POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Transient fault detectors for quantum circuits

Supervisors

Prof. Bartolomeo MONTRUCCHIO

Dr. Edoardo GIUSTO

Prof. Matteo SONZA REORDA

Prof. Paolo RECH

Candidate

Nicola DILILLO

October 2022

Summary

In the last decade, a new computing paradigm, which is gaining a lot of attention, has been proposed: Quantum Computing. Through quantum computing, it would be possible to execute a set of algorithms in several seconds, rather than hundreds or even thousands of years on classical computers.

The basic information in quantum computers, equivalent to a bit in classical ones, is the quantum bit, called qubit. While the bit can be 0 or 1, the qubit can be 1 and 0, according to a certain probability, and it can be in a superposition of both states, at the same time. This state can be described through the usage of the Bloch sphere, as shown in figure 1.

Quantum algorithms are described using quantum circuits. It is possible to act on a qubit in three ways: applying a gate, measuring the qubit, or resetting it. All these elements can be visualized as blocks on a staff, where each line represents a qubit, following the execution order.

For the probabilistic nature of quantum computation, executions on quantum devices are repeated more than once, associating each obtained output with a probability. In the end, the one which the highest probability will be picked as the actual result.

Real quantum computers, that consist of qubits and gates that are not perfect, are placed in a laboratory, in the presence of noise. Quantum errors can be uncorrelated or correlated. Uncorrelated errors happen independently from time and space and they can be suppressed using techniques of quantum error correction, QEC, based on repetition code, where more than one physical qubits is used to generate one logical qubit.
While the uncorrelated errors can be exponentially suppressed with the scaling devices, the correlated ones affect almost all the qubits simultaneously, causing logical faults. On superconducting qubits, based on silicon and widely used today, cosmic rays are one of the main causes of this problem. They are heavy particles that impact the coherence of the superconducting qubits, generating quasiparticles, which may cause the collapse of the qubit or a transient shift of the state.

Google researchers made a very interesting experiment, using superconducting qubits of a real quantum processor, demonstrating how difficult is to deal with this

type of error. A simple quantum circuit has been built to initially set all the qubits to an excited state, $|1\rangle$, and then to measure them, re-executing this operation periodically. Instead of measuring all 1s, as expected, they measured always some 0s, because of uncorrelated errors. At a certain point, all qubits in the device have been measured as 0s for a while. This behaviour has been caused by correlated errors and it has lasted for around 25 ms, generating a transient fault. With all qubits affected is impossible to apply currently existing QEC techniques!
In this thesis, this particular behaviour of quantum transient faults, which tended to collapse all the qubits to ground state, $|0\rangle$.

To detect the presence of these transient faults, a specific circuitry, called *Transient Fault Detector* ??TFD), has been designed. These ??s are put at the beginning and at the end of each algorithm circuit, wrapping it. The basic idea consists in creating a circuit that measures always the same output. If the measured values are different with respect to the expected ones, it is probable that a correlated error has arisen in the circuit.

Different kinds of wrappers have been tested with different results. The purpose of each of them is to move along all the possible positions inside the Bloch sphere (figure 1.2), which represent all the possible superpositions of the qubit, trying to catch an unwanted shift state, highlighted by the ??s. In order to do this, following the experiment of Google's team, the qubit starts from state $|0\rangle$ to arrive at $|1\rangle$, trying to cover the whole surface of the sphere. If an error is detected, the final result measured at the barriers will be no more $|1\rangle$, but $|0\rangle$.

To better understand how these errors can affect the circuit and the efficiency of ??s, transient faults have been simulated by software, using the QuFI (Quantum Fault Injector) framework. Through QuFI it has been possible to tune the phase shift magnitude based on the proximity of the qubit to the particle strike location, to see how the fault is propagated throughout the circuit.

The effect of transient faults is evaluated using a metric called Quantum Vulnerability Factor, QVF. It is based on Michelson Contrast, which gives a measure of distinguishable between two objects that belong to the same family. In this case, QVF has been used to see the differences between a faulty-free system and one with an injected fault. The minimum value of QVF is 0, which means that, nevertheless the injected fault, the correct result has still the same probability to be measured, while the highest value of QVF is 1, which means that the injected fault makes the probability of observing the correct result equal to 0.

Because the quantum outputs are probabilistic, the values of the measured ??s are probabilistic as well. A threshold percentage is set to distinguish between detected faults and not detected ones.
The faults marked as not detected by barriers could belong to one of these error classes.

- Undetectable, with a QVF between 0 and 0.3.

- Silent corruption, with a QVF between 0.3 and 0.7.
- Untriggered, with a QVF between 0.7 and 1.

Instead, the faults marked as detected by barriers could belong to one of these error classes.

- False positive, with a QVF between 0 and 0.3.
- Detectable 0, with a QVF between 0.3 and 0.7.
- Detectable 1, with a QVF between 0.7 and 1.

QuFI has been used to evaluate two types of wrappers, using several quantum circuits.
A first type of wrapper has been tested using QuFI, where each of the injected faults consists of a shift of the qubit state. It is more suitable for detecting errors described using a more general representation, like angle combinations.
A second type of wrapper has been tested using QuFI, where each of the injected faults consists of a reset gate. It is more suitable for detecting errors described using Google's model, where in case of fault the qubit state collapsed to the ground.

For each quantum circuit, the cardinality of all error classes has been evaluated, using different threshold percentages. On average, a good portion of injected faults has been detected, although there is a certain amount of false positives, while the number of silent and untriggered faults is pretty low. In table 1 there are the most significant threshold percentages set for the ??s, assigned to test a wrapped quantum circuit, and their respective percentage of error classes.

According to these results, it is possible to understand that these ??s, which add some extra gates useless for elaboration, could be a good tool to improve the reliability of quantum computers.
Different ??s have been proposed and others could be designed to better cover different types of errors. Actually, there are no valid methods used to efficiently detect if transient faults occur in quantum circuits and this seems to be one of few methods, if not the only one, able to do so.

%	Undetectable	Silent	Untriggered	False	Detectable 0	Detectable 1
20%	48.2%	19.6%	7.3%	14.0%	5.2%	5.7%
25%	44.4%	17.4%	5.3%	17.8%	7.4%	7.7%
30%	42.5%	16.0%	4.9%	19.7%	8.8%	8.1%
35%	38.8%	12.4%	4.3%	23.4%	12.4%	8.7%
40%	36.8%	10.2%	4.1%	25.4%	14.6%	8.9%
45%	31.9%	5.4%	3.5%	30.3%	19.4%	9.4%
50%	29.8%	2.9%	3.3%	32.4%	21.9%	9.7%
55%	26.4%	0.6%	2.7%	35.8%	24.2%	10.3%
60%	24.0%	0.1%	2.3%	38.2%	24.7%	10.7%
65%	19.4%	0.0%	1.7%	42.8%	24.8%	11.3%
70%	16.3%	0.0%	1.2%	45.9%	24.8%	11.8%
75%	12.9%	0.0%	0.7%	49.3%	24.8%	12.2%
80%	8.7%	0.0%	0.3%	53.5%	24.8%	12.7%
85%	0.1%	0.0%	0.0%	62.1%	24.8%	13.0%
90%	0.0%	0.0%	0.0%	62.2%	24.8%	13.0%

Table 1: The main percentages of error classes, obtained by testing a circuit wrapped with the first type of Detector Barrier.

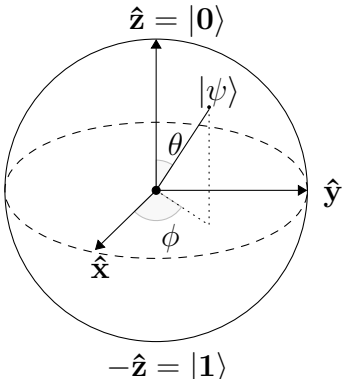

Figure 1: Bloch sphere visualization of a qubit state $|\Psi\rangle$.

Acknowledgements

Ringrazio il Prof. Bartolomeo Montrucchio, correlatore della tesi, e tutto il gruppo di ricerca, il Dr. Edoardo Giusto, il Prof. Paolo Rech e il Dr. Emanuele Dri, per avermi guidato nella realizzazione di ogni capitolo.

Ringrazio i miei genitori, Anna e Giuseppe, che in questi anni non mi hanno fatto mancare nulla e che hanno sempre creduto in me. Siete la mia forza.

Ringrazio i miei fratelli, Davide e Gianluca, che nonostante la mia assenza fisica negli ultimi anni, sono cresciuti nel migliore dei modi. Sono fiero di voi.

Ringrazio la mia ragazza, Fiorangela, che mi ha sempre sostenuto nelle mie scelte, decidendo di intraprendere una nuova vita con me. Ti amo.

Ringrazio tutti i miei cugini, zii e nonni per avermi sempre fatto sentire ben accolto ogni volta che tornavo a casa. Per avermi fatto capire che in ogni momento no, avevo un posto sicuro dove andare.

Ringrazio il mio coinquilino, Salvatore, con cui ho condiviso gioie e dolori durante la nostra avventura, iniziata cinque anni fa. Senza di te non sarebbe stato lo stesso.

Ringrazio i miei amici di sempre che, nonostante la distanza, mi sono rimasti accanto, regalandomi indimenticabili momenti di spensieratezza. Vi voglio bene.

Ringrazio tutti i miei amici conosciuti a Torino, per essere sempre stati presenti negli ultimi anni. Grazie a voi ho sentito di far parte di una seconda famiglia.

Nicola Dilillo

Table of Contents

List of Tables

List of Figures

Acronyms

NISQ
Noisy Intermediate Scale Quantum

QC
Quantum Computing

QEC
Quantum Error Correction

cQED
Circuit Quantum Electrondynamics

QVF
Quantum Vulnerability Factor

QuFI
Quantum Fault Injector

Qubit
Quantum binary digit

TMR
Triple Modular Redundancy

DWC
Duplication With Comparison

TFD
Transient Fault Detector

Chapter 1

Introduction to Quantum Computing

In the last years, a new computing paradigm has been proposed, which is gaining a lot of attention: Quantum Computing. Through quantum programming, it is possible to speed up a set of algorithms that on classical computers would require years to be performed.

1.1 Computers history

The history of computers starts two hundred years ago. The first computers were based on mechanical components that allowed the execution of arithmetic operations through the usage of gears and levers. Mechanical computers started to be used during War World II, for military applications, and one of the most famous machines was the Enigma, a cipher device used for encryption or decryption messages by Nazi Germany [1].

In '70, mechanical computers were replaced by electrical ones, based on vacuum tubes. The most famous was ENIAC (Electronic Numerical Integrator and Computer), completed in 1945, it was the primary programmable, electronic, all-purpose computing device. It was expensive, it costed 400.000$, it occupied an entire room, and it generated 174 kilowatts of heat. Besides ENIAC, all other computers that supported this technology were massive and consumed a huge amount of power [2].

Nowadays, the fundamental component of each modern electronic device is the transistor, a digital switch that is exploited to build advanced operations. Compared with vacuum tubes they are smaller and use considerably less power to work, generating less heat to dissipate [3]. A particular class of transistors, called MOS transistors, uses even less power and, for this reason, they're used

in large-scale production. Thanks to MOS, which are so small to be invisible to human eyes, it is possible to create processors consisting of millions or even billions of transistors.

Over the years, transistors' dimension shrank, increasing speed and reducing power dissipation. This evolution has been predicted by Moore's Law, enunciated by Gordon Moore in 1965, which observed that the number of transistors on a chip doubles every 2 years. Moore's Law has driven the semiconductor factories for almost 50 years, as a reference for targeting technology to study and develop. The graph in figure 1.1 demonstrates that Moore's Law wasn't wrong.

Experts say that Moore's law will reach an end within a few years. The length of the last transistors based on silicon is 5 nm that, in the coming years, can become 2 nm, approaching more and more the size of a silicon atom, which is 0.2 nm. New materials, such as graphene, can be used to design transistors, being able to push their dimension below 1 nm [4, 5, 6].

Transistors can't shrink any further, going under the size of the atom. To solve this problem, different programming paradigms have been proposed.
Over the years one of the most successful is Quantum Computing.

1.2 Quantum Computers

The quantum computing paradigm is based on quantum mechanics, which is the science that takes care of the physical properties of atomic and subatomic particles, boned to explain phenomena that don't follow the rule of classical physics.

In the beginning, quantum systems ran on classical computers, using quantum simulators. In 1982, Richard Feynman and Yuri Manin at MIT affirmed that classical computers are not able to simulate quantum systems efficiently and to solve this problem they proposed Quantum Computers, able to simulate quantum systems in a faster way. Unluckily, this proposal did not generate much interest in quantum computing. Only in the '90, after 10 years, quantum programming has begun to be taken into consideration thanks to Shor's algorithm.

Peter Shor developed an efficient quantum algorithm able to factorize large integer numbers with exponential speedup with respect to classical computers [8]. The slowness of factorization is the key that guarantees the security of asymmetric cryptography algorithms. On classical computers, this operation will require billions of years while quantum computers can do that in a couple of hours. This made quantum computers fascinating and put them under the attention of many big companies.

The basic information in quantum computers, equivalent to a bit in classical computers, is the quantum bit, called qubit. While the bit can be 0 or 1, the qubit can be 1 and 0, according to a certain probability, and it can be in a superposition

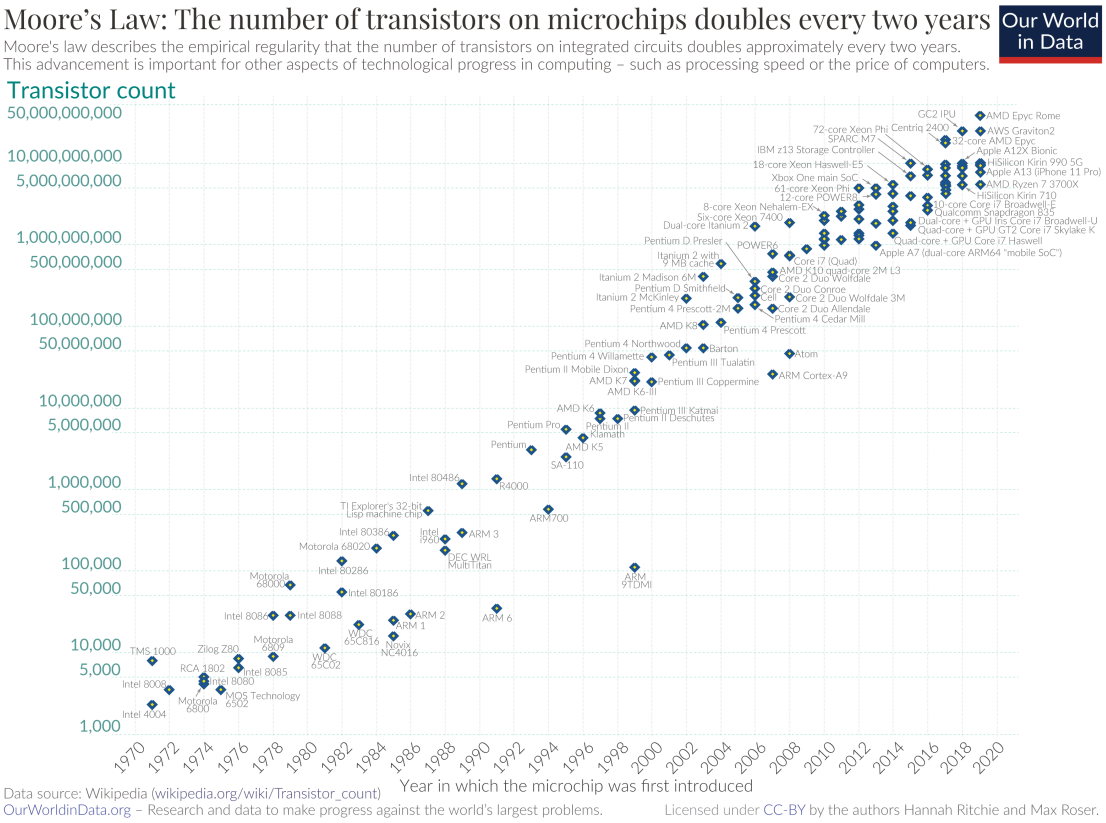

Figure 1.1: A logarithmic graph showing the timeline of how transistor counts in microchips almost doubled every two years from 1970 to 2020; Moore's Law [7]

of both states, at the same time. This phenomenon is called superposition.

Another important property is the entanglement, which describes a strong interaction between two or more qubits. Qubits in entanglement can influence each other. Superposition and entanglement are the two properties which allow quantum computers to create algorithms with an exponential speed up.

Quantum algorithms are described using quantum circuits [9]. It is possible to act on a qubit in three ways: applying a gate, measuring the qubit, or resetting it. All these elements can be visualized as blocks on a staff, where each line represents a qubit, following the execution order.

For the probabilistic nature of quantum computation, executions on quantum devices are repeated more than once, associating each obtained output with a probability. In the end, the one which the highest probability will be picked as the actual result.

The first quantum computers has been built in 1998, with only two qubits, and it was able to solve only a simple algorithm called Grover's algorithm. Nowadays

much more qubits are available on quantum devices and much more companies are involved in researching this field. The main ones are IBM, Google, D-Wave and Amazon. Actually, the biggest quantum computer has almost 500 qubits, but it's just a prototype, used only for research. IBM, probably the more involved in this topic, has announced that in 2026 it will be able to produce quantum computers with a number of qubits that goes from 10 thousand to 100 thousand.
Some fields where quantum computers could be revolutionary are:

- Artificial Intelligence and Machine Learning
- Computational Chemistry
- Drug Design & Development
- Financial Modelling
- Logistics Optimisation
- Weather Forecasting

1.3 From bit to qubit

Quantum information theory is the science that studies the state of a quantum system. As well as in classical computers the basic unit is 0 and 1, in quantum computing two pure states are defined, that generally are $|0\rangle$ (ket-zero) and $|1\rangle$ (ket-one), representing vectors and shown in 1.1. These basis states are used to represent the quantum information.

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

The qubit is able to handle these two states simultaneously, in *superposition*. Two states in superposition are described by a linear combination of them, as shown in 1.2. Through this property, operations can be executed much faster because it's possible to have different states at the same time.

$$|\Psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{1.2}$$

For example, three bits in classical computers can be used to represent 8 numbers, from 0 to 7. Instead, three quantum bits can represent 8 quantum states at the same moment.

As convection, any quantum state has to be normalized, so that its inner product must be equal to 1, as sown in 1.3. For instance, in the case of equal superposition ($\alpha = \beta$), it's needed to normalize like in formula 1.4.

$$\langle\Psi|\Psi\rangle = 1 \tag{1.3}$$

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \tag{1.4}$$

Different mathematical methods can be employed to describe quantum states.

1.3.1 Dirac notation

Let's remember that the conjugate of a complex number $a \in \mathbb{C}$, indicated as a^*, is obtained by inverting the sign of the imaginary. If $a = x + yi$, then $a^* = x - yi$. Let's be a and b two complex numbers; $a, b \in \mathbb{C}^2$:

- ket is $|a\rangle = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix}$
- bra is $\langle b| = |b\rangle^\dagger = \begin{pmatrix} b_0 \\ b_1 \end{pmatrix}^\dagger = \begin{pmatrix} b_0^* & b_1^* \end{pmatrix}$
- bra-ket: $\langle b|a\rangle = a_0 b_0^* + a_1 b_1^* = \langle a\rangle\, b^*$
- ket-bra: $|a\rangle\langle b| = \begin{pmatrix} a_0 b_0^* & a_0 b_1^* \\ a_1 b_0^* & a_1 b_1^* \end{pmatrix}$

This method of representation is called Dirac Notation. The ket-bra resulting from the combinations of all basis states are the following:

- $|0\rangle\langle 0| = \begin{pmatrix} 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$
- $|0\rangle\langle 1| = \begin{pmatrix} 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$
- $|1\rangle\langle 0| = \begin{pmatrix} 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}$
- $|1\rangle\langle 1| = \begin{pmatrix} 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$

In this way, any matrix can be described through the Dirac notation, as demonstrated in 1.5.

$$\rho = \begin{pmatrix} \rho_{00} & \rho_{01} \\ \rho_{10} & \rho_{11} \end{pmatrix} = \rho_{00} \cdot |0\rangle\langle 0| + \rho_{01} \cdot |0\rangle\langle 1| + \rho_{10} \cdot |1\rangle\langle 0| + \rho_{11} \cdot |1\rangle\langle 1| \tag{1.5}$$

1.3.2 Density matrices

Let's remember that a trace of a matrix A, usually defined as $tr(A)$, can be defined only for a square matrix and it is defined as the sum of all elements in the main diagonal, as shown in 1.6.

$$A = \begin{pmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{pmatrix} \quad \rightarrow \quad tr(A) = a_{00} + a_{11} + a_{22} \tag{1.6}$$

All quantum states, in any superposition, can be described by density matrices, usually identify as ρ, with the following properties:

- normalized, which means that the trace of matrices is 1.

$$tr(\rho) = \rho_{00} + \rho_{11} = 1 \tag{1.7}$$

- almost positivity, it means that sandwich a density matrices between any arbitrary vector Ψ, the result will be greater or equal to zero. This is equal to having all eigenvalues greater or equal to zero.

$$\langle \Psi | \rho | \Psi \rangle \geqslant 0, \quad \forall \, | \Psi \rangle \tag{1.8}$$

- Hermitian operators, it means that the transpose of the conjugate matrices, ρ^*, is equal to ρ. This means that ρ_{00} and ρ_{11} need to be real.

$$\rho^\dagger = \begin{pmatrix} \rho_{00}^* & \rho_{10}^* \\ \rho_{01}^* & \rho_{11}^* \end{pmatrix} = \rho \tag{1.9}$$

Every density matrix ρ has a spectral decomposition such that, defining orthogonal basis $|i\rangle$, it is possible to define the matrix ρ as $\sum_i \lambda_i \, |i\rangle \, \langle i|$, where $|i\rangle$ are the eigenstates and λ_i are the eigenvalues.
The sum of all eigenvalues is equal to 1, $\sum_i \lambda_i = 1$.

A matrix is pure if can be written as $\rho = \langle \Psi | \, | \Psi \rangle$, otherwise it is called mixed. Knowing that is always possible to find a decomposition, where the sum of all eigenvalues is equal to 1, if ρ is pure, there will be one eigenvalue equal to 1 and all the others equal to zero. This means that the trays of the squared matrix is equal to one, $tr(\rho^2) = \sum_i \lambda_i^2 = 1$. In case of mixed matrix, it will be lower than 1, $tr(\rho^2) < 1$.
Pure matrix is associated to pure state and likewise for mixed state. Look at the following examples which use as basis states $|0\rangle$ and $|1\rangle$.

- they are pure states because both have one eigenvalue equal to 1 and all the others equal to 0.

$$\rho = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = |0\rangle\langle 0| \qquad \rho = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = |1\rangle\langle 1| \tag{1.10}$$

- in this case the trace matrix is 1. Through the spectrum decomposition is possible to see that there are two eigenvalues equal to $\frac{1}{2}$. The sum of their square value is smaller then 1 and so this is a mixed state.

$$\rho = \frac{1}{2}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \frac{1}{2}(|0\rangle\langle 0| + |1\rangle\langle 1|) \tag{1.11}$$

- it' possible to notice that the 1.12 can be rewritten as 1.13, where is evident the form of a pure state, $\langle\Psi|\Psi\rangle$.

$$\rho = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \frac{1}{2}(|0\rangle\langle 0| - |0\rangle\langle 1| - |1\rangle\langle 0| + |1\rangle\langle 1|) \tag{1.12}$$

$$\rho = \frac{1}{2}\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} = \frac{1}{\sqrt{2}}(\langle 0| - \langle 1|) \cdot (|0\rangle - |1\rangle)\frac{1}{\sqrt{2}} = \langle\Psi|\Psi\rangle \tag{1.13}$$

1.4 Measurements

The measurements are state projections on the bases. To measure a quantum state, orthogonal bases are chosen, which usually are $|0\rangle$ and $|1\rangle$, orthogonal to each other, as demonstrated in 1.14.

Choosing orthogonal bases means that, if the qubit state is $|\Psi\rangle$, then it is impossible to measure $|\Phi\rangle$, and vice versa. In this way, there is no ambiguity in the result.

$$\langle 0|1\rangle = 1 \cdot 0 + 0 \cdot 1 = 0 \tag{1.14}$$

To the project of the qubit state on basis state can be done with the σ_z matrix, defines as 1.15. This is called Z-measurement. The final result could be either 1 or -1, which correspond respectively to $|0\rangle$ (0) and $|1\rangle$ (1).

$$\sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{1.15}$$

Infinite different bases exist, but only some of them are commonly used, such as 1.16 and 1.17, which correspond respectively to σ_x, X-measurement, and σ_y, Y-measurement.

$$\{|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \qquad |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\} \tag{1.16}$$

$$\{|+i\rangle = \frac{1}{\sqrt{2}}(|0\rangle + i\,|1\rangle), \qquad |-i\rangle = \frac{1}{\sqrt{2}}(|0\rangle - i\,|1\rangle)\} \tag{1.17}$$

1.4.1 Born rule

When a measurement is executed the qubit state collapses. The Born rule affirms that, given the state $|\Psi\rangle$ and the state basis $\{|x\rangle, |x^{\perp}\rangle\}$, orthogonal to each other, during measurements, the probability that it collapses onto the state $|x\rangle$ is given by the formula 1.18.

$$P(x_i) = |\langle x_i|\Psi\rangle|^2, \qquad \sum_i P(x_i) = 1 \tag{1.18}$$

Let's make an example. If the state $|\Psi\rangle$, in 1.19 must be measured in the basis $\{|0\rangle, |1\rangle\}$, then the probability of $|0\rangle$ will be 1.20, while the probability of $|1\rangle$ will be 1.21. As expected, the sum of the two probabilities is 1.

$$|\Psi\rangle = \frac{1}{\sqrt{3}}(|0\rangle + \sqrt{2}\,|1\rangle) \tag{1.19}$$

$$P(0) = |\langle 0|\frac{1}{\sqrt{3}}(|0\rangle + \sqrt{2}\,|1\rangle)\rangle|^2 = |\frac{1}{\sqrt{3}}\langle 0|0\rangle + \frac{\sqrt{2}}{\sqrt{3}}\langle 0|1\rangle|^2 = \frac{1}{3} \tag{1.20}$$

$$P(1) = |\langle 1|\frac{1}{\sqrt{3}}(|0\rangle + \sqrt{2}\,|1\rangle)\rangle|^2 = |\frac{1}{\sqrt{3}}\langle 1|0\rangle + \frac{\sqrt{2}}{\sqrt{3}}\langle 1|1\rangle|^2 = \frac{2}{3} \tag{1.21}$$

Now let's measure the state $|\Psi\rangle$, in 1.22 the basis $\{|+\rangle, |-\rangle\}$, then the probability of $|+\rangle$ will be 1.23, while the probability of $|-\rangle$ will be 1.24. The $|\Psi\rangle$ is equal to $|-\rangle$ and so the final result will be respectively 0 and 1.

$$|\Psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{1.22}$$

$$\begin{aligned} P(+) = |\langle +|\Psi\rangle|^2 = |\frac{1}{\sqrt{2}}(\langle 0| + \langle 1|) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)|^2 = \\ \frac{1}{4}|\langle 0|0\rangle - \langle 0|1\rangle + \langle 1|0\rangle - \langle 1|1\rangle|^2 = 0 \end{aligned} \tag{1.23}$$

$$\begin{aligned} P(-) = |\langle -|\Psi\rangle|^2 = |\frac{1}{\sqrt{2}}(\langle 0| - \langle 1|) \cdot \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)|^2 = \\ \frac{1}{4}|\langle 0|0\rangle - \langle 0|1\rangle - \langle 1|0\rangle + \langle 1|1\rangle|^2 = 1 \end{aligned} \tag{1.24}$$

1.5 Bloch sphere

Any pure normalized state can be represented through the formula 1.25, where $\varphi \in [0, 2\pi)$ is the relative phase, which describes the relative phase between the two states, and $\theta \in [0, \pi]$, which determines the probability to measure 0 or 1, according to the formula 1.26.

$$|\Psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\varphi}\sin\frac{\theta}{2}|1\rangle \tag{1.25}$$

$$P(0) = \cos^2\frac{\theta}{2}, \qquad P(1) = \sin^2\frac{\theta}{2} \tag{1.26}$$

In this case, using only two variables, it is possible to show the qubit state on the surface of a sphere, with a radius equal to 1, $|\vec{r} = 1|$, due to normalization. It's called the Bloch sphere, and it is shown in figure 1.2.

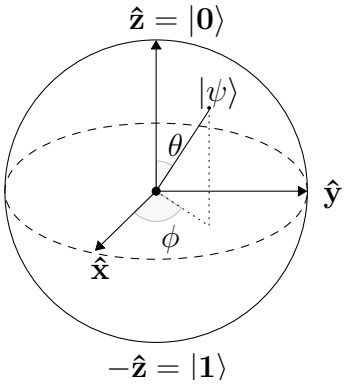

Figure 1.2: Bloch sphere visualization of a qubit state $|\Psi\rangle$.

The coordinates are given by the block vector, and typical are spherical coordinates, described as 1.27.

$$\vec{r} = \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix} \tag{1.27}$$

Some examples to better understand the comparison between the matrix representation and the Bloch sphere.

- $|0\rangle$ has $\theta = 0$ and φ arbitrary. The block vector become 1.28.

$$\vec{r} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{1.28}$$

- $|1\rangle$ has $\theta = \phi$ and φ arbitrary. The block vector become 1.29.

$$\vec{r} = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \tag{1.29}$$

- $|+\rangle$ has $\theta = \frac{\phi}{2}$ and $\varphi = 0$. The block vector become 1.30.

$$\vec{r} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \tag{1.30}$$

- $|-\rangle$ has $\theta = \frac{\phi}{2}$ and $\varphi = \phi$. The block vector become 1.31..

$$\vec{r} = \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \tag{1.31}$$

- $|+i\rangle$ has $\theta = \frac{\phi}{2}$ and $\varphi = \frac{\phi}{2}$. The block vector become 1.32..

$$\vec{r} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \tag{1.32}$$

- $|-i\rangle$ has $\theta = \frac{\phi}{2}$ and $\varphi = \frac{3\phi}{2}$. The block vector become 1.33.

$$\vec{r} = \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix} \tag{1.33}$$

On the Bloch sphere, angles are twice then their representation in Hilbert space, the mathematical environment used to describe these vectors. For example, the $|0\rangle$ and $|1\rangle$ are orthogonal but, on the Bloch sphere, their angles are 180°, both represented on the same axis. So θ is the angle in the Bloch sphere, while $\frac{\theta}{2}$ is the actual angle in the Hilbert space.

Through this interpretation of the space is also clear that a Z-measurement corresponds to a projection on the z-axis. The same thing is valid for X and Y measurements.

1.6 Gates

In quantum circuit are present a sequence of building blocks, called gates, which are employed in elementary operations. Gates can act on single or multiple qubits.

In linear algebra, especially in quantum mechanics, a complex square matrix, U, is unitary if the product with its Hermitian, $U^\dagger$, it's equal to the identity matrix, $\mathbb{1}$, as in 1.34. Since quantum theory is unitary, also quantum gates are based on unitary matrices.

$$U^\dagger U = UU^\dagger = \mathbb{1} \tag{1.34}$$

1.6.1 Pauli-X

It's the simplest gate, described by the following matrix 1.35.

$$\sigma_X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = |0\rangle\langle 1| + |1\rangle\langle 0| \tag{1.35}$$

As shown in 1.36 and 1.37, this gate executes a bit flip. On the Bloch sphere, this computation is translated as a rotation of 180° around the x-axis.

$$\sigma_X |0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \tag{1.36}$$

$$\sigma_X |1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle \tag{1.37}$$

On the Bloch sphere, this computation is translated as a rotation of 180° around the x-axis.

1.6.2 Pauli-Z

This gate is described by the matrix 1.15, shown previously in the chapter 1.4.

As shown in 1.38 and 1.39 this gate executes a phase flip. On the Bloch sphere, this computation is translated as a rotation of 180° around the z-axis.

$$\sigma_z |+\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = |-\rangle \tag{1.38}$$

$$\sigma_z |-\rangle = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = |+\rangle \tag{1.39}$$

1.6.3 Pauli-Y

This gate is described by the following matrix 1.40.

$$\sigma_y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} = i \cdot \sigma_x \sigma_z \tag{1.40}$$

As reported in 1.41 and 1.42 this gate executes a phase flip and a bit flip. On the Bloch sphere, this computation is translated as a rotation of 180° around the y-axis.

$$\sigma_y |i\rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix} = |-i\rangle \tag{1.41}$$

$$\sigma_z |-i\rangle = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} \cdot \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ -i \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix} = |i\rangle \tag{1.42}$$

1.6.4 Pauli matrices

All the matrices shown above are called Pauli matrices. The square of each of them is equal to the identity matrix, as shown in 1.43. If identity is multiplied by another matrix, then nothing happens, $A\mathbb{1} = \mathbb{1}A = A$.
This means that, if all these Pauli matrices are applied twice, on the same qubit line, this will correspond to applying the identity matrix, which will be the same as not performing operations. Indeed, a rotation of 360° around the Bloch sphere will be executed, which will take the qubit state back to the original position.

$$\sigma_i^2 = \mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{1.43}$$

Pauli matrices and identity together form a basis of 2×2 matrices. This means that any qubit rotation can be written as a combination of them.

1.6.5 Hadamard gate

This is the most famous and important gate and it is used to the creation of superposition in a qubit. It's described by the matrix in 1.44.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = |0\rangle\langle 0| + |0\rangle\langle 1| + |1\rangle\langle 0| - |1\rangle\langle 1| \tag{1.44}$$

Applying Hadamard gate to state $|0\rangle$ or to state $|1\rangle$, superposition is generated, according to 1.45 and 1.46.

$$H|0\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\cdot\begin{pmatrix}1\\ 0\end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\ 1\end{pmatrix} = |+\rangle \tag{1.45}$$

$$H|1\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\cdot\begin{pmatrix}0\\ 1\end{pmatrix} = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\ -1\end{pmatrix} = |-\rangle \tag{1.46}$$

If the gate is used again to $|+\rangle$ and to $|-\rangle$, they will come to the original state, as demonstrated in 1.47 and 1.48.

$$H|+\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\cdot\frac{1}{\sqrt{2}}\begin{pmatrix}1\\ 1\end{pmatrix} = \begin{pmatrix}1\\ 0\end{pmatrix} = |0\rangle \tag{1.47}$$

$$H|-\rangle = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\cdot\frac{1}{\sqrt{2}}\begin{pmatrix}1\\ -1\end{pmatrix} = \begin{pmatrix}0\\ 1\end{pmatrix} = |1\rangle \tag{1.48}$$

1.6.6 S gate

S gate adds 90° to the phase φ, and the matrix associated is the 1.49. S gate is used to change from Z to Y axis, as shown 1.50 and 1.51.

$$S = \begin{pmatrix}1 & 0\\ 0 & i\end{pmatrix} \tag{1.49}$$

$$S|+\rangle = \begin{pmatrix}1 & 0\\ 0 & i\end{pmatrix}\cdot\begin{pmatrix}1\\ 1\end{pmatrix} = \begin{pmatrix}1\\ i\end{pmatrix} = |i\rangle \tag{1.50}$$

$$S|-\rangle = \begin{pmatrix}1 & 0\\ 0 & i\end{pmatrix}\cdot\begin{pmatrix}1\\ -1\end{pmatrix} = \begin{pmatrix}1\\ -i\end{pmatrix} = |-i\rangle \tag{1.51}$$

1.6.7 U gate

The U gate is a gate that modifies the state of a qubit through a rotation in the Bloch sphere, according to three angles: θ, λ and ϕ. The matrix of U gate is shown in 1.52.

$$U3(\theta,\phi,\lambda) = \begin{bmatrix}\cos\left(\frac{\theta}{2}\right) & -e^{i\lambda}\sin\left(\frac{\theta}{2}\right)\\ e^{i\phi}\sin\left(\frac{\theta}{2}\right) & e^{i(\phi+\lambda)}\cos\left(\frac{\theta}{2}\right)\end{bmatrix} \tag{1.52}$$

It can be used as a general gate and it can replicate all the other ones. It is possible to have an X gate just by putting the parameters in the correct way, as shown in the example 1.53.

$$U(\pi,\pi,\frac{\pi}{2}) = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix} = X \tag{1.53}$$

1.7 Multiparty Quantum States

More than one qubit can be used to describe a state. For this purpose, tensor products are used, as shown in 1.54.

$$|a\rangle \otimes |b\rangle = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} \otimes \begin{pmatrix} b_0 \\ b_1 \end{pmatrix} = \begin{pmatrix} a_0 b_0 \\ a_0 b_1 \\ a_1 b_0 \\ a_1 b_1 \end{pmatrix} \tag{1.54}$$

Let's make an example where system A is in the state $|1\rangle_A$ and system B in the state $|0\rangle_B$. The total bipartite state, which indicates the quantum information shared between two parties, can be written as $|10\rangle_{AB}$, better represents in 1.55. Values in system A don't depend on values in system B. These states are not correlated with each other and are called uncorrelated.

$$|10\rangle_{AB} := |1\rangle_A \otimes |0\rangle_B \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \tag{1.55}$$

There are some bipartite states that can't be written as a tensor product. They are called correlated states. If this correlation is very strong it's called entangled. Quantum entanglement is the physical phenomenon that occurs when a group of particles are generated, interacting or sharing spatial proximity in such a way that all the particles depend on each other.

1.8 Two qubits gates

In classical computers, most of the gates used have at least two input ports. For example, in the XOR gate, where if the input bits are equal the output is 0, otherwise it is 1.

The limit with classical gates is that they are irreversible, after computing the result is no longer possible to reconstruct the original input. This is in contraposition to the unitary property of quantum theory. For this reason, only reversible gates are considered to get a quantum version of classical ones.

1.8.1 CNOT

CNOT is the most famous two qubits gate, described by the matrix in 1.56 or through Dirac notation in 1.57.

$$CNOT = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{1.56}$$

$$CNOT = |00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 11| + |11\rangle\langle 10| \tag{1.57}$$

The CNOT gate in a quantum circuit is represented in figure 1.3. The first qubit is called the control bit because, if it is set to 1, the second bit, called the target bit, will change.

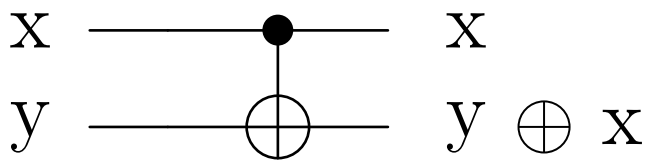

Figure 1.3: CNOT gate

Table 1.1 is the truth table of CNOT. From this table is possible to notice that the first qubit never changes its state, while the second one is 1 if both are different or 0 if they are equal, like the XOR gate.

The presence of the first qubit, which remains always the same, makes this gate reversible and so compatible with the unitary property of quantum theory.

Input		Output	
x	y	x	y
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Table 1.1: CNOT truth table

Every function can be described by a reversible circuit, just adding more output that will keep track of initial information. This means that every quantum circuit can be used to perform the same function as classical computers.

1.9 Entanglement

Entanglement, after superposition, is the second big feature of quantum computing. It is a very strong correlation between qubits that can't be described classically,

with the usage of the tensor product.
The Bell states represent the simplest form of quantum entanglement.

1.9.1 Bell states

The Bell states are four quantum states, each of which is built on two qubits and based on the basis $|0\rangle$ and $|1\rangle$. Different ways to evaluate the entanglement exist and, for each of them, Bell states reach always the maximum score.

- Bell state 1
$$|\Psi^{00}\rangle := \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{1.58}$$
- Bell state 2
$$|\Psi^{01}\rangle := \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{1.59}$$
- Bell state 3
$$|\Psi^{10}\rangle := \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \tag{1.60}$$
- Bell state 4
$$|\Psi^{11}\rangle := \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \tag{1.61}$$

A general form to write them can be 1.62, which is a more compact representation.

$$|\Psi^{ij}\rangle := (\mathbb{1} \otimes \sigma_x^j \sigma_z^i)\,|\Psi^{00}\rangle \tag{1.62}$$

To create them it's needed to have two qubits that implement the circuit shown in figure 1.4.

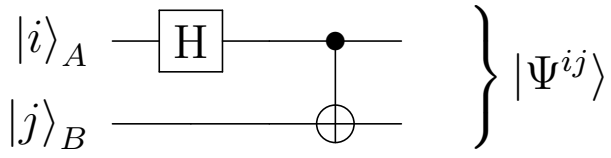

Figure 1.4: Quantum circuit to create Bell states.

This quantum circuit can be combine with the basis state in order to obtain the following results.

- Bell state 1 transaction

$$|00\rangle \xrightarrow{H_A} \frac{1}{\sqrt{2}}(|00\rangle + |10\rangle) \xrightarrow{CNOT_B} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \quad (1.63)$$

- Bell state 2 transaction

$$|01\rangle \xrightarrow{H_A} \frac{1}{\sqrt{2}}(|01\rangle + |11\rangle) \xrightarrow{CNOT_B} \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \quad (1.64)$$

- Bell state 3 transaction

$$|10\rangle \xrightarrow{H_A} \frac{1}{\sqrt{2}}(|00\rangle - |10\rangle) \xrightarrow{CNOT_B} \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \quad (1.65)$$

- Bell state 4 transaction

$$|11\rangle \xrightarrow{H_A} \frac{1}{\sqrt{2}}(|01\rangle - |11\rangle) \xrightarrow{CNOT_B} \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle) \quad (1.66)$$

Through the circuit implemented in figure 1.5, called Bell measurement, is possible to determine in which Bell state two qubits are.

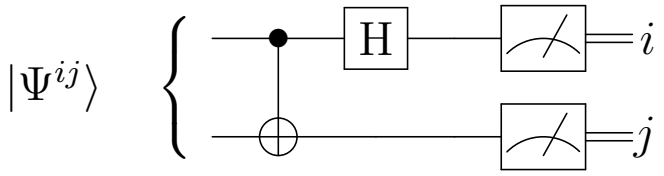

Figure 1.5: Bell measurement.

1.10 Teleportation

Teleportation is a quantum algorithm, which goal is to send the state $|\phi\rangle_S := \alpha|0\rangle_S + \beta|1\rangle_S$ of a system A to a system B, where system A can only send to B classical bits. To do that the two systems must share the maximally entangled state, $|\Psi^{ij}\rangle_{AB}$, in order to allow the exchange of data.

In summary, the Teleportation algorithm is used to transfer quantum information [10]. The initial state of the whole system is described by the following formulas.

In figure 1.6 is drawn the circuit used to implement this algorithm.

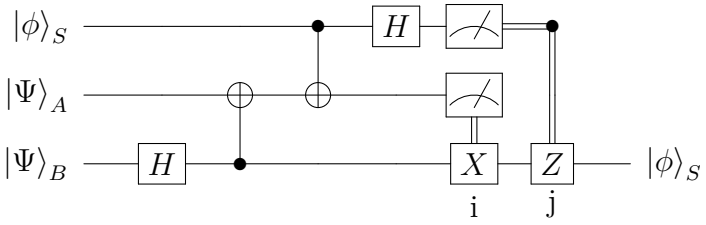

Figure 1.6: Teleportation circuit.

Let's start with both systems A and B sharing the state $|\Psi^{ij}\rangle$. System A has another state, $|\phi\rangle_S$, which wants to share with system B. To exchange quantum information there are three main steps.

- System A performs a Bell measurement between $|\Psi\rangle_A$ and $|\phi\rangle_S$, to know the state of system B, obtaining i and j.
- Then system A sends classical bits, i and j, to system B.
- System B, applying i and j, such that σ_z^i and σ_x^j, to its qubit, is able to get the state $|\phi\rangle_S$.

It's important to notice that the state $|\phi\rangle_S$, because of Bell measurement, collapses and no longer has its initial state $|\phi\rangle_S$.
This phenomenon is called the no-cloning theorem which states that quantum states cannot be copied [11].

1.11 Quantum algorithms

Some of the most know quantum circuits, used in the tests of this thesis as well, are the following ones:

- Deutsch Jozsa, this algorithm is able to recognize if a function is constant or balanced;
- Bernstein Vazirani, this algorithm is an extension of Deutsch Jozsa and it is able to recognize the string encoded in a function;
- Grover, this algorithm can find the unique sequence of input which, given a function, is able to produce a certain output.

These are some of the first algorithms that have demonstrated the power of quantum computing over classical computing.

1.12 How use Quantum computer

To access a quantum computer is necessary to sign in to an online service. Starting from our laptop, information is sent to the cloud, which is used to configure the quantum circuit according to the need.

In order to do this, specific tools exist that allow to simulate quantum hardware on classical computers. The most famous are:

- Qiskit, developed by IBM, allows access to real quantum computers for everyone;
- Cirq, developed by Google, allows access to real quantum computers only for some research teams;
- Pennylane, developed mainly for quantum machine learning.

Qiskit is the one used in this thesis because it allows placing measurement operations in the middle of a quantum circuit. This feature is not always present in these tools, which usually have only terminal measurements.

1.12.1 Transpilation

Quantum circuits can be defined using a wide range of gates. However, most of these abstract gates cannot be directly implemented on real quantum hardware. Transpilation is the process where abstract quantum circuits are turned into quantum circuits that can be directly implemented on a specific quantum computer, within the limitations of that hardware.

Chapter 2

Qubit physical implementations

2.1 Hilbert Space

Qubit superposition states are typically described using the two basis states, $|0\rangle$ and $|1\rangle$, that can be represented through ket-bra notation or through Dirac notation. Such basis states belong to Hilbert's space, a special kind of vector space that satisfies the following ones in addition to all classical properties:

1. It has an inner product (dot product), $\langle\psi_1|\psi_2\rangle \in \mathbb{C}$, that satisfy certain condition.
   (a) Conjugate symmetry.
   $$\langle\psi_1|\psi_2\rangle = \langle\psi_2|\psi_1\rangle^* \tag{2.1}$$
   (b) Linearity with respect to 2^{nd} vector.
   $$\langle\psi_1|a\psi_2 + b\psi_3\rangle = a\langle\psi_1|\psi_2\rangle + b\langle\psi_1|\psi_3\rangle \tag{2.2}$$
   (c) Anti linearity with respect to 1^{st} vector.
   $$\langle a\psi_1 + b\psi_2|\psi_3\rangle = a^*\langle\psi_1|\psi_3\rangle + b^*\langle\psi_2|\psi_3\rangle \tag{2.3}$$
   (d) Positive definiteness.
   $$\langle\psi|\psi\rangle = |\psi|^2 \geq 0 \tag{2.4}$$
   (e) Distance between two vectors.
   $$|\psi_2 - \psi_1| = \sqrt{\langle\psi_2 - \psi_1|\psi_1 - \psi_2\rangle} \tag{2.5}$$

2. Hilbert space is separable, so it contains a countable, dense, subset. In this way, if a countable subset S contain a vector ϕ_n, ($S = \{\phi_n\}$), it is possible to count the numbers of elements in that subset. Instead, a subset is dense if every element in the Hilbert space is either a member of that subset S or can be made arbitrarily close to one where closeness is determined according to distance formula 2.5.

3. Hilbert space is complete, there aren't gaps. For instance, in $\mathbb{Q}$ there are gaps that are occupied by elements in $\mathbb{R}$, which is a complete space. Every Cauchy sequence 2.6 converges to an element ϕ in Hilbert space 2.7.

$$\lim_{n,m\to\infty} |\psi_m - \psi_n| = 0 \tag{2.6}$$

$$\lim_{n,\to\infty} |\phi - \psi_n| = 0 \tag{2.7}$$

2.2 Qubit real implementation

The qubit idea is to represent the basis states using energy level. The minimum level of energy is associated with$|0\rangle$, while the maximum level of energy is associated with $|1\rangle$. To realize these two levels in practice the characteristics of atoms are exploited.

2.2.1 Atom

Let's make an experiment where a cloud of atoms has been put under some excitations, such as a voltage source, will shine producing light. If this light passes through some lens, hitting a prism, will be possible to observe different colours separated by a specific distance. Different atoms have different colour configurations but, regardless of the type of atom taken under consideration, the same colours appear always in the same position.

Niels Bohr proposed to treat atoms, not as part of classical physic but, instead, considering their energy as discretized, according to some specific levels and amounts [12]. In this way is possible to explain, in the previous experiment, the discrete emission of the atom colours, where each of the colours represents a different energy level. This phenom is easily recognised in fireworks.

In a regular system, the energy between the nuclear atom and the electron is inversely proportional to the distance (2.8). In quantum systems, these levels are quantized and it's not possible to stay in the middle.

$$V(r) \simeq \frac{1}{r} \tag{2.8}$$

Energy levels are quantized in a special way, called anharmonic, where the distance between them is not equally, as shown in figure 2.1.

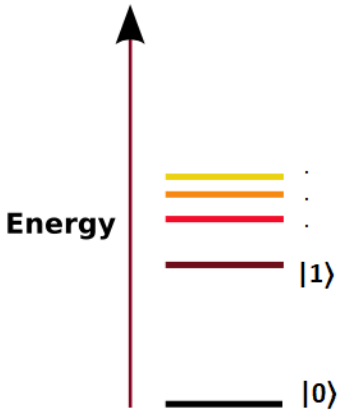

Figure 2.1: Atom energy levels.

If a microwave light, at a certain frequency, is applied to the atom, which starts from the ground state, $|0\rangle$, then that atom is subjected to a force. If the frequency is equal to a precise value, called transition frequency, then the energy level of the atoms moves to $|1\rangle$.

It will be impossible for the atoms to move to the upper levels, because of anharmonic, which also determines that different frequencies are needed to move between levels.
Once reached the $|1\rangle$ energy level, the transition frequency needed to move to the next layer isn't the same and so atoms remain in $|1\rangle$. This means that it is possible to take under consideration the only lowest two energy levels to isolate the qubit subspace.

2.3 Circuit Quantum Electrondynamics

Atoms energetic behavior can be generated artificially, through electromagnetic circuits made by a capacitor and an inductor. In this way, the electron magnetic energy levels created are harmonic, all separate by the same distance, and it is impossible to separate $|0\rangle$ and $|1\rangle$. To avoid this problem it's possible to introduce a non-linear inductor, called Joseph junction, which allows the creation of an

anharmonic oscillator. All these basic circuit elements are used to create a Circuit Quantum Electrondynamics (cQED).

Superconducting qubits [13] are a class of qubits which exploit superconductivity [14, 15], a property that allows some materials to conduct current without any loss of energy.
To give them superconductivity the following elements are needed:

- Nominally zero intrinsic dissipation and heat
- Nominally temperature far below energy level splitting
- Non-linear, robust Josephson tunnel junction effect

Transmon qubits are a type of superconducting qubits [16], implemented on dielectric, usually silicon. They are widely used nowadays and consist of:

- A substrate, composed of a dielectric, such as silicon.
- Two metal pads on top of the substrate.
- A non-linear inductor that connects the two pads, just like a Josephson junction.

An image of a simplified transmon qubit is shown in the figure 2.2. One pad has a positive charge while the other one has a negative charge. Electrons condense in superconducting pairs, creating Cooper Pairs, where they move without energy dissipation. This happens because the entire system is kept at a very low temperate, almost zero kelvin.

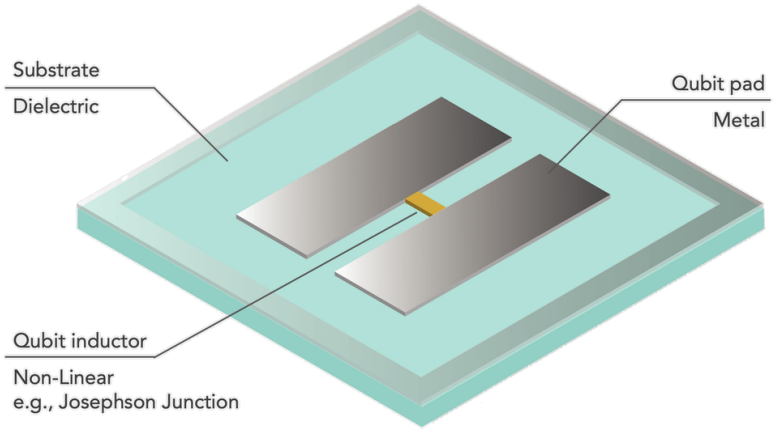

Figure 2.2: Transmon qubit, image taken from Zlatko Minev presentation.

Two electric charges in a dielectric create an electric field, which can be used to define a voltage between the two pads.

$$v(t) = -\int_{\vec{r_a}}^{\vec{r_b}} \vec{E}(\vec{r}, t) \cdot d\vec{l}(\vec{r}) \qquad v \in [-\infty, +\infty] \tag{2.9}$$

The pads can be modelled as a simple capacitor. If a charge moves across the junction a current is generated which creates a magnetic field. A lot of magnetic fields create a magnetic flux.

$$\frac{d}{dt}Q(t) = i(t) \qquad Q(t) = \int_{-\infty}^{t} v(\tau)d\tau \qquad Q(-\infty) = 0 \tag{2.10}$$

$$\frac{d}{dt}\Phi(t) = v(t) \qquad \Phi(t) = \int_{-\infty}^{t} v(\tau)d\tau \qquad \Phi(-\infty) = 0 \tag{2.11}$$

$$Q(t) = Cv(t) \qquad C \in [0, +\infty] \tag{2.12}$$

$$\Phi(t) = Li(t) \qquad L \in [0, +\infty] \tag{2.13}$$

The energy stored in a component, like a capacitor or an inductor, has a rate of change which determine the power from which it's possible to retrieve energy, using formula 2.14. According to this formula, it's possible to calculate the energy associated with a capacitor, formula 2.15, ad the one associated with an inductor, formula 2.16.

$$\frac{d}{dt}\varepsilon(t) = p(t) = v(t)i(t) \qquad \varepsilon(t) = \int_{t}^{-\infty} p(\tau)d\tau \qquad \varepsilon(-\infty) = 0 \tag{2.14}$$

$$\varepsilon_{cap}(\dot{\Phi}) = \frac{1}{2}C\dot{\Phi}^2 \tag{2.15}$$

$$\varepsilon_{cap}(\Phi) = \frac{\Phi^2}{2L} \tag{2.16}$$

In the circuit in figure 2.3, both the inductor and the capacitor have an associated charge and magnetic flux, related to each other.

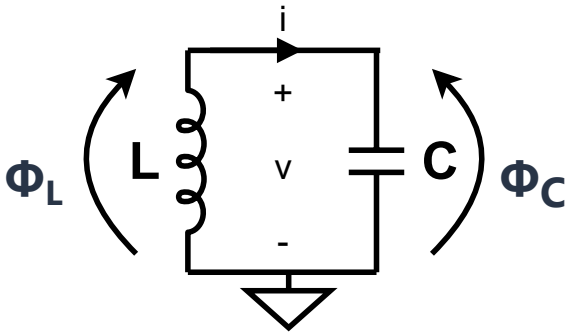

Figure 2.3: LC circuit

According to Kirchhoff's current law, the charge in the capacitor must be equal to the charge in the inductor, so that the sum of the currents must be zero, like in formula 2.17.

$$\dot{Q}_C + \dot{Q}_L = 0 \tag{2.17}$$

According to Kirchhoff's voltage law, also the sum of voltages across the branches must be zero, as in formula 2.18. From this assumption can be derived that the voltage across the capacitor is the same as the one across the inductor.

$$\dot{\Phi}_C - \dot{\Phi}_L = 0 \qquad \dot{\Phi}_C = \dot{\Phi}_L = \Phi \tag{2.18}$$

Thus, the charge on the capacitor, described by the formula 2.12, has its derivate equal to formula 2.19.

$$\dot{Q} = C\ddot{\Phi}_c \tag{2.19}$$

The inductor instead, has flux equal to 2.13, from which can be derivate the formula 2.20.

$$\Phi = L\dot{Q}_L \tag{2.20}$$

By replacing in the formula 2.13 the 2.19 and 2.20, a new one is obtained, the formula 2.21, in which it is possible to observe an oscillator analogy.

$$C\ddot{\Phi} + L^{-1}\Phi = 0 \tag{2.21}$$

The resonance frequency, $w0$, and the classical oscillator equation are given by formulas 2.22. The flux oscillates according to the formula 2.23.

$$\ddot{\Phi} = -w_0^2 \Phi, \qquad w_0 = \frac{1}{LC} \tag{2.22}$$

$$\Phi(t) = \Phi_0 e^{-iw_0 t} \tag{2.23}$$

Looking at a mass-spring harmonic oscillator it is possible to make a comparison. In the spring, the equilibrium position $x(t) = 0$ is the equivalent of $\Phi(t) = 0$ in the cQED. From Kirchhoff's observation, can derive an equation of motion to describe how magnetic flux oscillates.

Charge Q, in the formula 2.22, is no longer present, there is only variable Φ, which represents the position, while $\dot{\Phi}(t)$ is the velocity. This is mechanics and regular tools can be used to describe this dynamic system. Given a configuration space, the Lagrangian is the difference between the kinetic energy, due to the capacitor, and potential energy, due to the inductance, in formula 2.24 [17].

$$L(\Phi, \dot{\Phi}) = \varepsilon_{cap}(\dot{\Phi}) - \varepsilon_{ind}(\Phi) = \frac{1}{2} C \dot{\Phi}^2 - \frac{\Phi^2}{2L} \tag{2.24}$$

In the Lagrangian is very simple to take the value of a conserved quantity [17], while in mechanics it is the momentum, in circuits it is the charge. It can be derived in the following way, according to formula 2.25.

$$Q = \frac{\partial L}{\partial \dot{\Phi}} = C \dot{\Phi} \tag{2.25}$$

For further confirmation, using the Euler-Lagrange equation 2.26 [17], it's possible to derive the one that describes the oscillation 2.22.

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\Phi}} = \frac{\partial L}{\partial \Phi} \tag{2.26}$$

After all this mathematical observation it is possible to use the charge, intends like momentum, and the magnetic flux, intends like position, to define the Hamiltonian of an LC circuit, defined in formula 2.27 [18]. It is defined as the sum of kinetic energy and potential energy, which are respectively expressed in terms of charge (momentum) and magnetic flux (position).

$$\mathcal{H}(\Phi, Q) = \frac{Q^2}{2C} + \frac{\Phi^2}{2L} \tag{2.27}$$

The Hamiltonian is the total energy of the system. In Hamiltonian, a particle describes the parable on which can move at a certain velocity. Hamiltonian describes how the total energy is distributed between kinetic energy, on the capacitor, and potential energy, on the inductor, as shown in figure 2.4.

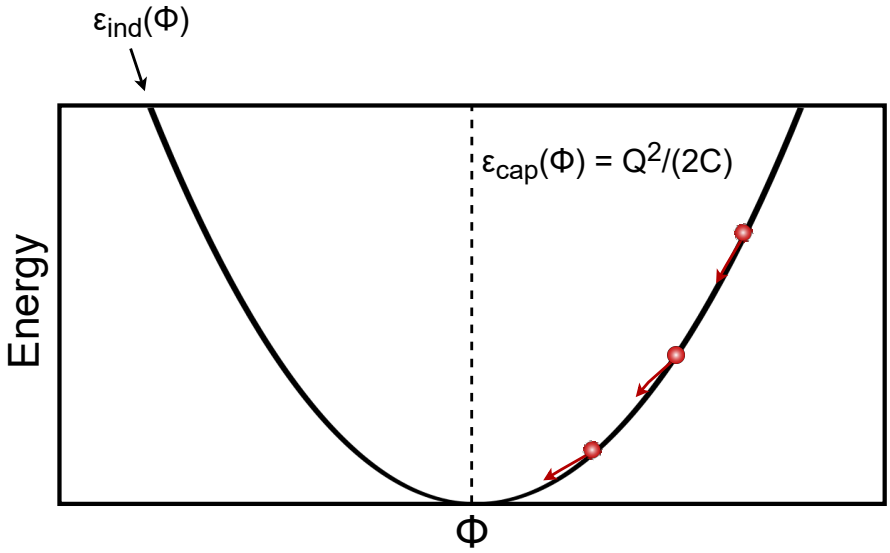

Figure 2.4: LC classical harmonic oscillator

The speed of movement is given by $\varepsilon_{cap}(Q)$ while the position is given by $\varepsilon_{ind}(\Phi)$, respectively the kinetic energy and the potential energy.

Now let's suppose to have $C = L = 1$, just for simplicity. It's possible to express the discretized energy of a quantum harmonic oscillator as in formula 2.28. In this way, through Plank's constant $\hbar$, a quantum action has been introduced, while w_0 is the resonance frequency of the oscillator.

The energy of the quantum oscillator can be put equal to the formula of classical energy, described in terms of Hamiltonian in formula 2.27, to represent the quantum system.

$$E = \hbar w_0 \left(n + \frac{1}{2}\right) = \frac{1}{2}(Q^2 + \Phi^2) \tag{2.28}$$

If the energy is fixed, in a space of Q and Φ, the trajectory gives a circle, which changes according to the chosen n, used to move along the rings. In classical physic is not mandatory to stay in one of these rings, but it's allowed to stay in between. In quantum, this is not possible.

From the Hamiltonian equation of motion, which is another way to rewrite Kirchhoff's law, it's possible to see how the electric charges move in the circuit, passing through the capacitor and the inductor and back and forth. To execute this movement the charge Q and the magnetic flux Φ change describing rings according to the graph in 2.5.

The movement in a ring can be described through complex numbers, using a very helpfully variable, as shown in formula 2.29.

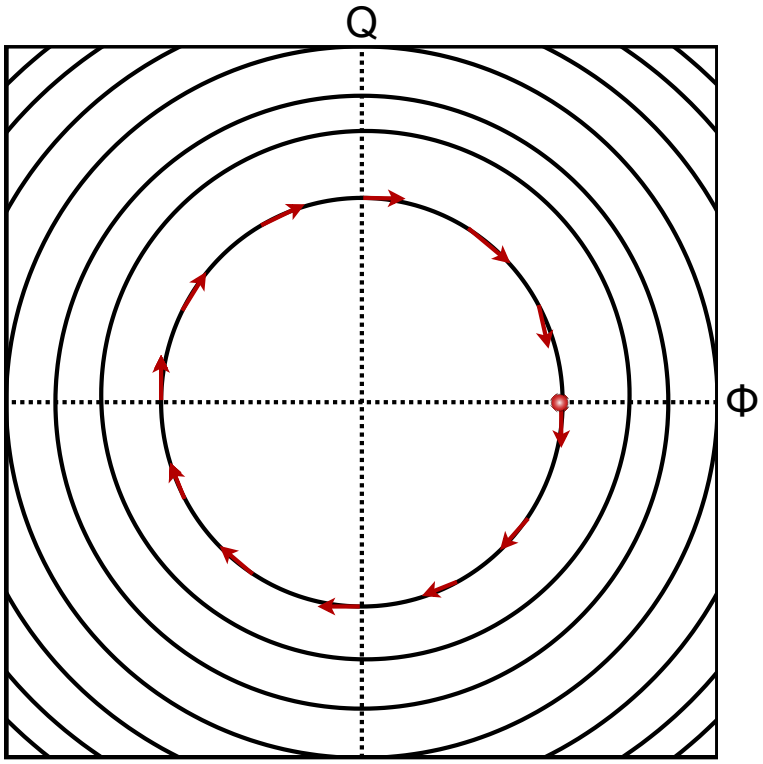

Figure 2.5: Particle trajectory in a space of Q and Φ, which represent the movement between the capacitor and inductor.

$$\alpha(t) = \sqrt{\frac{1}{2hZ}}[\Phi(t) + iZQ(t)] = a(0)e^{-iw_0 t} \qquad Z = \frac{L}{C} \tag{2.29}$$

Rewriting the Hamiltonian using this new variable, it becomes just a function of α and α^*. This $\alpha(t)$ is the base description of most of the transmon qubit.

$$\mathcal{H}(\alpha, \alpha*) = \frac{1}{2}hw_0(\alpha^*\alpha + \alpha\alpha*) \tag{2.30}$$

With these mathematical tools, it is possible to describe how the qubit is originate and how they work.

2.4 Qubit in quantum computer

Quantum computers are kept inside very expensive fridges, able to reach almost 0 Kelvin, to assure that qubits work correctly, to remain in a state of superconductivity.

Besides qubits, the other main components are wires and a lot of shields to prevent environmental noise.

Qubit is coupled to a readout resonator, a transmission line which allow to guide the microwave to force and to control the qubit [19]. It is also used to communicate the result of qubit measurement to the external environment. The figure 2.6 shown the schematic circuit used.

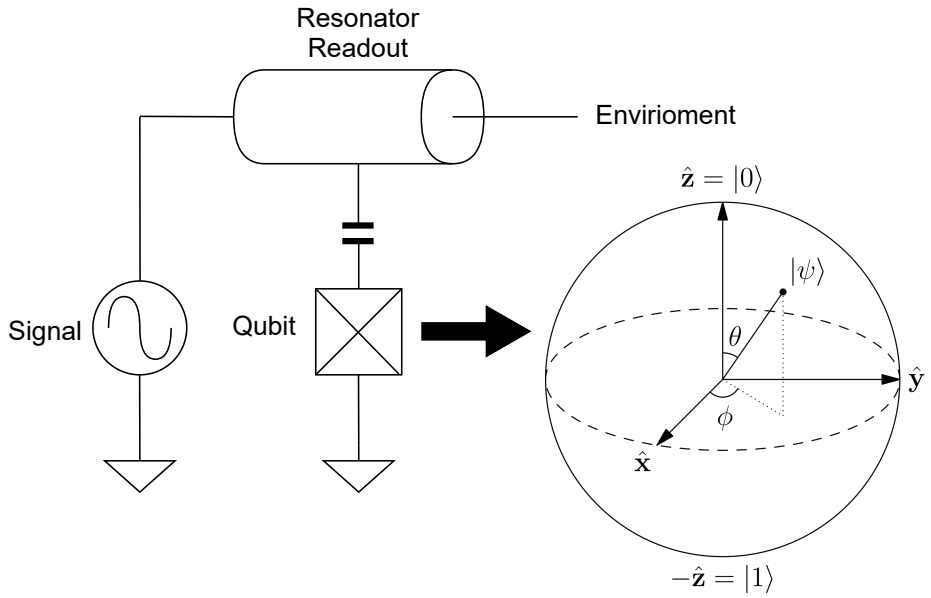

Figure 2.6: Standard qubit circuit setup.

Chapter 3

Noise in Quantum Computers

3.1 Noise problems

From a mathematical point of view, in the Hilbert's space can be described as the ideal behaviour of a qubit. Unlucky, executions of real quantum algorithms, on a real quantum machine, are slightly different and it is impossible to avoid errors.

> Quantum phenomena do not occur in a Hilbert space. They occur in a laboratory. (Asher Peres, Quantum Theory: Concepts and Methods [20])

With the elements used to define quantum circuits, IBM has run a simple experiment to demonstrate how errors affect quantum devices. In the experiment, they create N quantum circuits, all set at $|0\rangle$ at the beginning and with a Z-measurement at the end. The n-circuit, where n is an integer number including between 0 and $N-1$, has in the middle n X-gates, like in figure 3.1.

With this chain of X-gates, the qubit states change continuously from $|0\rangle$ to $|1\rangle$ and vice versa. If the n-circuit has a paired number of X-gates, the measured value is -1, otherwise, if the number of X-gates is odd, it is 1, as explained for Z-measurement in chapter 1.4.

In an ideal simulation, the results obtained from several runs are all 1s or all -1s, according to the depth of the circuit, as shown in the graph 3.2. Instead, on a real quantum device, no circuit has all results equal to 1 or -1, as shown in the graph 3.3.

From graph 3.3 it is possible to observe a kind of oscillation, related to the depth of the circuit, which highlights a relation with the number of the X-gates placed. This behaviour is characteristic of coherent quantum noise. All real gates have a fixed error ϵ, which is added each time a gate is employed and it is responsible for

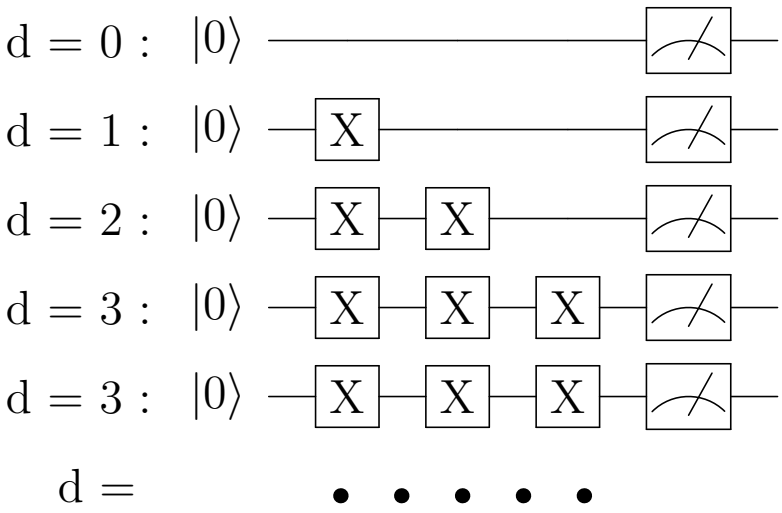

Figure 3.1: Different circuits with a variable number of X gates, identify as the depth of the circuit.

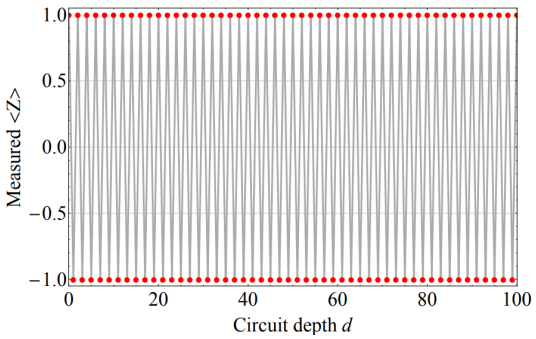

Figure 3.2: Experiment runs in an ideal case.

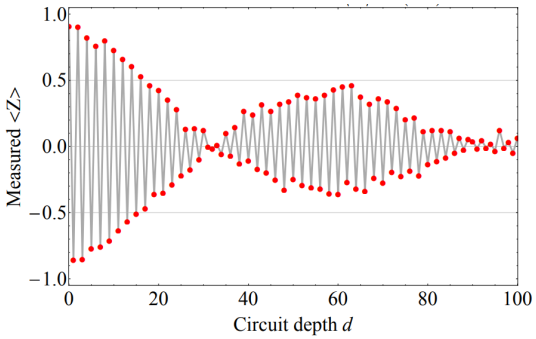

Figure 3.3: Experiment run in a real case.

the oscillation described above. Even when no gates are presented, the measurement is not perfect because the apparatus of measurement is quite noisy.

In addition, it is also possible to notice a decaying trend with the increase of depth, called incoherent noise, which represents the random dynamics of the quantum system.

Only by understanding how quantum noise acts on quantum computers, it is possible to avoid it, conceiving strategies and methods able to undo these processes in order to mitigate or completely delete these errors.

3.2 Quantum Error Correction

Quantum algorithms are designed without considering noise. During each computation quantum circuits are constantly affected by errors and, at the end, the correct result must be picked among all the wrong ones. Sometimes it is not so easy to make this choice, as show in figure 3.4.

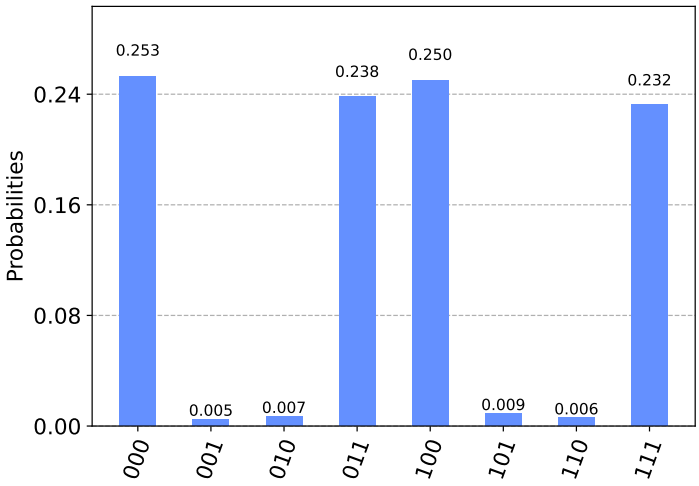

Figure 3.4: Output probabilities of a quantum execution where it is not easy to choose the correct result.

Quantum error correction, QEC is a useful set of techniques used to protect quantum information from noise. The basic idea of QEC is to use more than one physical qubits in order to create a logical qubit. Ideal qubits are called logical qubits, which represent perfectly the ideal qubit, in absence of noise. Instead, the real qubit, the one psychically present inside quantum computers, is called physical qubit. The more physical qubits are used, the more accurate the logical qubit will be.

This qubit overhead could be a drawback, especially in quantum machines that don't have a lot of psychical qubit available. Even the most recent and advanced quantum computers are still not able to perform such corrections to allow the execution of useful quantum algorithms.

Despite this, QEC protocols are continually proposed in order to create more reliably quantum devices.

3.2.1 Encoding and decoding

Decoding and encoding procedures are at the base of any protocol of quantum error correction.

The algorithms used for encoding and decoding must be known, respectively by the encoder and decoder, and must be chosen in order to make the message more robust against noise. To do that different elements are needed:

- Input, which is the information that must be protected.
- Encoding, which is the process that transforms the basic information, using more bits than in the beginning, to protect them.
- Error, which is a perturbation that randomly affects the encoded message. Of course, this element is not always present.
- Decoding, in this step the encoded message is recomposed to recreate the original input. It can understand if an error arises and how to correct it.

This approach fits very well for transmission data, where the encoding and decoding are applied only once at the beginning and at the end of the transmission. In quantum circuits, errors can occur at each operation performed and so, to avoid them, decoding and re-encoding must be performed for each operation. To be more efficiently these steps are not executed completely, but just in part in order to find and fix the faults.

3.2.2 Repetition code

Repetition code is the simplest technique of error correction [21].

A message sends through a nosy channel, can arrive damaged to the destination with a probability p. Sending the message more than once, the probability, P, to get a wrong message exponentially decreases whit the increase of the number of repetitions, according to formula 3.1, where d is the number of repetitions and $d/2$ is the minimum number of probability needed to create problem in the system.

$$P = \sum_{n=0}^{[d/2]} \binom{d}{n} p^n (1-p)^{d-n} \sim \left(\frac{p}{1-p}\right)^{[d/2]} \tag{3.1}$$

Among all the received messages, the one with the highest recurrence will be choosen. Until $d/2 - 1$ repetitions have an error, the correct value will be still confirmed by the other $d/2 + 1$ ones. While, if $d/2 + 1$ repetitions have an error, then the system could be no longer able to pick the correct value.

Repetition code works great for classical bits but could cause problems for qubits. Let's encode a superposition state as 3.2, where the used qubits have been triplicated compared to the beginning.

$$\alpha\,|0\rangle + \beta\,|1\rangle \rightarrow \alpha\,|000\rangle + \beta\,|111\rangle \tag{3.2}$$

The decoding step requires measurement, which implies the destruction of superposition of the qubit, a side effect that can cause other types of problems and that must be avoided. The purpose of this measurement is to discover the presence of possible errors and not measure the qubit to know in which state it is. In order to do this, it is only needed to understand which are the qubits that have different values.

This can be done by putting some extra qubits in the circuit, called ancilla, one for each pair of near qubits. Then a couple of CNOT gates are placed to detect possible differences between repetitions. One CNOT gate is controlled by the first qubit of the pair while the other is controlled by the second one, as in figure 3.5.

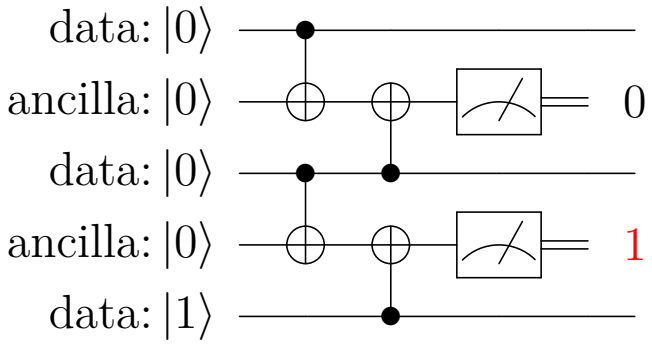

Figure 3.5: This circuit implements the repetition code and check the differences between qubits without collapsing the qubit state. In the third data qubit an error arises and it is detected by second ancilla qubit.

Let's analyze the case example in figure 3.5. In the first pair, both qubits are 0 and so the CNOT gate doesn't flip the ancilla qubit and, in the end, it measures 0. In case both qubits were 1, both CNOT gates flip the ancilla qubit and it measures always 0. Instead, in the second pair the qubits are different and, while the first qubit does nothing, the second one, which is a 1, flips the ancilla bit and, in the end, it measures 1, detecting the presence of an error.

The difference between qubits is highlighted when a 1 is measured on the ancilla bit. Moreover, the value measured on the ancilla bit can be used to drive an X-gate that compensates for the error.

Using this technique it is possible to avoid the collapse of the qubit states to know if there are some differences between repetitions.

3.2.3 Surface code

Surface code is the most popular and widely used technique of error correction in quantum computers. It is based on stabilizer code.

The stabilizer code is a type of encoding where a parity check is executed in order to take information about which type of error has affected the quantum

circuit [22]. In figure 3.6 and 3.7 are representing two circuits that respectively represent an X error detector, to detect a bit flip error, and a Z error detector, to detect a phase shift error. All the other errors are a combination of these ones. In case of an error, the ancilla qubit will report it.

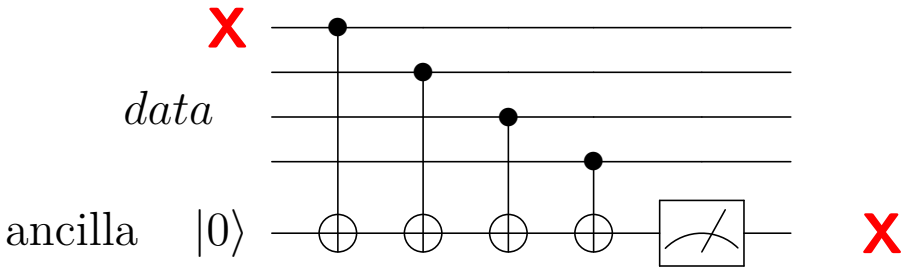

Figure 3.6: Circuit used to report an X error on one of the four qubits

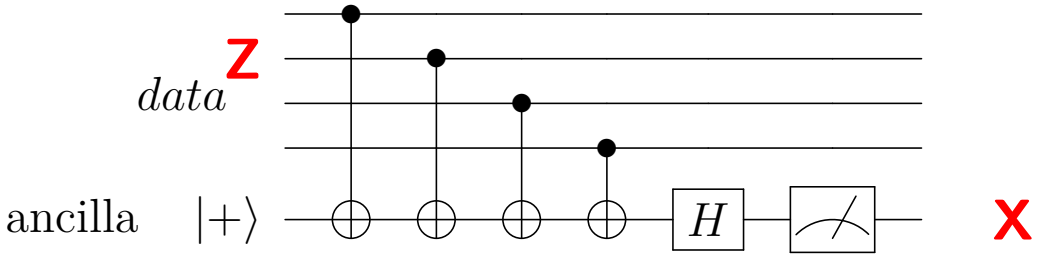

Figure 3.7: Circuit used to report a Z error on one of the four qubits

Error correction can detect just a subset of errors and it is particularly effective to correct errors on a single qubit but not errors that affect two or more. The basic idea in error correction is to find and correct the most likely errors while the less probably ones could be not identified.

In surface code, all the physical qubits, used to create a logic one, can be directly linked together on a 2D planar surface through local connections. Different type of encoding of surface code exists, which convert k logical qubits into n physical ones, where n is greater than k. The main characteristic is the distance, which indicates the number of errors that a specific surface code encoding is able to correct. A code with a distance d is able to correct up to $\frac{d}{2}$ errors.

Conventional computers have error rates lower than 10^{-17} while, current quantum devices, have error rates greater than 10^{-2} [23]. Through surface code will be possible to reach a very low error rate [24, 25], still far from the actual numbers of digital devices but, with the increased number of physical qubits, an acceptable error rate could be achieved which will allow the execution of useful quantum algorithms.

3.3 Correlated error

Quantum computers have intrinsic errors that must be reduced in order to allow the execution of useful algorithms, which employ a lot of gates. To fix these types of errors, called uncorrelated because they appear independently from time and space, the techniques explained before are used.

While the uncorrelated errors can be exponentially suppressed with the scaling devices, the correlated ones affect almost all the qubits simultaneously [26], causing logical faults [27], making useless techniques of QEC, like surface code [26, 28]. They are the main challenge for today's quantum computers.

Energetic radiations cause these correlated errors, in particular Cosmic rays, heavy particles coming from the sun at nearly the speed of light [29], are the main reason. Hitting the Earth's atmosphere, they generate secondary particles that increase the amount of radiation in the biosphere[30, 31, 26].

Superconducting qubits, which are the most used ones, are built using Silicon. It is well known, from the studies on semiconductor reliability [32], that the interaction of energetic particles with Silicon generates electron-hole pairs, which cause a non-equilibrium in the qubit. Cosmic rays impact the coherence of the qubit by broken Cooper pairs, generating quasiparticles [33, 34, 35], which may cause the collapse of the qubit or a transient change of the state [35, 36, 37, 38, 39, 40].

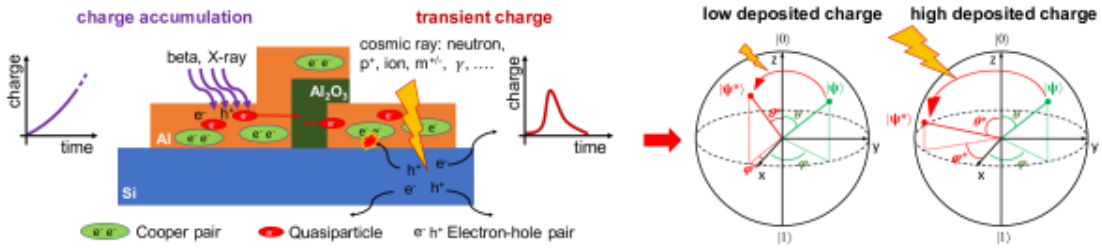

Figure 3.8: Impact of radiation on superconducting qubits [41], adapt from [33].

To avoid correlated errors quantum devices are placed in underground facilities, shielded by lead cryostat, in order to prevent the burst of quasiparticles [42, 34]. Other valid techniques do not yet exist.

3.3.1 Google research

Google researchers made a very interesting experiment where they show up how difficult is to deal with transient faults and all the problems caused by them [28].

They used a subset of qubits of Google Sycamore, a quantum process with 53 qubits [16], composed of 26 qubits. On each of them, a simple quantum circuit has been built to, first of all, set the quantum state to $|1\rangle$ and then, after an idle time

of 1 µs, to measure the qubits state, all at the same time. These operations have been executed periodically on the qubits with an interval of 100 µs.

In an ideal world, the expected result is made up of all 1s but, on Google quantum processor, each time qubits are measured, around 4 of 26 of them are 0. This is the expected behaviour caused by uncorrelated errors that can be detected and repaired by QEC strategies.

This experiment lasted 30 s and, at a certain point, all qubits have been measured as 0. The peak of errors lasts around 25 ms, generating a transient fault in the system, that survives even after reset and measurement operations. This is a clear signature of quasiparticle poisoning throughout the chip, caused by correlated error.

In figure 3.9 is drawn the implemented circuit and the design of the quantum processor. There is also a graph of the errors on qubits along the time, with a zoom in the interesting zone.

It is evident how, when a transient fault arises, it spreads out in all processor qubits, making useless QEC because all qubits are faulty, and it last longer than the QEC takes.

The same experiment has been performed also by setting qubits to $|0\rangle$ and then measuring them. Proceeding in this way no correlated error has been detected, confirming the presence of quasiparticles in the Joseph junction. This conclusion can be asserted because quasiparticles lead to a decay of the qubit states to ground and so, when qubits are set to $|0\rangle$, in case of a transient fault arises, they remain in the ground state.

This behaviour model of the Google Sycamore processor has been exploited in the chapter 5 to design a sensor able to discover the presence of correlated errors.

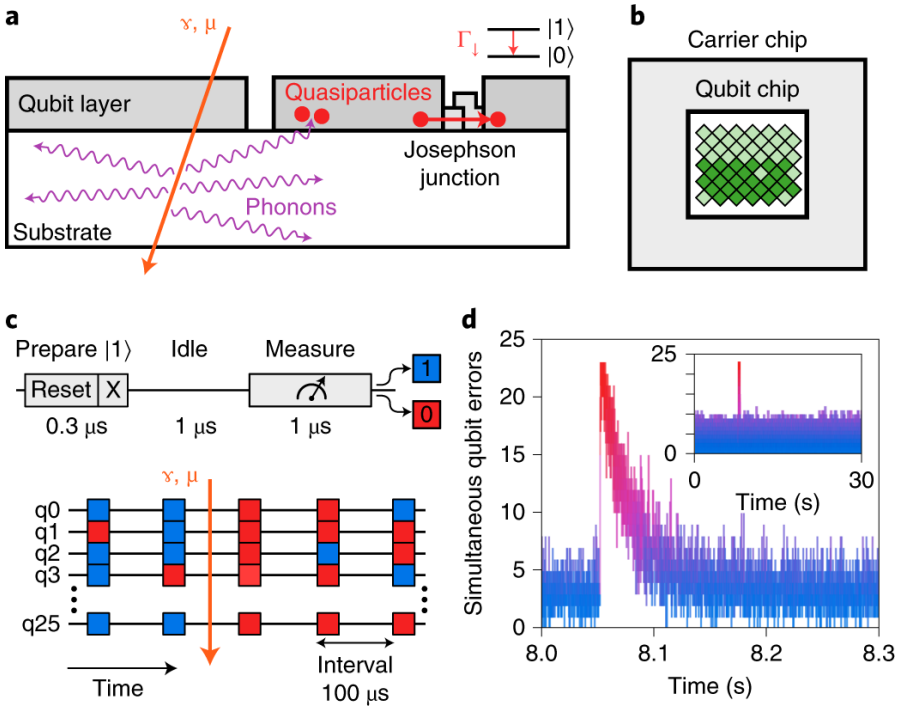

Figure 3.9: Google experiment results that have been used to model the behaviour of transient fault in quantum circuits [28].

3.4 QuFI

To better understand how these errors can affect the circuit, transient faults have been simulated by software, using the QuFI (Quantum Fault Injector) framework [41]. Through QuFI it has been possible to tune the phase shift magnitude based on the proximity of the qubit to the particle strike location, to see how the fault is propagated throughout the circuit.

3.4.1 Model of transient fault

When a superconducting qubit is affected by a charged deposition, its state starts to change. In order to model this error, a U gate, described by the matrix in 1.52, has been used to mimic the shift of the qubit state, by setting ϕ and θ values.

In all the possible points of the circuit, combinations of different parameters of the U gate, referred as error gate, have been injected to see which are the effects caused by different possible faults. In figure 3.10 is possible to observe an example of error gate injection. The range of values used by U gate as parameters are the following:

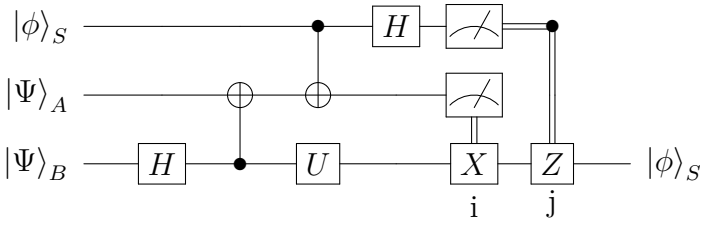

Figure 3.10: Teleportation circuit with an injected fault, simulated by a U gate.

- $\phi = [0,2\pi[$, with a step of $\frac{\pi}{12}$;
- $\theta = [0, \pi]$, with a step of $\frac{\pi}{12}$;
- $\lambda = 0$

3.4.2 Quantum Vulnerability Factor

To evaluate the effect of transient fault on a quantum circuit is used a parameter called Quantum Vulnerability Factor, QVF.

QVF describes the probability that a transient fault has to propagate to the output and which are the consequences. It is based on Michelson Contrast [43], which gives a measure of distinguishable between two objects that belong to the same family. It can be used for colours, luminous or, as in this case, to see the differences between a faulty-free system and one with an injected fault.

After the execution of a quantum circuit, different outputs will be obtained, with different probabilities associated. If $P(A)$ is the probability of the correct result and $P(B)$ is the highest probability between the wrong outputs, then the Contrast is calculated as in the formula 3.3.

$$Contrast = \frac{P(A) - P(B)}{P(A) + P(B)} \tag{3.3}$$

The Contrast value is included between -1, if $P(B) = 1$ and $P(A) = 0$, and 1, if $P(A) = 1$ and $P(B) = 0$. To normalize it to the range [0,1], in order to obtain QVF, the Contrast value is converted as in formula 3.4.

$$QVF = 1 - (Contrast + 1)/2 \tag{3.4}$$

A QVF value tending to 0 means that the correct result can be clearly identified among the wrong ones. Otherwise, a QVF tending to 1 means that an incorrect output has a probability much higher than the correct one. Instead, a QVF around 0.5 means that the correct and the wrong result has almost the same probability.

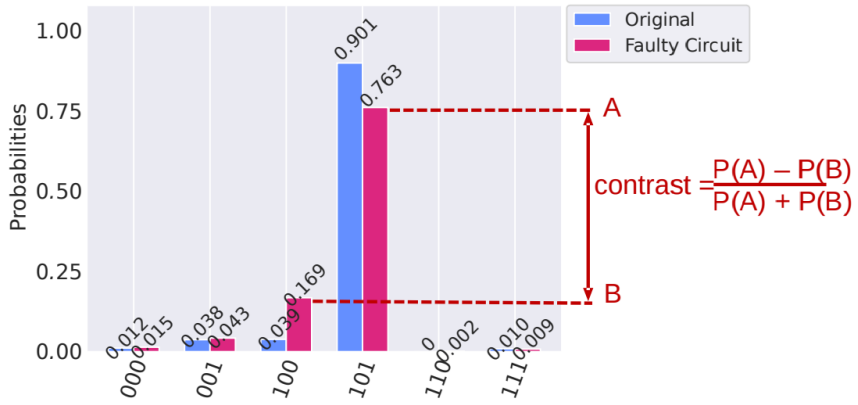

Figure 3.11: How to calculate Contrast for QVF evaluation, image taken from [44].

The effect that an injected fault produces on the circuit is evaluated through the value of QVF, in the following way:

- $QVF < 0.45$, the fault doesn't have a big impact on the circuit and the correct result is still the most probable one.
- $QVF > 0.55$, the fault has a big impact on the circuit and an incorrect result is of sure the most probable one.
- $0.45 < QVF < 0.55$, the fault has not a big impact on the circuit and the correct result and an incorrect one have a similar probability, in this situation the output becomes dubious.

At the same point in the circuit, all angle combinations for the U gate are generated, with the parameters shown above. For each point of in the circuit, the QVF values of all angle combinations have been calculated and then put together in a heatmap, as in figure 3.12.

In figure 3.12 the green squares ($QVF < 0.45$) represent the faults which don't have an effect on the circuit and don't produce any misbehaviour, they are harmless to the circuit execution. Red squares ($QVF > 0.55$) represent dangerous faults that create problems and they must be mitigated. White squares ($0.45 < QVF < 0.55$) indicate that more outputs have the same probability and it's not easy to determine which is the correct one.
Heatmaps with more green and white squares indicate more fault-tolerant circuits, more robust to transient faults.

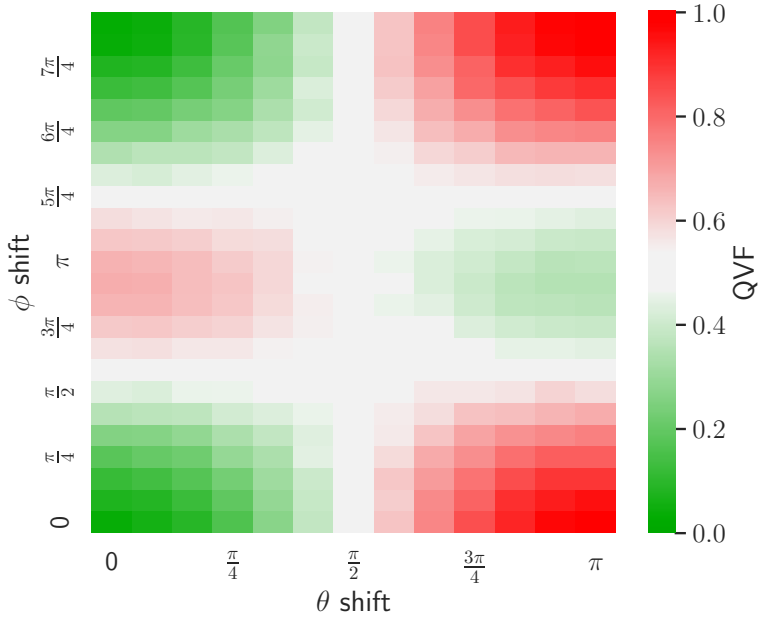

Figure 3.12: QVF heatmap example.

3.4.3 QuFI modification

As shown in chapter 3.3.1, after a reset or a measurement operation, transient errors are still there and last for several milliseconds.

In the previous version of QuFI, the error gates were placed in the circuit without checking the presence of a reset or a measurement operation.

To implement also this feature in QuFI, some controls have been inserted. For each error gate, placed on a specific qubit and at a precise point of the circuit, a check is performed to see if a reset or a measurement operation is present, always on the same qubit but after that point. If it is so, a further error gate is added to propagate the fault, as shown in figure 3.13. In this way, the simulator behaviour is much more realistic, as demonstrated by Google research.

3.5 NISQ era

With the passing of the years, quantum computers are becoming a more concrete reality. A lot of new algorithms have been designed to be executed on these devices with an exponential speed-up. One of the problems, that does not allow yet their execution, is the sensitivity of qubits to noise. Running a big useful algorithm will require a large number of qubits, still not available on modern quantum computers, and many operations that could lead to a spread of errors in the quantum circuit.

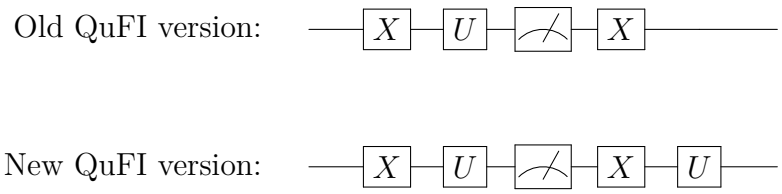

Figure 3.13: On the new version of QuFI is possible to see how the error gate is propagated also after the measurement operation. In the case of reset, the behaviour is the same.

Another issue is that nowadays qubits can store data only for a few milliseconds. This problem is called decoherence and determines the decline of the qubit states over an amount of time. The decoherence process can be measured at two different times:

- T1 also called energy relaxation, is the time taken for the excited state, $|1\rangle$, to decay toward the ground state, $|0\rangle$;
- T2, called dephasing, describes how long the phase of a qubit can remain intact.

Noisy Intermediate-Scale Quantum (NISQ) technology will be available in the near future, with quantum processors that will have a much greater number of qubits available, more accurate gates and eventually, fully, or almost, fault-tolerant quantum computers[45]. In this way, useful algorithms will be able to be executed without any problems.

Chapter 4

Technique to mitigate transient Fault in quantum computers

4.1 Comparison with digital circuit

In digital circuits, transient faults, which occur in a random way, are caused mainly by environmental perturbations. These perturbations arise because of the radiation source. The radiation-induced particles can hit atoms of semiconductor devices transferring their energy by means of ionizing or non-ionizing processes. Depending on the energy and flow of the particles, the effects of such energy on devices based on MOS technology can be transient, permanent, cumulative, or even destructive.

The no-deterministic nature of these errors makes them not easy to deal with. To avoid this problem the following techniques are used:

- hardened technologies;
- hardened by design techniques.

4.2 Hardened technologies

Human beings know how to be robust to radiation. In testing, a hardened technology is a method which makes hardware reliable with respect to the faults of concern. Using this technology should be a guarantee that a set of faults cannot happen during the lifetime of the systems [46, 47].

Commercial devices didn't use this kind of technology because it has a huge cost and also performance limitations. The hardened technologies method is mainly

used for integrated circuits designed for space, where reliability must be guaranteed. An example is the New Horizions, a NASA mission which aims to reach Pluto, launched in 2006 that is still going [48].

4.3 Hardened by design techniques

Hardened by design techniques are based on three main features:

- hardware redundancy;
- information redundancy;
- time redundancy.

4.3.1 Hardware redundancy

In hardware redundancy, there is an area overhead in order for dealing with transient faults.
A widely used technique is the Triple Modular Redundancy (TMR) [49], where 3 processors receive the same input and work synchronously, executing the same operations. In the end, a voter checks the three outputs to determine the correct result. If all processors produce the same result, it means that there are no errors in any processor and it is easy to determine the correct output. If only two of them produce the same result, it means that there is one or more errors in one processor, but the other two are used to confirm the correct output. If all produce a different result, it means that there are errors in at least two processors and it is not possible to determine the correct output. However, the latter situation has a very low probability of occurring, and the probability that two faulty systems producing the same result is even less.

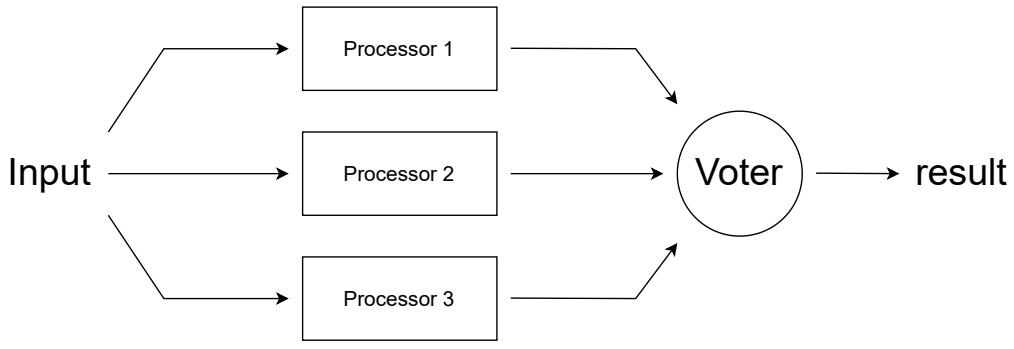

Figure 4.1: Triple Modular Redundancy (TMR) schematic implementation.

It is also possible to simply duplicate the processor, comparing only the two results. In this case, if the two results are different, it will be only possible to detect a problem, without performing any kind of correction. This technique is called Duplication With Comparison (DWC) and the two devices used could not be the same.

Increasing the number of hardware repetitions means that, the probability to get the wrong result, because of transient faults, decreases. Unfortunately, this also means more area overhead.

4.3.2 Information redundancy

Information redundancy is often used to improve the reliability of memories and communication channels. Repeating the same data more than once, it is possible to detect, and sometimes even correct, errors affecting the original information.

The original data D is encoded, over N bits, through an encoded function $F()$ that, according to input, produces the encoded information K (codeword), as shown in 4.1. From the codeword, it is possible to retrieve the original data D, through a decoding function F^{-1}, as shown in4.2. The bits used to encode information, to improve the reliability of the system, are greater than those used in the original data.

$$K = F(D) \tag{4.1}$$

$$D = F^{-1}(K) \tag{4.2}$$

F() is such that errors transform a codeword K in a non-codeword K^* so that for each D of the system, $K^* \neq F(D)$. According to the $F()$ chosen, the code becomes resilient to a set of faults. More bits are used for encoding and bigger will be the cardinality of this set.
Examples of information redundancy are the following:

- Parity codes: N bits are encoded in N+1 bits, wherein the extra bit stores the parity computation. If the number of 1s in data is even it becomes 1, otherwise, if it is odd, it becomes 0. This technique is able to detect all the situations where a pair number of qubits are affected by a fault. Any single error in the codeword can be detected but not corrected.
- Hamming codes: using more bits for error correction, it is possible to discover faulty bits. Hamming (11, 7) is also known as single error correction and double error detection (SEC/DEC). Using 11 bits to encode 7 bits, and through a specific encoded function $F()$, is possible not only to detect errors, in case of single or double faulty bits, but it is even capable to correct the error, in case of a single faulty bit.

4.3.3 Time redundancy

In time redundancy more time than necessary is used for processing inputs. The same operations are executed more than once, on the same device, but in different slots of time.

In a precise moment, a transient fault could appear, changing the expected result for that slot of time. The slot intervals are chosen in order to avoid the same fault affecting two slots. In the end, all the outputs feed a voter that, in case of double execution, just checks if an error arises while, in case of triple execution, decides the correct result among all the outputs.

This is similar to the hardware redundancy method but, in this case, less area is used while more time is spent. According to the implementation constrained, one of the two mechanisms can be selected, according to the need.

4.4 Use digital techniques on quantum computers

In quantum computers, the effects of transient faults are slightly different than in classical computers. However, the techniques explained above could be usable, with some adaptation.

While in classical computers it is known how to shield bits from radiation, it is not yet clear how to do the same thing for qubits. Qubits are very sensitive to environmental perturbations and so, in real quantum devices, all the available hardened techniques are employed to achieve better reliability. Nowadays, hardened technology is probably the most efficient way to protect quantum computers from noise [50].

Trying to implement techniques of time redundancy is still not possible. Qubits coherence time is still low, sometimes it is not even enough for one execution [51].

Information redundancy is a technique widely used in quantum computation that allows detecting, and in some cases also correcting faults. An example is the repetition code. Unluckily, in quantum devices is not always possible to apply these methods. The number of qubits available in the latest quantum computers is not enough for simple elaborations, let alone for repeated codes.

The previous proposal's techniques don't fit very well with modern quantum computers, which can't still implement the same strategies used to avoid transient faults in digital devices.

To solve this problem, in this thesis, new methods have been developed, able to detect the presence of transient faults. It's called *Transient Fault Detector* and it is better explained in the chapter 5.

Chapter 5

Quantum Transient Fault Detector

5.1 How they work

According to Google's paper, when in a superconducting qubit a transient fault arises and its state is $|1\rangle$, then it decays to $|0\rangle$ [28]. This behaviour can be described, in a more general way, as a phase shift of the qubit state and it can be simulated through QuFI [41].

In this research, this phenomenon is exploited in order to recognize the presence of transient faults. To do that the initial circuit is wrapped whit a particularly circuitry, called *Transient Fault Detector* (TFD).

These TFDs are added at the beginning and at the end of the initial circuit. A circuit modified in this way is called *wrapped circuit*, and its representation is shown in figure 5.1.

TFD can be added to all qubits used in the circuit algorithm, but also to all the unused ones that are still available in the quantum computer. These redundant gates have the purpose to put the qubit state at $|1\rangle$, starting from $|0\rangle$, and then measuring always a 1.

It is important to notice that, after the execution of the *start TFD*, the qubit state must be set to $|0\rangle$ and, even after the execution of the *general circuit*, the qubit state must be set to $|0\rangle$ before to beginning with the elaboration of *end TFD*. This could be achieved by exploited or the unitary property of the quantum gate, using the inverse circuit, or through reset gates.

In wrapped circuits, the final results will contain the values coming from the quantum circuit that implements the algorithm, referred as the algorithm circuit, and the ones coming from the two barriers. In the end, only the results containing all 1s in the classical bits used to measure TFD will be taken into consideration,

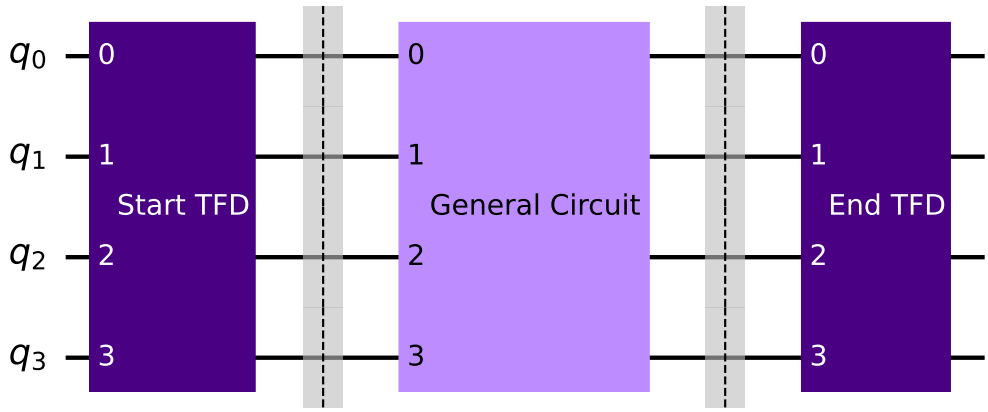

Figure 5.1: General wrapper implementation to detected faults.

the others will be marked as faulty.

5.1.1 Example

Let's suppose that the final result, of the not wrapped circuit, is $xxxx$, where x is a general value that could be 1 or 0. Instead, if the circuit is wrapped, then the final correct result becomes $1111xxxx1111$, where the two groups of 1s ensure the absence of faults and represent the bit of TFD.

5.2 Wrappers

Different kinds of wrappers have been tested with different results. The purpose of each of them is to move along all the possible positions inside the Bloch sphere, which represent all the possible superpositions of the qubit, trying to catch an unwanted shift state, highlighted by the TFD.

If an error is detected, the final qubit state that will be measured at the barriers will be no more $|1\rangle$, but $|0\rangle$. In order to do this, following the experiment of Google's team, the qubit starts from state $|0\rangle$ to arrive at $|1\rangle$, trying to cover the whole surface of the sphere. If an error is detected, the final state that will be measured at the barriers will be no more $|1\rangle$, but $|0\rangle$

Different wrappers are better to cover different types of errors, that will be mimicked through the use of QuFI [41].

5.2.1 Wrapper 1

The circuitry used for this scope is represented in figure 5.2. The *result_1* are the classical bits used to save the result while the others, *star_1* and *end_1*, are used to detect possible errors, respectively at the beginning and at the end of the circuit.

In this wrapper the start Barrier Detector, to reset the end of the qubits for the algorithm circuit, uses the inverse circuit, while, the end Barrier Detector uses the reset gate.

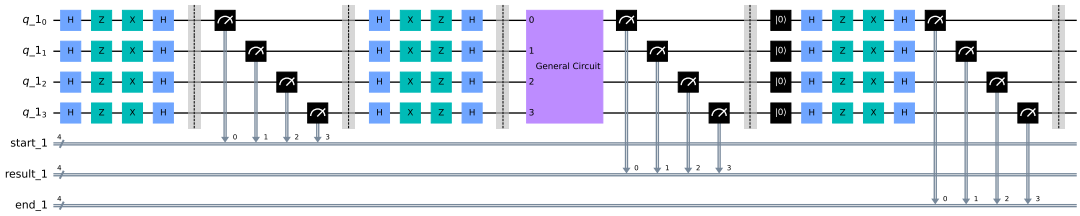

Figure 5.2: A first type of wrapper.

To explain how these barriers work, the following images show the qubit state, using Bloch sphere representation, associated with the position in the circuit.

In figure 5.3 the qubit starts from $|0\rangle$, independently if it is at the beginning or at the end of the circuit.

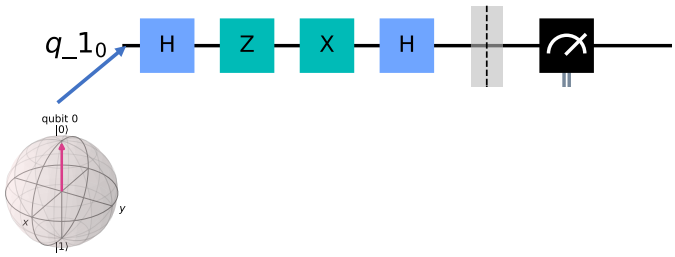

Figure 5.3: The first step of wrapper 1.

Then, through a Hadamard gate, it is put in superposition, figure 5.4, to try to detect all the possible errors that could modify the qubit position along the x-axis and y-axis.

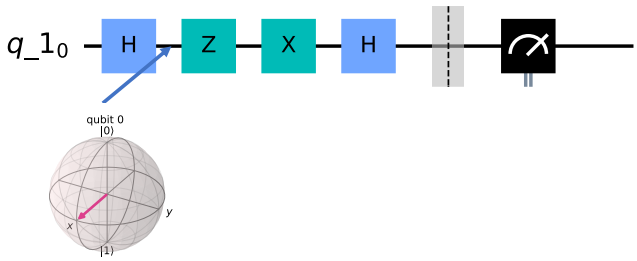

Figure 5.4: The second step of wrapper 1.

In the third step, in figure 5.5, the qubit position is moved in the opposite direction, with ϵ equal to 180°, trying to discover errors along the equator of the Bloch sphere.

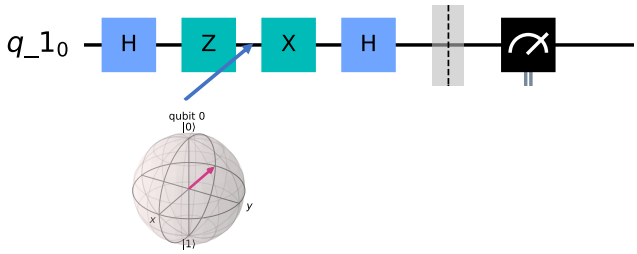

Figure 5.5: The third step of wrapper 1.

In the fourth step, in figure 5.6, seems like no action has been performed but, if a fault is present, it executes a shift of θ along the z-axis, changing the qubit state. The current qubit state will be reflected in the opposite hemisphere of the sphere.

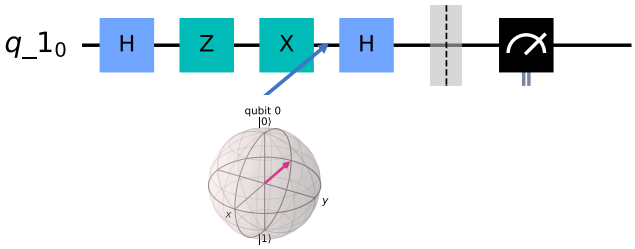

Figure 5.6: The fourth step of wrapper 1.

In the last step, figure 5.7, if no error occurs in the circuit then, through a Hadamard gate, the qubit will reach easily the $|1\rangle$ state. However, in case of problems, the probability of measured $|0\rangle$ increases, indicating that an error is affecting the circuit.

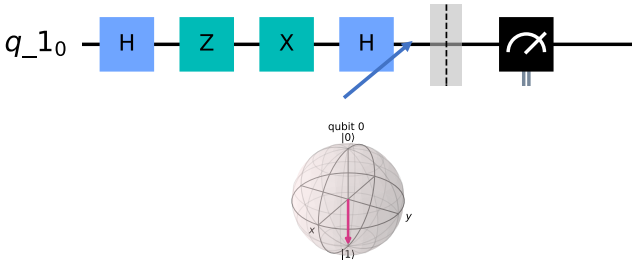

Figure 5.7: The fifth step of wrapper 1.

5.2.2 Wrapper 2

This kind of wrapper, shown in figure 5.8, has been derived from the transient faults model of Google research [28].

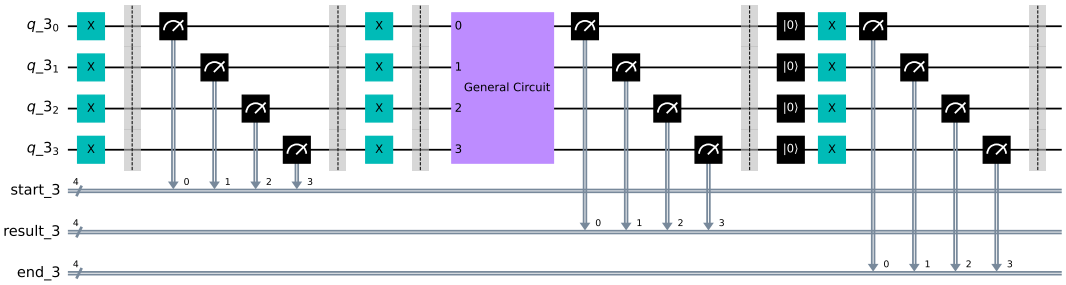

Figure 5.8: A second type of wrapper to detect faults.

A real hardware implementation of this kind of barrier has been already evaluated on Google Sycamore quantum processor [28], demonstrating its effective operation. In this wrapper, the TFDs implement the same circuit used in the Google experiment to highlight the presence of transient faults.

Even in this case the start Barrier Detector uses the inverse circuit, while the end Barrier Detector uses the reset gate.

5.3 QVF Taxonomy

To evaluate the efficiency of the wrappers the Quantum Vulnerability Factor has been used. QVF determine how many final results obtained, composed of quantum barrier measurements and circuit algorithm measurements, are affected by injected faults.

When in the classical bits, used to measure the TFD, a 0 is present, it means that a fault has been detected in the system. With the presence of that 0 the probability to get the correct output result, the one with all 1s in classical Detector barrier bits, decrease, while the probability to get a wrong result increase. In this way the also QVF values increase, as explained in the chapter 3.4.2.

QVF could be used to understand which is the efficiency of the wrapper against the injected fault. For each fault, it's possible to mark a label, according to the obtained QVF value:

- **good**, QVF values between 0.0 and 0.45, the green square in the heatmap, when it is sure of the absence of a fault;
- **faulty not detect**, QVF values between 0.45 and 0.55, the white, or almost, squares in the heatmap, when it is not sure of the presence of a fault;
- **faulty detect**, QVF values between 0.55 and 1.0, the red squares in the heatmap, when it is sure the presence of a fault.

5.3.1 Example 1

Image to have the following circuit, shown in figure 5.9, and to transpile it, as in figure 5.10. Then the circuit has been wrapped like in the figure 5.11.

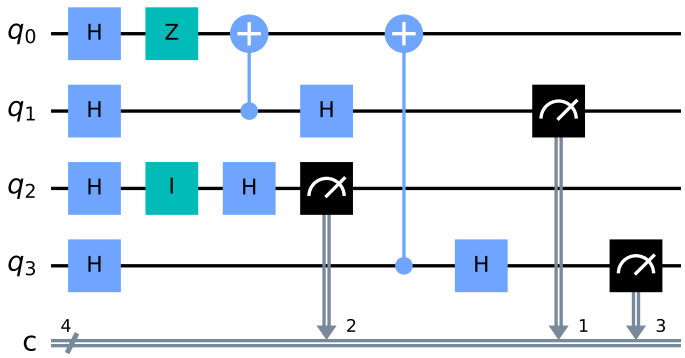

Figure 5.9: Bernstein-Vazirani_4 circuit.

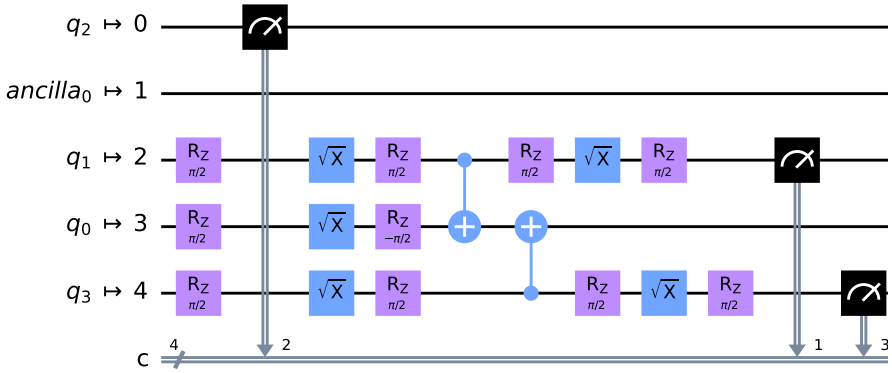

Figure 5.10: Bernstein-Vazirani_4 transpiled circuit.

For fault simulation, the transpiled circuit has been preferred to better analyse all the positions where a fault could be injected on the real hardware implementation of the circuit algorithm.

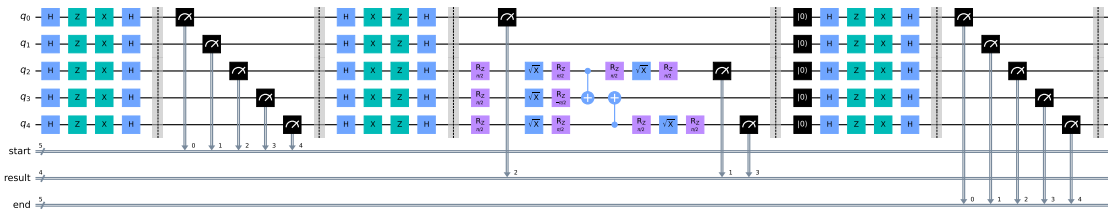

Figure 5.11: Bernstein-Vazirani_4 transpiled circuit with wrapper 1.

Now let's analyze what happens if different faults occur in the same point of the transpiled circuit, with and without the wrapper. In the unwrapped circuit, the fault has been placed in position 5, as in figure , while in the wrapped one it has been placed in position 53, as in figure 5.13.

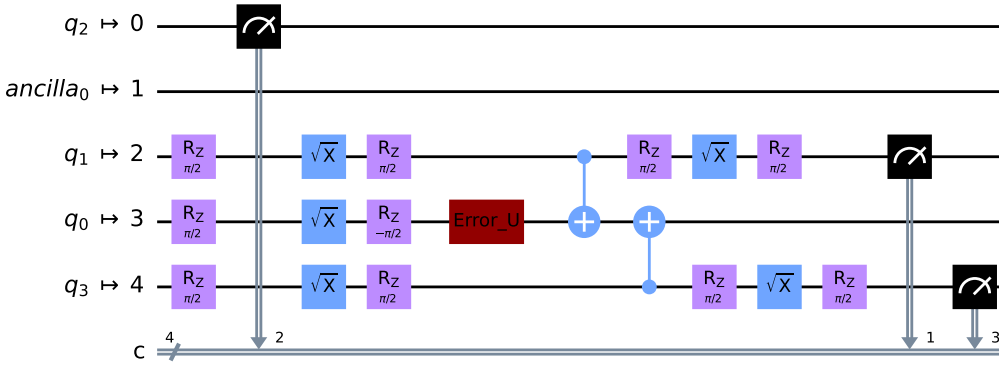

Figure 5.12: Bernstein-Vazirani_4 transpiled circuit without wrapper where a fault arise.

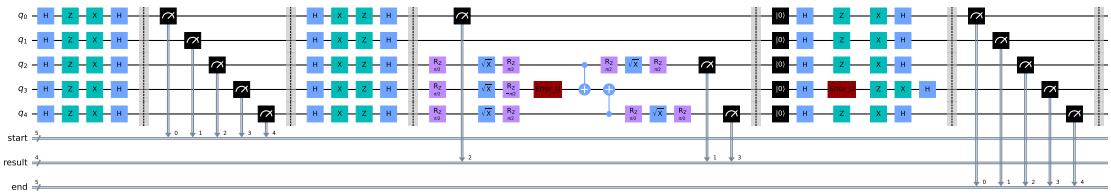

Figure 5.13: Bernstein-Vazirani_4 transpiled circuit with wrapper where a fault arise.

How it is possible to notice in the wrapped circuit, the error gate has been inserted also after the reset, as already said in chapter 3.4, to better mimicked the transient fault behaviour.

The injected fault is a U gate, where different combinations of ϕ and θ have been provided for different fault simulations. For each of them, has been calculated the QVF for the unwrapped circuit, shown in figure 5.14, and for the wrapped one, shown in figure 5.15.

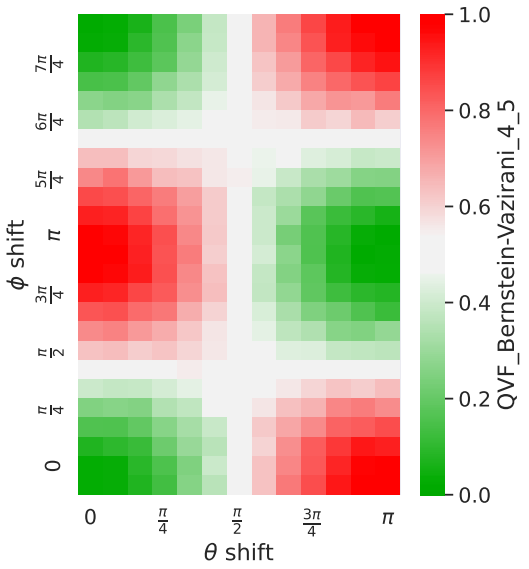

Figure 5.14: Heatmap of Bernstein-Vazirani_4 transpiled circuit without wrapper where a fault arises.

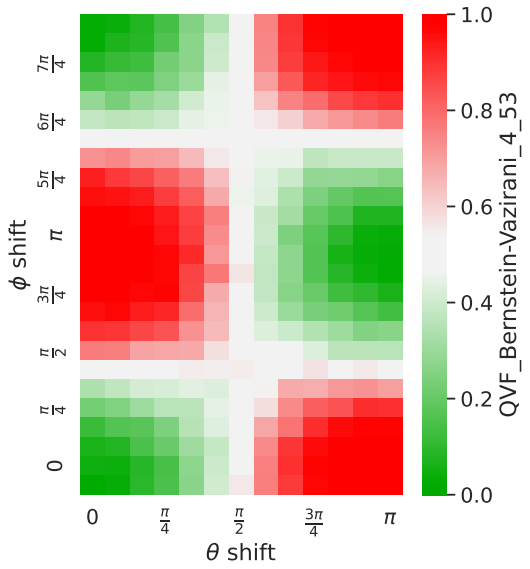

Figure 5.15: Heatmap of Bernstein-Vazirani_4 transpiled circuit with wrapper where a fault arises.

It's possible to notice that the two heatmaps are slightly different. In the heatmap related to the wrapper circuit, fewr white sections are present, while the green ones are almost the same. Instead, the number of red squares is increased, which indicated that different dangerous faults, through the wrappers, have been recognised by barriers.

5.3.2 Example 2

Let's analyze always the previous circuit, the one in figure 5.9, but now error arises in a different position. For for the unwrapped circuit it has been placed in position 0, as shown in figure 5.16, while in the wrapped one it has been placed in position 48, as shown in figure 5.17.

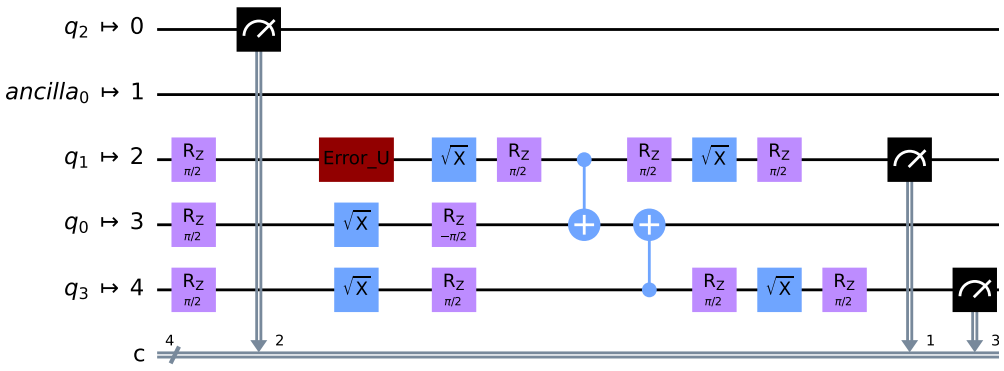

Figure 5.16: Bernstein-Vazirani_4 transpiled circuit without wrapper where a generic fault arises in position 4.

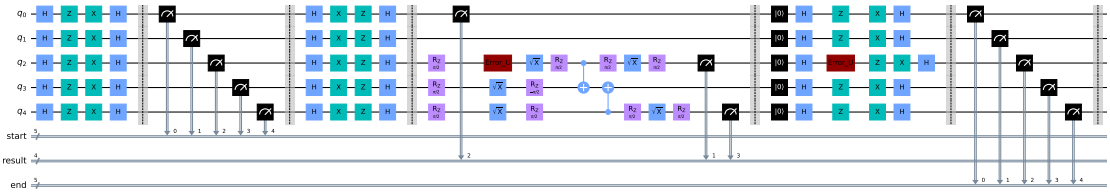

Figure 5.17: Bernstein-Vazirani_4 transpiled circuit with wrapper 1 where a generic fault arises in position 48.

This time, analysing the two heatmaps, shown figure 5.18 and figure 5.19, it's possible to observe that the number of green squares has decreased while the red and white ones have increased.

From the heatmap of the unwrapped circuit, the graph in 5.18, is evident that if the injected fault has a θ value in the range $[0, \pi/2[$, then the probability of correct output is still higher that the wrong ones. While, if the θ value is in the range $]\pi/2, \pi]$, the probability of correct output decreases.

Instead, in the heatmap of the unwrapped circuit, the graph in 5.19, it is possible to see red squares where they used to be green. Some angle combinations, that before were labelled as good (green squares), now have been labelled as fault detected (red squares). The identification of these new faults happens because the circuit used for the barriers detects the fault, even if it is harmless for the circuit, as previously shown in the heatmap of the unwrapped circuit 5.18.

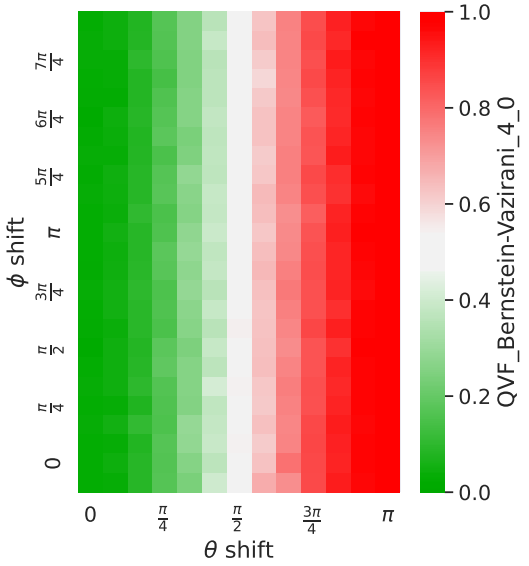

Figure 5.18: Heatmap of Bernstein-Vazirani_4 transpiled circuit without the wrapper, where faults arise in position 4.

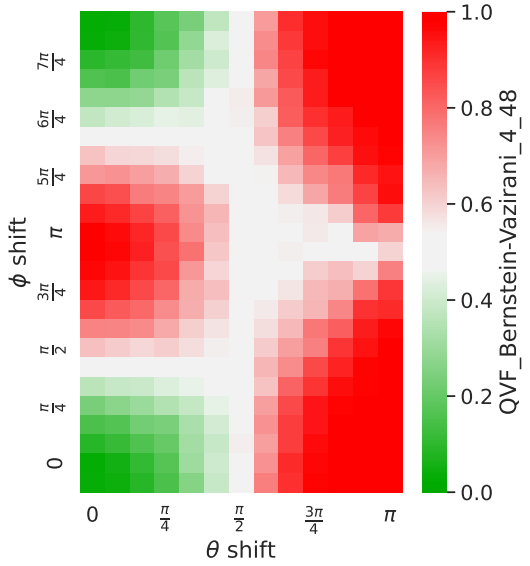

Figure 5.19: Heatmap of Bernstein-Vazirani_4 transpiled circuit with wrapper 1, where faults arise in position 48.

Likewise, also others angle combinations, that before were labelled as detected, now are labelled as not detected. The presence of these new not detectable faults happens because some angle combinations of injected fault can't be recognized by barriers. In this way, the classical bits measured associated with the barriers remain all 1s, so the obtained QVF value decreases compared to the one coming from the unwrapped circuit.

5.4 Real implementation taxonomy

Because the quantum outputs are probabilistic, the values of the measured TFD are probabilistic as well. On real quantum circuits, the only way to see if a real transient fault arises is by checking the percentage of the barriers that measure all 1s. A threshold percentage is set to distinguish between detected faults and not detected ones, referred as threshold barrier percentage.

On the real hardware, QVF can't be used as a metric to estimate if the final elaboration is fault free or if it has been affected by a transient fault. Despite this, QVF can be used to understand if the value of the threshold percentage chosen is good or not.

Both for detected faults and for undetected faults, three classes have been

defined, based on the QVF value, referred as error classes.
The faults marked as not detected by barriers could belong to one of these error classes:

- Undetectable (Undet), when a fault does not create any problems with the correct output. QVF is in the range [0.0,0.3[.
- Silent corruption (Silent), when a fault creates some problems with the correct output. QVF is in the range [0.3,0.7].
- Untriggered (Untr), when a fault creates serious problems with the correct output. QVF is in the range]0.7,1.0].

Instead, the faults marked as detected by barriers could belong to one of these error classes.

- False positive (False), when a fault does not create any problems with the correct output. QVF is in the range [0.0,0.3[.
- Detectable 0 (Det 0), when a fault creates some problems with the correct output. QVF is in the range [0.3,0.7].
- Detectable 1 (Det 1), when a fault creates serious problems with to the correct output. QVF is in the range]0.7,1.0].

The table 5.4summarizes all the possible combinations of these six classes that could be assigned to a fault.

QVF range	Barrier detected fault	Barrier not detected fault
[0.0,0.3[	False Positive	Undetectable
[0.3,0.7]	Detectable 0	Silent corruption
]0.7,1.0]	Detectable 1	Untriggered

Good wrappers have the lowest number of faults assigned to these error classes:

- Silent corruption, the barrier is not triggered but there is a possible dangerous error which affects the circuit.
- Untriggered, the barrier is not triggered but there is a dangerous error which affects the circuit.

- False positive, the barrier is triggered but there is not a dangerous error which affects the circuit.

Let's analyze some circuits to see which could be the better threshold barrier percentage to assign. In the following example, to simulate a quantum system has been used Qiskit. This SDK can perform measurement operations in the middle of the circuit. As backend simulation has been used *fake Santiago*, composed of 5 qubits.

5.4.1 Bernstein Vazirani 4

The initial circuit is the Bernstein Vazirani, based on 4 qubits, as shown in figure 5.9, which has been transpiled, as figure 5.10 and in the end has been wrapped, using wrapper 1, as in figure 5.11.

In the graphs in 5.24 it is possible to see, according to the different threshold barrier percentages, which are the percentages of each error class.

- With a threshold barrier percentage below 20%, less than 30% of the injected errors are detected and around 10% of them are false positive, while 30% of the faults not detected are silent or untriggered errors.

- Using a threshold barrier percentage equal to 55%, more than 60% of the errors are detected, 35.8% of them are false positives while only around 3% of the fault not detected are silent or untriggered errors.

- Using a threshold barrier percentage equal to 75%, silent or untriggered faults are no longer detected, but almost 50% of detected fault is false positives.

- With a threshold barrier percentage beyond 80%, the percentage of TFD all injected errors are detected and around 62% of them are false positives.

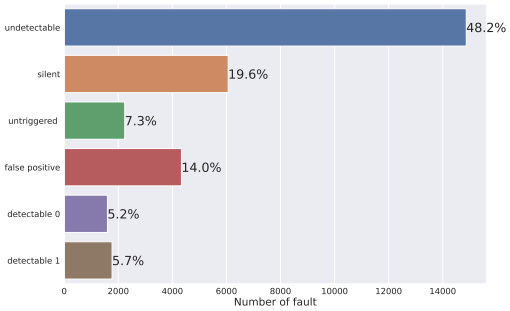

Figure 5.20: Bernstein Vazirani 4 percentage 20%.

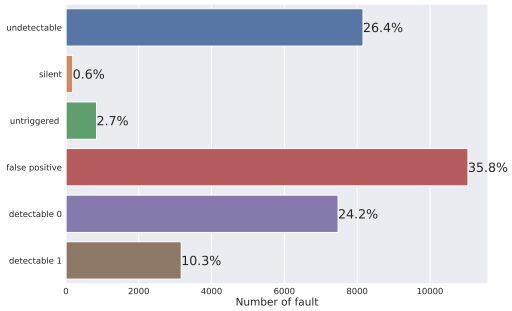

Figure 5.21: Bernstein Vazirani 4 percentage 55%.

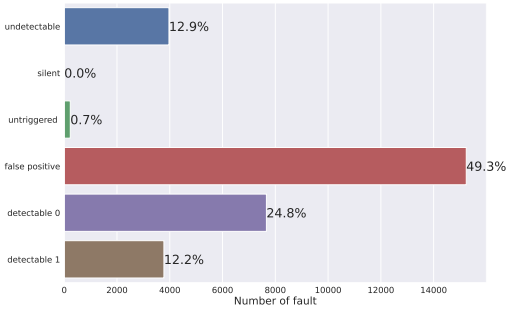

Figure 5.22: Bernstein Vazirani 4 percentage 75%.

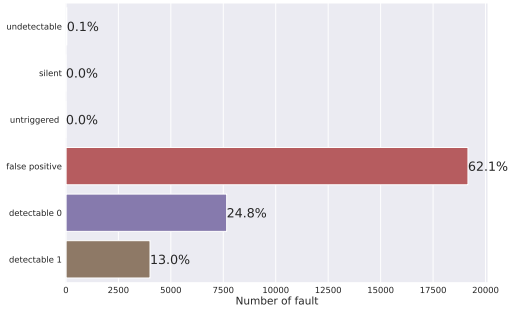

Figure 5.23: Bernstein Vazirani 4 percentage 85%.

Figure 5.24: Bernstein Vazirani, 4 main percentage graphs.

Thr %	Undet	Silent	Untr	False	Det 0	Det 1
5%	55.8%	24.0%	11.4%	6.4%	0.8%	1.6%
10%	53.0%	22.6%	9.9%	9.2%	2.3%	3.1%
15%	49.8%	20.9%	8.2%	12.4%	3.9%	4.7%
20%	48.2%	19.6%	7.3%	14.0%	5.2%	5.7%
25%	44.4%	17.4%	5.3%	17.8%	7.4%	7.7%
30%	42.5%	16.0%	4.9%	19.7%	8.8%	8.1%
35%	38.8%	12.4%	4.3%	23.4%	12.4%	8.7%
40%	36.8%	10.2%	4.1%	25.4%	14.6%	8.9%
45%	31.9%	5.4%	3.5%	30.3%	19.4%	9.4%
50%	29.8%	2.9%	3.3%	32.4%	21.9%	9.7%
55%	26.4%	0.6%	2.7%	35.8%	24.2%	10.3%
60%	24.0%	0.1%	2.3%	38.2%	24.7%	10.7%
65%	19.4%	0.0%	1.7%	42.8%	24.8%	11.3%
70%	16.3%	0.0%	1.1%	45.9%	24.8%	11.8%
75%	12.9%	0.0%	0.7%	49.3%	24.8%	12.2%
80%	8.7%	0.0%	0.3%	53.5%	24.8%	12.7%
85%	0.1%	0.0%	0.0%	62.1%	24.8%	13.0%
90%	0.0%	0.0%	0.0%	62.2%	24.8%	13.0%

Table 5.1: Bernstein Vazirani 4 percentage table.

5.4.2 Deutsch Jozsa 4

The initial circuit is shown in figure 5.25, then it is transpiled, figure 5.26 and then it is wrapped, using wrapper 1, as figure 5.27.

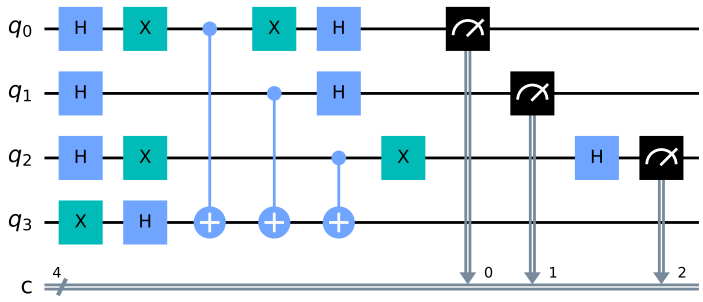

Figure 5.25: Deutsch Jozsa 4

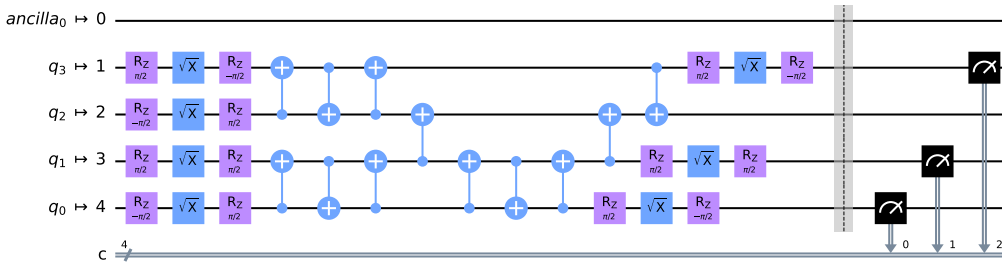

Figure 5.26: Deutsch Jozsa 4 transpiled.

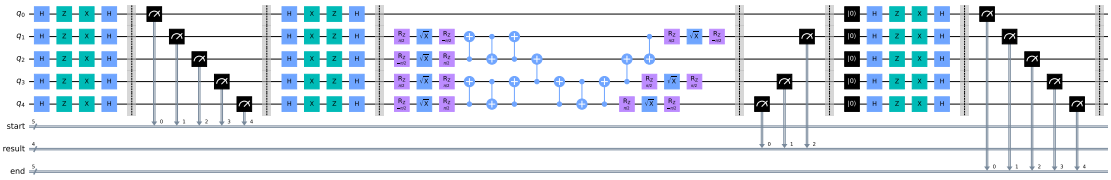

Figure 5.27: Deutsch Jozsa 4 transpiled with wrapper 1.

In the graphs in 5.31 it is possible to see, according to the different threshold barrier percentages, which are the percentages of each error class.

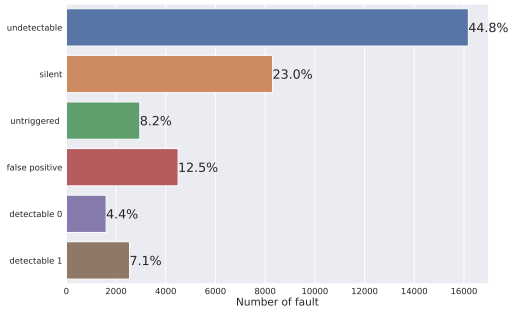

Figure 5.28: Deutsch Jozsa 4 percentage 20%.

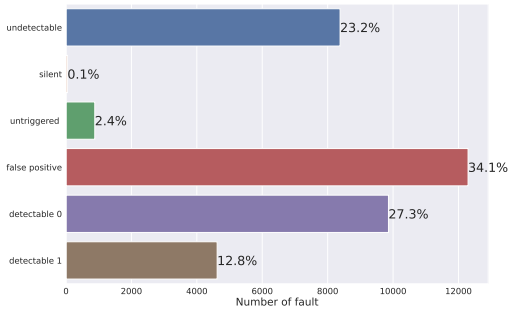

Figure 5.29: Deutsch Jozsa 4 percentage 60%.

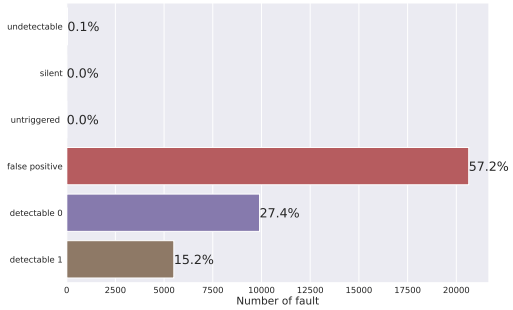

Figure 5.30: Deutsch Jozsa 4 percentage 85%.

Figure 5.31: Deutsch Jozsa, 3 main percentage graphs.

- With a threshold barrier percentage below 20%, less than 30% of the errors are detected and around 10% of the errors are false positive, while 30% of the faults not detected are or silent or untriggered errors.

- Using a threshold barrier percentage equal to 60%, almost 80% of the errors injected are detected, 34.1% of them are false positives while only 2.5% of the fault not detected are silent or untriggered errors.

- With a threshold barrier percentage beyond 85%, the percentage for TFD, all errors are detected and around 57% of them are false positive.

Thr %	Undet	Silent	Untr	False	Det 0	Det 1
5%	51.7%	26.7%	13.3%	5.6%	0.8%	1.9%
10%	49.2%	25.5%	11.5%	8.1%	2.0%	3.7%
15%	46.4%	24.1%	9.4%	10.9%	3.4%	5.8%
20%	44.8%	23.0%	8.2%	12.5%	4.4%	7.1%
25%	41.5%	21.0%	5.7%	15.8%	6.5%	9.5%
30%	39.9%	19.4%	5.2%	17.4%	8.1%	10.0%
35%	36.5%	15.2%	4.6%	20.8%	12.2%	10.6%
40%	34.8%	12.5%	4.4%	22.5%	15.0%	10.9%
45%	30.4%	6.6%	3.8%	26.9%	20.9%	11.5%
50%	28.7%	3.9%	3.5%	28.7%	23.6%	11.8%
55%	25.6%	0.9%	2.9%	31.7%	26.5%	12.3%
60%	23.2%	0.1%	2.4%	34.1%	27.3%	12.8%
65%	18.7%	0.0%	1.8%	38.6%	27.4%	13.5%
70%	15.5%	0.0%	1.3%	41.8%	27.4%	13.9%
75%	12.0%	0.0%	0.8%	45.3%	27.4%	14.4%
80%	7.9%	0.0%	0.3%	49.4%	27.4%	15.0%
85%	0.1%	0.0%	0.0%	57.2%	27.4%	15.2%
90%	0.0%	0.0%	0.0%	57.3%	27.4%	15.2%

Table 5.2: Deutsch Jozsa percentage table.

5.4.3 Grover

The initial circuit is shown in figure 5.32, then it is transpiled, figure 5.33 and in then it is wrapped, using wrapper 1, as figure 5.34.

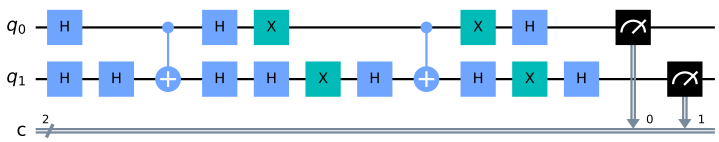

Figure 5.32: Grover quantum circuit.

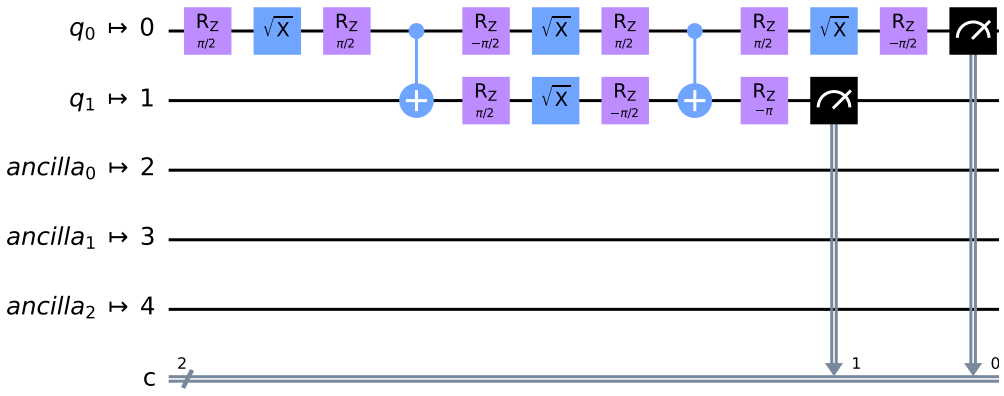

Figure 5.33: Grover transpiled.

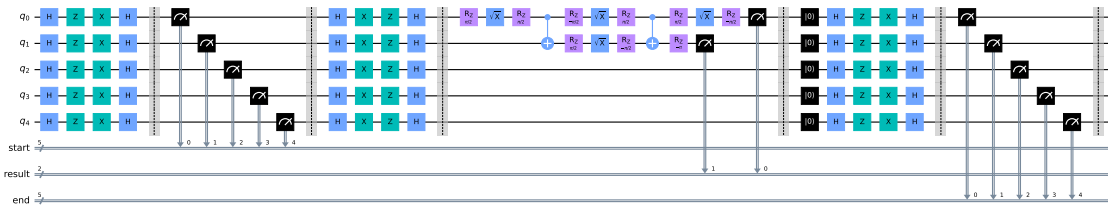

Figure 5.34: Grover transpiled with wrapper 1.

In the graphs in 5.39 it is possible to see, according to the different threshold barrier percentages, which are the percentages of each error class.

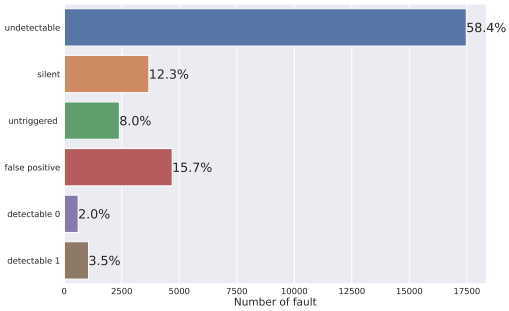

Figure 5.35: Grover percentage 15%.

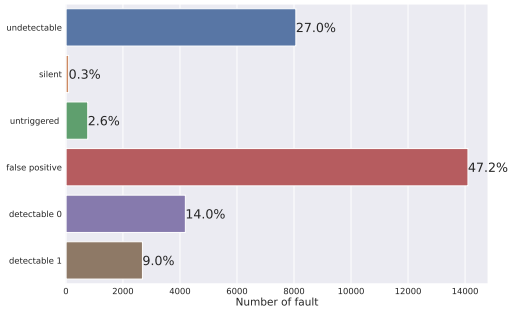

Figure 5.36: Grover percentage 55%.

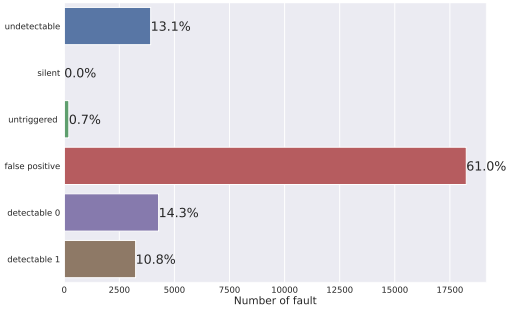

Figure 5.37: Grover percentage 75%.

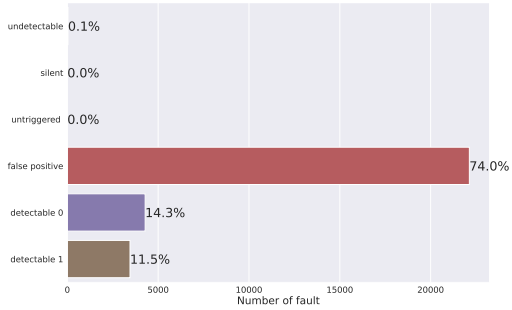

Figure 5.38: Grover percentage 85%.

Figure 5.39: Grover, 4 main percentage graph.

- With a threshold barrier percentage below 15%, less than 20% of the errors are detected and around 10% of the errors are false positive, while around 20% of the faults not detected are or silent or untriggered errors.

- With a threshold barrier percentage equal to 55%, almost 80% of the errors injected are detected, 47.2% of them are false positives while around 3% of the faults not detected are silent or Untriggered errors.

- With a threshold barrier percentage equal to 75%, silent or untriggered errors are no longer detected, but almost 61% of detected ones are false positives.

- With a threshold barrier percentage beyond 85%, all errors are detected and around 74% of them are false positives.

Thr %	Undet	Silent	Untr	False	Det 0	Det 1
5%	66.8%	13.9%	10.4%	7.4%	0.4%	1.1%
10%	62.8%	13.2%	9.3%	11.3%	1.2%	2.3%
15%	58.4%	12.3%	8.0%	15.7%	2.0%	3.5%
20%	55.8%	11.7%	7.3%	18.3%	2.7%	4.3%
25%	50.5%	10.5%	5.8%	23.6%	3.8%	5.8%
30%	48.2%	9.7%	5.3%	26.0%	4.6%	6.3%
35%	43.1%	7.6%	4.6%	31.0%	6.7%	7.0%
40%	40.4%	6.2%	4.2%	33.7%	8.1%	7.3%
45%	34.0%	3.1%	3.5%	40.1%	11.2%	8.0%
50%	31.1%	1.7%	3.1%	43.1%	12.6%	8.4%
55%	27.0%	0.3%	2.6%	47.2%	14.0%	9.0%
60%	24.2%	0.0%	2.1%	49.9%	14.3%	9.4%
65%	19.7%	0.0%	1.5%	54.5%	14.3%	10.0%
70%	16.6%	0.0%	1.1%	57.5%	14.3%	10.4%
75%	13.1%	0.0%	0.7%	61.0%	14.3%	10.8%
80%	9.0%	0.0%	0.2%	65.1%	14.3%	11.3%
85%	0.1%	0.0%	0.0%	74.0%	14.3%	11.5%
90%	0.0%	0.0%	0.0%	74.1%	14.3%	11.5%

Table 5.3: Grover percentage table

5.5 Reset injection

As explained in Google's paper [28], the presence of a transient leads to a decay of the quantum state. This means that whatever is the state of the qubit, in case of error, it will collapse to $|0\rangle$.

To better mimic this behaviour, instead of injecting a U gate with different angle combinations, a reset gate has been injected.

To better evaluate this model, for this experiment has been used the wrapper number 2, which has already been proven to work against this transient fault model.

An example of a circuit using the second type of wrapper is shown in the image 5.40.

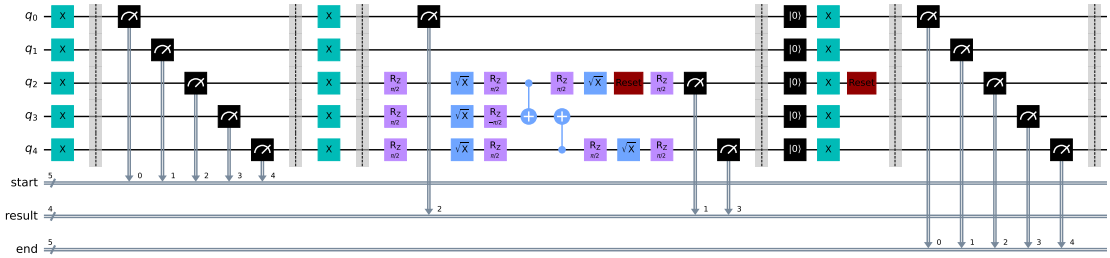

Figure 5.40: Bernstein Vazirani 4 with wrapper 2, where an error arises.

5.5.1 Bernstein Vazirani 4

The initial circuit is shown in figure 5.32, then it is transpiled, figure 5.33 and then it is wrapped, using wrapper 2, as in figure 5.41.

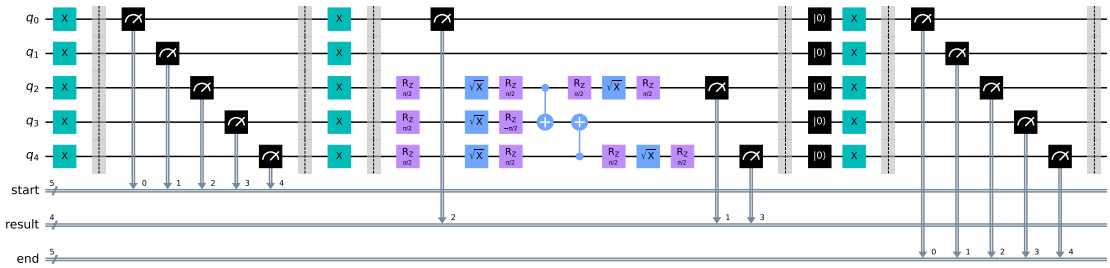

Figure 5.41: Bernstein Vazirani 4 with wrapper 2.

In the graphs in 5.44 it is possible to see, according to the different threshold barrier percentages, which are the percentages of each error class.

- With a threshold barrier percentage below 85%, 82.2% of the injected faults are detected and 51.8% of them are false positives. No silent or untriggered errors are present.

- With a threshold barrier percentage above 85%, all the injected faults are detected and 69.6% of them are false positives.

a

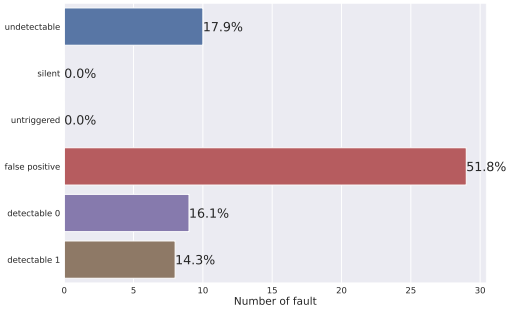

Figure 5.42: Bernstein Vazirani 4 percentage 80%.

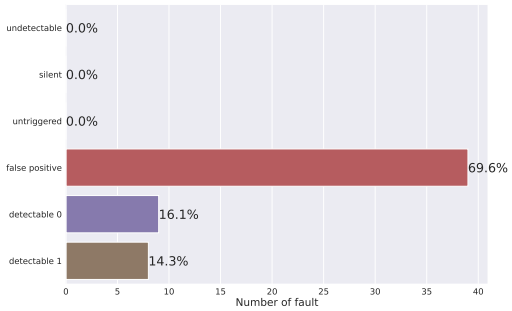

Figure 5.43: Bernstein Vazirani 4 percentage 85%.

Figure 5.44: Bernstein Vazirani, 2 main percentage graph using reset error gate.

Thr %	Undet	Silent	Untr	False	Det 0	Det 1
5%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
10%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
15%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
20%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
25%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
30%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
35%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
40%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
45%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
50%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
55%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
60%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
65%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
70%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
75%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
80%	10(17.9%)	0(0.0%)	0(0.0%)	29(51.8%)	9(16.1%)	8(14.3%)
85%	0(0.0%)	0(0.0%)	0(0.0%)	39(69.6%)	9(16.1%)	8(14.3%)
90%	0(0.0%)	0(0.0%)	0(0.0%)	39(69.6%)	9(16.1%)	8(14.3%)

Table 5.4: Bernstein Vazirani, percentage table using reset error gate.

5.5.2 Deutsch Jozsa 4

The initial circuit is shown in figure 5.25, then it is transpiled, figure 5.26 and the it is wrapped, using wrapper 1, as in figure 5.45.

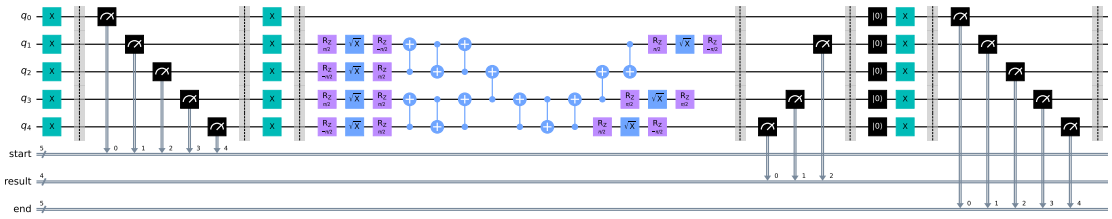

Figure 5.45: Deutsch Jozsa 4 transpiled with wrapper 2.

In the graphs in 5.48 it is possible to see, according to the different threshold barrier percentages, which are the percentages of each error class.

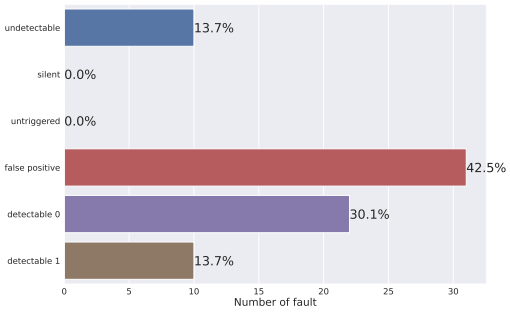

Figure 5.46: Deutsch Jozsa 4 percentage 80%.

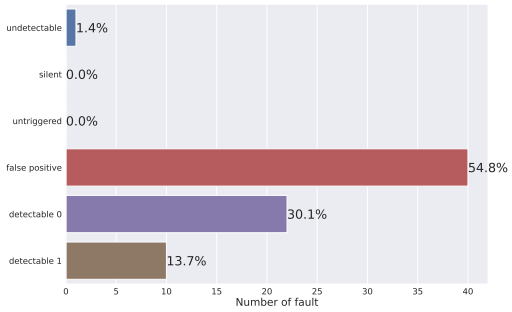

Figure 5.47: Deutsch Jozsa 4 percentage 85%.

Figure 5.48: Deutsch Jozsa 4, 2 main percentage graph using reset error gate.

- With a threshold barrier percentage below 85%, 86.3% of the injected faults are detected and around 42.4% of them are False positive. No silent or untriggered errors are detected.

- With a threshold barrier percentage above the 85%, all injected faults are detected as errors and around 55% of them are false positives.

Thr %	Undet	Silent	Untr	False	Det 0	Det 1
5%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
10%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
15%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
20%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
25%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
30%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
35%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
40%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
45%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
50%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
55%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
60%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
65%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
70%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
75%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
80%	10(13.7%)	0(0.0%)	0(0.0%)	31(42.5%)	22(30.1%)	10(13.7%)
85%	1(1.4%)	0(0.0%)	0(0.0%)	40(54.8%)	22(30.1%)	10(13.7%)
90%	0(0.0%)	0(0.0%)	0(0.0%)	41(56.2%)	22(30.1%)	10(13.7%)

Table 5.5: Deutsch Jozsa 4, percentage table using reset error gate.

5.5.3 Grove

The initial circuit is shown in figure 5.32, then it is transpiled, figure 5.33 and in the end it is wrapped, using wrapper 2, as figure 5.49.

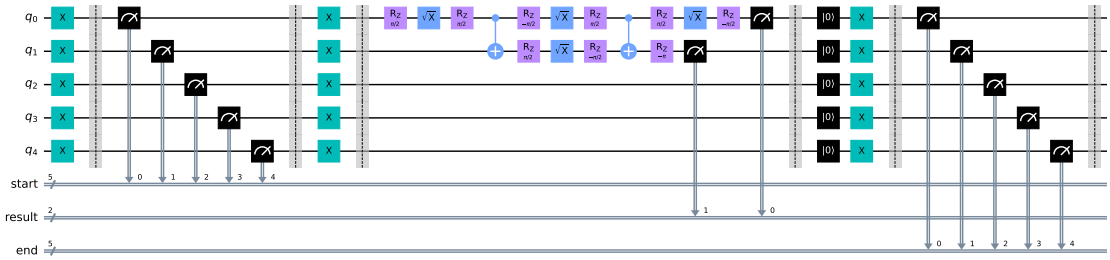

Figure 5.49: Grover with wrapper 2.

In the graphs in 5.48 it is possible to see, according to the different threshold barrier percentages, which are the percentages of each error class.

- With a threshold barrier percentage below 80%, 81.1% of the injected faults

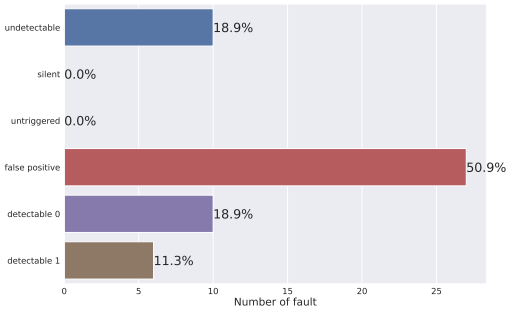

Figure 5.50: Grover percentage 75%.

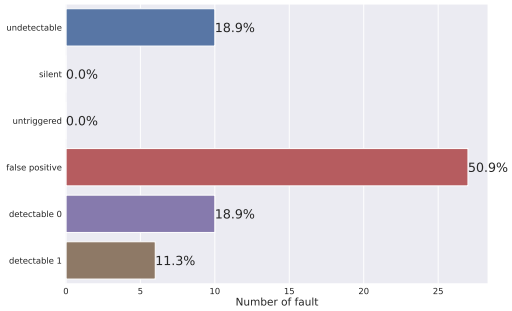

Figure 5.51: Grover percentage 80%.

Figure 5.52: Grover, 2 main percentage graph using reset error gate.

are detected as errors an around 50.9% of them are false positives. No silent or untriggered errors are presented.

- With a threshold barrier percentage above 80%, all injected faults have been detected as errors and around 70% of them are false positives.

5.6 Result analysis

For the first type of model error, the one which simulates the errors with the U gates, using threshold percentage values in the range of 40% - 60% could be a good choice. However, different percentages could be chosen to satisfy different needs.

For the second type of model error, the one which simulates the errors with the reset gates, regardless of the chosen threshold percentage, no silent or untriggered errors are identified. However, setting a threshold percentage value below around 80% means that some errors are identified as undetectable while, setting it above around 80% means that all errors are detected.

The value of the chosen threshold percentage is a critical point for the final evaluation of wrappers. A value too high means that much more errors will be labelled as false positives, while a value too low will mean that much more errors will be marked as silent or untriggered. To properly select the perfect value several simulations, with different threshold percentages, must be run and then the best value is selected, using properly criteria.

Thr %	Undet	Silent	Untr	False	Det 0	Det 1
5%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
10%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
15%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
20%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
25%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
30%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
35%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
40%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
45%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
50%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
55%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
60%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
65%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
70%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
75%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
80%	10(18.9%)	0(0.0%)	0(0.0%)	27(50.9%)	10(18.9%)	6(11.3%)
85%	1(1.9%)	0(0.0%)	0(0.0%)	36(67.9%)	10(18.9%)	6(11.3%)
90%	0(0.0%)	0(0.0%)	0(0.0%)	37(69.8%)	10(18.9%)	6(11.3%)

Table 5.6: Grover, percentage table using reset error gates.

Chapter 6

Conclusion

The TFDs work as a sensor, able to discover the presence of transient faults.

Different types of wrappers can be realized and used to test different quantum circuits. This study has been based on Sycamore quantum processor, developed by Google. Indeed, the used wrapped in this research are based on the model of Google transient fault.

On different devices, different models could appear and the effect of transient faults could be strictly dependent on the type of hardware. Probably, qubits more distance from each other could avoid a such spread out of quasiparticles in the whole device!

It is also important to consider how qubits are implemented. Today, the most used are the Transmon qubits, which are not yet an established technology that may not be present in the future.

In this thesis, different TFDs have been proposed and others could be designed to better cover different errors. The benchmarks executed in this thesis have produced interesting results. A good amount of dangerous injected faults has been discovered, even if a part of them are labelled as false positives. Even the number of silent and untriggered faults is pretty low, meaning that this technique could be an efficient mechanism against transient faults in quantum computers.

According to these results, it is possible to understand that these TFDs, which add some extra gates useless for elaboration, could be a good tool to improve the reliability of quantum computers.

6.1 Future Work

To improve the result obtained is possible to execute the following activity:

- tests the TFDs on real quantum hardware, such as Sycamore or also other systems to see if this implementation works;

- creates different types of wrappers, based on different models of correlated errors, and evaluates them all;
- test the TFDs on uncorrelated errors.

Trying to implement TFDs, to discover the presence of general noise, could be a valid solution to make quantum devices, even those that don't have a lot of physical qubits available, more reliable. Actually, there are no valid methods used to efficiently detect if transient faults occur in quantum circuits and this seems to be one of few methods, if not the only one, able to do so.

Bibliography

[1] Wikipedia contributors. *Mechanical computer — Wikipedia, The Free Encyclopedia*. [Online; accessed 12-October-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=Mechanical_computer&oldid=1112436061 (cit. on p. 1).

[2] Wikipedia contributors. *ENIAC — Wikipedia, The Free Encyclopedia*. [Online; accessed 11-October-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=ENIAC&oldid=1115292812 (cit. on p. 1).

[3] Wikipedia contributors. *UNIVAC — Wikipedia, The Free Encyclopedia*. [Online; accessed 11-October-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=UNIVAC&oldid=1111530235 (cit. on p. 1).

[4] Daniel Etiemble. *Technologies and Computing Paradigms: Beyond Moore's law?* 2022. DOI: 10.48550/ARXIV.2206.03201. URL: https://arxiv.org/abs/2206.03201 (cit. on p. 2).

[5] Suhas Kumar. *Fundamental Limits to Moore's Law*. 2015. DOI: 10.48550/ARXIV.1511.05956. URL: https://arxiv.org/abs/1511.05956 (cit. on p. 2).

[6] Mohamed Warda and Khodr Badih. *Graphene Field Effect Transistors: A Review*. 2020. DOI: 10.48550/ARXIV.2010.10382. URL: https://arxiv.org/abs/2010.10382 (cit. on p. 2).

[7] Wikimedia Commons. *File:Moore's Law Transistor Count 1970-2020.png — Wikimedia Commons, the free media repository*. [Online; accessed 11-October-2022]. 2022. URL: https://commons.wikimedia.org/w/index.php?title=File:Moore%27s_Law_Transistor_Count_1970-2020.png&oldid=691059914 (cit. on p. 3).

[8] Peter W. Shor. «Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer». In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. DOI: 10.1137/s0097539795293172. URL: https://doi.org/10.1137%2Fs0097539795293172 (cit. on p. 2).

[9] Stephen P. Jordan. *Quantum Computation Beyond the Circuit Model*. 2008. DOI: 10.48550/ARXIV.0809.2307. URL: https://arxiv.org/abs/0809.2307 (cit. on p. 3).

[10] Dik Bouwmeester, Jian-Wei Pan, Klaus Mattle, Manfred Eibl, Harald Weinfurter, and Anton Zeilinger. «Experimental quantum teleportation». In: *Nature* 390.6660 (Dec. 1997), pp. 575–579. DOI: 10.1038/37539. URL: https://doi.org/10.1038%2F37539 (cit. on p. 17).

[11] Richard Jozsa. *A stronger no-cloning theorem*. 2002. DOI: 10.48550/ARXIV.QUANT-PH/0204153. URL: https://arxiv.org/abs/quant-ph/0204153 (cit. on p. 18).

[12] «371Bibliography». In: *Niels Bohr and the Quantum Atom: The Bohr Model of Atomic Structure 1913–1925*. Oxford University Press, May 2012. ISBN: 9780199654987. DOI: 10.1093/acprof:oso/9780199654987.001.0001. eprint: https://academic.oup.com/book/0/chapter/149026690/chapter-ag-pdf/45509674/book_5807_section_149026690.ag.pdf (cit. on p. 22).

[13] Morten Kjaergaard, Mollie E. Schwartz, Jochen Braumüller, Philip Krantz, Joel I.-J. Wang, Simon Gustavsson, and William D. Oliver. «Superconducting Qubits: Current State of Play». In: *Annual Review of Condensed Matter Physics* 11.1 (Mar. 2020), pp. 369–395. DOI: 10.1146/annurev-conmatphys-031119-050605. URL: https://doi.org/10.1146%2Fannurev-conmatphys-031119-050605 (cit. on p. 24).

[14] Michael Tinkham. *Introduction to superconductivity*. Courier Corporation, 2004 (cit. on p. 24).

[15] M. H. Devoret, A. Wallraff, and J. M. Martinis. *Superconducting Qubits: A Short Review*. 2004. DOI: 10.48550/ARXIV.COND-MAT/0411174. URL: https://arxiv.org/abs/cond-mat/0411174 (cit. on p. 24).

[16] Frank Arute et al. «Quantum supremacy using a programmable superconducting processor». In: *Nature* 574.7779 (Oct. 2019), pp. 505–510. DOI: 10.1038/s41586-019-1666-5. URL: https://doi.org/10.1038%2Fs41586-019-1666-5 (cit. on pp. 24, 37).

[17] David W Dreisigmeyer and Peter M Young. «Nonconservative Lagrangian mechanics: a generalized function approach». In: *Journal of Physics A: Mathematical and General* 36.30 (July 2003), pp. 8297–8310. DOI: 10.1088/0305-4470/36/30/307. URL: https://doi.org/10.1088%2F0305-4470%2F36%2F30%2F307 (cit. on p. 27).

[18] Dominic Branford, Oscar C. O. Dahlsten, and Andrew J. P. Garner. «On Defining the Hamiltonian Beyond Quantum Theory». In: *Foundations of Physics* 48.8 (Aug. 2018), pp. 982–1006. DOI: 10.1007/s10701-018-0205-9. URL: https://doi.org/10.1007%2Fs10701-018-0205-9 (cit. on p. 27).

[19] François Mallet, Florian R. Ong, Agustin Palacios-Laloy, François Nguyen, Patrice Bertet, Denis Vion, and Daniel Esteve. «Single-shot qubit readout in circuit quantum electrodynamics». In: *Nature Physics* 5.11 (Sept. 2009), pp. 791–795. DOI: 10.1038/nphys1400. URL: https://doi.org/10.1038%2Fnphys1400 (cit. on p. 30).

[20] C. Anastopoulos. In: *Foundations of Physics* 31.11 (2001), pp. 1545–1580. DOI: 10.1023/a:1012690715414. URL: https://doi.org/10.1023%2Fa%3A1012690715414 (cit. on p. 31).

[21] James R. Wootton and Daniel Loss. «Repetition code of 15 qubits». In: *Physical Review A* 97.5 (May 2018). DOI: 10.1103/physreva.97.052313. URL: https://doi.org/10.1103%2Fphysreva.97.052313 (cit. on p. 34).

[22] Daniel Gottesman. *An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation*. 2009. DOI: 10.48550/ARXIV.0904.2557. URL: https://arxiv.org/abs/0904.2557 (cit. on p. 36).

[23] Samudra Dasgupta. *Adaptive channel estimation for mitigating circuits executed on noisy quantum devices*. 2022. DOI: 10.48550/ARXIV.2208.10263. URL: https://arxiv.org/abs/2208.10263 (cit. on p. 36).

[24] Sebastian Krinner et al. «Realizing repeated quantum error correction in a distance-three surface code». In: *Nature* 605.7911 (May 2022), pp. 669–674. DOI: 10.1038/s41586-022-04566-8. URL: https://doi.org/10.1038%2Fs41586-022-04566-8 (cit. on p. 36).

[25] Youwei Zhao et al. «Realization of an Error-Correcting Surface Code with Superconducting Qubits». In: *Physical Review Letters* 129.3 (July 2022). DOI: 10.1103/physrevlett.129.030501. URL: https://doi.org/10.1103%2Fphysrevlett.129.030501 (cit. on p. 36).

[26] Won Young Hwang, Doyeol (David) Ahn, and Sung Woo Hwang. «Correlated errors in quantum-error corrections». In: *Physical Review A* 63.2 (Jan. 2001). DOI: 10.1103/physreva.63.022303. URL: https://doi.org/10.1103%2Fphysreva.63.022303 (cit. on p. 37).

[27] Austin G. Fowler and John M. Martinis. «Quantifying the effects of local many-qubit errors and nonlocal two-qubit errors on the surface code». In: *Physical Review A* 89.3 (Mar. 2014). DOI: 10.1103/physreva.89.032316. URL: https://doi.org/10.1103%2Fphysreva.89.032316 (cit. on p. 37).

[28] Matt McEwen et al. «Resolving catastrophic error bursts from cosmic rays in large arrays of superconducting qubits». In: *Nature Physics* 18.1 (Dec. 2021), pp. 107–111. DOI: 10.1038/s41567-021-01432-8. URL: https://doi.org/10.1038%2Fs41567-021-01432-8 (cit. on pp. 37, 39, 48, 52, 53, 67).

[29] Wikipedia contributors. *Cosmic ray — Wikipedia, The Free Encyclopedia*. [Online; accessed 1-October-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=Cosmic_ray&oldid=1110136323 (cit. on p. 37).

[30] J. F. Ziegler. «Terrestrial cosmic rays». In: *IBM Journal of Research and Development* 40.1 (1996), pp. 19–39. DOI: 10.1147/rd.401.0019 (cit. on p. 37).

[31] D. J. Bird et al. «Detection of a cosmic ray with measured energy well beyond the expected spectral cutoff due to cosmic microwave radiation». In: *The Astrophysical Journal* 441 (Mar. 1995), p. 144. DOI: 10.1086/175344. URL: https://doi.org/10.1086%2F175344 (cit. on p. 37).

[32] R. Baumann. «Soft errors in advanced computer systems». In: *IEEE Design Test of Computers* 22.3 (2005), pp. 258–266. DOI: 10.1109/MDT.2005.69 (cit. on p. 37).

[33] Antti P. Vepsäläinen et al. «Impact of ionizing radiation on superconducting qubit coherence». In: *Nature* 584.7822 (Aug. 2020), pp. 551–556. DOI: 10.1038/s41586-020-2619-8. URL: https://doi.org/10.1038%2Fs41586-020-2619-8 (cit. on p. 37).

[34] C. D. Wilen et al. «Correlated charge noise and relaxation errors in superconducting qubits». In: *Nature* 594.7863 (June 2021), pp. 369–373. DOI: 10.1038/s41586-021-03557-5. URL: https://doi.org/10.1038%2Fs41586-021-03557-5 (cit. on p. 37).

[35] John M. Martinis. *Saving superconducting quantum processors from qubit decay and correlated errors generated by gamma and cosmic rays*. 2020. DOI: 10.48550/ARXIV.2012.06137. URL: https://arxiv.org/abs/2012.06137 (cit. on p. 37).

[36] John M. Martinis, M. Ansmann, and J. Aumentado. *Energy Decay in Josephson Qubits from Non-equilibrium Quasiparticles*. 2009. DOI: 10.48550/ARXIV.0904.2171. URL: https://arxiv.org/abs/0904.2171 (cit. on p. 37).

[37] Chen Wang et al. «Measurement and control of quasiparticle dynamics in a superconducting qubit». In: *Nature communications* 5.1 (2014), pp. 1–7 (cit. on p. 37).

[38] Gianluigi Catelani, Jens Koch, Luigi Frunzio, RJ Schoelkopf, Michel H Devoret, and LI Glazman. «Quasiparticle relaxation of superconducting qubits in the presence of flux». In: *Physical review letters* 106.7 (2011), p. 077002 (cit. on p. 37).

[39] K Serniak, M Hays, G De Lange, S Diamond, Sh Shankar, LD Burkhart, L Frunzio, M Houzet, and MH Devoret. «Hot nonequilibrium quasiparticles in transmon qubits». In: *Physical review letters* 121.15 (2018), p. 157701 (cit. on p. 37).

[40] M Lenander et al. «Measurement of energy decay in superconducting qubits from nonequilibrium quasiparticles». In: *Physical Review B* 84.2 (2011), p. 024501 (cit. on p. 37).

[41] Daniel Oliveira, Edoardo Giusto, Emanuele Dri, Nadir Casciola, Betis Baheri, Qiang Guan, Bartolomeo Montrucchio, and Paolo Rech. *QuFI: a Quantum Fault Injector to Measure the Reliability of Qubits and Quantum Circuits*. 2022. DOI: 10.48550/ARXIV.2203.07183. URL: https://arxiv.org/abs/2203.07183 (cit. on pp. 37, 39, 48, 49).

[42] L. Cardani et al. «Reducing the impact of radioactivity on quantum circuits in a deep-underground facility». In: *Nature Communications* 12.1 (May 2021). DOI: 10.1038/s41467-021-23032-z. URL: https://doi.org/10.1038%2Fs41467-021-23032-z (cit. on p. 37).

[43] Wikipedia contributors. *Contrast (vision) — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Contrast_(vision)&oldid=1086867694. [Online; accessed 1-October-2022]. 2022 (cit. on p. 40).

[44] Daniel Oliveira, Edoardo Giusto, Betis Baheri, Qiang Guan, Bartolomeo Montrucchio, and Paolo Rech. *A Systematic Methodology to Compute the Quantum Vulnerability Factors for Quantum Circuits*. 2021. DOI: 10.48550/ARXIV.2111.07085. URL: https://arxiv.org/abs/2111.07085 (cit. on p. 41).

[45] John Preskill. «Quantum Computing in the NISQ era and beyond». In: *Quantum* 2 (Aug. 2018), p. 79. DOI: 10.22331/q-2018-08-06-79. URL: https://doi.org/10.22331%2Fq-2018-08-06-79 (cit. on p. 43).

[46] Katsuya Yonehara. «Radiation hardened beam instrumentations for multi-Mega-Watt beam facilities». In: *arXiv preprint arXiv:2203.06024* (2022) (cit. on p. 44).

[47] Di Chen, Jiankun Li, Zheng Wei, Xinjian Wei, Maguang Zhu, Jing Liu, Guangyu Zhang, Zhiyong Zhang, and Jian-Hao Chen. «Radiation-hardened and Repairable MoS _2 Field Effect Devices with Polymer Solid Electrolyte Gates». In: *arXiv preprint arXiv:2110.02601* (2021) (cit. on p. 44).

[48] Wikipedia contributors. *New Horizons — Wikipedia, The Free Encyclopedia*. [Online; accessed 16-October-2022]. 2022. URL: https://en.wikipedia.org/w/index.php?title=New_Horizons&oldid=1115707678 (cit. on p. 45).

[49] B. Baykant Alagoz. «Hierarchical Triple-Modular Redundancy (H-TMR) Network For Digital Systems». In: (2009). DOI: 10.48550/ARXIV.0902.0241. URL: https://arxiv.org/abs/0902.0241 (cit. on p. 45).

[50] Antonio D Córcoles, Jerry M Chow, Jay M Gambetta, Chad Rigetti, James R Rozen, George A Keefe, Mary Beth Rothwell, Mark B Ketchen, and Matthias Steffen. «Protecting superconducting qubits from radiation». In: *Applied Physics Letters* 99.18 (2011), p. 181906 (cit. on p. 47).

[51] Pengfei Wang et al. «Single ion qubit with estimated coherence time exceeding one hour». In: *Nature Communications* 12.1 (Jan. 2021). DOI: 10.1038/s41467-020-20330-w. URL: https://doi.org/10.1038%2Fs41467-020-20330-w (cit. on p. 47).