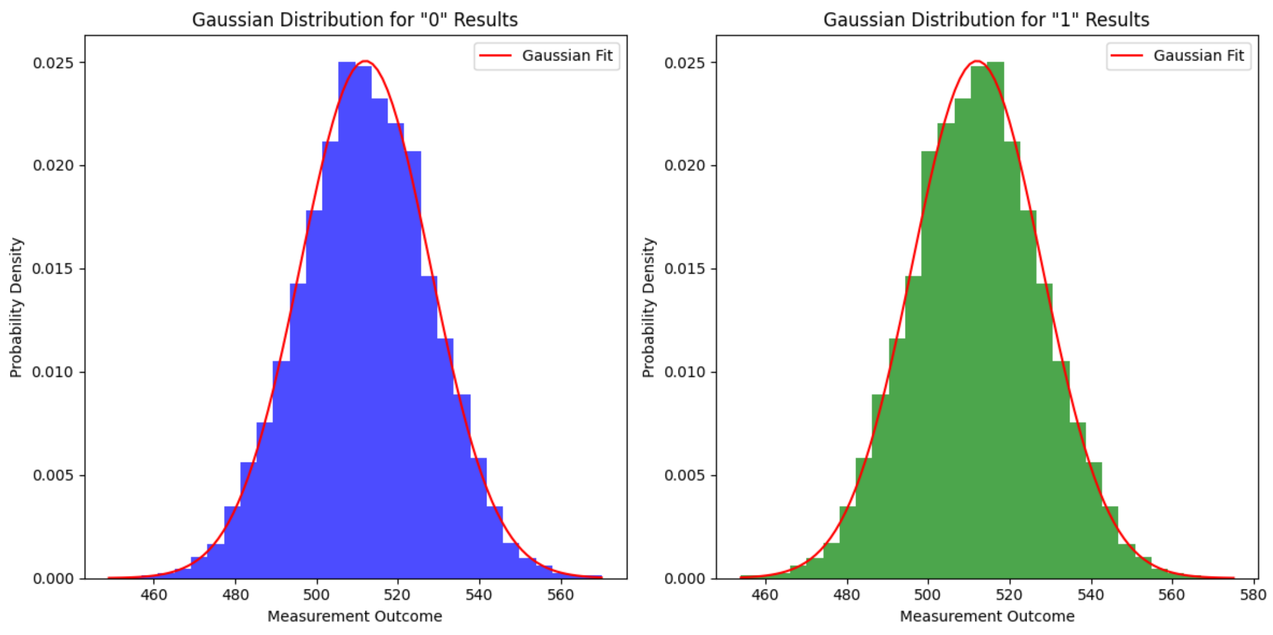


Figure 3.6: Plotted result of the '0' and '1' occurrences for the complete circuit. The histograms (blue and green) are fitted with a Gaussian distribution (red).

The results shown in all three Tables 3.1, 3.2, 3.3 and on all three Figures 3.4, 3.5, 3.6 how the simulator recreates the ideal scenario of the measurement of the superposition state $|+\rangle$. Indeed, the fact that not all the 10000 instances returned perfectly 50% '0' and 50% '1' as results is expected due to scarse data sample (`shots`=1024). As a matter of fact, if one were to run a circuit with the simulator with an increasing `shots` value, the outcome would always progressively approach the theoretical value and the Gaussian distribution fitting the plot would steadily peak more and more. Therefore, the error $\sigma_{\overline{p}}$ is only a statistical error, due to the scaristy of the data sample.
It is also interesting to notice how the simulated nature of this data is highlighted by the symmetry of the '0' and '1' plots.

3.2 QEC on a real quantum device

In this section, the research progresses towards the practical execution of the designed quantum error correction (QEC) code on ibm_nairobi, the real quantum hardware provided by IBM. This step serves as an empirical validation of the established theoretical foundations. The focus of this section is to assess the viability of QEC implementations in protecting quantum computing against bit flip errors.

Differently from what has been done in Section 3.1, in this section the initial state studied is the $|0\rangle$, meaning that at the start of the circuit, the first qubit is initialized at the state $|0\rangle$. This variation has been applied since the $|+\rangle$ state is not the ideal configuration to test the effects of a bit flip error. This is due to the fact that the $|+\rangle$ state is a superposition of state $|0\rangle$ and state $|1\rangle$ where both have the same coefficient ($\alpha = \beta = \frac{1}{\sqrt{2}}$) and consequently the same probability. Due to that, a bit flip error on a qubit in $|+\rangle$ state has, in the ideal condition, no effect, since the only effect would be swapping the two coefficients (α and β) but since these two coefficients are equal for the $|+\rangle$ state, the qubit's state after the application of an X gate is still $|+\rangle$. It was chosen then the $|0\rangle$ state since, as for the $|1\rangle$, is the state on which a bit flip error is more evident. The $|0\rangle$ state was preferred over the $|1\rangle$ state since the former is set by default automatically by Qiskit when a qubit is prepared. This last decision is not as trivial as it may seem since choosing the $|1\rangle$ state would have implied adding an additional X gate in the circuit that, when executed on the real quantum device, presents the possibility of an additional error.

The study on the real quantum computer is structured as on the simulator presented in Section 3.1. There will be three circuits:

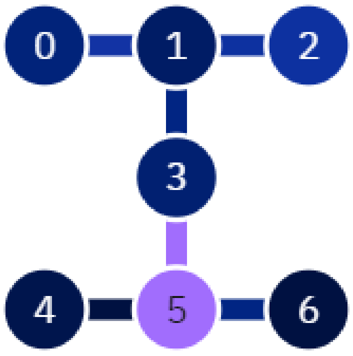
1) the basic circuit, which evaluates the qubit's state without any error correction or encoding process.
2) the encoding circuit, which assesses the effects of the encoding procedure on the state of the original qubit.
3) the complete circuit, which evaluates the validity of QEC implementations on protecting the qubit's state from bit flip errors.

The experimental outcomes of these three circuits are then compared reciprocally and with the ideal results provided by the theoretical formulation to assess the utility of QEC implementations.

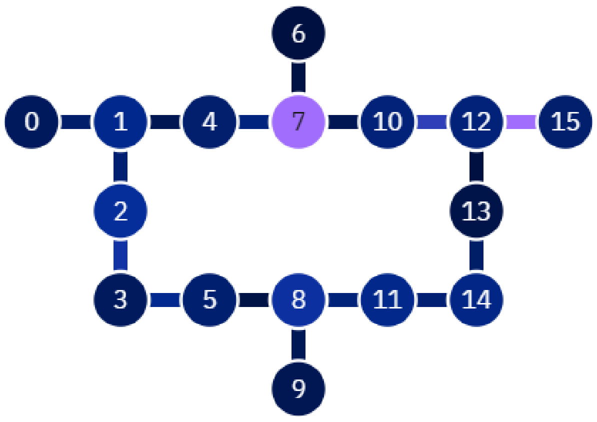
3.2.1 Choosing the quantum device

IBM, at the time of the draft of this thesis, offers to the public a free usage of 4 quantum devices, which run on a different processor, characterized by a *Family*, such as Eagle and Falcon, which refers to the chip architecture, and a *Revision* name, such as r3, r5.11H, which indicates design variants within a given family [16]:

- `ibm_brisbane`, which offers up to 127 qubits and runs an Eagle r3 processor.
- `ibm_perth`, which offers up to 7 qubits and runs a Falcon r5.11H processor.
- `ibm_lagos`, which offers up to 7 qubits and runs a Falcon r5.11H processor.
- `ibm_nairobi`, which offers up to 7 qubits and runs a Falcon r5.11H processor.



(a) ibm_nairobi



(b) ibmq_guadalupe

Figure 3.7: Physical disposition of the qubits in two IBM quantum devices [16].

As already said in the introduction to this section, the quantum device chosen is `ibm_nairobi`, and now the reasons for this decision are presented: to begin with, `ibm_brisbane` is not a suitable device for this task since it has largely more qubits than what is needed since the maximum number of qubits utilized in this project is 5, and due to `ibm_brisbane` being the only large quantum device provided by IBM for free, the queue to submit a job to the device causes computation times to be invalidating for the execution of this project.
The other three devices have all the same number of qubits and run on the same processor, therefore, choosing which one to utilize requires the introduction of two other parameters of these devices: the physical disposition of the qubits and, most importantly, the calibration data provided by IBM on any device.
Firstly, the physical disposition of the qubits is assessed. This is best by looking at Figure 3.7. Figure 3.7 shows the physical disposition of two IBM quantum devices: ibm_nairobi and ibmq_guadalupe (which is a 16-qubit device provided by IBM to the membership owners). These schematics have to be taken into account when choosing a quantum device for multiple reasons, the first of which is the fact that CNOT gates apply only between adjacent qubits; therefore, as an example, as can be understood by looking at the physical disposition of the qubits in ibm_nairobi shown in Figure 3.7, a CNOT gate cannot be applied between qubit 0 and qubit 3 of ibm_nairobi, since they are not adjacent.
However, all three 7-qubit devices provided for free by IBM have the same physical disposition of the qubits, as the one of ibm_nairobi shown in Figure 3.7. Hence, this feature cannot be discriminative to determine which quantum device to utilize. Here the second parameter previously introduced comes in handy: the calibration data provided by IBM on any real quantum device. This data is collected at various times each day in order

to monitor the performances of the quantum devices and to provide useful insights on their specifics. For conciseness reasons, the specifics of this data are not discussed in this thesis; however, if one is interested, a copy of the calibration data provided by IBM for ibm_nairobi, collected at the time of the redacting of this thesis, is available at my GitHub repository[17].
Anyway, if this calibration data is compared between the three remaining systems (ibm_perth, ibm_lagos, ibm_nairobi) three principal parameters are of interest for the scope of this project:

- **Readout assignment error** which is the probability of having an error on reading the state of a qubit.
- **Pauli X-gate error** which is the probability of an error occurring on the Pauli X-gate.
- **CNOT error** which is the probability of an error occurring on a CNOT gate.

The mean values of these three parameters (expressed as percentages) for the three remaining quantum devices are as follows:

Error type	ibm_nairobi	ibm_lagos	ibm_perth
Readout	$2.510e^{-2}$	$1.780e^{-2}$	$2.780e^{-2}$
X-gate	$2.565e^{-4}$	$2.565e^{-4}$	$3.379e^{-4}$
CNOT gate	$7.922e^{-3}$	$8.572e^{-3}$	$1.002e^{-2}$

Table 3.4: Calibration data for ibm_nairobi, ibm_lagos, ibm_perth in date 3/10/2023-11:32 [16].

As one can see in Table 3.4, ibm_perth has a higher error rate for all of the three error types and therefore is already crossed out as an eligible candidate. Next, one can see that ibm_nairobi has a lower CNOT gate error rate while ibm_lagos has a lower readout error rate. Due to this, ibm_nairobi was chosen as the quantum device since in the complete circuit there are 8 CNOT gates and it is therefore crucial to minimize the probability of that error type.

Now that the quantum device has been chosen, the three circuits implemented to run on the real quantum system are described, starting with the basic circuit.

3.2.2 Implementing the circuits on the real quantum device

The majority of the code necessary to implement these three circuits is similar or equal to what was already seen for the simulation in Section 3.1, consequently the description in this section will be limited to the differences with what was previously presented for the simulation.

Firstly, some new libraries have to be imported and some others are not necessary anymore. The libraries required are hence implemented through these command lines:

```
import qiskit
from qiskit import IBMQ # Necessary to get access to IBM Quantum devices.
import matplotlib.pyplot as plt
from qiskit import QuantumCircuit, QuantumRegister, ClassicalRegister,
↪  transpile, assemble, execute
from qiskit.visualization import plot_histogram
from qiskit_ibm_runtime import QiskitRuntimeService, Options # Required to
↪  run the real quantum device and to set some running options.
from qiskit_ibm_provider import IBMProvider # To run a circuit on an IBM
↪  device is necessary to retrieve the provider that grants access to IBM
↪  Quantum resources.
```

After this one has to load their personal IBM account and retrieve the provider, in order to be granted access to IBM devices:

```
# Load one's IBM Quantum account
IBMQ.load_account()

# Get the provider
provider = IBMQ.get_provider(hub='ibm-q', group='open', project='main')
```

The parameters of the method `IBMQ.get_provider()` are by default set as above. Lastly, the device has to be selected through the following code line:

```
# Select the 'ibm_nairobi' device
nairobi = provider.get_backend('ibm_nairobi')
```

The rest of the code is analog to what is shown in Section 3.1.
However, there is a major difference in this second implementation, which consists of the absence of the `for loop`. Instead of having the loop, the number of `shots` in a single instance is raised to 20000 and only one instance is run. This is forced by the fact that on a real quantum device, only one instance of a circuit can be submitted at the time, therefore is not possible to replicate what was done with the simulator and run 10000 instances. Nonetheless, the statistical validity of this operation is conserved since the data sample is large enough. Naturally, the data is not going to be plotted in a Gaussian distribution but it will be organized in a histogram.

3.2.3 Result and analysis

The results for the basic, encoding, and complete circuit run on the real quantum device ibm_nairobi are displayed in Tables 3.5, 3.6 and 3.7.

State	Expected value	p	σ
0	1.0000	0.9902	0.0002
1	0.0000	0.0098	0.0002

Table 3.5: Probability of each state and associated error for the basic circuit.

State	Expected value	p	σ
0	1.0000	0.9821	0.0002
1	0.0000	0.0179	0.0002

Table 3.6: Probability of each state and associated error for the encoding circuit.

State	Expected value	p	σ
0	1.0000	0.9988	0.0002
1	0.0000	0.0012	0.0002

Table 3.7: Probability of each state and associated error for the complete circuit.

The error σ on p is given by the readout error for ibm_nairobi calculated over the data sample of 20000. These results are then plotted on three histograms.

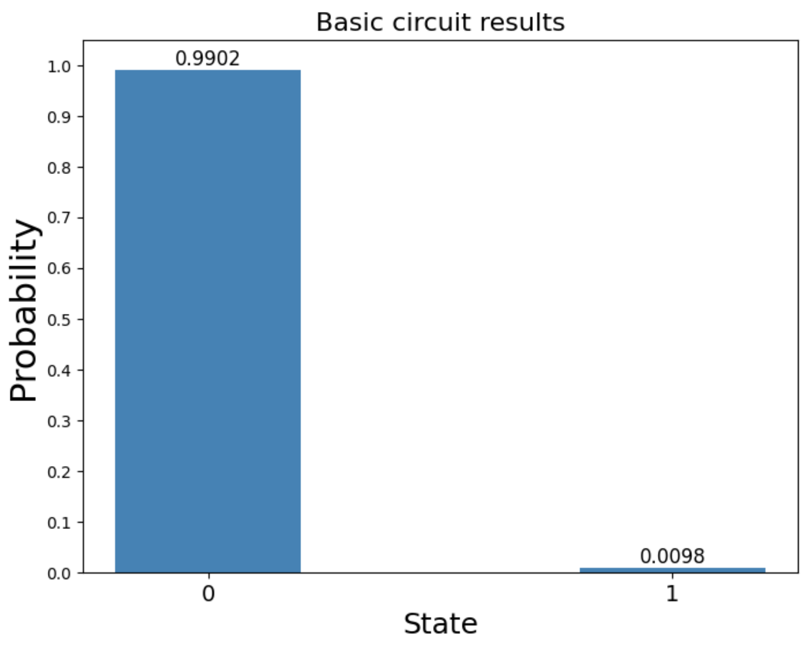


Figure 3.8: Basic circuit results on the real quantum device.

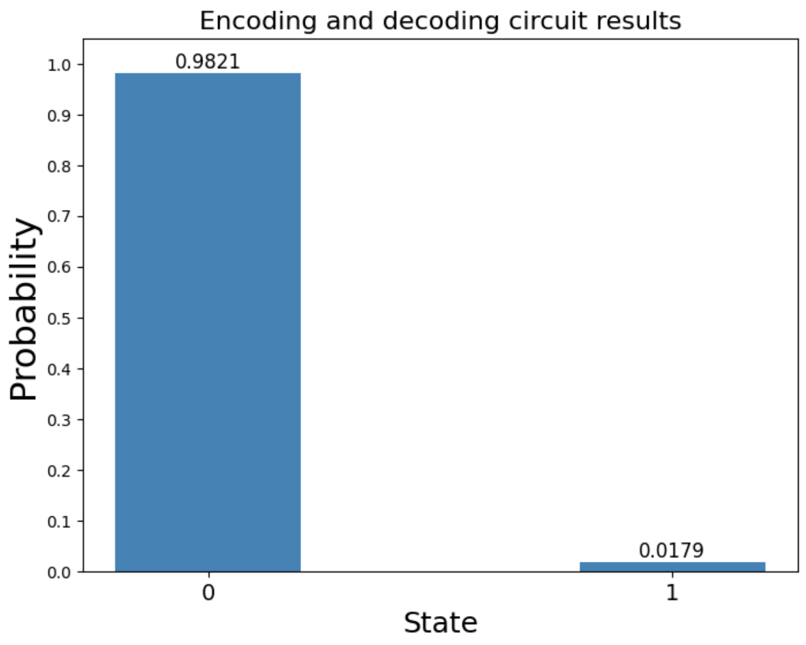


Figure 3.9: Encoding circuit results on the real quantum device.

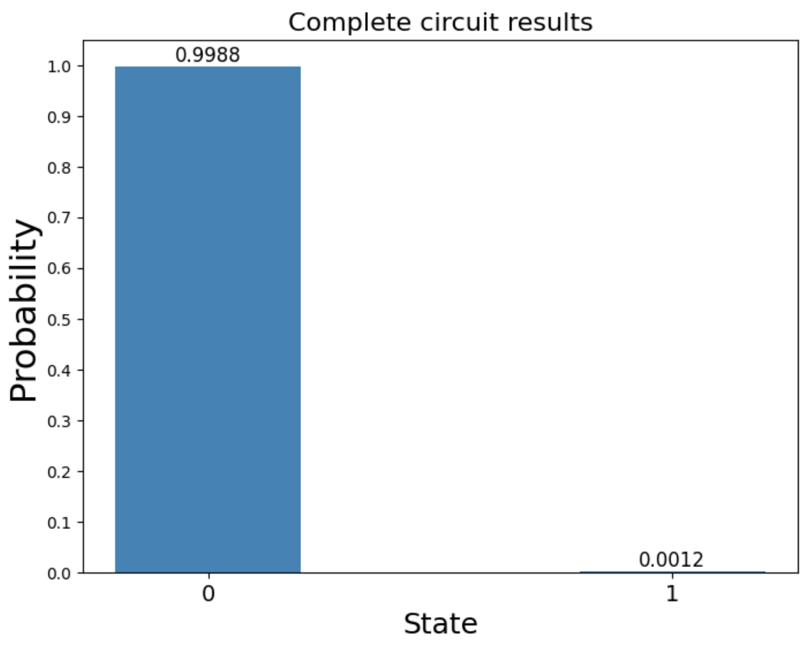


Figure 3.10: Complete circuit results on the real quantum device.

Now the results just shown in Tables 3.5, 3.6, 3.7 and Figures 3.8, 3.9, 3.10 are analyzed.

The first thing to notice is that, as seen from the results of the basic circuit on Table 3.5 and Figure 3.8, ibm_nairobi offers an already high accuracy level, with a bit flip error rate of 0.98% without the help of any QEC implementation. Next, one can notice from comparing the results for the encoding circuit in Table 3.6 and Figure 3.9 the fact that only encoding (and then decoding) with a three-qubit code a single-qubit system is not sufficient to provide protection from bit flip errors since for the encoding circuit the bit flip error rate is shown to be 1.79%, higher than the one for the basic circuit (0.98%). Lastly, the hypothesis of this project is confirmed by the bit flip error rate for the complete circuit, shown in Table 3.7 and 3.10. This error rate is 0.12%, gaining a 0.86% in accuracy over the basic circuit through the implementation of a three-qubit bit flip error correction code.

Conclusions

This thesis offers a basic description of quantum computing and quantum error correction (QEC) and its direct applicability to safeguarding quantum information. The central aim of this thesis has been to evaluate the tangible impact of QEC techniques, with a particular focus on mitigating bit flip errors, by employing the advanced tools offered by IBM Quantum.

First and foremost, it has been shown how the simulator offered by IBM Quantum (in this case, the QASM Simulator) is capable of perfectly recreating the results foreseen by the theoretical analysis. Indeed, it has been exhibited how the results of the analysis on the simulator depict the symmetric nature of the superposition state $|+\rangle$ and that the error returned is only a statistical error caused by the scarcity of the data sample. Positively, if one were to progressively increase the data sample capacity, the statistical error would progressively decrease and the Gaussian distribution that fits the data plot would progressively decrease in width, peaking on the theoretical value.

Furthermore, the empirical assessments conducted within this research have conclusively demonstrated that QEC, as realized through the instruments offered by IBM Quantum, possesses the potential to significantly enhance the accuracy and reliability of quantum computing systems. The outcomes of the analysis on the physical quantum device, IBM Nairobi, offer evidence that QEC methodologies effectively shield quantum information from the effects of quantum decoherence, in particular for bit flip errors. Indeed, through the implementation of a QEC procedure, in particular a three-qubit bit flip error correction code, on a single-qubit system in the state $|0\rangle$ positive results have been obtained. The measurements on the three studied circuits returned the following probabilities for the correct output: $p = (0.9988 \pm 0.0002)$ for the basic circuit, which is limited to measuring the state of the qubit, $p = (0.9821 \pm 0.002)$ for the encoding circuit and $p = (0.9988 \pm 0.0002)$ for the complete circuit, which includes the QEC implementation. The encoding circuit having the lowest probability of success is consistent with what is expected since in this circuit there are four additional CNOT gates in comparison with the basic circuit but no QEC procedure is applied and therefore the four additional gates only raise the error rate. Conversely, in the complete circuit, the error rate is found to be lowered to 0.12%, with a gain of 0.86% in accuracy over the 0.98% of the basic circuit only measuring the $|0\rangle$ state. Overall, these results show how a three-qubit bit flip error correction code is a valid tool to protect a single-qubit system from bit flip errors.

Looking ahead, the field of quantum computing presents exciting potentialities for being the main character in the next technological leap. The research for advanced QEC techniques that could shield from quantum decoherence is a promising field of study that could lead to great technological advances.

Bibliography

[1] David J. Griffiths and Darrell F. Schroeter. *Introduction to Quantum Mechanics.* Cambridge University Press, 3 edition, 2018.

[2] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, 2010.

[3] Qiskit Development Team. Qiskit 0.44 documentation. `https://qiskit.org/documentation/`, 2017-2023.

[4] Joschka Roffe. Quantum error correction: an introductory guide. *Contemporary Physics*, 60(3):226–245, jul 2019.

[5] A. Ekert, P. M. Hayden, and H. Inamori. Basic concepts in quantum computation. In *Les Houches - Ecole d'Ete de Physique Theorique*, pages 661–701. Springer Berlin Heidelberg, 2002.

[6] Richard Jozsa. Entanglement and quantum computation. *Geometric Issues in the Foundations of Science*, 1997.

[7] Yuying Guo. Introduction to quantum entanglement. *AIP Conference Proceedings*, 2066(1):020009, 01 2019.

[8] Huawei HiQ Revision. Quantum computing. `https://hiqsimulator.readthedocs.io/en/latest/quantumcomputing.html`, 2019.

[9] Yaoyun Shi. Both toffoli and controlled-not need little help to do universal quantum computation. *Quantum Information & Computation*, 3, 2002.

[10] W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, 10 1982.

[11] William P. Livingston, Machiel S. Blok, Emmanuel Flurin, Justin Dressel, Andrew N. Jordan, and Irfan Siddiqi. Experimental demonstration of continuous quantum error correction. *Nature Communications*, 13(1), apr 2022.

[12] Peter W. Shor. Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, 52:R2493–R2496, Oct 1995.

[13] Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation, 2009.

[14] M. A. Nielsen. The entanglement fidelity and quantum error correction, 1996.

[15] Takashi Imamichi. Quantum error correction. `https://github.com/qiskit-community/qiskit-community-tutorials/blob/master/awards/teach_me_quantum_2018/intro2qc/10.Quantum%20error%20correction.ipynb`, 2019.

[16] IBM Quantum Platform. Compute resources. `https://quantum-computing.ibm.com/services/resources`, 2019-2023.

[17] Pietro Malagoli. Quantum error correction. `https://github.com/pietromalagoli/Quantum-Error-Correction.git`, 2023.

Acknowledgements

To conclude this thesis I would like to express my gratitude to the people who helped me in the completion of this thesis. Since all of them are Italians, you will excuse me if I will continue in Italian.

In primo luogo ringrazio la Professoressa Ercolessi, poiché il suo corso e la sua passione che è riuscita a trasmettere a noi studenti, rendendoci partecipi ad ogni lezione, mi ha fatto scoprire un campo di studio che mi ha colpito profondamente e che continuerò ad approfondire nel mio percorso magistrale. La ringrazio inoltre per avermi accettato come suo laureando ed avermi dato l'ooportunità di lavorare ad un argomento così interessante.
Ringrazio poi Andrea, Callo e Giada per la compagnia durante le lezioni, il supporto in sessione di esame e i bei moment dentro e fuori la facoltà. Avete reso questi tre anni di università pieni di risate e affetto e sono molto contento di condividere con voi questi prossimi due anni a Padova.
Il mio prossimo ringraziamento va ad Arnaldo, la persona più genuinamente buona che io conosca. Sei stato un grande amico per me in questi tre anni e sono sicuro che abbiamo posto le basi per un'amicizia che durerà tutta la vita.
Ringrazio ora Alessandra, che mi è stata vicina in questi anni e in particolare nel mio periodo di Erasmus all'estero. Sei una persona su cui so che posso sempre contare e per me questo vuol dire tanto.
A questo punto voglio dire grazie a Giulio, un amico leale e sempre disponibile. Ti ringrazio per avermi seguito in un viaggio in Spagna all'ultimo minuto ed esserci stato fino alla fine nonostante le numerose peripezie. Ti ringrazio anche per i pranzi in balcone e per l'impegno che stai mettendo in questa amicizia. Grazie.
Vorrei poi ringraziare Riccardo, per le serate in collina dopo aver ordinato da Bolin. Sei un amico il cui affetto e attenzione all'altro mi fanno sentire molto fortunato di averti come amico.

Questo mio penultimo ringraziamento va alla Ceci. In questi due volte tre mesi ho trovato una persona che mi ha fatto ridere da star male, commuovere fino alle lacrime e che mi ha davvero reso una persona più felice. Prima di te non avrei mai pensato di potermi sentire così vicino a qualcun altro. Non so cosa il futuro riservi per me, ma spero davvero che tu ci sia.
Infine, ringrazio la mia famiglia. Ringrazio i nonni Gianni e Brunella per la loro generosità che mi ha sempre scaldato il cuore. Ringrazio la nonna Renza per avermi dato la possibilità di avere la mia autonomia, senza chiedere nulla in cambio. Ringrazio quindi mio fratello Giorgio per le continue risate di gusto e mia sorella Maya per avermi dimostrato sempre grande affetto. Ringrazio infine i miei genitori. Grazie a voi che mi avete trasmesso gli ideali che ora mi fanno la persona che sono e che se si vuole vedere un cambiamento nel mondo bisogna rimboccarsi le maniche. Grazie a voi di avermi insegnato la pace e la fratellanza. Grazie a voi di avermi insegnato a fare sempre la cosa giusta, anche se la più difficile e faticosa. Grazie di avermi sempre sostenuto e fatto sentire a casa.