Introduction to Quantum Computing

Lectures by Leon Riesebos

Steven Oud
soud@pm.me

May 2, 2019

Chapter 1

Fundamentals: A One-Qubit World

Quantum computers are machines that rely on characteristically quantum phenomena, such as quantum interference and quantum entanglement, in order to perform computation.

— Artur Ekert

It is tempting to say that a quantum computer is one whose operation is governed by the laws of quantum mechanics. But since the laws of quantum mechanics govern the behavior of all physical phenomena, this temptation must be resisted. Your laptop operates under the laws of quantum mechanics, but it is not a quantum computer.

— N. David Mermin

1.1 The Qubit

A quantum bit (*qubit*) - like a classical bit - has a *state*. In classical bits this is 0 or 1. Two possible states for a qubit are $|0\rangle$ and $|1\rangle$ (denoted in *Dirac notation*), which correspond to the states 0 and 1 for a classical bit. The difference between bits and qubits is that a qubit can be in other states than $|0\rangle$ or $|1\rangle$, called *superpositions*. The state of a qubit can be denoted as following:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \tag{1.1}$$

where α and β are complex numbers. The special states $|0\rangle$ and $|1\rangle$ are known as *computational basis states*. We cannot examine a qubit to determine its quantum state. Quantum mechanics tells us that we can only obtain much more restricted information about the quantum state. When we measure a qubit we get either 0, with probability $|\alpha|^2$, or 1, with probability $|\beta|^2$. The sum of all probabilities is always equal to 1:

$$|\alpha|^2 + |\beta|^2 = 1. \tag{1.2}$$

For example, a qubit in the state

$$\frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle \tag{1.3}$$

gives 0 fifty percent of the time ($|1/\sqrt{2}|^2$), and 1 fifty percent of the time. This state is often denoted by $|+\rangle$.

1.2 The Bloch Sphere

The Bloch sphere is a geometrical representation of a qubit's state. It's a spherical coordinate system in which a quantum state can be described as

$$|\psi\rangle = e^{i\delta}\left(\cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle\right), \tag{1.4}$$

where δ, θ and ϕ are real numbers. Factor $e^{i\delta}$ is the global phase of the state. This factor does not influence measurement probabilities, since $\left|e^{i\delta}\right| = 1$. Therefore we can often omit it, allowing us to write

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle. \tag{1.5}$$

The numbers θ and ϕ define a point on the three-dimensional sphere (Figure 1.1). The Bloch sphere visualization can be very useful for describing single qubit operations. It is however limited in that there is no simple generalization of the Bloch sphere known for multiple qubits.

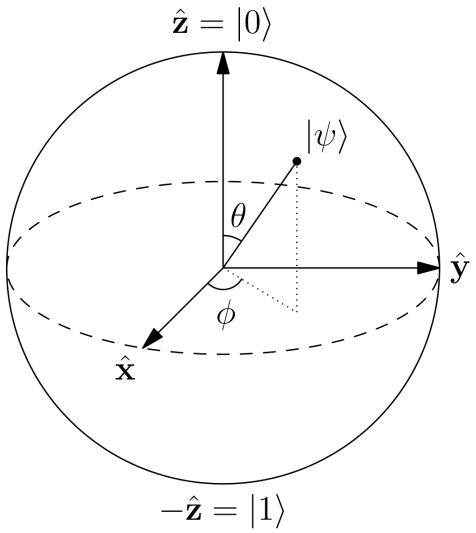


Figure 1.1: *Bloch sphere representation of a qubit.*

1.3 Single-Qubit Quantum Gates

Single-qubit quantum gates can be seen as counter-clockwise rotations on the Bloch sphere. These gates only have one limitation: they have to be *unitary*, that is $U^\dagger U = UU^\dagger = I$, where $U^\dagger$ is the conjugate transpose of U and I the identity operation. Therefore, any $2^n \times 2^n$ unitary operation is a valid gate which acts on n qubits. We will often use circuit notation to describe the transformation of state $|\psi\rangle$ to $|\psi'\rangle$ by a unitary operation U as following:

$$|\psi\rangle \; — \boxed{U} — \; |\psi'\rangle$$

Below are some notable single-qubit gates described and visualized.

1.3.1 Pauli Gates

The most simple quantum gates are the *Pauli gates* I, X, Y and Z. I is the identity gate, which does nothing. The other gates rotate π radians (180 degrees) around the X, Y or Z-axis. These gates are self-inverse, meaning $U^2 = I$.

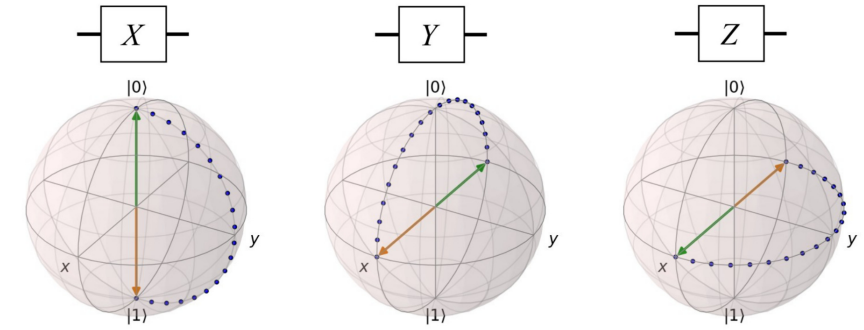


Figure 1.2: *Pauli gates X, Y and Z visualized on the Bloch sphere. The green vector is the starting position and the orange vector the final position.*

1.3.2 Hadamard Gate

The *Hadamard* (H) gate (Figure 1.3) maps the basis states $|0\rangle$ and $|1\rangle$ to superposition states with equal weight:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = |+\rangle \tag{1.6}$$

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) = |-\rangle\,. \tag{1.7}$$

It is the combination of two rotations, π radians about the Z-axis followed by $\pi/2$ radians about the Y-axis. This gate is sometimes described as a "square root of NOT" gate. The Hadamard gate belongs to the Clifford gates.

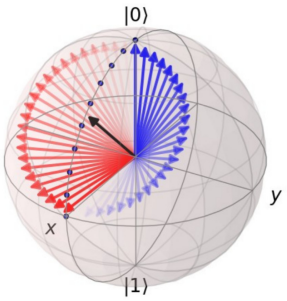


Figure 1.3: *Hadamard gate visualized on the Bloch sphere.*

1.3.3 Phase Gates

Rotation gates that rotate around the Z axis are considered *phase gates*. They rotate the phase of the $|1\rangle$ state by an angle θ and leave the $|0\rangle$ state unchanged:

$$\begin{aligned} R_z(\theta)\,|0\rangle &= |0\rangle \\ R_z(\theta)\,|1\rangle &= e^{i\theta}\,|1\rangle\,. \end{aligned} \tag{1.8}$$

The probability of measuring a $|0\rangle$ or $|1\rangle$ is unchanged after a phase gate, however it modifies the phase of the quantum state. A common phase gate is the S gate, where $\theta = \pi/2$ (Figure 1.4). The Pauli Z gate can be thought of as a phase gate where $\theta = \pi$, because $e^{i\pi} = -1$. Thus you could also think of the S gate as half a Pauli Z gate. Another common phase gate is the T gate, where $\theta = \pi/4$ (half an S gate).

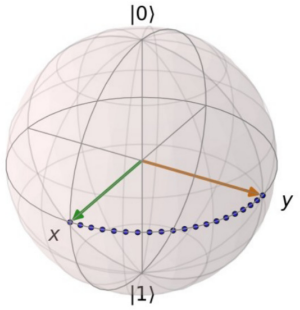


Figure 1.4: *S gate rotation visualized on the Bloch sphere.*

1.4 Measurement

Measuring a quantum state $|\psi\rangle$ *collapses* the quantum superposition to a computational basis state $|j\rangle$. The state $|\psi\rangle$ has "disappeared" and the state is left in $|0\rangle$ or $|1\rangle$. The probabilistic result of a measurement of a quantum state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ can be calculated based on

the probability amplitudes:

$$P(|0\rangle) = |\alpha|^2 \tag{1.9}$$

$$P(|1\rangle) = |\beta|^2 . \tag{1.10}$$

Consider the following simple single-qubit circuit:

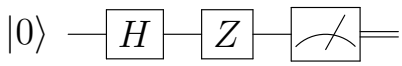


Figure 1.5: *A simple quantum circuit. The output of measuring a qubit is a classical bit, which is distinguished from a qubit by drawing a double-line wire. A visualization of this circuit on the Bloch sphere can be seen in Figure 1.6.*

We start with computing $H|0\rangle = |+\rangle$, followed by $Z|+\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$ (or $|-\rangle$). Finally we measure, giving us a computational basis state $|j\rangle$ and collapsing the state. Which state we will see is not determined in advance; the only thing we can say is that we will see $|j\rangle$ with probability $|a_j|^2$:

$$P(|0\rangle) = \left|\frac{1}{\sqrt{2}}\right|^2 = \frac{1}{2} \tag{1.11}$$

$$P(|1\rangle) = \left|\frac{-1}{\sqrt{2}}\right|^2 = \frac{1}{2}. \tag{1.12}$$

Giving us equal probabilities of our state being measured as $|0\rangle$ or $|1\rangle$.

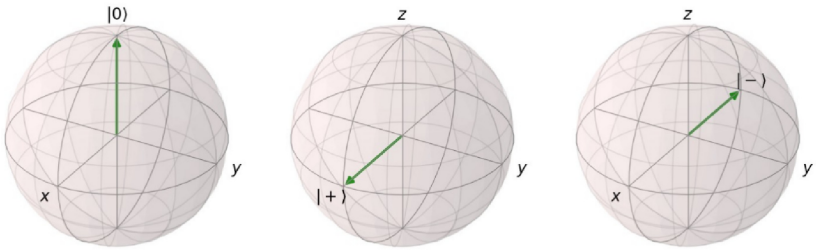


Figure 1.6: *States of the qubit throughout the circuit from left to right:* $|0\rangle \rightarrow H|0\rangle \rightarrow ZH|0\rangle$.

1.5 Vector Notation

Earlier we showed that we can represent a quantum state as

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle . \tag{1.13}$$

The quantum states $|\psi\rangle$, $|0\rangle$ and $|1\rangle$ in the formula above are vectors. We define $|0\rangle$ and $|1\rangle$, the computational basis states, as following:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}; \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{1.14}$$

A quantum state has to be a normalized vector. In general, a qubit's state is a unit vector in a two-dimensional complex vector space. A n qubit state has a 2^n dimensional *Hilbert space*. We can consider a quantum state to be a linear combination of the computational basis states:

$$|\psi\rangle = \alpha \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \beta \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha \\ \beta \end{pmatrix}. \tag{1.15}$$

Single-qubit gates can then be represented by 2×2 unitary matrices:

$$I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}; \qquad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}; \qquad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix};$$

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}; \qquad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}; \qquad H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}.$$

Then a computation like $X|0\rangle$ can be calculated by matrix vector multiplication:

$$X|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{1.16}$$

And we find that $X|0\rangle = |1\rangle$. Or, more general:

$$X \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \beta \\ \alpha \end{pmatrix}. \tag{1.17}$$

You can combine quantum gates by multiplying their matrices. For example, let's verify that H is self-inverse by applying it to itself:

$$H^2 = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I. \tag{1.18}$$

The result is the identity gate, showing that the inverse of H is indeed itself. It is also a *Hermitian matrix*, because it is equal to its own conjugate transpose: $H = H^\dagger$.

Chapter 2

Fundamentals: A Multi-Qubit World

We can represent multiple qubit states using the *Kronecker product*. Let A be a $m \times n$ matrix and B a $p \times q$ matrix, then

$$A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mn}B \end{pmatrix}, \tag{2.1}$$

resulting in a $mp \times nq$ matrix. We can then represent the two qubit state $|00\rangle$ as

$$|00\rangle = |0\rangle \otimes |0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ 0 \begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{2.2}$$

Throughout this document we will use different kinds of notation for multiple qubits depending on context, all of which are equivalent: $|00\rangle = |0\rangle|0\rangle = |0\rangle \otimes |0\rangle$. We can write a two qubit state as following in Dirac notation:

$$|ab\rangle = |a\rangle \otimes |b\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle\,. \tag{2.3}$$

More generally, we say that a linear combination $\sum_j \alpha_j\,|\psi_j\rangle$ is a quantum state with states $|\psi_j\rangle$ and amplitude α_j for state $|\psi_j\rangle$.

Note that the state vector of two qubits is twice as big as the state vector for one qubit. This is where some of the potential power of quantum computers comes from. With n qubits you can represent 2^n states - the state space grows exponentially with the number of qubits, unlike classical bits. For multi-qubit states the rule remains that the state vector has to be normalized. For a n qubit state:

$$\sum_{j=0}^{2^n-1} |\alpha_j|^2 = 1. \tag{2.4}$$

2.1 Quantum State Evolution

Doing single-qubit operations on a multi-qubit state is possible by combining the identity and our single-qubit gate. Say we want to compute $H_1|0_0 0_1\rangle$. That is, put the second qubit through a Hadamard gate.[1] To do so, the gate matrix's column width has to be equal to the

[1] Note that when I say "second" qubit, I'm talking about the most right, or least significant qubit. I will number them for this example but assume it from here on out.

quantum state vector's dimension. We can achieve this by taking the Kronecker product of the identity matrix and our single-qubit matrix (in this case H):

$$H_1 \left|0_0 0_1\right\rangle = \left(I_0 \otimes H_1\right) \left|0_0 0_1\right\rangle . \tag{2.5}$$

Writing it out:

$$I_0 \otimes H_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{2.6}$$

$$= \frac{1}{\sqrt{2}} \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \right] \tag{2.7}$$

$$= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix} . \tag{2.8}$$

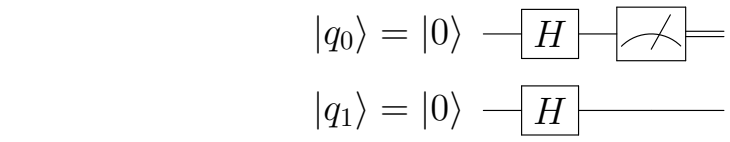
Then we can put our $|00\rangle$ state through it:

$$\frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} , \tag{2.9}$$

which can also be written in Dirac notation as $(|00\rangle + |01\rangle)/\sqrt{2}$.

2.2 Partial Measurement

Say we measure the qubit $|q_0\rangle$ in the following circuit:



First, we put both qubits in our state $|00\rangle$ through a Hadamard gate. This puts it in the state

$$\left(H \otimes H\right) |00\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle). \tag{2.10}$$

When we measure $|q_0\rangle$ we have a 50/50 probability of getting a 0 or 1. Measuring $|q_0\rangle$ collapses the state to one of the following states:

$$|\psi\rangle = \begin{cases} \frac{1}{\sqrt{2}}(|\underline{0}0\rangle + |\underline{0}1\rangle) \text{ if } M(q_0) = 0 \\ \frac{1}{\sqrt{2}}(|\underline{1}0\rangle + |\underline{1}1\rangle) \text{ if } M(q_0) = 1 \end{cases} \tag{2.11}$$

We have two possible states for q_0 after measurement: 0 or 1. Qubit $|q_1\rangle$ will stay in superposition because we haven't measured it. Notice how the first qubits (underlined) in both states are the same. This makes sense, we've measured that one so we're certain of its state.

2.3 Common Two-Qubit Gates

2.3.1 CNOT Gate

The quantum gate controlled-NOT (CNOT, sometimes called controlled-X) is comparable to a classical computer's XOR, but it's reversible. This gate has two input qubits, the *control* qubit and *target* qubit. If the control qubit is set to 0, then the target qubit is left alone. If the control qubit is set to 1, then the target qubit is flipped. The circuit representation for CNOT can be seen in Figure 2.1. Qubit $|q_0\rangle$ represents the control qubit and $|q_1\rangle$ represents the target qubit. It's essentially a Pauli X gate with a control qubit. CNOT is Hermitian and belongs to the Clifford gates.

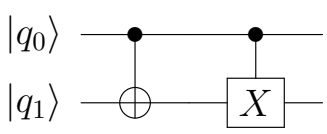
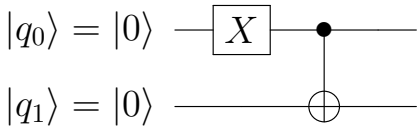


Figure 2.1: *Two different circuit representations of* CNOT.

$$U_{CX} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Figure 2.2: *Matrix representation of* CNOT.

Let's look at an example of a CNOT gate. Consider the following circuit:



First we put $|q_0\rangle$ (the control qubit) in the $|1\rangle$ state by applying an X gate, giving us the state $|10\rangle$. Then we apply the CNOT gate, giving

$$\text{CNOT}\,|10\rangle = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = |11\rangle\,. \tag{2.12}$$

The target qubit was flipped because the control qubit was set to $|1\rangle$, giving us $|11\rangle$.

2.3.2 CZ Gate

CZ, or the controlled-Z gate, acts in a similar way to other controlled gates. That is, do the operation on the target qubit if the control qubit is $|1\rangle$, otherwise do nothing. In CZ the operation is the Pauli Z gate. CZ is also Hermitian and belongs to the Clifford gates.

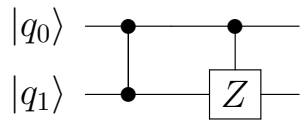


Figure 2.3: *Two different circuit representations of* CZ.

2.3.3 Controlled Gates

Controlled gates act on two or more qubits, where one or more qubits act as control for some operation. Generally, if U is a gate that operates on single qubits with the following matrix representation:

$$U = \begin{pmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{pmatrix}, \tag{2.13}$$

then the controlled-U gate is a gate that operates on two qubits where the first qubit serves as control. The general matrix representation of the controlled-U then, if qubit 0 is the control and qubit 1 is the target, looks as following:

$$C_U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{pmatrix}. \tag{2.14}$$

2.4 Toffoli Gate

The Toffoli gate has three inputs and outputs (Figure 2.4), where two of the input qubits act as control bits. The third qubit is the target bit which is flipped if both control qubits are set to $|1\rangle$, otherwise it's left alone. For example, applying the Toffoli gate to the state $|110\rangle$ flips the third qubit, resulting in the state $|111\rangle$.

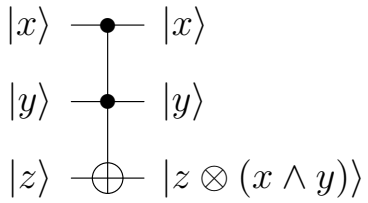


Figure 2.4: *Circuit representation of the Toffoli gate, where $\otimes$ is binary sum (*XOR*).*

The Toffoli, or controlled-controlled-X (CCX) gate can be represented by a 8×8 matrix:

$$U_{CCX} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}. \tag{2.15}$$

2.5 Universal Gate Sets

In classical systems the NAND gate is a universal gate, meaning that any other gate can be represented as a combination of NAND gates. In quantum computing there exist universal gate sets. A universal gate set requires the full Clifford and Pauli groups, and one or more non-Clifford gates.

We've seen the Pauli gates in Section 1.3.1. On their own, Pauli gates have no interesting computational capabilities. The Clifford gates we've seen are H, S, CNOT and CZ. Clifford gates introduce the quantum phenomena superposition and entanglement. The Pauli and Clifford gates can be simulated efficiently by classical computers (*Gottesman-Knill theorem*) - showing no increase in efficiency over classical computers.

The non-Clifford gates, which are required for universal quantum computing, cannot be simulated efficiently and are exponentially hard to simulate. Some non-Clifford gates are Toffoli, T and the rotation gates R_x, R_y and R_z (which do arbitrary rotations around the axes). One universal gate set that allows universal quantum computing is $\{H, T, \text{CNOT}\}$.

2.6 Entanglement

Two qubits are *entangled* if and only if the state of those two qubits can *not be expressed as two individual states* (non-separable). Let's first take a look at a separable, or non-entangled state

$$|\psi\rangle = \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle). \tag{2.16}$$

This state can be separated and expressed as the following two individual states:

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle). \tag{2.17}$$

However, consider the state

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle). \tag{2.18}$$

This is an entangled state, it cannot be expressed as two individual states. We say two qubits are entangled if they have *nonzero concurrence*. The concurrence of a state can be calculated as following:

$$C(|\psi\rangle) = 2|\alpha_{00}\alpha_{11} - \alpha_{01}\alpha_{10}|. \tag{2.19}$$

We can check if our entangled state (2.18) is indeed entangled:

$$C\left(\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)\right) = 2\left|\frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}\right)\right| = 1. \tag{2.20}$$

It has a non-zero concurrence, so we can say it's entangled. How about the non-entangled state in 2.16?

$$C\left(\frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle)\right) = 2\left|\frac{1}{2}\left(\frac{1}{2}\right) - \frac{1}{2}\left(\frac{1}{2}\right)\right| = 0. \tag{2.21}$$

A concurrence of 0, so it is not entangled.

2.7 The Bell States

The *Bell states* are four maximally entangled quantum states of two qubits:

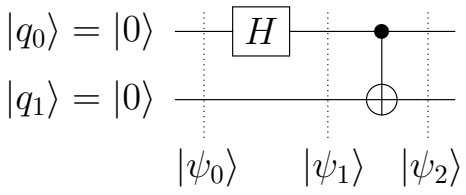
$$\left|\Phi^+\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{2.22}$$

$$\left|\Phi^-\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \tag{2.23}$$

$$\left|\Psi^+\right\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{2.24}$$

$$\left|\Psi^-\right\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \tag{2.25}$$

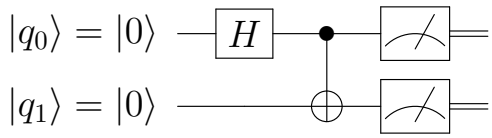
We can create a Bell state with the following circuit:



We start with our state $|00\rangle$ at $|\psi_0\rangle$. We put $|q_0\rangle$ through a Hadamard gate, giving us the state $(|00\rangle + |10\rangle)/\sqrt{2}$ at $|\psi_1\rangle$. Finally we CNOT that state giving us the final Bell state $|\Phi^+\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ at $|\psi_2\rangle$.

The significance of Bell states becomes apparent when we start measuring qubits of a Bell state. Take the circuit we used before to create the Bell state $(|00\rangle + |11\rangle)/\sqrt{2}$ and

measure both qubits.



You will find that the measurement results are correlated:

$$M(q_1) = 0 \text{ if } M(q_0) = 0 \tag{2.26}$$
$$M(q_1) = 1 \text{ if } M(q_0) = 1 \tag{2.27}$$

If you measure $|q_0\rangle$ to be 0, $|q_1\rangle$ will also be 0 and vice versa. Note that in our example entangled state they correlate as being equal, but for the entangled state $(|01\rangle + |10\rangle)/\sqrt{2}$ they correlate as being opposites (measuring $|q_0\rangle$ as 0 means $|q_1\rangle$ will be 1).

2.8 Greenberger-Horne-Zeilinger State

A Greenberger-Horne-Zeilinger (GHZ) state is a certain type of entangled state. It is a $M > 2$ system state:

$$|\text{GHZ}\rangle = \frac{|0\rangle^{\otimes M} + |1\rangle^{\otimes M}}{\sqrt{2}}, \tag{2.28}$$

where $|j\rangle^{\otimes M}$ means the Kronecker product with itself M times. For example $M = 3$:

$$|\text{GHZ}\rangle = \frac{|000\rangle + |111\rangle}{\sqrt{2}}. \tag{2.29}$$

GHZ states are used for example in cryptography for secret sharing.

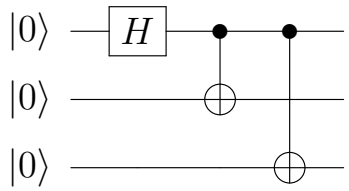


Figure 2.5: *Circuit creating a three-qubit GHZ state.*

2.9 Calculating Parity

Parity checking is one of the simplest forms of error detecting code. It tells us if the number of ones in a set of bits is even or odd. For example, 001 has a parity of 1 (odd) and 110 has a parity of 0 (even). We introduce a quantum algorithm for calculating the parity of an n-qubit

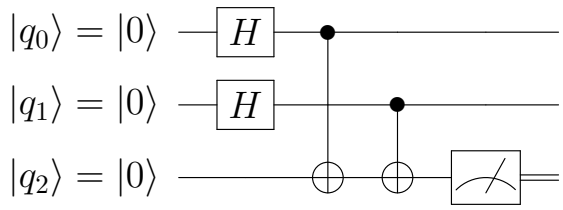


Figure 2.6: *Circuit for calculating parity:* $\text{CNOT}_{1,2}\text{CNOT}_{0,2}H_1H_0\,|000\rangle$.

state. For this example we'll determine the parity of a two-qubit state. Consider the circuit in Figure 2.6. We will calculate the parity of $|q_0\rangle|q_1\rangle$ with $|q_2\rangle$.

We start with putting $|q_0\rangle$ and $|q_1\rangle$ in an arbitrary superposition. In this case we apply a Hadamard gate to both qubits, giving us the state

$$|\psi\rangle = \frac{1}{2}(|000\rangle + |010\rangle + |100\rangle + |110\rangle). \tag{2.30}$$

Then we apply a $\text{CNOT}_{0,2}$ operation, giving

$$|\psi\rangle = \frac{1}{2}(|000\rangle + |010\rangle + |101\rangle + |111\rangle), \tag{2.31}$$

and a $\text{CNOT}_{1,2}$:

$$|\psi\rangle = \frac{1}{2}(|000\rangle + |011\rangle + |101\rangle + |110\rangle). \tag{2.32}$$

If you look closely at this state, you can see that $|q_2\rangle$ corresponds to the parity of $|q_0\rangle|q_1\rangle$. We have essentially calculated the parities of all possible states of $|q_0\rangle|q_1\rangle$ in parallel. However, we cannot observe a quantum state to extract all the possible states' information. We are limited to measuring one outcome. When we measure $|q_2\rangle$ the state partially collapses leaving only the states with parity q_2. Let's take a look at the possible states after measuring $|q_2\rangle$:

$$|\psi\rangle = \begin{cases} \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \otimes |0\rangle \text{ if } M(q_2) = 0 \\ \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \otimes |1\rangle \text{ if } M(q_2) = 1 \end{cases} \tag{2.33}$$

Measuring a 0 leaves us with even parity states and measuring a 1 leaves us with odd parity states. $|00\rangle$ and $|11\rangle$ have even parity ($q_2 = 0$), $|01\rangle$ and $|10\rangle$ have odd parity ($q_2 = 1$).

2.10 Quantum Teleportation

Quantum teleportation it a technique for moving arbitrary quantum states around. It uses an *EPR pair* (a pair of qubits that is in a Bell state) that is shared between the sender and receiver. Note that it is impossible to clone a qubit state. This is referred to as the *no-cloning theorem*. Quantum teleportation works as following: Alice and Bob generate an EPR pair

and both take one qubit before they get separated. Alice wants to deliver a qubit $|\phi\rangle$, whose state is unknown, to Bob. Alice interacts the qubit $|\phi\rangle$ with her half of the EPR pair, and then measures the two qubits in her possession. At this point, Alice's qubits are in one of the four classical states 00, 01, 10 or 11. She sends this information to Bob. Bob then performs one of four operations on his half of the EPR pair, recovering Alice's quantum state $|\phi\rangle$.

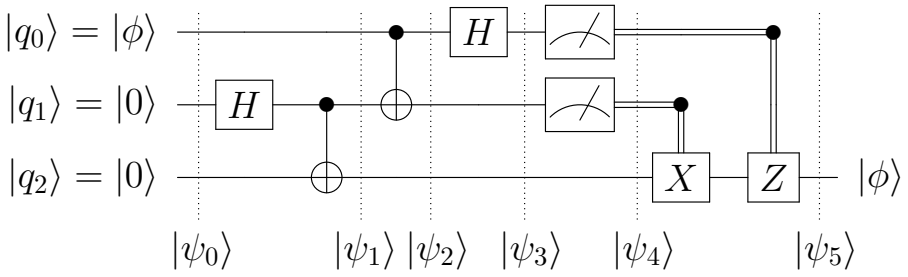


Figure 2.7: *Quantum circuit teleporting a quantum state $|\phi\rangle$.*

The circuit in Figure 2.7 teleports the unknown state $|\phi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$. We start at $|\psi_0\rangle$ with the state $|\phi\rangle|00\rangle$. Then we create an EPR pair with $|q_1\rangle|q_2\rangle$, where $|q_1\rangle$ belongs to Alice and $|q_2\rangle$ to Bob. This gives us the following state at $|\psi_1\rangle$:

$$|\psi_1\rangle = |\phi\rangle\,\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle), \tag{2.34}$$

which we can rewrite as following

$$|\psi_1\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \otimes \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{2.35}$$

$$= \frac{1}{\sqrt{2}}\Big(\alpha\,|0\rangle\,(|00\rangle + |11\rangle) + \beta\,|1\rangle\,(|00\rangle + |11\rangle)\Big). \tag{2.36}$$

Alice then sends her qubits through a $\text{CNOT}_{0,1}$ gate, obtaining

$$|\psi_2\rangle = \frac{1}{\sqrt{2}}\Big(\alpha\,|0\rangle\,(|00\rangle + |11\rangle) + \beta\,|1\rangle\,(|10\rangle + |01\rangle)\Big). \tag{2.37}$$

She then sends $|q_0\rangle$ through a Hadamard gate, giving

$$|\psi_3\rangle = \frac{1}{2}\Big(\alpha(|0\rangle + |1\rangle)(|00\rangle + |11\rangle) + \beta(|0\rangle - |1\rangle)(|10\rangle + |01\rangle)\Big). \tag{2.38}$$

This state can be rewritten in the following way:

$$\begin{aligned}|\psi_3\rangle = \frac{1}{2}\Big(&|00\rangle\,(\alpha\,|0\rangle + \beta|1\rangle) + |01\rangle\,(\alpha\,|1\rangle + \beta\,|0\rangle)\\ &+ |10\rangle\,(\alpha\,|0\rangle - \beta\,|1\rangle) + |11\rangle\,(\alpha\,|1\rangle - \beta\,|0\rangle)\Big).\end{aligned} \tag{2.39}$$

This expression breaks down in four terms. The first term has Alice's qubits in state $|00\rangle$ and Bob's qubit in state $\alpha|0\rangle + \beta|1\rangle$, which is our original state $|\phi\rangle$. So in the case that Alice measures $M(q_0q_1) = 00$ at $|\psi_4\rangle$, Bob's qubit will be in state $|\phi\rangle$. Depending on Alice's measurement at $|\psi_4\rangle$, Bob may have to "fix" his state to recover $|\phi\rangle$ by applying the appropriate gate(s). For example, if Alice measures 00, Bob doesn't have to do anything. If Alice measures 01, Bob can fix his state by applying an X gate. If Alice measures 10, Bob can fix his state by applying a Z gate. And if Alice measures 11, Bob can fix his state by first applying an X gate and then a Z gate. After fixing his state, Bob ends up with Alice's state $|\phi\rangle$ at $|\psi_5\rangle$.

Note that quantum teleportation does *not* allow for faster than light communication. This is because Bob needs to be told of the result of Alice's measurements through a classical channel in order to complete the teleportation. Also note that we did not clone the quantum state $|\phi\rangle$, we merely moved it. Teleporting a state depends on the measurement and thus collapsing of the original state $|\phi\rangle$, so it never allows for cloning.

Chapter 3

Quantum Algorithms

Quantum computers promise to solve problems intractable by classical computers. One application is quantum simulation. For example: simulating chemical processes, which are actually quantum processes. As Richard Feynman said: "*Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical.*" Other possible applications include optimization, cryptography and machine learning. There are a lot of efforts in trying to apply quantum computing, but many remain skeptic. We should assume no guarantees in the applications of quantum computers, but continue exploring the possibilities.

When a quantum computer can calculate something efficiently that a classical computer cannot calculate efficiently, we will have reached *quantum supremacy*. The boundary of reaching this is around 50 qubits, when it becomes impossible to simulate a quantum computer. We have not yet reached quantum supremacy, but we are getting close to it. There have been found quantum algorithms which provide a speedup over classical algorithms, like Shor's algorithm for factoring integers and the Deutch-Jozsa algorithm for determining if a boolean function is balanced or unbalanced.

3.1 Quantum Arithmetic

Any classical circuit can be simulated by a quantum circuit. We can show this by creating quantum versions of classical arithmetic functions. One problem with simulating classical gates as quantum gates is that most of them are non-reversible. In quantum computing, it is required for gates to be reversible. This is often solved by adding an output qubit for every input qubit and an extra result qubit (Figure 3.1). We can create a quantum half adder by

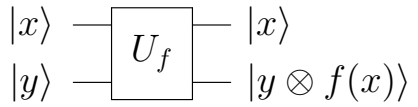


Figure 3.1: *Create output qubits for every input qubit in $|x\rangle$ and store the result of $f(x)$ in $|y\rangle$. This transformation can be described as $U_f\,|x\rangle|y\rangle = |x\rangle|y \otimes f(x)\rangle$. The $\otimes$ symbol in this context means the binary sum (*XOR*).*

creating sum and carry circuits, just like in classical computers. The truth table of the sum function $f_s(x)$ can be found in Figure 3.2. We can implement this as a quantum circuit using CNOT gates as seen in Figure 3.3.

x_0x_1	$f_s(x)$
00	0
01	1
10	1
11	0

Figure 3.2: *Bit sum truth table.*

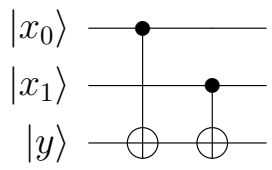


Figure 3.3: *Quantum circuit for summing two bits using two* CNOT *gates.*

To fully add two bits we need to account for the carry. The carry function $f_c(x)$ is 1 if and only if both inputs are 1 (Figure 3.4). This truth table corresponds directly to the Toffoli gate (Figure 3.5).

x_0x_1	$f_c(x)$
00	0
01	0
10	0
11	1

Figure 3.4: *Bit carry truth table.*

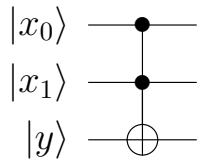


Figure 3.5: *Quantum circuit for taking the carry of two bits implemented using a Toffoli gate.*

We can combine our sum and carry circuit into one, creating a quantum half adder.

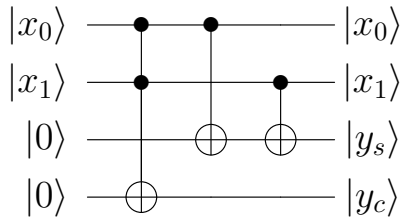


Figure 3.6: *Quantum half adder. The sum ends up in $|y_s\rangle$ and the carry in $|y_c\rangle$.*

Let's check if this circuit works. Say we want to add $|x_0\rangle = |1\rangle$ and $|x_1\rangle = |1\rangle$. We would expect to get 0 as sum and 1 as carry. Our initial state is $|1100\rangle$. We apply a Toffoli gate, setting the carry qubit to 1: $|1101\rangle$. Then we set the sum qubit by doing a $\text{CNOT}_{0,2}$ and a $\text{CNOT}_{1,2}$ ending up with the state $|1101\rangle$. Measuring the sum and carry qubits gives $M(y_s) = 0$ and $M(y_c) = 1$, giving us what we expected and thus having done a quantum half addition of two qubits.

3.2 Deutch-Jozsa Algorithm

We have shown that we can simulate classical circuits on a quantum computer. This itself isn't very impressive however, we could already do that on a classical computer. In this

section we will describe an algorithm with quantum speedup. Consider the four single-bit operations identity, NOT, reset and set.

Identity
$f_i(x) = x$

x	$f_i(x)$
0	0
1	1

NOT
$f_n(x) = \neg x$

x	$f_n(x)$
0	1
1	0

Reset
$f_r(x) = 0$

x	$f_r(x)$
0	0
1	0

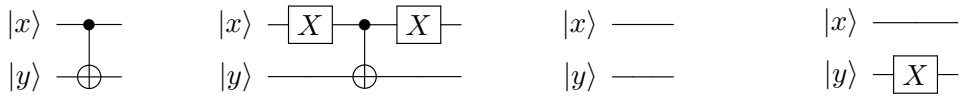
Set
$f_s(x) = 1$

x	$f_s(x)$
0	1
1	1

Figure 3.7: *Truth tables for the four single-bit operations.*

Say we are faced with the following problem: we have a "black box" with one of the four one-bit functions, but we're not told which one. Determine if the function is *balanced* or *unbalanced*. A balanced function is a function which returns the same amount of 0 and 1s it received as input, like identity and NOT. Unbalanced functions are functions which return a constant value, like reset and set. This is known as *Deutch's problem* and is one of the first examples of a problem where there exists a quantum algorithm that is exponentially faster than any deterministic classical algorithm.

On a classical computer, to determine if the black box is balanced or unbalanced, we need to do two function calls. On a quantum computer, using the *Deutch-Jozsa* algorithm, we can figure it out using one call to the black box. To use this algorithm for Deutch's problem we need to translate the four single-bit operations to the quantum world. The quantum circuits corresponding to each single-bit operation can be found in Figure 3.8.



Identity $f_i(x) = x$ ***NOT*** $f_n(x) = \neg x$ ***Reset*** $f_r(x) = 0$ ***Set*** $f_s(x) = 1$

Figure 3.8: *Quantum circuit representation of the four single-bit operations.*

With these definitions we can start using the Deutch-Jozsa algorithm to efficiently determine what kind of function our black box is. The circuit to do so is described in Figure 3.9. In this circuit, U_f refers to our black box function, or as it is sometimes also called, *oracle*.

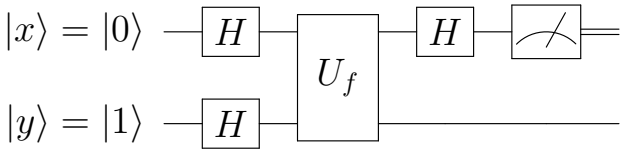


Figure 3.9: *Quantum circuit to determine if f is a balanced or unbalanced function using the Deutch-Jozsa algorithm. U_f is the quantum circuit which transforms $|x\rangle|y\rangle$ to $|x\rangle|y \otimes f(x)\rangle$.*

We can try it out by trying to determine if the identity function $f_i(x) = x$ is balanced or unbalanced. We start with putting $|x\rangle$ and $|y\rangle$ through a Hadamard gate:

$$|\psi\rangle = \frac{1}{2}(|00\rangle - |01\rangle + |10\rangle - |11\rangle). \tag{3.1}$$

We evaluate our identity function using by applying U_f:

$$\begin{aligned} |\psi\rangle = \frac{1}{2}\big(&|0\rangle|0 \otimes f_i(0)\rangle - |0\rangle|1 \otimes f_i(0)\rangle \\ &+ |1\rangle|0 \otimes f_i(1)\rangle - |1\rangle|1 \otimes f_i(1)\rangle \big). \end{aligned} \tag{3.2}$$

This state contains information about both $f_i(0)$ and $f_i(1)$! We have essentially evaluated two values of $f_i(x)$ simultaneously, a feature known as *quantum parallelism*. Actually applying the function $f_i(x)$, which as we can see in Figure 3.8 is just a $\text{CNOT}_{0,1}$, gives us the state

$$|\psi\rangle = \frac{1}{2}(|00\rangle - |01\rangle + |11\rangle - |10\rangle). \tag{3.3}$$

Finally we apply another Hadamard gate on $|x\rangle$:

$$\begin{aligned} |\psi\rangle &= \frac{1}{2\sqrt{2}}(|00\rangle + |10\rangle - |01\rangle - |11\rangle \\ &\qquad + |01\rangle - |11\rangle - |00\rangle + |10\rangle) && (3.4) \\ &= \frac{1}{2\sqrt{2}}(|10\rangle + |10\rangle - |11\rangle - |11\rangle) && (3.5) \\ &= \frac{1}{\sqrt{2}}(|10\rangle - |11\rangle). && (3.6) \end{aligned}$$

We end up with the final state $(|10\rangle - |11\rangle)/\sqrt{2}$ before measurement. We can see from this state that $M(x) = 1$. Curiously, the following measurement outcomes apply:

$$\begin{aligned} |0_x\rangle \otimes \frac{1}{\sqrt{2}}(|0_y\rangle - |1_y\rangle) \text{ if } f(0) = f(1) \\ |1_x\rangle \otimes \frac{1}{\sqrt{2}}(|0_y\rangle - |1_y\rangle) \text{ if } f(0) \neq f(1) \end{aligned} \tag{3.7}$$

That is, we get $M(x) = 1$ if $f(0) \neq f(1)$, which means f is a balanced function. Likewise, we get $M(x) = 0$ if $f(0) = f(1)$, which means f is an unbalanced function. Since we got a measurement of 1 for f_i, and given the fact that the identity operation is balanced, we can say that we effectively determined that f_i is a balanced function with one call to the black box.

3.3 Quantum Fourier Transform

An important operation in classical computing and quantum computing as we will see is the *Fourier transform*. The Fourier transform transforms a function $f(t)$ in the time domain to

another function $F(x)$ in the frequency domain. The opposite (transforming a function in the frequency domain to a function in the time domain) can be achieved by applying the *inverse* Fourier transform. Consider the sinusoidal function $f(t) = \cos(t)$. Figure 3.10 graphs this function in the time domain, with time on the x axis and amplitude on the y axis. Applying the Fourier transform on this function gives us a function of the frequency domain, as seen in Figure 3.11.

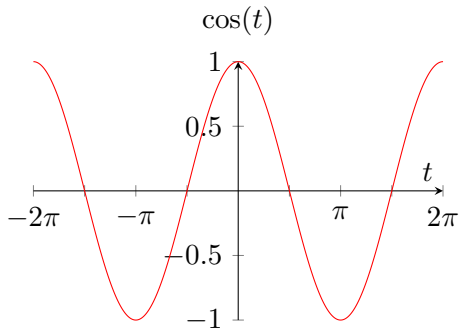


Figure 3.10: *Time domain plot of* $\cos(t)$.

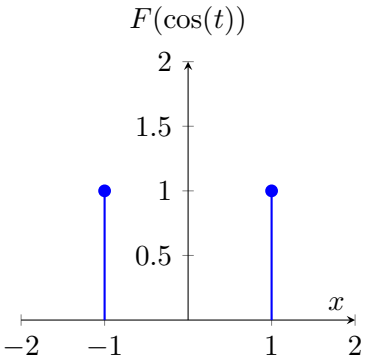


Figure 3.11: *Frequency domain plot of* $F(\cos(t))$.

The Fourier transform is a very useful tool in computer science, used for example in signal processing. The signal shown in Figure 3.10 is a continuous signal, meaning it is assumed to extend to infinity. Computers however can only deal with finite, non-continuous signals. This means we have to discretize the continuous function into discrete counterparts as seen in Figure 3.12. We can then transform the discrete signal using the *discrete Fourier transform*.

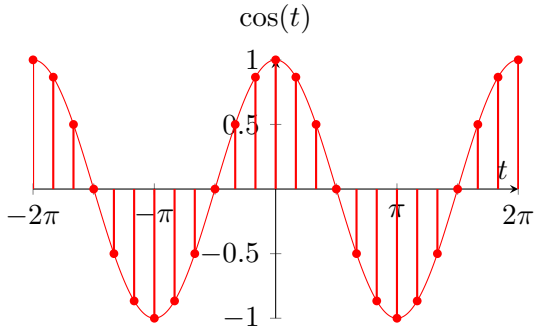


Figure 3.12: *Discretized counterpart of the continuous function* $\cos(t)$. *Each dot represents a sample of the signal. The time between the samples is called the sampling interval.*

The discrete Fourier transform can be thought of as a linear operation. This means we can map a N-dimensional state vector to another N-dimensional state vector by a $N \times N$

matrix F_N. For a n qubit state N is equal to 2^n. Such transformation may be written as

$$|x\rangle = \sum_{j=0}^{N-1} x_j |j\rangle \longrightarrow \sum_{k=0}^{N-1} y_k |k\rangle , \tag{3.8}$$

where the amplitudes y_k are the discrete Fourier transform of the amplitudes x_j. In other words, the quantum Fourier transform (QFT) transforms the computational basis states as following

$$F_N |j\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{2\pi ijk/N} |k\rangle . \tag{3.9}$$

Note that this is the equivalent of a classical *inverse* discrete Fourier transform. The quantum Fourier transform (QFT) has the same effect as the classical inverse Fourier transform and vice versa. The quantum Fourier transform provides an exponential speedup over the classical fast Fourier transform algorithm.

We have already seen and used the QFT on a single qubit. The unitary matrix of the QFT on one qubit is the Hadamard gate:

$$F_2 = H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} . \tag{3.10}$$

For a multi-qubit system it becomes a bit more involved. To describe the circuit for the n-qubit QFT we need the Hadamard and controlled phase gate. We will write the phase gate as R_k where

$$R_k = \begin{pmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^k} \end{pmatrix} . \tag{3.11}$$

The circuit for a QFT on 3 qubits can be seen in Figure 3.13. Note that $S = R_2$ and $T = R_3$. The crossed gate at the end is the swap gate, which swaps two qubits. The order of qubits has to be reversed at the end of a QFT to get the correct order as result. Note that inverting the circuit by reversing the order of the gates and taking the inverse $U^\dagger$ of each gate U gives an equally efficient circuit for the inverse quantum Fourier transform $F_N^\dagger$.

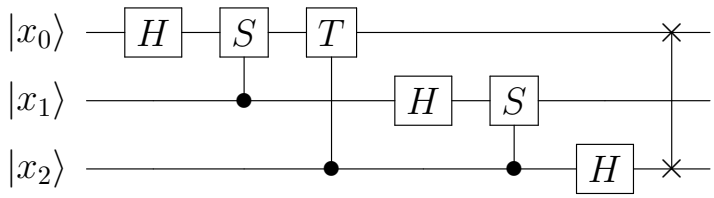


Figure 3.13: *Quantum Fourier transform on 3 qubits.*

The matrix of a QFT can be written out using $\omega = e^{2\pi i/N}$. For 3 qubits, where $N = 8$:

$$F_8 = \frac{1}{\sqrt{8}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^1 & \omega^2 & \omega^3 & \omega^4 & \omega^5 & \omega^6 & \omega^7 \\ 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 1 & \omega^3 & \omega^6 & \omega^1 & \omega^4 & \omega^7 & \omega^2 & \omega^5 \\ 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 & 1 & \omega^4 \\ 1 & \omega^5 & \omega^2 & \omega^7 & \omega^4 & \omega^1 & \omega^6 & \omega^3 \\ 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ 1 & \omega^7 & \omega^6 & \omega^5 & \omega^4 & \omega^3 & \omega^2 & \omega^1 \end{pmatrix}. \tag{3.12}$$

A more general circuit of the QFT for a n qubit state is described below in Figure 3.14.

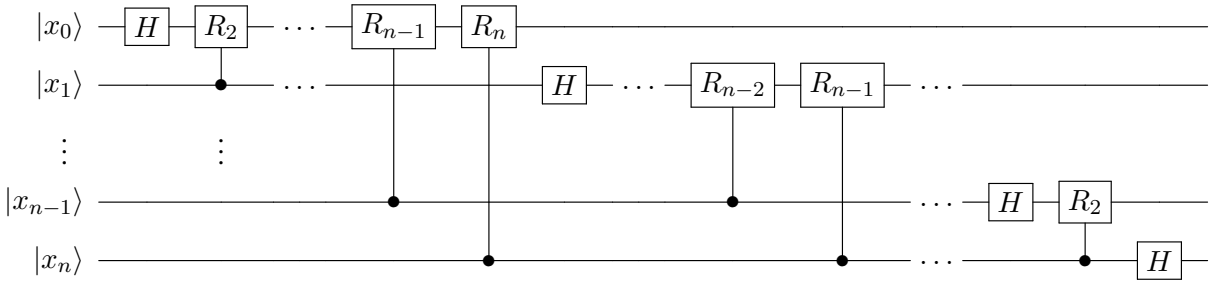


Figure 3.14: *Quantum Fourier transform on n qubits. Swap gates required to get the correct order at the end are omitted.*

The result of the Fourier transform of the original state is stored in the amplitudes. Remember however that these amplitudes cannot be extracted. Thus there is no way of determining the Fourier transform of the original state using the QFT. Even though we can't extract the transformed values after a QFT, the QFT has its use in quantum algorithms like Shor's algorithm for factoring integers and the quantum phase estimation algorithm.

3.4 Quantum Phase Estimation Algorithm

The quantum phase estimation algorithm (QPE) is a quantum algorithm to estimate the phase of an eigenvector of a unitary operation. Say we prepare an eigenstate $|u\rangle$ of a unitary operator U, where $U|u\rangle = e^{2\pi i\varphi}|u\rangle$ and the value of φ is unknown ($0 \leq \varphi \leq 1$). The goal is to estimate the value of φ when we don't necessarily know U or $|u\rangle$, but have available black boxes capable of preparing $|u\rangle$ and applying controlled-U operations.

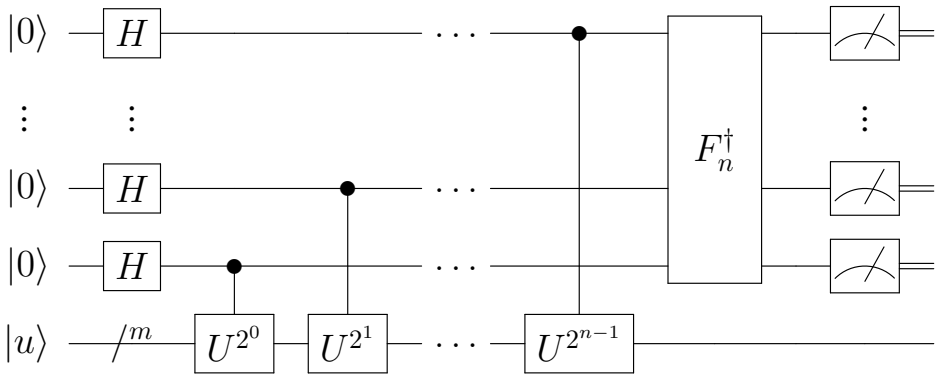


Figure 3.15: *Quantum phase estimation circuit, where $F_n^\dagger$ represent the inverse quantum Fourier transform on n qubits. The "/" denotes a register of m qubits.*

The circuit for the QPE algorithm found in Figure 3.15 consists of two registers. The upper n qubits comprise the first register which will be used to calculate the estimate of φ. The size of the first register can be chosen based on two factors: accuracy and success probability. More qubits will give you better accuracy and a higher success probability. The lower m qubits are the second register which will be prepared with the eigenstate $|u\rangle$ whose phase we want to estimate.

We start with the system in the state $|0\rangle^{\otimes n}|u\rangle$. After applying n Hadamard operations $H^{\otimes n}$ on the first register we end up with the state

$$\frac{1}{\sqrt{2^n}}(|0\rangle + |1\rangle)^{\otimes n}|u\rangle\,. \tag{3.13}$$

We continue by applying the controlled unitary operators U. Remembering that $U|u\rangle = e^{2\pi i\varphi}|u\rangle$, then $U^{2^j}|u\rangle = e^{2\pi i 2^j\varphi}|u\rangle$. A controlled-$U^{2^j}$ operator then transforms a state as following

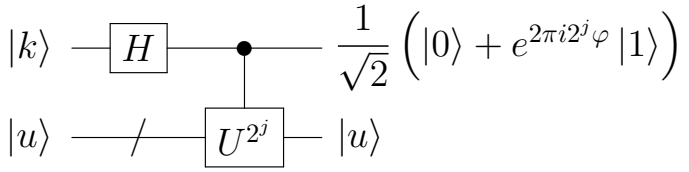


Figure 3.16: *Controlled-U^{2^j} operation on a qubit $|k\rangle$ and eigenstate $|u\rangle$.*

Note that the phase ends up in $|k\rangle$ while $|u\rangle$ stays the same. This is a phenomenon called *quantum phase kickback* and can be explained by examining the state throughout the transformation. The system in Figure 3.16 after the Hadamard gate is in the state

$$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|u\rangle = \frac{1}{\sqrt{2}}(|0\rangle|u\rangle + |1\rangle|u\rangle). \tag{3.14}$$

Then a controlled-U^{2^j} is applied, meaning that U^{2^j} is only applied when the control qubit ($|k\rangle$) is $|1\rangle$. We then end up with the state

$$\frac{1}{\sqrt{2}}\left(|0\rangle|u\rangle + |1\rangle\, U^{2^j}|u\rangle\right) \tag{3.15}$$

$$= \frac{1}{\sqrt{2}}\left(|0\rangle|u\rangle + |1\rangle\, e^{2\pi i 2^j \varphi}|u\rangle\right) \tag{3.16}$$

$$= \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 2^j \varphi}|1\rangle\right)|u\rangle\,. \tag{3.17}$$

So, after applying the controlled-U operators in the QPE circuit as seen in Figure 3.15, the state of the first register becomes

$$\frac{1}{\sqrt{2^n}}\left(|0\rangle + e^{2\pi i 2^{n-1}\varphi}|1\rangle\right)\cdots\left(|0\rangle + e^{2\pi i 2^1 \varphi}|1\rangle\right)\left(|0\rangle + e^{2\pi i 2^0 \varphi}|1\rangle\right) \tag{3.18}$$

$$= \frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i \varphi k}|k\rangle\,. \tag{3.19}$$

The second register stays in the state $|u\rangle$ throughout this computation. The total circuit is in the following state after the controlled-U operations:

$$\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i \varphi k}|k\rangle|u\rangle\,. \tag{3.20}$$

Note that this state is similar to the result of a QFT (Equation 3.9). By performing the *inverse* QFT on the state of the first register (Equation 3.20) we get

$$\frac{1}{2^n}\sum_{k=0}^{2^n-1}\sum_{j=0}^{2^n-1} e^{-2\pi i j k/2^n} e^{2\pi i \varphi k}|j\rangle|u\rangle\,. \tag{3.21}$$

If φ has a n bit binary fraction representation $\varphi = j/2^n$, measuring the first register in the computational basis will give us exactly φ. Note that φ doesn't always have a n bit binary fraction. In this case, we will get probabilistic measurement results. The probability $P(j)$ is large for values of j where $\varphi \approx j/2^n$, so in this case we can only get an approximation of φ.

An important idea in this algorithm is the ability of the inverse quantum Fourier transform to perform the transformation

$$F_{2^n}^{\dagger}\frac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1} e^{2\pi i \varphi k}|k\rangle|u\rangle = |\tilde{\varphi}\rangle|u\rangle\,, \tag{3.22}$$

essentially "extracting" an estimation $|\tilde{\varphi}\rangle$ of φ.

3.5 Superdense Coding

Suppose Alice and Bob want to share some information. They are allowed to meet up and share information of any size beforehand. After that they get separated, after which they are only allowed to communicate one classical bit of information. In classical computing, they can only communicate two possible values (one bit): 0 or 1. This seems pretty straightforward and obvious, but consider what happens when we allow them to communicate one *qubit* instead of one bit. How many possible values can they communicate with one qubit?

As it turns out, you can communicate two classical bits of information using one qubit. This is referred to as *superdense coding*. Recall the Bell states from Section 2.7:

$$\left|\Phi^+\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \tag{3.23}$$

$$\left|\Phi^-\right\rangle = \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle) \tag{3.24}$$

$$\left|\Psi^+\right\rangle = \frac{1}{\sqrt{2}}(|01\rangle + |10\rangle) \tag{3.25}$$

$$\left|\Psi^-\right\rangle = \frac{1}{\sqrt{2}}(|01\rangle - |10\rangle). \tag{3.26}$$

These maximally entangled two-qubit states have variations in parity and phase. If we were somehow able to "decode" the parity and phase of these states by mapping them to separate computational basis states, we could extract two classical bits of data from one qubit. It turns out this can be done by applying the circuit in Figure 3.17 on a Bell state. Note that this is the reverse of the circuit for creating the Bell state $|\Phi^+\rangle$. You can interpret the decode circuit as following: the CNOT gate decodes the parity of $|q_0\rangle|q_1\rangle$ to $|q_1\rangle$, and the Hadamard gate decodes the phase to $|q_0\rangle$.

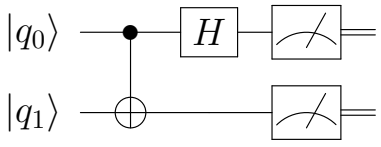


Figure 3.17: *Circuit to decode the Bell states to separate computational basis states.*

Input	$\text{CNOT}_{0,1}$	H_0
$\|\Phi^+\rangle$	$\frac{1}{\sqrt{2}}(\|00\rangle + \|10\rangle)$	$\|00\rangle$
$\|\Phi^-\rangle$	$\frac{1}{\sqrt{2}}(\|00\rangle - \|10\rangle)$	$\|10\rangle$
$\|\Psi^+\rangle$	$\frac{1}{\sqrt{2}}(\|01\rangle + \|11\rangle)$	$\|01\rangle$
$\|\Psi^-\rangle$	$\frac{1}{\sqrt{2}}(\|01\rangle - \|11\rangle)$	$\|11\rangle$

Figure 3.18: *Output of applying the decoding circuit in Figure 3.17 on every Bell state.*

A Bell state can be transformed to any other Bell state by manipulating a single qubit. For example, to go from $|\Phi^+\rangle$ to $|\Phi^-\rangle$, we simply apply a Z gate on $|q_0\rangle$. Given this information, we can think of a circuit allowing Alice and Bob to communicate two classical bits of information by communicating one qubit. Such circuit can be seen in Figure 3.19. Alice and Bob start with preparing and distributing a Bell state $|\Phi^+\rangle$. Alice, our sender, gets qubit $|q_0\rangle$ and Bob, our receiver gets qubit $|q_1\rangle$. Then they get separated, and from there on out are

only allowed to communicate one qubit. Alice encodes the two classical bits of information c_0c_1 in the entangled state by applying the appropriate gates on $|q_0\rangle$. Applying no gates will leave the state in $|\Phi^+\rangle$. Applying an X gate transforms the state to $|\Psi^+\rangle$. Applying a Z gate transforms it to $|\Phi^-\rangle$, and applying both transforms it to $|\Psi^-\rangle$. So if Alice wants to send the bits 11, she applies both the X and Z gate, giving the state $|\Psi^-\rangle$. After the second box the state is in one of the four Bell states, depending on the gates she applied. Alice then sends her qubit $|q_0\rangle$ to Bob (this is the one qubit of communication), who can then use the decode circuit from Figure 3.17 to extract the two bits of classical information. If Alice applied both gates, so the state is $|\Psi^-\rangle$, Bob will decode 11 as seen in Figure 3.18.

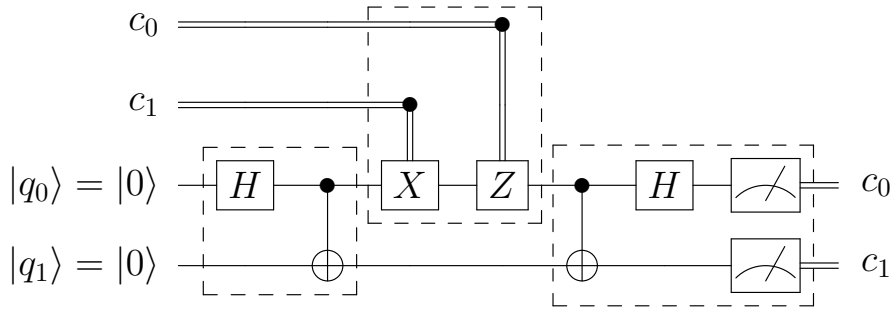


Figure 3.19: *Superdense coding circuit. Two bits of classical information c_0c_1 can be encoded in $|q_0\rangle$ by pre-sharing a Bell state and applying conditional single-qubit gates.*

We can send two bits of classical information with a single qubit using superdense encoding, assuming both parties are allowed to share information beforehand. Superdense coding is also the basis for secure quantum secret coding. It's impossible to eavesdrop when the Bell state was shared in a secure way. There is one qubit which is being sent, and even if an eavesdropper intercepts that qubit, there's no way of decoding it without the second qubit.

Chapter 4

Quantum Error Correction

Quantum computers, like classical computers, suffer from *noise*. That is, unwanted interactions from the outside world which causes errors in our data. For example, qubits lose their state in a very short period of time (called *decoherence*). To protect against noise, there exist quantum algorithms called *quantum error-correcting codes*. These algorithms encode a quantum state to make it resilient to noise. When the original state wishes to be recovered, the state can be decoded again. Applying such a code costs time and space, but gives us longer coherence times and lower error rates. To explain quantum error correction schemes, it's useful to first take a look at classical error correction.

4.1 Classical Error Correction

If you want to counter noise, you need to understand where the noise comes from. To identify a source of noise, we define a *noise channel*. The most simplistic classical noise channel is the *binary symmetric channel* (Figure 4.1). This model assumes we have independent single-bit errors. On the left is the state of our bit. Then some time passes, after which there is a probability of p that the bit has flipped due to noise, and a probability of $1 - p$ that our bit stayed in its correct state.

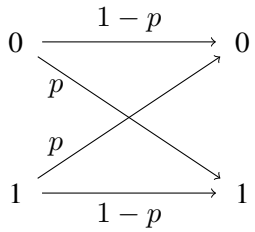


Figure 4.1: *Binary symmetric channel.*

A simple way to mitigate these errors is by using a repetition code, meaning we encode the message by repeating it a number of times:

$$0 \to 000 \tag{4.1}$$

$$1 \to 111. \tag{4.2}$$

The bit strings 000 and 111 are referred to as *logical* 0 and *logical* 1, since they represent 0 and 1. The method of decoding is called *majority voting*, which outputs the bit that appears most in the actual output. Say we want to send a bit with state 1. We encode it with three copies as 111. If an error occurs over the channel, for example the channel outputs 101, majority voting gives us the correct output: 1. When two errors occur, for example the channel outputs 001, majority voting gives the wrong output: 0. Majority voting with three bits fails when two errors occur, but succeeds otherwise. This code makes communication more reliable whenever $p < 1/2$.

4.2 Quantum Error Correction

We would like to develop a quantum error-correction code based on similar ideas from the classical error-correction code. However, when trying to apply these ideas to the quantum world we are faced with some difficulties:

1. *Measurement is destructive:* Attempting to observe the value of a quantum state by measuring it destroys the quantum state.
2. *No cloning:* Cloning a quantum state is impossible.
3. *Errors are continuous:* In the classical world, errors are discrete: a bit flip either happens or doesn't happen. With quantum states, a continuum of different errors can occur.

As it turns out, none of these problems are fatal and quantum error-correction is possible.

We start off with looking at the noise channels: where do errors come from and how can we mitigate them. A simplistic quantum noise channel is the *bit flip channel*, which is equivalent to the classical bit flip:

$$|\psi\rangle|e\rangle \to X\,|\psi\rangle|\hat{e}\rangle\,. \tag{4.3}$$

That is, the state $|\psi\rangle$ is taken to state $X|\psi\rangle$ with probability p, and is left in state $|\psi\rangle$ with probability $1-p$. Here $|e\rangle$ is the environment and $|\hat{e}\rangle$ the evolved environment, representing the evolution of a state and its environment. A similar noise channel is the *phase flip channel*, where a state $|\psi\rangle$ is taken to state $Z|\psi\rangle$ with probability p, and left in state $|\psi\rangle$ with probability $1-p$:

$$|\psi\rangle|e\rangle \to Z\,|\psi\rangle|\hat{e}\rangle\,. \tag{4.4}$$

For simplicity sake we will only look at the bit and phase flip channels. However, these kind of errors are not very realistic. A more realistic noise channel is the *amplitude dampening* or *relaxation* channel. In the hardware implementation of qubits we speak of a two-state system with an excited state $|e\rangle$ and a ground state $|g\rangle$, which correspond to the computational basis states $|1\rangle$ and $|0\rangle$. One of these states is more energetic than the other, like in the DRAM of your computer: you charge the capacitor to represent a 1, and uncharge it to represent 0. The uncharged state 0 is very stable, it remains uncharged. The 1 state however slowly loses its charge, slowly turning into 0 state. The same goes for a two-level quantum system: the ground state $|g\rangle$ is very stable, while the excited state $|e\rangle$ slowly turns into the lower energetic state $|g\rangle$.

4.2.1 Quantum Bit Flip Code

The idea of quantum error-correction is to encode a quantum state redundantly into a larger Hilbert space. For example, we encode a single qubit $\alpha\,|0\rangle + \beta\,|1\rangle$ as $\alpha\,|000\rangle + \beta\,|111\rangle$. For convenience we write these encoded states, called *code words*, as following:

$$|0_L\rangle = |000\rangle \tag{4.5}$$

$$|1_L\rangle = |111\rangle\,. \tag{4.6}$$

We refer to $|0_L\rangle$ and $|1_L\rangle$ as the *logical* $|0\rangle$ and *logical* $|1\rangle$ states. This encoding is achieved by the circuit in Figure 4.2.

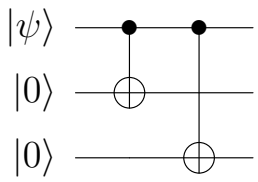


Figure 4.2: *Quantum circuit for encoding a qubit $|\psi\rangle$ for the three qubit bit flip code.*

Suppose we encoded our state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ as $|\psi_L\rangle = \alpha|000\rangle + \beta|111\rangle$ and a bit flip error occurs on one qubit which we would like to correct. The first step in correcting a bit flip error is detecting if such error happened. Error detection, or *syndrome diagnosis*, can be achieved through *stabilizer measurements*. It is useful to make ourselves familiar with the *stabilizer formalism* to understand stabilizer measurements. Consider the Bell state $|\psi\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$. This state is *stabilized* by the operators X_1X_2 and Z_1Z_2, meaning $X_1X_2|\psi\rangle = |\psi\rangle$ and $Z_1Z_2|\psi\rangle = |\psi\rangle$. The basic principle behind the stabilizer formalism is that we can (often more easily) describe a state by what operators stabilize them. Stabilizers also force discrete errors, meanings all errors from all channels can be decomposed in X and Z errors. If we can then make an error-correction code that corrects both X and Z errors, we can correct *any* error.

Stabilizer measurements allow us to learn information about a state without fully collapsing it. We have already seen a stabilizer measurement in Section 2.9 to calculate the parity of a state. By calculating and measuring the parity of a state, we extracted partial information about the state without fully collapsing it.

Consider a two-qubit state $|\psi\rangle = \alpha|00\rangle + \beta|11\rangle$. We can detect what, if any, bit flip error occurred with a Z_1Z_2 stabilizer measurement (Figure 4.3). The measurement result is called the *error syndrome*. A Z_1Z_2 stabilizer measurement essentially measures the parity of the qubits $|d_0\rangle|d_1\rangle$ using an *ancilla* (helper) qubit $|a_0\rangle$.

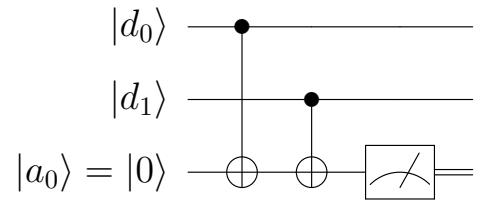


Figure 4.3: *Quantum circuit of a Z_1Z_2 stabilizer measurement.*

We can extract the error syndrome of an encoded state $|\psi_L\rangle = \alpha|000\rangle + \beta|111\rangle$ by using the two stabilizers Z_1Z_2 and Z_2Z_3 (Figure 4.4).

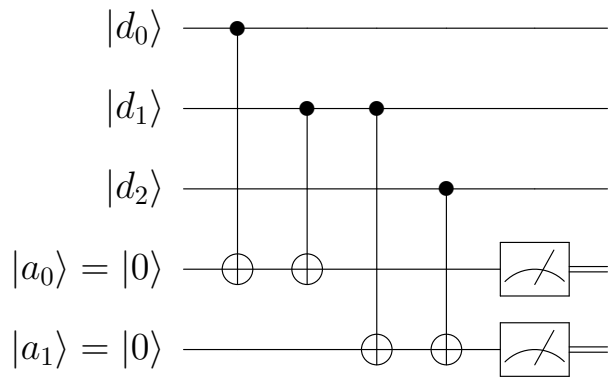


Figure 4.4: *Measuring error syndrome with stabilizer measurements Z_1Z_2 and Z_2Z_3.*

Suppose a bit flip occurred on our encoded state $|\psi_L\rangle$ and we're left with $|\psi'_L\rangle = \alpha|001\rangle + \beta|110\rangle$. The ancilla qubits $|a_0\rangle$ and $|a_1\rangle$ correspond to the stabilizer measurement of two qubits of the state:

$$|a_0\rangle = Z_1Z_2\left|\psi'_L\right\rangle \tag{4.7}$$
$$|a_1\rangle = Z_2Z_3\left|\psi'_L\right\rangle. \tag{4.8}$$

For an encoded state $\alpha|000\rangle + \beta|111\rangle$ the error syndrome measurement results are both zero: $M(a_0) = M(a_1) = 0$. When this is not the case, we know we are in an erroneous state. For the state $|\psi'_L\rangle$ we can see that

$$M(a_0) = 0 \tag{4.9}$$
$$M(a_1) = 1. \tag{4.10}$$

These measurements tell us an error has occurred, and where it occurred. Assuming single bit flip errors only, an encoded state can be in one of the following states:

$$\alpha|000\rangle + \beta|111\rangle \text{ no error} \tag{4.11}$$
$$X_{d_0} \to \alpha|100\rangle + \beta|011\rangle \text{ bit flip on qubit zero} \tag{4.12}$$
$$X_{d_1} \to \alpha|010\rangle + \beta|101\rangle \text{ bit flip on qubit one} \tag{4.13}$$
$$X_{d_2} \to \alpha|001\rangle + \beta|110\rangle \text{ bit flip on qubit two.} \tag{4.14}$$

We can find out where the error occurred by the result of the error syndrome measurement. The error syndrome measurement results corresponding to the qubit at which the error occurred can be seen in Figure 4.5. To correct the state, we can simply apply an X operation on the flipped qubit, flipping it back to the correct state. Error syndromes can be decoded with the help of classical computing.

This code can detect two errors, but only correct one. For example, consider the following erroneous states for an encoded state $\alpha|000\rangle + \beta|111\rangle$:

$$X_{d_0} \to \alpha|100\rangle + \beta|011\rangle \tag{4.15}$$
$$X_{d_1}X_{d_2} \to \alpha|011\rangle + \beta|100\rangle. \tag{4.16}$$

Error	$M(a_0)$	$M(a_1)$
None	0	0
X_{d_0}	1	0
X_{d_1}	1	1
X_{d_2}	0	1

Figure 4.5: *Error syndrome measurement results corresponding to where the bit flip occurred.*

We can detect that both states are erroneous, since for both states the error syndrome measurement results are $M(a_0) = 1$ and $M(a_1) = 0$. When correcting an erroneous state, we always choose the least amount of errors that explains the error syndrome. There's a probability of p that a single error happened, and a probability of p^2 that two errors happened. The state in 4.16 would get corrected to the wrong state $\alpha |111\rangle + \beta |000\rangle$, because there's a higher probability that X_{d_0} happened than $X_{d_1} X_{d_2}$. So in the case of multiple bit flip errors, corrections can be wrong and generate an unrecoverable error.

After correction we are left with an encoded state $|\psi_L\rangle = \alpha |000\rangle + \beta |111\rangle$. To get the original state $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$ back, we need to decode the state. This can be achieved by simply reversing the encoding circuit in Figure 4.2 (since CNOT is Hermitian, the reverse of the encoding circuit is itself). Note that if you just want to do measurement, you can measure any of the qubits in $|\psi_L\rangle$ instead of decoding it.

4.2.2 Quantum Phase Flip Code

The bit flip code only catches X errors. If the phase of a state $|\psi\rangle$ flipped to $Z|\psi\rangle$ and we want to correct it, we need a phase flip code. The quantum phase flip code is based on the same principles as the bit flip code. The difference with the bit flip code is that we change the basis of the state. We do this by encoding the qubits in the same way as the bit flip code, but we apply a Hadamard gate to each qubit (Figure 4.6). This will encode a state

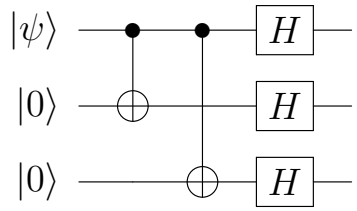


Figure 4.6: *Quantum circuit for encoding a qubit $|\psi\rangle$ for the three qubit phase flip code.*

$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$ as $|\psi_L\rangle = \alpha |{+}{+}{+}\rangle + \beta |{-}{-}{-}\rangle$. Now when a phase flip occurs, for example $|\psi'_L\rangle = \alpha |{+}{-}{+}\rangle + \beta |{-}{+}{-}\rangle$, we can extract the error syndrome in a similar way to the bit flip code. Error syndrome measurement can be done with the stabilizers $X_1 X_2$ and $X_2 X_3$ as seen in Figure 4.7. In the case of $|\psi'_L\rangle$, syndrome measurements are $M(a_0) = 1$ and $M(a_1) = 1$, telling us the error is in qubit $|d_1\rangle$. We can then correct the state by applying a Z gate on $|d_1\rangle$. Decoding the original state is done by reversing the encoding circuit in

Figure 4.6.

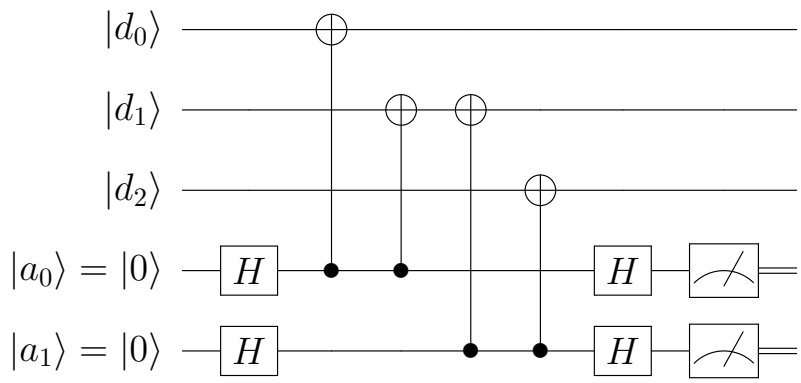


Figure 4.7: *Measuring error syndrome with stabilizer measurements X_1X_2 and X_2X_3.*

4.2.3 Shor Code

We have seen a code which corrects bit flip and phase errors, but what about a code which corrects *both* errors? It turns out that this can be achieved through a *concatenated* code. The *Shor code* is a combination of the three qubit bit flip and phase flip code and can correct arbitrary single-qubit errors. The encoding circuit for this code can be seen in Figure 4.8. We first apply the encode step of the phase flip code, giving

$$|0\rangle \to |{+}{+}{+}\rangle \tag{4.17}$$
$$|1\rangle \to |{-}{-}{-}\rangle\,. \tag{4.18}$$

Then on every encoded qubit we apply the bit flip code, meaning

$$|+\rangle \to \frac{1}{\sqrt{2}}(|000\rangle + |111\rangle) \tag{4.19}$$
$$|-\rangle \to \frac{1}{\sqrt{2}}(|000\rangle - |111\rangle). \tag{4.20}$$

This results in a nine qubit code with code words

$$|0_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle) \tag{4.21}$$
$$|1_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle - |111\rangle)(|000\rangle - |111\rangle). \tag{4.22}$$

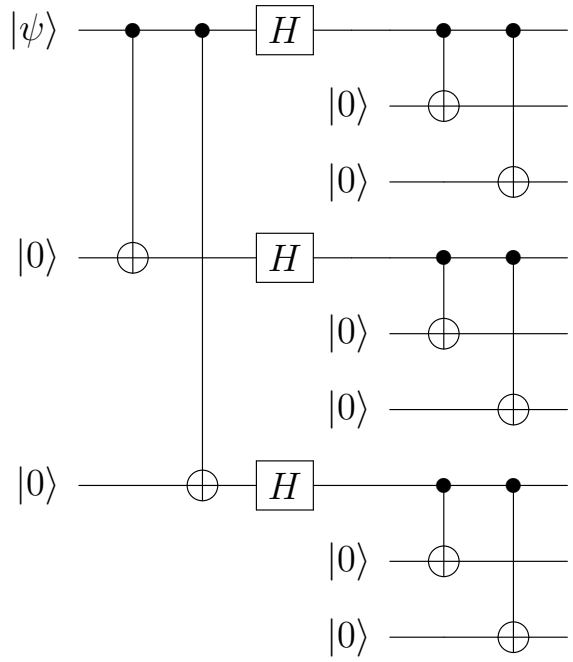


Figure 4.8: *Quantum circuit for encoding a qubit $|\psi\rangle$ for the nine qubit Shor code.*

Error syndrome measurement for the Shor code gets a bit more hairy. The stabilizers are described in Figure 4.9. Suppose a bit flip happens on the first qubit of a state $|\psi_L\rangle = |0_L\rangle$:

$$|\psi'_L\rangle = \frac{1}{2\sqrt{2}}(|100\rangle + |011\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle). \tag{4.23}$$

The stabilizer measurement $Z_1 Z_2 |\psi'_L\rangle$ gives us 1, so we know a bit flip occurred on the first or second qubit. Next, the stabilizer measurement $Z_2 Z_3 |\psi'_L\rangle$ gives us 0, so we can conclude a bit flip happened on the first qubit and we can correct it by flipping it back.

Z stabilizers	X stabilizers
$Z_1 Z_2$	$X_1 X_2 X_3 X_4 X_5 X_6$
$Z_2 Z_3$	$X_4 X_5 X_6 X_7 X_8 X_9$
$Z_4 Z_5$	
$Z_5 Z_6$	
$Z_7 Z_8$	
$Z_8 Z_9$	

Figure 4.9: *Shor code stabilizers.*

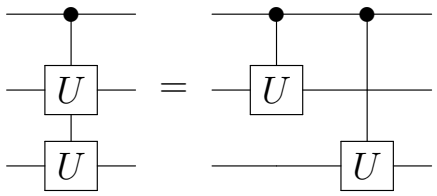
What about a phase flip? Suppose a phase flip happens on the first qubit of a state $|\psi_L\rangle =$

$|0_L\rangle$. This phase flip flips the sign of the first block of qubits:

$$\left|\psi'_L\right\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle). \tag{4.24}$$

A phase flip on *any* of the first three qubits has this effect, and we can correct any of these errors with the same procedure. Error syndrome measurement for a phase flip begins with comparing the phase of the first two blocks of three qubits with the stabilizer measurement $X_1X_2X_3X_4X_5X_6\,|\psi'_L\rangle$. The first two blocks $(|000\rangle - |111\rangle)(|000\rangle + |111\rangle)$ have different signs, and the stabilizer measurement will measure 1. The second two blocks $(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)$ have equal signs, and the stabilizer measurement $X_4X_5X_6X_7X_8X_9\,|\psi'_L\rangle$ measures 0. We can conclude that the phase flipped in the first block of three qubits and we can correct it by flipping the sign of the first block. These methods to correct errors will work for any of the nine qubits.

To describe more complex circuits it's useful to adopt a shorthand notation:



We can then describe the error syndrome measurement circuit of the Shor code as seen in Figure 4.10 on the next page.

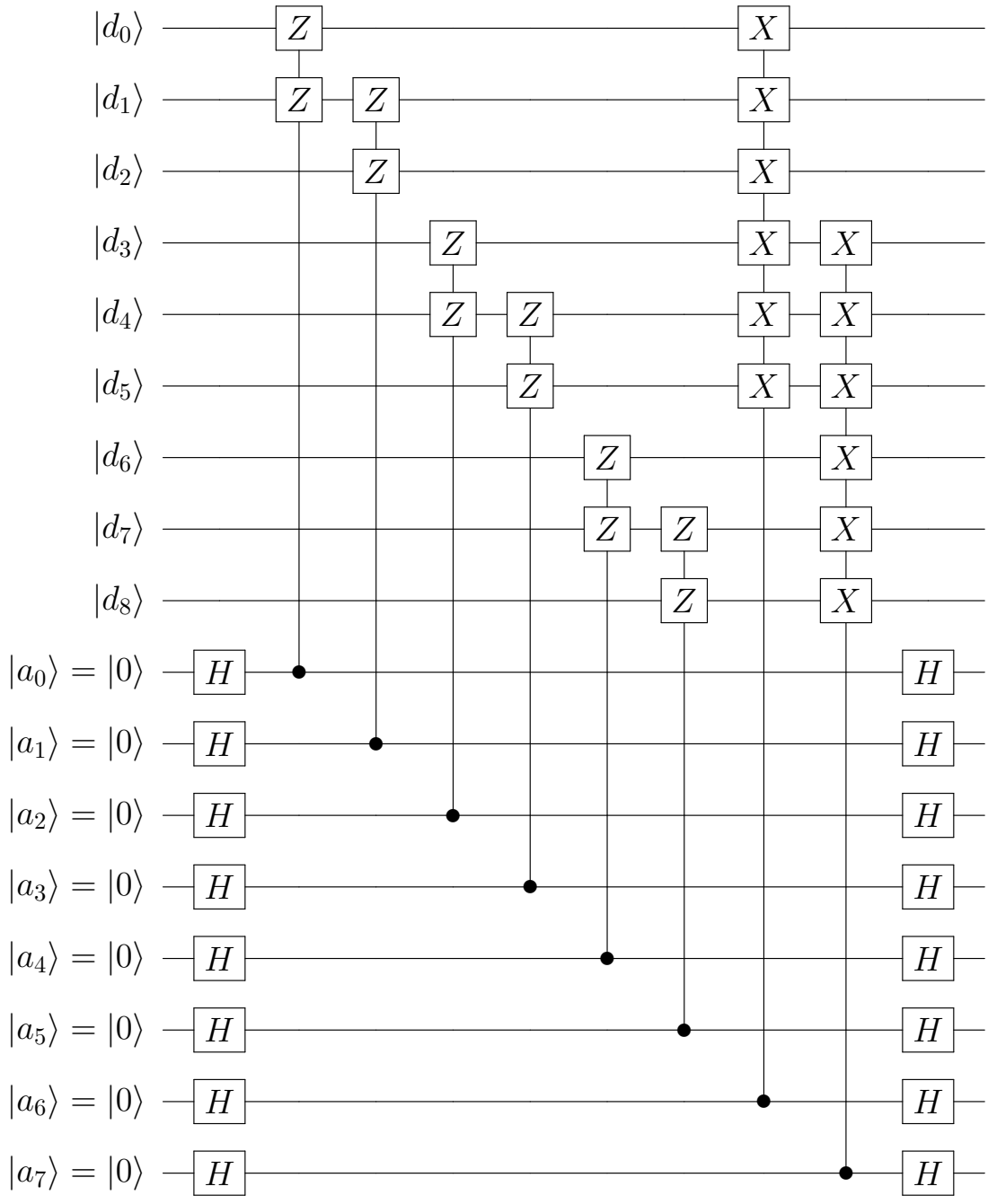


Figure 4.10: *Error syndrome measurement circuit for the nine qubit Shor code.*

4.2.4 Steane Code

The Shor code was the first complete quantum error-correction code presented, but it is not the most efficient. The *Steane code* is a seven qubit error-correction code which can correct arbitrary single-qubit errors. It turns out that the most efficient complete quantum error-correction code can be achieved with five qubits, but we will look at the Steane code because it's mathematically very convenient. The Steane code encode circuit is described in Figure 4.11, and gives the following code words:

$$|0_L\rangle = \frac{1}{\sqrt{8}}\Big(|0000000\rangle + |1010101\rangle + |0110011\rangle + |1100110\rangle + |0001111\rangle + |1011010\rangle + |0111100\rangle + |1101001\rangle \Big) \tag{4.25}$$

$$|1_L\rangle = \frac{1}{\sqrt{8}}\Big(|1111111\rangle + |0101010\rangle + |1001100\rangle + |0011001\rangle + |1110000\rangle + |0100101\rangle + |1000011\rangle + |0010110\rangle \Big). \tag{4.26}$$

Error syndrome measurement is achieved in a similar way as the Shor code, but with six ancilla qubits and different stabilizers. The stabilizers used in error syndrome measurement can be found in Figure 4.12.

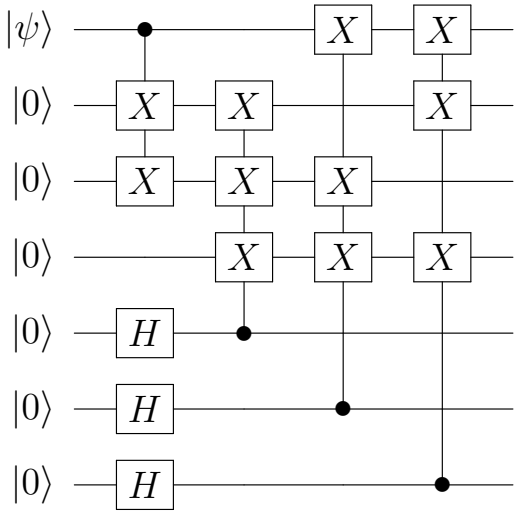


Figure 4.11: *Quantum circuit for encoding a qubit $|\psi\rangle$ for the seven qubit Steane code.*

Z stabilizers	X stabilizers
$Z_1Z_3Z_5Z_7$	$X_1X_3X_5X_7$
$Z_2Z_3Z_6Z_7$	$X_2X_3X_6X_7$
$Z_4Z_5Z_6Z_7$	$X_4X_5X_6X_7$

Figure 4.12: *Steane code stabilizers.*

4.3 Towards Fault-tolerant Universal Quantum Computing

In Section 2.5 we briefly discussed universal quantum computing. Universal quantum computing can be achieved through universal gate sets consisting of the complete Clifford set

and one or more non-Clifford gates. We have seen how we can use quantum error-correction codes to mitigate errors from noise. *Fault tolerance* means that we can not only mitigate and fix some of these errors, but that we can contain these errors. That is, if an error occurs it doesn't spread out and infect your whole system with errors. So when we talk about fault-tolerant universal quantum computing, we talk about being able to do computation in a universal way, and also being able to contain and correct errors.

In order to achieve fault-tolerant universal quantum computing, we need to be able to perform logical operations on our encoded logical qubits. To do useful computation, we also need the ability of multiple logical qubits. So far we have only looked at single logical qubits. Furthermore, errors can occur on the ancillary qubits and inject errors into the data qubits, which we need to be resistant to. These are some of the problems we need to overcome to achieve fault-tolerant universal quantum computing.