Quantum Error Correction Methods

Yashar Alizadeh

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Applied Mathematics and Computer Science

Eastern Mediterranean University
September 2016
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Mustafa Tümer
Acting Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Applied Mathematics and Computer Science.

Prof. Dr. Nazım Mahmudov
Chair, Department of Mathematics

We certify that we have read this thesis and that in our opinion, it is fully adequate in scope and quality, as a thesis of the degree of Master of Science in Applied Mathematics and Computer Science.

Asst. Prof. Dr. Mustafa Riza
Supervisor

Examining Committee

1. Assoc. Prof. Dr. S.Habib Mazharimousavi

2. Asst. Prof. Dr. Arif Akkeleş

3. Asst. Prof. Dr. Mustafa Riza

ABSTRACT

This study surveys the mathematical structure of a quantum error correcting codes and the way they are developed through certain stages of error correction. In particular, the families of Calderbank-Shor-Steane codes (CSS) and the stabilizer codes are discussed and through elaborative examples it will be shown that the CSS codes are in the family of the stabilizer codes. Since the study of the CSS codes depends on a firm knowledge of classical coding theory, a rigorous mathematical review of the linear codes is done separately. Analysing the structure of the stabilizer formalism is highly depended on the effective use of some group theoretic notions. This structure is discussed in more detail and examples will be given. As the ultimate application of the quantum error correction the rules of the fault-tolerant quantum computing is explored and finding the threshold condition of an example will be done.

Keywords: QEC, Coding theory, Stabilizer formalism, CSS codes, Fault-tolerant quantum computing, Threshold condition

ÖZ

Bu çalışma kuantum hata düzeltimin kodlarının matematiksel yapısını ve belli hata düzeltilim evrelerinden nasıl geçtiğini incelemektedir. özellikle, Calderbank-Shor-Steane (CSS) kod ailesi ve stabilizatör kodları ele alınarak, ve ayrıca ayrıntılı örnekler ile CSS kodları stabilizatör kodlar ailesinden olduğunu gösterilmektedir. CSS kodları klasik kodlama teorisine dayandığı için, matematiksel ayrıntılı bir şekilde lineer kodlar gözden geçirilmiştir. Stabilizatör biçimciliğin strüktürünün analizi gurup teorisi tabanında yapılmıştır. Bu biçimcilik detaylı şekilde tartışılacaktır ve örneklerle desteklenecektir. Kuantum hata düzeltimi kurallarının en uç uygulaması kusura dayanıklı kuantum hesaplamaları incelenmiştir ve bir örnekte eşik seviyesinin nasıl bulunduğu gösterilecektir.

Anahtar Kelimeler:Kuantum Hata Düzeltme, Kodlama teorisi, Stabilizör biçimciliği, CSS kodları, hata düzeltimi kuantum hesaplamalar, eşik seviyesi şartı

***To Nikou**,for her love and devotion*

ACKNOWLEDGMENT

I would like to express my sincere gratitude to my supervisor Asst. Prof. Dr. Mustafa Riza for his guidance and patience as I was writing this thesis. His motivating discussions were of utmost help all the way through different stages of the thesis.

My gratitude also goes to all the academic members of the Department of Mathematics at the Eastern Mediterranean University whose knowledge shone light on the way and helped me finish the Master's of Applied Mathematics and Computer Science.

I would also like to thank my wife, Nikou, who made this thesis possible with her great patience and love during the hard time that we had.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

In 1984 Bennett and Brassard [1] invented a cryptographic key distribution using the principles of quantum mechanics. The basic idea was if an Eve wants to read the message sent by Alice to Bob through a transmission channel, she needs to measure it and since the measurement destroys the state of a quantum system the existence of the Eve can be detected. However the dawn of quantum computing goes back 1982 when Richard Feynman[30] put forward his conjecture that there is no general purpose method that can simulate the evolution of a quantum system on a classical computer. Later in 1985 D. Deutsch [3] proposed the concept of a quantum Turing machine which is supposed to efficiently simulate any physical system including quantum systems. A formal definition like this established a more formal way to think about quantum computing however it was until early 90's that the first quantum algorithms appeared[4],[2],[12],[11]. The first of these algorithms was created by Deutsch and Jozsa[4] in 1992 and speeded up the procedure of finding the solution of a classical problem. As others' attempts were turning into victory the field of quantum computation proved itself worthy of finding fast solutions for classical problems which were known as NP-complete problems. The Shor algorithm [11],[10] contrived in prime factorization of large integers in an efficient way and succeeded to break the RSA public-key cryptosystem (This is the cryptosystem on which the internet security is based).

Furthermore, L. Grover[5] made a great effort to propose an algorithm which made a speed-up in unstructured search and found applications in optimization problems. And finally Seth Lloyd[13] responded to Feynman's conjecture by inventing the quantum simulator algorithm in 1996. Meanwhile a question of the plausibility of making a quantum computer which can perform these quantum computations was being asked. The question stems from the extreme vulnerability of the quantum systems and the fact that their interaction with the environment or the control devices make them loose their quantumness and act like classical systems. In mid 90's the pioneers of quantum computing came up with protocols to correct for the errors arising in noisy computation with quantum systems[26],[16]. While few proposals for the special-purpose quantum computers, usually running a certain algorithm have been implemented to date, there is still a long way to constructing a scalable general-purpose quantum computer due to the substantial noise effects of the environment and controlling devices and the entanglement of the many qubits existing in such a quantum computer

This thesis is devoted to the study of the algorithms called quantum error correcting codes (QECC). In addition, I give the basic concepts and methods in coding theory upon which the QEC is developed since its advent. The tools from quantum computation needed to understand and analyse these methods are also given in the preliminaries. To start analysing the framework of quantum error correction we need to know about its more general stages, modelling the error, encoding, and error recovery steps. This can be done using a formalism called operator-sum representation. To instance the general scheme of QEC we look at the simplest case, the three-qubit code and

compare it to its classical version and then move on to other examples such as Shor nine-qubit code. These examples are repeated throughout the thesis since different aspects of quantum error correcting codes can be well clarified using them. The nature of error in quantum computation and also the condition of quantum error correction is discussed and compared with their classical counterparts. To this end, the formal construction of QEC is pinned down, however the properties and algorithms of the main families of quantum codes need to be discussed which will be brought into the focus of this thesis. The family of the CSS codes and then the greater family of the stabilizer codes will be met and their structures and stages of error correction will be discussed in detail. Finally the theory of fault-tolerant quantum computing which is the ultimate application of the quanmtum error correction will be inspected.

Chapter 2

PRELIMINARIES

This chapter briefly reviews the postulates of quantum mechanics and the tools required in analysing the quantum error correction codes. It aims to give a review on the useful tools required to read the rest of this thesis on quantum error correction approaches and serves as a refresher for the reader and by no means is a review of either quantum mechanics or quantum computation.

2.1 Quantum Mechanics

2.1.1 Postulates of quantum mechanics

this section we review the postulate of quantum mechanics[28] and in an axiomatic approach we refer to these postulates whenever needed.

- **State Space Postulate**

  The state $|\psi\rangle$ of a quantum system is described using a unit vector in a Hilbert space H. In classical physics and in a two-state system with sates 0 and 1, the classical system can be either in state 0 or in state 1. If it is in a some possible state, the probability of the system being in that state can only be one and the probability of the system being in the other state is definitely zero. However this is not the case when it comes to quantum systems such as a two level atom, systems with two states of spin, etc. the system is in a superposition of states $|0\rangle$

and $|1\rangle$, each with some amplitude α_0 and α_1.

$$\alpha_0 |0\rangle + \alpha_1 |1\rangle \tag{2.1}$$

- **Evolution Postulate**

  The evolution of a closed quantum system can be described by a unitary operator. This means that if the initial state of a closed quantum system is $|\psi_i\rangle$, the state of this system after an evolution can be written as

$$|\psi_f\rangle = U |\psi_i\rangle \tag{2.2}$$

  The reversibility of quantum computation is actually based on the unitarity of quantum evolution. In quantum physics the continuous time evolution of a closed quantum system follows from the Schrödinger equation.

$$i\hbar \frac{d|\psi(t)\rangle}{dt} = H(t)|\psi(t)\rangle \tag{2.3}$$

  where $H(t)$ is the Hamiltonian of the quantum system.

- **Composition of Systems Postulate**

  When two quantum systems are treated as a single entity within a composite system, the Hilbert space of the combined system is the tensor product of the Hilbert spaces of each of the subsystems, $H_A \otimes H_B$. In this case the state of the system is $|\psi_A\rangle \otimes |\psi_B\rangle$, where $|\psi_A\rangle$ and $|\psi_B\rangle$ are the state of the two subsystems.

- **Measurement Postulate**

  For a given orthonormal basis $B = \{|\phi_i\rangle\}$ of a state space H_A for a system A, it is

possible to perform a Von Neumann measurement on system H_A with respect to the basis B that, given a state $|\psi\rangle = \sum_i \alpha_i |\phi_i\rangle$, outputs a label i with probability $|\alpha_i|^2$ and leaves the system in state $|\phi_i\rangle$.

2.1.2 Pure and mixed states, density matrix

describe the state of a quantum system which is in a pure state with a state vector $|\psi\rangle$. To describe a system which is in an ensemble of states with different probabilities we should mention the state of the system using the following set.

$$\{(|\psi_1\rangle, p_1), (|\psi_2\rangle, p_2), ..., (|\psi_n\rangle, p_n)\} \tag{2.4}$$

To avoid this hurdle we may use an operator to define a quantum system of mixed states called density operator[28]. A density operator for a pure state can be written as

$$\rho = |\psi\rangle \langle\psi| \tag{2.5}$$

The mixed state of a quantum system is written as

$$\rho = \sum_i p_i |\psi_i\rangle \langle\psi_i| \tag{2.6}$$

The trace of the density operator (density matrix) equates to 1 which is the sum of the probabilities of the results of the quantum measurement, $tr(\rho) = 1$. A calculation which we encounter very often in our journey to quantum error correction is applying a unitary transformation on the density operator. If we would not use the density operator formalism we would have to denote the use of the transformation U in the following format.

$$\{(U|\psi_1\rangle, p_1), (U|\psi_2\rangle, p_2), ..., (U|\psi_n\rangle, p_n)\} \tag{2.7}$$

Instead using a unitary transformation in this formalism is actually transforming the density operators of the pure states in the ensemble of states which is

$$\sum_i p_i U |\psi_i\rangle \langle\psi_i| U^\dagger = U \left(\sum_i p_i |\psi_i\rangle \langle\psi_i| \right) U^\dagger = U \rho U^\dagger \tag{2.8}$$

To check whether a quantum system is in pure or mixed state we can simply check for the following inequality.

$$tr(\rho^2) \leq 1 \tag{2.9}$$

The equality occurs when the system is in pure state and the otherwise case is reserved for the mixed state.

The probability of being in a state $|\phi\rangle$ after measurement is calculated in this way.

$$\begin{aligned} &\sum_i p_i tr(|\phi\rangle \langle\phi| |\psi_i\rangle \langle\psi_i|) \\ &= tr\left(\sum_i p_i |\phi\rangle \langle\phi| |\psi_i\rangle \langle\psi_i| \right) \\ &= tr\left(|\phi\rangle \langle\phi| \sum_i p_i |\psi_i\rangle \langle\psi_i| \right) \\ &= tr(|\phi\rangle \langle\phi| \rho) \end{aligned} \tag{2.10}$$

So to find the probability of being a certain state all we need to know is the density matrix of the system and this means that if two system have the same density operators they are indistinguishable. Another important usage of density operator arises when we talk about the composite systems. Suppose a composite system consisting of two systems A and B is constructed over $H_A \otimes H_B$ then the density operator of this system is $\rho^A \rho^B$ and thus the trace of the density operator of the system is $tr(\rho^A \rho^B)$. A usual

operation that we face in our calculations is the trace-out operation which results in the density operator of one of the systems building the composite system.

$$\rho^A = tr_B(\rho^A \rho^B) \tag{2.11}$$

2.1.3 General quantum operations

general quantum operation[28], often referred to as superoperator, appears in the framework of quantum error correction and is the operation that attaches two subsystems to form a composite system (one could be the state of the system of question and the other subsystem the the ancilla states helping to carry out some operation in quantum computation), transforms the state of the composite system using some unitary operation and then ignores some subsystem or in other words traces out that subsystem.

$$\rho_{in} \mapsto \rho_{out} = tr_{subsystem}\left(U(\rho_{in} \otimes |\psi_{sub}\rangle \langle\psi_{sub}|)U^\dagger\right) \tag{2.12}$$

It can be shown that a superoperator can be written in the following form

$$\rho_{in} \mapsto \sum_i A_i \rho_{in} A_i^\dagger \tag{2.13}$$

where A_i is called is called Kraus operator and satisfies the following identity.

$$\sum_i A_i A_i^\dagger = I \tag{2.14}$$

2.2 Qubit and the model of quantum computation

2.2.1 Qubit and the Bloch sphere

cording to the postulate of state space, the state of a quantum system can be described

by a unit vector in the Hilbert space.

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \tag{2.15}$$

The probability amplitudes $|\alpha_0|^2$ and $|\alpha_1|^2$ add up to one. In a classical physical system a bit accept values from $\{0,1\}$ and it can be 0 or 1 at a time. This is a deterministic system in which the probability of the system being 0 can be 1 while the probability of the other state of the system is definitely 0. To visualize this system we may consider the following diagram.

Figure 2.1: Visualization of the deterministic bit

We need to see the notion of a classical probabilistic bit as a stepping-stone towards realizing a quantum bit (qubit) and its difference with the classical deterministic bit. A probabilistic bit can take values from $\{0,1\}$ with different probabilities simultaneously and this means that it can be 0 with probability p_0 and at the same time 1 with probability p_1. A line connecting the two points in figure 2.2 can display a probabilistic classical bit.

If we interpret the coefficients of $|0\rangle$ and $|1\rangle$ in the state of a state of a quantum system or a qubit as the probability amplitudes, what is the difference between a probabilistic

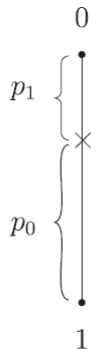

Figure 2.2: Visualization of the probabilistic bit

and a qubit?

The difference is that α_0 and α_1 are complex numbers while p_0 and p_1 are real. A complex number like α can be written as $e^{i\phi}|\alpha|$ where $e^{i\phi}$ is called the phase factor. The most general form of a single qubit is

$$|\psi\rangle = \cos(\frac{\theta}{2})|0\rangle + \sin(\frac{\theta}{2})e^{i\phi}|1\rangle \tag{2.16}$$

Where θ and ϕ take values from $0 < \theta < \pi$ and $0 < \phi < 2\pi$. The geometrical interpretation of a single qubit in its most general form is a complex unit vector in unit sphere with its head on the surface of the sphere pointing outwards. This sphere is called the Bloch sphere[28]. Using the general form of a qubit, the polar points of the Bloch sphere represent $|0\rangle$ and $|1\rangle$. The points on the equatorial plane are also indicated in figure 2.3. The use of the Bloch sphere guides us to a better understanding of the effect of the quantum gates on qubits.

2.2.2 Circuit model of quantum computation

quantum computation, to visualize the effects of the quantum operations on the qubits

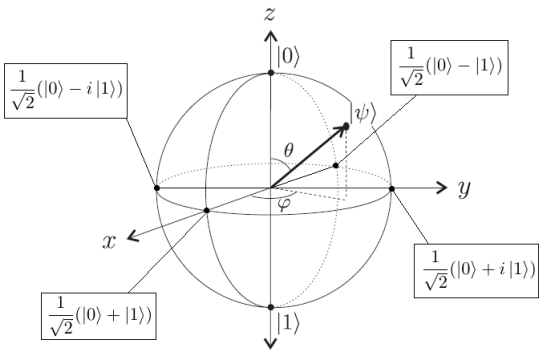

Figure 2.3: The Bloch sphere - visualization of a single qubit

we use the circuit model analogous to that of the classical computation. In this model we have some wires or registers through which we feed in the input qubits and some gates representing the quantum operations on the input qubits to give the outputs. In

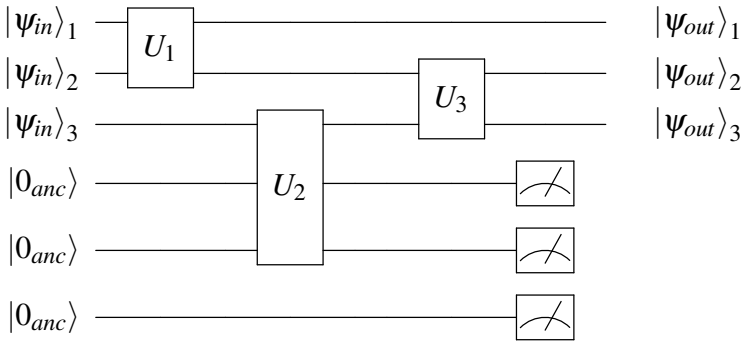

Figure 2.4: Example of a circuit diagram

the example circuit diagram given in the above, the main input qubits $|\psi_{in}\rangle_i$ are different from the ancillary qubits called ancilla. The ancilla, often initialized to zero, help us through different quantum operation such as swapping the values of two registers, recovery operation in a quantum error correction framework, measuring a quantum observable, etc. According to the postulate of measurement in quantum mechanics,

when a measurement is performed the result of the measurement is an eigenvalue of the observable. In fact the output of the measurement on a circuit diagram is the eigenstate associated with that particular eigenvalue while the result of the measurement is a completely classically probabilistic value which is the eigenvalue.

When computing the output of a circuit the following hints come very useful. The input consisting of multiple qubits is, in fact, the tensor product of those qubits. So the input to the above diagram is

$$|\psi_{in}\rangle_1 \otimes |\psi_{in}\rangle_1 \otimes |\psi_{in}\rangle_1 \otimes |0_{anc}\rangle \otimes |0_{anc}\rangle \otimes |0_{anc}\rangle \tag{2.17}$$

Furthermore, to apply the gates on the tensored input qubit we need to tensor the gate along the same instance of the circuit, that is

$$(U_1 \otimes U_1 \otimes I \otimes I \otimes I \otimes I)(I \otimes I \otimes U_2 \otimes U_2 \otimes U_2 \otimes I)(I \otimes U_3 \otimes U_3 \otimes I \otimes I \otimes I) \tag{2.18}$$

where I is the identity operator. Each parenthesis corresponds to an instance of the circuit. An instance is a time slice of the circuit during which at least one qubit passes through a gate and the circuit goes one step further.

2.2.3 Quantum gates

quantum gate[38] that are usually dealt with throughout this thesis are the Pauli matrices.

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$Y = \begin{pmatrix} 0 & -i \\ -i & 0 \end{pmatrix} \qquad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \tag{2.19}$$

These are unitary operators which are in line with the reversible nature of quantum computation. The effect of the X gate is in fact the action of a NOT gate, which means that

$$X(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_1 |0\rangle + \alpha_0 |1\rangle \tag{2.20}$$

Here is the effect of the Z gate.

$$Z(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_0 |0\rangle - \alpha_1 |1\rangle) \tag{2.21}$$

The Dirac notations of the X and Z gates are really helpful.

$$X = |1\rangle \langle 0| + |0\rangle \langle 1| \tag{2.22}$$

$$Z = |0\rangle \langle 0| - |1\rangle \langle 1| \tag{2.23}$$

And since $Y = iXZ$ the effect of Y can be simply found out as below.

$$Y(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_1 |0\rangle - \alpha_0 |1\rangle \tag{2.24}$$

Since two qubits are equivalent up to a global phase we can neglect the factor i in equation 2.24. Using the Bloch sphere we can visualize the effect of these gates. The X gate rotates a state vector on the sphere with π. This means that if the state vector is at the north pole $|0\rangle$ it will be rotated to the opposite pole $|1\rangle$. If the state vector is at

the north pole of the sphere then the Z gate has no effect on it, however it changes $|1\rangle$ to $-|1\rangle$. These gates are all single qubit gates where the inputs and outputs are only one qubit. Among the multiple qubit gates, the CNOT gate is very useful. The CNOT gate is a two-qubit gate in which one register acts a control wire on the action of the gate and the other register acts as the target wire on which the gate is applied. The scheme of the gate is as follows. If the value of the control qubit $|x\rangle$ is equal to one

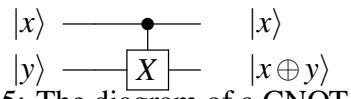

Figure 2.5: The diagram of a CNOT gate

the X gate flips the value of the the target qubit $|y\rangle$ and if it is equal to zero the X gate takes no action. In practice, the value of the target qubit is often set to zero. In classical CNOT gate if the value of the target bit is set to zero then the CNOT gate is used for copying the value of the control bit onto the output on the target wire. However this is not the case when it comes to quantum computation due to the no-cloning theorem as we will see in section 4.1. In general a controlled U gate (c-U) can be constructed using a control wire and a target wire.

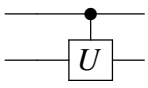

Figure 2.6: Controlled U gate (c-U)

The action of this gate[28] can be shown using the Dirac notation.

$$|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U \tag{2.25}$$

If the value of the control qubit is zero only the identity gate on the target qubit acts and in another case if this value is one the U gate comes into play and the identity gate does nothing.

Another useful single qubit gate which will have widespread use throughout the chapters is the Hadamard gate with the matrix representation below.

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.26}$$

The effects of the Hadamard gate on $|0\rangle$ and $|1\rangle$ are given.

$$H|0\rangle = |+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{2.27}$$

$$H|1\rangle = |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \tag{2.28}$$

The $|+\rangle$ and $|-\rangle$ are called the Hadamard basis and are sometimes used in measurements. $|0\rangle$ and $|1\rangle$ are called the computational basis. Sometimes in quantum circuits we may desire to measure an observable in the Hadamard basis so that we need to apply the Hadamard transformation to the computational basis. The output of this gate on computational basis vectors can be realized using the Bolch sphere. The Hadamard basis vectors are the points on the X axis of the equatorial plane shown in figure 2.3. This means that the Hadamard gate applies a 90 degree rotation when applied on $|0\rangle$ and $|1\rangle$.

A very useful 3-qubit gate is the Toffoli gate. The Toffoli gate consists of two control wires and a target wire. It is in fact a controlled controlled NOT gate (CCNOT) with the two control wires being in state $|1\rangle$ for the X gate to act on the target.

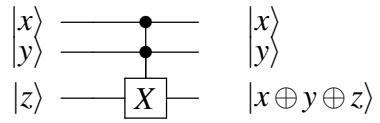

Figure 2.7: Toffoli gate

2.3 Classical noise, quantum operations, and quantum noise

discussion of this section is motivated by its many later uses throughout the thesis[27]. A closed quantum system is the one that has no undesired interaction with its environment. Undesired interactions with the outside world provoke noise in quantum information processing systems. In order to suppress the errors in a noisy channel, first we will need to model the error where quantum operations are powerful tools to do so. As one of the postulates of quantum mechanics, the dynamics of closed quantum systems is described by unitary transformations which is a function of time. Here in analysing the open quantum systems, quantum operations provide us with transformations that describe the state of a quantum system at a final time with respect to the initial state of the system. It also enables us to consider different scenarios including nearly closed systems which have weak interactions with their environment, open system with strong environmental coupling, as well as closed systems which become abruptly open to a measurement operation. Prior to any description of quantum noise through quantum operations it is worth having a short look at the classical noise. Classical information systems made of 0s and 1s may undergo noisy channels where these bits get scrambled.

Given that X and Y are the initial and final states of the bit, respectively, the probability of the state of the bit being in Y is

$$p(Y=y)=\sum_{x} p(Y=y|X=x)\,p(X=x) \tag{2.29}$$

where the conditional probability $p(Y=y|X=x)$ is the transition probability between the initial and final states. The probabilities of the states of the bit being in 0 and 1 are p_0 and p_1 for the initial state, and q_0 and q_1 for the final state. These equations in (1) can be written as

$$\begin{pmatrix} q_0 \\ q_1 \end{pmatrix} = \begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix} \begin{pmatrix} p_0 \\ p_1 \end{pmatrix} \tag{2.30}$$

The probability of a bit-flip error is denoted by p in the above equation and therefore the probability of no error is given $p-1$. Also in this equation the 2×2 matrix is the transition matrix which in general can be written using this equation

$$\vec{q}=E\vec{p} \tag{2.31}$$

Here E, the evolution (transition) matrix holds the following properties: *i. Positivity: All the entries of E must be non-negative otherwise the probability of the final state would be negative.ii. Completeness: All the columns of E must sum to one since we need the sum of the probabilities to be normalized to one.*In multiple-stage processes (like having several gates in a circuit) it is reasonable to assume that that each gate works correctly is independent of other gates working correctly or faulty. This process which is known as a Markov process.To tackle the quantum noise we use density matrix since we have an ensemble of states instead of using a vector with its entries being

probability values. One can write this transformation as

$$\rho' = \mathscr{E}(\rho) \tag{2.32}$$

where $\mathscr{E}$ is a quantum operation. Unitary transformation $U\rho U^\dagger$ and measurement $M\rho M^\dagger$ are examples of quantum operation where the density matrix is transformed using the corresponding operator. Now in describing the changes (initial and final states) of an open quantum system subject to noise we may take two approaches which are equivalent. The first approach is the natural way of thinking about noise in quantum systems and that is assuming the state of the system and the environment (source of noise) as a product state of both. So the joint state is $\rho \otimes \rho_{env}$. This assumption may not be realistic since when the system undergoes a faulty channel and couples with the environment it in fact gets correlated with the environment's state. Nevertheless, in practice we need to avoid these correlations experimentally and therefore this assumption is reasonable. Thus to obtain the state of the system we need to trace out the environment's contribution

$$\mathscr{E}(\rho) = tr_{env}[U(\rho \otimes \rho_{env})U^\dagger] \tag{2.33}$$

The other approach which leaves room for more mathematical convenience is called operator-sum representation. Equation 2.33 can be re-stated as below.

$$\begin{aligned} \mathscr{E}(\rho) &= \sum_k \langle e_k|U[\rho \otimes |E\rangle \langle E|]U^\dagger|e_k\rangle \\ &= \sum_k \langle e_k|U[|E\rangle \rho \langle E|]U^\dagger|e_k\rangle = \sum_k \mathscr{E}_k \rho \mathscr{E}_k^\dagger \end{aligned} \tag{2.34}$$

Where $|E\rangle$ is the initial state of the environment and $|e_k\rangle$ are a basis for the environment space. Here $\mathscr{E}_k = \langle e_k|U|e_k\rangle$ are Kraus operators and satisfy the condition below

$$\sum_k \mathscr{E}_k^\dagger \mathscr{E}_k = I \tag{2.35}$$

with I being the Identity. This condition is a result of the quantum mechanical version of the completeness relation which requires that the trace of the system's final state density matrix must sum to one.

$$\begin{aligned} tr(\mathscr{E}(\rho)) &= tr[\sum_k \mathscr{E}_k \rho \mathscr{E}_k^\dagger] \\ &= tr[\sum_k \mathscr{E}_k^\dagger \mathscr{E}_k \rho] = 1 \Rightarrow \sum_k \mathscr{E}_k^\dagger \mathscr{E}_k = I \end{aligned} \tag{2.36}$$

Here we use the cyclic property of trace and the fact that the trace of ρ must sum to one. The operators $\{\mathscr{E}_k\}$ of the quantum operation are called the operation elements which will meet them later as error operator when modelling the errors in quantum error correction procedures.

Chapter 3

CLASSICAL CODING THEORY

This section provides an introduction to the coding theory and specifically the linear codes[45]. The two main families of error correcting codes are linear and cyclic codes. The study of the structure and construction of linear codes is motivated by their use in establishing the early quantum error correcting codes such as Steane 7-qubit code and quantum Golay code, a 23-qubit quantum code[34]. The constructions of these codes are based on the 7-bit Hamming code and Golay code[46], respectively.

The main aim of creating error correcting codes is to protect the information sent through a transmission channel by adding some extra bits. The physical implementation of a transmission channel as well as the process of sending and receiving a message are prone to error, in such a way that a bit with a bit value of 0 may be changed to 1 or vice versa. It is in this context that we need to implement protocols known as error correcting codes alongside the main procedure of sending and receiving messages.

Before formally introducing the linear codes, it is worth reviewing the simple example of the three-bit repetition code. Studying this code helps us pin down the basic notions of coding and further to extend these concepts to the quantum version of it.

When being transmitted from one point to another point of a computer or even being stored, the information may be affected in an undesired way. We usually call this unwanted influence a channel. The simplest channel we know is called a bit-flip channel and is described in the concept of the binary symmetric channel.

In a binary symmetric channel

- The probability with which the bit value is flipped from 0 to 1 is equal to the probability of flipping a 1 to a 0.
- The probability p of flipping is equal for all bits and is $p < \frac{1}{2}$.
- The bit flips on distinct bits occur independently from each other.

The diagram below illustrates this channel. So the probability of no error is $1 - p$.

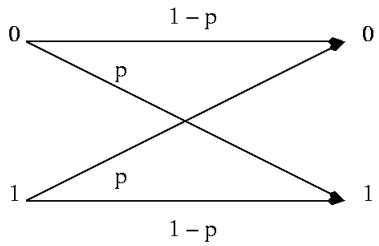

Figure 3.1: The scheme of the binary symmetric channel

It is note-worthy that the channel through which no error occurs is called an identity channel.

3.1 Three-bit Repetition Code

three-bit repetition code[28] is a toy model to demonstrate the procedure of protecting information against one such channel. The first step, as it was pointed out earlier, is to

add a few ancillary bits (redundancy). So in a 3-bit code a single bit (0 or 1) is extended to a string of three bits. As we will see later these ancilla need to be initialized to 0. Therefor, we have

$$0 \rightarrow 0\mathbf{00}$$

$$1 \rightarrow 1\mathbf{00}$$

As the next step, encoding maps the resulted string to some altered bit sequence of the same length according to the code. In the specific case of the repetition code, It is mapped to a sequence of repeated bits, repeating the first bit; consequently called the repetition code.

$$0 \rightarrow 0\mathbf{00} \rightarrow 0\mathbf{00}$$

$$1 \rightarrow 1\mathbf{00} \rightarrow 1\mathbf{11}$$

After encoding the ancilla (the last two bits in boldface) are called parity-check bits. In general encoding extends the vector space of the message to a larger space. The vector space of all triplets of $\{0,1\}$ has 8 elements out of which we choose only 2 of them through encoding to serve as our logical zero $0_L = 000$ and logical one $1_L = 111$. The encoding process is being done through a couple of classical CNOT gates as shown in the circuit diagram where $l \in \{0,1\}$. A code is in fact a set consisting of these encoded bit strings. The elements of a code are called codewords. The next stage of an error correction is the recovery operation and decoding to get back the original message. We need to recover the initial message through recovering the codeword undergone a number of bit-flip errors. The idea of adding redundancy to the sent

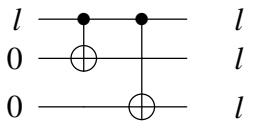

Figure 3.2: Encoding circuit for the three-bit code

message is actually based on the fact that we may decontaminate the encoded message using the information from the ancilla bits with some probability. The entire set of erred codewords are given below provided that the input codeword is 000.

$$000 \rightarrow \{((1-p)^3,000),(p(1-p)^2,100),(p(1-p)^2,010),(p(1-p)^2,001),$$
$$(p^2(1-p),011),(p^2(1-p),101),(p^2(1-p),110),(p^3,111)\} \quad (3.1)$$

For example, if the error has occurred on the first bit with probability p the other two bit remain intact with probability $(1-p)^2$ and if the first and the third bits have been flipped the probability of occurrence of this word will be $p^2(1-p)$. The recovery operation can be done by comparing the value of each bit with the other two and try to guess the flipped bit based on the value of the majority bits. For instance, if the word 101 is received, one may conjecture that the second bit has been flipped and reverting to 1 yields the initial codeword 111 and therefore the message has to be 1. While this could be a possibility another one is that two errors might have happened instead of one and hence the sent codeword might have been 000. As we will see later through the notions of coding theory the 3-bit repetition code is a single-error correcting code in a sense that having two or more errors leads to a set of unrecoverable errors. So the set of correctable errors of this code is $\{000,100,010,001\}$. As we will see the idea of

comparing the bit values coincides with the notion of nearest neighbor decoding.

Although comparing the bit values in order to determine the corrupted bit is possible in classical computation, due to the restrictions imposed by quantum mechanics we will not be able to do so in quantum error correction and since we are interested to extend the results of this study to quantum codes we must devise another scheme. As long as we suppose that up to one bit flip has occurred we can determine the bit flipped through computing the bits parities. The bits parity is computed by adding modulo 2 (XOR) of two bits. In this way when the values of the two bits agree the parity will be zero (even) and it is one (odd) when the two are not like bit values. For example, if the received word is 100 then the parity of the first and the second bits as well as that of the first and the third bits is 1 which implies that the first bit is different from the other two and needs to be reversed so as to get the recovered codeword. The computed parities is called syndrome and plays an important role throughout the study of classical error correcting codes as well as the quantum error correction. In a circuit diagram the syndrome measurement is demonstrated by CNOT gates and the recovery operation is displayed using a three-bit Toffoli gate as shown below. The additional ancilla bits

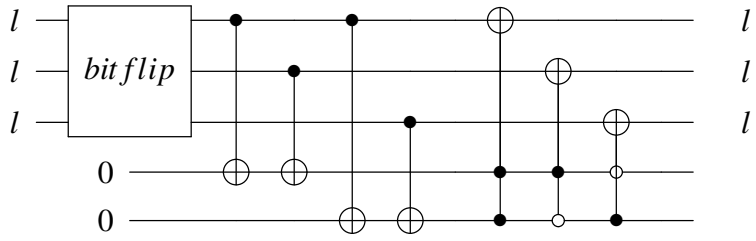

Figure 3.3: The circuit diagram for the three-bit repetition code

initialized to zero are reserved for the syndrome measurement purpose and the hollow controls in the recovery gates show that the corresponding wires are conditioned to be zero in order to get the Toffoli gates act on the target bits. As we can see the idea of computing the syndrome is essentially based on the comparison we made between the bit values.

As we have seen having indistinguishable received codewords we encounter a set of unrecoverable errors. So as to deal with a correctable set of errors we should consider only those errors that yield codewords from disjoint subsets, i.e.

$$\mathscr{E}_i(k) \neq \mathscr{E}_j(l) \qquad ; \qquad l \neq k \tag{3.2}$$

where l and k are the logical zero and one and i and j indicate specific correctable errors. This is the general criterion for error correction and the fulfillment of this condition guarantees the existence of a recovery operation. Since this condition requires the errors to be from disjoint subsets of the entire set of errors one can think of an orthogonality condition for the correctable errors, however, this is not necessarily the case in quantum error correction and the condition that the correctable errors need to be orthogonal is relaxed to some extent(see section 4.2).

The following includes the definitions and propositions needed for studying the linear codes. *Word w* is a string over an alphabet F_q or $GF(q)$, with F_q ($GF(q)$) being a finite field of q elements. For the purpose of error correcting codes $q = 2$ with the elements of the finite field or the alphabet being $\{0,1\}$. *Hamming space* $F_q^n = H(n,q)$

is the vector space of all words of length n over $GF(q)$. A *code* is a subspace of the Hamming space H which contains at least two words and therefore the elements of this subspace are called *codeword*. The 3-bit repetition code is a subspace with two elements of F_2^3 which has eight elements and the codewords are 000 and 111. Given these codewords, the *Hamming distance d* is the minimum number of entires of one of the bit strings which need to be altered in order to get the other one, hence $d = 3$ for the 3-bit repetition code. The Hamming distance has the properties of a distance (metric) function. We have the following proposition.

Proposition 1 *Given that u, v, and w are codewords,*

i $d(v,w) \geq 0$ *with equality iff* $v = w$.

ii $d(v,w) = d(w,v)$.

iii $d(u,v) + d(v,w) \geq d(u,w)$.

The smallest distance between two distinct codewords of a code is called the *minimum distance*. The following proposition helps us understand why the 3-bit repetition code is a single-error correcting code.

Proposition 2 *A code with minimum distance d is capable of correcting up to t errors iff* $d \geq 2t + 1$.

Knowing that $d = 3$ the maximum number of error that can be corrected by the afore-said code is 1, using the above proposition. Another notion that may be useful is the *Hamming weight* of word w, $wt(w)$, which is the number of nonzero bits of the word.

The *minimum weight* of a code is the smallest weight of a nonzero codeword of the code.

Notation We often use [n,k,d] or [n,k] to denote a certain code minimum distance d and encoded words of length n while the length of the word before encoding is k. The 3-bit code is a [3,1,3].

3.2 Linear Codes

this section we deal with the definitions and useful theorems and propositions to study the linear codes so this is a more formal approach to coding theory. As mentioned earlier linear codes set the foundations for us to further study the quantum error correcting codes and in particular the Calderbank-Shor-Steane codes or CSS codes in short.

A linear code C of length n over $GF(2)$ is a subspace of the Hamming space $H(n,2)$. The repetition code is an example of linear codes.

$$C = \{(l,...,l) \quad ; \quad l \in F_2\} \tag{3.3}$$

Given that $C^{\perp}$ is a subspace of $H(n,2)$ it is the dual code of C if any vector v in $C^{\perp}$ is orthogonal to any element of C. So we can write

$$C^{\perp} = \{v \in H(n,2) \quad ; \quad v.w = 0 \quad \forall w \in C\} \tag{3.4}$$

where $v.w$ is the inner product of the two vectors. We have the following theorem for C and its dual.

Theorem 1 *Let C be a linear code of length n over GF*(2)*. Then,*

1 $|C| = 2^{dim(C)}$*, i.e.* $dim(C) = \log_2 |C|$*;*

2 $C^{\perp}$ *is a linear code and* $dim(C) + dim(C^{\perp}) = n$*;*

3 $(C^{\perp})^{\perp} = C$

If $C \subseteq C^{\perp}$ then C is called a self-orthogonal code and if $C = C^{\perp}$, it is called a self-dual code.

The two algorithms to be specified below provide us with a method to find a basis (not unique) for C and $C^{\perp}$ and gives out the generator and the parity-check matrices for C which are the building block of encoding and decoding in error correction.

Algorithm 1

Input A nonempty subset S of $H(n,2)$.

Output A basis for $C = Span(S)$, $Span(S)$ being the span of S and therefore S is the spanning set of C.

Description

1. Form the matrix A with its columns the vectors in S.
2. Take this matrix to the row echelon form (REF).
3. Pick the nonzero rows to form a basis for C. These rows are independent.

Algorithm 2

Input A nonempty subset S of $H(n,2)$.

Output A basis for $C^{\perp}$; $C = Span(S)$.

Description

1. Form the matrix A with its columns the vectors in S.

2. Take this matrix to the reduced row echelon form (RREF).

3. Form the matrix $\mathscr{G}$ to be the $k \times n$ matrix of nonzero rows of the RREF of A.

$$A \to \begin{pmatrix} \mathscr{G} \\ \\ 0 \end{pmatrix} \tag{3.5}$$

   This is called the generator matrix of C.

4. Permute the columns of $\mathscr{G}$ to form

$$\begin{pmatrix} I_k & | & X \end{pmatrix} \tag{3.6}$$

   where I_k is the $k \times k$ identity matrix. This called the standard form of the generator matrix $\mathscr{G}$.

5. Form the following matrix.

$$\begin{pmatrix} X^T & | & I_{n-k} \end{pmatrix} \tag{3.7}$$

   This is called the parity-check matrix $\mathscr{H}$ of C.

6. Pick the rows of $\mathscr{H}$ to form a basis for $C^{\perp}$. These rows are independent.

For a [n,k,d] linear code the generator matrix $\mathscr{G}$ is a $k \times n$ matrix and the parity-check matrix $\mathscr{H}$ is a $(n-k) \times n$ matrix. The second algorithm also provides a basis for C by

computing the generator matrix $\mathscr{G}$. To verify this one can simply check that the rows of $\mathscr{G}$ are in C and for the linear independence of these rows.

Since the rows of $\mathscr{H}$ form a basis for $C^{\perp}$ then the span of its rows must be in $C^{\perp}$. As we know the vectors (codewords) in $C^{\perp}$ are orthogonal to the elements of C or equivalently the span of rows of $\mathscr{G}$. Therefore a $(n-k)\times n$ matrix $\mathscr{H}$ with linearly independent rows that meets the requirement $\mathscr{H}\mathscr{G}^T = 0$ is a parity-check matrix for C. This could be verified by simply writing the matrices $\mathscr{G}$ and $\mathscr{H}$ in their standard forms.

$$\mathscr{H}\mathscr{G}^T = \begin{pmatrix} X^T & | & I_{(n-k)} \end{pmatrix} \begin{pmatrix} I_k \\ -- \\ X^T \end{pmatrix} = X^T + X^T = 0 \tag{3.8}$$

The last addition is addition modulo 2 and hence results in zero when adding two equal digits. In practice what we are given is the generator or the parity-check matrix of a particular code and we can find the other one using algorithm 2 for our encoding and decoding purposes.

3.3 Encoding

basis for C, [n,k] can be written as $\{r_1,...,r_k\}$. C as subspace of the Hamming space $H(n,2)$ possesses 2^k elements of the total elements of $H(n,2)$ which are 2^n. Each of these 2^k codewords of C can be written as a linear combination of the basis. Having $v \in C$,

$$v = u_1 r_1 + ... + u_k r_k \tag{3.9}$$

where $u_i \in GF(2)$ or we can consider the base vectors r_i as the rows of the generator matrix $\mathscr{G}$ and u_is as the coordinates of the vector $u = (u_1, ..., u_k) \in F_2^k$. Then we can write

$$v = u\mathscr{G} = u_1 r_1 + ... + u_k r_k. \tag{3.10}$$

Every word in F_2^k can be encoded using the generator matrix $\mathscr{G}$ whose rows are a basis for the code C. The encoding gives the codewords in C. Given C in its standard form we get an insight into the result of the encoding procedure, namely the codeword.

$$v = u\mathscr{G} = u\begin{pmatrix} I & | & X \end{pmatrix} = \begin{pmatrix} u & | & uX \end{pmatrix} \tag{3.11}$$

So the first k bits of the codeword are in fact the initial word that was encoded and called the *message bits* and the last $n-k$ bits which are the redundancy added to protect the information against noise are called the *check bits*.

3.4 Decoding

to any attempt to decode an encoded message the notion of coset of the code C and its associated definition, coset leader, should be pinned down. A *coset* of C determined by u is defined

$$C + u = \{v + u \quad ; \quad v \in C\} \tag{3.12}$$

where $u \in H(n,2)$. For example the cosets of the following code

$$C = \{0000, 1011, 0101, 1110\}$$

are arranged in an array called (Slepian) standard array. A word of minimum weight in a coset is called a *coset leader*. So the first element of each coset is a coset leader for that coset. The following is a very useful theorem about the properties of cosets of

Codeword→		0000	1011	0101	1110
Cosets↓	0000+C	0000	1011	0101	1110
	0001+C	0001	1010	0100	1111
	0010+C	0010	1001	0111	1100
	1000+C	1000	0011	1101	0110

Table 3.1: Slepian standard array

a code C.

Theorem 2 *Let C be an [n,k,d] linear code over GF*(2)*. Then,*

i every vector of F_2^n *is in some coset of C;*

ii for all $u \in F_2^n$, $|C+u| = |C| = 2^k$;

iii for all $u, v \in F_2^n$, $u \in C+v$ *implies that* $C+u = C+v$;

iv two cosets are either identical or have empty intersection;

v there are 2^{n-k} *different cosets of C;*

vi for all $u, v \in F_2^n$, $u-v \in C$ *iff u and v are in the same coset.*

3.4.1 Nearest Neighbor Decoding

idea upon which the nearest neighbor decoding is based is to find a codeword that is closest to the received word w. In other words the codeword with the minimum distance is the most probable codeword that was initially sent before any error has occurred. Given that the original codeword sent is v, then the distance between v and w

gives the error e.

$$e = w - v \tag{3.13}$$

Since v is in C and using the fact that addition and subtraction are equivalent in binary arithmetic then we can say $e = w - v \in w + C$ where $w + C$ is a coset of C. Therefore $w - e = v \in C$ and by theorem 2 (vi) both w and e are in the same coset. The error string is supposed to be of least weight or in other words very small (like one bit flip, otherwise it leads to the unrecoverable set of errors) so it can be considered as the coset leader in $w + C$. Adding the coset leader to the received word w we most probably reach at the original codeword that was sent.

The following example uses the nearest neighbor decoding with $C = \{0000, 1011, 0101, 1110\}$ and the standard array given before.

Suppose that received word is $w = 1101$. We look up for w in the standard array which is in the last coset $1000 + C$. Then we find the coset leader and add it to the given word to find the codeword which is closest to w.

$$\begin{aligned} \text{coset leader} &= 1000 \\ \text{codeword} &= 1101 + 1000 = 0101 \end{aligned}$$

0101 is the element at the top of the same column as w is located. Noted that the coset $0100 + C$ as well as cosets determined by words of weight greater that 2 are not included in the standard array since they are the same as the present cosets and therefore lead to ambiguity.

3.4.2 Syndrome Decoding

is a time-consuming task for codes of larger length n to search for the received word in a look-up table like the standard array. Syndrome decoding uses the notion of syndrome in coding theory and its properties to find the word in a look-up table called standard decoding array (SDA) and then to decode it.

Given a [n,k,d] linear code with the parity-check matrix H , the syndrome of a word $w \in H(n, 2)$ is given by

$$S(w) = wH^T \tag{3.14}$$

The word syndrome is in F_2^{n-k}. The following theorem states the important properties of syndrome.

Theorem 3 *For $v, w \in H(n, 2)$, we have*

i $S(v+w) = S(v) + S(w)$;

ii $S(v) = 0$ *iff v is in C;*

iii $S(v) = S(w)$ *iff v and w are in the same coset of C.*

iii in theorem 3 enables us to indicate a coset by its syndrome and on this ground we can set up look-up tables which do not require us to search for the received word. Instead, we can compute the syndrome of the given word using the definition of syndrome and look for the coset leader associated with that syndrome. After finding the

proper coset the received corrupted word belongs to we are able to flip back the erred bit in the word by adding it to the coset leader which is in fact the error pattern. Since $S(w) \in F_2^{n-k}$, there are at most 2^{n-k} distinguishable syndromes which is equal to the number of distinct cosets. The example of the [7,4] Hamming code illustrates the procedure of syndrome decoding.

The family of Hamming codes developed by R. W. Hamming. For $r \geq 2$ a Hamming code is a code of length $n = 2^r - 1$ and has a parity-check matrix $\mathscr{H}$ with its columns being all nonzero words in F_2^r. The distance of all Hamming codes is 3 and hence they are all single-error correcting.

The parity-check matrix for the [7,4] Hamming code is given as

$$\mathscr{H} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{3.15}$$

With $r = 3$ this code transforms a word of length 4 to a codeword of length 7, so there are 16 codewords in total. We can arrange the SDA of the code alongside its cosets in a single look-up table.

$$C = \{0000000, 0001111, 0010011, 0011100, 0100101, 0101010, 0110110, 0111001 \\ , 1000110, 1001001, 1010101, 1011010, 1100011, 1101100, 1110000, 1111111\} \tag{3.16}$$

Using Table **??** we may compute the decontaminated codeword. Given that the received word is 1101001 we compute it syndrome which is

$$S(1101001) = w\mathscr{H}^T = (1101001)\begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

So the original codeword must be $1101001 + 0100000 = 1001001$.

$C[7,3]$ is the dual code of C but it is not a Hamming code since $n \neq 2^r - 1$ with $r = 4$, however it is a linear code. $C[7,4]$ and its dual $C^{\perp}[7,3]$ are the materials from which we construct the most well-known member of the family of CSS codes, 7-qubit Steane code.

I. Gachkov has developed a useful Mathematica package "CodingTheory". One can use this package for encoding, decoding, finding the coset leaders, syndromes, etc. and to regenerate the same results.

Coset	Syndrome	Coset leader														
0000000+C	000	0000000	0001111	0010011	0011100	0100101	0101010	0110110	0111001	1000110	1001001	1010101	1011010	1100011	1101100	1110000	1111111
1000000+C	110	1000000	1001111	1010011	1011100	1100101	1101010	1110110	1111001	0000110	0001001	0010101	0011010	0100011	0101100	0110000	0111111
0100000+C	101	0100000	0101111	0110011	0111100	0000101	0001010	0010110	0011001	1100110	1101001	1110101	1111010	1000011	1001100	1010000	1011111
0010000+C	011	0010000	0011111	0000011	0001100	0110101	0111010	0100110	0101001	1010110	1011001	1000101	1001010	1110011	1111100	1100000	1101111
0001000+C	111	0001000	0000111	0011011	0010100	0101101	0100010	0111110	0110001	1001110	1000001	1011101	1010010	1101011	1100100	1111000	1110111
0000100+C	100	0000100	0001011	0010111	0011000	0100001	0101110	0110010	0111101	1000010	1001101	1010001	1011110	1100111	1101000	1110100	1111011
0000010+C	010	0000010	0001101	0010001	0011110	0100111	0101000	0110100	0111011	1000100	1001011	1010111	1011000	1100001	1101110	1110010	1111101
0000001+C	001	0000001	0001110	0010010	0011101	0100100	0101011	0110111	0111000	1000111	1001000	1010100	1011011	1100010	1101101	1110001	1111110

Table 3.2: Syndrome look-up table for the Hamming

Chapter 4

GENERAL SCHEME OF QUANTUM ERROR CORRECTION

Based on our knowledge of the main procedure of classical error correction, in this section I will review the most important aspects of a general scheme of quantum error correction. Prior to exploring these steps, first, I will look into the differences from classical error that quantum errors feature, and secondly, conditions for quantum error correction will be briefly discussed. The final discussion will scrutinize the causes of errors in quantum systems and modeling them.

4.1 Error considerations in quantum error correction

coding theory data repetition or copying is vastly used in order to encode a given bit of information. One of the differences with classical error correcting codes one should be aware of is the restriction which no-cloning theorem brings about. According to the no-cloning (or non-cloning)theorem there is no such transformation (superoperator or general quantum operation for the case of mixed states)that results in the following mapping

$$|\psi\rangle|\phi\rangle \rightarrow |\psi\rangle|\psi\rangle \tag{4.1}$$

where $|\phi\rangle$ is the ancilla. In other words, there is no perfect copying of a quantum state. The theorem and a simple proof[35] are given below:

Theorem 4 (No-cloning) *.There is no quantum operation that takes a state* $|\psi\rangle$ *to* $|\psi\rangle \otimes |\psi\rangle \qquad \forall |\psi\rangle$.

Proof*,The proof is based on the fact that due to the linearity of quantum mechanics, transformations must be linear while the cloning operation is not,*

$$|\psi\rangle \rightarrow |\psi\rangle |\psi\rangle \tag{4.2}$$

$$|\phi\rangle \rightarrow |\psi\rangle |\psi\rangle \tag{4.3}$$

$$(|\psi\rangle + |\phi\rangle) \rightarrow (|\psi\rangle + |\phi\rangle)(|\psi\rangle + |\phi\rangle) \tag{4.4}$$

The last mapping shows the cloning operation defying linearity.

Another important fact in which quantum error correcting codes differ from the classical ones is that the direct measurement of the encoded qubits destroys it since it leads the quantum state to collapsing into one of its eigenstates. In other words, we cannot collect any information about the coefficients α and β in $|\psi\rangle$ through direct measurement. Instead, we can measure the syndrome of the encoded qubits. In addition to bit flip which occurs in classical coding theory, qubits are also subject to phase flip. These errors can be simply modeled using X and Z operators from the Pauli group.

$$X(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_0 |1\rangle + \alpha_1 |0\rangle \tag{4.5}$$

$$Z(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_0 |0\rangle - \alpha_1 |1\rangle \tag{4.6}$$

Furthermore, unlike the classical coding, errors influencing qubits have continuous nature, i.e. bit flip and phase flip do not occur in their complete form, instead qubits experience any arbitrary angular shift[40]. However we can show that the continuous evolution of a single qubit coupled with the environment can be digitized which is

a requirement for error correction. First, let us consider an abstract case where the system and the environment are in $|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle$ and $|E\rangle$, respectively. Suppose the continuous evolution of the product state of the system-environment requires the environment evolves into different states so

$$\begin{aligned}(\alpha_0|0\rangle + \alpha_1|1\rangle)|E\rangle \rightarrow & \alpha_0|0\rangle|E_0^0\rangle + \alpha_0|1\rangle|E_0^1\rangle + \alpha_1|1\rangle|E_1^1\rangle + \alpha_1|0\rangle|E_1^0\rangle \\ &= \frac{1}{2}(\alpha_0|0\rangle + \alpha_1|1\rangle)(|E_0^0\rangle + |E_1^1\rangle) \\ &+ \frac{1}{2}(\alpha_0|0\rangle - \alpha_1|1\rangle)(|E_0^0\rangle - |E_1^1\rangle) \\ &+ \frac{1}{2}(\alpha_0|1\rangle + \alpha_1|0\rangle)(|E_0^1\rangle + |E_1^0\rangle) \\ &+ \frac{1}{2}(\alpha_0|1\rangle - \alpha_1|0\rangle)(|E_0^1\rangle - |E_1^0\rangle). \end{aligned} \tag{4.7}$$

Here, the sub- and super-scripts of the state into which environment evolves indicate the initial state and the final state of the qubit, respectively. Now we can rewrite the states of the qubit in terms of the Pauli matrices as follows.

$$\alpha_0|0\rangle + \alpha_1|1\rangle = I|\psi\rangle \tag{4.8}$$

$$\alpha_0|1\rangle + \alpha_1|0\rangle = X|\psi\rangle \tag{4.9}$$

$$\alpha_0|0\rangle - \alpha_1|1\rangle = Z|\psi\rangle \tag{4.10}$$

$$\alpha_0|1\rangle - \alpha_1|0\rangle = XZ|\psi\rangle \tag{4.11}$$

Therefore the final state can be restated as

$$\begin{aligned}|\psi\rangle|E\rangle \rightarrow & \frac{1}{2}I|\psi\rangle(|E_0^0\rangle+|E_1^1\rangle) \\ & +\frac{1}{2}X|\psi\rangle(|E_0^1\rangle+|E_1^0\rangle) \\ & +\frac{1}{2}Z|\psi\rangle(|E_0^0\rangle-|E_1^1\rangle) \\ & +\frac{1}{2}XZ|\psi\rangle(|E_0^1\rangle-|E_1^0\rangle)\end{aligned} \tag{4.12}$$

So while the errors are continuous in nature we can be sure when correcting errors we are dealing with a finite set of discrete errors. If we are aware of a bit-flip error having occurred in the qubit state, then we can simply deduce that $|E_0^0\rangle = |E_1^1\rangle$ and $|E_0^1\rangle = |E_1^0\rangle$ which yields the final state of the system-environment to be

$$|\psi\rangle|E_0^0\rangle + X|\psi\rangle|E_0^1\rangle \tag{4.13}$$

As an effect of bad control of the gates through which the qubit is passed it can undergo an inaccurate rotation about the x axis of the Bloch sphere which may result in $|E_0^0\rangle = c|E_0^1\rangle$ for some constant c. Hence, the environment state can factored out and the final state can be written as $(cI+X)|\psi\rangle \otimes |E_0^1\rangle$ in which case the error is called coherent. The otherwise case where the environment cannot be pulled out is called incoherent error. The discussion above is rather abstract regarding the continuous nature of quantum noise. The following example can realize it. Assume that the error is coherent and the environmental coupling runs the system into a dephasing (an arbitrary rotation about the z axis of the Bloch sphere) of the qubit, i.e.

$$(\alpha_0|0\rangle + \alpha_1|1\rangle)|E\rangle \rightarrow \alpha_0|0\rangle + \alpha_1 e^{i\theta}|1\rangle \tag{4.14}$$

Note that the arbitrary phase shift also influences $|0\rangle$ through another angle, say, $e^{i\theta}$ but its phase change can be factored out resulting in an immeasurable global phase. We can see what we mean by "*continuous*" when we talk about quantum noise. Slightly changing the angle θ results in close erred states, although this can be treated as discrete error as we discussed earlier.

$$\begin{aligned}\alpha_0|0\rangle + e^{i\theta}\alpha_1|1\rangle &= (\frac{1+e^{i\theta}}{2})(\alpha_0|0\rangle + \alpha_1|1\rangle)|E\rangle + (\frac{1-e^{i\theta}}{2})(\alpha_0|0\rangle - \alpha_1|1\rangle)|E\rangle \\ &= (\frac{1+e^{i\theta}}{2})|\psi\rangle|E\rangle + (\frac{1-e^{i\theta}}{2})Z|\psi\rangle|E\rangle\end{aligned} \tag{4.15}$$

While we can digitize the continuous error, we can see that the probability amplitude is continuous and it is totally different from the case that we have a complete phase-flip error. It is worth noting that for a system of mixed states we can find the error discretezation, however it should be stated in terms of operator-sum representation.

4.2 Quantum error correction criteria

the following we will examine the necessary condition for an error correcting code to exist regardless of the structure of the recovery quantum operation. So we will establish our discussion on errors acting on qubit states. As for our discussions on representing quantum noise in the first chapter, we would use the operator-sum representation in this chapter. So the errors are the operation elements or the Kraus operators introduced in that section.In classical error correction, to correct two different errors we must make sure that all codewords are different after being influenced by the erros. In other words different errors map the codewords to disjoint subspaces of the codespace, which means that we are allowed to consider an orthogonolity condition for these er-

rors, that is zero for different errors and nonzero for the same error mappings. Now the requirement for a quantum error correcting code to exist looks like the same, i.e. different error operators must meet the orthogonality condition as well as the qubit states (or in fact the codewords consisting of qubits):

$$\langle l|\mathscr{E}_i^{\dagger}\mathscr{E}_j|m\rangle = 0 \tag{4.16}$$

Where m and l are the codewords in some codespace (or as we call them encoded qubits). In fact the above equation is a sufficient condition but not a necessary one. To obtain the necessary condition for the existence of a quantum error correcting code we will do an observation through an example. The example is the well-known nine-qubit Shor code which we will explore it later in exclusive sections elaborately. Consider the encoded qubits (codewords)

$$|0\rangle \rightarrow |0_L\rangle = (|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \tag{4.17}$$

$$|1\rangle \rightarrow |1_L\rangle = (|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle) \tag{4.18}$$

where $|0_L\rangle$ and $|1_L\rangle$ are called logical 1 and logical 0, respectively. As for the classical case, here we encode the original qubits to protect them using ancilla qubits (analogous to check bits in classical coding theory). The Shor code can correct a bit flip and a phase flip even if they have occurred on different qubits. The bit flip errors can be corrected through the inner layer [35] of the code which is (as an example)

$$|100\rangle + |101\rangle \rightarrow |000\rangle + |111\rangle \tag{4.19}$$

Here the correction has been done using the majority vote of the qubits where we need to measure the encoded qubit and therefore to disturb it. However, the original correction procedure is done by measuring the syndrome (parity) introduced to some ancilla so we do not observe the encoded qubits.

Regarding the phase flip errors, they are corrected using the outer layer of the code, i.e. the majority of signs of each block of three qubits.

$$(|000\rangle - |111\rangle)(|000\rangle + |111\rangle)(|000\rangle - |111\rangle) \quad \rightarrow (|000\rangle - |111\rangle)(|000\rangle - |111\rangle)$$
$$(|000\rangle - |111\rangle) \tag{4.20}$$

Now consider a phase flip error on the first qubit and the same error on the second qubit. These two different error operations lead to the same erred codeword and therefore a degeneracy is observed in the code. So the Shor code cannot tell us where the phase flip error has occurred unlike the case of the bit flip error where the corrupted qubits were determined by the code. Nevertheless, it can correct the corrupted codeword with a single Z operation on either qubits, the first or the second one. This observation is telling us that equation 4.16 is insufficient in establishing a solid basis for the existence of a recovery operation (one that does not disturb the encoded qubit state as in the approach based on the majority vote). We expect to have distinct codewords after the error has occurred on two different codewords to meet the orthogonality condition

$$\langle 0_L|\mathscr{E}_i^{\dagger}\mathscr{E}_j|1_L\rangle = 0 \tag{4.21}$$

which agrees with equation 4.16. Moreover, if one consider the case where the initial codewords are the same while the error operators are different then the condition below

is not necessarily needed and it may not even be correct for all error operators.

$$\langle 0_L|\mathscr{E}_i^\dagger \mathscr{E}_j|0_L\rangle = \langle 1_L|\mathscr{E}_i^\dagger \mathscr{E}_j|1_L\rangle = 0 \tag{4.22}$$

To continue our observation and to find out more we will compute the value of the braket 4.16 for the logical zero given above in the remaining of this section. In fact, values of 4.16 for different phase flip errors that act on different qubits of logical zero form a matrix of the form below. The error operators are just the tensor product of Z and I. Here are examples of error operators.

$$\mathscr{E}_1 = Z_1 = Z \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \tag{4.23}$$

$$\mathscr{E}_2 = Z_2 = I \otimes Z \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I \tag{4.24}$$

$$\begin{pmatrix}
8 & 8 & 8 & & & & & & \\
8 & 8 & 8 & & 0 & & & 0 & \\
8 & 8 & 8 & & & & & & \\
 & & & 8 & 8 & 8 & & & \\
 & 0 & & 8 & 8 & 8 & & 0 & \\
 & & & 8 & 8 & 8 & & & \\
 & & & & & & 8 & 8 & 8 \\
 & 0 & & & 0 & & 8 & 8 & 8 \\
 & & & & & & 8 & 8 & 8
\end{pmatrix}$$

Obviously, we can see that not all nondiagonal elements are zero which confirm our expectation with several other cases of different error operations and nonzero valus of

4.16. So one can write 4.16 more properly as

$$\langle 0_L|\mathscr{E}_i^\dagger \mathscr{E}_j|0_L\rangle = \langle 1_L|\mathscr{E}_i^\dagger \mathscr{E}_j|1_L\rangle \tag{4.25}$$

This equation allows the mapping of two different errors into the same subspace of the codespace which is not allowable in classical error correction since it makes indistinguishable states. One might ask whether these errors are correctable instead of being distinct. In other words, we should ask if there is any invertible operation that can recover the codeword and correct the error. So in quantum error correction orthogonality is not the condition that has to be met in order to have an error correcting code correct the errors but instead we can write

$$\langle l|\mathscr{E}_i^\dagger \mathscr{E}_j|m\rangle = C_{ij}\delta_{lm}. \tag{4.26}$$

To see a derivation of the sufficient and necessary condition (Knill-Laflamme criteria) you can see [29]. A quantum error correcting code corrects the errors of an error family $\mathscr{E}$ if and only if the condition in 4.26 is satisfied. Which means that if we have some family of errors fulfilling this requirement we are able to find a recovery operation $\mathscr{R}$ that can correct those errors. Equation 4.26 also implies that the relative coefficients of different qubits undergone an error do not change which in fact means that we do not face any more complication.For further discussion we can consider all the errors of an error family $\mathscr{E}$ that fulfill the orthogonality condition. To find such error operators we should notice that C_{ij} is a Hermitian matrix and therefore diagonalizable. Diagonalizing it yields

$$\begin{pmatrix}
0 & 0 & 0 & & & & & & \\
0 & 0 & 0 & & 0 & & & 0 & \\
0 & 0 & 0 & & & & & & \\
 & & & 0 & 0 & 0 & & & \\
 & 0 & & 0 & 0 & 0 & & 0 & \\
 & & & 0 & 0 & 0 & & & \\
 & & & & & & 24 & 0 & 0 \\
 & 0 & & & 0 & & 0 & 24 & 0 \\
 & & & & & & 0 & 0 & 24
\end{pmatrix}$$

Before considering writing the orthogonality condition, we should note that if a QECC corrects two errors $\mathscr{E}_i$ and $\mathscr{E}_j$ it also corrects any linear combination of the two errors [35], i.e. maps the codewords to the same codespace. So redifing the quantum error correction condition in terms of orthogonality raises the rescaling of the errors in such a way that a condition like 4.26 becomes[3,8,9]

$$\langle l|F_i^\dagger F_j|m\rangle = \delta_{ij}\delta_{lm} \tag{4.27}$$

in which case, Fs should be written as a linear combination of the basis errors Z, For instance, $F_1 = Z_1 + Z_2$ and $F_1 = Z_1 - Z_2$. Choosing the new basis errors results in nondiagonal elements of zero value but on the other hand we can see that not all diagonal elements are nonzero. The selection of errors of the first type F_1 gives rise to the nonzero values in the diagonalized matrix while the second type results in the

zero value diagonal elements. In fact, the errors of the first type create the orthogonal states that satisfy 4.27 and can be inverted using a single $Z^{\dagger}$ however the second type of errors annihilate any codewords since they act like the zero operator. These operators which annihilate the codewords cannot be inverted (corrected) and basically we do not need to since the probability of them occurring is zero.

$$\langle l|F_2^{\dagger}F_2|m\rangle = 0 \tag{4.28}$$

The codes with C_{ij} that does not have the maximum ranks and therefore is singular (since its determinant is zero) are called degenerate codes. So the nine-qubit Shor code is a degenerate code.[35],[41]

4.3 General procedure of quantum error correction

4.3.1 Error models

the first step in an error correction procedure[28], modeling errors helps us to treat them in an error correction procedure as we have seen in the case of three-bit repetition code. As we discussed earlier coupling with the environment is known as the source of noise in quantum information processing systems which was explained in 2.3. The operator-sum representation of this coupling helps us to formally describe a quantum channel which is in fact the effects of the quantum errors (bit flip, phase flip or both)on the state of the system represented by the density matrix of the system ρ. We use the density matrix representation since we need to deal with an ensemble of states with different probabilities.Suppose that the system is initially in the mixed state $|\psi\rangle$ and the eviron-ment is initially in the state $|E\rangle$ which may or may not be a pure state. Then the joint state of the system-environment coupling is given as $\rho_{sys-env} = |\psi\rangle\langle\psi| \otimes |E\rangle\langle E|$ or

equivalently $|\psi\rangle|E\rangle\langle E|\langle\psi|$. Therefore the of the joint state of the system-environment coupling, using the operator-sum representation, is

$$\rho_{f_{sys-env}} = U_e|\psi\rangle|E\rangle\langle E|\langle\psi|U_e^\dagger \tag{4.29}$$

where U_e is the error operator. If we want to find the final state of the error we can just trace out the environment and here is what we get.

$$\rho_f = tr_{env}(U_e|\psi\rangle|E\rangle\langle E|\langle\psi|U_e^\dagger) = \sum_k \mathscr{E}_i \rho \mathscr{E}_i^\dagger \tag{4.30}$$

with $\mathscr{E}_i$ being the Kraus operators in the operator-sum representation. In fact these operators contain the information about the error model. For instance, suppose that a qubit $|\psi\rangle\langle\psi|$ has undergone a bit-flip channel represented by the Pauli matrix X, then the probability with which the error has occurred is p while the probability of no error is $1-p$ and the final state of the system is

$$\begin{aligned}\rho \to \rho_f = \sum_{i=0}^{1} \mathscr{E}_i|\psi\rangle\langle\psi|\mathscr{E}_i^\dagger &= \mathscr{E}_0|\psi\rangle\langle\psi|\mathscr{E}_0^\dagger + \mathscr{E}_1|\psi\rangle\langle\psi|\mathscr{E}_1^\dagger \\ &= (1-p)|\psi\rangle\langle\psi| + pX|\psi\rangle\langle\psi|X\end{aligned} \tag{4.31}$$

So the operation elements of this general quantum operation or superoperator are $\sqrt{1-p}\mathbb{I}$ and $\sqrt{p}X$.

4.3.2 Encoding procedure

second step in an error correcting scheme is protecting information so that when transmitted and some error occurs we can recover the original qubit. Doing so, we need to add some ancillary qubits (or ancilla) to the original qubit and then encode the enlarged qubit by a unitary transformation. The coding space is a subspace of the Hilbert space

and the encoded qubit is a two-dimensional subspace. So the enlarged qubit has the form $|\psi\rangle|00...0\rangle$ and the encoded qubit, denoted by $|\psi_{enc}\rangle$ is

$$|\psi_{enc}\rangle = U_{enc}|\psi\rangle|00...0\rangle \tag{4.32}$$

where U_{enc} is denotes the encoding unitary transformation. The next stage is to find some other transformation correcting the error that has occurred on the encoded qubit whose existence depends on the quantum error correction condition discussed before.Due to the no-cloning theorem, applying the idea of a repetition code like that of classical error correction is not allowed, However we would like to extend a similar approach to the quantum regime. Since the advent of the quantum error correcting codes several different codes with different number of ancilla qubits have been put forward some of which are three-qubit code, nine-qubit Shor code, and the seven-qubit Steane code. As a concluding example to this section suppose that a qubit is in a pure state $\alpha_0|0\rangle + \alpha 1|1\rangle$ and we are supposed to encode it using a three-qubit code where it is encoded as below

$$(\alpha_0|0\rangle + \alpha_1|1\rangle)|00\rangle \xrightarrow{\text{encoding}} \alpha_0|000\rangle + \alpha_1|111\rangle \tag{4.33}$$

Note that this is not a simple copying, if it were we would have $(\alpha_0|0\rangle + \alpha_1|1\rangle)^{\otimes 3}$ and therefore it is not the violation of no-cloning theorem.

4.3.3 Recovery operation

ing a promising operation that can invert the effect of the error operators on an encoded qubit (codeword) is the target of the final step in a quantum error correction scheme. As with the other quantum operations that have been considered so far, a recovery

operation is also a superoperator that transforms the the encoded qubits after being influenced by an error and is the sum of recovery operators. Denoted by $\mathscr{R}$, we expect that after decoding the qubit and applying $\mathscr{R}$ we obtain the initial encoded qubit. However the final step is accomplished once we trace out the ancilla while it contains the noise sifted out by the recovery operation. Equation below shows the steps of a complete error correction procedure where $U_{enc}^{\dagger}$ (applied after the error operator acts) represents the decoding step[1].

$$\rho_{|\psi\rangle\langle\psi|} = tr_{anc}(\sum_{j}\sum_{i}\mathscr{R}_j U_{enc}^{\dagger}\mathscr{E}_i U_{enc}\,|\psi\rangle\,|00...0\rangle\,\langle 00...0|\,\langle\psi|\,U_{enc}^{\dagger}\mathscr{E}_i^{\dagger}U_{enc}\mathscr{R}_j^{\dagger}) \quad (4.34)$$

where $\rho_{|\psi\rangle\langle\psi|} = |\psi\rangle\langle\psi|$. Although this equation includes a decoding operation, we are not really interested in such an intermediate step in our error correcting code since this operation is prone to error itself. So omitting this operation from the stages of a QECC gives way to a more robust technique and finally paves the road for a fault-tolerant error correction which will be discussed in chapter 8. To clarify the recovery operation we can conclude with the following explanation. Suppose that an arbitrary code has encoded a qubit to $|\psi_{enc}\rangle$ and some errors have occurred. What we expect from our recovery operation is to refine the final corrupted state and gives out the original initial state of the qubit or in other words

$$\sum_{i}\mathscr{R}U_{enc}^{\dagger}\mathscr{E}_i\,|\psi_{enc}\rangle\,\langle\psi_{enc}|\mathscr{E}_i^{\dagger}U_{enc}\mathscr{R}^{\dagger} = |\psi\rangle\,\langle\psi|\otimes|\phi\rangle\,\langle\phi|. \quad (4.35)$$

Here, $|\phi\rangle\langle\phi|$ is denotes the ancilla which carry the noise pushed out by the recovery operation $\mathscr{R}$, tracing out the ancilla we obtain the initial state of the qubit. To sum up, correcting a given set of correctable errors requires us to search for an encoding

operator and an appropriate recovery operator.

Chapter 5

THREE-QUBIT CODE AND NINE QUBIT SHOR CODE

Before exploring the codes which are able to correct both bit-flip and phase-flip errors, we start with the three-qubit code which can be considered as a toy model in studying the quantum error correcting codes. Here, we develop the code to separately correct these two types of errors.

5.1 Bit-flip channel

breaking the overall procedure of QEC we thought of three different steps through which we could correct a single error. So we should see these steps in every code. In the case of the bit-flip error we have already seen the model using the operator-sum representation.

$$(1-p)\,|\psi\rangle\langle\psi| + pX\,|\psi\rangle\langle\psi|X \tag{5.1}$$

In the three-qubit code we encode a single qubit using two ancilla bits initialized to $|0\rangle$ and using a sequence of CNOT gates the information of the original qubit is copied to the ancilla. However, the copying process does not violate the effect of the no-cloning theorem as discussed before (sec. section 4.3.2). The logical qubits or the codewords of this code are $|0_L\rangle = |000\rangle$ and $|1_L\rangle = |111\rangle$.

$$\alpha_0\,|0\rangle + \alpha_1\,|1\rangle \to \alpha_0\,|000\rangle + \alpha_1\,|100\rangle \to \alpha_0\,|000\rangle + \alpha_1\,|111\rangle = \alpha_0\,|0_L\rangle + \alpha_1\,|1_L\rangle \tag{5.2}$$

The reason why this code can correct only one error and fails for more than one error of course lies in its correctable set of errors which are distinguishable. However we can explain the number of errors corrected by the three-qubit code using the code distance (Hamming distance). The encoding operation actually limits the number of possible codewords from 8 codewords forming the codespace to 2 ($|0_L\rangle$ and $|1_L\rangle$). The number of bit flips needed to reach one of these logical qubits from the other is 3 and therefore the binary distance between them is $d = 3$. Using $t = \frac{(d-1)}{2}$ from section 3.1, the maximal number of errors, t, is obtained $t = 1$.

As with the classical three-bit repetition code, the syndrome (parity) measurement is done using two additional ancilla introduced to the circuit. Then, the results of the parity measurement will be obtained by adding the first and the second qubits as well as the first and the third qubits where the addition operation is done modulo two. In this way the location of the error is spotted and the error will be corrected by inverting the flipped qubit. The following table shows the errors, the error operators and the results of the parity check when the input qubit is $\alpha_0 |000\rangle + \alpha_1 |111\rangle$.

Flipped qubit	Error operator	Qubit state	Syndrome 1	Syndrome 2
no error	$I \otimes I \otimes I$	$\alpha_0 \|000\rangle + \alpha_1 \|111\rangle$	0	0
first qubit	$X \otimes I \otimes I$	$\alpha_0 \|100\rangle + \alpha_1 \|011\rangle$	1	1
second qubit	$I \otimes X \otimes I$	$\alpha_0 \|010\rangle + \alpha_1 \|101\rangle$	1	0
third qubit	$I \otimes I \otimes X$	$\alpha_0 \|001\rangle + \alpha_1 \|110\rangle$	0	1

Table 5.1: The syndrome look-up table of the three-qubit repetition code

In this code the distinguishable syndromes leads us to the qubit where the error has occurred. As before the syndrome measurement is carried out using CNOT gates. However, the remaining task is to use the result of this measurement. Looking at the table, we can conclude that when both parities are equal to 1 the first qubit needs to be flipped. This implies that applying a Toffoli gate (controlled-controlled NOT gate) can correct the error on the first qubit. For the other two qubits one of the control wires must be conditioned to zero, i.e. when the value on the wires disagree the second or the third qubit need to be flipped, according to the table. The following are the circuit diagrams for encoding process in the three-qubit code and the recovery circuit correcting a single bit flip using Toffoli gate. In figure 5.3, the recovery operation is

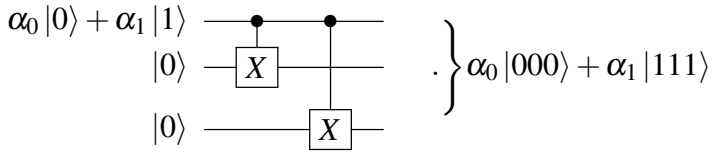

Figure 5.1: The circuit encoding three qubits for the repetition code

shown using three Toffoli gates taking the results of the syndrome measurements as the value of the control qubits. The double lines connecting the measurement operators and the CCNOT gates indicate that controlling over the NOT gates (X) is done classically instead of being done quantumly. This means that we can use the classical output of the measurements, or in fact the classical bits, to control the NOT gates. But why are we able to place these measurement operators right after the syndrome measurement circuit? The reason is based on the fact that the measurement operators, which are

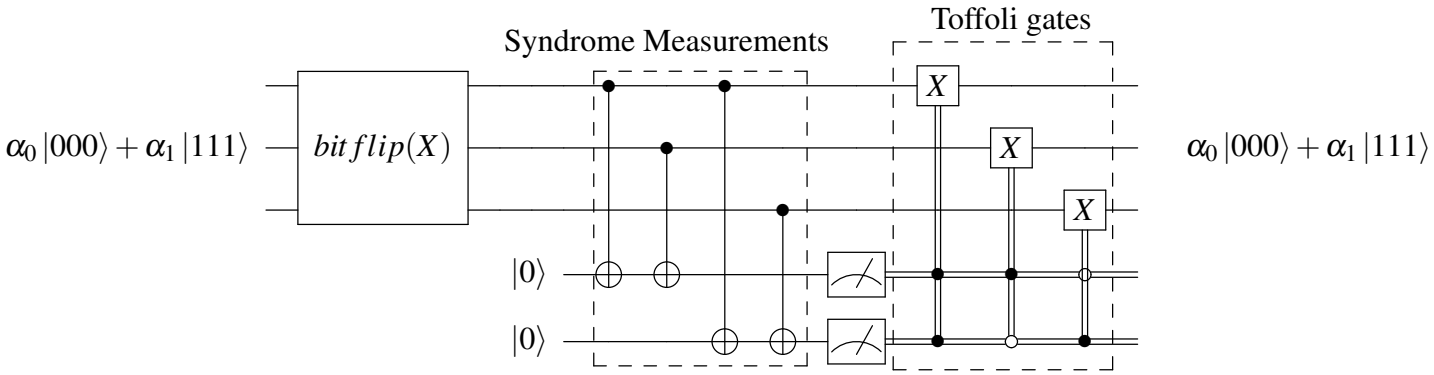

Figure 5.2: The circuit diagram of the three-qubit code

basically placed at the end of the circuit, commute with the controlled gates in the recovery circuit. To see this we can simply show that the result of the commutator of a projective measurement $M = \sum_{i=\{0,1\}} m_i |i\rangle \langle i|$ and a controlled gate such as U_c is zero. In advance of computing this commutator we should note that we can write a controlled gate U_c consisting of two 1-qubit gates, A and B in the following way

$$U_c = |0\rangle \langle 0| \otimes A + |1\rangle \langle 1| \otimes B \tag{5.3}$$

This equation shows the mechanism through which a controlled gate acts. When the control qubit is condition to be zero it allows the action of the 1-qubit gate A and in the otherwise case the gate gives way to the action of B [27]. The following diagram clarifies this fact. $|c\rangle$ and $|t\rangle$ are used to show the control and the target qubits, respectively.

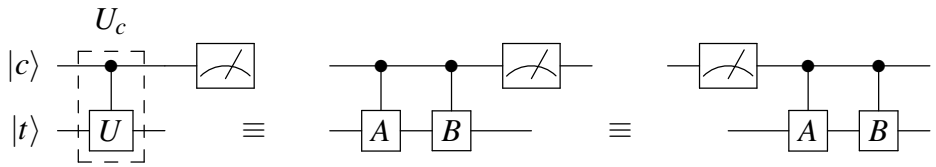

Figure 5.3: Commutation of measurement and controlled gate

Now we need to show that $[U_c, M] = 0$. Using equation 5.3 we have

$$
\begin{aligned}
[U_c, M]\,|ct\rangle &= (U_c M - M U_c)\,|ct\rangle \\
&= \Big[\big[\,|0\rangle\langle 0| \otimes A + |1\rangle\langle 1| \otimes B\big] M - M\big[\,|0\rangle\langle 0| \otimes A + |1\rangle\langle 1| \otimes B\big]\Big]\,|ct\rangle \\
&= |0\rangle\langle 0|M\,|c\rangle \otimes A\,|t\rangle + |1\rangle\langle 1|M\,|c\rangle \otimes B\,|t\rangle - \big[M\,|0\rangle\langle 0| \otimes A + M\,|1\rangle\langle 1| \otimes B\big]\,|ct\rangle \\
&= m_0\,|0\rangle\langle 0| \otimes A\,|ct\rangle + m_1\,|1\rangle\langle 1| \otimes B\,|ct\rangle - m_0\,|0\rangle\langle 0| \otimes A\,|ct\rangle - m_1\,|1\rangle\langle 1| \otimes B\,|ct\rangle \\
&= 0 \qquad (5.4)
\end{aligned}
$$

So we can switch the places of the measurement operation and the recovery operation consisting of Toffoli gates provided that the control qubits are measured. This fact is also of fundamental use in stabilizer formalism of QEC which we will explore later.

5.2 Phase-flip channel

tackle a single phase-flip error the approach is similar to that of the bit-flip error underlining the difference lies in the basis in which the operations are carried out. Specifically, the basis we pick is the Hadamard basis instead of the computational basis. In other words, by changing the basis from computational to Hadamrad the phase-flip error behaves in the same way as the bit-flip error does. It is easy to show this equivalence. Using the X, Z, and H gates given in section 2.2.3 and that the Hadamard gate

is unitary we have

$$\begin{aligned} HZH &= \frac{1}{2}(X+Z)Z(X+Z) \\ &= \frac{1}{2}(XZX + XZ^2 + Z^2X + Z^3) \\ &= \frac{1}{2}(-Z + 2X + Z) = X \end{aligned} \tag{5.5}$$

where $XZX = |1\rangle\langle 1| - |0\rangle\langle 0| = -Z$ and $Z^2 = I$. This means that to approach a phase-flip error we may Hadamard the ancilla-added input qubit and then subject it to a phase flip and Hadamard it again so that we can treat as a bit flip. As a matter of fact, the first Hadamard transformation is the final part of the encoding circuit and the second one is part of the syndrome measurement. What we need to construct the circuit diagram of this code is a few slight changes to that of the 3-qubit code correcting a single bit-flip error. The effect of the encoding operation (including the first Hadamard

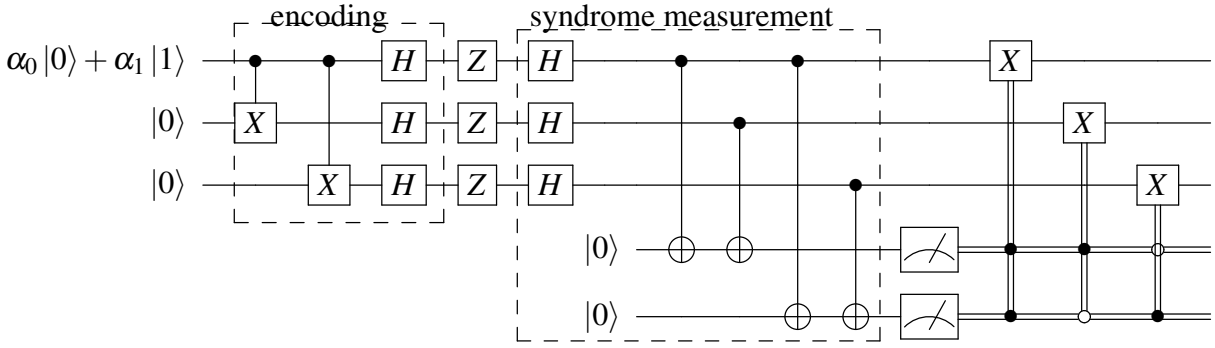

Figure 5.4: Three-qubit code correcting a single phase-flip error

transformation) can be realized in the following way.

$$\alpha_0|000\rangle + \alpha_1|100\rangle \to \alpha_0|000\rangle + \alpha_1|111\rangle \to \alpha_0|+++\rangle + \alpha_1|---\rangle \tag{5.6}$$

where $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, the Hadamard bases. Thus, we see that using Hadamard gate to transform the computational basis we can turn an error with no classical analogous to one with classical counterpart to solve it using the known techniques in classical coding.

5.3 On fidelity of the three-qubit code

is the probability of success of the three-qubit code? The aim of this section is to show how close is the final state of the qubit to its initial state after a simple error correction scheme has been applied. In other words, we need to analyze it using the notion of fidelity. However this will be a simplistic approach it will be still extendable to other codes. What we need to do is simply comparing the fidelities of an unencoded state and an encoded state. In a more realistic view we should relax the assumption of occurrence of a single error, for example, the first and the third qubit both may be subject to error. However we keep the assumption saying that full bit-flip or phase-flip errors happen while we know that the errors in a quantum system have a continuous nature (see section 4.1). We can choose either bit-flip or phase-flip errors and the error operator will be

$$\begin{aligned}
\mathscr{E} &= \big((1-p)I + pX\big)^{\otimes 3} \\
&= (1-p)^3 I\otimes I\otimes I + p(1-p)^2 X\otimes I\otimes I \\
&+ p(1-p)^2 I\otimes X\otimes I + p(1-p)^2 I\otimes I\otimes X \\
&+ p^2(1-p)X\otimes X\otimes I + p^2(1-p)X\otimes I\otimes X \\
&+ p^2(1-p)I\otimes X\otimes X + p^3 X\otimes X\otimes X
\end{aligned} \tag{5.7}$$

Where p is the probability of occurrence of a single error. After the syndrome measurement block is applied in the circuit, the full quantum state of the system is obtained by coupling the syndrome measurement ancilla.

$$\begin{aligned}
\mathscr{E}|\psi_L\rangle = (1-p)^3 I\otimes I\otimes I|\psi_L\rangle|00\rangle \\
+ p(1-p)^2 X\otimes I\otimes I|\psi_L\rangle|11\rangle \\
+ p(1-p)^2 I\otimes X\otimes I|\psi_L\rangle|10\rangle \\
+ p(1-p)^2 I\otimes I\otimes X|\psi_L\rangle|01\rangle \\
+ p^2(1-p) X\otimes X\otimes I|\psi_L\rangle|01\rangle \\
+ p^2(1-p) X\otimes I\otimes X|\psi_L\rangle|10\rangle \\
+ p^2(1-p) I\otimes X\otimes X|\psi_L\rangle|11\rangle \\
+ p^3 X\otimes X\otimes X|\psi_L\rangle|00\rangle
\end{aligned} \tag{5.8}$$

where $|\psi_L\rangle = \alpha_0|0_L\rangle + \alpha_1|1_L\rangle$ is the logical qubit state. When applying a cycle of error correction the state of the system collapses into one of the superposition states depending on the syndrome measurement represented by the ancilla. The following table shows such states. So the final state after a round of quantum error correction is a superposition of the treated clean qubit state and an erred state. In fact, no error correcting code is able to thoroughly correct a qubit. This is essentially because the uncorrectable set of errors, for which the code fails to correct, occur alongside the correctable set (erred states with an error on a single qubit). Given that the fidelity[28] is

$$F = \sqrt{\langle\psi|\rho|\psi\rangle} \tag{5.9}$$

Ancilla	Superposition state
00	$(1-p)^3 \left\|\psi_L\right\rangle + p^3 X \otimes X \otimes X \left\|\psi_L\right\rangle$
01	$p(1-p)^2 \left\|\psi_L\right\rangle + p^2(1-p) X \otimes X \otimes X \left\|\psi_L\right\rangle$
10	$p(1-p)^2 \left\|\psi_L\right\rangle + p^2(1-p) X \otimes X \otimes X \left\|\psi_L\right\rangle$
11	$p(1-p)^2 \left\|\psi_L\right\rangle + p^2(1-p) X \otimes X \otimes X \left\|\psi_L\right\rangle$

Table 5.2: Ancilla states and the superposition states

It is the minimized value of $\sqrt{\langle\psi|\rho|\psi\rangle}$ over all encoded qubits to have a high fidelity. In general it is a measure to show how two density matrices are close. So we need to establish the density matrix for the final state of the encoded qubit coupled with the syndrome ancilla and then trace over the ancilla to have a refined encoded qubit.

$$\left[(1-p)^3 + 3p(1-p)^2\right] \left|\psi_L\right\rangle \left\langle\psi_L\right| + \left[p^3 + 3p^2(1-p)\right] \mathscr{E}' \left|\psi_L\right\rangle \left\langle\psi_L\right| \mathscr{E}' \tag{5.10}$$

where $\left|\psi_L\right\rangle \left\langle\psi_L\right|$ is the density matrix for the logical encoded qubit and $\mathscr{E}'$ is a short notation for $X \otimes X \otimes X$, noticing that $\mathscr{E}' = \mathscr{E}'^{\dagger}$. Getting back to fidelity, the fidelity of an unencoded qubit with the density matrix $(1-p)\left|\psi\right\rangle \left\langle\psi\right| + pX \left|\psi\right\rangle \left\langle\psi\right| X$ is

$$F_{unencoded} = \left[\left\langle\psi\right| \left((1-p)\left|\psi\right\rangle \left\langle\psi\right| + pX \left|\psi\right\rangle \left\langle\psi\right| X\right) \left|\psi\right\rangle\right]^{\frac{1}{2}} = \left[(1-p) + p\left\langle\psi\right| X \left|\psi\right\rangle^2\right]^{\frac{1}{2}} \tag{5.11}$$

To have this minimized and have a high fidelity we may consider the case where $\left\langle\psi\right| X \left|\psi\right\rangle = 0$ and therefore,

$$F_{unencoded} = \sqrt{1-p}. \tag{5.12}$$

On the other hand, the fidelity of the encoded qubit is found as below.

$$F_{encoded} = \left[\langle \psi_L | \left((1-p)^3 + 3p(1-p)^2 |\psi_L\rangle \langle \psi_L| + \left(p^3 + 3p^2(1-p)\right) \mathscr{E}' |\psi_L\rangle \langle \psi_L| \mathscr{E}' \right) |\psi_L\rangle \right]^{\frac{1}{2}}$$

$$= \left[(1-p)^3 + 3p(1-p)^2 + \left(p^3 + 3p^2(1-p)\right) \langle \psi_L | \mathscr{E}' |\psi_L\rangle^2 \right]^{\frac{1}{2}} \tag{5.13}$$

Minimizing this yields

$$F_{encoded} = \sqrt{(1-p)^3 + 3p(1-p)^2} \tag{5.14}$$

Comparing the two fidelities for $p < \frac{1}{2}$ shows a higher value for the encoded qubit and this means that the three-qubit code is able to suppress the error with this probability. As was discussed before nor the three-qubit code neither other codes are able to output a full clean corrected state. The following graph compares the fidelity of no error correction and applying a single error correcting three-qubit repetition code.

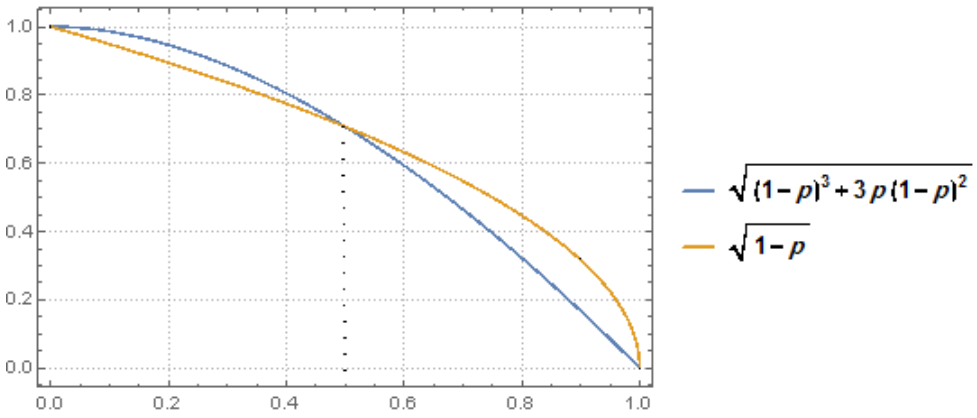

Figure 5.5: Fidelity vs. the probability

5.4 Nine-qubit Shor code

Shor nine-qubit code[26] is the first of a list of full quantum error correcting codes correcting both bit-flip and phase-flip errors. It is vastly based on the three-qubit code we have just reviewed.

To encode a qubit $|\psi\rangle$ first we start with the approach we took when correcting a phase-flip error, meaning that

$$|0\rangle \rightarrow |+++\rangle \quad , \quad |1\rangle \rightarrow |---\rangle \tag{5.15}$$

where $|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$. Then each $|+\rangle$ and $|-\rangle$ should be re-encoded in exactly the same way we did for the bit-flip three-qubit repetition code.

$$\begin{aligned} |+\rangle &\rightarrow \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \\ |-\rangle &\rightarrow \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle) \end{aligned} \tag{5.16}$$

So the logical qubits of the encoding operation will be

$$\begin{aligned} |0_L\rangle &= \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \\ |1_L\rangle &= \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle) \end{aligned} \tag{5.17}$$

The circuit diagram of this code helps us understand the mechanism of its error correction.

As it can be observed from the above diagram the three inner blocks (located between the two H gates)correct a single bit-flip error on any of the nine qubits. This correction takes place inside each triplet of $(|000\rangle \pm |111\rangle)$. The task of the outer block is to

correct a single phase-flip error. In other words, the outer block of the circuit checks for any changes in the signs in $\alpha_0 |+++\rangle + \alpha_1 |---\rangle$ and then treats it as if it were a bit-flip error since Z acts like a bit flip in the Hadamard basis. What this code is not able to precisely detect is where the phase-flip error is located within each inner block. This should not be considered as a drawback of the Shor code. The reason why this code works despite the lack of information about the exact qubit where the phase flip occurs is based on the fact that the Z error affects the relative phase of $|000\rangle$ and $|111\rangle$ by a factor of $e^{i\pi}$. This phase shift leads to a change of sign of $|111\rangle$ while leaves $|000\rangle$ unaffected in their superposition $|000\rangle \pm |111\rangle$ and hence, inverting this can be done by changing any other qubit within the block. Here is the example.

$$|000\rangle + |111\rangle \xrightarrow{\text{phase flip}} |000\rangle - |111\rangle \xrightarrow{\text{recovery}} |000\rangle + |111\rangle \tag{5.18}$$

This code is called a degenerate code since there are more than one error that map the codewords to the same erred qubit. The shor code is also able to correct a single Y error since it is only a product of X and Z (ignoring the global phase i). Even though it can correct up to three bit-flip errors each occurring in a different block but it is not a multiple error correcting code since if multiple errors happen within a certain block it fails to correct them.

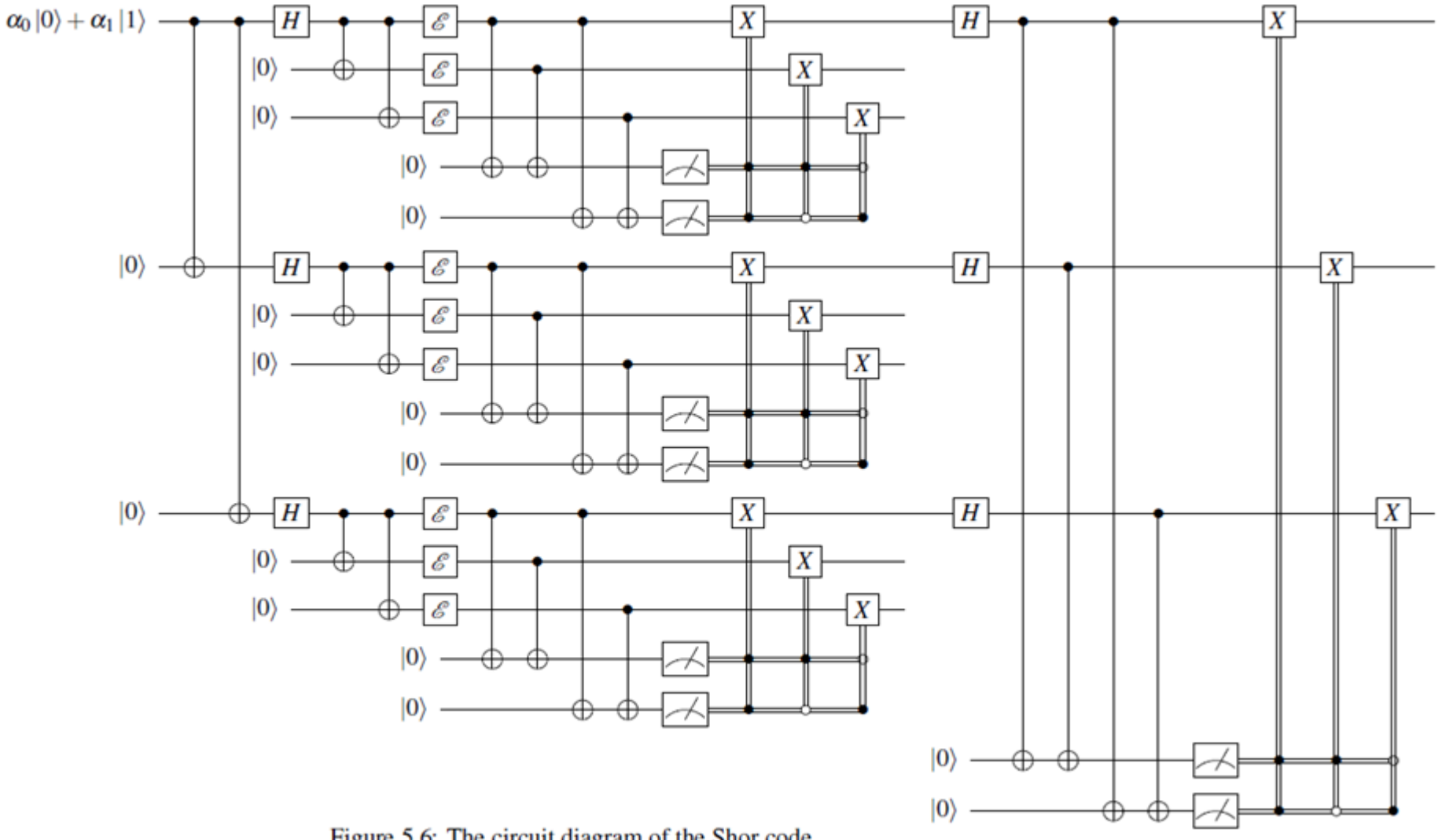

Figure 5.6: The circuit diagram of the Shor code

Chapter 6

CALDERBANK-SHOR-STEANE CODES (CSS CODES)

So far we have seen the Shor code which is very limited in the number of errors it can correct. However it corrects both bit-flip and phase-flip errors, it is still a single-error correcting code since both errors should occur on a single qubit. The family of Calderbank-Shore-Steane codes or CSS codes[31] are able to correct up to t errors depending on the code distance $d(C)$. It is note-worthy that the Shor code itself is a member of CSS family.

CSS codes, benefit from two linear codes, C_1 and C_2, where C_2 is a subcode of C_1, i.e. $C_2 \subset C_1$. C_1 and C_2 possess the $(n-k_1)\times n$ parity-check matrix $\mathscr{H}_1$ and $(n-k_2)\times n$ parity-check matrix $\mathscr{H}_2$, respectively. Using the notation to denote a code's parameter in classical coding theory, we show these linear codes as $[n,k_1,d_1]$ and $[n,k_2,d_2]$. Yet, there is another condition to be met by a CSS code; C_1 and $C_2^{\perp}$ must be capable of correcting the same number of errors t, i.e. they have to have the same code distance. So the distance of the CSS code will be $d = d_1 = d_2^{\perp}$. Although this condition is useful to determine the distance of the CSS code it does not stem from a necessity since the two code's distances, $(C_1, C_2^{\perp})$, can be different and we are still able to determine the distance of the CSS code, which is

$$d(C) = min(d_1, d_2^{\perp}) \tag{6.1}$$

k_1 and k_2 are the number of bits to be encoded by C_1 and C_2, respectively, and $k_2 < k_1$. The number of qubits that can be encoded by a CSS code over these two linear codes is given by $k = k_1 - k_2$.

Notation To be able to distinguish the CSS codes from the linear codes on which they are established, we denote them by $[[n,k,d]]$.

Construction of the qubit states in a CSS code is highly dependent on the notion of cosets of a linear code and its properties we studied in section 3.4. Suppose $w \in C_1$ and $v \in C_2$, then a qubit state in the CSS code over C_1 and C_2 is a uniform superposition of all the elements in coset $w + C_2$. We write

$$|\tilde{w}\rangle = |w + C_2\rangle = \frac{1}{\sqrt{2^{k_2}}} \sum_{v \in C_2} |w + v\rangle \tag{6.2}$$

By theorem 2(v) in section 3.4 the number of distinct cosets of a code is given by 2^{n-k} with the word u which determines the coset is in F_2^n and the number of codewords in code C is 2^k. However the number of cosets used to create the logical states of a CSS code is determined by $2^{k_1-k_2}$ since the codeword w which specifies a certain coset over C_2 is in $F_2^{k_1}$ which is the codespace of C_1. By the same theorem part iv these coset are either identical or disjoint and therefore orthogonal. Summarizing, we have $2^{k_1-k_2}$ normalized, mutually orthogonal qubit states. As we know, the components of a qubit can only take values from the two logical qubits $|0_L\rangle$ and $|1_L\rangle$. This implies that the total number of distinguishable cosets that we need must be exactly 2, i.e. $k_1 - k_2 = 1$.

In the following we prove two important results required to understand the procedure of syndrome measurement and the recovery operation of a CSS code. The first one is an identity that relates a code C to its dual $C^{\perp}$.

$$\sum_{v \in C} (-1)^{v.u} = \begin{cases} 2^k & u \in C^{\perp} \\ \\ 0 & u \notin C^{\perp} \end{cases} \tag{6.3}$$

v and u are both bit strings where v takes the value of the codewords in C. When $u \in C^{\perp}$ the result is straightforward since $C^{\perp}$ is the orthogonal complement of C and $|C| = 2^k$. However the second result where $u \notin C^{\perp}$ is not so trivial. It basically follows from

$$\sum_{x \in \{0,1\}^k} (-1)^{x.y} = 0 \quad ; \quad y \neq 0 \tag{6.4}$$

The above equation states that no matter what y is (must be of the same length as x) as long as it is not equal to a zero word the above summation yields 0. That is because the inner product of the two words $x.y$ results in 0 and 1 equally for the words $x \in \{0,1\}^k$. On the other hand v is, in fact, obtained by encoding a word m of length k using the generator matrix of C, i.e. $v = m\mathscr{G}$. So

$$\sum_{v \in C} (-1)^{v.u} = \sum_{m \in \{0,1\}^k} (-1)^{m\mathscr{G}.u} = \sum_{m \in \{0,1\}^k} (-1)^{m.\mathscr{G}u} = 0 \quad ; \quad \mathscr{G}u \neq 0 \tag{6.5}$$

Since $\mathscr{G}u \neq 0$ we can conclude that $u \notin C^{\perp}$ when the sum is zero.

The second important thing arises when we recover the phase-flip errors and we need to Hadamard the qubit so as to transform the Z error into an X error as we did before.

Saying that the qubit is given by equation 6.2 we apply a bitwise Hadamard gate to it. That is,

$$H^{\otimes n}|w+C_2\rangle = \frac{1}{\sqrt{2^n}}\sum_u \frac{1}{\sqrt{2^{k_2}}}\sum_{v\in C_2}(-1)^{v.u}(-1)^{w.u}|u\rangle$$
$$= \frac{1}{\sqrt{2^{n-k_2}}}\sum_{u\in C_2^{\perp}}(-1)^{w.u}|u\rangle \tag{6.6}$$

The last expression uses the result from equation 6.3. Regarding the error models, we can think of error operators whose effect is determined by an error string e, meaning that the bit-flip (X) or the phase-flip (Z) operators act only when there is a 1 at position i of the error string. For example, if the error string is 0110000 the error operators only act at positions 2 and 3 and do nothing where the bit values are 0. We can model the effect of these error operators in the following way.

$$\mathscr{E}_{bf} : |\tilde{w}\rangle \rightarrow |\tilde{w}+e\rangle \tag{6.7}$$

$$\mathscr{E}_{pf} : |\tilde{w}\rangle \rightarrow (-1)^{\tilde{w}.e}|\tilde{w}\rangle \tag{6.8}$$

Here, the subscripts bf and pf stand for the bit-flip and phase-flip, respectively. The syndrome measurement of a bit-flip error is done by adding some ancilla to the erred qubit, applying the parity-check matrix of C_1 to the ancilla, and finally measuring the state of the ancilla.

$$|\tilde{w}\rangle \otimes |0\rangle \rightarrow |\tilde{w}\rangle \otimes |\mathscr{H}_1\tilde{w}\rangle \tag{6.9}$$

The phase-flip error is also treated as a bit flip after being transformed using a bit-wise Hadamard. To measure the syndrome when this error occurs we can simply apply the

generator matrix of C_2 which is also the parity-check matrix of $C_2^{\perp}$. The use of $\mathscr{G}_2$ is validated when we look at equation 6.6.

$$|\tilde{w}\rangle \otimes |0\rangle \rightarrow |\tilde{w}\rangle \otimes |\mathscr{G}_2\tilde{w}\rangle \tag{6.10}$$

Applying the X gate on the gubits where the syndrome shows an error has occurred, we can recover the corrupted information. In the case of a phase flip we need to re-Hadamard the final state to get back to the initial basis.Obviously, the number of errors coming out of a syndrome measurement should not exceed the number of correctable errors of the CSS code. The Steane code elaborates the error correcting procedure of the CSS codes. This code uses the notion of a self-orthogonal code when $C_2 = C_1^{\perp}$.

6.1 Seven-qubit Steane code

an example of the CSS codes we explore the mechanism through which the 7-qubit Steane code[34], [14], [15],[16], [**?**] corrects errors. This code is based on the 7-bit Hamming code $C[7,4]$ and its dual $C^{\perp}[7,3]$. Note that $C^{\perp} \subset C$ is a requirement for the Steane code to be a CSS code which is fulfilled by $C[7,4]$ and its dual. The qubit states of the Steane code is obtained by superposing the elements of the cosets of $C[7,4]$ over its dual $C^{\perp}[7,3]$. The look up table on the next page shows these cosets. As we can see there are only two distinct cosets and therefore we have two qubit states, one for $|0_L\rangle$ and the other one for $|1_L\rangle$. This is no surprise at all since the number of distinct cosets is given by $2^{k_2-k_1}$ where $k_1 = 4$ and $k_2 = 3$ according to theorem 2(v). Equation 6.2 which gives the qubit states of a CSS code expands in the following form for the Steane code.

$$|0_L\rangle = \frac{1}{\sqrt{8}}\Big[|0000000\rangle + |0111001\rangle + |1011010\rangle + |1100011\rangle + |1101100\rangle + |1010101\rangle + |0110110\rangle + |0001111\rangle\Big] \tag{6.11}$$

$$|1_L\rangle = \frac{1}{\sqrt{8}}\Big[|1111111\rangle + |1000110\rangle + |0100101\rangle + |0011100\rangle + |0010011\rangle + |0101010\rangle + |1001001\rangle + |1110000\rangle\Big] \tag{6.12}$$

To measure the error syndrome for each type of the errors we must add some ancilla so as to measure the effect of the parity-check matrix on the codeword using those ancilla. Here, in the case of the Steane code and for the bit-flip error type the we use the parity-check matirx of the Hamming [7,4] code. For measuring the syndrome of the phase-flip error type, according to equation 6.6 the parity-check matrix of the dual of [7,3] code must be used since we do bit-wise Hadamard transformation. Since $C_1 = C$ and $C_2 = C^{\perp}$ the dual of the [7,3] code is the Hamming [7,4] code itself and therefore we use the same parity-check matrix as for the case of the bit flip to measure the phase-flip error syndrome. The circuit diagrams in figures 6.1 and 6.2 illustrate the syndrome measurement procedure provided that the Hamming parity-check matrix is given as

$$\mathscr{H} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{6.13}$$

Since the distances of the [7,4] Hamming code and its dual are 3 and the Steane code encodes one qubit to seven qubits, this is a CSS [[7,1,3]] code and is able to correct a single error.

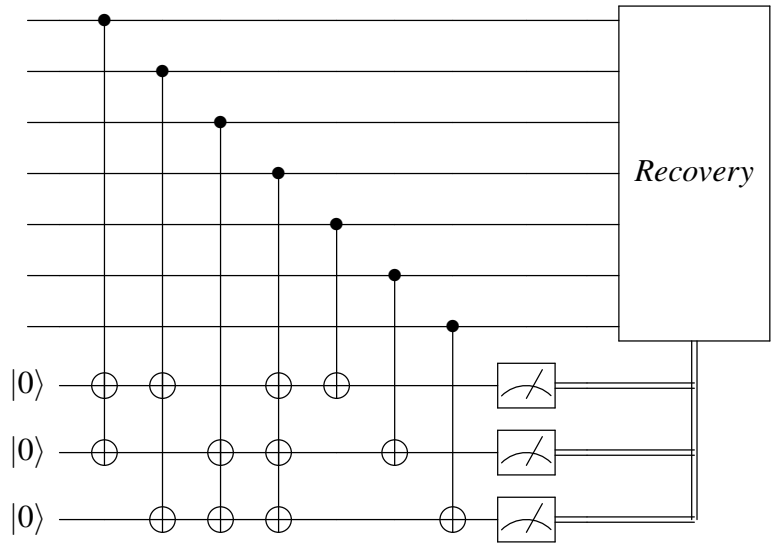

Figure 6.1: Syndrome measurement of the bit-flip error in Steane code

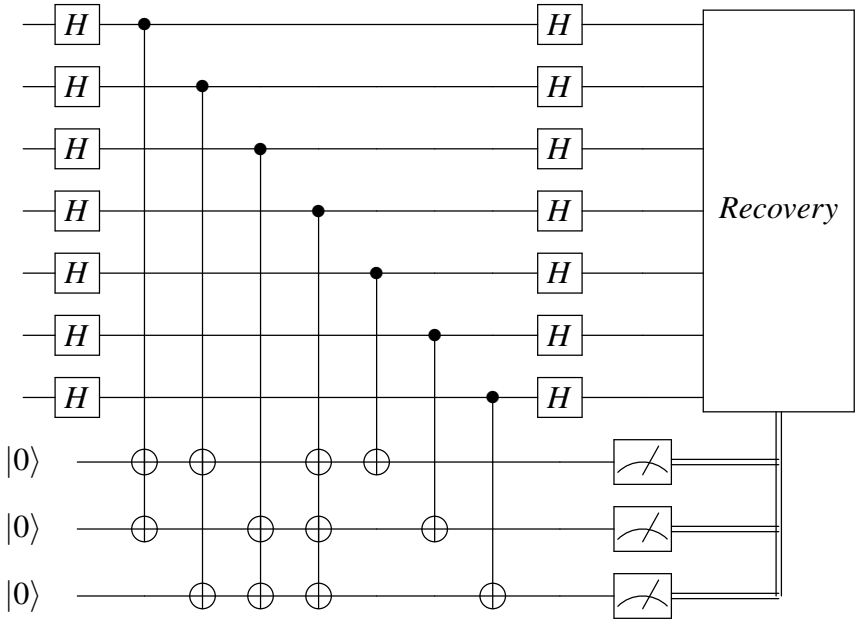

Figure 6.2: Syndrome measurement of the phase-flip error in Steane code

Chapter 7

STABILIZER FORMALISM

In this section we will explore the structure of a larger family of codes [41], the way to construct them, and a few examples to instance this structure and construction. The use of stabilizer codes is realized when we measure the error syndrome. In previous cases where we have to perform a syndrome measurement we measure the effect of the error operators on the codewords, however in stabilizer formalism what needs to be measured is the effect of these operators on some other operators, called the generators of the stabilizer group. This effect, as we will see, is the commutation or the anti-commutation of the error operators and the newly introduced operators, stabilizer generators. The insight into the subject will be developed through using the following equivalence. To prove this equivalence we need to consider these operators in Dirac

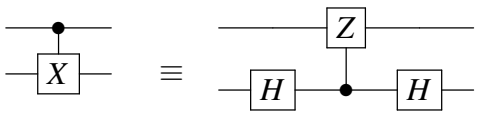

Figure 7.1: The equivalence of a CNOT gate and a Hadamard-conjugated Z gate

notation. The CNOT gate on the left is

$$\text{CNOT or } I \otimes X \equiv |0\rangle\langle 0| \otimes |0\rangle\langle 1| + |0\rangle\langle 0| \otimes |1\rangle\langle 0| + |1\rangle\langle 1| \otimes |0\rangle\langle 1| + |1\rangle\langle 1| \otimes |1\rangle\langle 0| \tag{7.1}$$

The equivalent operation can be written as

$$\begin{aligned}(I\otimes H)(Z\otimes I)(I\otimes H) &= |0\rangle|0\rangle\langle 0|\langle 1| + |0\rangle|1\rangle\langle 0|\langle 0| + |1\rangle|0\rangle\langle 1|\langle 1| + |1\rangle|1\rangle\langle 1|\langle 0| \\ &= |0\rangle\langle 0|\otimes|0\rangle\langle 1| + |0\rangle\langle 0|\otimes|1\rangle\langle 0| + |1\rangle\langle 1|\otimes|0\rangle\langle 1| + |1\rangle\langle 1|\otimes|1\rangle\langle 0| \\ &= I\otimes X \end{aligned} \tag{7.2}$$

where

$$\begin{aligned}I\otimes H = \frac{1}{\sqrt{2}}\Big(&|0\rangle|0\rangle\langle 0|\langle 0| + |0\rangle|1\rangle\langle 0|\langle 0| + |0\rangle|0\rangle\langle 0|\langle 1| - |0\rangle|1\rangle\langle 0|\langle 1| \\ &+ |1\rangle|0\rangle\langle 1|\langle 0| + |1\rangle|1\rangle\langle 1|\langle 0| + |1\rangle|0\rangle\langle 1|\langle 1| - |1\rangle|1\rangle\langle 1|\langle 1|\Big)\end{aligned} \tag{7.3}$$

and

$$Z\otimes I = |0\rangle|0\rangle\langle 0|\langle 0| + |0\rangle|1\rangle\langle 0|\langle 1| - |1\rangle|0\rangle\langle 1|\langle 0| - |1\rangle|1\rangle\langle 1|\langle 1|. \tag{7.4}$$

Measuring the second qubit in the computational basis in the CNOT gate is equivalent to measuring the observable Z in the circuit given on the right. So the circuit diagram for recovering the bit-flip error using the three-qubit repetition code turns into the following circuit. Now, measuring the syndromes has been turned into measuring

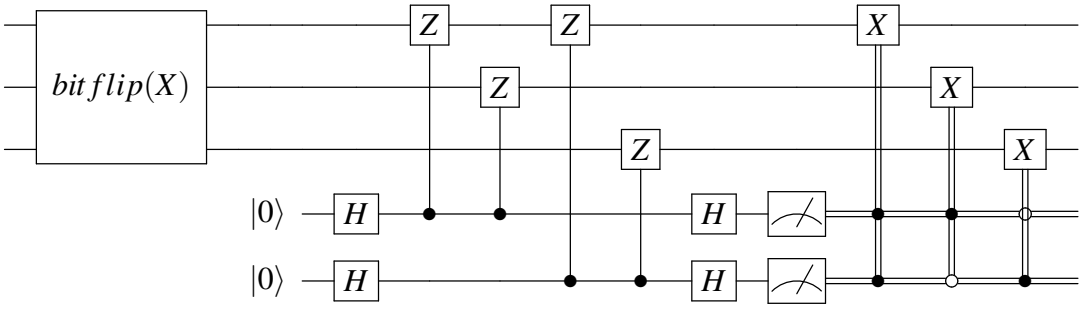

Figure 7.2: Three-qubit code using the mentioned equivalence

these observables

$$Z \otimes Z \otimes I \tag{7.5}$$

$$Z \otimes I \otimes Z \tag{7.6}$$

These are consisted of the operators of the Pauli group and therefore the codewords $\alpha_0 |000\rangle + \alpha_1 |111\rangle$ are eigenstates of these operators with eigenvalue $+1$. The error operators are also the tensor product of the Puali operators, X and I, and the codewords afflicted by the error operators are still eigenstates of the operators 7.5 but this time with eigenvalues either $+1$ or -1. The following table shows the 1-1 correspondence between the two mappings using the eigenvalues of the syndrome operators ($Z \otimes Z \otimes I, Z \otimes I \otimes Z$) when the codewords have undergone the single or no error operators. This unique correspondence helps us to find the location of the error provided

	$Z \otimes Z \otimes I$	$Z \otimes I \otimes Z$
$I \otimes I \otimes I$	$+1$	$+1$
$X \otimes I \otimes I$	-1	-1
$I \otimes X \otimes I$	-1	$+1$
$I \otimes I \otimes X$	$+1$	-1

Table 7.1: The syndrome look-up table for the three-qubit bit flip code

that a single error or no error has occurred. The correspondences can also be obtained by computing the commutators of the syndrome operators and the error operators as

long as the errors are formed by the tensor product of the Pauli operators, just like the syndrome operators. For instance, the bit flip on the second qubit associated with $I \otimes X \otimes I$ commutes with $Z \otimes I \otimes Z$ whereas anti-commutes with the other operator $Z \otimes Z \otimes I$. This property forms the cornerstone of the stabilizer codes. The syndrome operators like 7.5 define a subspace of codewords where the codewords are the eigen-states of the syndrome operators with $+1$ eigenvalue. These operators generate a group called stabilizer and the operators themselves are called the generators of the stabilizer group. Measuring the generators or in other words the effect of them on the erred states gives the syndrome.

For the phase-flip error the controlled Z operator can be replaced by a combination of the Hadamard and CNOT gates similar to that in figure 7.2. So the generators of the stabilizer group will be

$$X \otimes X \otimes I \tag{7.7}$$

$$X \otimes I \otimes X \tag{7.8}$$

7.1 Structure and construction

family of stabilizer codes form a larger group of codes which contains the CSS codes as well. The group theoretic notions and in particular the operators that anticommute with each other are the cornerstone of the structure of these codes [29].

Suppose that a set of states $|\psi_i\rangle$ satisfy this equation

$$S|\psi_i\rangle = |\psi_i\rangle \tag{7.9}$$

where S is an operator. This equation defines an eigenspace of S with eigenvalue +1. Now consider another operator E that anticommutes with S, i.e. $SE = -ES$ therefore

$$S(E|\psi_i\rangle) = -ES|\psi_i\rangle = -E|\psi_i\rangle \tag{7.10}$$

Starting out in +1 eigenspace of S the error operator takes us to -1 eigenspace of S. Having two or more of the operators like S limits the eigenspace of question and therefore gives a more unique subspace which enables us to spot the location of error provided the error operator takes us to that restricted subspace and there is no other error that takes the state to that eigenspace.

In general, the QEC criterion requires that

$$\langle l|\mathscr{E}_i^\dagger \mathscr{E}_j|m\rangle = C_{ij}\delta_{lm} \tag{7.11}$$

Suppose that $S = \{s_k\}$ where $s_k|\psi\rangle = |\psi\rangle$, i.e. This equation defines the +1 eigenspace of S. There is at least one s_k such that it anticommutes with $\mathscr{E}_i^\dagger \mathscr{E}_j$, therefore

$$\langle l|\mathscr{E}_i^\dagger \mathscr{E}_j|m\rangle = \langle l|\mathscr{E}_i^\dagger \mathscr{E}_j s_k|m\rangle = -\langle l|s_k \mathscr{E}_i^\dagger \mathscr{E}_j|m\rangle = -\langle l|\mathscr{E}_i^\dagger \mathscr{E}_j|m\rangle \tag{7.12}$$

So $\langle l|\mathscr{E}_i^\dagger \mathscr{E}_j|m\rangle = 0$. This means that if we choose proper operators (and thus the proper eigenspace) that anticommute with the error operators the code with the eigenstates of the chosen operators corrects the errors. The operators with which the Pauli error operators either commute or anticommute is the Pauli group itself. Before looking at the properties of stabilizer group it is worth reviewing a few properties of the Pauli

group. The Pauli group $\mathscr{P}$ consists of the following elements

$$i^k\{I,X,Y,Z\} \quad ; \quad k=\{0,1,2,3\} \tag{7.13}$$

The representation of the Pauli group that we use in quantum computation is an n fold tensor product of these operators where each element is in the from

$$i^k(P_1 \otimes P_2 \otimes ... \otimes P_n) \quad ; \quad k=\{0,1,2,3\} \tag{7.14}$$

The properties of the Pauli group for elements P and Q are

- The elements of the Pauli group square to $\pm I$. $P^2 = \pm I$
- The elements of the Pauli group either commute or anticommute. $PQ = QP$ or $PQ = -QP$
- The elements of the Pauli group are all unitary. $PP^\dagger = I$

A stabilizer group $\mathscr{S}$ is a subgroup of $\mathscr{P}$ where all elements commute and it does not contain the element -I. e.g. a choice of stabilizers for the three-qubit code can be $\{III, ZZI, ZIZ, IZZ\}$ (We may use this notation III instead of $I \otimes I \otimes I$). Among these stabilizers there are elements upon which other elements of the group can be formed. For example, $III = (ZZI)^2$ and $IZZ = (ZZI)(ZIZ)$. Generators of a stabilizer group is a minimal set of independent elements of the stabilizer that we can derive other elements of the stabilizer by multiplying them. So the generators of the three-qubit stabilizer group are ZZI and ZIZ which we denote it as $\mathscr{S} =< ZZI, IZZ >$. We do not need all the elements of the stabilizer group the only elements that we need are the generators.

When considering the QEC requirement in equation 7.11, two cases arise provided that the error operators are in the Pauli group. Suppose that $|\rangle\, l$ and$|m\rangle$ are in +1 subspace of the stabilizer. Then the first case is when P is not in the stabilizer group. That is

$$\langle l|\, P\,|m\rangle = \langle l|\, Ps_k\,|m\rangle = -\langle l|\, s_k P\,|m\rangle = -\langle l|\, P\,|m\rangle \tag{7.15}$$

So we have $\langle l|\, P\,|m\rangle = 0$. P can be a product of the error operators like $\mathscr{E}_i^\dagger \mathscr{E}_j$. The second case occurs where P is the stabilizer group, $P \in \mathscr{S}$ in which case

$$\langle l|\, \mathscr{E}_i^\dagger \mathscr{E}_j\,|m\rangle = \langle l|\, s_k\,|m\rangle = \langle l|m\rangle = \delta_{lm} \tag{7.16}$$

So the Pauli error operators also fulfil this condition so long as the stabilizer group provides operators that anticommute with these errors. As an example we can look at the three-qubit code again with given generators as before. We know that the correctable set of errors only consist of error operators that causes no error or at last one error on one of the qubits, $\{III, XII, IXI, IIX\}$. The set comprised of the products of these error operators is $\{III, XII, IXI, IIX, XXI, XIX, IXX, XXX\}$. Having a look at this set, III is in the stabilizer so the first case appears and the rest of the set anticommute with the given generators *ZZI* and *ZIZ*.

To check whether two Pauli group elements $P_1 \otimes P_2 \otimes ... \otimes P_n$ and $Q_1 \otimes Q_2 \otimes ... \otimes Q_n$ commute or anticommute we need to check for the locations in which P and Q are different. In case of an even number of different locations in P and Q they commute

otherwise they anticommute.

Finding the dimension of the stabilizer subspace is worthwhile since it tells us about the number of the generators we need to define a codespace. First, we need to consider two important points. Since the stabilizer group does not include the element -I they square to +I and have eigenvalues ± 1. The other point is that all the Pauli group elements of which the stabilizer elements are comprised have trace 0 except I which has trace 2. So among the generators of a given stabilizer $s_1, s_2, ..., s_r$ if we choose the first one $Tr(s_1) = 0$. Note that the element I is not a generator but is in the stabilizer and -I is excluded at all. Since s_1 squares to +I it has eigenvalues ± 1 and since its trace is zero we can say that half of the eigenstates of s_1 has eigenvalue +1 and the other half are with eigenvalue -1. So the equation $s_1 |\psi\rangle = |\psi\rangle$ split the eigenspace of s_1 into halves. In a stabilizer code the +1 eigenspace is where we should start out. The codespace is 2^n-dimensional. Thus the dimensions of the +1 eigenspace of s_1 is $\frac{1}{2}(2^n) = 2^{n-1}$. The second generator s_2 with the same properties restricts the eigenspace of s_1 to the eigenstates that satisfy $s_2 |\psi\rangle = |\psi\rangle$. To see the share of +1 eigenspace of s_2 in +1 eigenspace of s_1 we may use the following operator since it projects to +1 eigenspace of s_1.

$$\frac{1}{2}(I + s_1) \tag{7.17}$$

Since $Tr(\frac{1}{2}(I + s_1)s_2) = 0$ and there are two eigenvalues ± 1, we again conclude that half of the eigenspace of the 2^{n-1}-dimensional +1 eigenspace of s_1 has +1 eigenstates of s_2. So it has 2^{n-2} dimensions. By induction, the dimensions of the stabilizer subspace with r generators is 2^{n-r}. For instance, for the three-qubit stabilizer code with

generators *ZZI* and *ZIZ*, the dimension of the stabilizer subspace is $2^{3-2} = 2$ as we expect since there are only two codewords that span the codespace, $|000\rangle$ and $|111\rangle$. In brief, the r generators that characterize the stabilizer code limit the codespace to an eigenspace so that when a certain error from the set of correctable errors occur the code would be able to spot the error location. In other words, it gives us a unique syndrome for every correctable error operator provided that the elements of the stabilizer all commute and does not contain the element -I.

In exploring the structure of the stabilizer codes, it is worthwhile to take a look at those operators which represent the action of a bit flip and phase flip on an n-qubit logical codeword. These operators which are not in the stabilizer but has a relation with the stabilizer and are n fold operators in the Pauli group. We need to see the following definitions.

Definition 1 *The centralizer of the stabilizer group $\mathscr{S}$ in $\mathscr{P}_n$ is the set of operators $p \in \mathscr{P}_n$ that satisfy $ps = sp \; \forall \; s \in \mathscr{S}$. So it is the set of Pauli operators that commute with every s in the stabilizer.*

Definition 2 *The normalizer $\mathscr{N}$ of $\mathscr{S}$ in $\mathscr{P}_n$ is the set of operators $p \in \mathscr{P}_n$ such that $psp^\dagger \in \mathscr{S} \; \forall \; s \in \mathscr{S}$.*

Since the stabilizer group does not contain the element -I, its centralizer and normalizer coincide. The stabilizer itself is in the normalizer since if p is a stabilizer $psp^\dagger = spp^\dagger = s \in \mathscr{S}$. We used the fact that the stabilizers are unitary. So $\mathscr{S} \subset \mathscr{N}$.

The logical X and Z operators of a stabilizer code, denoted by $\bar{X}$ and $\bar{Z}$, respectively, are also called the encoded operations. These operators are not in the stabilizer group but are in the rest of the normalizer set of the stabilizer group. So they are in $\mathscr{N} - \mathscr{S}$. For the aforementioned example of the three-qubit code, the operators which are in $\mathscr{N} - \mathscr{S}$ are given as below with $|0_L\rangle = |000\rangle$ and $|1_L\rangle = |111\rangle$.

$$i^k\{XXX, YYX, YXY, XYY, ZII, IZI, IIZ, ZZZ, YYY, XXY, XYX, YXX\} \tag{7.18}$$

The encoded bit-flip operation is

$$\bar{X} = XXX \quad \text{which acts as a bit flip} \quad \bar{X}|0_L\rangle = |1_L\rangle \quad \text{and} \quad \bar{X}|1_L\rangle = |0_L\rangle \tag{7.19}$$

and the encoded phase-flip operation is

$$\bar{Z} = ZZZ \quad \text{which acts as a phase flip} \quad \bar{Z}|0_L\rangle = |0_L\rangle \quad \text{and} \quad \bar{Z}|1_L\rangle = -|1_L\rangle \tag{7.20}$$

The following theorem determines if a set of errors is correctable using a given stabilizer code or not.

Theorem 5 *Suppose $\mathscr{S}$ is a stabilizer with normalizer $\mathscr{N}$. Let $\mathscr{E}_i$ denote a set of Pauli error operators. If $\mathscr{E}_i^\dagger \mathscr{E}_j \notin \mathscr{N} - \mathscr{S}$, then $\mathscr{E}_i$ is a set of correctable errors.*

Proof, *Two cases arise here. The first case is when $\mathscr{E}_i^\dagger \mathscr{E}_j$ is in $\mathscr{S}$. then*

$$\langle l|\, \mathscr{E}_i^\dagger \mathscr{E}_j\, |m\rangle = \langle l|\, s\, |m\rangle = \delta_{lm}$$

The second case occurs when $\mathscr{E}_i^\dagger \mathscr{E}_j$ is neither in $\mathscr{S}$ nor in $\mathscr{N}$. Since for the stabilizer group $\mathscr{S}$ the normalizer and the centralizer are the same the only possibility is that the product of the error operators $\mathscr{E}_i^\dagger \mathscr{E}_j$ anticommutes with the stabilizer operator $\mathscr{E}_i^\dagger \mathscr{E}_j s =$

$-s\mathscr{E}_i^\dagger\mathscr{E}_j$. *The same reasoning as for 7.16 applies and therefore* $\langle l|\,\mathscr{E}_i^\dagger\mathscr{E}_j\,|m\rangle = 0$. *Thus if the set of errors is not in* $\mathscr{N}-\mathscr{S}$ *that is a correctable set of errors.*

The set of correctable errors for the three-qubit code is the set of single errors $\{III, XII, IXI, IIX\}$ and is not contained in $\mathscr{N}-\mathscr{S}$, 7.18, for $\mathscr{S} =< ZZI, ZIZ >$. A useful tool in representing the generators of a stabilizer group is the check matrix which is the counterpart of the parity-check matrix in classical coding theory. Suppose the stabilizer has r=n-k (n being the number of qubits and k the number of qubits to be encoded) generators, $\mathscr{S} =< s_1, s_2, ..., s_r >$. The check matrix is a $r \times 2n$ matrix in which each row represent a generator. The rules to form the check matrix of a stabilizer $\mathscr{S}$ is given as follows.

- The left-hand side of the matrix is reserved for the generators which only include I and X.
- The right-hand side of the matrix is reserved for the generators which only include I and Z.
- If there are 1s on both sides of the matrix, then there exists Y or the product of X and Z in the generator.
- In the ith row of the matrix if s_i contains an I on the jth qubit (column), then the jth and the jth+n column in that row are 0.
- In the ith row of the matrix if s_i contains an X on the jth qubit, then the jth column in that row is 1 and the jth+n column in the ith row is 0.
- In the ith row of the matrix if s_i contains a Z on the jth qubit, then the jth column in the ith row is 0 and the jth+n column in that row is 1.

Below is the check matrix of the seven-qubit Steane code.

$$\left(\begin{array}{ccccccc|ccccccc} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}\right) \tag{7.21}$$

To check whether the generators (roughly, the rows of the check matrix) commute or in other words to see if a given check matrix represents a stabilizer we can define a $2n \times 2n$ matrix, Λ

$$\Lambda = \begin{pmatrix} 0 & I_{n\times n} \\ I_{n\times n} & 0 \end{pmatrix} \tag{7.22}$$

using which s and s' commute if and only if the following condition is met.

$$r(s)\Lambda r(s')^T = 0 \tag{7.23}$$

where $r(s)$ is the 2n-dimensional row vector that contains the generator s. The reason why this twisted inner product implies the commutation of the generators is that the rows contain 0 and 1 and the addition is done modulo 2. If two generators commute the number of locations where X and Z differ in those generators must be even. This implies an even number of 1s in the addition operation of the product so that 1 annihilates while adding them modulo 2. So the odd number of these additions implies that

the two generators anticommute which is not desired in a stabilizer group. As a special case, we explore the check matrix of the CSS codes, CSS(C_1, C_2) since CSS codes are a special case of the stabilizer codes. Each generator of these codes only contain I and X or I and Z and not both.

Suppose that C_1 and C_2 are [n,k_1] and [n,k_2] codes where C_2 is a subcode of C_1, $C_2 \subset C_1$ and C_1 and $C_2^{\perp}$ are both t-error correcting codes. The check matrix of the CSS stabilizer code is

$$\left(\begin{array}{c|c} \mathscr{H}(C_2^{\perp}) & 0 \\ 0 & \mathscr{H}(C_1) \end{array}\right) \tag{7.24}$$

To see if this check matrix represent a stabilizer we the following commutativity condition based on 7.23.

$$\mathscr{H}(C_2^{\perp})\mathscr{H}(C_1)^T = 0 \tag{7.25}$$

It turns out that this condition is met by the CSS codes since

$$\mathscr{H}(C_2^{\perp})\mathscr{H}(C_1)^T = (\mathscr{H}(C_1)\mathscr{G}(C_2))^T = 0 \tag{7.26}$$

Here, we used the fact that $C_2 \subset C_1$ and therefore the rows of $G(C_2)$ are all contained in $G(C_1)$ and since $G(C_1)$ is orthogonal to $G(C_2)$ we have the condition above.

7.2 Examples of stabilizer codes

7.2.1 Steane code

of the instances of the stabilizer codes is the seven-qubit Steane code. As we know the Steane code [34] is a CSS code whose generators possess either X or Z and not both.

Using the check matrix given in 7.24 and the fact that in the Steane code $C_2^{\perp} = C_1$, the parity-check matrices nested in the check matrix of a CSS code are the same and equal to the parity-check matrix of the [7,4] code. If we take the parity-check matrix of the [7,4] code to be (the choice of the parity-check matrix is not unique)

$$\mathscr{H} = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \tag{7.27}$$

the check matrix of the Steane code will be the one given in 7.21. According to this check matrix the generators of the Steane code must be of the following form. Since the stabilizer subspace of this code is a 2-dimensional space (there are only two states which span the stabilizer space, the logical qubits) the number of the generators required is $n-1=r$ and $r=6$. The encoded operations (logical X and Z) of the Steane

code are

$$\bar{X} = XXXXXXX \tag{7.28}$$

$$\bar{Z} = ZZZZZZZ \tag{7.29}$$

These logical operations are in $\mathscr{N} - \mathscr{S}$ and direct operations of them on the logical qubits $|0_L\rangle$ and $|1_L\rangle$ in 6.11 give the bit flip and phase flip operations. Of course the correctable set of errors of the Steane code are not in $\mathscr{N} - \mathscr{S}$. The syndrome look-up table which demonstrates the correspondence between the syndrome measured (generator measured) and the error operator (location of the error) is given for the bit-flip

generator	operator
s_1	$IIIXXXX$
s_2	$IXXIIXX$
s_3	$XIXIXIX$
s_4	$IIIZZZZ$
s_5	$IZZIIZZ$
s_6	$ZIZIZIZ$

Table 7.2: The generators of the Steane code

errors. The unique correspondences that guide us to the location of the error can be easily seen from the table above. If we start out in +1 eigenspace of the stabilizer codespace $s_i|\psi\rangle = |\psi\rangle$ and if $|\psi\rangle$ undergoes a bit-flip error like $XIIIIII$ the final state is still an eigenstate of all the generators of the stabilizer code this time with eigenvalues ± 1 depending on the generator. In fact the generators specify the eigenspace in which the error has occurred and so looking for that specific eigenspace would be an easy job.

We can also construct the circuit diagram which outputs the syndromes which are in essence the generators measured. To perform the generator measurements we may use the following equivalence. When measuring a the output of a Hermitian and unitary gate U ($UU^\dagger = I$ and $U^2 = I$)we may think of this operator either as a quantum gate

	IIIXXXX	*IXXIIXX*	*XIXIXIX*	*IIIZZZZ*	*IZZIIZZ*	*ZIZIZIZ*
IIIIIII	+1	+1	+1	+1	+1	+1
XIIIIII	+1	+1	+1	+1	+1	-1
IXIIIII	+1	+1	+1	+1	-1	+1
IIXIIII	+1	+1	+1	+1	-1	-1
IIIXIII	+1	+1	+1	-1	+1	+1
IIIIXII	+1	+1	+1	-1	+1	-1
IIIIIXI	+1	+1	+1	-1	-1	+1
IIIIIIX	+1	+1	+1	-1	-1	-1

Table 7.3: The syndrome look-up table of the Steane code

acting on the input qubit or as an observable. We can devise a circuit that gives out the state of the output qubit after being passed through U. Now if want to measure U as an observable what we are looking for is actually its eigenvalues and since it squares to I it has two eigenvalues ± 1. Instead of destructively measuring the output state of the

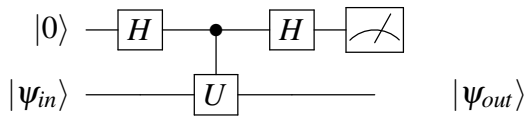

Figure 7.3: The circuit used to measure an observable

qubit we may measure the control qubit conjugated by Hadamard gates. The reason why we can alternatively measure the control qubit is given in the following calcula-

tions. Using the Dirac notation we may write the effect of this circuit, knowing that $U|\psi_{in}\rangle = |\psi_{out}\rangle$.

$$\begin{aligned}
(H\otimes I)(|0\rangle\langle 0|\otimes I + |1\rangle\langle 1|\otimes U)(H\otimes I)|0\rangle|\psi_{in}\rangle = \\
\frac{1}{\sqrt{2}}(H\otimes I)(|0\rangle\langle 0|\otimes I + |1\rangle\langle 1|\otimes U)(|0\rangle + |1\rangle)\otimes|\psi_{in}\rangle = \\
\frac{1}{\sqrt{2}}(H\otimes I)\Big[|0\rangle|\psi_{in}\rangle + |1\rangle|\psi_{out}\rangle\Big] = \\
\frac{1}{\sqrt{2}}\Big[|+\rangle|\psi_{in}\rangle + |-\rangle|\psi_{out}\rangle\Big]
\end{aligned} \tag{7.30}$$

The two eigenvalues of the observable U can be associated to each of the Hadamard bases appeared in the first qubit out of the circuit and therefore measuring the first qubit necessarily gives the result of the measurement of the observable. Using the above equivalence we can now construct the circuit diagram used for the syndrome measurement in the Steane code. In this circuit the first six qubits started at $|0\rangle$ are the ancilla to measure the syndromes and the seven registers at the bottom are the input qubits.

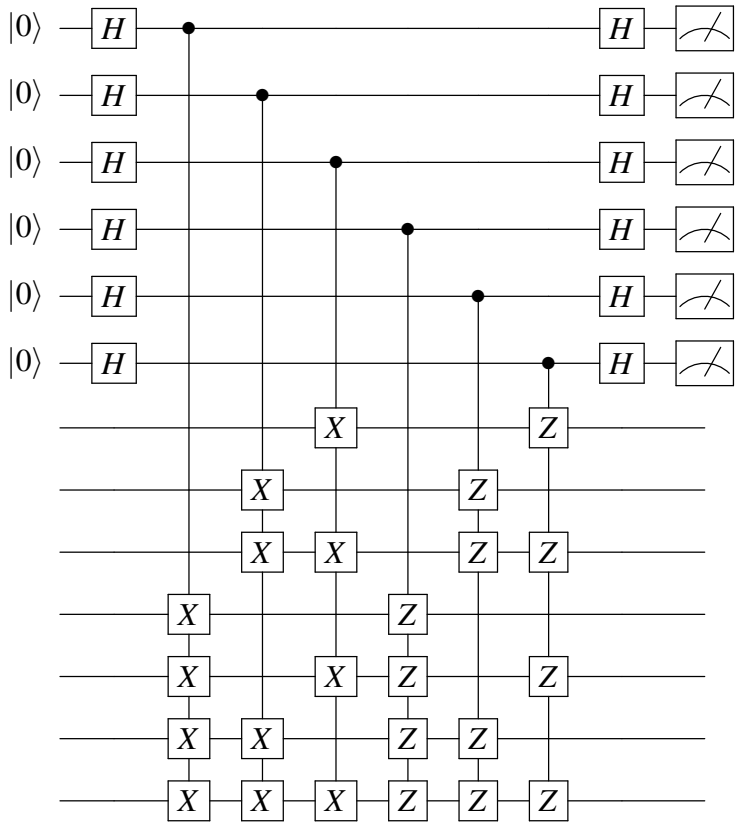

Figure 7.4: The circuit diagram for measuring the generators of the Steane code

7.2.2 Shor code

other well-known CSS code is the nine-qubit Shor code[26]. Finding the generators of this code is more of an intuitive approach rather than forming the check matrix and translate into operator language, however, I would give its check matrix alongside the more straightforward way we conclude the generators. The shor code is in fact form by concatenation of two three-qubit code, the three-qubit bit-flip code and the three-qubit phase-flip code. The Shor code has eight generators. Knowing the generators of each code and considering the circuit diagram of the Shor code leads us to pick these generators for this code. The following operators are shown to be the logical operators in

generator	operator
s_1	ZZIIIIIII
s_2	ZIZIIIIII
s_3	IIIZZIIII
s_4	IIIZIZIII
s_5	IIIIIIZZI
s_6	IIIIIIZIZ
s_7	XXXXXXIII
s_8	XXXIIIXXX

Table 7.4: The generators of the Shor code

the Shor code by direct application of them on the logical qubits 5.17.

$$\bar{X} = XXXXXXXXX \tag{7.31}$$

$$\bar{Z} = ZZZZZZZZZ \tag{7.32}$$

We can also form the check matrix of the Shor code.

$$\left(\begin{array}{ccccccccc|ccccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1
\end{array}\right) \tag{7.33}$$

The circuit diagram for the Shor code generator measurement is given on the next page.

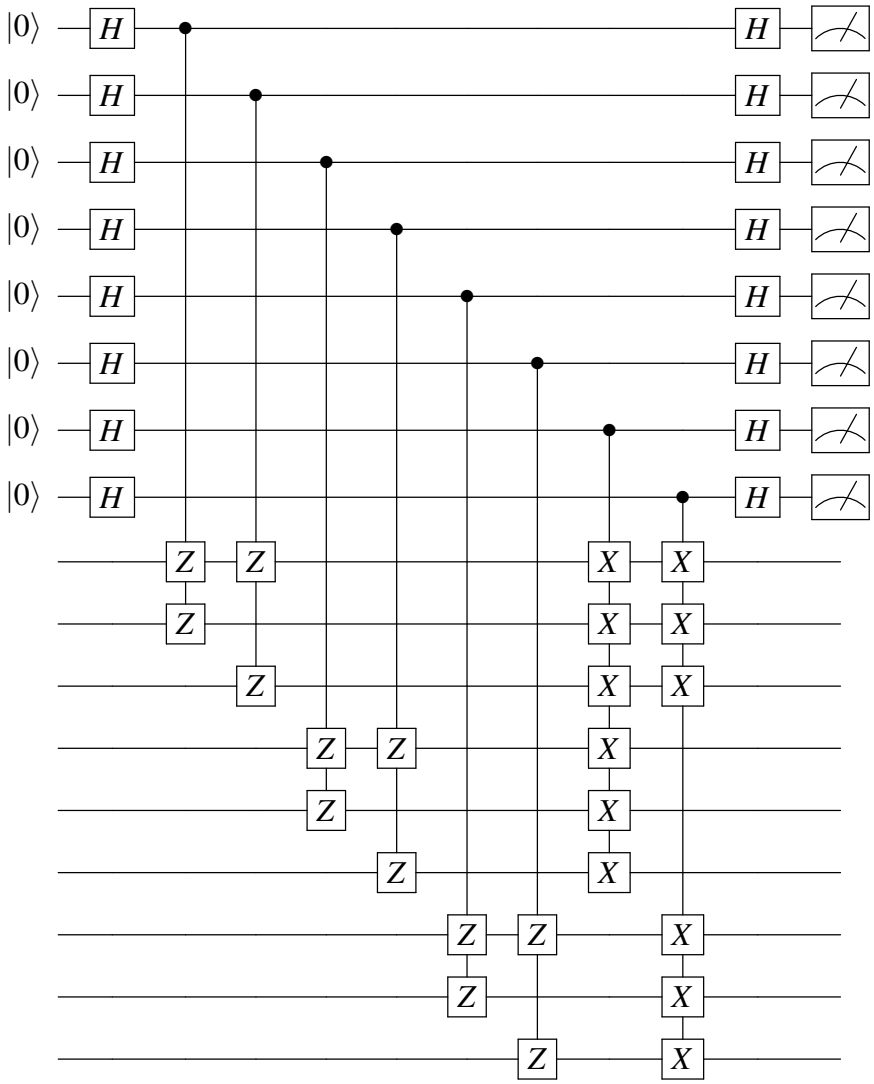

Figure 7.5: The circuit diagram for measuring the generators of the Shor code

7.2.3 Five-qubit code

far we have only seen the CSS stabilizer codes. An example for the non-CSS codes is the five-qubit code [42] whose generators do not solely include X or Z. On the contrary to the CSS codes, the five-qubit code do not possess separate generators consisting of only X or only Z. Five qubits is the minimum number of qubits for quantum error correction so that the code can detect and correct a single error. The four generators of this code are As it can be seen from the table the each generator is the cyclic permuta-

generator	*operator*
s_1	$XZZXI$
s_2	$IXZZX$
s_3	$XIXZZ$
s_4	$ZXIXZ$

Table 7.5: The generators of the five-qubit code

tion of the previous one and thus we can derive all other generators from s_1. The fifth permutation is not a generator since it is not independent of the others. Besides the

number of generators is only four. The logical qubits are

$$\begin{aligned}|0_L\rangle &= \tfrac{1}{4}\Big[|00000\rangle + |10010\rangle + |01001\rangle + |10100\rangle \\ &\quad + |01010\rangle - |11011\rangle - |00110\rangle - |11000\rangle \\ &\quad - |11101\rangle - |00011\rangle - |11110\rangle - |01111\rangle \\ &\quad - |10001\rangle - |01100\rangle - |10111\rangle + |00101\rangle\Big] \end{aligned} \tag{7.34}$$

and

$$\begin{aligned}|1_L\rangle &= \tfrac{1}{4}\Big[|11111\rangle + |01101\rangle + |10110\rangle + |01011\rangle \\ &\quad + |10101\rangle - |00100\rangle - |11001\rangle - |00111\rangle \\ &\quad - |00010\rangle - |11100\rangle - |00001\rangle - |10000\rangle \\ &\quad - |01110\rangle - |10011\rangle - |01000\rangle + |11010\rangle\Big] \end{aligned} \tag{7.35}$$

The logical X and Z of this code are given as below.

$$\bar{X} = XXXXX \tag{7.36}$$

$$\bar{Z} = ZZZZZ \tag{7.37}$$

7.3 State preparation

give a complete picture of the stabilizer codes one should be aware of all the capabilities of the stabilizers. The encoding procedure [40] or the logical state preparation in a stabilizer code is the procedure through which the logical qubits $|0_L\rangle$ and $|1_L\rangle$ are constructed. These codewords are +1 eigenstates of the generators of the stabilizer code since we should start out in +1 eigenspace of the stabilizer. The idea of producing

the logical states in the stabilizer codes is based on the fact that we start with an n fold input state and check whether it is a +1 eigenstate of the generators and if it is not using a proper operator we force it to be in such eigenspace. Since the stabilizers are in Pauli group and hence Hermitian and unitary they square to I and have ± 1 eigenvalues. The following schematic circuit diagram shows the measurement procedure of such operator. It is similar to that we already used in measuring the generators, figure. Tracking

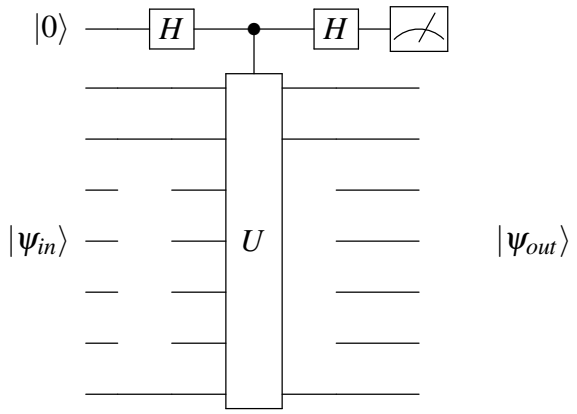

Figure 7.6: The encoding circuit diagram for the stabilizer codes

the steps of this circuit we will find out in what final state the composite system is. The final state is

$$|\psi_{out}\rangle = \frac{1}{2}\Big(|\psi_{in}\rangle + U|\psi_{in}\rangle\Big)|0\rangle + \frac{1}{2}\Big(|\psi_{in}\rangle - U|\psi_{in}\rangle\Big)|1\rangle \tag{7.38}$$

If the ancilla (the upper $|0\rangle$) is measured to be in the state $|0\rangle$ then the state of the input qubits is

$$|\psi_{out}\rangle = |\psi_{in}\rangle + U|\psi_{in}\rangle \tag{7.39}$$

Since U is Hermitian and unitary and by applying it to the equation above we get $U|\psi_{out}\rangle = |\psi_{out}\rangle$. So if the ancilla is in $|0\rangle$ the final state is a +1 eigenstate of U.

The other case occurs when the ancilla is measured $|1\rangle$ in which case the final state of the n fold qubit is a -1 eigenstate of U. In practice, we have a number of generators which all together determine whether the input state which is going to be encoded is in simultaneous +1 or -1 eigenspace of all of them. A proper operator can take the state to +1 eigenspace if it is in the opposite one.

Chapter 8

FAULT-TOLERANT QUANTUM COMPUATION

Prior to introducing any issue regarding the fault-tolerant quantum computing it is noteworthy that the errors with which we are concerned in this chapter are incoherent errors. Coherent errors are those which lead to a factorizable state of the environment when speaking about the state of the system coupled with the environment's state. Back to section 4.1, if the coupled state of the system-environment is $|\psi\rangle$ it can be written as

$$|\psi\rangle = (\alpha_0 |0\rangle + \alpha_1 |1\rangle)_{sys} \otimes |E\rangle_{env} . \tag{8.1}$$

So the otherwise case is referred to as the incoherent error. The difference which our assumption of having incoherent errors makes in fault-tolerant quantum computation is in the probability of being at least one error in the output. If we assume that the probability of introducing one error to the qubits being processed in a quantum circuit due to the failure of each gate is p, then the probability of introducing at least one error in the output of a circuit with S gates is Sp provided that the errors are all incoherent. We may show that in case of having all errors coherent this probability increases to $S^2 p$. To show that we consider a partial rotation of the qubit about the x-axis (There is no preferred choice of axis, so it can be any of the x,y, or z axes.) In section 4.1 we argued that a partial rotation of the qubit about an axis leads to a coherent error. Now

we assume that we have such a unitary rotation [40]

$$e^{-i\delta X} = (cos\delta)I - i(sin\delta)X \tag{8.2}$$

Applying this to the qubit state

$$e^{-i\delta X}\,|\psi\rangle = cos\delta\,|\psi\rangle - isin\delta X\,|\psi\rangle \tag{8.3}$$

Equation 8.3 implies that the probability of occurring no error to the qubit is

$$p(|\psi\rangle) = cos^2\delta \approx 1 - \delta^2 \tag{8.4}$$

and the probability of having an X error is

$$p(X\,|\psi\rangle) = sin^2\delta \approx \delta^2 \tag{8.5}$$

The coherent error usually occurs due to the bad control of the quantum gates[28] so the probability of having an error in the output qubit of a gate being δ^2 is in fact the probability of the failure of the gate. Now for S gates in the circuit with p being the probability of failure of one gate, the unitary rotational transformation is

$$e^{-iS\delta X} = cos(S\delta)I - isin(S\delta)X \tag{8.6}$$

and the probability of having at least one error in the output qubit is

$$p(X\,|\psi\rangle) = sin^2(S\delta) \approx S^2\delta^2 = S^2p \tag{8.7}$$

Fault-tolerant computation enables us to do computation using faulty gates. In fact the fault-tolerant (FT) circuits are robust enough so that the computation performed by them does not result in an unrecoverable set of errors provided that the probabil-

ity p of failure of each gate in the circuit is below a constant threshold. Instead, they introduce an improvement to the probability of such in the output. To do so we replace each qubit with an encoded block of qubits using a certain error-correcting code such as the Steane code and each gate with an encoded gate as explained below and finally we periodically apply a QEC method after the state preparation (encoding the qubits) and after each gate. The QEC method also needs to be done fault-tolerantly. With this interleaved application of QEC the FT quantum computation prevents error accumulation as the input qubit undergoes various stages of the circuit. Without the idea of fault-tolerance we expect to have an probability $O(p)$ in the output, however carrying out the same quantum computation fault-tolerantly results in an improvement of $O(p^2)$. Figure 8.1 displays a circuit consists of only a Hadamard gate and an CNOT gate and instances the scheme of a FT circuit [27]. The idea of encoding the input

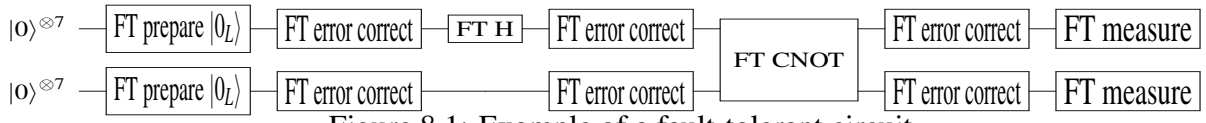

Figure 8.1: Example of a fault-tolerant circuit

qubits using QEC methods lies on the fact that if any one of the qubits is subject to error the output data is still recoverable but apart from the components of a FT circuit which are error prone, error can propagate through the circuit from one component to another. The error itself may occur as a pre-existing error introduced in the input or due to the failure of the components. Thus the idea of fault tolerance by itself cannot fulfil the bid to have robust recoverable data and the circuits must be constructed in such a way that they prevent error propagation or be tolerant to this problem, hence

we should include combating this problem in the notion of fault-tolerance. Figures 8.2 and 8.3 and the calculation that follows illustrate an example of error propagation in and encoding the CNOT gate using the three-qubit code [28]. If an error occurs

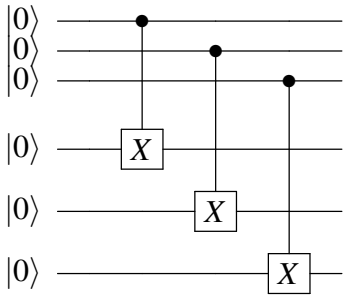

Figure 8.2: Fault-tolerant encoded CNOT gate

in the first qubit of the first encoded block it only affects the first qubit of the second block of encoded qubits. Figure 8.3 shows how such an error propagates throughout the second block if the circuit is not implemented fault-tolerantly. Through proving

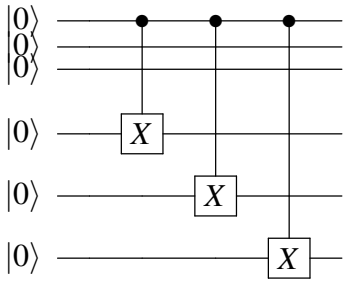

Figure 8.3: Encoded CNOT gate

an equivalence we show that an error on the control wire can propagate to the target qubit in a CNOT gate and the result can be extended to the FT encoded CNOT gate. The equivalence implies that a bit-flip error on the first qubit before the CNOT gate afflicts the output as if one bit-flip error has occurred to each of the control and target

qubits after the CNOT gate has been carried out. Figure 8.4 shows this equivalence.

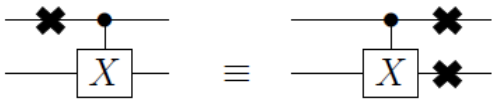

Figure 8.4: Propagation of error through a CNOT gate

Having $CNOT \equiv U = |0\rangle\langle 0| \otimes I_2 + |1\rangle\langle 1| \otimes X_2$ we write

$$\begin{aligned} UX_1 &= U(X_1 \otimes I_2) = U(X_1 \otimes I_2)U^\dagger U = U(X_1 \otimes I_2)\Big[|0\rangle\langle 0| \otimes I_2 + |1\rangle\langle 1| \otimes X_2\Big]^\dagger U \\ &= \Big[|0\rangle\langle 0| \otimes I_2 + |1\rangle\langle 1| \otimes X_2\Big]\Big[|1\rangle\langle 0| \otimes I_2 + |0\rangle\langle 1| \otimes X_2\Big]U \\ &= \Big[|0\rangle\langle 1| \otimes X_2 + |1\rangle\langle 0| \otimes X_2\Big]U \\ &= (X_1 \otimes I_2)(I_1 \otimes X_2)U \end{aligned} \tag{8.8}$$

This trick is also used throughout the calculations of FT quantum computation. In the following we employ the example of the CNOT gate to shed light on the condition of having a fault-tolerant circuit which is threshold condition on the probability of failure of a gate in the circuit.

First we construct a FT CNOT gate by replacing each qubit with a block of encoded qubits using e.g. Steane code and then replace the unencoded CNOT gate with its encoded version. The rest is applying error correction using the Steane code while the whole procedure must be done fault-tolerantly. Figure 8.5 illustrates this construction[27].

We want to show that the probability of introducing two or more errors to the out-

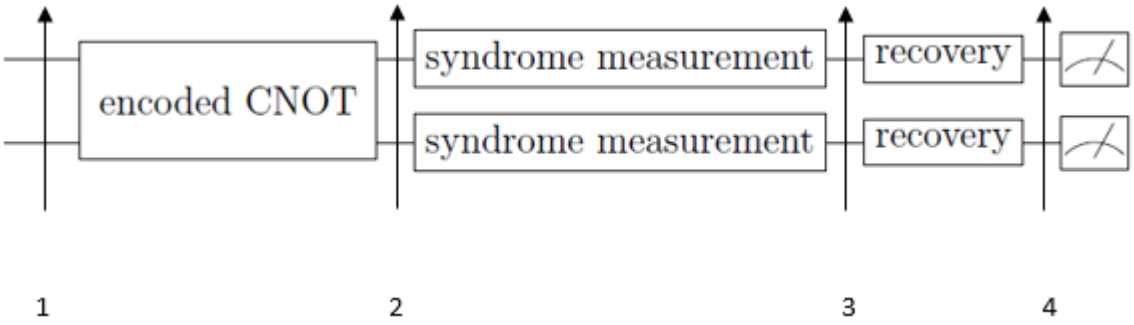

Figure 8.5: Fault-tolerant CNOT gate with implemented QEC

put from the fist block of encoded qubits is $O(p^2)$ which is an improvement over the probability of the same event when fault-tolerance is not implemented. In addition, we find a threshold for the probability p of failure of each component in the FT circuit. To do so we explore all the possible ways that lead to two or more errors in the output of the first block. Furthermore we do this analysis at four stages of the circuit, before the CNOT gate, after the CNOT gate, after the syndrome measurement, and after the recovery operation [see figure 8.5]. The possibilities are as below[27].

1. A pre-existing error may be introduced to each block from the state preparation or the error correction unit (syndrome measurement or recovery) after that. The error on the second block can propagate to the first block through the CNOT

gate whose probability is $c_0 p$. This is the probability of failure of each of the components before the CNOT gate and c_0 is the number of locations at which a failure may occur. For convenience we assume that the probability of a single pre-existing error entering the second block is $c_0 p$. Since these events are independent the probability of having two or more errors in the output of the first block is $c_0^2 p^2$. For the certain example of the Steane code we may have an estimate of the number of points of failure. The syndrome measurement includes measuring six generators each of which has approximately 10 components prone to failure. The recovery operation needs 7 components that add up to $c_0 \approx 70$ locations.

2. There may be a single pre-existing error in either the first or the second block of qubits and that the CNOT gate fails to work correctly. The probability of having two or more error in the output of the first block is $c_1 p$ where c_1 is the number of pairs of points at which a single failure may occur. Our estimate for the syndrome measurement and recovery operation in the stages prior to the CNOT gate also apply here however the number of points of failure increases to $2 \times 70 = 140$ due to the possible contribution of either of the blocks. There are also 7 components in the encoded CNOT gate that add up the total number of pairs of the failure points to $c_1 = 7 \times 140 \approx 10^3$.

3. Two failures may independently occur in the CNOT gate with probability $c_2 p^2$ where c_2 is the total number of pairs of points of failure. For the case of the

Steane code $c_2 \approx 10^2$.

4. There may be a failure in the CNOT and one in the syndrome measurement that comes afterwards with probability $c_3 p^2$ where $c_3 p^2 \approx 10^2$ for the Steane code.

5. Two or more failures may occur during syndrome measurement on each block independently with the probability $c_4 p^2$ where c_4 is the total number of pairs of points of failure in the syndrome measurement circuit. For the Steane code example $c_4 = 70 \times 70 \approx 5 \times 10^3$.

6. There may exist independent failures in the syndrome measurement and the recovery operations after the CNOT gate with probability $c_5 p^2$ where for the Steane code $c_5 = 70 \times 7 \approx 5 \times 10^2$.

7. There may exist two or more failures in the recovery operations of each block independently with probability $c_6 p^2$ where for the case of the Steane code $c_6 = 7^2 \approx 50$.

So the total number of places at which a single failure may occur is $c = c_0^2 + c_1 + c_2 + c_3 + c_4 + c_5 + c_6 \approx 10^4$ for the Steane code. The probability of having two or more errors in the output from the first block of encoded qubits is cp^2 and the probability of success of such a FT procedure is $1 - cp^2$ provided that the probability of failure of each component of the FT circuit is $p < \frac{1}{c} = 10^{-4}$ for the Steane code. In general, fault-tolerant quantum computation suppresses the errors due to the failure of the components from propagating to a large number of qubits and therefore the output data is

recoverable.

However the fault-tolerant quantum computations provides an improved error rate, yet there is another way to mitigate the effect of failure in the components of a FT circuit. Concatenation of codes is also used to improve the success probability of a FT computation. The idea of concatenated codes is encoding an input qubit at the first level of encoding and then encoding each qubit at the second level and doing it iteratively till the highest level. So if the probability of having unrecoverable error at the first level of encoding is cp^2 at the second level it will be $c(cp^2)^2 = c^3p^4$ and so on. At the highest level of encoding this probability is $\frac{(cp)^{2^k}}{c}$ provided that the failure probability of each component is $p < \frac{1}{c}$. Figure 8.6 shows the scheme of concatenation[28]. We must

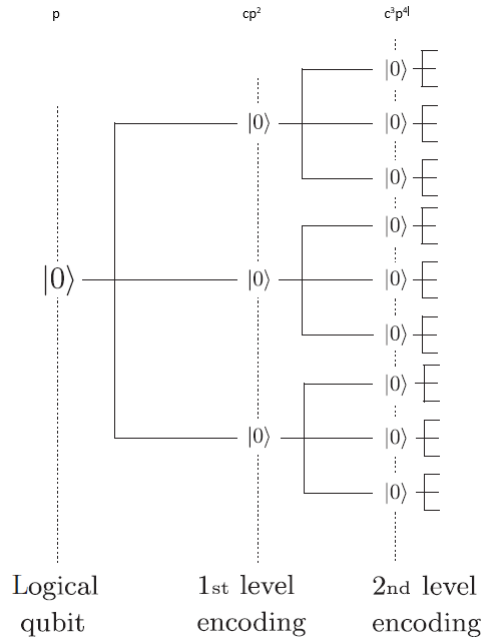

Figure 8.6: Levels of encoding in in a concatenated code

find a bound on the number of physical gates required to implement each FT gate for k levels of concatenations in such a way that the error rate decreases faster than the size of the circuit. Assume that the number of such gates is d^k with d being a constant. Concatenating a circuit of S gates for k levels we wish to have an overall error probability of ε. So the error rate of the fault-tolerant implementation of each gate is $\frac{\varepsilon}{S}$. So we wish to have

$$\frac{(cp)^{2^k}}{c} \leq \frac{\varepsilon}{S} \tag{8.9}$$

Taking logarithm of equation 8.9

$$2^k \leq \frac{log(\frac{S}{c\varepsilon})}{log(\frac{1}{cp})} \tag{8.10}$$

and using $2 = d^{\frac{1}{\log_2 d}}$ yields

$$d^k \leq \left(\frac{\log(\frac{S}{c\varepsilon})}{\log(\frac{1}{cp})}\right)^{\log_2 d} \tag{8.11}$$

The right-hand side of the inequality 8.11 is of the order $O\left(\log^m(\frac{S}{\varepsilon})\right)$ for a positive constant $m \geq 1$. So the size of the circuit (the number of gates) is bounded by

$$Sd^k = O\left(S\left(\log^m(\frac{S}{\varepsilon})\right)\right) \tag{8.12}$$

The threshold theorem[20] summarizes the implementation of fault-tolerant circuits using concatenation.

Theorem 6 *A quantum circuit containing S gates may be simulated with a probability of error at most ε using*

$$O\left(S\left(\log^m(\frac{S}{\varepsilon})\right)\right) \tag{8.13}$$

gates on hardware whose components introduce errors with probability p, provided p

is below some constant threshold, $p < p_{th}$.

As already mentioned fault tolerance not only applies to the gates in a circuit but also it includes other components such as the logical state preparation, measurement, transforming the qubits through the quantum wires, storing data in quantum memories, etc. In this chapter the basic ideas of implementing fault-tolerant circuits emphasising on the performance of gates were studied.

Chapter 9

CONCLUSION

In this survey a full review the early-stage quantum error correction has been done. It turns out that the CSS codes which are based on the idea of coding theory is contained in the larger family of the stabilizer codes. In fact, what distinguishes the subgroup of the CSS stabilizers from within the group of more general stabilizers is the form of their generators which is separated by their dependence on X or Z. Also, the stabilizer formalism has an edge over the framework of the CSS codes which is eventually needed to measure the error syndrome which is in essence touching the erred codeword. Instead the stabilizer formalism looks for the generators and measures them as observable which does not have any analogous in QEC. In this sense, measuring the generators of the CSS codes which are separated by X or Z is done more efficient than the non-CSS codes such as the five-qubit code.

Since the advent of the quantum error correcting codes in mid 90's several other codes have been proposed. Not all these codes are useful from a practical point of view while there are only a few proposals among these advancements which seem promising for the practical uses and in particular the implementation of quantum computers. This study with its extensive analysis of quantum error correction methods paves the way for further studies of the progresses in quantum error correction. One of these advancements is the surface code with a high threshold condition and therefore offers a more

robust code when subject to error. However one should note that the threshold analysis is architecture dependent and that different types of noise can be introduced to the qubits by different implementations of quantum computers and this affects the result of the threshold analysis substantially. Table 9.1 compares different architectures leading to different threshold conditions[40].

	Code	Threshold	Architecture
Aharanov, Ben-Or [19]	Steane code	$O(10^{-6})$	None
Gottesman [41]	Steane code	$O(10^{-4} - 10^{-6})$	None
Knill, Laflamme, Zurek [24]	Steane code	$O(10^{-6})$	None
Metodiev et al. [47]	Steane code	$O(10^{-4})$	Specific to Ion-Traps
Stephens, Fowler, Hollenberg [48]	Steane code	$O(10^{-6})$	Kane P:Si system [49]
Szkopek et al. [50]	Steane code	$O(10^{-7})$	General Nearest Neighbour Systems
Svore, DiVincenzo, Terhal [51]	Steane code	$O(10^{-5})$	General Nearest Neighbour Systems
Fowler et al. [52]	Steane code	$O(10^{-6})$	Superconducting qubits
Balesiefer, Kregor-Stickles, Oskin [53]	Steane code	$O(10^{-9})$	Ion-Traps
Wang et al. [54]	Surface code	$O(10^{-2} - 10^{-3})$	Quantum Dots, Diamond [55],[56]

Table 9.1: Different thresholds from different architectures

REFERENCES

[1] Bennett, CH and Brassard, G. BB84.(1984) *Proc. IEEE International Conference on Computers, Systems, and Signal Processing*, IEEE Press, Los Alamitos, Calif.

[2] Bernstein, E and Vazirani, U.(1993) *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*. ACM, New York,11.

[3] Deutsch, David,(1985) Quantum theory, the Church-Turing principle and the universal quantum computer, *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 400, 1818, 97–117.

[4] Deutsch, David and Jozsa, Richard,(1992) Rapid solution of problems by quantum computation, *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 439, 1907, 553–558.

[5] Grover, Lov K,(1996) A fast quantum mechanical algorithm for database search, *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 212–219.

[6] Grover, Lov K,(1997) Quantum mechanics helps in searching for a needle in a haystack, *Physical review letters*, 79, 2, 325.

[7] Grover, Lov K,(1998) Quantum computers can search rapidly by using almost any transformation, *Physical Review Letters*, 80, 19, 4329.

[8] Grover, Lov K,(1997) Quantum computers can search arbitrarily large databases by a single query, *Physical review letters*, 79, 23, 4709.

[9] Grover, Lov K,(1998) A framework for fast quantum mechanical algorithms, *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, 53–62.

[10] Shor, Peter W,(1994) Algorithms for quantum computation: Discrete logarithms and factoring, *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, 124–134.

[11] Shor, Peter W,(1994) Polynomial-Time Algorithm For Prime Factorization And Discrete Logarithms On A Quantum Computer: Proc, *35th Annual Symp. on the Foundations of Computer Science*, 124.

[12] Simon, Daniel R,(1997) On the power of quantum computation, *SIAM journal on computing*, 26, 5, 1474–1483.

[13] Lloyd, Seth,(1996) Universal quantum simulators, *Science*, 273, 5278, 1073.

[14] Steane, Andrew M,(1997) Active stabilization, quantum computation, and quantum state synthesis, *Physical Review Letters*, 78, 11, 2252.

[15] Steane, AM,(1998) Introduction to quantum error correction, *Philosophical Transactions-Royal Society of London Series A Mathematical Physical And Engineering Sciences*, 1739–1756.

[16] Steane, Andrew M,(1996) Error correcting codes in quantum theory, *Physical Review Letters*, 77, 5, 793.

[17] Shor, Peter W,(1996) Fault-tolerant quantum computation, *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, 56–65.

[18] DiVincenzo, David P and Shor, Peter W,(1996) Fault-tolerant error correction with efficient quantum codes, *Physical review letters*, 77, 15, 3260.

[19] Aharonov, Dorit and Ben-Or, Michael,(1997) Fault-tolerant quantum computation with constant error, *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, 176–188.

[20] Gottesman, Daniel,(1998) Theory of fault-tolerant quantum computation, *Physical Review A*, 57, 1, 127.

[21] Preskill, John,(1998) Fault-tolerant quantum computation, *Introduction to quantum computation and information*, 213.

[22] Knill, Emanuel and Laflamme, Raymond and Zurek, Wojciech H,(1998) Resilient quantum computation, *Science*, 279, 5349, 342–345.

[23] Knill, Emanuel and Laflamme, Raymond and Zurek, Wojciech H,(1998) Resilient quantum computation: error models and thresholds, *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 454, 1969, 365–384.

[24] Knill, Emanuel and Laflamme, Raymond and Zurek, Wojciech,(1996) Accuracy threshold for quantum computation. *Citeseer*,

[25] Kitaev, A Yu,(1997) Quantum error correction with imperfect gates, *Quantum Communication, Computing, and Measurement*, Springer, 181–188.

[26] Shor, Peter W,(1995) Scheme for reducing decoherence in quantum computer memory, *Physical review A*, 52, 4, R2493.

[27] Nielsen, Michael A and Chuang, Isaac L,(2010) *Quantum computation and quantum information*, Cambridge university press.

[28] Kaye, Phillip and Laflamme, Raymond and Mosca, Michele,(2007) *An introduction to quantum computing*, Oxford University Press.

[29] Lidar, Daniel A and Brun, Todd A,(2013) *Quantum error correction*, Cambridge University Press.

[30] Feynman, Richard P,(1982) Simulating physics with computers, *International journal of theoretical physics*, 21, 6, 467–488.

[31] Calderbank, A Robert and Shor, Peter W,(1996) Good quantum error-correcting codes exist, *Physical Review A*, 54, 2, 1098.

[32] Steane, Andrew M,(1999) Efficient fault-tolerant quantum computing, *Nature*, 399, 6732, 124–126.

[33] Knill, Emanuel and Laflamme, Raymond,(1997) Theory of quantum error-correcting codes, *Physical Review A*, 55, 2, 900.

[34] Steane, Andrew M,(1996) Simple quantum error-correcting codes, *Physical Review A*, 54, 6, 4741.

[35] Gottesman, Daniel,(2009) An introduction to quantum error correction and fault-tolerant quantum computation, *Quantum Information Science and Its Contributions to Mathematics*, 68, 13–60.

[36] Gottesman, Daniel,(2002) An introduction to quantum error correction, *Proceedings of Symposia in Applied Mathematics*, 58, 221–236.

[37] Kempe, Julia,(2006) Approaches to quantum error correction, *Quantum Decoherence*, Springer, 85–123.

[38] Barenco, Adriano and Bennett, Charles H and Cleve, Richard and DiVincenzo, David P and Margolus, Norman and Shor, Peter and Sleator, Tycho and Smolin,

John A and Weinfurter, Harald,(1995) Elementary gates for quantum computation, *Physical review A*, 52, 5, 3457.

[39] Raussendorf, Robert,(2012) Key ideas in quantum error correction, *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 370, 1975, 4541–4565.

[40] Devitt, Simon J and Munro, William J and Nemoto, Kae,(2013) Quantum error correction for beginners, *Reports on Progress in Physics*, 76, 7, 076001.

[41] Gottesman, Daniel,(1997) Stabilizer codes and quantum error correction, *arXiv preprint quant-ph/9705052*.

[42] Laflamme, Raymond and Miquel, Cesar and Paz, Juan Pablo and Zurek, Wojciech Hubert,(1996) Perfect quantum error correcting code, *Physical Review Letters*, 77, 1, 198.

[43] Gottesman, Daniel,(1996) Class of quantum error-correcting codes saturating the quantum Hamming bound, *Physical Review A*, 54, 3, 1862.

[44] Knill, Emanuel and Laflamme, Raymond.(1996) Concatenated quantum codes. *arXiv preprint quant-ph/9608012*.

[45] Ling, San and Xing, Chaoping,(2004) *Coding theory: a first course* Cambridge University Press.

[46] Golay, MJEE,(1954) Binary coding, *Transactions of the IRE Professional Group on Information Theory*, 4, 4, 23–28.

[47] T Metodiev, et al.(2004) Preliminary results on simulating a scalable fault-tolerant Ion-Trap system for quantum computation, *3rd Workshop on Non-Silicon Computing*.

[48] A. Stephens, A.G. Fowler, and L.C.L. Hollenberg,(2008) Universal Fault-Tolerant Computation on bilinear nearest neighbour arrays, *Quant. Inf. comp.*, 8, 330.

[49] B.E. Kane, (1998) A Silicon-Based nuclear spin Quantum Computer. *Nature* (London), 393:133,.

[50] T. Szkopek, P.O. Boykin, H. Fan, V.P. Roychowd- hury, E. Yablonovitch, G. Simms, M. Gyure, and B. Fong, (2006) Threshold Error Penalty for Fault-Tolerant Computation with Nearest Neighbour Communication, *IEEE Trans. Nano.*, 5:42.

[51] K.M. Svore, D.P. DiVincenzo, and B.M. Terhal,(2007) Noise Threshold for a Fault-Tolerant Two-Dimensional Lattice Architecture. *Quant. Inf. Comp.*, 7:297.

[52] A.G. Fowler, W.F. Thompson, Z. Yan, A.M, Stephens, B.L.T Plourde, and F. K. Wilhelm,((2007)) Long- range coupling and scalable architecture for superconduct- ing qubits. *Phys. Rev. B.*, 76:174507.

[53] S. Balensiefer, L. Kregor-Stickles, and M. Oskin.(2005) An Evaluation Framework and Instruction Set Architec- ture for Ion-Trap based Quantum Micro-architectures. *SIGARCH Comput. Archit. News*, 33:186.

[54] D.S. Wang, A.G. Fowler, A.M. Stephens, and L.C.L. Hollenberg, (2010) Threshold Error rates for the toric and surface codes. *Quant. Inf. Comp.*, 10:456.

[55] N. Cody Jones, R. Van Meter, A.G. Fowler, P.L. McMahon, J. Kim, T.D. Ladd, and Y. Yamamoto,(2012) A Lay- ered Architecture for Quantum Computing Using Quan- tum Dots. *Phys. Rev. X.*, 2:031007, .

[56] N.Y. Yao, L. Jiang, A.V. Gorshkov, P.C. Maurer, G. Giedke, J.I. Cirac, and M.D. Lukin,(2012) Scalable Architec- ture for a Room Temperature Solid-State Quantum Infor- mation Processor. *Nature Communications*, 3:800.