UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Fisica e Astronomia "Galileo Galilei"

Corso di Laurea in Fisica

Tesi di Laurea

Quantum computing on the IBM processor

Relatore

Prof. Simone Montangero

Laureando

Marco Ballarin

Anno Accademico 2018/2019

To my parents.

Contents

Abstract

We analyze the basic aspects of the quantum information theory, namely the quantum version of the theory that is behind all the classical implementations like computers and communications. We focus on quantum computation, defining its fundamental unit: the qubit, namely a quantum two-level system that will be described in detail in this thesis. We then concentrate on the possible transformations gates that can be applied to qubits, both unitary and non-unitary ones.

The challenges in the construction of qubits are impressive: in particular, it is difficult to isolate the system from the environment. This interaction is modeled by non-unitary transformations. The characteristic times connected to these processes are the relaxation time, the time in which a state decays in another state, and the decoherence time, the time in which quantum coherence is lost. These times are fundamental in quantum computing since they quantify how many operations can be performed on qubits and still obtain reliable results. We measure these characteristic times on the IBM quantum processor 'ibmq_16_melbourne' performing three different experiments: one for the relaxation time and two for the decoherence time, namely Ramsey and Echo experiments.

Finally, we implement a quantum algorithm, an algorithm that uses the resources of quantum mechanics, like entanglement. We focus on finding a protocol to define the quantum version of the elementary cellular automata, the quantum elementary cellular automata, and run it on the IBM processor. They are dynamical systems defined on a lattice in which the evolution is defined using simple local update rules.

Comparing the physical results with theoretical expectations and noise-affected simulations we show the performance of the IBM processor.

Introduction

Quantum mechanics revolutionized every aspect of physics, and in particular information theory[1]. The products of classical information theory permeate every-day life: communications, computers, cryptography. A quantum version of these classical implementations exists, but it is really difficult to develop experimentally. The improvement that could be obtained by implementing quantum technology is great. It is simple to understand its possible impact in scientific research: a good example is the quantum algorithm developed to find the eigenvalues of a given Hamiltonian, the variational eigensolver[2].

We will focus on the area of the Quantum Information Theory that deals with Quantum Computers, the quantum version of the classical computers. Their fundamental unit is the qubit, a quantum two-level system. The research in this field obtained a great boost in 1994 when Shor demonstrated the existence of a polynomial quantum algorithm for the factorization of prime numbers[3]. The importance of this demonstration is that the difficulty of factorizing the product of prime numbers with lots of digits that is at the base of the security of one of the most used cryptography protocols, the RSA[4]. Indeed, in the public key distributed freely and used to encode the message there is the number N, which is the product of two prime numbers. The private key, namely the key used to decode the message, is easily obtained by the public key and the two factors on N.

One of the most advanced companies in this topic is IBM, namely the "International Business Machine Corporation". IBM quantum computers, like any other quantum computer currently available, are noisy and have a limited number of qubits: at the moment we achieved at most around 50 qubits. IBM makes quantum computers available on the cloud so that all the scientific community could run experiments on the real quantum computer from everywhere in the world.

The purpose of this thesis is to characterize the *ibmq_16_melbourne*, the IBM processor with 14 qubits by measuring the relaxation time, namely the time of decay of the excited state of qubits, and the decoherence time, the time in which quantum coherences are lost. We will understand the current limit of this particular machine and the main source of errors and noise that afflict the computer. Then we will focus on the quantization of one interesting classical application of computers: cellular automata. From simple starting condition

and simple evolutions, they create complex structures and are used in various areas for simulations. Their quantum version should be even more interesting: the simplest elementary systems of our world are quantum, so from the evolutions of even the simplest quantum automata we could learn fascinating developments. Using the results obtained in the characterization we will try to understand if it is possible to run a quantum cellular automaton on the IBM quantum computer, or if the various noise sources makes the experiment meaningless.

In the first chapter, we introduce quantum computation, showing how its fundamental ingredients are related to the classical one. We define in detail the fundamental unit of quantum computation, the qubit, and how to handle many-qubits systems. We then focus on the evolution of quantum systems with particular attention to the distinction of the unitary evolution, through quantum gates, and the non-unitary evolution, through quantum channels. We will stretch the importance of quantum channels, introducing them rigorously.

In the second chapter, we present the IBM quantum processor, describing qualitatively the physical implementation of the IBM qubit, an improvement of the charge qubit, made using a superconducting circuit. We will then define all the important quantities that characterize the ibmq_16_melbourne quantum processor. They are the coupling map, namely the possible two-qubits connections, the set of fundamental quantum gates, their application times and the qubit gate error, namely the probability that a gate applied to a determined qubit fails. In the end, we will introduce the programming language used to communicate to the quantum processor, showing some of its basic commands and two examples of code.

In the third chapter, we measure the characteristic qubits times through experiments run on the real IBM processor. We will see if they follow the theory introduced in Chapter One. In the end, we will compare the results obtained in the experiments to the values presented by IBM on their site.

In the fourth chapter, we introduce the concept of cellular automata, namely systems defined on a lattice which evolve using simple local update rules. We then introduce their quantum counterpart, defining quantum cellular automata, and we focus on the simplest of those systems: the quantum elementary cellular automata. In the end, we run the automata on the IBM processor, comparing the experimental results with theoretical expectations and noise-affected simulations.

We can't solve problems by using the same kind of thinking
we used when we created them.

Albert Einstein

1

An introduction to Quantum Information

In this chapter, we introduce quantum information, showing how its fundamental ingredients are related to their classical counterparts. First, we recall the difference between pure and mixed states in quantum mechanics[5]; in particular, we focus on the qubit, a two-level system, which constitutes the fundamental information unit in a quantum computer. Then we show how quantum states can be transformed: this is done via quantum gates, namely particular unitary evolution protocols, or via non-unitary transformations called quantum channels.

1.1 Classical Computation in a nutshell

Calculators, mobile phones, computers and lots of other electronic devices are based on classical information theory and computation.

The fundamental unit of classical computation is the bit, a variable which can assume only two values: $\{0, 1\}$. Every task of a classical computer is translated, in fact, into operations, called gates, applied to bits: these operations are modeled by binary functions $f : \{0, 1\}^n \to \{0, 1\}$, which take the state of n bits as an input and return 0 or 1 as an output.

Computers cannot directly implement all possible functions f: complex operations need to be decomposed in simpler operations, the ones implemented in the construction of the machine. This is possible since it has been demonstrated that each binary function introduced above is modular in a set of elementary logic gates, namely can be obtained through a

proper sequence of fundamental gates.

In the following we list some examples of logic gates, distinguishing between those which take one or two bits as an input, but we need first to understand how the results of a gate are shown, namely how to read a truth table. We consider a Table: in the left n columns are listed all the possible input configurations: we obtain 2^n combination of 0 and 1, since the gates are modeled by binary functions. In the right columns instead are listed the outputs: they are generated by applying the gate to its relative input state, namely the one on the same row. We see some examples of truth tables in Table 1.1. Now that we know how to read a truth table we can list some example:

- Gates that takes 1 bit as an input (Truth table is in Table 2.1a):
  identity, which do nothing on the bit.
  negation(NOT), which flips the bit state and turns 0 into 1 and vice-versa;

- Gates that takes 2 bits as an input (Truth table is in table 2.1b):
  AND, which gives 1 if and only if both bits are in 1;
  OR, which gives 1 if at least a bit is in 1;
  XOR, which gives 1 if a bit is in 1 and the other in 0;
  COPY, which copy the state of the bit on another bit;
  SWAP, which swaps the state of the bit a with the state of the bit b.

The only gates needed to codify all the f described above are AND, OR, NOT, COPY, and this are called universal gates[6].
Now that we now the fundamentals of classical computation we are ready to go on and analyze its quantum counterpart.

a	$\mathbb{I}$	$\bar{a}$
0	0	1
1	1	0

(a) Truth table of gates that takes 1 bit as an input.

a	b	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

(b) Truth table of gates that takes 1 bit as an input.

Table 1.1: Truth tables of gates. On the first row we have the name of the bit and of the gate applied and on the right columns the input state. On the right columns we have the outputs relative to the input on the same row after the gate on the column's top has been applied.

In order to perform a quantum counterpart of classical computation first of all we need to define a quantum fundamental information unit. To this purpose we introduce the quantum version of a bit, the qubit. It is a two level system and, due to its quantum nature, it does not admit only the states $\{|0\rangle, |1\rangle\}$, but each linear combination of these two, as shown in Equation 1.1.

There are several physical two-level system that can be modeled by a qubit: an example is the spin state of a particle, that can be 1, -1 or each linear combination of these states. The most general state can be written as:

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \qquad \text{with} \qquad |\alpha_0|^2 + |\alpha_1|^2 = 1, \quad \alpha_0, \alpha_1 \in \mathbb{C}. \tag{1.1}$$

A way to represent single-qubit states is the Bloch sphere (Figure 1.1). It is a 2-dimensional

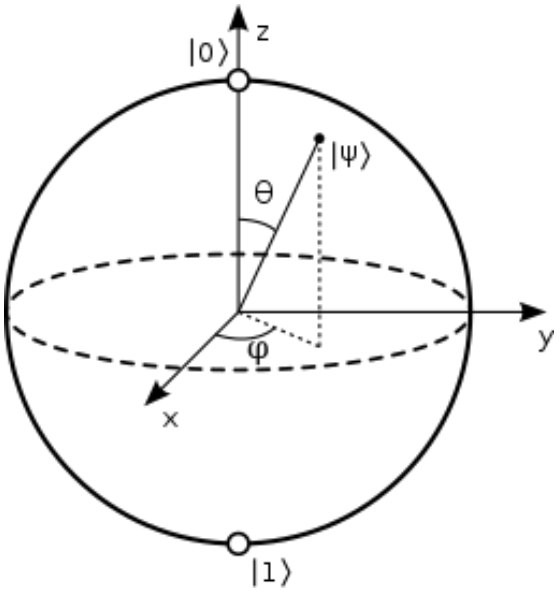


Figure 1.1: The Bloch Sphere. It is a sphere with the radius bound to one. The angles θ and ϕ identify every pure state as shown in Equation 1.2. The $\hat{z}$ unit vector represent the $|0\rangle$ state and $-\hat{z}$ the $|1\rangle$ state. Image from: `https://upload.wikimedia.org/wikipedia/commons/6/6b/Bloch_sphere.svg`.

manyfold, a sphere with the radius bound to be one, immersed in a 3-dimensional space. Each point on the surface of the sphere is a state and can be described by using the angles θ

and phi with the boundaries $0 \leq \theta \leq \pi$ and $0 \leq \phi \leq 2\pi$ as shown by the equation:

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + \sin\frac{\theta}{2}e^{i\phi}|1\rangle . \tag{1.2}$$

A state ψ can be described also by introducing the density matrix, namely:

$$\rho(\theta,\phi) = |\psi\rangle\langle\psi| = \begin{pmatrix} \cos^2(\theta/2) & \cos(\theta/2)\sin(\theta/2)e^{-i\phi} \\ \cos(\theta/2)\sin(\theta/2)e^{i\phi} & \sin^2(\theta/2) \end{pmatrix}. \tag{1.3}$$

It has some important properties, but to enunciate them we need to introduce the trace of an arbitrary matrix A as $\mathrm{Tr}(A) = \sum_k^n \langle k|A|k\rangle$ with $\{|k\rangle\}_{k=1,\cdots,n}$ is a basis of the Hilbert space.
These properties are:

1. ρ is hermitian;
2. ρ is a non-negative operator: $\forall\,|\psi\rangle \qquad \langle\psi|\rho|\psi\rangle \geq 0$;
3. $Tr(\rho) = 1$. This property is really important: it ensures that the sum of the probabilities over a set of basis vectors is equal to one.

If now we use the spherical coordinates $\{x = \sin\theta\cos\phi, y = \sin\theta\sin\phi, z = \cos\theta\}$ and the basis formed by the Pauli matrices and the identity matrix, we can write the density matrix as:

$$\rho = \frac{1}{2}(\mathbb{I} + x\hat{\sigma_x} + y\hat{\sigma_y} + z\hat{\sigma_z}) = \frac{1}{2}\begin{pmatrix} 1+z & x-iy \\ x+iy & 1-z \end{pmatrix}, \tag{1.4}$$

where $\hat{\sigma_x} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, $\hat{\sigma_y} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$, $\hat{\sigma_z} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$ are the Pauli matrixes.

It is important to stress that a qubit can encode more info then a classical bit: despite the fact that, after a measurement, an arbitrary state $|\psi\rangle$ collapses in either $|0\rangle$ or $|1\rangle$, as in the classical case. We can however use quantum mechanics during the calculus: the possibilities of using the coherent superposition of quantum states gives us the ability to implement protocols impossible with the classical computation, as Grover [7] or Shor [3] algorithms.
The qubit analyzed in this section is an example of a isolated quantum state: in the next section we extend the definition of quantum states to the case in which they are coupled to an external environment.

The state $|\psi\rangle$ defined in the previous section is completely characterized and it is said to be a pure state: as shown before it is represented by a ray vector in a Hilbert space over the complex numbers with unitary length.

In some cases nevertheless, we do not have access to all the state, but only a portion of it; Our state is not therefor completely characterized and it is called mixed state. An example when is the state interacts with the environment, where the environment is meant to be everything outside our system and that we cannot control.

A state ρ is mixed if we only know the probability p_i associated to each state $|\psi_i\rangle$. Its density matrix ρ has the form:

$$\rho = \sum_i p_i \left|\psi_i\right\rangle \left\langle\psi_i\right| . \tag{1.5}$$

It is possible to describe a pure state with a mixed state: it is the particular case when there is a basis such that $p_i = 0 \quad \forall i \neq \tilde{i}$ and $p_i = 1$ for $i = \tilde{i}$, so $\rho = |\psi\rangle \langle\psi|$. The possibility of describing a pure state with a mixed state makes important to know how to distinguish between mixed and pure states once we have access to its density matrix. For example we have that:

- A state described by a density matrix is pure if it is invariant by squaring $\rho^2 = \rho$;
- A state described by a density matrix is pure if the trace of ρ^2 is $Tr(\rho^2) = 1$ and mixed if $\mathrm{Tr}(\rho^2) < 1$.

As an example we will show how to quantify the purity of a qubit state given a ρ in the form of Equation 1.4. By taking the determinant of the matrix in Equation 1.4 and remembering that in polar coordinates $r^2 = x^2 + y^2 + z^2$ we obtain: $\det(\rho) = \frac{1}{4}(1 - |r|^2)$. By recalling that a pure state has a probability $p_1 = 1$ and $p_2 = 0$, we deduce $det(\rho) = p_1 p_2 = 0$, therefore for a pure state $|r| = 1$.

A mixed state has instead $|r| \leq 1$. As an example we analyze the so called maximally mixed state: the one in which each state is equiprobable:

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

In this case $\det(\rho) = \frac{1}{4}$, that means $\vec{r} = (0, 0, 0)$, the origin of the axis and the most distant from the surface of the sphere.

Finally, we enunciate how it is defined the expectation value of an observable $\hat{A}$ relative a state ρ:

$$\langle A \rangle_\rho = \mathrm{Tr}\Big(\rho \hat{A}\Big). \tag{1.6}$$

1.4 Composite systems and entanglement

We illustrate now some properties of quantum-manybody systems which will be necessary to describe systems made up of many qubits interacting among each other or with the environment. A quantum many body system lives in an Hilbert space which is given by the tensor product of the Hilbert spaces of the singles bodies $\mathcal{H}_i$, so for a system of n interacting objects we have:

$$\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \cdots \otimes \mathcal{H}_n. \tag{1.7}$$

An arbitrary pure state $|\psi\rangle \in \mathcal{H}$ can be written using a linear combination of the tensor product of the orthonormal basis of $\{\mathcal{H}_i\}_{i=1,\cdots,n}$, namely $\{|j\rangle_i\}_{i=1,\cdots,n}$ into the form:

$$|\psi\rangle = \sum_{\vec{j}} c_{\vec{j}} \, |j\rangle_1 \, |j\rangle_2 \cdots |j\rangle_n \,, \qquad \text{with } \sum_{\vec{j}} \left| c_{\vec{j}} \right|^2 = 1. \tag{1.8}$$

According to the definition of density matrix given in Equation 1.5, for a general state in $\mathcal{H}$ we have:

$$\begin{gathered} \rho = |\psi\rangle \langle\psi| = \textstyle\sum_{\vec{j}} \sum_{\vec{k}} c_{\vec{j}} c^*_{\vec{k}} \, |j\rangle_1 \cdots |j\rangle_n \langle k|_1 \cdots \langle k|_n \Rightarrow \\ \rho = \textstyle\sum_{\vec{j}} \sum_{\vec{k}} \rho_{\vec{k}\vec{j}} \, |j_1 \cdots j_n\rangle \langle k_1 \cdots k_n| \,. \end{gathered} \tag{1.9}$$

We move into the computation of observables expectation value in quantum many body systems. In particular, we focus on local observables represented by operators that can be written as $\hat{A}_{TOT} = (\mathbb{I} \otimes \cdots \otimes \hat{A} \otimes \cdots \otimes \mathbb{I})$. According to the definition we may take the trace $\mathrm{Tr}\Big(\rho \hat{A}_{TOT}\Big)$, but there is a much simpler method. To this purpose we define the *reduced density matrix* relative to the i-th body as the trace over the $h \neq i$-th spaces of the entire density matrix:

$$\rho_i = \mathrm{Tr}_{h \neq i}(\rho). \tag{1.10}$$

It can be shown that $\mathrm{Tr}\Big(\rho_i \hat{A}\Big)$ coincides with the expectation value $\mathrm{Tr}\Big(\rho \hat{A_{TOT}}\Big)$. It is important to observe that ρ_i is a density matrix if ρ is a density matrix itself, but it doesn't

conserve purity: if ρ is pure, then ρ_i can be mixed.

Let us define now *entanglement*, one of the most powerful resources in quantum computation. A quantum state that is classically correlated is defined separable. A separable state can be written as the tensor product of the single states, so that perform an operation on a state, like a measurement, doesn't influence the other. An example with a 2-qubit state is $|00\rangle = |0\rangle \otimes |0\rangle$. The definition above can be generalized to any number of states as:

$$\rho_{AB}^{S} = \sum_{i} p_i (\rho_i^A \otimes \rho_i^B). \tag{1.11}$$

A state that cannot be written as a separable state is entangled and an example of a 2-qubits entangled state is the Bell couple: $|\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. An entangled state has no classical comparison and it is one of the most important and innovative concepts in quantum mechanics: an example of how entanglement can be quantified is the violation of the Bell inequalities, related to the existence of a set of local hidden variables which describe quantum mechanics [8]. The evidence that an entangled state is the only one able to violate the Bell inequalities lends to the solution of the EPR paradox [9] and the incompatibility of locality and realism in quantum mechanics.

1.5 State Evolution

Now that we know how to write the state of a many-qubit system, we want to understand how to compute their time evolution. The evolution of pure states is obtained by solving the Schrödinger differential equation:

$$i\hbar \frac{\partial \psi}{\partial t} = \hat{H}\psi, \tag{1.12}$$

where $\hat{H}$ is the Hamiltonian of the system. We define the eigenfunction $|n\rangle$ and the eigenvalue ϵ_n of the operator H as those that respects the following equation, called the *Schrödinger eigenvalue equation*:

$$H|n\rangle = \epsilon_n |n\rangle . \tag{1.13}$$

The solution of Eq 1.12 for the eigenfunction of H gives the state $|n(t)\rangle = U(t)|n\rangle$ where we define the time evolution operator as $U(t) = e^{-\frac{i}{\hbar}\epsilon_n t}$. If we use the basis formed by the

eigenfunction of H $\{|n\rangle\}$ to write the general state we obtain, at time t, the state:

$$|\psi(t)\rangle = \sum_n e^{-\frac{i}{\hbar}\epsilon_n t} c_n |n\rangle . \tag{1.14}$$

We show now how to compute the evolution of a density matrix. If we define $|\psi(t)\rangle = U(t)|\psi(t_0)\rangle$ then:

$$\rho(t) = \sum_k p_k |\psi_k(t)\rangle \langle\psi_k(t)| = U\rho_0 U^\dagger, \tag{1.15}$$

where we define ρ_0 as the density matrix at $t = 0$ and $U^\dagger$ as the adjoint of U.

Now we describe how a subsystem evolves: to this purpose we focus on a 2-bodies system and we define $\rho = \rho_1 \otimes \rho_2$ where ρ_1 is the state of the subsystem 1, ρ_2 is the state of the subsystem 2 and $\rho(t)$ is the evolution of the state ρ as defined in Equation 1.15. In order to compute the evolution of the subsystem 1 we need to introduce the *Kraus representation*[10] which is defined as follows:

$$\rho_1' = \mathrm{Tr}_2(\rho(t)) = \sum_k E_k \rho_1 E_k^\dagger \qquad \text{with } \sum_k E_k^\dagger E_k = 1, \tag{1.16}$$

where E_k are operators called Kraus operators. In general, we introduce a map $\mathcal{S} : \rho_1 \to \rho_1'$ with the following properties:

- $\mathcal{S}$ preserves the Hermitianity, the Trace and the non-negativity;
- If $\rho' = \mathcal{S}_1(\rho)$ and $\rho'' = \mathcal{S}_2(\rho')$ then $\rho'' = \mathcal{S}_2(\mathcal{S}_1(\rho))$;
- Its inverse exists if and only if $\mathcal{S}$ is unitary.

The operators $\mathcal{S}$ are also called *superoperators*.

1.6 Quantum Gates

After having shown the time evolution of quantum states we focus on the evolution of a qubit. We have described in Section 1.1 how bits are manipulated, namely by applying gates: in analogy with the classical case, we now define quantum gates. They are modeled by unitary

operations because these are the protocols that conserve coherence; they are implemented by applying a properly chosen Hamiltonian.

The systems made by qubits and quantum gates are called *quantum circuits*. In their graphical representation qubits are represented by lines and gates by boxes with the gate's name inside. In the following we list the most important quantum gates; then we will focus on the X-gate, showing the Hamiltonian which can implement it.

One qubit's gates are unitary 2×2 matrix since they act on a 2-dimensional Hilbert space. They are:

- the Hadamard gate (Figure 1.3a).
  We can visualize it on the Bloch sphere as a $\frac{\pi}{2}$ rotation of the state vector in the (x,z) plane around the y-axis.
  It acts on the computational basis as follows: $\{|0\rangle, |1\rangle\} \xrightarrow{H} \{\frac{|0\rangle+|1\rangle}{\sqrt{2}}, \frac{|0\rangle-|1\rangle}{\sqrt{2}}\}$.
  $$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$
- The phase shift (Figure 1.3b).
  We can visualize it on the Bloch sphere as a Φ rotation of the state around the z axis.
  It adds a relative phase to the qubit's state: $\frac{|0\rangle+|1\rangle}{\sqrt{2}} \xrightarrow{R_z(\Phi)} \frac{|0\rangle+e^{i\Phi}|1\rangle}{\sqrt{2}}$.
  $$R_z(\Phi) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\Phi} \end{pmatrix}.$$
- The NOT or X gate (Figure 1.3c).
  We can visualize it on the Bloch sphere as a π rotation of the state on the x axis.
  It flips the qubit's state: $\{|0\rangle, |1\rangle\} \xrightarrow{X} \{|1\rangle, |0\rangle\}$.
  $$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

We show now in detail an example of how the quantum gates can be implemented by evolving a qubit with a proper Hamiltonian. The gate that we choose is the X gate and the Hamiltonian that we consider is:

$$H = -\mu(B_0\hat{\sigma_z} + B_1[\cos(\omega t)\hat{\sigma_x} + \sin(\omega t)\hat{\sigma_y}]), \tag{1.17}$$

with B_0, B_1 magnetic fields, $\{\hat{\sigma_i}\}$ the Pauli matrices. If we call the generic state $|\psi(t)\rangle =$

$\alpha(t)\,|0\rangle + \beta(t)\,|1\rangle = (\alpha(t), \beta(t))$ and perform the variables change $(\alpha(t), \beta(t)) \to (a(t)e^{-i\omega\frac{t}{2}}, b(t)e^{i\omega\frac{t}{2}})$ we obtain the time indipendent Hamiltonian $H_1 = \frac{\hbar}{2}(\omega_1\hat{\sigma_x} + (\omega_0 - \omega)\hat{\sigma_z})$ with ω_0 and ω_1 defined below. Solving the Schrödinger equation with the Hamiltonian H_1, and calling:

$$\omega_0 = -2\mu B_0/\hbar, \qquad \omega_1 = -2\mu B_1/\hbar, \qquad \Delta\omega_0 = \omega_0 - \omega,$$

$$\tan\theta = \omega_1/\Delta\omega_0, \quad E_1 = \tfrac{\hbar}{2}\sqrt{\Delta\omega_0^2 + \omega_1^2}, \quad E_2 = -\tfrac{\hbar}{2}\sqrt{\Delta\omega_0^2 - \omega_1^2},$$

we write the evolved state $|\psi(t)\rangle = \cos\frac{\theta}{2}e^{-\frac{i}{\hbar}E_1 t}\,|E_1\rangle - \sin\frac{\theta}{2}e^{-\frac{i}{\hbar}E_2 t}\,|E_2\rangle$. So the probability of being in the excited state $|1\rangle$ is:

$$p_{|1\rangle}(t) = \frac{\omega_1^2}{\Delta\omega_0^2 + \omega_1^2}\sin^2\left(\sqrt{(\omega_0 - \omega_1)^2 + \omega_1^2}\,\frac{t}{2}\right). \tag{1.18}$$

This above is the Rabi formula(fig 1.2), where we define the angular speed $\Omega = \sqrt{(\omega_0 - \omega_1)^2 + \omega_1^2}$, the amplitude $A = \frac{\omega_1^2}{\Delta\omega_0^2 + \omega_1^2}$ and the period $T = \frac{2\pi}{\Omega}$. If we define the frequency ω in resonance with the energy difference between levels $|0\rangle$ and $|1\rangle$, such that $\Delta\omega_0 = 0$, and stopping the evolution at $t = \frac{T}{2}$, we obtain the NOT gate as we see in Figure 1.2.

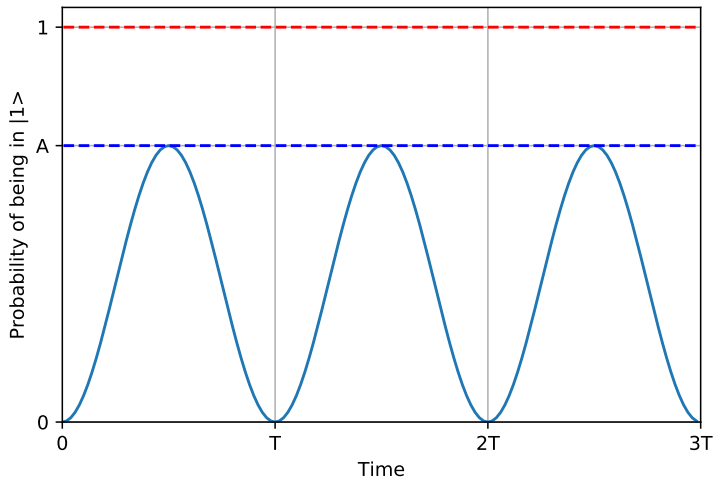
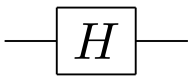


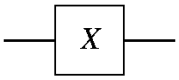
Figure 1.2: The probability of being in the state $|1\rangle$ is plotted as described in Equation 1.18. We can see that if the coefficient A tend towards 1 by stopping the evolution at $\frac{T}{2}$ we can perform an X gate on a given qubit.

Before the introduction of two-qubits gates we need to underline a fundamental differ-



(a) The H in the box means that an Hadamard gate is applied to the qubit. The line identify the qubit to which the H is applied to.

(b) The $R\Phi$ in the box means that a Phase Shift of Φ is applied to the qubit. The line identify the qubit to which the Phase Shift is applied to.



(c) The X in the box means that a NOT gate is applied to the qubit. The line identify the qubit to which the X is applied to.

Figure 1.3: Gates that are applied to a single qubit

ence between them and their classic counterparts: in the quantum case there isn't an output qubit, while in the classical case an output bit exists. The state of the first qubit, called *target*, undergoes a certain transformation if the state of the second qubit, called *control*, satisfy some determined conditions. Two-qubits gates are represented by unitary 4×4 matrix; in particular, we will see that they allow us to create entangled states, starting from separable ones. They are:

- Control Not (Figure 1.4a), also named as CNOT or CX. It applies a NOT gate on the target qubit if and only if the control qubit is in the $|1\rangle$ state.

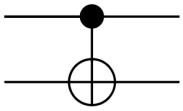
$$CX = \begin{pmatrix} \mathbb{I}_{2\times2} & \mathbb{O}_{2\times2} \\ \mathbb{O}_{2\times2} & X \end{pmatrix}.$$

  We used the block notation, so that $\mathbb{I}_{2\times2}$ is the 2×2 identity matrix and $\mathbb{O}_{2\times2}$ is the 2×2 o matrix.

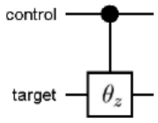
- Control Phase Shift (Figure 1.4b), it applies a phase shift represented by $R_z(\theta)$ to the target qubit if and only if the control qubit is in the $|1\rangle$ state.

$$CPHASE(\theta) = \begin{pmatrix} \mathbb{I}_{2\times2} & \mathbb{O}_{2\times2} \\ \mathbb{O}_{2\times2} & R_z(\theta). \end{pmatrix}$$

It has been shown that a universal set of qubits gates is formed by the Hadamard gate H, the $\pi/4$ phase shift $T = R_z(\pi/4)$ and the control NOT CX. In particular, it is possible to build each classical gate with the universal quantum gates. We will show this building the Toffoli gate(fig 1.6), a three-qubits gate equivalent to the classical AND: it flips the target qubit if and only if both controls are $|1\rangle$ (Fig 1.5). The qubit c is equivalent to the output



(a) This is the CNOT graphical representation: a X gate, represented by a cross inside a circle, is applied to the target qubit if the control is in $|1\rangle$



(b) This is the Control Phase Shift graphical representation: a $R_z(\theta)$ gate, represented by a θ_z inside the box, is applied to the target qubit if the control is in $|1\rangle$

Figure 1.4: Gates that are applied to two qubits. The black dot indicates the target qubit.

qubit in the classical circuit, and a and b are the input. The Toffoli gate leaves a and b unvaried and apply a NOT to c.

Inputs			Outputs		
a	b	c	a'	b'	c'
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

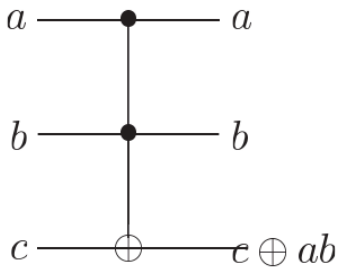


Figure 1.5: The truth table of the Toffoli gates is shown above. In the graphical representation the black dots represents the target and the ' is omitted to underline that a and b are unvaried. The $\oplus$ represent the addiction module 2, a mathematical representation of a Control Control NOT.

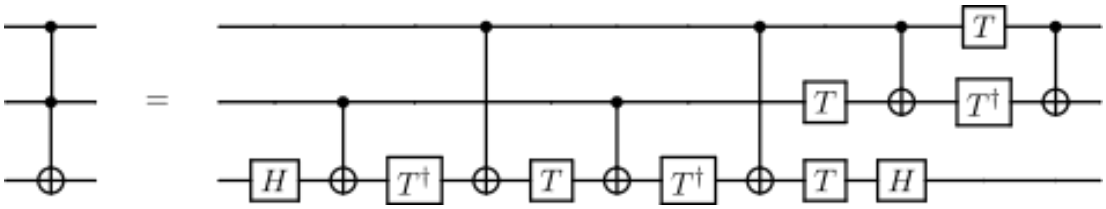


Figure 1.6: Composition of toffoli gate using only the fundamentals ones: H, CNOT and $T = R_z(\pi/4)$. Image from: `https://upload.wikimedia.org/wikipedia/commons/9/9d/Toffoli_decomposition.svg`.

Isolated systems are ideal: it is impossible to build a system that doesn't interact with the environment. As a consequence non-unitary effects are involved in the dynamics of the system, which alter the state of the system as we show here after. It is important to take these effects into account since, as we will see in Chapter 3, they limit the efficiency of a quantum computer. Since we cannot express such processes as a unitary evolution, we adopt the formalism of Kraus operators, according to which the evolution is obtained by applying a superoperator: $\rho_S \to \rho'_S = \sum_k E_k \rho_S E_k^\dagger$. We want to show that the action of the superoperator S on ρ can be interpreted as an affine transformation. Using the representation in polar coordinates in Eq 2.6 we can write an arbitrary qubit state as $\rho = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma})$. The transformation has the following form:

$$\vec{r} \to \vec{r'} = M\vec{r} + \vec{c}. \tag{1.19}$$

The related Kraus operators $E_k = \gamma_k \mathbb{I} + \sum_{l=1}^{3} a_{kl}\hat{\sigma}_l$ are defined as:

$$M_{jk} = \sum_{l=1}^{3} \left[a_{lj}a_{lk}^* + a_{lj}^* a_{lk} + \left(|\gamma_l|^2 - \sum_{p=1}^{3} |a_{lp}|^2 \right) \delta_{jk} + i \sum_{p=1}^{3} \epsilon_{jkl}(\gamma_l a_{lp}^* - \gamma_l^* a_{lp}) \right], \tag{1.20}$$

$$c_j = 2i \sum_{k,l,m} \epsilon_{jlm} a_{kl} a_{km}^*, \tag{1.21}$$

where ϵ_{ijk} is the pseudotensor of Levi-Civita. This transformation is in general non-invertible. It is also possible to visualize the action of a quantum channel on a qubit by considering its Bloch sphere: indeed, a transformation as the one in Equation 1.19 transforms the Bloch sphere in a translated ellipsoid with a volume minor or equal than the sphere's volume, as we can see in Fig 1.7. We analyze now three one-qubit quantum channels which are related to three different processes, the amplitude dunmping, the phase dumping, and the depolarizing channel.

- *Amplitude damping.*
  It is the process that makes the excited state $|1\rangle$ decaying into the ground state $|0\rangle$. Given a density matrix ρ, it is defined as the process $\rho \to \rho' = E_0 \rho E_0^\dagger + E_1 \rho E_1^\dagger$,

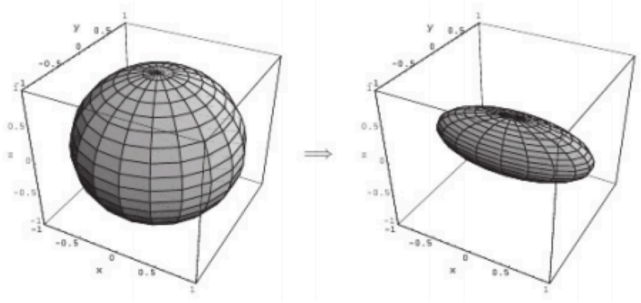


Figure 1.7: Graphical representation of the application of the affine map in Equation 1.19 to the Bloch sphere: it transforms the sphere in a translated ellipsoid.

where the Kraus operators E_0 and E_1 are defined as follows:

$$E_0 = \begin{pmatrix} 1 & 0 \\ 0 & \sqrt{1-p} \end{pmatrix}, \qquad E_1 = \begin{pmatrix} 0 & \sqrt{p} \\ 0 & 0 \end{pmatrix}. \tag{1.22}$$

We obtain:

$$\rho' = \begin{pmatrix} \rho_{00} + (p)\rho_{11} & (\sqrt{1-p})\rho_{01} \\ (\sqrt{1-p})\rho_{10} & (1-p)\rho_{11} \end{pmatrix}, \tag{1.23}$$

where ρ_{ij} are the elements of the density matrix ρ. By recalling that $\rho = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma})$ with $\vec{r} = (x, y, z)$ and calling $\rho' = \frac{1}{2}(\mathbb{I} + \vec{r'} \cdot \vec{\sigma})$ we see that the Bloch-sphere coordinates transform as follows:

$$x' = \sqrt{1-p}x, \qquad y' = \sqrt{1-p}y, \qquad z' = p + (1-p)z. \tag{1.24}$$

The sphere becomes an ellipsoid centered in $(0, 0, p)$, where p is the *damping probability*, namely the probability that the state $|1\rangle$ decays in the fundamental state $|0\rangle$. By applying this channel n times we observe that the probability of being in $|1\rangle$ drops exponentially:

$$p_1^{(n)} = (1-p)^n p_1^{(0)} = e^{n \ln(1-p)} p_1^{(0)}. \tag{1.25}$$

This means that for $n \to \infty$ the system is driven to the pure state $\rho^{(\infty)} = |0\rangle \langle 0|$.
We can write a continuous time version of this discrete process by defining $p = \Gamma \Delta t$ and $t = n\Delta t$, with $\Delta t \to 0$:

$$p_1(t) = \lim_{\Delta t \to 0} (1 - \Gamma \Delta t)^{\frac{t}{\Delta t}} p_1(0) = e^{-\Gamma t} p_1(0). \tag{1.26}$$

We obtain the transition rate Γ, and we call its inverse the *relaxation time* of a qubit. We indicate it with $T1$ and we will see that it plays an important role in quantum computing.

- *Phase damping.*
  It is the process that eliminates the coherence: it trasforms the state $\frac{1}{2}(|0\rangle + |1\rangle)(\langle 0| + \langle 1|)$ into the state $\frac{1}{2}(|0\rangle \langle 0| + |1\rangle \langle 1|)$ Let us start by noting that an arbitrary state of a qubit can be written as :

$$\rho = \begin{pmatrix} p & \alpha \\ \alpha^* & 1-p \end{pmatrix}, \tag{1.27}$$

  with $p \in [0,1]$ and $|\alpha| = \sqrt{p(1-p)}$. We model the decoherence process by using a phase kick: namely we replace the interaction with the environment with a set of rotations around the z-axis of the Bloch sphere $R_z(\theta)$, where θ is drawn from the random distribution:

$$p(\theta) = \frac{1}{\sqrt{4\pi\lambda}} e^{-\frac{\theta^2}{4\lambda}}. \tag{1.28}$$

  By applying this process we can compute the new density matrix ρ', obtained averaging over θ:

$$\rho' = \int_{-\infty}^{+\infty} d\theta p(\theta) R_z(\theta) \rho R_z^\dagger(\theta) = \begin{pmatrix} p & \alpha e^{-\lambda} \\ \alpha^* e^{-\lambda} & 1-p \end{pmatrix}. \tag{1.29}$$

  The Bloch-sphere coordinates of ρ' are, in this case:

$$x' = e^{-\lambda} x, \qquad y' = e^{-\lambda} y, \qquad z' = z. \tag{1.30}$$

  Note that also in this case we obtain a prolate ellipsoid. As expected, we see that coherence drops exponentially to zero when we apply the channel n times, since we obtain $\alpha^{(n)} = e^{-\lambda n}\alpha$. We can give a continuous time version of the coherence decay, similarly to the damping amplitude case. If we call $\lambda = \Theta \Delta f$ and $t = n\Delta t$ with $\Delta t \to 0$ we obtain:

$$\alpha(t) = e^{-\Theta t}\alpha(0), \tag{1.31}$$

  where Θ represents the decoherence rate. We define the *decoherence time* $T2 = \frac{1}{\Theta}$ the time in which the coherent state $|0\rangle + |1\rangle$ decays in the classical state $|0\rangle \langle 0| + |1\rangle \langle 1|$.

- *Depolarizing channel.*
  It is the process that depolarize the qubit, diminishing the projections of the state over the axis x,y and z. Over the time the arbitrary state decays in the state $|0\rangle \langle 0| + |1\rangle \langle 1|$. It can be also defined by using the decomposition of the density matrix in polar coordinate of Equation 1.4, namely $\rho = \frac{1}{2}(\mathbb{I} + \vec{r} \cdot \vec{\sigma})$, with $r = (0,0,0)$.
  The Kraus operators associated with this channel are:

$$E_k = \{\sqrt{1-p}\mathbb{I}, \sqrt{\frac{p}{3}}\sigma_x, \sqrt{\frac{p}{3}}\sigma_y, \sqrt{\frac{p}{3}}\sigma_z\}, \tag{1.32}$$

where p is the depolarizing probability.
The transformation is such that:

$$\rho' = \frac{1}{3}p(\sigma_x\rho\sigma_x^\dagger + \sigma_y\rho\sigma_y^\dagger + \sigma_z\rho\sigma_z^\dagger) + (1-p)\rho. \tag{1.33}$$

If we use the notation with polar coordinates reminded above we can note a simpler way to characterize this channel:

$$\vec{r'} = \left(1 - \frac{4}{3}p\right)\vec{r}, \tag{1.34}$$

and in this notation it is clear that the repeatedly application of this channel lends the state to $\rho = \frac{1}{2}\mathbb{I}$, the one with $\vec{r} = 0$.

2

The IBM quantum processor

In this chapter we will discuss the hardware and software specifics of the quantum computer on which we are focusing on this thesis, namely the 'ibmq_16_melbourne'. First, we will briefly explain how the qubits of the processor are realized; then, we will discuss in detail which operations can be implemented on the real quantum computer. In the end, we will focus on the programming language which allows users to interface with the machine, showing some examples of codes.

2.1 Qubit implementation

The physical qubits used by the IBM are fixed-frequency superconducting transmon qubits[11]. A transmon qubit is a type of charge qubit designed to have reduced sensitivity to charge noise. In the following we briefly illustrate how a charge qubit works, but a deep analysis is beyond the aim of this thesis.

A charge qubit is a Josephon-junction-based qubit[12] [10]. A josephon junction consists of two superconducting regions separated by a weak link, usually an insulator. The low dissipation which characterize to superconductors makes possible, in principle, long decoherence times. The fundamental ingredient for a material to behave as a superconductors is the presence of Coopers Pairs: they are pairs of bounded electrons which form a system of integer spin and thus behave like bosons. In particular they can tunnel through the insulator foil.

Let us now explain in detail the circuit in Figure 2.1, which contains a Josephon junction:

there are an insulator with intrinsic capacity C_j, a capacitor with capacity C_g and tension applied V. The superconductor between the insulator and the capacity is called *island*.

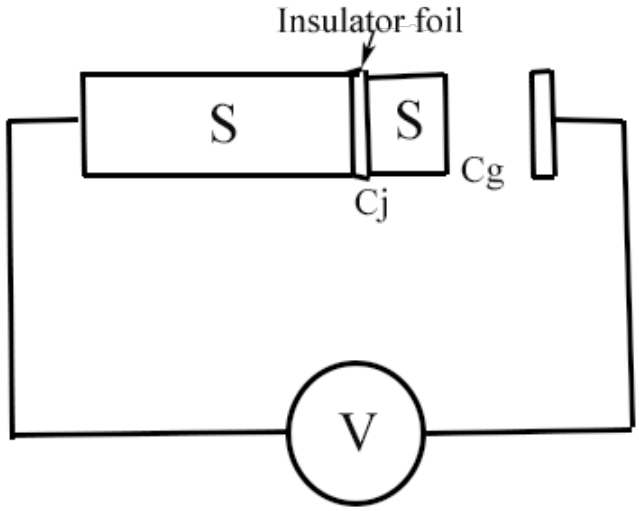


Figure 2.1: Circuit of a charge qubit. The S is for superconductive, C_j is the capacity related to the insulator, C_g is a capacity and V the voltage applied. The superconductor between the insulator and the capacity is called island.

There are two quantum numbers which allows to describe the junction: n and ϕ, where n is the number of cooper pairs in the island and ϕ is called superconductive phase. We can now show the Hamiltonian related to this system:

$$H = E_c(n - n_g)^2 - E_j \cos\phi, \tag{2.1}$$

where $n_g = C_g \frac{V}{2e}$ is the number of cooper pairs present in the system with e the electron charge, $E_c = \frac{(2e)^2}{2(C_j+C_g)}$ is the term related to the energy that a difference between the effective and aspected cooper pairs creates, E_j is related to the state of the island and $\cos\phi$ is the tunneling term. We can write this Hamiltonian in the basis of the cooper pairs $|n\rangle$, namely:

$$H = E_c \sum_n (n - n_g)^2 |n\rangle \langle n| - \frac{1}{2} E_j \sum_n |n+1\rangle \langle n| + |n\rangle \langle n+1|, \tag{2.2}$$

where we can see in a clearer way that the second term is a coupling term, that connects the state $|n+1\rangle$ with the state $|n\rangle$. The two levels of the qubit are made using the interaction between the state $|n=0\rangle$ and $|n=1\rangle$. After having explained how a single qubit is imple-

mented, we move to the description of the quantum processor "ibmq_16_melbourne".

2.2 Specifics of the quantum processor

In an ideal quantum processor all the qubits are connected between each other, in fact it is not possible due to the spatial arrangement of the qubits: indeed a *coupling map* is defined, namely a graph that shows the allowed connections among the qubits (Figure 2.2). Notice that the connections are also characterized by arrows: they indicate the default direction of the two-qubit operation which can be applied, the CNOT: the arrow goes from the control to the target qubit.

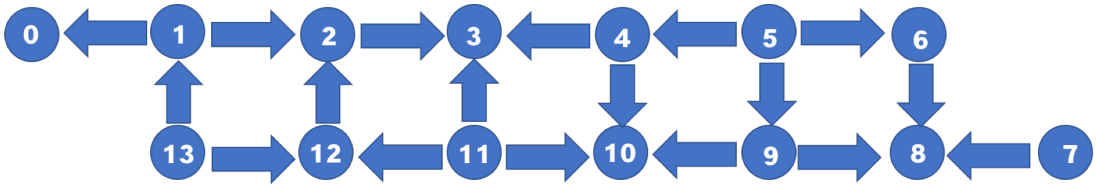


Figure 2.2: Coupling map of the "ibmq_melbourne_16" processor. The arrow shows which is the default direction of the CNOT applied.
Image from: `https://github.com/Qiskit/ibmq-device-information/blob/master/backends/melbourne/images/melbourne-connections.png`

In order to ask the processor to execute a general operation we need to know which are the fundamental gates built into the machine. They are:

- $u1$, which implements a phase shift with angle λ:

$$u1(\lambda) = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{pmatrix}.$$

- $u2$, which allows us to create superpositions :

$$u2(\phi, \lambda) = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -e^{i\lambda} \\ e^{i\phi} & e^{i(\phi+\lambda)} \end{pmatrix}.$$

  An example of a gates which is realized by the $u2$ is the Hadamard gate: we have $H = u2(0, \pi)$.

- $u3$, which is the most general unitary transformation implemented:

$$u3(\theta, \phi, \lambda) = \begin{pmatrix} \cos(\theta/2) & -e^{i\lambda}\sin(\theta/2) \\ e^{i\phi}\sin(\theta/2) & e^{i(\phi+\lambda)}\cos(\theta/2) \end{pmatrix}.$$

An example of a gate which can be realized by the $u3$ gate is the NOT gate X, since $X = u3(\pi, 0, \pi)$.

- The CNOT, the two-qubit gate defined in Section 1.6.

It is important to know that each qubit has a different gate failure probability, namely the probability that the gate is not applied correctly. It is defined as $\frac{n_g^f}{n_g^s}$ with n_g^f the number of gate failed and n_g^s the number of gates successed. The one-qubit gate errors are reported in Table 2.1 and the two-qubit gate errors in Table 2.2.

Qubit	Gate Error	Qubit	Gate Error
0	0.004	1	0.01
2	0.006	3	0.003
4	0.004	5	0.006
6	0.003	7	0.004
8	0.004	9	0.006
10	0.003	11	0.3
12	0.008	13	0.01

Table 2.1: The gate error, namely the probability of failure of a gate applied to the qubit. It is defined as n_g^f/n_g^s, with n_g^f the number of gate failed, and n_g^s the number of gate successed.

Each gate is realized by applying some pulses to the qubits: in Figure 2.3 we show the decomposition of the gates in pulses[13]. They are the physical operations, namely the operations that are performed on physical qubit. The analysis of the steps which compose a gate is beyond the aim of this thesis, but we mention them to define the gate application time, since the IBM defines only their times. The pulse called FC has an application time of $0ns$, the pulse GD takes $100ns$ and the pulse GF times are listed in Table 2.2. After each GD or GF there is a buffer time of $20ns$. In Chapter 3 we will take into account that each gate has an an effective time of gate application, t_{gate}^{eff}, given by the sum of the gate time and of the buffer. It is important to underline the application time of the identity gate, because it will be used extensively in the next chapter: it is $100ns$.

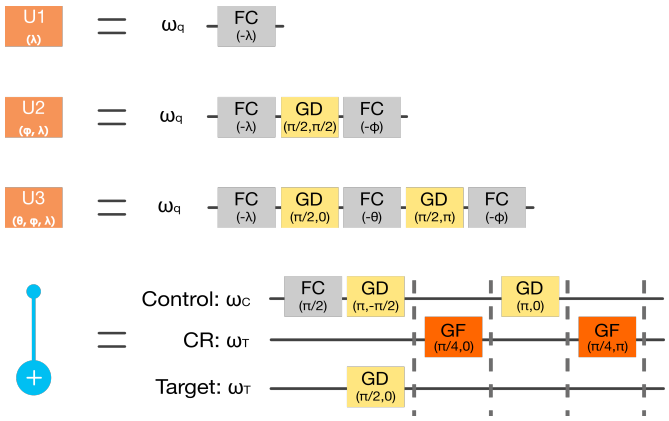


Figure 2.3: The composition of the fundamental gates in pulses. The Frame Change (FC) has an application time of $0ns$, the Gaussian Derivative (GD) of $100ns$ and the Gaussian Flattop (GF) times are listed in Table 2.2. After each GD or GF there is an additional buffer of 20ns.
Image from: `https://github.com/Qiskit/ibmq-device-information/blob/master/backends/melbourne/images/gatedef_U1U2U3_CNOT.png`

CX Gate	GF Gate Time $[ns]$	Gate Error	CX Gate	GF Gate Time $[ns]$	Gate Error
CX1_0	239	0.04	CX1_2	174	0.07
CX2_3	261	0.05	CX4_3	266	0.04
CX5_4	300	0.05	CX5_6	300	0.07
CX7_8	220	0.04	CX9_8	400	0.04
CX9_10	300	0.05	CX11_10	261	0.18
CX11_12	217	0.13	CX13_12	300	0.04
CX13_1	652	0.17	CX12_2	1086	0.07
CX11_3	286	0.12	CX4_10	261	0.04
CX5_9	348	0.07	CX6_8	300	0.03

Table 2.2: CNOT application time and Gate error, namely the probability of failure of the gate.

2.3 The Qiskit software

Qiskit[14] is the software interface, written as a python librar, developed to run experiment and simulations by the IBM community. It is divided in four macroareas of interest, called "elements":

1. Terra: its scope is to provide a bedrock for composing quantum programs at level of circuits and pulses, to optimize them for the constraints of the device and manage the execution of experiment on the real remote-access devices;

2. Aer, namely the part build to perform numerical simulations. We can use Aer to verify that the real processors work properly, simulating the theoretical evolution of the sys-

tem with numerical methods. It can be also used to simulate systems with the effects of realistic noise;

3. Ignis, which is dedicated to overcome noise and errors. This includes error correction and computing the presence of noise. For example it gives functions to measure the decoherence time and the relaxation time, both defined in Section 1.7, and gates errors;

4. Aqua, which is dedicated to some possible real-world applications, for examples in chemistry, optimization or artificial intelligence. In order to use Aqua, one doesn't need to translate from scratch the problem into the language of the quantum machine, because there are pre-constructed functions.

In this thesis we mainly focused on the application of fundamental gates to the real processor, therefore we mainly used the element Qiskit Terra.

We show now some of the basic commands of qiskit Terra, then we illustrate below an example in which these commands are applied. These examples are representative to understand all the gates which will be discussed in Chapter 3. The first element to define is the Register, which specifies how many qubits we will use in our circuit and how many bits we will use to store the outcome of a measure: the command is

```
q=QuantumRegister('q',n),    c=ClassicalRegister('c',m),
```

where `'q'` and `'c'` are the name of the registers and n,m are the number of qubits and bits respectively. Then we use them in a quantum circuit with the command:

```
qc=QuantumCircuit(q,c),
```

where `q` must be a QuantumRegister and `c` a ClassicalRegister.

We can build our quantum circuit applying various gates. The sintax is the following:

```
qc.gate(q[i]) ,
```

where `qc` is a quantum circuit, `gate` is the gate that we want to implement, `q` is a QuantumRegister and `i` is the label which represent the qubit to which we want to apply the gate. In this case, for example, if `q` is a register with 5 qubits and we write `qc.h(q[3])` we are applying an Hadamard gate to qubit 3.

We define now the command needed to launch the experiment, stressing the importance of some of its parameters:

```
job = execute(qc, backend, initial_layout, shots),
```

where `qc` is a QuantumCircuit, `backend` is the device on which the job will be run, `initial_layout` is the map between virtual qubits (the ones in the register) and physical qubits and `shots` is the number of times the experiment will be repeated in order to perform a valid statistical analysis.
In the end, we define the commands necessary to read the results:

```
res=job.result(),     counts=res.get_counts().
```

The `counts` structure contains the results of the experiment, namely it literally counts the number of times in which the system has been found in a determined state.

We show now two examples of circuits realized through the qiskit code: in Listing 2.1 we show how to reverse the target and the control qubits in a CNOT (Figure 2.4). This gate

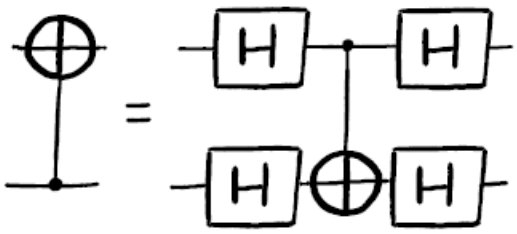


Figure 2.4: Reversing the direction of a CNOT, namely change the roles between target qubit and control qubit, using 4 Hadamard gate and a CNOT gate. Image from: `https://dal.objectstorage.open.softlayer.com/v1/AUTH_039c3bf6e6e54d76b8e66152e2f87877/images-classroom/cnot_reversetvhxy1y40307ldi.png`

is very important because, as we have explained before (see Figure 2.2), target and control qubits are fixed into the device.

```
q = QuantumRegister( 'q', 2) # We define the quantum
#register a 2 qubit.
# We won't define a classical register because in this
#example we don't need to make a measure.
qc = QuantumCircuit(q)
qc.h(q) # We apply an Hadamard gate to each qubit
#of the register.
qc.cx(q[0],q[1]) # The first qubit is the control qubit,
#the second is the target qubit.
```

```
qc.h(q)
```

Listing 2.1: Code to reverse the direction of a CNOT gate.

In Listing 2.2 we show, instead, how to build the Bell Couple $|\phi^+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$ starting from the state $|00\rangle$, run the experiment on the Melbourne device and take the results.

```
q= QuantumRegister('q',2)
c= ClassicalRegister('c',2)
qc= QuantumCircuit(q,c)
backend = IBMQ.get_backend('ibmq_16_melbourne')
qc.h(q[0])
qc.cx(q[0],q[1])
job = execute(qc, backend, shots=1024)
res = job.result()
counts = res.get_counts()
```

Listing 2.2: Construction of a Bell Couple.

We have to remember that what we observe is not nature in itself but nature exposed to our method of questioning.

Werner Heisenberg

3

Characterization of the quantum processor

The aim of this chapter is to analyze quantitatively the qubits of the quantum processor through some apposite experiments. In particular, we measure the relaxation time and the coherence time, defined in Section 1.7. As discussed before they are fundamental in order to understand the current limits of a quantum computer. As wee will show in the particular case we investigate, these times change from qubit to qubit. Moreover, the measure of the characteristic times isn't stable over time; the results presented in this chapter are relative to the calibration of the machine, made on 13/06/2019.

The initial state for all the experiments shown below is $|0\rangle$, and to acquire sufficient data we acquire 50 points in each experiment, namely we make 50 iteration of the protocol. In order to compute the expectation value we take the average over 1024 repetition of each experiment, so each iteration is made of 1024 shots. We have adopted this protocol because to simulate the time evolution of the system we must prepare the system in the initial state and then wait the opportune time. It is not possible to make various measure on the same system at different time steps, due to the nature of quantum mechanics: a measure disrupt the system.

3.1 The relaxation time $T1$

The relaxation time is the time in which the $|1\rangle$ state decays in the $|0\rangle$ state via the non-unitary process described in Section 1.7. In order to measure the former $T1$ we set up an experiment

in which an X gate is applied to a single qubit to bring it in the excited state $|1\rangle$.
At the time $t_0 = 0$ we apply 45 identity gate for each iteration, namely the time t_i corresponds to $45i$ application of the identity gate. The total time is Nt_i, with N the number of iterations. We recall that each identity gate has an effective application time of
$t^{eff} = 120ns$ and that the identity gate does nothing. Then we measure the system on the computational basis $\hat{\sigma_z}$ at each time t_i. The corresponding quantum circuit for a single time step is shown in Figure 3.1.

If no non-unitary effect affects the system we should find the qubit in the state $|1\rangle$ at any

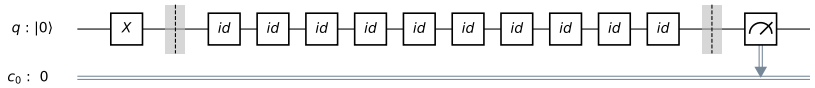


Figure 3.1: Quantum circuit used to measure $T1$. We bring the qubit to the $|1\rangle$ state with the X gate and then apply $45i$ identities gates, whit i the iteration index. The number of identity gates (id) in this figure is indicative.

time because the state should not evolve. Instead, we see in Figure 3.2 that the probability of measuring the system in the state $|1\rangle$ is disposed on a decreased exponential. This result is compatible with the theory of the Amplitude dumping quantum channel, described by Equation 1.26.

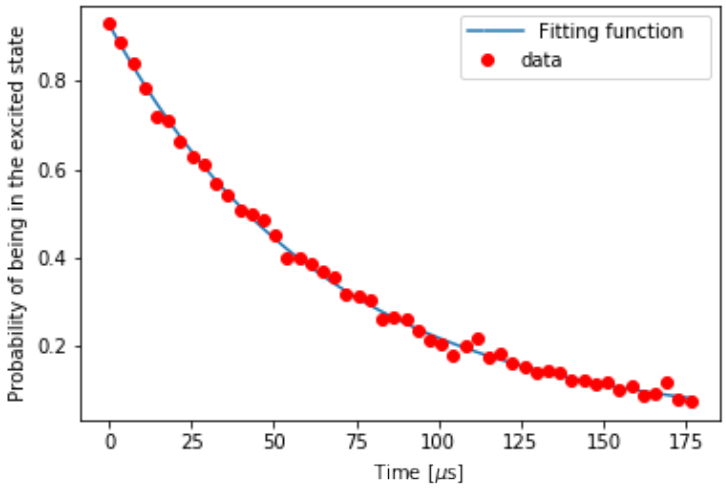


Figure 3.2: Decay of qubit 0 from the state $|1\rangle$ to the state $|0\rangle$ over time. Note the agreement with the experimental fitting function.

In order to measure the decaying rate with a better precision we linearize it, so we make the fit using the function:

$$\ln\left(p_{|1\rangle}(t)\right) = -mt + q. \tag{3.1}$$

The value of the relaxation time $T1$ is obtained by taking the inverse of the factor m. In Figure 3.3 we plot two different data-sets corresponding to two qubits in order to make it visible that all the qubits have similar exponential behavior, while in Table 3.1 we show the results obtained for all the qubits.

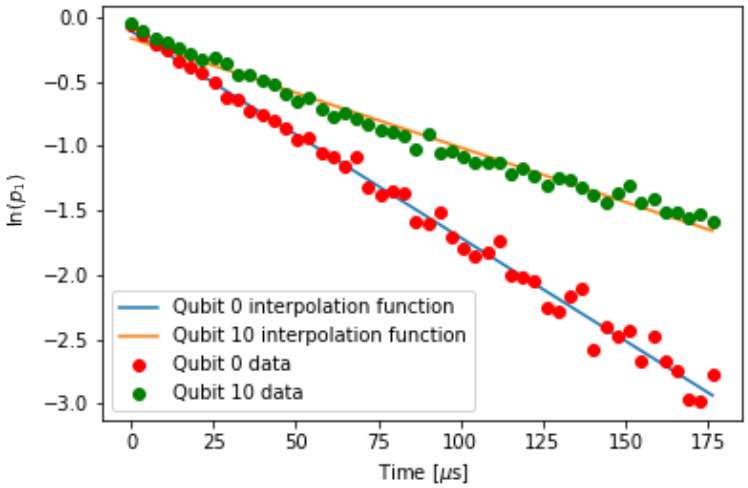


Figure 3.3: Decay of the state from $|1\rangle$ to $|0\rangle$ over time. With $\ln(p_1)$ we indicates the natural logarithm of the probability of being in the excited state. We can see the linearization of the exponential decay for qubit 0 and 10, because these are the qubits with the most different $T1$.

Qubit	$T1$ [μs]	Qubit	$T1$ [μs]
0	62.49 ± 0.02	1	56.74 ± 0.02
2	77.55 ± 0.03	3	57.45 ± 0.02
4	76.41 ± 0.02	5	77.50 ± 0.02
6	90.61 ± 0.03	7	78.51 ± 0.02
8	78.38 ± 0.02	9	80.18 ± 0.03
10	118.18 ± 0.03	11	84.13 ± 0.02
12	75.75 ± 0.03	13	76.38 ± 0.02

Table 3.1: Measured relaxation time ($T1$), namely the time in which the excited state $|1\rangle$ decays in the ground state $|0\rangle$, with errors. These values are obtained from the fit.

3.2 The decoherence time $T2$ and $T2^*$

The decoherence time is the time in which the coherent state $|+\rangle\langle+|$ decays in the mixed state $|0\rangle\langle 0| + |1\rangle\langle 1|$, where we remind that $\{|+\rangle, |-\rangle\}$ are the eigenstates of σ_x. There are two different experiments, presented below, that allow to measure the qubit decoherence time, and so two different exstimates, that we indicate with $T2$ and $T2^*$. We expect the decoherence time to be small in relation to $T1$, so we will work in the approximation that the amplitude dumping effects are negligible. The results are in Table 3.2.

Qubit	$T2$ [μs]	$T2^*$ [μs]
0	24.35 ± 0.03	21.60 ± 0.05
1	20.94 ± 0.03	21.70 ± 0.05
2	19.82 ± 0.03	21.02 ± 0.04
3	23.51 ± 0.03	22.04 ± 0.04
4	22.60 ± 0.03	22.06 ± 0.04
5	23.46 ± 0.02	21.93 ± 0.05
6	20.77 ± 0.04	22.57 ± 0.04
7	22.96 ± 0.03	23.89 ± 0.05
8	24.52 ± 0.03	21.87 ± 0.04
9	23.00 ± 0.03	21.91 ± 0.05
10	22.51 ± 0.03	21.21 ± 0.05
11	23.05 ± 0.03	21.60 ± 0.05
12	22.87 ± 0.04	21.83 ± 0.05
13	24.95 ± 0.03	21.73 ± 0.05

Table 3.2: Measured decoherence time, namely the time in which the coherent state $|+\rangle\langle+|$ decays in the mixed state $|0\rangle\langle 0| + |1\rangle\langle 1|$, with errors. These values are obtained from the fit.

3.2.1 Ramsey experiment: $T2$

In this experiment, we want to measure the decoherence time $T2$ by simulating the evolution of the qubit under a two-level Hamiltonian. We choose to analyze a time interval which covers three periods of the system. By recalling that we divide the evolution time into 50 iterations, we have that the period needs to be equal to $T = 50/3$; in order to to simulate the dynamics of the system we will use a pulsation $\omega = \frac{2\pi}{T} = \frac{6\pi}{50}$. The expected behavior of the system is a cosine, precisely $P_{|+\rangle} \propto \cos(\omega j)$ where j is the time discretization. The behavior that we observe by applying the decoherence quantum channel is, instead, an exponentially dumped cosine. Now we explain how we experimentally simulate this system.

First, we apply a Hadamard gate to put the state in the initial state $|\psi_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Then we apply a phase shift $R_z(\frac{6\pi j}{50})$ where j is the index of the j-th iteration. Then $5j$ identities are applied to recreate the time evolution. This is very important: we first create the state that we expect at the time j and then we let it evolve until the time j applying only identity gates. This method is preferable compared to applying j times a fixed phase shift because, as we have seen in Table 2.1, single-qubit gates have an error probability: it is more convenient to apply one phase shift $R_z(\frac{6\pi j}{50})$ than j phase shift $R_z(\frac{6\pi}{50})$. We are now interested in the population of the $|+\rangle$ state, so in the end we perform a measure along the X axis, that we achieve by applying an H before measuring the state in the computational basis σ_z (Fig 3.4). In the following we show the steps necessary to compute the probability p_+ of the time j:

$$|\psi_0\rangle = H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \tag{3.2}$$

$$|\psi_1\rangle = R_z|\psi_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + e^{i\frac{6\pi j}{50}}|1\rangle), \tag{3.3}$$

$$|\psi_2\rangle = (Id)^{5j}|\psi_1\rangle = |\psi_1\rangle, \tag{3.4}$$

$$|\psi_3\rangle = H|\psi_2\rangle = \cos\left(\frac{3\pi j}{50}\right)|+\rangle + \sin\left(\frac{3\pi j}{50}\right)|-\rangle, \tag{3.5}$$

$$p_+ = |\langle\psi_3|+\rangle|^2 = \cos^2(\frac{3\pi j}{50}) = \frac{1}{2} + \frac{1}{2}\cos\left(\frac{6\pi j}{50}\right), \tag{3.6}$$

where it is evident that the population of the state $|+\rangle$ should follow a cosine.

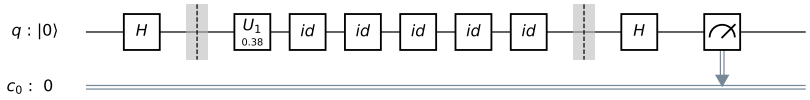


Figure 3.4: Quantum circuit to calculate $T2$. The u1 gate is the one defined in Chapter 2 and it is equivalent to a phase shift. The number in the relative box is the phase expressed in radiants.

In Fig 3.5 we see that datas are disposed on an exponentially dumped cosine, as predicted by the theory in Section 1.7. The inverse of the coefficient of the exponential is the decoherence time $T2$. The fitting function is:

$$p_{|+\rangle} = Ae^{-\Gamma t}\cos(\omega t + \delta) + B. \tag{3.7}$$

In Fig 3.5 we show the results obtained for two qubits, there labeled with 2 and 13; the others exhibit similar results.

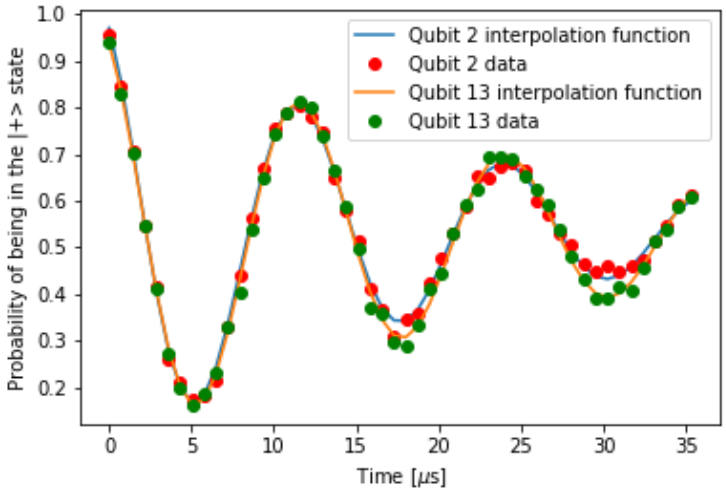


Figure 3.5: Decay of the $|+\rangle$ state on the $|0\rangle\langle 0| + |1\rangle\langle 1|$ state over time with Ramsey experiment for qubits 2 and 13.

3.2.2 Echo experiment: $T2^*$

In this experiment we want to measure the decoherence time $T2^*$ by observing how the coherent state $\frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$ decays in the mixed state $\frac{1}{2}\mathbb{I}$, but we want to make our estimation independent by any possible effects that may effect only the state $|0\rangle$ or $|1\rangle$. We achieve these hypothesis through the experimental setup described below.

We start from the same state of the other experiment, namely $|\psi_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)$. Then we apply $15i$ identities, an X gate to flip the population of $|0\rangle$ and $|1\rangle$ states and $15i$ identities again, with i the index of the i-th iteration of the experiment.

The application of the X gate is fundamental: in this way, we eliminate all sorts of non-symmetrical effects that are applied only to the state $|1\rangle$, like the amplitude dumping channel, or vice-versa. The application of the same number of identity gates, guarantees that the state and its inversion evolve for the same time: in this way the non-symmetrical effects are even. We are interested in the population of the $|+\rangle$ state so, at the end, we perform an X-measure, that we achieve applying an H before the measure on the computational basis σ_z (Fig 3.6).

In Fig 3.7 we see how the evolution of the probability of measuring the state $|+\rangle$ follows a decreasing exponential curve, as expected from what we explained in Section 1.7 and Equa-

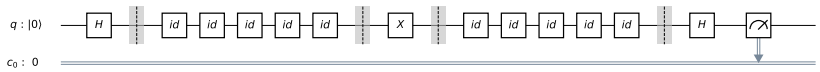


Figure 3.6: Quantum circuit to calculate T2*. The number of identity gates is indicative: in the real experiment there are 15 identity before the X gate and 15 after.

tion 1.31. The exponential fitting function is:

$$p_{|+\rangle}(t) = Ae^{-\Gamma t} + B. \tag{3.8}$$

In Fig 3.7 there are only two qubits data-set because the other are analogous.

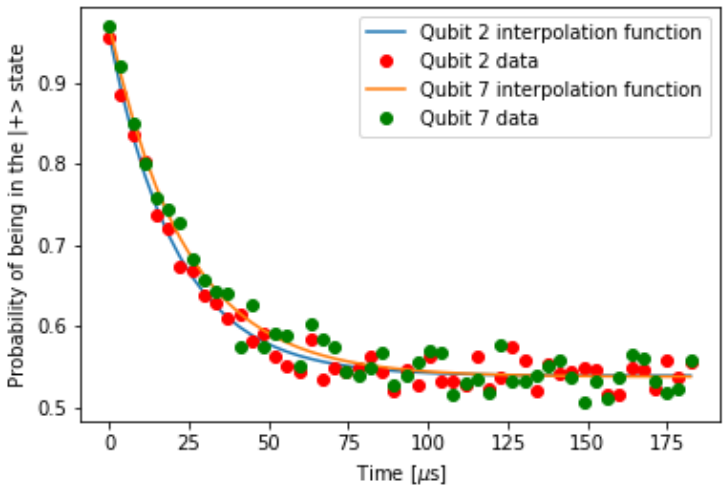


Figure 3.7: Decay of the $|+\rangle$ state on the $|0\rangle\langle 0| + |1\rangle\langle 1|$ state over time with Echo experiment for qubits 2 and 7.

3.3 Comparison between IBM value and measured time

Now that we have discussed the experiments with which we have measured the relaxation and decoherence time, we can check if the times presented on the IBM site are compatible with the ones measured. The IBM presented only the mean values of these quantities, so we will use the mean weighted with the errors of our results to make the comparison.

We define the compatibility z[15] between two quantities $x \pm \sigma_x$ and $y \pm \sigma_y$ as follows:

$$z = \frac{|x - y|}{\sqrt{\sigma_x^2 + \sigma_y^2}}. \tag{3.9}$$

This quantity z is an estimator of how much probable is that x and y belong to the same Gaussian distribution. We define an optimal compatibility if $z < 1$, a good compatibility if $1 < z < 2$ and an awful compatibility if $2 < z < 3$. If $z > 3$ the two quantities are incompatible.

IBM values are given without an error estimation, so we apply Equation 3.9 with $\sigma_y = 0$. The result are shown in Table 3.3. All the quantities are incompatible: this is caused by the really small error and by the great fluctuations of these values over time: the IBM makes a calibration each week, and the values change each week, so we can assume that they aren't even stable over the day.

Quantity	Measured [μs]	IBM value [μs]	Compatibility
$T1$	75.120 ± 0.006	62.90	2036 ⇒Incompatible
$T2$	22.769 ± 0.008	22.70	8 ⇒ Incompatible
$T2^*$	21.94 ± 0.01	22.70	75 ⇒Incompatible

Table 3.3: In this table the weighted mean values of the measurement, the IBM values, namely the ones reported on the IBM site, and their compatibility are presented. They are incompatible, due to their fluctuations over time.

The quantum processor is now characterized, and we are ready to use it for the simulation of a specific model, a quantum elementary cellular automata. This is the object of the next chapter.

4

Quantum Cellular Automata

In this chapter, we use the IBM quantum processor to investigate the dynamics of a peculiar quantum system called Quantum Cellular Automata. We start from a brief overview of classical cellular automata to better understand the quantum one. Then we show how to simulate a quantum cellular automaton on the IBM processor, and try to understand whether the results are reliable.

4.1 Classical Cellular Automata

Cellular automata are dynamical systems that evolve on a discrete lattice[16]. Their global dynamics is controlled by simple local transition functions. Each spatial discrete unit of the lattice, called *site*, can be in one of a finite number of states. The evolved state of a site can be updated by using only local information relative to the site's neighborhood and the local transition function. A single evolution step of a Cellular Automata, called an iteration, is complete once the evolved state of each site has been computed and all sites have been updated simultaneously.

We define now the quantities involved in the Automata: we consider a 1-dimensional lattice of L sites, enumerated from 0 to $L-1$ and we denote the state of the i-th site at the iteration t as x_i^t. Each site can be found in one of k possible states in a given local space$Q$, where Q is the k-member local state space.

We suppose that each site has a contiguous neighborhood of N sites, namely the sites that

will be taken as an input for the evolution function. Also, the evolving state himself is in the neighborhood. In the following, we will assume N to be odd, but it isn't necessary to the theory.
The state of the i-th site is updated using the $N_i = \{i - \frac{N-1}{2}, \cdots, i, \cdots, i + \frac{N-1}{2}\}$ set of sites. This means that the state space of a neighborhood is an N-fold product space of the local state spaces $Q_N = Q^{\otimes N}$. The local transition function is a map $f : Q_N \to Q$. A complete iteration consists of applying f simultaneously to all the sites in the lattice:

$$x_i^{t+1} = f(x_{N_i}^t) \qquad \forall i. \tag{4.1}$$

It is important to note that updating all the sites simultaneously means that we implicitly need to make a copy of the state at the time t in order to compute the evolved state at the time $t + 1$.

We will work only with homogenous Cellular Automata, namely Automata in which the same local transition function is uniform along the lattice. There are k^{N^k} possible functions, and it is conventional to enumerate all the possible f with integers $R \in [0, k^{N^k} - 1]$ which can be used used to define the map: first we expand R into k^N digits of a base-k number. Then we enumerate the digits of this expansion from the least significant to the most significant with integers $r \in [0, k^N - 1]$. Then we have to expand r into N digits of base-k, and interpret the expansion of r as the neighborhood state which results in site i transitioning to the state given by the r-th bit of the R expansion.

We consider as an example the elementary cellular automata with $k = 2$, $N = 3$ on a 1-dimensional lattice of L sites. In this case there are $2^{2^3} = 256$ possible evolving functions. If we choose the rule $R = 6$ we must write 6 in the binary base using 7 bits, namely $6 = 00000110$ corresponding to the rule presented in Table 4.1. Now that we have understood

bit significance, $x_{N_i}^t$	111	110	101	100	011	010	001	000
rule number, x_i^{t+1}	0	0	0	0	0	1	1	0

Table 4.1: The update for rule R=6, an irreversible Elementary Cellular Automata. Notice that the bits in the second row are the number 6 in binary. Only the middle neighborhood is changed in the transformation, flipping the bit.

the base of classical cellular automata we can define its quantum counterpart.

It has been recently formulated a quantum formulation of the cellular automata called quantum cellular automata[16]. The difficulties in defining a Quantum Automata which adhere to the postulates of quantum mechanics. In this thesis, we will focus on the simplest quantum counterpart of the example defined in Section 4.1: the Quantum Elementary Cellular Automata (QECA). We define our system as a 1-dimensional lattice of L qubits. If $|\psi(0)\rangle$ is the initial state, then the state of the lattice at the time t is $|\psi(t)\rangle$. We are assuming that the system is isolated and the state remains coherent and pure, nevertheless, this hypothesis isn't satisfied. As we will see later the real quantum computers which are now available can't maintain a state pure for long times. We will show in which conditions the approximation of pure state can be done. The task now is to define the local transition function for QECA, namely adapting the numbering scheme of Cellular Automata to the quantum version. It must be a unitary operation, and therefore a reversible protocol. This requirement suggests that our QECA evolution scheme has to be based on what action we perform on the center site and that our numbering scheme needs only to consider the right and the left neighborhood, excluding the center site itself because we can't make a copy of it and then apply a Controlled-V operation due to the No cloning theorem[17]. We have $N = 2, k = 2$ resulting in $2^{2^2} = 16$ possible update rules.

As we did for the classical case, we list all the possible input configurations and then we assign to 1 the application of a single qubit operator V to the qubit i, and a 0 to the identity. Note that in the quantum case we can choose the function to apply to the bit among all the unitary gates, so we generally call it V. Table 4.2 gives an example of an update rule for $S = 6$. From the update table we can build a corresponding three qubits operator $U_S(V)$. The construction of $U_S(V)$ relies on expanding S into four digits of binary as $S = s_{11}2^3 +$

$\|i-1, i+1\rangle$	11	10	01	00
Rule numbering s_{mn}	0	1	1	0
Operator applied	$\mathbb{I}$	V	V	$\mathbb{I}$

Table 4.2: The update for rule R=6. We apply the operator V if the rule numbering is 1, and the identity in the other case. Note that the number in the second row are the binary representation of the number 6.

$s_{10}2^2 + s_{01}2^1 + s_{00}2^0$: all the information in S is encoded in a 2×2 matrix with elements

s_{mn}. So we can write $U_S(V)$ as:

$$U_S(V) = \sum_{m,n=0}^{1} |m\rangle \langle m| \otimes V^{mn} \otimes |n\rangle \langle n| , \tag{4.2}$$

where $V^{mn} = s_{mn}V + (1 - s_{mn})\mathbb{I}$.

Now we must define the iteration protocol. In the classical case all the local operators are applied at the same time, but in the quantum case it isn't possible. Due to the No Cloning Theorem it isn't possible to copy an arbitrary quantum state, so we must relax the requirement of a simultaneous update for each iteration. This means that we have to do the updates in a specific order. In our case, due to the experimental layout and the necessity of minimizing the application time, we will divide each iteration into three layers: we will first the operations to the sites which number i in such that $i \bmod 3 = 0$, than $i \bmod 3 = 1$ and at last $i \bmod 3 = 2$

The last element that we need to define is the boundary conditions. We will use two different sets of conditions: the first is *periodic boundary conditions*, showed in Figure 4.1, so that the site 0 and the site $L - 1$ are neighborhood. The second consider *open boundary conditions*, showed in Figure 4.2, so we add two auxiliary qubits with the state open to $|0\rangle$ at the ends of the lattice. Now that we have defined the QECA we can set up a protocol and run it on the real quantum processor in order to test whether this quantum computer is able to reproduce the dynamics of a given model.

4.3 Implementation on the IBM quantum processor

We have defined the QECA from a theoretical point of view: now we must define in detail the automata that we will run on the real processor. We will use the rule number 6, defined in Table 4.2. First of all, we need to decide the operator V to apply, if the rule numbering is 6. We must note that the rule 6 is easily achieved by each $Controlled - V$ operator such that $V^2 = \mathbb{I}$ with the circuit shown in Figure 4.3 as we see in Table 4.3. We choose V to be the X operator, which is the quantum counterpart of the classical NOT, and satisfy $X^2 = \mathbb{I}$. The CNOT has a mean application time, as reported in from Table 2.2 of $676ns$: applying V^{mn} requires $1352ns$ and a complete iteration, which requires 3 layers of V^{mn}, takes $4.056\mu s$. This means that, for example, $n = 15$ iterations require a total application time of $60.84\mu s$, which is far beyond the decoherence time presented in Table 3.3. We nevertheless consider

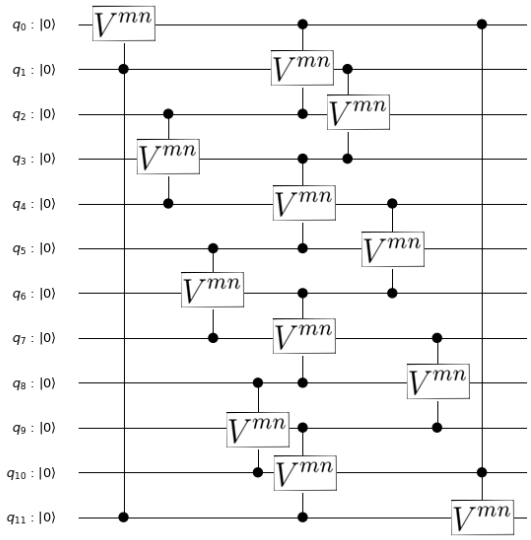


Figure 4.1: Quantum circuit with L=12 qubits where V^{mn} is the update rule. The boundary condition are perioc, so the qubit 0 is controlled by qubits 1 and 11 and qubit 11 is controlled by qubit 10 and 0. The update rule is applied first to the qubit which number i is such that i mod $3 = 0$, then i mod $3 = 1$ and at last i mod $3 = 2$.

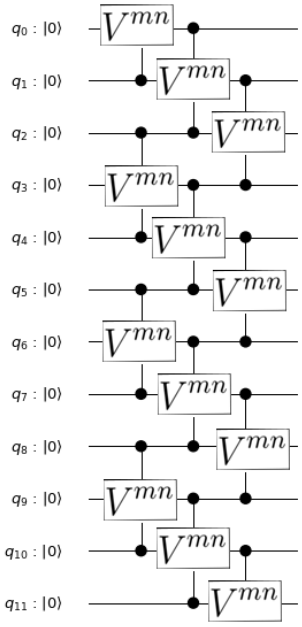


Figure 4.2: Quantum circuit with L=12 qubits where V^{mn} is the update rule. The boundary condition are open, so the qubit 0 is controlled only by qubits 1 and qubit 11 is controlled only by qubit 10. The update rule is applied first to the qubit which number i is such that i mod $3 = 0$, then$i$ mod $3 = 1$ and at last i mod $3 = 2$.

$\|i-1, i+1\rangle$	11	10	01	00
Rule numbering s_{mn}	0	1	1	0
C-V 1 application	yes	yes	no	no
C-V 2 application	yes	no	yes	no
Operator applied	$VV = \mathbb{I}$	$\mathbb{I}V = V$	$V\mathbb{I} = V$	$\mathbb{II} = \mathbb{I}$

Table 4.3: In this table we show how two Controlled-V operator such that $V^2 = \mathbb{I}$ disposed as shown in Figure 4.3 can implement a rule 6 QECA.

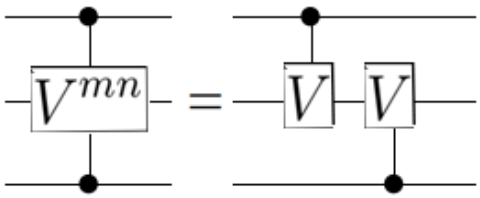


Figure 4.3: Composition of the gate V^{ms} that fulfills the rule 6 in Table 4.3. Two Controlled-V gates are used.

$n = 15$ iterations in order to observe the effects of the decoherence: nevertheless it is important to remember that only the first 5 iterations which require up to $20.28\mu s$. We have to define the length of the qubits lattice, and we will use three different lengths $L = 12, 6, 3$. The last element to define is the initial state: we will choose the simplest, a qubit in the middle of the lattice in the state $|1\rangle$ and all the other in the state $|0\rangle$. All this informations are summarized in Table 4.4.

Quantities	Definitions
Lattice length	$L = 12, 6, 3$
Number of iterations	$n = 15$
Update rule	$R = 6$
Update operator	X
Initial state	$\|\psi\rangle = \|0\rangle_0 \cdots \|0\rangle_{i-1} \|1\rangle_i \|0\rangle_{i+1} \cdots \|0\rangle_{L-1}$
Boundary conditions	Periodic conditions, open conditions

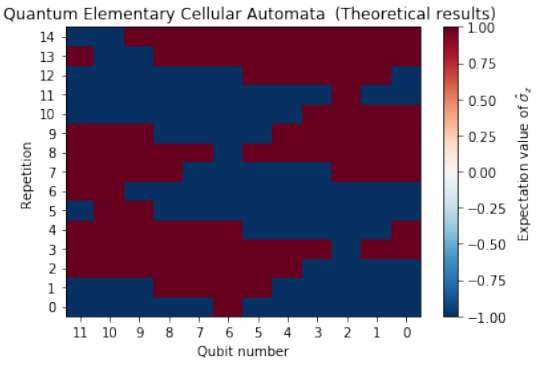
Table 4.4: Recap of the parameters of the QECA that will be run on the real processor.

We show now the way to read the graph in Figure 4.4, 4.5 and 4.6. On the y-axis there is the iteration, and on the x-axis there is the qubit number. The color code is linked to the expectation value of σ_z: the state $|0\rangle$ is linked to the value -1 and the state $|1\rangle$ to the value

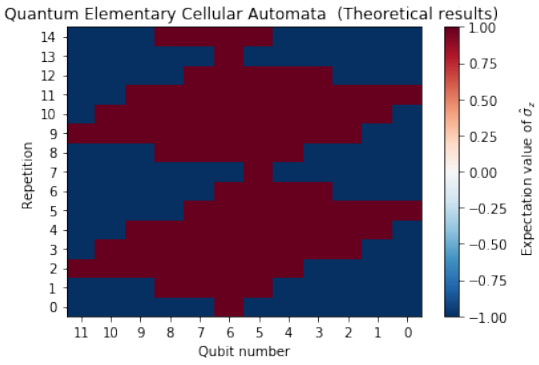
1. For example the state $|0\rangle$ blu and the state $|1\rangle$ red.

   We compute the dynamics pf the QECA in three different ways for each configuration:

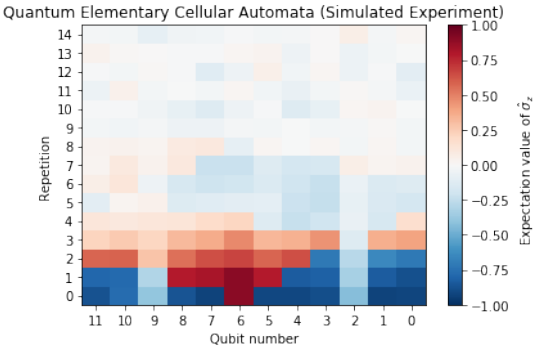
   1. The real experiment performed on the quantum processor;
   2. A numerical simulation of the experiment, which uses the noise model provided by the IBM, it takes into account the polarization, relaxation and decoherence effects described in Section 1.7;
   3. The numerical simulation of the isolated model, namely the results obtained by running the circuit on the simulator without any noise model, which corresponds the evolution of the pure state.



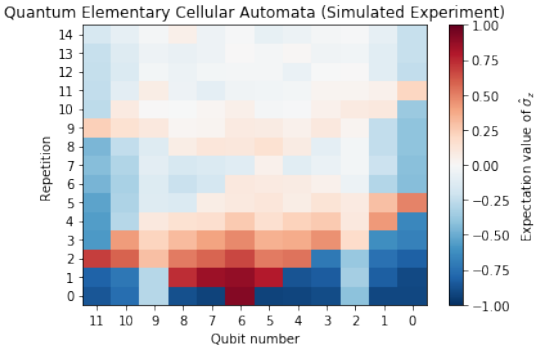
(a) Theoretical results for a 12 qubits lattice with periodic boundary conditions.



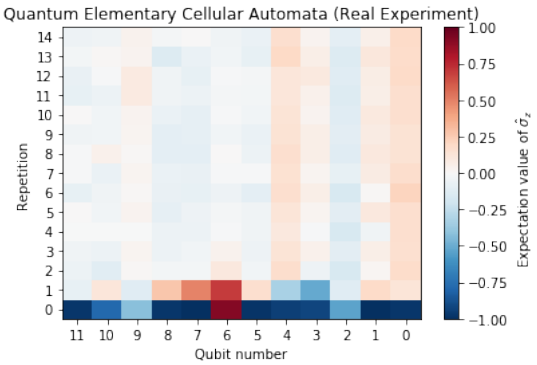
(b) Theoretical results for a 12 qubits lattice with open boundary conditions.



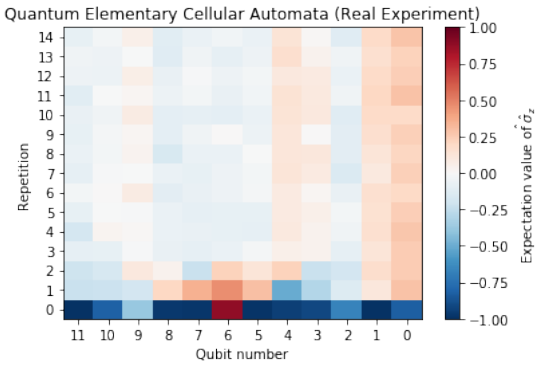
(c) Simulated experiment on a 12 qubits lattice with periodic boundary conditions.



(d) Simulated experiment on a 12 qubits lattice with open boundary conditions.

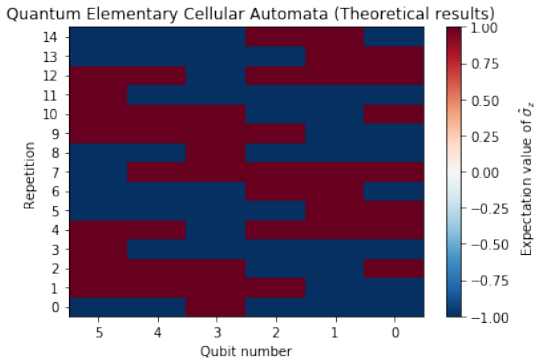


(e) Real experiment on a 12 qubits lattice with periodic boundary conditions.

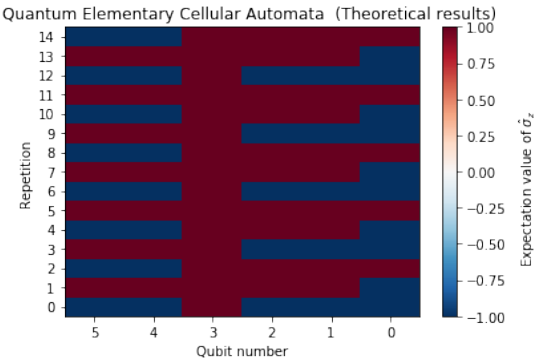


(f) Real experiment on a 12 qubits lattice with open boundary conditions.

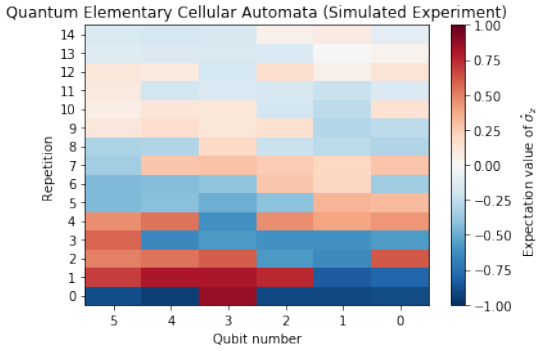
Figure 4.4: In the left column there are QECA with the periodic condition, on the right side the QECA with the open condition. On the y-axis there are the iterations, on the x-axis the qubit number and the color code is to indicates the expectation values of σ_z. It is important to remember that only the first five iteration have an application time lesser then the decoherence time.



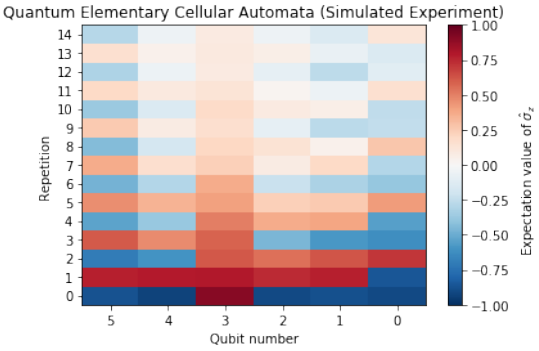
(a) Theoretical results for a 6 qubits lattice with periodic boundary conditions.



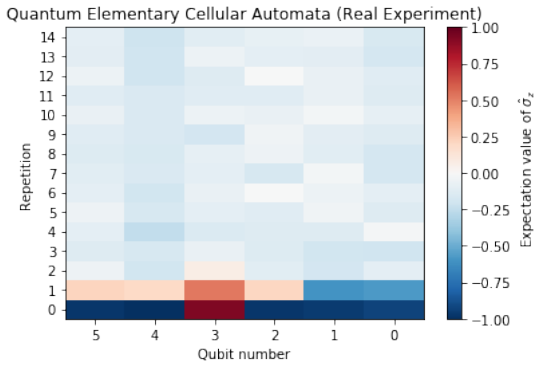
(b) Theoretical results for a 6 qubits lattice with open boundary conditions.



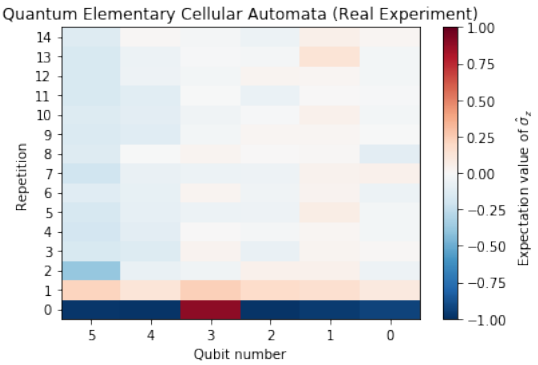
(c) Simulated experiment on a 6 qubits lattice with periodic boundary conditions.



(d) Simulated experiment on a 6 qubits lattice with open boundary conditions.

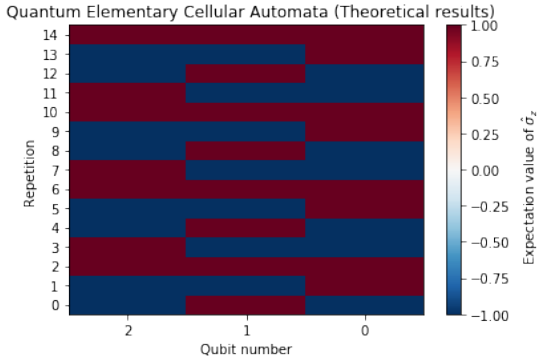


(e) Real experiment on a 6 qubits lattice with periodic boundary conditions.

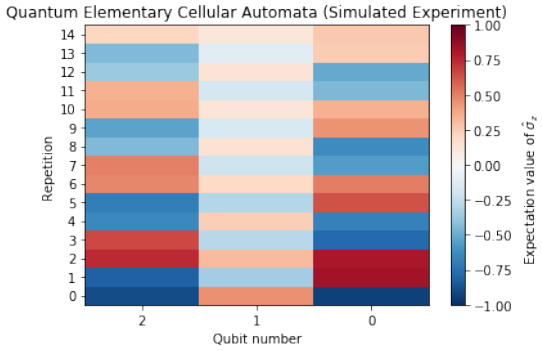


(f) Real experiment on a 6 qubits lattice with open boundary conditions.

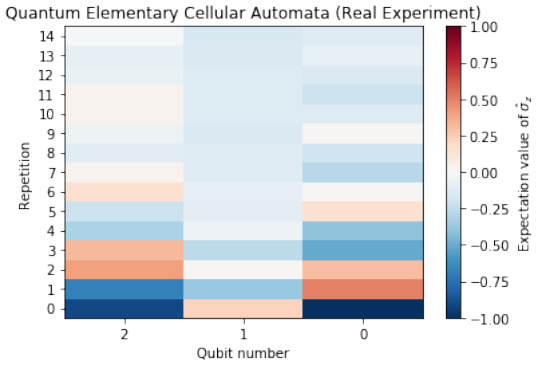
Figure 4.5: This is the lattice with 6 qubits. In the left column there are QECA with the periodic condition, on the right side the QECA with the open condition. On the y-axis there are the iterations, on the x-axis the qubit number and the color code is to indicates the expectation values of σ_z. It is important to remember that only the first five iteration have an application time lesser then the decoherence time.



(a) Theoretical results for a 3 qubits lattice with open boundary conditions.



(b) Simulated experiment on a 3 qubits lattice with open boundary conditions.



(c) Real experiment on a 3 qubits lattice with open boundary conditions.

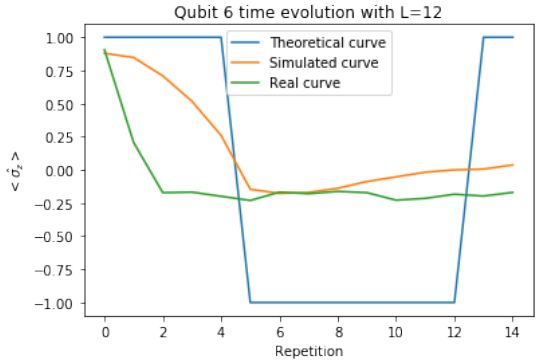
Figure 4.6: This is the lattice with 3 qubits. There is only the configuration with open boundary conditions because the coupling map of the real device doesn't support such circuit. On the y-axis there are the iterations, on the x-axis the qubit number and the color code is to indicates the expectation values of σ_z. It is important to remember that only the first five iteration have a application time lesser then the decoherence time.

We start by considering the case with $L = 12$: as we can see in Figure 4.4 there isn't any similarity between the real and the simulated results, apart what can be observed during the first two iterations. We do proceed with the $L = 6$ qubits lattice, in order to check whether that with a minor number of qubits errors would be less overwhelming, but as we see in Figure 4.5 no better results than the previous case is achieved. In the end, we try to the simplest system possible: a $L = 3$ qubits lattice, with open boundary conditions. As we see in Figure 4.6, in this case, the agreement between the simulations and the experiment until iteration 3 is slightly better than the other cases.

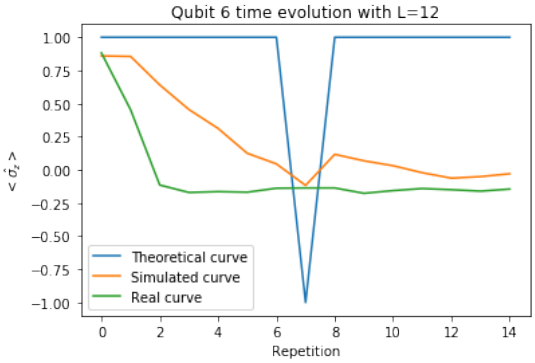
Another useful instrument that we can use to understand the actual limit of this quantum computer is to plot the time evolution of the expectation value of σ_z for single qubit: noting

the differences in the slope of the curve between the theoretical, the simulated and the real case. For each automata we choose to show the plot of the expectation value $< \sigma_z >$ relative to the only qubit excited in the initial state of the automata, as we see in Figure 4.7. We can see how the experimental curves rapidly decays towards the expectation value $< \sigma_z >= 0$. Figure 4.7d is particularly explicative: the theoretical curve is constant on $< \sigma_z >= 1$, the simulated curve slowly decays towards $< \sigma_z >= 0$ and the experimental curve drops to $< \sigma_z >= 0$ at the first iteration and after that remains almost constant. It shows the inability of the qubit of remaining in a coherent state, due to the difficulty of being isolated it from the environment.

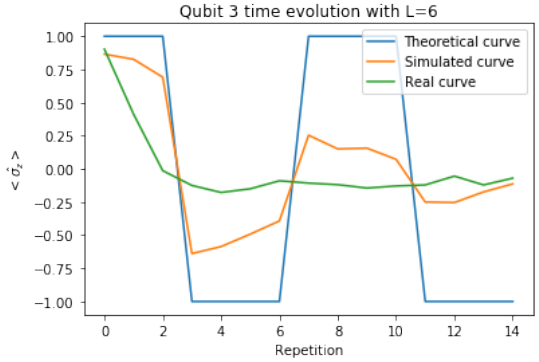
We can then deduce that the IBM quantum computer isn't able to run QECA properly.



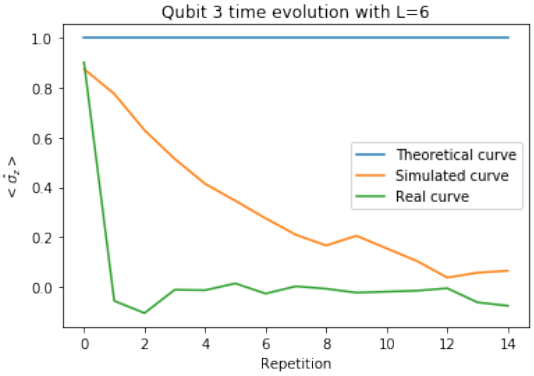
(a) 12 qubit lattice with periodic boundary conditions.



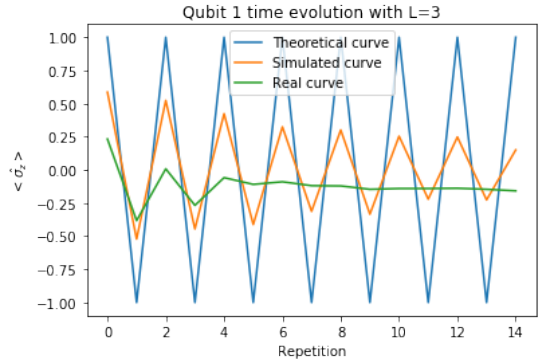
(b) 12 qubit lattice with open boundary conditions.



(c) 6 qubit lattice with periodic boundary conditions.



(d) 6 qubit lattice with open boundary conditions.



(e) 3 qubit lattice with open boundary conditions.

Figure 4.7: Curves relative to the only excited state qubit in the initial state of the automata. We indicates with $< \hat{\sigma_z} >$ the expectation value of $\hat{\sigma_z}$. We can notice how the real curve decays rapidly towards the state $\frac{1}{2}(|0\rangle \langle 0| + |1\rangle \langle 1|)$.

Conclusions

In this thesis, we have reviewed the basis of quantum computation, with particular attention to its fundamental unit: the qubit. We have shown how to describe a many-qubits system and their time evolution, both unitary and non-unitary. We focused on the non-unitary effects generated by the interactions with the environment, described by quantum channels.

We have then characterized the IBM quantum processor, describing the qubit physical implementation and its fundamental elements, namely the coupling map, the set of fundamental gates, their implementation time and the qubit gate error. We have presented the programming language developed by IBM to communicate with the quantum processor, Qiskit, and showed two examples of working programs.

After that, we measured the relaxation and the decoherence time of the 'ibmq_16_melbourne'. We compared them with the ones presented by the IBM on their site: the results are incompatible with the IBM values, due to the great fluctuations of these values over time.

In the end, we have defined the classical and quantum cellular automata. We found a protocol to implement the Quantum Elementary Cellular Automata with rule 6 on the quantum computer and ran it with different lattice length L and boundary conditions (open or periodic). We have seen that in all but the case of $L = 3$ the results of the experiments are incompatible with the theory after the second total application of the evolution rule. We showed the evolution of a single qubit of the lattice, understanding how fast the experimental evolution moves away from the theoretical and simulated ones.

In conclusion, we have characterized the current limits of IBM quantum computer, through the analysis of its characteristic times and the application of a quantum elementary cellular automata. This quantum computers avaiable is still too noisy to be usable for complex task: the operations that can be applied before the lost of coherence are not enough. Researchers and private companies, like IBM or Google[18], are however investing great quantities of money and effort in the improvement of quantum computers and so there is the possibility that they could become very powerful tools in not too many years.

References

[1] Nielsen, M. A. and Chuang, I. (2002), Quantum computation and quantum information.

[2] Higgott, O., Wang, D., and Brierley, S. (2018) Variational quantum computation of excited states. *arXiv preprint arXiv:1805.08138*.

[3] Shor, P. W. (1994) Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings 35th annual symposium on foundations of computer science*, pp. 124–134, Ieee.

[4] Rivest, R. L., Shamir, A., and Adleman, L. (1978) A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21, 120–126.

[5] Fano, U. (1957) Description of states in quantum mechanics by density matrix and operator techniques. *Reviews of Modern Physics*, 29, 74.

[6] Sheffer, H. M. (1913) A set of five independent postulates for boolean algebras, with application to logical constants. *Transactions of the American mathematical society*, 14, 481–488.

[7] Grover, L. K. (1996) A fast quantum mechanical algorithm for database search. *arXiv preprint quant-ph/9605043*.

[8] Tittel, W., Brendel, J., Zbinden, H., and Gisin, N. (1998) Violation of bell inequalities by photons more than 10 km apart. *Physical Review Letters*, 81, 3563.

[9] Einstein, A., Podolsky, B., and Rosen, N. (1935) Can quantum-mechanical description of physical reality be considered complete? *Physical review*, 47, 777.

[10] Benenti, G., Casati, G., and Strini, G. (2007) *Principles of quantum computation and information: Volume II: Basic Tools and Special Topics*. World Scientific Publishing Company.

[11] Koch, J., Terri, M. Y., Gambetta, J., Houck, A. A., Schuster, D., Majer, J., Blais, A., Devoret, M. H., Girvin, S. M., and Schoelkopf, R. J. (2007) Charge-insensitive qubit design derived from the cooper pair box. *Physical Review A*, 76, 042319.

[12] Martinis, J. M. and Osborne, K. (2004) Superconducting qubits and the physics of josephson junctions. *arXiv preprint cond-mat/0402415*.

[13] (2011), IBM Q 16 Melbourne.

[14] Qiskit Documentation.

[15] Maltoni, M. and Schwetz, T. (2003) Testing the statistical compatibility of independent data sets. *Physical Review D*, 68, 033020.

[16] Hillberry, L. E. (2016) *Entanglement and complexity in quantum elementary cellular automata*. Ph.D. thesis, Colorado School of Mines. Arthur Lakes Library.

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

[18] Mohseni, M., Read, P., Neven, H., Boixo, S., Denchev, V., Babbush, R., Fowler, A., Smelyanskiy, V., and Martinis, J. (2017) Commercialize quantum technologies in five years. *Nature*, 543, 171?174.