

**AGH University of Science and Technology**

---

Faculty of Computer Science, Electronics and Telecommunications

Department of Computer Science



MASTER OF SCIENCE THESIS

**ANALYSIS OF QUANTUM ERROR CORRECTION APPLICABILITY  
ON IBM-Q**

ANALIZA MOŻLIWOŚCI ZASTOSOWANIA KWANTOWEJ KOREKCJI  
BŁĘDÓW DLA PRZYKŁADOWYCH PROBLEMÓW NA KOMPUTERZE  
IBM-Q

**JULIA SYPIEŃ**

FIELD OF STUDY:  
Computer Science

SUPERVISOR:  
dr inż. Katarzyna Rycerz

---

Krakow, 2020

## Oświadczenie studenta

Uprzedzony(-a) o odpowiedzialności karnej na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2018 r. poz. 1191 z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, a także uprzedzony(-a) o odpowiedzialności dyscyplinarnej na podstawie art. 307 ust. 1 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.) „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta.”, oświadczam, że niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i nie korzystałem(-am) ze źródeł innych niż wymienione w pracy.

Jednocześnie Uczelnia informuje, że zgodnie z art. 15a ww. ustawy o prawie autorskim i prawach pokrewnych Uczelnia przysługuje pierwszeństwo w opublikowaniu pracy dyplomowej studenta. Jeżeli Uczelnia nie opublikowała pracy dyplomowej w terminie 6 miesięcy od dnia jej obrony, autor może ją opublikować, chyba że praca jest częścią utworu zbiorowego. Ponadto Uczelnia jako podmiot, o którym mowa w art. 7 ust. 1 pkt 1 ustawy z dnia 20 lipca 2018 r. – Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668 z późn. zm.), może korzystać bez wynagrodzenia i bez konieczności uzyskania zgody autora z utworu stworzonego przez studenta w wyniku wykonywania obowiązków związanych z odbywaniem studiów, udostępniać utwór ministrowi właściwemu do spraw szkolnictwa wyższego i nauki oraz korzystać z utworów znajdujących się w prowadzonych przez niego bazach danych, w celu sprawdzania z wykorzystaniem systemu antyplagiatowego. Minister właściwy do spraw szkolnictwa wyższego i nauki może korzystać z prac dyplomowych znajdujących się w prowadzonych przez niego bazach danych w zakresie niezbędnym do zapewnienia prawidłowego utrzymania i rozwoju tych baz oraz współpracujących z nimi systemów informatycznych.

.....  
(czytelny podpis studenta)

## Abstract

Quantum computers are one of the main areas of interest in computer science nowadays, thanks to their computational power. By launching the IBM Quantum Experience platform in 2016, IBM gave access to real quantum devices to internet users worldwide. Unfortunately, errors occurring in quantum computers are a severe obstacle to reliable quantum computations, and quantum error-correction is a field of study aimed at dealing with this problem. As part of this thesis, research among existing quantum error-correction methods was conducted, and two error-correcting codes were realized on IBM Q: the five-qubit perfect code and the three-qubit bit-flip code. To the best of the author's knowledge, it is the first implementation of the five-qubit perfect code on this quantum computing platform. Techniques such as randomized benchmarking and quantum state tomography were used to assess the effectiveness of implemented methods of error correction. The obtained results have shown that the three-qubit bit-flip code can improve the fidelity of quantum computations.

## Streszczenie

Komputery kwantowe cieszą się obecnie ogromnym zainteresowaniem ze względu na ich wyjątkowe możliwości obliczeniowe. W 2016 roku firma IBM uruchomiła platformę IBM Q Experience, dając tym samym dostęp do prawdziwych urządzeń kwantowych użytkownikom internetu na całym świecie. Niestety, występujący w tych urządzeniach szum kwantowy powoduje, że wyniki obliczeń uzyskiwanych na tych komputerach są obarczone dużymi błędami. Kwantowa korekcja błędów to dziedzina nauki obejmująca różne metody zwalczania błędów kwantowych. W ramach tej pracy wykonany został przegląd istniejących metod kwantowej korekcji błędów. Ponadto, przy użyciu platformy IBM Q zostały zaimplementowane dwa kwantowe kody korekcji błędów: „perfekcyjny” kod pięciobitowy i kod trzycybitowy „bit-flip”. Według najlepszej wiedzy autora jest to pierwsza implementacja „perfekcyjnego” kodu pięciobitowego na tej platformie. Skuteczność zaimplementowanych kodów została przetestowana przy użyciu technik takich jak randomizowane testy porównawcze i kwantowa tomografia stanu. Wyniki eksperymentów pokazały, że kod trzycybitowy „bit-flip” może poprawić wiarygodność wyników uzyskiwanych z komputerów kwantowych.

## Acknowledgments

*I would like to express gratitude to my supervisor, dr inż. Katarzyna Rycerz, for her guidance, patience, engagement, and immense knowledge that she is always willing to share. Her continuous support and valuable insights were a huge help for me in the process of writing this thesis.*

# Contents

<b>1</b>	<b>Introduction</b>	<b>8</b>
1.1	Motivation . . . . .	8
1.2	Related work . . . . .	9
1.3	Objectives of the thesis . . . . .	10
1.4	Structure of the work . . . . .	10
<b>2</b>	<b>Quantum computation and quantum errors</b>	<b>11</b>
2.1	Bra-ket notation . . . . .	11
2.2	Quantum state . . . . .	12
2.3	Density matrix . . . . .	15
2.4	Quantum operators . . . . .	15
2.5	Errors in quantum computers . . . . .	18
2.6	Summary . . . . .	20
<b>3</b>	<b>Quantum error-correcting codes</b>	<b>21</b>
3.1	Introduction and basic terms . . . . .	21
3.2	General structure . . . . .	22
3.3	Three-qubit bit-flip code . . . . .	24
3.4	Three-qubit phase-flip code . . . . .	26
3.5	Nine-qubit Shor's code . . . . .	29
3.6	The perfect five-qubit code . . . . .	34
3.7	Evaluation of quantum error-correcting codes . . . . .	38
3.8	Summary . . . . .	39
<b>4</b>	<b>Quantum Error Correction Methods Applications on IBM Q</b>	<b>40</b>
4.1	Quantum repetition codes . . . . .	40
4.2	Quantum error-detecting codes with post selection . . . . .	40
4.3	Nondestructive discrimination and automated error-correction of fully-entangled states . . . . .	41
4.4	Qiskit framework . . . . .	42
4.5	Summary . . . . .	43
<b>5</b>	<b>Solution</b>	<b>44</b>
5.1	Technologies . . . . .	44
5.2	New implementation of the perfect five-qubit code . . . . .	44
5.3	Implementation of the three-qubit bit-flip code . . . . .	54
5.4	Summary . . . . .	57

---

<b>6</b>	<b>Evaluation</b>	<b>58</b>
6.1	Experimental verification of correctness . . . . .	58
6.2	Evaluation with Randomized Benchmarking . . . . .	60
6.3	Summary . . . . .	66
<b>7</b>	<b>Summary, conclusions and future work</b>	<b>67</b>
7.1	Summary . . . . .	67
7.2	Conclusions . . . . .	67
7.3	Future work . . . . .	68
	<b>List of Figures</b>	<b>69</b>
	<b>List of Tables</b>	<b>71</b>
	<b>Bibliography</b>	<b>72</b>
<b>A</b>	<b>Appendix to Chapter 6</b>	<b>75</b>

---

# Chapter 1

## Introduction

### 1.1 Motivation

Quantum computing is one of the most popular and promising subjects of study nowadays. Due to their special characteristics, quantum computers may outcompete classical computers in dealing with many types of computational problems, especially those that require performing operations on voluminous data or performing calculations on big numbers. A well-known example demonstrating the power of quantum computers is how they can break public-key cryptographic systems, such as the widely used RSA (Rivest–Shamir–Adleman) cryptosystem, in polynomial time, using the Shor’s factorization algorithm [2].

The fundamental difference between quantum and classical computers is how a unit of information is stored in each of them. In classical computers, a unit of information is represented by a bit, which can be in either 0 or 1 logical state. In quantum computers, a single piece of information is stored in a qubit, which may be in one of the 0 and 1 logical states, but also in an arbitrary superposition of these states. It means that a qubit can be in both of its base states at the same time, from which the capabilities of quantum computers arise.

IBM is one of the companies that invest in quantum computing. It was the first to have launched, in 2016, a cloud computing platform, named IBM Quantum Experience (shortly IBM Q), through which real quantum devices can be accessed by anyone with stable access to the Internet. Since 2017, we have the ability to program these quantum computers with Qiskit [6], an open-source Python [9] framework. Thanks to IBM Q, more and more theoretically known quantum programs are realized on real quantum devices, including optimization algorithms [3] and quantum games [4].

Currently (as for 21st August 2020), there are ten quantum computers accessible through IBM Q Experience, one one-qubit quantum computer, eight five-qubit quantum computers, and one fifteen-qubit quantum computer. Also, a quantum simulator is available through the platform, which can simulate a quantum device of up to 32 qubits. Assessment of available software tools for programming IBM Q devices can be found in [5].

Besides all the advantages that come with the development of quantum com-



---

puters, and the fact that many quantum algorithms can at present be implemented and run on real quantum devices, some obstacles must be overcome to ensure the fidelity of the results from quantum computers. Quantum states in real quantum computers are very fragile, which is why information that they store can be easily lost. As a result, computations run on quantum computers are highly error-prone. Extensive research is conducted nowadays, in order to enable fault-tolerant quantum computations, and there is a lot to be done in this field. One way to mitigate the destructive influence of errors on computations run on quantum devices is the usage of quantum error-correction methods.

## 1.2 Related work

In this section, an overview of publications regarding the current state of knowledge on quantum error-correction and a quick review of the selection of publications presenting realizations of quantum error-correcting methods on IBM Q will be given. A more detailed study of existing quantum error correction applications on IBM Q is given in Chapter 4.

### 1.2.1 Theory of quantum error-correction

The first goal of this thesis is to conduct research among methods of quantum error-correction, especially on quantum error-correcting codes. There are scientific publications that cover this subject, including books and papers.

In [10], most of the known quantum error-correcting codes are described, with the theoretical background of these codes explained in detail. The author begins with basic examples of error-correcting codes, such as the three-qubit bit-flip code, and the three-qubit phase-flip code, followed by the nine-qubit Shor's code, capable of correcting an arbitrary error on a single qubit. Later on, he describes more advanced and powerful quantum error-correcting codes, named the stabilizer codes. Stabilizer codes were first proposed in [11] by D. Gottesman, and they form a more general family of quantum error-correcting codes, which includes, for example, the *perfect* five-qubit code or the seven-qubit Steane code. D. Gottesman presented an overview of quantum-error correction as well, in [12]. Another source of the theoretical knowledge on quantum error correction is chapter 5 of [13]. It gives a very clear explanation of some of the quantum error-correcting codes. Moreover, quantum circuits for these codes are presented, and their architecture is discussed in detail, which is helpful for practical applications of quantum error-correcting codes. Theoretical information on quantum error-correction can also be found in chapter 7 of [14]. Papers, which give an introduction to quantum error-correction are [15, 16].

### 1.2.2 Applications of quantum error-correcting methods on IBM Q

The second goal of this thesis is to analyze the applicability of quantum error-correction on IBM quantum computers. There are papers, which present applications of quantum error-correction methods on this platform.

---

A realization of repetition codes on IBM Q devices is shown in [17]. In this paper, the repetition codes are implemented on up to fifteen qubits and evaluated on both a real quantum device and a quantum simulator. Realizations of four-qubit quantum error-detecting codes were presented in [19, 20, 21, 22]. Quite a large number of implementations of this type of quantum codes on IBM Q emerges from the fact that it only demands five qubits, as stated in [18], and most of the real quantum devices available through the IBM Q Experience platform have only five qubits. On the other hand, in [24, 25] a completely different approach is shown. In these papers, implementations of quantum error-correcting programs specifically for fully entangled states, such as the Bell states, and based on these states' special features, are presented.

### 1.3 Objectives of the thesis

This thesis's first objective is to conduct **research among existing methods of quantum error-correction**, especially among quantum error-correcting codes, to find methods that can be realized on IBM Q devices. The limited number of available qubits on these devices and the impossibility to perform measurements multiple times during an execution of a quantum program must be taken into account.

Afterward, a **review of existing applications of quantum error-correction methods on IBM quantum computers** should be provided. Special attention will be paid to how the current limitations of IBM quantum devices can be overcome.

As a result of this research, a **new implementation of a quantum error-correction method on IBM Q should be realized**. Also, the **effectiveness of this method in protecting a quantum state in the presence of quantum noise should be assessed**. An appropriate method for evaluating the proposed solution must be chosen, taking into account a quantum nature of the program, and limitations of the IBM Q devices.

Another goal of this thesis is to **compare results for more than one method of quantum error-correction**, which would allow even deeper analysis. Various aspects should be regarded during the evaluation, such as the effectiveness in protecting quantum states, but the ease of implementation and understandability of a method, as well as the number of qubits and quantum operations needed for each method, should also be taken into account.

### 1.4 Structure of the work

In Chapter 2 the basic terms related to quantum computing are introduced, and also an overview of errors in quantum computers is given. Chapter 3 gives an introduction to quantum error-correcting codes and also to some of the available methods for evaluating such codes. Moreover, examples of quantum error-correcting codes are discussed. In Chapter 4, an overview of existing applications of quantum error-correcting codes is presented. In Chapter 5 a new implementation of the perfect five-qubit code is presented, and it's evaluated in Chapter 6. Chapter 7 concludes the thesis and gives thoughts on future work.

---

# Chapter 2

## Quantum computation and quantum errors

In this chapter, basic terms related to quantum computing will be introduced. Also, an overview of errors that occur in quantum computers will be given.

### 2.1 Bra-ket notation

The bra-ket notation, also known as the Dirac's notation after it's inventor, was first introduced in [26]. It's used in quantum mechanics to describe states in quantum systems. Quantum states are vectors from a Hilbert space over the field of complex numbers. Primarily, such vectors are denoted as column vectors.

In the bra-ket notation, a vector is written in between the  $|$  and  $\rangle$  characters, and can be referred to as *ket*. For example, one of the base states of a qubit, the zero state, can be written as

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \quad (2.1)$$

In general, a  $\psi$  vector representing a quantum  $n$ -qubit state in the bra-ket notation will be written as

$$|\psi\rangle = \begin{pmatrix} \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \psi_n \end{pmatrix}. \quad (2.2)$$

A hermitian conjugate of a quantum state in the bra-ket notation is written between the  $\langle$  and  $|$  characters, and it's called *bra*.

$$\langle\psi| = (\psi_1^* \quad \psi_2^* \quad \cdot \quad \cdot \quad \cdot \quad \psi_n^*), \quad (2.3)$$

where  $\psi_i^*$  is the complex conjugate of  $\psi_i$ .

---

The bra-ket notation can also be used to express some of the operations on vectors representing quantum states. The **scalar product** of two quantum states  $\phi$  and  $\psi$  can be written as

$$\langle\phi|\psi\rangle = (\phi_1^* \quad \phi_2^* \quad \cdot \quad \cdot \quad \cdot \quad \phi_n^*) \begin{pmatrix} \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \psi_n \end{pmatrix}. \quad (2.4)$$

On the other hand, the **outer product**, or matrix multiplication, of these vectors can be written as

$$|\phi\rangle\langle\psi| = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_n \end{pmatrix} (\psi_1^* \quad \psi_2^* \quad \cdot \quad \cdot \quad \cdot \quad \psi_n^*). \quad (2.5)$$

Moreover, the **tensor product** of these vectors can be denoted with

$$|\phi\rangle|\psi\rangle = |\phi\psi\rangle = \begin{pmatrix} \phi_1 \\ \phi_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_n \end{pmatrix} \otimes \begin{pmatrix} \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \psi_n \end{pmatrix} = \begin{pmatrix} \phi_1\psi_1 \\ \phi_1\psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_2\psi_1 \\ \phi_2\psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_n\psi_1 \\ \phi_n\psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \phi_n\psi_n \end{pmatrix}. \quad (2.6)$$

## 2.2 Quantum state

Quantum computers differ from their classical counterparts in the way that information is represented in each of them. Also, they differ in how this stored information can be operated on. In this section, a brief note on what quantum states and operations are will be given. A more in-depth discussion on this topic can be found in chapter 1 of [13].

---

### 2.2.1 Quantum unit of information

In classical computers, the unit of information is the bit. A bit can be in one of two possible states, represented as 0 and 1. On the other hand, in quantum computation, the unit of information is a **qubit**, and two base, orthogonal states of a qubit are  $|0\rangle$  and  $|1\rangle$ .

In contrast to a classical bit, a qubit can not only be in one of its base states, but its state can be any unit vector from the two-dimensional, vector space spanned by the base states. A general form which describes a state of a qubit,  $|\psi\rangle$ , is

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \longleftrightarrow \begin{pmatrix} \alpha \\ \beta \end{pmatrix}, \quad (2.7)$$

where  $\alpha$  and  $\beta$  are complex numbers, satisfying the condition

$$|\alpha|^2 + |\beta|^2 = 1. \quad (2.8)$$

It can be said that the state  $|\psi\rangle$  is a *superposition* of the base states  $|0\rangle$  and  $|1\rangle$  with *amplitudes*  $\alpha$  and  $\beta$ .

### 2.2.2 Multi-qubit quantum state

In classical computers, an ordered set of  $n$  bits is called a *word*. For example, the possible, classical, two-bit words are

$$00, 01, 10, 11, \quad (2.9)$$

or in the bra-ket notation

$$|00\rangle, |01\rangle, |10\rangle, |11\rangle. \quad (2.10)$$

In general, there are  $2^n$  possible words of length  $n$ , representing  $n$ -bit classical states, and a classical system comprising of  $n$  bits can be in one of them at a time. The  $n$ -bit words can be obtained as all possible tensor products of length  $n$  of the one-bit states. For example, in the case of the state  $|01\rangle$

$$|01\rangle = |0\rangle \otimes |1\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}. \quad (2.11)$$

The  $n$ -qubit quantum state, on the other hand, can be an arbitrary superposition of the  $2^n$  possible classical states

$$|\Psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle_n, \quad (2.12)$$

---

so that  $\alpha_x$  are complex numbers satisfying the condition

$$\sum_{x=0}^{2^n-1} |\alpha_x|^2 = 1. \quad (2.13)$$

For example, a general form of the two-qubits quantum states is

$$|\psi\rangle = \alpha_0 |00\rangle + \alpha_1 |01\rangle + \alpha_2 |10\rangle + \alpha_3 |11\rangle, \quad (2.14)$$

where

$$|\alpha_0|^2 + |\alpha_1|^2 + |\alpha_2|^2 + |\alpha_3|^2 = 1. \quad (2.15)$$

### 2.2.3 Entanglement

An attentive reader may notice that the general form of a multi-qubit quantum state shown in section 2.2.2 does not restrict it to be a tensor product of one-qubit quantum states. For example, the two-qubit state

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) + 0(|01\rangle + |10\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \quad (2.16)$$

is a valid quantum state, but no two single-qubit quantum states have the tensor product equal to this state.

In general, there are multiple-qubit quantum states which can't be expressed through the states of the individual, component qubits of these states, and we call them *entangled states*. If two qubits are entangled, the state of one of them implies the state of the other one.

Among entangled states, there are *maximally entangled states*, for which the state of one of the component qubits determines the states of *all* other qubits in this state. Maximally entangled two-qubit states are called the **Bell's states**. The states are

$$|\Psi_{00}^{\pm}\rangle = \frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle), \quad (2.17)$$

$$|\Psi_{01}^{\pm}\rangle = \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle). \quad (2.18)$$

The maximally entangled three-qubit states are called Greenberger–Horne–Zeilinger (shortly GHZ) states [28]. The states are

$$|\Psi_{000}^{\pm}\rangle = \frac{1}{\sqrt{2}}(|000\rangle \pm |111\rangle), \quad (2.19)$$

---


$$|\Psi_{001}^{\pm}\rangle = \frac{1}{\sqrt{2}}(|001\rangle \pm |110\rangle), \quad (2.20)$$

$$|\Psi_{010}^{\pm}\rangle = \frac{1}{\sqrt{2}}(|010\rangle \pm |101\rangle), \quad (2.21)$$

$$|\Psi_{011}^{\pm}\rangle = \frac{1}{\sqrt{2}}(|011\rangle \pm |100\rangle). \quad (2.22)$$

## 2.3 Density matrix

Quantum states which can be expressed with a ket are ideal quantum states, called *pure* quantum states. In real quantum computers, however, a quantum state can be represented by some *classical* probability distribution over pure states and such a quantum state is called a *mixed* quantum state. To describe both pure and mixed quantum states, the *density matrix* formalism can be used. The density matrix for a pure quantum state  $|\psi\rangle$  is defined as

$$\rho = |\psi\rangle\langle\psi|. \quad (2.23)$$

On the other hand, a density matrix for a mixed state, which can be one of  $n$  pure states with some probabilities, is defined as

$$\rho = \sum_{i=1}^n p_i |\psi_i\rangle\langle\psi_i|, \quad (2.24)$$

so that

$$\sum_{i=1}^n p_i = 1, \quad (2.25)$$

where the pure state  $\psi_i$  occurs with classical probability  $p_i$ .

## 2.4 Quantum operators

A quantum state of  $n$  qubits can be transformed into another quantum state of  $n$  qubits with the use of quantum operators, which are also called *quantum gates*. The reversibility of quantum computation draws from the fact that quantum operators must be *unitary* operators. An operator  $U$  is unitary if and only if

$$U^\dagger U = U U^\dagger = I \iff U^\dagger = U^{-1}, \quad (2.26)$$

where  $U$  and  $I$  are of the same dimensions. It can be noticed that, by definition, the *inverse* of a unitary operator always exists.

Quantum operators have matrix representations, but also they have graphical representations. In general, quantum programs are often presented as *quantum circuits*, where each qubit is embodied by a horizontal line, and operators by named boxes.

Examples of single and multi-qubit operators, with their matrix and graphical representations, are shown in Table 2.1 and 2.2 respectively.

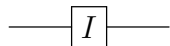
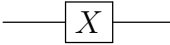
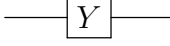
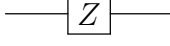
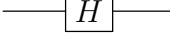
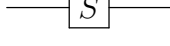
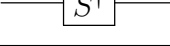
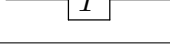
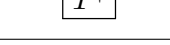
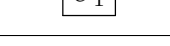
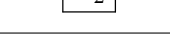
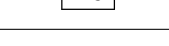
Operator	Matrix	Gate
I	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	
X	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	
Y	$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$	
Z	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$	
H	$\begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{bmatrix}$	
S	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$	
$S^\dagger$	$\begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$	
T	$\begin{bmatrix} 1 & 0 \\ 0 & \frac{1+i}{\sqrt{2}} \end{bmatrix}$	
$T^\dagger$	$\begin{bmatrix} 1 & 0 \\ 0 & \frac{1-i}{\sqrt{2}} \end{bmatrix}$	
$U_{1(\theta)}$	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{bmatrix}$	
$U_{2(\theta,\gamma)}$	$\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{e^{i\theta}}{\sqrt{2}} \\ \frac{e^{i\gamma}}{\sqrt{2}} & \frac{e^{i\theta+i\gamma}}{\sqrt{2}} \end{bmatrix}$	
$U_{3(\theta,\gamma,\phi)}$	$\begin{bmatrix} \cos \frac{\phi}{2} & -e^{i\theta} \sin \frac{\phi}{2} \\ e^{i\gamma} \sin \frac{\phi}{2} & e^{i\theta+i\gamma} \cos \frac{\phi}{2} \end{bmatrix}$	

Table 2.1: Single-qubit quantum operators.



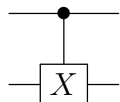
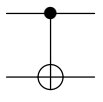
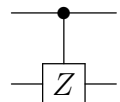
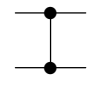

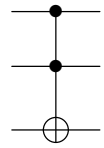
Operator	Matrix	Gate	Alternative
controlled-NOT	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$		
controlled-Z	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$		
SWAP	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$		
Toffoli	$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$		

Table 2.2: Multi-qubit quantum operators.

Important examples of single-qubit quantum operators are the  $X$ ,  $Y$  and  $Z$ , called the **Pauli operators** [27]. Together with the identity operator,  $I$ , the Pauli operators form a *basis* in the space of  $2 \times 2$  quantum operators.

An application of quantum operator  $A$  on a quantum state  $|\psi\rangle$  is denoted as

$$A|\psi\rangle. \tag{2.27}$$

If the state is represented by a density matrix

$$\rho = |\psi\rangle\langle\psi|, \tag{2.28}$$

then an application of quantum operator  $A$  on this state is denoted as

$$A\rho A^\dagger. \tag{2.29}$$

---

### 2.4.1 Measurement

The only non-reversible operation in quantum computations is the state *measurement*. As the *Bohr's rule* states, when a measurement is performed on a  $n$ -qubit state  $|\psi\rangle$ ,

$$|\psi\rangle = \sum_{x=0}^{2^n-1} \alpha_x |x\rangle, \quad (2.30)$$

where  $|x\rangle_0 \dots |x\rangle_n$  form a basis in the space of  $n$ -qubit space, then with probability

$$p_x = |\alpha_x|^2, \quad (2.31)$$

the result of the measurement will be  $|x\rangle$ .

Just as a quantum state can be expressed with respect to different bases, a measurement can also be performed in different bases. If a measurement of a single qubit's state can result in either state  $|0\rangle$  or  $|1\rangle$ , such a measurement is called a Z basis, or computational basis, measurement.

## 2.5 Errors in quantum computers

Both classical bits and qubits, in classical and quantum computers respectively, are exposed to the influence of their outer environments, for example to temperature fluctuations or collisions with surrounding particles [13].

In the case of classical bits, the size of their physical representation, huge compared to the size of an atom, makes them almost immune to such interactions. The approximate size of a transistor in current computers is  $10^{-8}m$  [32], while the size of an atom is around  $10^{-10}m$ . Thus, the probability of a classical bit flipping from one of its possible states to the other is minuscule. On the other hand, physical qubits are of an order of magnitude of atoms, making them highly responsive to even the slightest changes in their surroundings. As a result, a piece of information stored in a qubit can easily be lost. Full isolation of qubits in quantum computers would solve the problem but is impossible. Even though states in existing quantum devices, for example in IBM Q devices, are well isolated and kept in immensely low temperatures, of an order of magnitude of 0.01 Kelvin [31], it is not enough to guarantee fault-tolerant quantum computations.

### 2.5.1 Quantum channel

The environment where we store or transmit quantum information may be referred to as a **quantum channel**. Ideally, a quantum state represented by a density matrix  $\rho$ , sent through a quantum channel  $E$  remains untouched

$$E(\rho) = \rho. \quad (2.32)$$

---

In real quantum devices, however, this is usually not the case. Instead, with some *classical probability*  $p$  a random, unitary operator  $A$  is applied to the initial state causing a **quantum error**, and with probability  $(1 - p)$  the state is preserved

$$E(\rho) = (1 - p)\rho + pA\rho A^\dagger. \quad (2.33)$$

Because the error occurs with some classical probability, the result quantum state is also described by a classical probability of the original and erroneous quantum states occurring. That means that a *mixed* quantum state is obtained.

There are some quantum error channels that can be distinguished from others.

### 2.5.2 Bit-flip channel

Once a quantum state is sent through a *bit-flip channel*, with some probability  $p$  the  $X$  Pauli operator is applied to this state

$$E(\rho) = (1 - p)\rho + pX\rho X^\dagger. \quad (2.34)$$

The influence of this operator on a single-qubit quantum state is as follows,

$$\begin{aligned} X|0\rangle &= |1\rangle, \\ X|1\rangle &= |0\rangle, \\ |\psi\rangle = \alpha|0\rangle + \beta|1\rangle &\longrightarrow X|\psi\rangle = \beta|0\rangle + \alpha|1\rangle. \end{aligned} \quad (2.35)$$

The bit-flip channel is the only one that has an equivalent in classical computers.

### 2.5.3 Phase-shift channel

In a *phase-shift channel*, with some probability the *phase* of the original state is changed by an angle  $\theta$

$$E(\rho) = (1 - p)\rho + pR_\theta\rho R_\theta^\dagger. \quad (2.36)$$

where the operator  $R_\theta$  is of the form

$$R_\theta = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\theta} \end{pmatrix}, \quad (2.37)$$

and its impact on a single-qubit is

$$\begin{aligned} R_\theta|0\rangle &= |0\rangle, \\ R_\theta|1\rangle &= e^{i\theta}|1\rangle, \\ |\psi\rangle = \alpha|0\rangle + \beta|1\rangle &\longrightarrow R_\theta|\psi\rangle = \alpha|0\rangle + e^{i\theta}\beta|1\rangle. \end{aligned} \quad (2.38)$$

An important example of a phase-shift is a *phase-flip* when the angle by which the phase is changed is equal to  $\pi$ , and the rotation operator is the  $Z$  Pauli operator

---


$$E(\rho) = (1 - p)\rho + pZ\rho Z^\dagger. \quad (2.39)$$

As a result, the initial qubit state changes the following way

$$\begin{aligned} Z|0\rangle &= |0\rangle, \\ Z|1\rangle &= -|1\rangle, \\ |\psi\rangle = \alpha|0\rangle + \beta|1\rangle &\longrightarrow Z|\psi\rangle = \alpha|0\rangle - \beta|1\rangle. \end{aligned} \quad (2.40)$$

### 2.5.4 Depolarizing channel

In general, a quantum error channel where with probabilities  $p_x$ ,  $p_z$  and  $p_y$  Pauli operators  $X$ ,  $Z$  and  $Y$ , respectively, are applied, and with probability  $1 - (p_x + p_z + p_y)$  the state remains unchanged, is called a **Pauli channel**

$$E(\rho) = (1 - (p_x + p_z + p_y))\rho + p_x X\rho X^\dagger + p_z Z\rho Z^\dagger + p_y Y\rho Y^\dagger. \quad (2.41)$$

A special type of a Pauli channel where  $p_x = p_z = p_y$  is called the **depolarizing channel**,

$$E(\rho) = (1 - p)\rho + \frac{p}{3}(X\rho X^\dagger + Z\rho Z^\dagger + Y\rho Y^\dagger), \quad (2.42)$$

where  $p = p_x + p_z + p_y$ . In case if  $p = 1$ , we call such a channel a *completely depolarizing channel*. On the other hand, if  $p < 1$ , we call such a channel a *partially depolarizing channel*.

## 2.6 Summary

In this chapter, basic terms, which are used further in this work, were explained. Also, the cause of noise in quantum computers was discussed, and an overview of quantum errors was given. The next chapter focuses on quantum error-correcting codes.

---

# Chapter 3

## Quantum error-correcting codes

In this section, an introduction to quantum error-correcting codes will be given, and examples of such codes will be presented. Also, a selection of methods that can be used for evaluating quantum error-correcting codes will be described.

### 3.1 Introduction and basic terms

A *quantum error-correcting code* is a sequence of quantum operations able to correct quantum errors that impact the quantum state passed to the code. With these codes, the original quantum state is encoded in such a way that information needed to detect and correct an error can be obtained from the encoded state. There are common steps that can be distinguished in the majority of quantum error-correcting codes.

#### 3.1.1 Notation

Usually, quantum codes take the initial,  $k$ -qubit quantum state and transform it into its representation as a  $n$ -qubit quantum state, where  $k \leq n$ . Such a quantum code is denoted as

$$[[n, k, d]], \tag{3.1}$$

where  $d$  represents the *distance* of the code. For two  $n$ -qubit quantum states, the *Hamming distance* is defined as the minimum number of qubits on which a quantum operator must be applied to obtain one state from the other, and the *distance*  $d$  of a  $[[n, k, d]]$  quantum error-correcting code, is defined as the minimum Hamming distance between a pair of valid codewords for that code.

#### 3.1.2 Syndrome measurement and error syndrome

In quantum programs, qubits can be measured only as the final operation. In quantum error-correcting codes, errors are detected without performing a measurement, only with the use of other quantum operators, and with the help of *ancilla*

---

*qubits*. Ancilla qubits are additional qubits which don't store the encoded quantum state, but some other information based on which appropriate action can be chosen to restore the original quantum state. The process of gathering the needed information in ancilla qubits is called the *syndrome measurement*, and the quantum state stored in ancilla qubits, after the syndrome measurement is performed, is called the *error syndrome*.

### 3.1.3 Quantum Hamming bound

If a quantum error-correcting code uses unique error syndromes to detect Pauli  $X$ ,  $Y$ , and  $Z$  errors on each code qubit, then the *quantum Hamming bound* defines the minimum number,  $n$ , of qubits on which a  $k$ -qubit state can be encoded to correct an arbitrary, quantum error on at least  $t$  qubits. For such a code, the condition

$$\sum_{j=0}^t \binom{n}{j} 3^j 2^k \leq 2^n \quad (3.2)$$

must be satisfied [10].

In case of a code able to correct an arbitrary, quantum error on a single qubit, and if the original state is a one-qubit state, and therefore  $t = 1$  and  $k = 1$ , the condition is of the form

$$2(1 + 3n) \leq 2^n \quad (3.3)$$

Quantum error-correcting codes that use the minimum number of qubits  $n$  satisfying the quantum Hamming bound for given  $k$  and  $t$ , able to correct arbitrary quantum errors on  $t$  qubits, are called the *perfect codes*.

### 3.1.4 Codewords

For a given  $[[n, k, d]]$  error-correcting code, *codewords* are all,  $n$ -qubit quantum states, which can be obtained in the process of encoding a  $k$ -qubit quantum state with this code. If the initial base states are  $|\psi_1\rangle$  and  $|\psi_2\rangle$ , then the all codewords can be expressed as superpositions of new,  $n$ -qubit, encoded base states, which are denoted as  $|\overline{\psi_1}\rangle$  and  $|\overline{\psi_2}\rangle$  respectively. The encoded base states are called the *logical base states*.

## 3.2 General structure

In most of the quantum error-correcting codes, there are several common steps that can be distinguished. A circuit visualizing a general quantum error-correcting code is shown in Figure 3.1, where the initial  $k$ -qubit state is encoded onto  $n$  qubits, and  $m$  ancilla qubits are used.

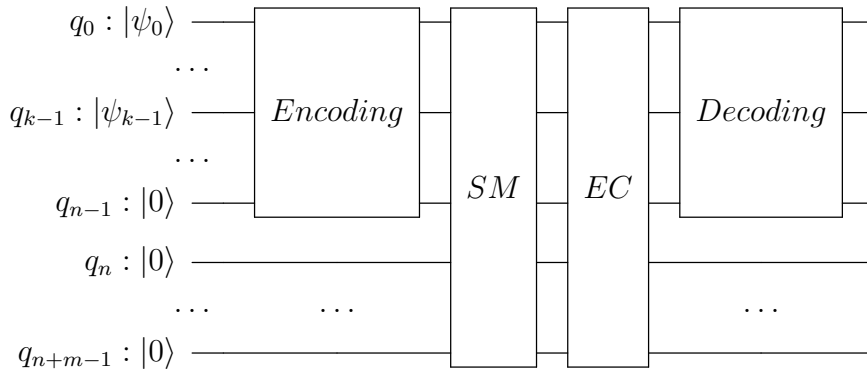


Figure 3.1: General structure of a quantum error-correcting code. SM stands for Syndrome Measurement, and EC for Error Correction.

### 3.2.1 State encoding

Usually, quantum error-correcting codes begin with the encoding of the initial state to a larger Hilbert space. The initial state may be simply repeated across a larger number of qubits, but also a more complex encoding may be performed. A  $[[n, k, d]]$  code encodes a  $k$ -qubit states to  $n$ -qubit codewords.

### 3.2.2 Syndrome measurement

*Syndrome measurement* is the procedure where errors are detected. Despite its name, no actual measurement, in the sense of collapsing a quantum state to one of the base states and reading its value, is performed. It would destroy the superposition of the original state if it was present. Moreover, when it comes to existing quantum devices, e. g. those available through the IBM Q Experience platform, further operations on a circuit are not possible after the final measurement was performed. Instead, *error syndrome* is obtained through the application of adequate quantum gates, often controlled gates, and it is usually stored in *ancilla* qubits. For example, information on the error that occurred can be deduced by comparing phases and parity of code qubits. As a result, an error is detected, whereas the procedure does not know the actual value of the code qubits; thus, the routine works regardless of the initial state. The measured error syndrome must clearly indicate the quantum operation which must be applied to recover the correct state.

In the case of all codes mentioned in this work, ancilla qubits are always initialized with the  $|0\rangle$  state.

### 3.2.3 Error correction

Based on the syndrome measurement stored in ancilla qubits, the error-correcting procedure applies adequate quantum operations to correct the erroneous state if an error has been detected. When considering quantum errors that can be represented

---

as unitary operators, these errors can be removed by applying the same operator once again.

### 3.2.4 State decoding

At the end, the initial state, which was passed to the procedure, is recovered from the encoded, corrected state. After detection and correction routines, the encoded state is expected to be the same as right after the encoding procedure. As quantum gates are unitary operators, the state can be decoded by applying the encoding procedure backward.

## 3.3 Three-qubit bit-flip code

The three-qubit bit-flip code is a very simple example of a quantum error-correcting code. It protects a single qubit's state against bit-flip errors [10].

### 3.3.1 State encoding

The three-qubit bit-flip code takes a single qubit's state,  $|\psi\rangle$ , as input,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \quad (3.4)$$

and encodes it as

$$|\psi'\rangle = \alpha |000\rangle + \beta |111\rangle. \quad (3.5)$$

Repeating the base states across three qubits is done using *cNOT* gates. The theoretical circuit performing this procedure can be found e. g. in [13, 10], and is illustrated in Figure 3.2.

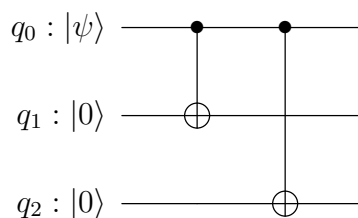


Figure 3.2: Quantum state encoding for the three-qubit bit-flip code.

### 3.3.2 Syndrome measurement

The syndrome measurement protocol for the three-qubit bit-flip code uses two ancilla qubits. The first ancilla qubit stores the result of parity check of the first and the second code qubits, and the second ancilla the result of parity check of the second and the third code qubits.



The parity check of two qubits is done by applying two  $cNOT$  operators, both with the ancilla as the target qubit, and one of the compared qubits as the control. If the states are different, the ancilla qubit is flipped once and will be in the state  $|1\rangle$ . If both of the states are equal, the ancilla state will be  $|0\rangle$ , as it will either be flipped zero or two times. The circuit for this procedure can be found in [13], and is shown in Figure 3.3.

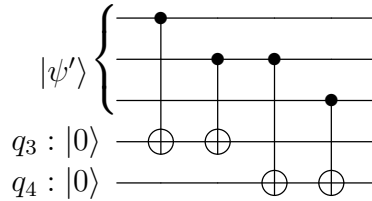


Figure 3.3: Syndrome measurement circuit for the three-qubit bit-flip code.

Equivalently, the circuit can be realized as shown in Figure 3.4, by substituting quantum operations with their identities, as presented in [13]. The transformed circuit stands a good basis for designing the syndrome measurement procedure for other, more complex quantum error-correcting codes.

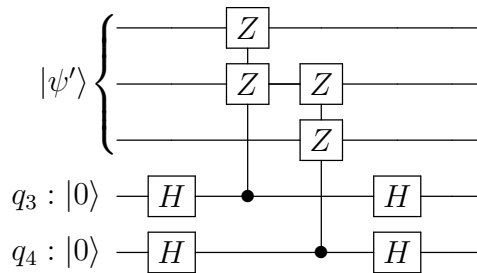


Figure 3.4: Syndrome measurement circuit for the three-qubit bit-flip code after transformations.

With  $Z_0Z_1$  and  $Z_1Z_2$  describing the results of the parity check of qubits 0 with 1, and 1 with 2, respectively, the interpretation of error syndrome for this code is as shown in Table 3.1, where  $X_i$  denotes a bit-flip on  $i_{th}$  qubit.

	$X_0$	$X_1$	$X_2$	I
$Z_0Z_1$	1	1	0	0
$Z_1Z_2$	0	1	1	0

Table 3.1: Interpretation of syndrome measurement for the three-qubit bit-flip code.

---

### 3.3.3 Error correction

Based on the measured error syndrome, appropriate action should be taken to recover the correct state. *CNOT* and Toffoli gates can be used for that purpose. An example procedure is illustrated in Figure 3.5 [13]. In the case if the error syndrome is either  $|10\rangle$  or  $|01\rangle$ , either the first or the second *cNOT* gate flips the erroneous qubit back to the correct state. However, if the error syndrome is  $|11\rangle$ , indicating an error on the middle qubit, first, the qubits 0 and 2 are flipped to an incorrect state. However, with the use of a Toffoli gate with multiple targets, all qubits are flipped back to their original state.

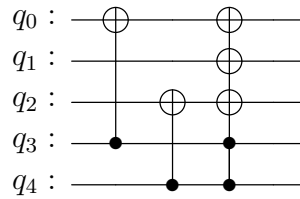


Figure 3.5: Error correction circuit for the three-qubit bit-flip code.

### 3.3.4 State decoding

The encoded state can be decoded back to the initial state by performing the encoding procedure, presented in Figure 3.2, backward to it, due to the unitarity of quantum operations. The decoding circuit for the three-qubit bit-flip code is illustrated in Figure 3.6.

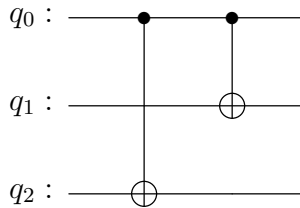


Figure 3.6: Decoding circuit for the three-qubit bit-flip code.

## 3.4 Three-qubit phase-flip code

The three-qubit phase-flip code is another example of a quantum error-correcting code able to correct one type of a Pauli error. It can correct a phase-flip error on one of the code qubits [10].

### 3.4.1 State encoding

The three-qubit phase-flip code encodes the initial state  $|\psi\rangle$ ,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \quad (3.6)$$

as

$$|\psi'\rangle = \alpha |+++ \rangle + \beta |-- \rangle, \quad (3.7)$$

where

$$\begin{aligned} |+\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \\ |-\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle). \end{aligned} \quad (3.8)$$

First, the base states are repeated across the code qubits with *cNOT* gates, and then, with Hadamard gates, the encoding of  $|0\rangle$  to  $|+\rangle$  and  $|1\rangle$  to  $|-\rangle$  is performed [10]. Figure 3.7 illustrates the circuit for this procedure.

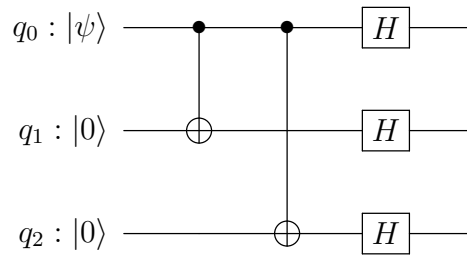


Figure 3.7: State encoding circuit for the three-qubit phase-flip code.

### 3.4.2 Syndrome measurement

The identity in Equation 3.9 [13] helps deduce the syndrome measure process for the phase-flip code. In fact, as compared to the syndrome measurement procedure for the bit-flip code, shown in Figure 3.4, *X* gates are used in the places of the *Z* gates to compare the phases of two pairs of qubits.

$$\begin{array}{c} \bullet \\ | \\ \text{---} \\ | \\ \boxed{X} \\ \text{---} \end{array} = \begin{array}{c} \bullet \\ | \\ \text{---} \\ | \\ \boxed{H} \text{---} \boxed{Z} \text{---} \boxed{H} \\ \text{---} \end{array} \quad (3.9)$$

The syndrome measurement circuit for the three-qubit phase-flip code is presented in Figure 3.8, and the error syndrome interpretation is shown in Table 3.2, where  $X_i X_j$  is the result of phase check on qubits  $i$  and  $j$ , and  $Z_i$  denotes the *Z* error on qubit  $i$ .

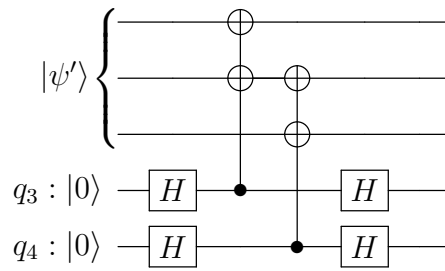


Figure 3.8: Syndrome measurement circuit for the three-qubit phase-flip code.

	$Z_0$	$Z_1$	$Z_2$	I
$X_0X_1$	1	1	0	0
$X_1X_2$	0	1	1	0

Table 3.2: Interpretation of syndrome measurement for the three-qubit phase-flip code.

### 3.4.3 Error correction

The methodology of the error correction procedure for the three-qubit phase-flip code is analogous to the one for the bit-flip code, shown in Figure 3.5. However, if a qubit was impacted by a  $Z$  error, the  $Z$  operator must be used to restore the qubit's state. Figure 3.9 show the circuit performing error correction for the phase-flip code.

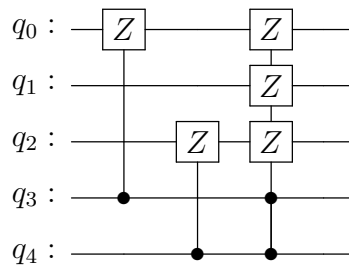


Figure 3.9: Error correction circuit for the three-qubit phase-flip code.

### 3.4.4 State decoding

The encoding procedure, shown in Figure 3.7, must be applied backward to the code qubits to obtain the quantum state that was initially passed to the error-correcting code. The decoding procedure for the three-qubit phase-flip code is presented in Figure 3.10.

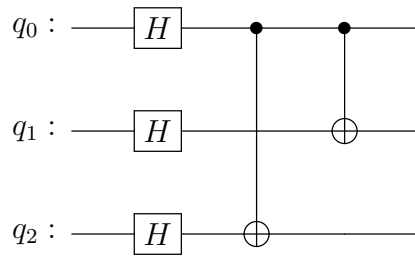


Figure 3.10: State decoding circuit for the three-qubit phase-flip code.

## 3.5 Nine-qubit Shor's code

Quantum error-correcting codes can be combined to create codes with greater capabilities. The nine-qubit Shor's code is an example of a code built from the three-qubit bit-flip code and the three-qubit phase flip code. It uses nine qubits to encode a single qubit quantum state and is able to correct an arbitrary error on one of the code qubits.

### 3.5.1 State encoding

The encoding procedure for the Shor's code includes two steps. First, the initial state  $|\psi\rangle$ ,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \quad (3.10)$$

is encoded with the three-qubit phase-flip code. In the second step, each of these qubits is encoded with the three-qubit bit flip code [10]. The encoded state is of the form

$$|\psi'\rangle = \frac{\alpha}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) + \frac{\beta}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle). \quad (3.11)$$

Circuit in Figure 3.11 presents the encoding procedure for the Shor's code, which was presented e. g. in [13].

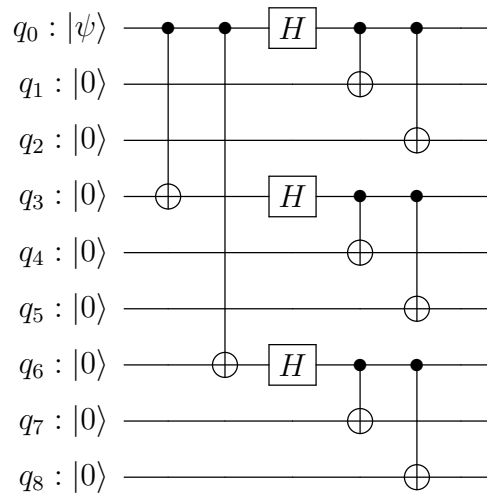


Figure 3.11: Encoding circuit for the Shor code.

### 3.5.2 Syndrome measurement

In the syndrome measurement procedure for the Shor’s code, the bit-flip and the phase-flip errors are checked independently.

It can be noticed that in the codeword for this code, qubits 0 to 2, 3 to 5, and 6 to 8 have common phases. In the first step, the phase check of qubits 0 to 2 with the phase of qubits 3 to 5 is performed, and the result of this check is stored in one ancilla qubit. The same is performed to compare the phases of qubits 3 to 5 with phases of qubits 6 to 8.

In the next step, the parity checking procedure, equivalent for the one used for the three-qubit bit-flip code, shown in Figure 3.4, is performed for qubits 0 to 2, 3 to 5, and 6 to 8.

Moreover, it can be noticed that as

$$Y = iXZ, \tag{3.12}$$

the  $Y$  single-qubit error will also be detected by the Shor’s code [12]. In total, the Shor’s code uses eight ancilla qubits, two of them storing the results of the phase check, and six storing parity check results.

The error syndrome circuit for the nine-qubit Shor’s code was proposed in [13] and is illustrated in Figure 3.12.

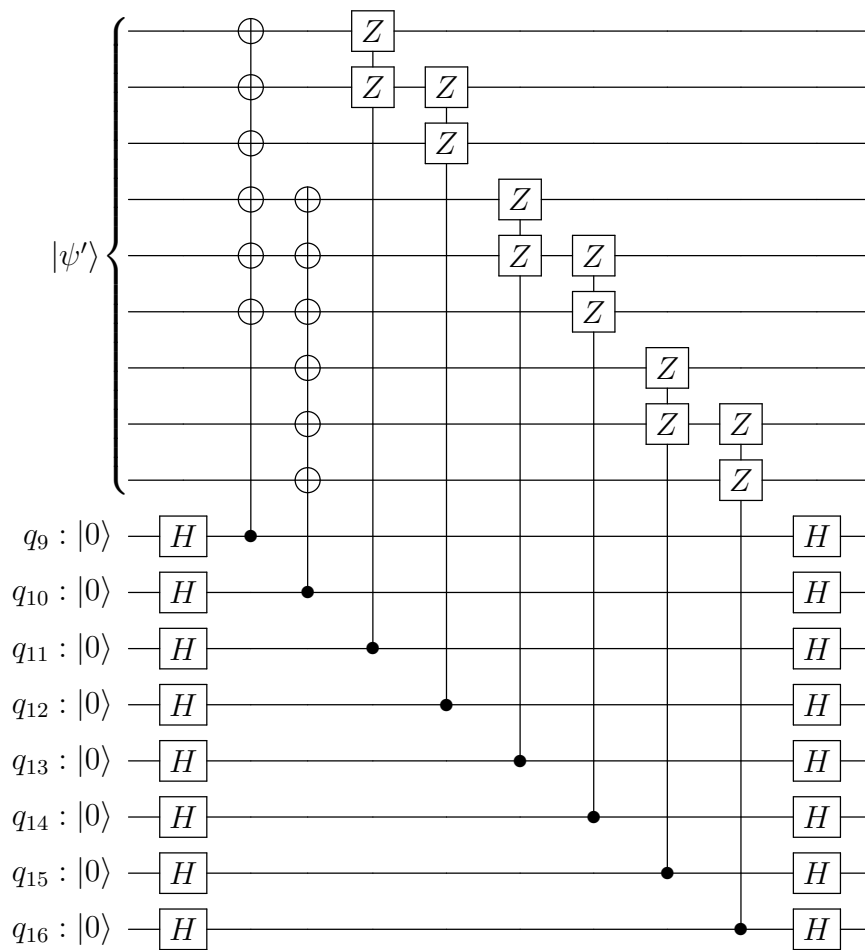


Figure 3.12: Syndrome measurement circuit for the Shor code.

Tables 3.3 and 3.4 show how to interpret error syndromes of the Shor's code. The result of comparing the phase of qubits  $i$ ,  $i + 1$  and  $i + 2$  with the phase of qubits  $i + 3$ ,  $i + 4$  and  $i + 5$  is denoted as  $X_i - X_{i+5}$ , and the result of parity check on qubits  $i$  and  $j$  is represented as  $Z_i Z_j$ . Phase error on qubit  $i$ ,  $j$  or  $k$  is denoted as  $Z_{ijk}$ , and  $X$  or  $Y$  error on qubit  $i$  is denoted as  $X_i$  or  $Y_i$  respectively.

---

	$Z_{012}$	$Z_{345}$	$Z_{678}$	$X_0$	$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$X_6$	$X_7$	$X_8$
$X_0 - X_5$	1	1	0	0	0	0	0	0	0	0	0	0
$X_3 - X_8$	0	1	1	0	0	0	0	0	0	0	0	0
$Z_0Z_1$	0	0	0	1	1	0	0	0	0	0	0	0
$Z_1Z_2$	0	0	0	0	1	1	0	0	0	0	0	0
$Z_3Z_4$	0	0	0	0	0	0	1	1	0	0	0	0
$Z_4Z_5$	0	0	0	0	0	0	0	1	1	0	0	0
$Z_6Z_7$	0	0	0	0	0	0	0	0	0	1	1	0
$Z_7Z_8$	0	0	0	0	0	0	0	0	0	0	1	1

Table 3.3: Interpretation of syndrome measurement for the Shor's code, part 1.

	$Y_0$	$Y_1$	$Y_2$	$Y_3$	$Y_4$	$Y_5$	$Y_6$	$Y_7$	$Y_8$	I
$X_0 - X_5$	1	1	1	1	1	1	0	0	0	0
$X_3 - X_8$	0	0	0	1	1	1	1	1	1	0
$Z_0Z_1$	1	1	0	0	0	0	0	0	0	0
$Z_1Z_2$	0	1	1	0	0	0	0	0	0	0
$Z_3Z_4$	0	0	0	1	1	0	0	0	0	0
$Z_4Z_5$	0	0	0	0	1	1	0	0	0	0
$Z_6Z_7$	0	0	0	0	0	0	1	1	0	0
$Z_7Z_8$	0	0	0	0	0	0	0	1	1	0

Table 3.4: Interpretation of syndrome measurement for the Shor's code, part 2.

### 3.5.3 Error correction

In the error correction protocol for the Shor's code, first, the phase-flip error is corrected, and subsequently the bit-flip error is corrected. Like in case of the three-qubit phase-flip code,  $Z$  gates are used to restore the initial phase of a quantum state. For correcting bit-flip errors,  $X$  gates are used, like in the case of the three-qubit bit-flip code. The whole error correcting procedure is shown in Figure 3.13.



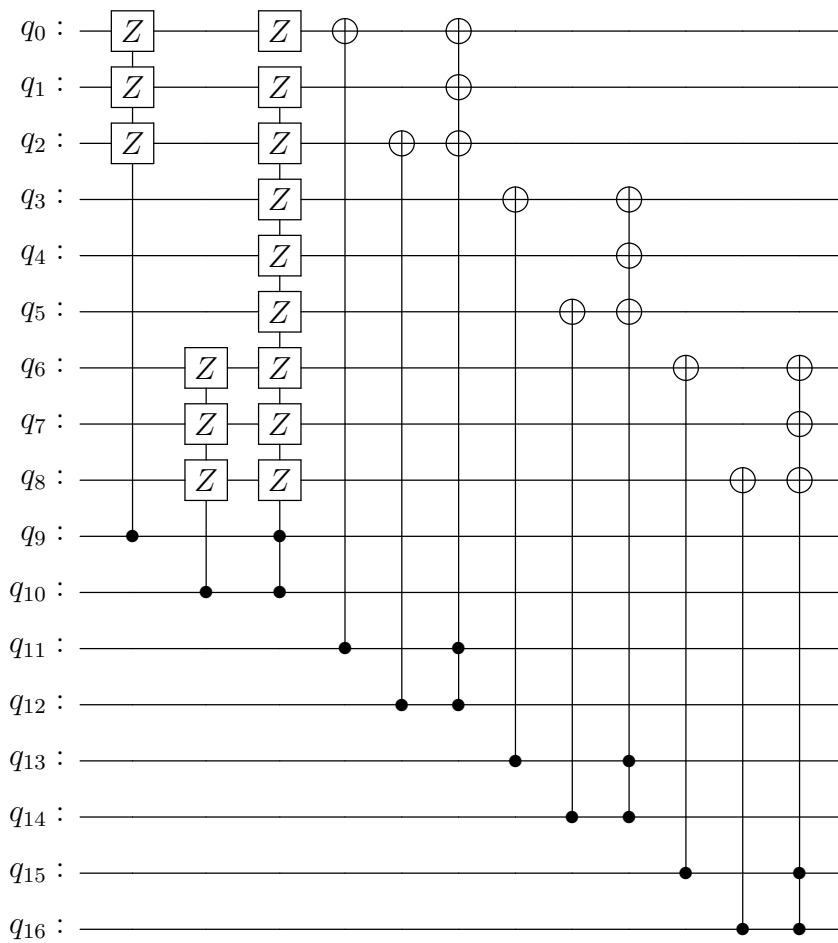


Figure 3.13: Error correction circuit for the Shor code.

### 3.5.4 State decoding

Finally, to decode the encoded and corrected state, the encoding procedure, illustrated in Figure 3.11, must be applied to the code qubits in reverse. The decoding circuit for the Shor's code is shown in Figure 3.14.

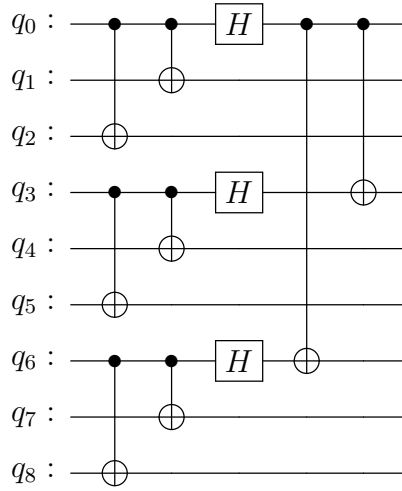


Figure 3.14: Decoding circuit for the Shor code.

### 3.6 The perfect five-qubit code

The five-qubit  $[[5, 1, 3]]$  code is an example of a *perfect* code, in the sense of the quantum Hamming bound. It is able to protect a single-qubit state and correct an arbitrary single-qubit error on one of the five encoding qubits. In this section, the structure and characteristics of this quantum error-correcting code will be discussed.

Let's define four five-qubit quantum operators

$$\begin{aligned}
 M_0 &= Z_1 X_2 X_3 Z_4, \\
 M_1 &= Z_0 Z_2 X_3 X_4, \\
 M_2 &= X_0 Z_1 Z_3 X_4, \\
 M_3 &= X_0 X_1 Z_2 Z_4,
 \end{aligned} \tag{3.13}$$

where

$$\begin{aligned}
 Z_0 &= Z \otimes I \otimes I \otimes I \otimes I, \\
 Z_1 &= I \otimes Z \otimes I \otimes I \otimes I, \\
 Z_2 &= I \otimes I \otimes Z \otimes I \otimes I, \\
 Z_3 &= I \otimes I \otimes I \otimes Z \otimes I, \\
 Z_4 &= I \otimes I \otimes I \otimes I \otimes Z, \\
 X_0 &= X \otimes I \otimes I \otimes I \otimes I, \\
 X_1 &= I \otimes X \otimes I \otimes I \otimes I, \\
 X_2 &= I \otimes I \otimes X \otimes I \otimes I, \\
 X_3 &= I \otimes I \otimes I \otimes X \otimes I, \\
 X_4 &= I \otimes I \otimes I \otimes I \otimes X.
 \end{aligned} \tag{3.14}$$

### 3.6.1 State encoding

The five-qubit perfect code encodes the initial quantum states  $|\psi\rangle$ ,

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle, \quad (3.15)$$

as

$$|\bar{\psi}\rangle = \alpha |\bar{0}\rangle + \beta |\bar{1}\rangle, \quad (3.16)$$

where

$$\begin{aligned} |\bar{0}\rangle = \frac{1}{4}(I + M_0)(I + M_1)(I + M_2)(I + M_3) |00000\rangle = \\ |00000\rangle + |10010\rangle + |01001\rangle + |10100\rangle \\ + |01010\rangle - |11011\rangle - |00110\rangle - |11000\rangle \\ - |11101\rangle - |00011\rangle - |11110\rangle - |01111\rangle \\ - |10001\rangle - |01100\rangle - |10111\rangle + |00101\rangle, \end{aligned} \quad (3.17)$$

$$\begin{aligned} |\bar{1}\rangle = \frac{1}{4}(I + M_0)(I + M_1)(I + M_2)(I + M_3) |11111\rangle = \\ |11111\rangle + |01101\rangle + |10110\rangle + |01011\rangle \\ + |10101\rangle - |00100\rangle - |11001\rangle - |00111\rangle \\ - |00010\rangle - |11100\rangle - |00001\rangle - |10000\rangle \\ - |01110\rangle - |10011\rangle - |01000\rangle + |11010\rangle. \end{aligned} \quad (3.18)$$

The quantum circuit which performs such encoding, for a quantum state originally stored in qubit  $q_0$ , was proposed in [13], and is shown in Figure 3.15.

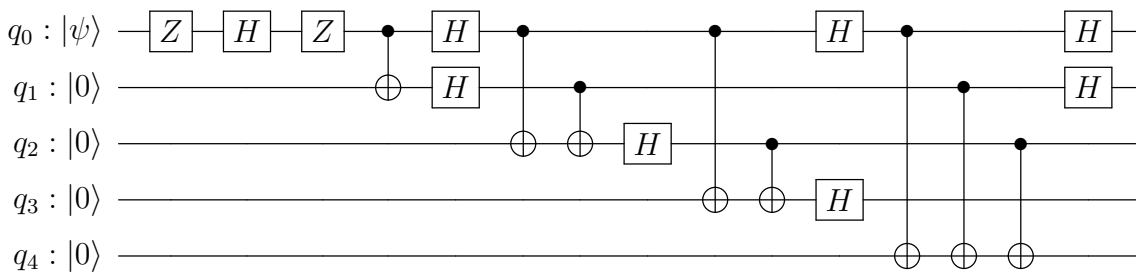


Figure 3.15: Encoding circuit for the five-qubit code.

### 3.6.2 Syndrome measurement

The syndrome measurement for the five-qubit perfect code is obtained by applying the  $M_0$ ,  $M_1$ ,  $M_2$ ,  $M_3$  operators, and storing the result of each of them in a separate ancilla qubit. At the end of the syndrome measurement procedure, each ancilla qubit will be in either  $|0\rangle$  or  $|1\rangle$  state. As a result, there are 16 different possible error syndromes that can be detected, and all syndromes indicate the occurrence

of a different quantum error or no error. The interpretation of results of syndrome measurement is shown in Table 3.5, where  $X_i$ ,  $Y_i$  and  $Z_i$  denotes respectively  $X$ ,  $Y$  and  $Z$  error on  $i_{th}$  qubit. For example, if the error syndrome is equal to  $|0100\rangle$ , it indicates that  $X$  error occurred on qubit 0.

	$X_0$	$Y_0$	$Z_0$	$X_1$	$Y_1$	$Z_1$	$X_2$	$Y_2$	$Z_2$	$X_3$	$Y_3$	$Z_3$	$X_4$	$Y_4$	$Z_4$	I
$M_0$	0	0	0	1	1	0	0	1	1	0	1	1	1	1	0	0
$M_1$	1	1	0	0	0	0	1	1	0	0	1	1	0	1	1	0
$M_2$	0	1	1	1	1	0	0	0	0	1	1	0	0	1	1	0
$M_3$	0	1	1	0	1	1	1	1	0	0	0	0	1	1	0	0

Table 3.5: Interpretation of syndrome measurement for the five-qubit code.

The quantum circuit which performs syndrome measurement for the perfect five-qubit code, and stores the results of  $M_0$ ,  $M_1$ ,  $M_2$  and  $M_3$  operators in ancilla qubits  $q_5$ ,  $q_6$ ,  $q_7$  and  $q_8$  respectively, was presented e. g. in [13] and is shown in Figure 3.16.

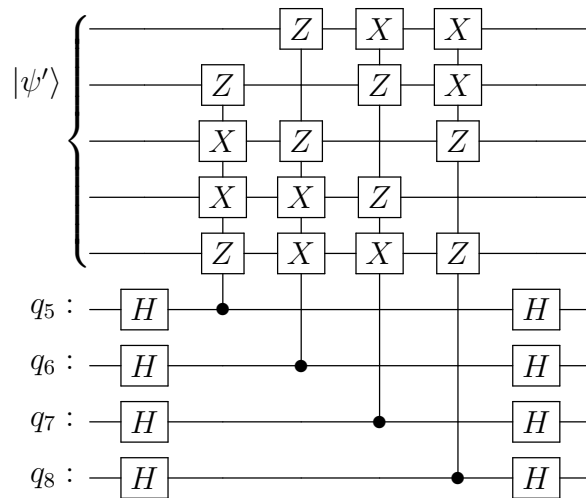


Figure 3.16: Syndrome measurement circuit for the five-qubit code.

### 3.6.3 Error correction

Based on the error syndrome, appropriate action must be taken to recover the correct, encoded state. Due to the reversibility of quantum operations, the Pauli error that corrupted the state can be corrected by applying the same Pauli operator to the impacted qubit once again. An example of correction of the  $X_0$  error is shown in figure 3.17.

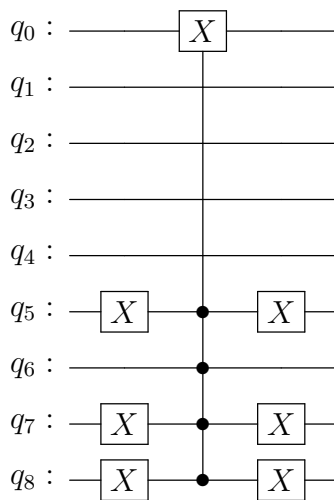


Figure 3.17: Circuit correcting the  $X_0$  error.

First, the ancilla qubits expected to be in the  $|0\rangle$  state for a particular error syndrome are negated. Then, if the actual error syndrome was equal to the expected one, the ancilla qubits will be in the state  $|1111\rangle$  at this point, and a four-controlled operator will apply a correcting gate to the impacted qubit. Then, the original error syndrome has to be restored by flipping the negated ancilla qubits to their original states.

### 3.6.4 State decoding

To restore the original single-qubit state, decoding is performed. As quantum operations are reversible, the encoding procedure, shown in Figure 3.15, can be applied to the circuit backward to decode the state. In Figure 3.18, a quantum circuit which performs decoding for the five-qubit code is shown.

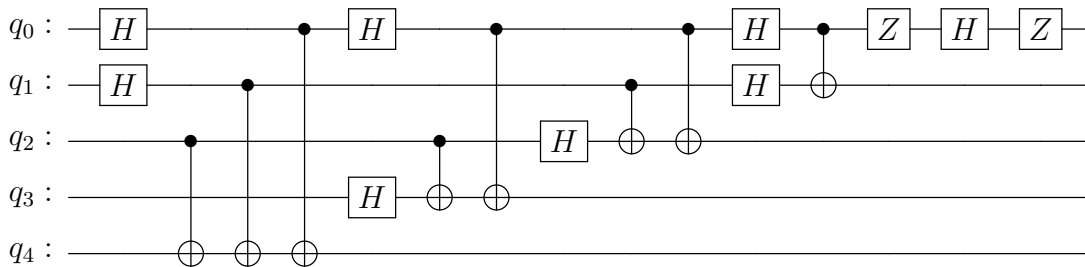


Figure 3.18: Decoding circuit for the five-qubit code.

---

## 3.7 Evaluation of quantum error-correcting codes

This section presents a selection of techniques that can be used to assess quantum error-correcting codes.

### 3.7.1 Quantum state tomography

To assess the quantum state obtained from executing a quantum program in the presence of noise, the *quantum state tomography* can be performed, which will allow further characterization of the state. Quantum state tomography is a method to experimentally reconstruct the density matrix for a quantum state, possibly a mixed quantum state, by measuring it in different bases.

#### State fidelity

*State fidelity* is a measure of how two quantum states differ from each other. Often it is defined with respect to the expected quantum state and calculated for the experimentally obtained quantum state. Fidelity of two quantum states, represented by density matrices  $\rho$  and  $\sigma$ , is given by

$$\left[ \text{Tr} \left( \sqrt{\sqrt{\rho}\sigma\sqrt{\rho}} \right) \right]^2, \quad (3.19)$$

where  $\text{Tr}$  denotes the matrix trace.

#### State purity

*State purity* is a measure of how pure a quantum state is. Given a quantum state represented by a density matrix,  $\rho$ , the purity of this state is defined as

$$\text{Tr} (\rho^2). \quad (3.20)$$

### 3.7.2 Clifford group

*Clifford group* is a set of quantum operators of the form

$$C_n = \{V : VP_nV^\dagger = P_n\}, \quad (3.21)$$

where  $P_n$  denotes elements from the *Pauli group*, defined as

$$P_n = \{e^{i\theta} P_1 \otimes P_2 \otimes \cdots \otimes P_n, \theta \in [0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}], P_1 \in [X, Y, X, I]\}. \quad (3.22)$$

As stated in the *Gottesman-Knill theorem* [30], quantum circuits which contain only the gates from the Clifford group, called the *Clifford gates*, can be efficiently simulated on classical computers.

Sequences of Clifford gates can be referred to as *Clifford sequences*.

---

### 3.7.3 Randomized benchmarking

Randomized benchmarking is a technique used to characterize quantum devices in terms of an average error rate per a quantum gate, which was first proposed in [29]. It uses sequences of different length of randomly chosen gates from the Clifford group to study how the average gate error changes with the length of the used sequence.

For an increasing number of Clifford gates,  $k$ , Clifford sequences of the form

$$C_1 C_2 \dots C_k (C_1 C_2 \dots C_k)^{-1}, \quad (3.23)$$

where  $C_i$  denotes  $i_{th}$  Clifford gate in the sequence, are applied to a quantum circuit.

Each sequence of the form in Equation 3.23 would produce identity if there was no noise in the system, so the sequence should not change the input quantum state. However, in the presence of noise, the fidelity of the output state decays with Clifford sequences' length applied to the circuit.

## 3.8 Summary

In this chapter, the basis and a general scheme of quantum error-correcting codes, as well as some examples of such codes, were presented. In Chapter 5, realization of the five-qubit perfect code, described in section 3.6, and the three-qubit bit-flip code, presented in section 3.3, on IBM Q will be discussed.

In the next chapter, an overview of existing applications of quantum error-correcting methods on IBM Q will be given.

---

## Chapter 4

# Quantum Error Correction Methods Applications on IBM Q

Through the IBM Q Experience platform, real quantum devices, as well as a reliable quantum simulator, can be accessed online for free. Moreover, convenient software tools for programming these devices are available. As a result, research oriented on quantum computing is often conducted using the IBM Q platform nowadays, and quantum error correction is no exception. In this section, an overview of existing applications of quantum error correction methods on IBM Q will be given. Also, a selection of features of the Qiskit framework will be discussed.

### 4.1 Quantum repetition codes

Repetition codes are one of the groups of quantum error-correction methods implemented with the use of the IBM Q Experience platform. In [17], repetition codes of the distance  $d$  between 3 and 8 were studied. In this approach, the original qubit's state is repeated across  $d$  qubits and ancilla qubits are used for storing information about the parity of pairs of qubits. Look-up tables filled with experimental data, containing the probabilities of different inputs, based on the received output, are used for restoring the initial state. The analysis showed that the probability of an error occurring decays exponentially with the code distance. Thus, quantum information stored across many qubits is more reliable than information stored in a single qubit. Moreover, it was verified that with the use of information about code qubits' parity, stored in ancilla qubits, the encoded states can be protected with even greater fidelity.

### 4.2 Quantum error-detecting codes with post selection

In [18], it was stated that the  $[[4, 2, 2]]$  quantum error-detecting code, which can be implemented with merely five qubits, can be used to improve a quantum circuit's fault-tolerance. This code is capable of detecting an error, but it is not capable



---

of correcting it. However, the faulty states can be discarded after the procedure completes, which is called *post-selection*. The implementation of this code with the use of IBM Q Experience platform was presented in a couple of publications.

In [19], an implementation of a  $[[4, 2, 2]]$  *coherent parity check* code, targeted at the IBMQX4 5-qubit IBM quantum device, was presented. The procedure was run in 154 batches, each including 8192 experimental runs of the circuit. With the use of Qiskit, output state-tomography was performed to reconstruct the approximate density matrix for the output states. The purity and fidelity of the experimentally obtained density matrices before and after post-selection were compared. The results showed that the  $[[4, 2, 2]]$  code can successfully detect quantum errors. With the use of post-selection, it was proven to improve the output state's purity from  $0.52 \pm 0.02$  to  $0.74 \pm 0.03$ , and its fidelity from  $0.62 \pm 0.03$  to  $0.75 \pm 0.04$ . Unfortunately, only  $(54 \pm 2)\%$  of the results survived the post-selection procedure.

In [20], a  $[[4, 2, 2]]$  quantum error detecting code was implemented with the QASM assembly language, and it was tested on the IBMQX5 16-qubit IBM quantum device. The infidelities of the output states obtained from a raw two-qubit quantum state, and a state encoded with the  $[[4, 2, 2]]$  code with post-selection, were experimentally obtained and compared. Experiments were carried out with the use of the *randomized benchmarking* protocol, including 30 random Clifford sequences of lengths from 2 to 92 applied to the quantum circuit, each executed 1024 times. The results showed that the average output state infidelity dropped from 5.8(2)% for a raw physical state to 0.60(3)% for the encoded state. A very similar experiment was performed as a part of [21]. Its results confirmed a factor of ten decrease in the infidelity of the output states with the use of the  $[[4, 2, 2]]$  quantum error-detecting code, from 2.8% for the raw physical state, to 0.19% for the encoded state. In [22], an implementation of the  $[[4, 2, 2]]$  quantum error-detecting code is presented as well. Experiments comparing the performances of a bare circuit, and a circuit with the  $[[4, 2, 2]]$  code with post-selection, for twenty different input states, were run on several IBM 5Q chips. It was proven that fault-tolerantly designed, longer circuits can outperform shorter, non-fault-tolerant circuits.

### 4.3 Nondestructive discrimination and automated error-correction of fully-entangled states

There are papers that show a different approach to quantum error-correction. In classical quantum error-correcting codes, the original state is encoded in an enlarged Hilbert space, and a *majority voting* method is used to restore the initial value. In [23], a completely new method is presented. An error-correction procedure for fully entangled quantum states, the Generalized Bell states in particular, able to correct a single bit-flip error or an arbitrary phase-change error, is designed. This method hinges on specific properties, e. g. symmetry, of the Generalized Bell states. The original state is not repeated across multiple qubits, but information about the phase and the parity of the state is stored in ancilla qubits, enabling the initial state to be recovered without interrupting the original state. A proof of correctness of the method was presented, and it was stated that it can be implemented on real quantum devices.

---

An implementation of this procedure on a five-qubit quantum computer, available through the IBM Q Experience platform, was presented in [24]. Due to limitations of the used device, like lack of connectivity between all pairs of qubits, the original circuit was modified to match its target device. First, the phase-checking circuit and the parity-checking circuit were realized separately. For each circuit, all four Bell states were prepared, and experimental density matrices were obtained through quantum state tomography procedure. The calculated output state's fidelities were on average for all Bell states 0.8027 for the phase-checking circuit and 0.80755 for the parity-checking circuit. A circuit realizing both phase and parity correction was also implemented. However, the results obtained from it were less successful as a consequence of the increased number of used gates. An implementation of nondestructive discrimination and automated error correction for the generalized Bell states is also shown in [25]. Experiments are run on a five-qubit quantum computer available through the IBM Q experience platform, and experimental density matrices for the output states are obtained through the state tomography procedure. The average absolute deviation of the measured output states from the ideal states is 0.28%. Moreover, an analogous procedure for fully entangled three-qubit quantum states, the GHZ states, is presented in this publication. Due to the limitations of the used quantum device, discrimination, and correction of errors for these states were performed separately. The results for GHZ states had an average absolute deviation of 0.17% from the ideal states.

## 4.4 Qiskit framework

Qiskit is an open-source, Python framework, with which quantum programs can be implemented and run on quantum devices, including those available through the IBM Q Experience platform. Qiskit consists of four elements: Qiskit Terra, Qiskit Aer, Qiskit Ignis, and Qiskit Aqua, each oriented on a different aspect of quantum programming. Qiskit offers a wide range of tools that can be very helpful in dealing with quantum errors and quantum noise.

Qiskit Terra is a foundation for all other Qiskit modules. It provides its users with access to quantum backends and offers basic functionalities, such as constructing and running quantum circuits. Quantum operations can be applied as quantum gates or pulses. The latter work closer to the hardware and can be used to reduce errors in quantum circuits. Moreover, Qiskit Terra enables the transpilation of quantum circuits to analogous circuits, optimized for a particular target backend. Transpilation can reduce the number of quantum gates in the circuit, and thus shorten the program's runtime on this device. Also, Qiskit Terra has visualization tools, which are needed for the users to interpret results obtained from running quantum programs.

Qiskit Aer is a module with which reliable quantum simulations can be carried out. Simulations run on a bare quantum simulator imitate an ideal quantum device's behavior, without quantum noise. Qiskit Aer, however, offers mechanisms for introducing realistic, and at the same time highly configurable, artificial quantum noise into simulations. Other limitations of real quantum devices, like non-fully-connected coupling map, or a restricted set of available quantum gates, can be simulated with Qiskit Aer.

---

Qiskit Ignis is a module particularly useful for implementing and testing quantum error-correction methods. It's oriented on analyzing and mitigating noise and errors in quantum devices. Functionalities of Qiskit Ignis are divided into three sections, which are *characterization*, *verification* and *mitigation*. The characterization module is aimed at measuring noise parameters of quantum systems, such as gate control errors. The verification module provides the user with tools for verifying gates and circuits' performance. Among all, it enables the user to perform quantum state tomography or test quantum circuits with the randomized benchmarking protocol. Also, it contains an implementation of quantum repetition codes. The mitigation module is oriented on reducing the effects of errors in quantum circuits. Calibration measurements, obtained from running calibration circuits, can be used to fit actual, erroneous output results to less noisy results, based on statistical data.

## 4.5 Summary

In this chapter, existing applications of quantum error-correcting codes on IBM Q, and also some features of the Qiskit framework, were discussed. It can be noticed that available quantum devices impact the choice of quantum error-correcting methods for implementation on IBM Q, and methods which can be realized on the five-qubit quantum devices are especially popular, like the  $[[4, 2, 2]]$  code. The next chapter focuses on a new implementation of the perfect-five qubit code realized on IBM Q as a part of this thesis.

---

# Chapter 5

## Solution

The five-qubit perfect code is the smallest, in the sense of the quantum Hamming Bound, quantum-error correcting code for a one qubit state, able to correct one arbitrary single-qubit quantum error. Using the error correction implementation proposed in section 5.2.3, it demands twelve qubits in total, so it can be run on both the ibmq 16 melbourne and the ibmq qasm simulator. To the best of the author's knowledge, it has not been realized on IBM Q before, and therefore it was chosen to be implemented as a part of this thesis.

One of the goals of this thesis was to compare the effectiveness of more than one method of quantum error-correction. Due to its simplicity, the three-qubit bit-flip code was chosen to be implemented as a part of this thesis as well, so that the efficacy of the two codes can be assessed.

In this chapter, the realization of both quantum error-correcting codes on IBM Q is presented. The changes to known theoretical circuits for the codes, needed to make them compliant with the used software tools, are explicated. Moreover, a new error-correction procedure for the five-qubit perfect code is shown.

### 5.1 Technologies

Quantum programs realized as a part of this work were implemented with the *Python 3.7* programming language [9]. To access quantum devices available through the IBM Quantum Experience platform, the *Qiskit 0.19.6* library [6] was used.

### 5.2 New implementation of the perfect five-qubit code

According to the quantum Hamming bound, the five-qubit perfect code, described in section 3.6, is the smallest quantum code of distance three, able to correct a single, arbitrary quantum error. This code uses five qubits to encode a single-qubit state and seven ancilla qubits, so it requires twelve qubits in total. As a quantum simulator able to simulate up to thirty-two qubits, and a real quantum device with

fifteen qubits are available through the IBM Q Experience platform, this code can be implemented and executed on this platform, however, to the best of the author's knowledge, such implementation has not been realized before.

As mentioned in section 3.6, the five-qubit perfect code comprises of four steps: encoding, syndrome measurement, error correction and decoding. Realization of these protocols are described in sections 5.2.1, 5.2.2, 5.2.3 and 5.2.4, respectively.

The first part of the work done to realize the five-qubit code on IBM Q was to transform known circuits, proposed e. g. in chapter 5 of [13], to make them applicable on IBM Q. Such changes were introduced to the syndrome measurement procedure, described in section 5.2.2.

The second part was to propose new quantum circuits that were not shown in any of the available resources known to the author, based on theoretical information found in available resources, in particular in [13]. Based on the syndrome interpretation table for this code, which is shown in [13], a new, optimized error-correction circuit for the five qubit code was designed and realized, and it's presented in section 5.2.3. Also, the decoding circuit for the five-qubit perfect code, shown in Figure 3.18, was proposed as a part of this thesis based on the encoding procedure for this code, which can be found in [13], and the unitarity of quantum operators.

The last part was to implement the code using the Qiskit framework.

### 5.2.1 State encoding

The theoretical quantum circuit for the state encoding procedure is shown in Figure 3.15. The procedure was presented in chapter 5 of [13]. As a part of this work, the procedure was implemented with Qiskit, and no changes were introduced to the circuit. Figure 5.1 shows a quantum circuit generated from the Python code with Qiskit visualization tools.

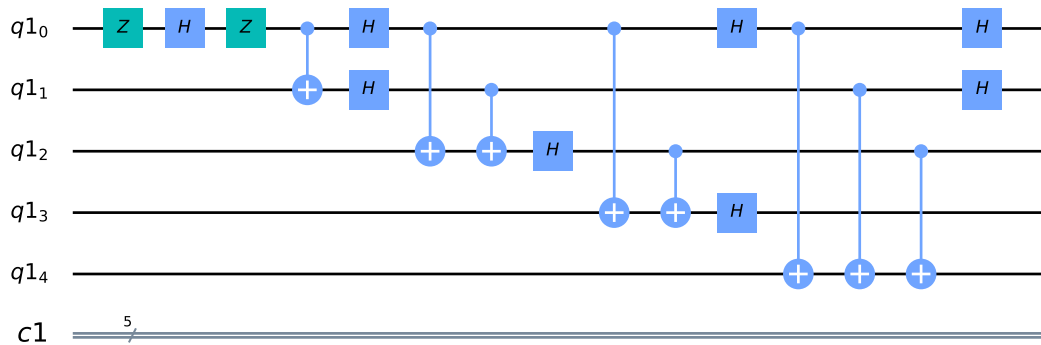


Figure 5.1: Implementation of encoding for the five-qubit code.

---

## 5.2.2 Syndrome measurement

The theoretical syndrome measurement circuit for the five-qubit code is presented in [13], and it's shown in Figure 3.16. In section 3.6, operators  $M_0$ ,  $M_1$ ,  $M_2$  and  $M_3$  were introduced, and their controlled versions are used in the syndrome measurement procedure. In order for the circuit to be implemented with Qiskit, multi-target controlled- $M_i$  gates, where  $M_i$  represents operators  $M_0$ ,  $M_1$ ,  $M_2$  and  $M_3$ , had to be decomposed into controlled gates with single targets.

In general, a quantum controlled gate with a single control qubit and  $n$ -qubit target operator  $A$ , where  $A$  is a tensor product of  $n$  single-qubit operators  $A_1, A_2, \dots, A_n$ , can be rewritten as a product of  $n$  separate, single-target controlled operations [13]. A gate of the form shown in Figure 5.2, can be equivalently realized as presented in Figure 5.3.

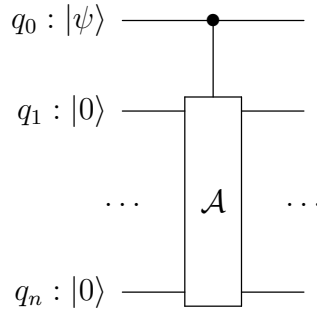


Figure 5.2: Multi-target controlled gate.

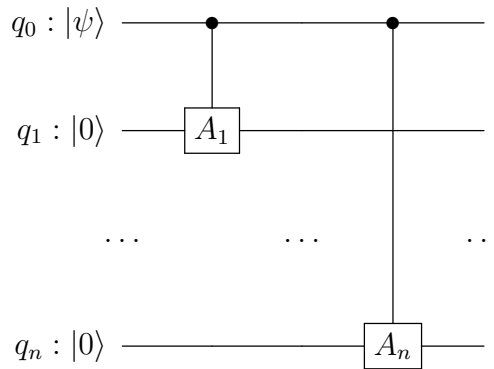


Figure 5.3: Decomposed multi-target controlled gate.

The controlled- $M_i$  operators, which can be denoted as  $cM_i$ , were transformed as shown in Equation 5.1.

$$\begin{aligned}
cM_0 &\equiv cZ_1cX_2cX_3cZ_4, \\
cM_1 &\equiv cZ_0cZ_2cX_3cX_4, \\
cM_2 &\equiv cX_0cZ_1cZ_3cX_4, \\
cM_3 &\equiv cX_0cX_1cZ_2cZ_4
\end{aligned}
\tag{5.1}$$

The transformed, compiled circuit is presented in Figure 5.4.

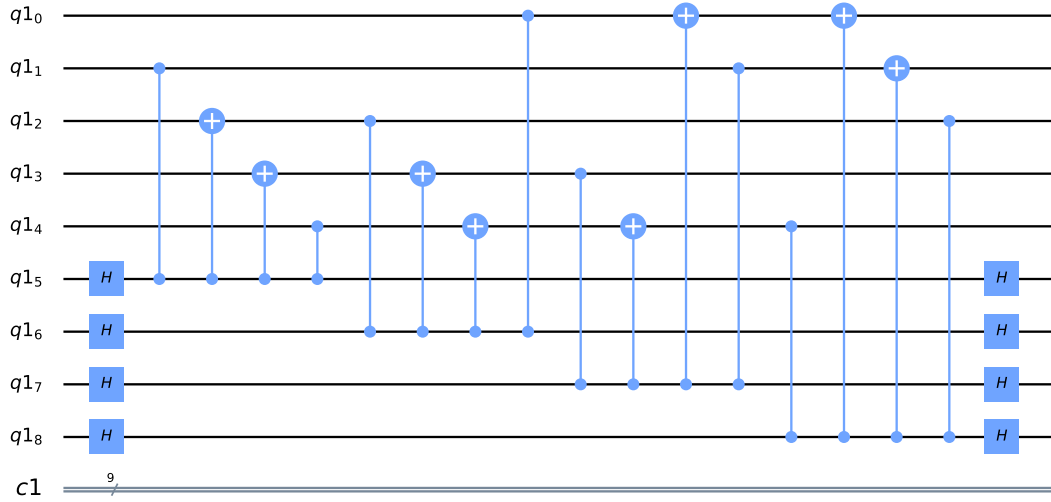


Figure 5.4: Implementation of syndrome measurement for the five-qubit code.

### 5.2.3 Error correction

The error correction procedure was designed based on the error syndromes interpretation illustrated in Table 3.5.

After the syndrome measurement procedure, the state of each ancilla qubit is either  $|0\rangle$  or  $|1\rangle$ . To apply the appropriate Pauli operator on erroneous qubit, *four-controlled operators* were used, where all ancilla qubits are the control qubits, the controlled operation is the Pauli operator which caused an error, and affected qubit is the target qubit. Before the four-controlled operator can be applied, the ancilla qubits which are in the  $|0\rangle$  state must be first negated, because a controlled gate only impacts its target if all control qubits are in the  $|1\rangle$  state. For example, the correction  $X_0$  should be applied if the state of ancilla qubits is  $|0100\rangle$ ,

$$|0100\rangle = |0\rangle \otimes |1\rangle \otimes |0\rangle \otimes |0\rangle, \tag{5.2}$$

and to detect this state with a controlled operation, it must be changed to

$$|0100\rangle \longrightarrow X|0\rangle \otimes |1\rangle \otimes X|0\rangle \otimes X|0\rangle = |1111\rangle. \quad (5.3)$$

Afterward, the original state of ancilla qubits must be restored to perform analogous controlled operations for all possible error syndromes, based on the original state of ancillas. The procedure which corrects this error, proposed in this thesis, is shown in Figure 3.17.

For the code to be able to correct an arbitrary error on one of the code qubits, an analogous procedure must be performed for all possible error syndromes in Table 3.5. For each error syndrome, the ancillas that are in the state  $|0\rangle$  in this error syndrome are negated, and only if the actual error syndrome matched the checked one all control qubits will be in the state  $|1\rangle$ , and the correcting operator will be applied to the erroneous qubit. The full error correction algorithm for the five-qubit perfect code for the is shown in Algorithm 1.

---

**Algorithm 1:** Error correction algorithms for the five-qubit perfect code

---

```

for all possible error syndromes do
    negate ancillas which are in  $|0\rangle$  state for this error syndrome;
    if all ancilla qubits are in state  $|1\rangle$  at this point then
        apply appropriate operator to the erroneous qubit based on Table
        3.5;
    end
    flip negated ancilla qubits back to their original state;
end

```

---

### Four-controlled gate implementation

A four-controlled quantum operation, illustrated in Figure 5.5, is not directly available through the Qiskit framework. However, it can be expressed with *Toffoli gates*, with the use of three ancilla qubits [34]. A decomposed four-controlled gate is presented in Figure 5.6.

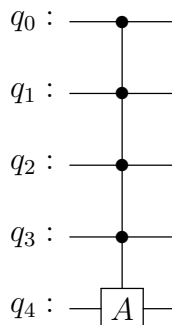


Figure 5.5: General four-controlled gate.



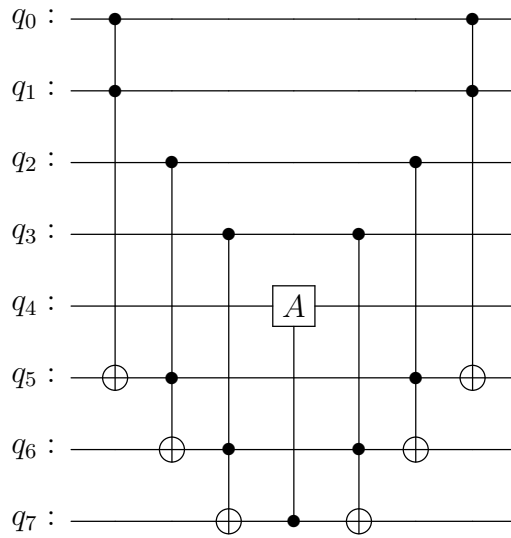


Figure 5.6: General four-controlled gate build with Toffoli gates.

Toffoli gate isn't directly applicable on IBM Q quantum devices, as there's no physical counterpart of this operation. Instead, it's further decomposed single-qubit and two-qubit operators. The default decomposition of the Toffoli gate in Qiskit contains six *cNOT* gates, and such decomposition is shown in Figure 5.7, with  $q_0$  and  $q_1$  as control qubits, and  $q_2$  as the target qubit.

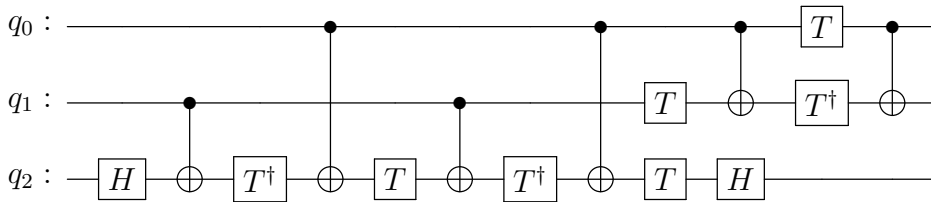


Figure 5.7: Decomposition of Toffoli gate with six *cNOT* gates.

In real quantum computers, the more quantum operations are used in a circuit, the more costly it is to execute this circuit. To lower the number of quantum operations used in the error correction procedure for the five-qubit code, an alternative implementation of the Toffoli gate, which was presented in [36], was used. The alternative decomposition uses four *cNOT* gates and is show in figure 5.8.

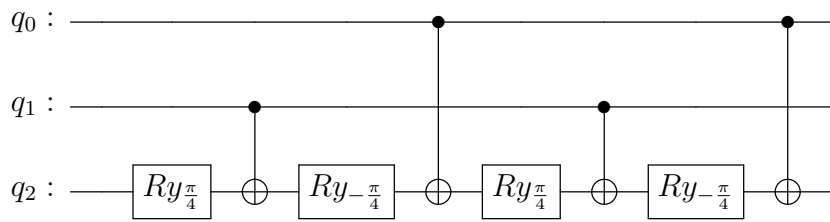


Figure 5.8: Decomposition of Toffoli gate with four cNOT gates.

### Further optimization

It can be noticed that based on the order in which individual error-correcting protocols for all Pauli errors and all qubits, it is possible to get rid of some operators by reusing the state of intermediate parity checks. In general, if two identical gates are adjacent, in terms that there are no other gates between them, both gates can be removed from the circuit without changing its functionality. For example the sequence shown in Figure 5.9 is equivalent to the sequence presented in Figure 5.10.

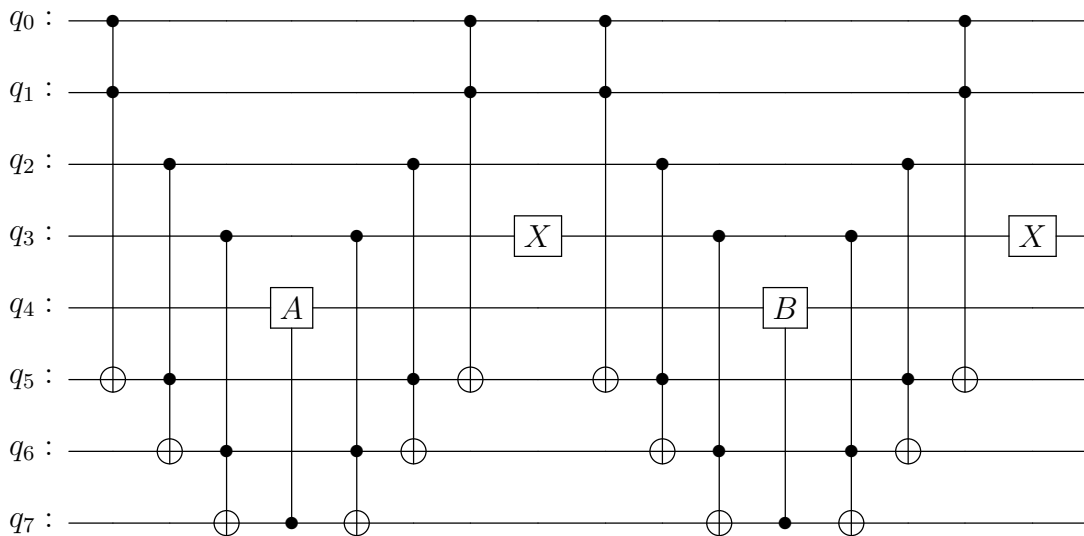


Figure 5.9: Example of adjacent four-controlled gates application.

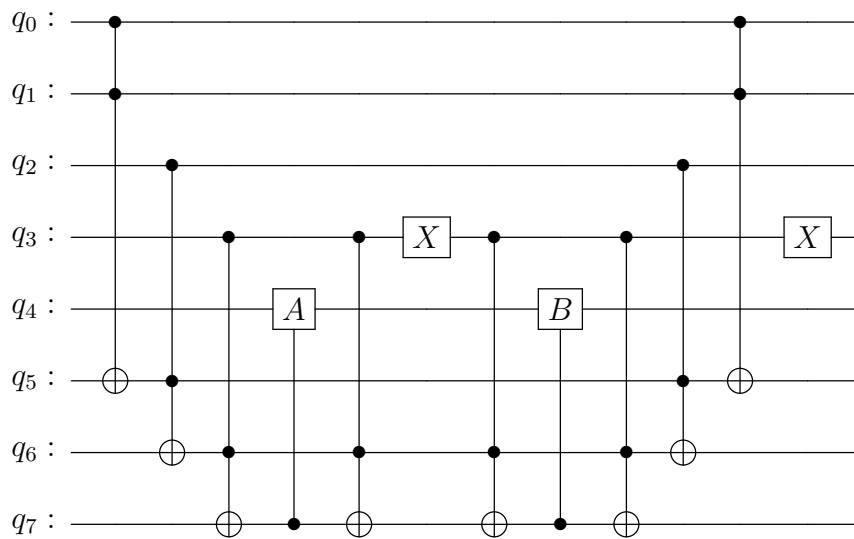


Figure 5.10: Example of adjacent four-controlled gates application after optimization.

In the case of the error-correcting procedure realized in this work, it can be noticed that there are two situations when duplicated gates from the adjacent, individual error-correcting sequences can be removed.

1. If two adjacent error syndromes have equal state of the first two ancilla qubits, the last Toffoli gate from the sequence for the first error syndrome and the first Toffoli gate from the sequence for the second error syndrome can be removed. In case if the first or second ancilla qubit's state is zero for both error syndromes, the duplicated  $X$  gates would also be removed from the circuit.
2. If two adjacent error syndromes have equal state of the first three ancilla qubits, the two last Toffoli gates from the first sequence and the two first Toffoli gates from the second sequence can be removed. Like in the previous case, if the first, second, or third ancilla qubit's state is zero for both error syndromes, the duplicated  $X$  gates would also be removed from the circuit.

As a result, by scheduling the correction of errors appropriately, the number of gates that can be removed from the circuit can be increased. The optimal order contains as many adjacent correcting sequences for error syndromes with equal states of the first three ancilla qubits, and then as many adjacent correcting sequences for error syndromes with equal states of the first two ancilla qubits, as possible.

Assuming that error syndromes represent binary numbers, with the first ancilla as the most significant bit, the optimal scheduling is to perform error correcting sequences in the order of error syndromes representing subsequent numbers. Such order is shown in Equation 5.4, where  $X_i$ ,  $Y_i$  and  $Z_i$  denotes the procedure which corrects the  $X$ ,  $Y$  or  $Z$  error, respectively, on the  $i_{th}$  qubit.

---


$$Z_1 X_3 Z_0 X_0 X_2 Z_4 Y_0 Z_2 X_4 X_1 Y_1 Z_3 Y_2 Y_3 Y_4 \quad (5.4)$$

It can be seen in Table 3.5 that the error syndrome for error  $Z_1$  is  $|0001\rangle$ , and  $0001 \equiv \text{bin}(1)$ . Then, the error syndrome for error  $X_3$  is  $|0010\rangle$ , and  $0010 \equiv \text{bin}(2)$ , and so on. Finally, the error syndrome for the error  $Y_4$  is  $|1111\rangle$ , and  $1111 \equiv \text{bin}(16)$ . Using this order of error correcting sequences in the realization of error correcting procedure for the five-qubit perfect code, and removing duplicate gates, the number of used Toffoli gates in total was decreased from 90 to 54.

In Figure 5.11, a part of the final error correcting circuit, with the first two error-correcting sequences, is shown. The correction of  $Z_1$  and  $X_3$  error is marked with an orange and a green box, respectively. The full circuit can be found at the link [35].

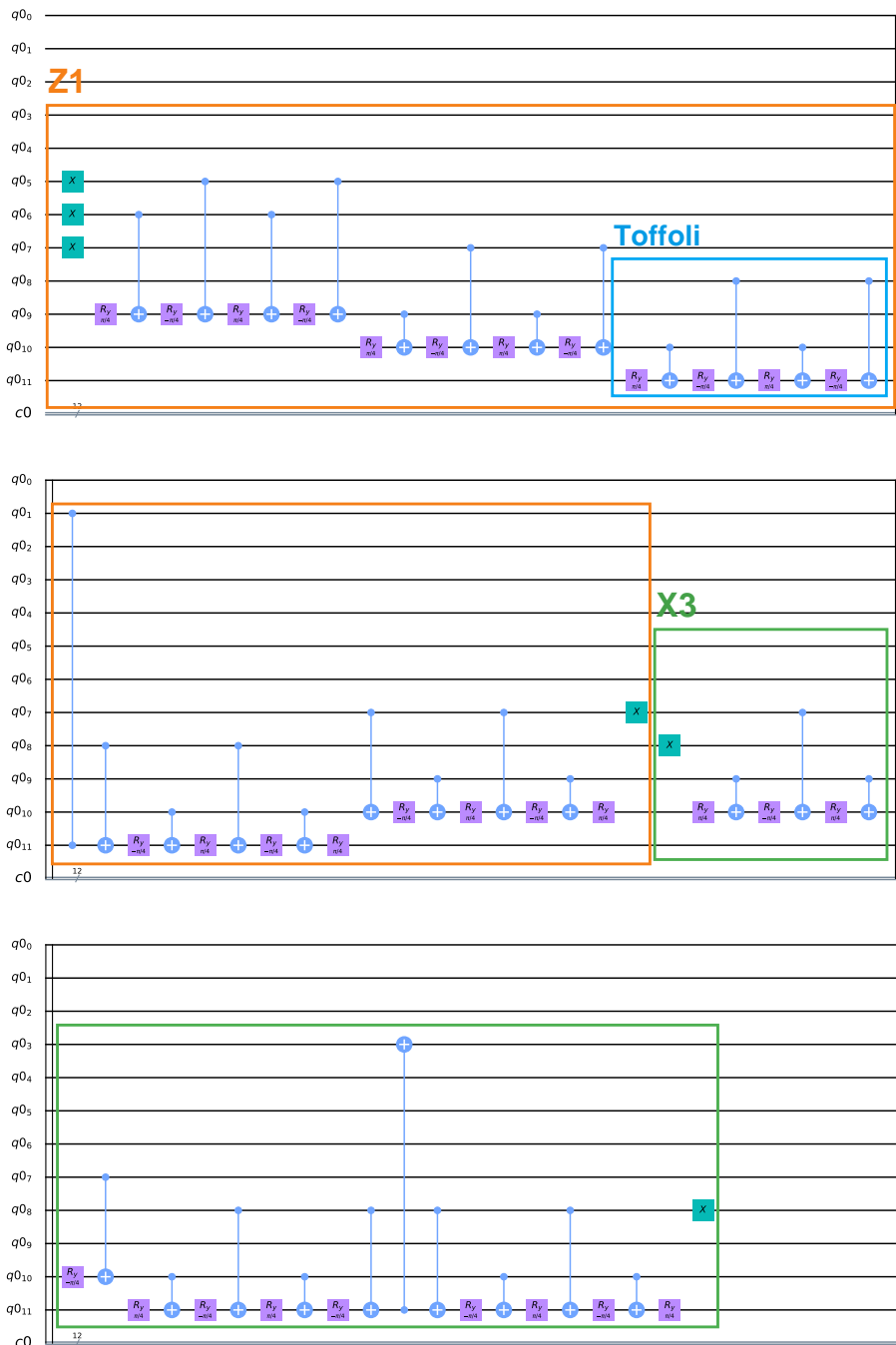


Figure 5.11: A snippet of implementation of error correction for the five qubit code.

---

## 5.2.4 State decoding

The decoding circuit for the five-qubit code is shown in Figure 3.18. It was proposed based on the encoding procedure illustrated in Figure 3.15, and the fact that quantum operators are unitary. Figure 5.12 shows the circuit generated from the Python code.

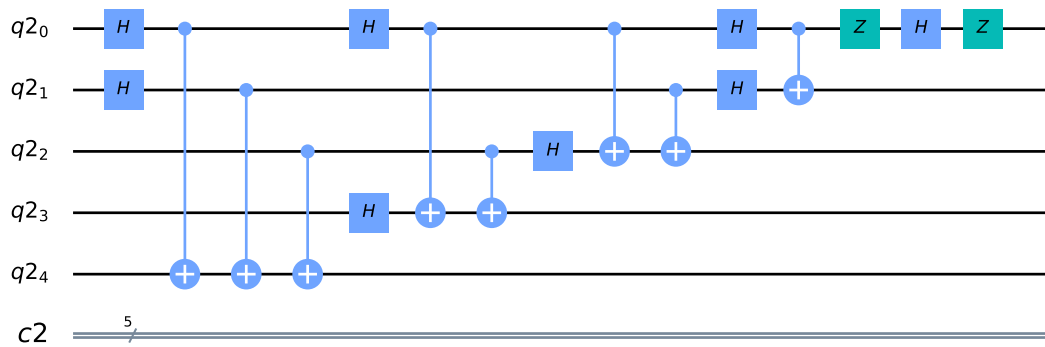


Figure 5.12: Implementation of decoding for the five qubit code.

## 5.3 Implementation of the three-qubit bit-flip code

The three-qubit bit-flip code was also realized on IBM Q as a part of this thesis. This code was chosen for implementation due to its simplicity, for the purpose of comparing its efficacy with the efficacy of the five-qubit perfect code.

This code is theoretically described in section 3.3, and realization of encoding, syndrome measurement, error correction and decoding for this code is shown in sections 5.3.1, 5.3.2, 5.3.3 and 5.3.4 respectively. Although the circuits for all steps of the three-qubit bit-flip code are known, the syndrome measurement and error correction circuits had to be transformed as a part of this thesis to be applicable on IBM Q.

### 5.3.1 State encoding

The theoretical state encoding procedure is presented in Figure 3.2, and it was implemented without introducing any changes. The equivalent quantum circuit, generated from the Python implementation of the three-qubit bit-flip code, is shown in Figure 5.13.

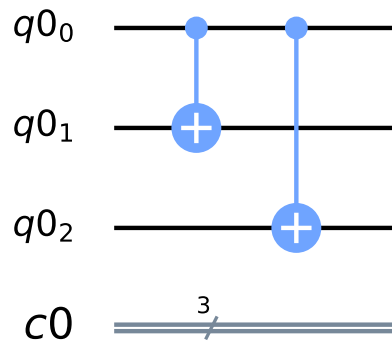


Figure 5.13: Implementation of encoding for the three-qubit bit-flip code.

### 5.3.2 Syndrome measurement

To realize the syndrome measurement procedure for the three-qubit bit-flip code, shown in Figure 3.4, with Qiskit, the controlled gates with two target qubits was decomposed into single-target controlled gates like it's shown in section 5.2.2. The controlled gate with a multi-qubit target operator  $cZ_iZ_j$  was decomposed as presented in Equation 5.5.

$$cZ_iZ_j \equiv cZ_icZ_j. \quad (5.5)$$

The result quantum circuit is presented in Figure 5.14.

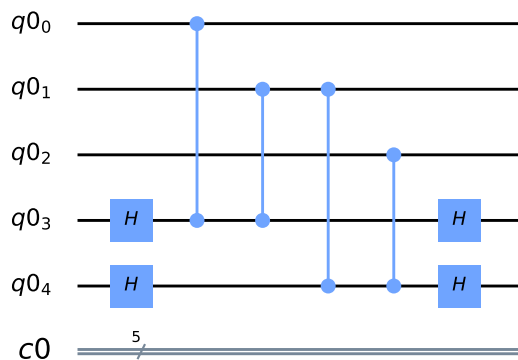


Figure 5.14: Implementation of syndrome measurement for the three-qubit bit-flip code.

### 5.3.3 Error correction

The theoretical error correction procedure for the three-qubit bit-flip code is presented in Figure 3.5. To implement it with Qiskit, the Toffoli gate with three target qubits had to be split to three Toffoli gates with a single target qubit, by analogy to decomposing controlled gates with one control qubit. The gate  $ccZ_0Z_1Z_2$  was transformed as shown in Equation 5.6.

$$ccZ_0Z_1Z_2 \equiv ccZ_0ccZ_1ccZ_2 \quad (5.6)$$

Figure 5.15, generated from the Python code, shows the circuit realized on IBM Q.

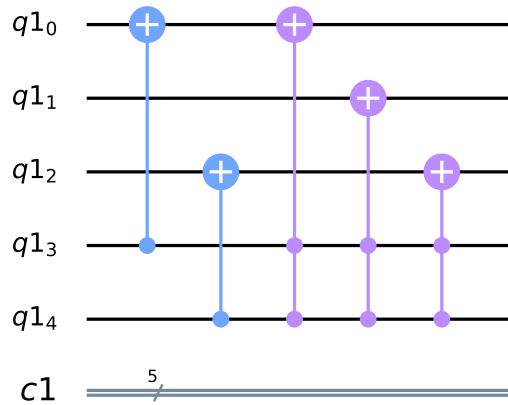


Figure 5.15: Implementation of error correction for the three-qubit bit-flip code.

### 5.3.4 State decoding

In the case of the three-qubit bit-flip code, the encoding and decoding procedures are the same, as can be noticed in Figures 3.7 and 3.6. The decoding procedure was realized on IBM Q unchanged. The circuit generated with Qiskit is shown in Figure 5.16.



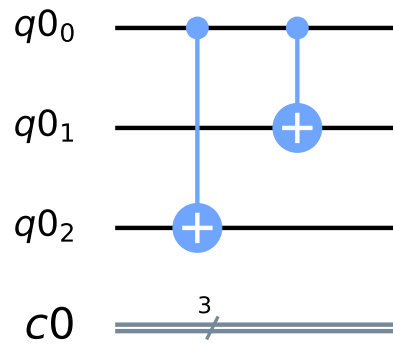


Figure 5.16: Implementation of decoding for the three-qubit bit-flip code.

## 5.4 Summary

In this chapter, the new implementation of the five-qubit perfect code on IBM Q was presented. Changes to the theoretical circuits were explained, and a procedure for error correction was proposed. Also, the implementation of the three-qubit bit-flip code on IBM Q was shown. In the next chapter, the efficacy of both quantum error-correcting codes in a depolarizing channel is assessed and compared.

---

# Chapter 6

## Evaluation

In this chapter, an evaluation of the realization of the perfect five-qubit code and the three-qubit bit-flip code, which were presented in Chapter 5, will be shown. First, the correctness of both methods will be experimentally verified. Then, the effectiveness of both codes in the presence of quantum noise will be measured and compared.

### 6.1 Experimental verification of correctness

This section shows the verification of the correctness of implemented quantum error-correcting codes. The verification procedure for both codes was carried out on the **ibmq qasm simulator**. By default, this device simulates a perfect quantum computer, so the results obtained from it are compliant with theoretical results.

In each experiment, quantum errors were introduced to the circuits, after the state encoding procedure, and before syndrome measurement, by applying quantum operators. Then, it was verified that the originally encoded state is preserved using the quantum error-correcting code. It was also verified that the error syndrome stored in ancilla qubits correctly indicates the error that corrupted the encoded state. In all experiments, the measurements were performed in the computational basis. All experiments were initialized with the  $|0\rangle$  state and included 1024 circuit executions.

#### 6.1.1 The five-qubit perfect code

The implemented five-qubit code is expected to correct an arbitrary, single qubit-quantum error on one of the code qubits. In this section, a set of experiments aimed at proving the correctness of the code is presented.

Nine experiments were run in total. The results of all experiments are given in Table 6.1, where the first column contains the experiments' identifiers. The second and third columns indicate what gate was applied to which qubit to simulate a quantum error, respectively. The fourth and the last columns show all of the output states measured in an experiment, together with the percentage of occurrences of a particular output state in this experiment.

---

No.	Gate	Qubit	Output	Percentage
1.	X	2	$ 000000101000\rangle$	100%
2.	Y	2	$ 000001101000\rangle$	100%
3.	Z	2	$ 000001000000\rangle$	100%
4.	X	4	$ 000001001000\rangle$	100%
5.	Y	4	$ 000001111000\rangle$	100%
6.	Z	4	$ 000000110000\rangle$	100%
7.	H	3	$ 000001100000\rangle$	50.7%
			$ 000000010000\rangle$	49.3%
8.	$U_3$	0	$ 000000000000\rangle$	25.3%
			$ 000000100000\rangle$	28.1%
			$ 000000111000\rangle$	44.8%
			$ 000000011000\rangle$	1.8 %
9.	$U_3$	1	$ 000000000000\rangle$	1%
			$ 000001011000\rangle$	60.9%
			$ 000001010000\rangle$	9.5%
			$ 000000001000\rangle$	28.6 %

Table 6.1: Results of verification experiments for the five-qubit perfect code.

In the first six experiments, all possible Pauli errors were applied to qubits 2 and 4, one type of error to one qubit at a time. Obtained results for each of these experiments showed that the initial state, stored in qubit 0, is preserved with 100% accuracy. Moreover, the measured error syndromes, stored in qubits 5 to 8, are compliant with expected ones, shown in Table 3.5.

In the experiment no. 7, it was confirmed that the code corrects not only Pauli errors but also an arbitrary single-qubit error. The error corresponding to the Hadamard gate, which takes a state into an equal superposition of the two basis states, was applied to the qubit 3. In this case, the initial state is always  $|0\rangle$ , as expected. Moreover, the observed error syndromes are  $|1100\rangle$  and  $|0010\rangle$ , each with probabilities close to 50%. These error syndromes correspond to the  $Z_3$  and  $X_3$  errors, as can be read in Table 3.5. It is the expected result as

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \frac{1}{\sqrt{2}} \left[ \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \right] = \frac{1}{\sqrt{2}} [Z + X]. \quad (6.1)$$

Finally, experiments no. 8 and 9. were performed to test the implementation of the five-qubit perfect code against arbitrary single-qubit quantum errors. Randomly selected with each execution, arbitrary single-qubit errors were applied to qubits 0 and 1, one qubit at a time. It can be seen that with 100% accuracy, the initial  $|0\rangle$  state is preserved. Moreover, for both tested qubits, measured states indicate all possible error syndromes for errors that can occur on this qubit, with different probabilities.

---

### 6.1.2 The three-qubit bit-flip code

The implemented three-qubit bit-flip code should correct the bit-flip, or X, error on any code qubit. In this section, a set of experiments aimed at proving the correctness of the code is discussed.

In Table 6.2, the results of all verification experiments for the three-qubit bit-flip code are presented. The interpretation of data in each column is the same as for Table 6.1.

No.	Gate	Qubit	Output	Percentage
1.	X	0	00010⟩	100%
2.	X	1	00011⟩	100%
3.	X	2	00001⟩	100%

Table 6.2: Results of verification experiments for the three-qubit bit-flip code.

Three experiments were carried out in total, and each of them tested the resilience of the code from the bit-flip error on subsequent code qubits. In all experiments, the initial  $|0\rangle$  state was preserved with 100% accuracy. What is more, the obtained error syndromes for each error are compliant with the expected ones, presented in Table 3.1.

#### Summary

The verification experiments have shown that the five-qubit perfect codes' implementation preserves the initial state with 100% accuracy if an arbitrary error occurs on one of the code qubits. Moreover, the experiments for the three-qubit bit-flip code have confirmed that this code corrects a single bit-flip error on any code qubits 100% of the times.

## 6.2 Evaluation with Randomized Benchmarking

An evaluation procedure was performed to assess the efficiency of implemented quantum error-correcting codes. In the presence of artificial quantum noise, both codes were tested using the randomized benchmarking protocol, described in section 3.7.3, and quantum state tomography, described in section 3.7.1. The efficacy of implemented codes was compared to the one of a raw qubit.

#### Technologies

The evaluation procedure for realized quantum error-correcting codes was implemented in the Python 3.7 programming language [9]. Benchmarking methods available in Qiskit [6], especially in Qiskit Ignis, were used. The Matplotlib library [7] was used for generating plots, and measured data was operated on using the SciPy library [8].

---

## Environment configuration

Both of the implemented codes can be run on a device with at least twelve qubits and, at first, the experiment was meant to be run on a real quantum device, the fifteen-qubit **ibmq 16 melbourne**. The first obstacle was that, at first, the five-qubit perfect code could not be run on this device due to too large a number of operations. This problem was dealt with by reducing the number of used Toffoli gates, and also using an alternative implementation of this gate, as mentioned in Section 5.2.3. However, the noise of **ibmq 16 melbourne** was too high to be able to obtain interpretable results from runs on this backend.

Instead, the experiment was run on the **ibmq qasm simulator**, and it was configured to simulate a real backend with slight quantum noise.

Before the experiment was executed, all circuits were transpiled to equivalent circuits built only with gates available in the **ibmq 16 melbourne**. The used gates set comprised of gates  $U_1, U_2, U_3$  and  $cNOT$ .

To simulate an environment of a real quantum device on **ibmq qasm simulator**, an artificial noise model was applied to it. A depolarizing channel, shown in Equation 6.2, was applied to all code qubits for the five-qubit code, and the three-qubit code, and to the raw qubit, where  $\rho$  denotes a general state of a qubit which is sent through the channel.

$$E(\rho) = 0.997\rho + 0.001(X\rho X^\dagger + Z\rho Z^\dagger + Y\rho Y^\dagger), \quad (6.2)$$

## Methodology

The aim of the described experiment was to test the ability of implemented quantum error-correcting codes to protect a single-qubit quantum state in a simulated environment, resembling a real quantum device environment. The general structure of a single run of the experiment was as follows.

Three identical, single qubit quantum states were initialized with the same quantum state. The first one was left untouched, the second one was encoded with the five-qubit perfect code, and the third one with the three-qubit bit-flip code. Then, sequences of randomly generated single-qubit Clifford gates were applied to the raw qubit and each qubit in the encoded states. After the syndrome measurement, error correction, and decoding procedures, the final measurements were performed, and the obtained output states were compared to the initial ones to see how well the original state was preserved with each method. A general scheme for running this experiment for a quantum-error correcting code is shown in Figure 6.1.

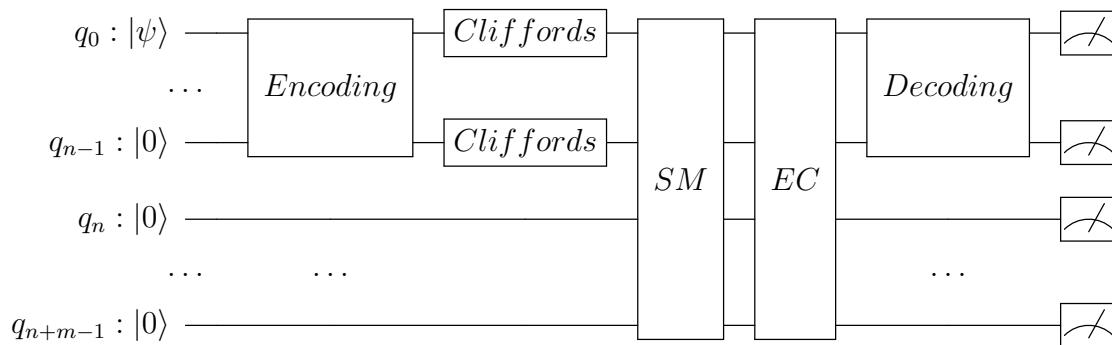


Figure 6.1: General scheme for a single run of the experiment for a  $n$ -qubit quantum error-correcting code using  $m$  ancilla qubits.

Such an experiment was performed for Clifford sequences of length from 1 to 1801 by 200. For each of the lengths, the experiment was performed five times to obtain averaged results.

In more detail, the following steps were included in the experiments.

1. Clifford sequences comprising only single-qubit Clifford gates, of lengths from 1 to 1801 by 200, were generated. For each length, five different sequences were generated to obtain averaged results.
2. Generated Clifford sequences were applied to both quantum error-correcting codes and the raw qubit. In the case of quantum-error correcting codes, Clifford sequences were applied to code qubits, one sequence to one qubit, between the encoding and syndrome measurement procedures, as shown in Figure 6.1.
3. The obtained circuits were transpiled to the set of available gates  $U_1$ ,  $U_2$ ,  $U_3$ , and  $cNOT$ .
4. Transpiled circuits were executed on ibmq qasm simulator with artificial noise of the form in Equation 6.2. Each circuit was executed 1024 times.
5. For each circuit, the density matrices for the output states were reconstructed using quantum state tomography.
6. From obtained density matrices, state fidelities and purities were calculated for each output state. Then, fidelities and purities for all Clifford sequences of the same length and the same error-correction method were averaged.

The experiment was performed for initial states  $|0\rangle$  and  $|1\rangle$ .

## Results

The results of run experiment were visualized using the Matplotlib library [7]. Plots showing the changes in average fidelities of measured output states for increasing length of Clifford sequences for initial states  $|0\rangle$  and  $|1\rangle$  are illustrated in

Figures 6.2 and 6.3, respectively. On the other hand, Figures 6.4 and 6.5 present the changes in the purity of measured output states based on the length of used Clifford sequence. The full results are given in Appendix A.

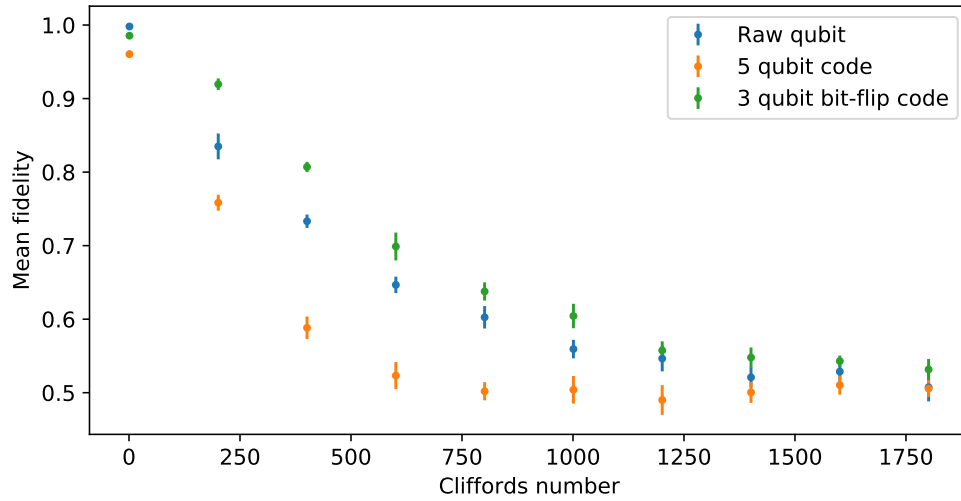


Figure 6.2: Fidelity comparison for the  $|0\rangle$  state.

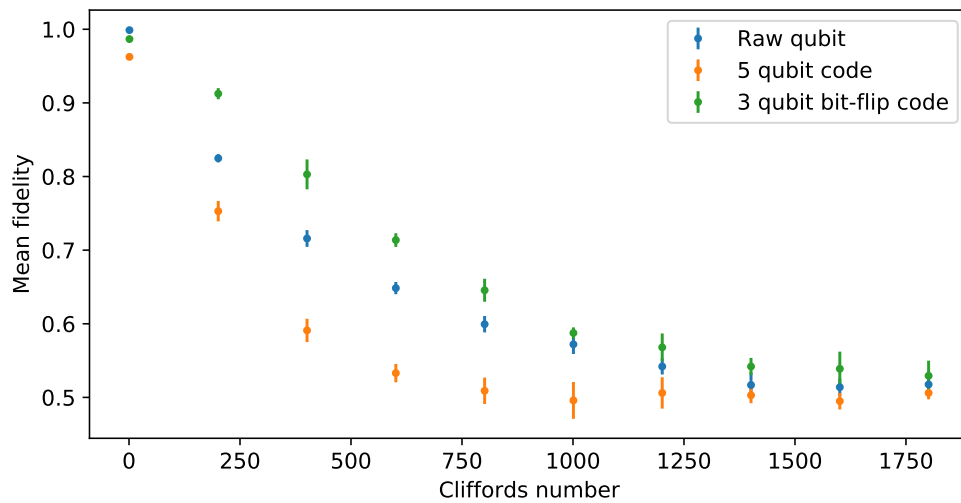


Figure 6.3: Fidelity comparison for the  $|1\rangle$  state.

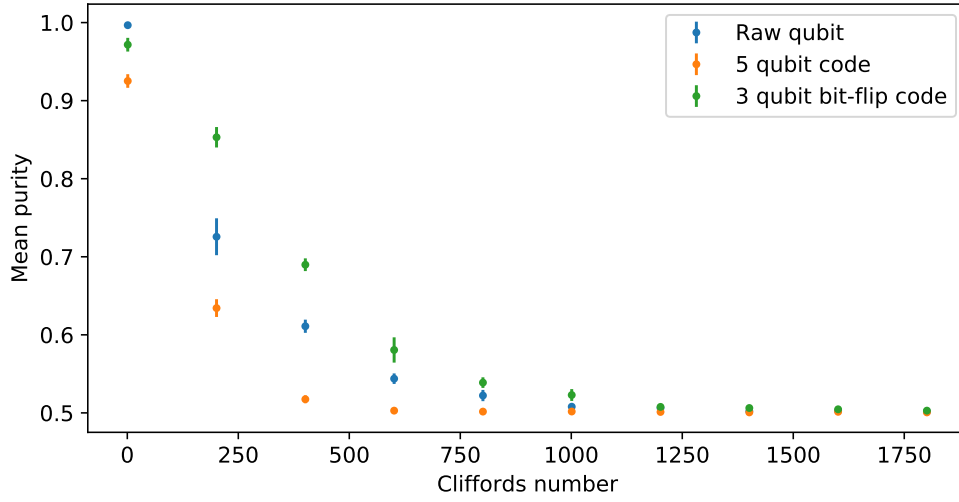


Figure 6.4: Purity comparison for the  $|0\rangle$  state.

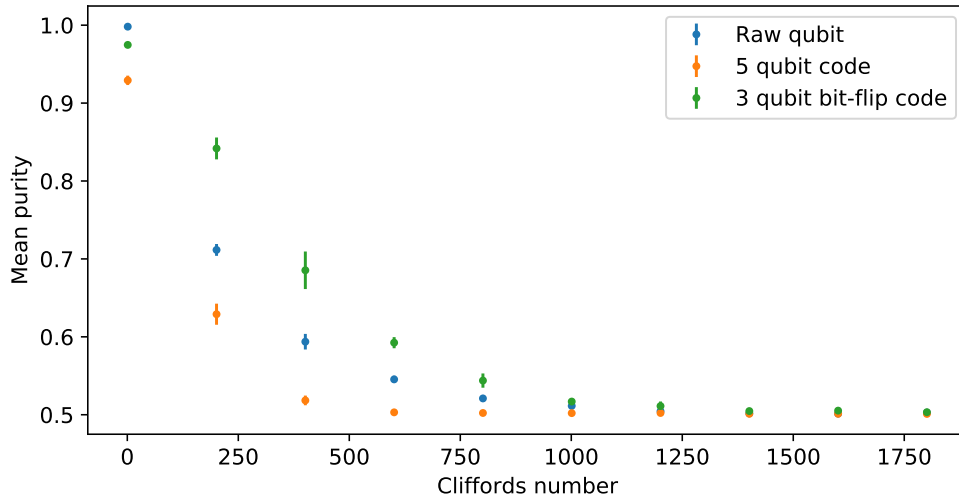


Figure 6.5: Purity comparison for the  $|1\rangle$  state.

The shape of visualized data for both initial states and for both fidelities as well as purities resembles exponential functions. In fact, in the general Randomized Benchmarking protocol, it is suggested that the dependence of how well is the original qubit's state preserved on the length of the applied Clifford sequence is an exponential function of the form

$$a\alpha^n + b, \tag{6.3}$$

where  $n$  is the length of the Clifford sequence [29].

The result data were fitted to exponential decays with the use of the SciPy library [8]. The calculated formulas interpolating measured fidelities are listed in Table 6.3, and fitted formulas for state purities are given in Table 6.4.



---

Method	Initial state	Formula
Raw qubit	$ 0\rangle$	$0.5044 * 0.9980^n + 0.4946$
Raw qubit	$ 1\rangle$	$0.4970 * 0.9979^n + 0.5029$
5-qubit code	$ 0\rangle$	$0.4712 * 0.9966^n + 0.4926$
5-qubit code	$ 1\rangle$	$0.4678 * 0.9964^n + 0.4967$
3-qubit code	$ 0\rangle$	$0.5680 * 0.9990^n + 0.4255$
3-qubit code	$ 1\rangle$	$0.6019 * 0.9990^n + 0.3869$

Table 6.3: Fitted exponential functions for state fidelity data.

Method	Initial state	Formula
Raw qubit	$ 0\rangle$	$0.4979 * 0.9961^n + 0.5007$
Raw qubit	$ 1\rangle$	$0.4992 * 0.9959^n + 0.5009$
5-qubit code	$ 0\rangle$	$0.4303 * 0.9928^n + 0.5007$
5-qubit code	$ 1\rangle$	$0.4321 * 0.9925^n + 0.5012$
3-qubit code	$ 0\rangle$	$0.4948 * 0.9974^n + 0.4962$
3-qubit code	$ 1\rangle$	$0.4814 * 0.9972^n + 0.4967$

Table 6.4: Fitted exponential functions for state purity data.

It can be noted that in an ideal quantum computer, the output states would always be equal to the initial state. For that reason, calculated fidelities and purities would always be equal to 1, so the Equation 6.3 would be a constant function  $y = 1$ . It would be the case if, for example,  $\alpha$  was equal to 1, and  $a + b$  was equal to 1. It can be deduced that the closer the fitted exponential to a constant function  $y = 1$ , the better the code for which the output state's fidelity and purity decay with this function.

## Conclusions

1. It can be noticed that the input state does not impact how well a particular error correcting method works. As for the average fidelity of the output states, the base of the exponent for fitted exponential functions differs merely by  $\pm 0.0001$ ,  $\pm 0.0002$ , and  $\pm 0$  for the raw qubit, the five-qubit code, and three-qubit code, respectively. In the case of the average purities of output states, the differences in the bases of the exponents are  $\pm 0.0002$ ,  $\pm 0.0003$ , and  $\pm 0.0002$  for the raw qubit, the five-qubit code, and three-qubit code, respectively.
2. For all experiments, the three-qubit bit-flip code best preserved both the fidelity and purity of output states. As for the average state fidelities, the calculated base of the exponent function is the closest to 1, by  $0.00105 \pm 0.00005$  closer than the raw qubit and by  $0.0025 \pm 0.0001$  closer than the five-qubit code. In the case of the purity of output states, the base of the exponent of

---

fitted function for the three-qubit bit-flip code is also the closest to 1 compared to other methods. It's by  $0.0013 \pm 0.0002$  closer than the raw qubit, and by  $0.00465 \pm 0.00025$  closer than the five-qubit perfect code.

3. The superiority of the three-qubit bit-flip code over the raw qubit shows that using simple, repetition codes can improve quantum computation's fidelity.
4. The fact that the five-qubit code gave worse results than the other two methods may be caused by the number of operations that it is built with, as these operations introduce noise to the circuit themselves.
5. From the exact results given in Appendix A, it can be noticed, both the fidelity and the purity of measured output states converges to the value 0.5, indicating that the state becomes a maximally mixed state.

## 6.3 Summary

In this chapter, the experiments aimed at proving the correctness of implemented codes, and evaluating these codes, were described. Conclusions drawn from the assessment of the effectiveness of implemented quantum error-correcting methods can be used to improve the fidelity and purity of the results obtained from quantum computations and stand a basis for further studies. In the next chapter, a summary of the whole thesis is given, and thoughts on possible future work are expressed.

---

# Chapter 7

## Summary, conclusions and future work

This section summarizes the thesis. First, an overview of what has been done as a part of this work is given. Then the conclusions that can be drawn from the obtained results of run experiments are listed. Moreover, the possibilities for future work regarding topics covered in this thesis are outlined.

### 7.1 Summary

The work done as a part of this thesis began with research conducted in the subjects of quantum computation and quantum error-correcting methods, in particular quantum error-correcting codes. Special attention was paid to existing applications of quantum error-correcting methods on IBM Q. The results of this study are covered in Chapters 2, 3 and 4.

In the next step, the Qiskit library and architecture of quantum devices available through IBM Q were studied, and two quantum error-correcting codes were chosen to be realized on this quantum computing platform - the five-qubit perfect code and the three-qubit bit-flip code. Changes to existing circuits, as well as a new, optimized circuit for the error correction procedure for the five-qubit perfect code, were proposed, and both codes were implemented using the Qiskit library. To the best of the author's knowledge, it was the first realization of the five-qubit perfect code on IBM Q. The details of the implementation can be found in Chapter 5.

Several experiments were carried out, which confirmed the correctness of implemented quantum error-correcting methods. Then, the efficiency of both codes in the presence of quantum noise was tested on ibmq qasm simulator, using techniques such as randomized benchmarking and quantum state tomography. The results obtained from these experiments are discussed in Chapter 6.

### 7.2 Conclusions

The results of this thesis show that

- 
1. The five qubit perfect and the three-qubit bit-flip code can be successfully realized on IBM Q.
  2. The fidelity and purity of quantum computation can be improved using simple repetition codes, such as the three-qubit bit-flip code.
  3. A simple method can outperform more powerful, complex quantum-error correction methods, able to correct more types of errors. It shows that it's best to use a quantum error-correcting method built with as few quantum operators as possible. However, it may change with the development of existing quantum devices.

### 7.3 Future work

As a part of this thesis, it has been shown that the three-qubit bit-flip code improves the fidelity of quantum computation. This should be further investigated, and the abilities of this error-correcting code on a real quantum device should be assessed.

The new realization of the five-qubit-perfect code on IBM Q indicates that there still are new, theoretically known quantum error-correcting methods, which can be implemented using this platform, for example, the seven-qubit  $[[7, 1, 3]]$  code [10]. Moreover, combining quantum-error correcting codes with other quantum error mitigation methods, such as the calibration of qubits [6] is also worth studying. The new, implemented methods could be then compared to the ones realized in this thesis.

Moreover, more experiments can be carried out to test the effectiveness of quantum error-correcting codes realized as part of this thesis in the presence of different quantum noise types. It would be very valuable to use a real quantum device for this purpose.

Another possible direction of future works would be to implement the five-qubit perfect code and the three-qubit bit-flip code on a different quantum platform, for example the D-Wave 2000Q [38], and compare the efficacy of this method on different architectures of quantum computers.

# List of Figures

3.1	General structure of a quantum error-correcting code. SM stands for Syndrome Measurement, and EC for Error Correction. . . . .	23
3.2	Quantum state encoding for the three-qubit bit-flip code. . . . .	24
3.3	Syndrome measurement circuit for the three-qubit bit-flip code. . . . .	25
3.4	Syndrome measurement circuit for the three-qubit bit-flip code after transformations. . . . .	25
3.5	Error correction circuit for the three-qubit bit-flip code. . . . .	26
3.6	Decoding circuit for the three-qubit bit-flip code. . . . .	26
3.7	State encoding circuit for the three-qubit phase-flip code. . . . .	27
3.8	Syndrome measurement circuit for the three-qubit phase-flip code. . . . .	28
3.9	Error correction circuit for the three-qubit phase-flip code. . . . .	28
3.10	State decoding circuit for the three-qubit phase-flip code. . . . .	29
3.11	Encoding circuit for the Shor code. . . . .	30
3.12	Syndrome measurement circuit for the Shor code. . . . .	31
3.13	Error correction circuit for the Shor code. . . . .	33
3.14	Decoding circuit for the Shor code. . . . .	34
3.15	Encoding circuit for the five-qubit code. . . . .	35
3.16	Syndrome measurement circuit for the five-qubit code. . . . .	36
3.17	Circuit correcting the $X_0$ error. . . . .	37
3.18	Decoding circuit for the five-qubit code. . . . .	37
5.1	Implementation of encoding for the five-qubit code. . . . .	45
5.2	Multi-target controlled gate. . . . .	46
5.3	Decomposed multi-target controlled gate. . . . .	46
5.4	Implementation of syndrome measurement for the five-qubit code. . . . .	47
5.5	General four-controlled gate. . . . .	48
5.6	General four-controlled gate build with Toffoli gates. . . . .	49
5.7	Decomposition of Toffoli gate with six cNOT gates. . . . .	49
5.8	Decomposition of Toffoli gate with four cNOT gates. . . . .	50
5.9	Example of adjacent four-controlled gates application. . . . .	50
5.10	Example of adjacent four-controlled gates application after optimization. . . . .	51
5.11	A snippet of implementation of error correction for the five qubit code. . . . .	53
5.12	Implementation of decoding for the five qubit code. . . . .	54
5.13	Implementation of encoding for the three-qubit bit-flip code. . . . .	55
5.14	Implementation of syndrome measurement for the three-qubit bit-flip code. . . . .	55
5.15	Implementation of error correction for the three-qubit bit-flip code. . . . .	56

---

5.16	Implementation of decoding for the three-qubit bit-flip code. . . . .	57
6.1	General scheme for a single run of the experiment for a n-qubit quantum error-correcting code using m ancilla qubits. . . . .	62
6.2	Fidelity comparison for the $ 0\rangle$ state. . . . .	63
6.3	Fidelity comparison for the $ 1\rangle$ state. . . . .	63
6.4	Purity comparison for the $ 0\rangle$ state. . . . .	64
6.5	Purity comparison for the $ 1\rangle$ state. . . . .	64

# List of Tables

2.1	Single-qubit quantum operators. . . . .	16
2.2	Multi-qubit quantum operators. . . . .	17
3.1	Interpretation of syndrome measurement for the three-qubit bit-flip code. . . . .	25
3.2	Interpretation of syndrome measurement for the three-qubit phase-flip code. . . . .	28
3.3	Interpretation of syndrome measurement for the Shor's code, part 1. . . . .	32
3.4	Interpretation of syndrome measurement for the Shor's code, part 2. . . . .	32
3.5	Interpretation of syndrome measurement for the five-qubit code. . . . .	36
6.1	Results of verification experiments for the five-qubit perfect code. . . . .	59
6.2	Results of verification experiments for the three-qubit bit-flip code. . . . .	60
6.3	Fitted exponential functions for state fidelity data. . . . .	65
6.4	Fitted exponential functions for state purity data. . . . .	65
A.1	Fidelity table for the $ 0\rangle$ state. . . . .	75
A.2	Fidelity table for the $ 1\rangle$ state. . . . .	76
A.3	Purity table for the $ 0\rangle$ state. . . . .	76
A.4	Purity table for the $ 1\rangle$ state. . . . .	77

# Bibliography

- [1] IBM Quantum Experience Platform, <https://quantum-computing.ibm.com/>.
- [2] P. Shor, "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer," *SIAM Journal on Computing*. 26. 1484-1509. 10.1137/S0097539795293172, 1997.
- [3] M. Stachoń; Master of Science Thesis supervised by K. Rycerz, "Solving optimisation problems using Qiskit Aqua," AGH University of Science and Technology, Department of Computer Science, Krakow, Poland 2020.
- [4] F. Galas; Master of Science Thesis supervised by K. Rycerz, "Quantum games on IBM-Q," AGH University of Science and Technology, Department of Computer Science, Krakow, Poland 2019.
- [5] Z. Chrzastek, M. Bubak, T. Stopa, and K. Rycerz, "Assessment of the IBM-Q Quantum Computer and Its Software Environment," CGW 18 International Workshop, 2018.
- [6] Héctor Abraham et al., "Qiskit: An Open-source Framework for Quantum Computing", 10.5281/zenodo.2562110, 2019.
- [7] Matplotlib, <https://matplotlib.org/>, 2020.
- [8] SciPy, <https://www.scipy.org/>, 2020.
- [9] Van Rossum, Guido and Drake Jr, Fred L, "Python tutorial", Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.
- [10] M. A. Nielsen, I. L. Chuang, "Quantum Computation and Quantum Information, 10th Anniversary Edition," Chapter 10, 10.1017/CBO9780511976667. 2010.
- [11] D. Gottesman, "Stabilizer Codes and Quantum Error Correction," 1997.
- [12] Daniel Gottesman, "Quantum Error Correction and Fault Tolerance," 12th Canadian Summer School on Quantum Information, June 2012.
- [13] N. David Mermin, "Lecture Notes on Quantum Computation," Cornell University, Physics 481-681, CS 483; Spring, 2006.
- [14] J. Preskill, "Lecture Notes for Physics 229: Quantum Information and Computation," September 1998.



- 
- [15] S. Devitt, W. Munro and K. Nemoto, “Quantum Error Correction for Beginners,” Reports on progress in physics. Physical Society (Great Britain). 76. 076001. 10.1088/0034-4885/76/7/076001, 2013.
- [16] J. Roffe, “Quantum error correction: an introductory guide,” Contemporary Physics. 60. 1-20. 10.1080/00107514.2019.1667078, 2019.
- [17] J. R. Wootton and D. Loss, “Repetition code of 15 qubits,” Physical Review A, vol. 97, no. 5, May 2018.
- [18] D. Gottesman, “Quantum fault tolerance in small experiments,” 2016.
- [19] J. Roffe, D. Headley, N. Chancellor, D. Horsman, and V. Kendon, "Protecting quantum memories using coherent parity check codes," Quantum Science and Technology. 3. 10.1088/2058-9565/aac64e, May 2018.
- [20] R. Harper and S. T. Flammia, “Fault-Tolerant Logical Gates in the IBM Quantum Experience,” Physical Review Letters, vol. 122, no. 8, Feb. 2019.
- [21] A. Hu and J. Li and R. Shapiro, “Quantum Benchmarking on the  $[[4, 2, 2]]$  Code,” 2018.
- [22] C. Vuillot, “Is error detection helpful on IBM 5Q chips?,” Quantum information computation. 18. 0949-, 2018.
- [23] P. Pandey, E. Sriram Prasath, M. Gupta and P. Panigrahi, “Automated Error Correction For Generalized Bell States,” 2010.
- [24] M. Sisodia, A. Shukla, A. Pathak, “Experimental realization of nondestructive discrimination of Bell states using a five-qubit quantum computer,” Phys. Lett. A 381(46), 3860–3874, 2017.
- [25] D. Ghosh, P. Agarwal, P. Pandey, B. K. Behera, and P. K. Panigrahi, “Automated error correction in IBM quantum computer and explicit generalization,” Quantum Information Processing, vol. 17, no. 6, May 2018.
- [26] P. A. M. Dirac, “A new notation for quantum mechanics,” Mathematical Proceedings of the Cambridge Philosophical Society, vol. 35, no. 3, pp. 416–418, 1939.
- [27] W. Pauli, “Zur Quantenmechanik des magnetischen Elektrons,” Z. Physik 43, 601–623, 1927.
- [28] D. Greenberger, M. Horne and A. Zeilinger, “Going Beyond Bell’s Theorem,” 10.1007/978-94-017-0849-4\_10, 2008.
- [29] E. Magesan, J. M. Gambetta, J. Emerson, “Scalable and Robust Randomized Benchmarking of Quantum Processes,” Physical review letters. 106. 180504. 10.1103/PhysRevLett.106.180504, 2011.
- [30] D. Gottesman, “The Heisenberg Representation of Quantum Computers,” 1998.

- 
- [31] Fabian Schmidt, “How a quantum computer works,” <https://www.dw.com/en/how-a-quantum-computer-works/a-50969000>, 2019.
- [32] Avery Thompson, “Scientists Have Made Transistors Smaller Than We Thought Possible,” <https://www.popularmechanics.com/technology/a23353/1nm-transistor-gate/>, 2016.
- [33] <https://qiskit.org/textbook/ch-quantum-hardware/randomized-benchmarking.html>
- [34] A. Matuschak and M. A. Nielsen, “How does the quantum search algorithm work?,” <https://quantum.country/search>, San Francisco 2019.
- [35] J. Sypień, Realization of the five-qubit perfect code, [https://github.com/Regulareveryday/ibmq\\_qec\\_applicability/blob/master/five\\_qubit\\_qecc\\_code\\_rot.py](https://github.com/Regulareveryday/ibmq_qec_applicability/blob/master/five_qubit_qecc_code_rot.py), 2020.
- [36] M. Mottonen, J. J. Vartiainen, V. Bergholm and M. M. Salomaa, “Transformation of quantum states using uniformly controlled rotations,” 2004.
- [37] <https://qiskit.org/documentation/stubs/qiskit.providers.aer.noise.NoiseModel.html>, 2019.
- [38] The D-Wave 2000Q System, <https://www.dwavesys.com/d-wave-two-system>, 2020.

---

# Appendix A

## Appendix to Chapter 6

Cliffords	Raw qubit		5-qubit code		3-qubit code	
	Mean	Std-dev	Mean	Std-dev	Mean	Std-dev
1	0.998044	0.001190	0.960352	0.005008	0.985547	0.004496
201	0.834961	0.017551	0.758398	0.010844	0.919531	0.007891
401	0.733203	0.009067	0.588281	0.015326	0.807031	0.006926
601	0.646680	0.011100	0.523242	0.018444	0.698828	0.018931
801	0.602539	0.015254	0.501953	0.012391	0.637695	0.012391
1001	0.559375	0.012571	0.503906	0.018746	0.604297	0.016636
1201	0.546484	0.017477	0.490039	0.020243	0.557422	0.012418
1401	0.520898	0.013786	0.500391	0.014229	0.547852	0.013672
1601	0.528711	0.012422	0.510352	0.012967	0.542969	0.007564
1801	0.507422	0.019194	0.505664	0.011584	0.531641	0.014179

Table A.1: Fidelity table for the  $|0\rangle$  state.

Cliffords	Raw qubit		5-qubit code		3-qubit code	
	Mean	Std-dev	Mean	Std-dev	Mean	Std-dev
1	0.998821	0.000804	0.962500	0.002897	0.986719	0.002025
201	0.824805	0.005593	0.752930	0.013862	0.912500	0.007583
401	0.715820	0.011368	0.591016	0.015843	0.802930	0.020267
601	0.648438	0.008286	0.533008	0.012476	0.713672	0.009275
801	0.599414	0.011100	0.508984	0.017893	0.645508	0.015610
1001	0.572070	0.013091	0.495898	0.025006	0.587500	0.007583
1201	0.541992	0.010853	0.506055	0.021299	0.567969	0.018893
1401	0.516797	0.017296	0.502930	0.010764	0.541992	0.011637
1601	0.513867	0.012068	0.494922	0.011250	0.538867	0.023289
1801	0.517578	0.017373	0.506055	0.008663	0.529297	0.020612

Table A.2: Fidelity table for the  $|1\rangle$  state.

Cliffords	Raw qubit		5-qubit code		3-qubit code	
	Mean	Std-dev	Mean	Std-dev	Mean	Std-dev
1	0.996667	0.002065	0.925177	0.008715	0.971670	0.008777
201	0.725647	0.023630	0.634262	0.011367	0.853091	0.013098
401	0.611006	0.008437	0.517459	0.005035	0.689802	0.008149
601	0.543823	0.006751	0.502872	0.002680	0.580606	0.016187
801	0.522215	0.007199	0.501646	0.000665	0.538797	0.006788
1001	0.507913	0.002999	0.501770	0.001472	0.522889	0.007579
1201	0.506275	0.003321	0.501198	0.000827	0.507687	0.003268
1401	0.502047	0.001182	0.500629	0.000214	0.506252	0.002265
1601	0.502692	0.001718	0.501373	0.000755	0.504613	0.001469
1801	0.501660	0.001462	0.500639	0.000607	0.502879	0.001562

Table A.3: Purity table for the  $|0\rangle$  state.

---

Cliffords	Raw qubit		5-qubit code		3-qubit code	
	Mean	Std-dev	Mean	Std-dev	Mean	Std-dev
1	0.998088	0.001522	0.929189	0.006026	0.974681	0.003381
201	0.711541	0.007596	0.629025	0.013526	0.841788	0.014034
401	0.593740	0.010011	0.518375	0.006224	0.685402	0.024097
601	0.545394	0.004657	0.503079	0.001601	0.592469	0.007125
801	0.520942	0.004434	0.502259	0.002025	0.543842	0.009143
1001	0.511337	0.004335	0.502073	0.000956	0.516907	0.003447
1201	0.504459	0.001634	0.502257	0.001385	0.511158	0.005837
1401	0.501705	0.001228	0.501219	0.001252	0.504712	0.001541
1601	0.501404	0.001114	0.501234	0.000590	0.505255	0.003845
1801	0.502694	0.002497	0.501057	0.000837	0.503261	0.001767

Table A.4: Purity table for the  $|1\rangle$  state.