Theoretical Foundations of Quantum Computing and the Implementation of the Quantum Fourier Transform

Natalia Dziubelski
Bard College, nd7666@bard.edu

Theoretical Foundations of Quantum Computing and the Implementation of the Quantum Fourier Transform

A Senior Project submitted to
The Division of Science, Mathematics, and Computing
of
Bard College

by
Natalia Dziubelski

Annandale-on-Hudson, New York
Sep, 2022

Abstract

Quantum computing is a growing field with the potential to revolutionize computation. This thesis explores the foundations of quantum computing with specific focus on the efficacy of the Quantum Fourier Transform (QFT). The fundamentals of quantum computing were described through an explanation of quantum mechanics and the mathematics needed to understand the quantum computing model and its operations. Using IBM's simulators and quantum processors, the QFT was implemented on a classical data set, and the results were compared to the predicted output values. It was found that the QFT simulator was able to produce results consistent with Discrete Fourier Transform, which affirmed its potential to be a successful method of extracting frequencies from a set of data in the time domain. The quantum processors were able to produce results with small percent error for small qubit systems, but increased in error as the number of qubits within the system grew. Despite this, with the appropriate error correction methods, it was concluded that the QFT is a functional method for transforming a state from the time domain to the frequency domain.

Contents

Acknowledgments

First and foremost, I would like to thank my advisor Antonios Kontos for his guidance and support through this project and my time at Bard. I am grateful for the countless explanations and time taken to discuss complicated topics. This project would not have been possible without his input and dedication. I would also like to thank all of the professors in the physics department for constantly challenging me to be at my full potential, as well as fostering a supportive community. It has been a privilege to be under your tutelage.

I would like to extend my gratitude to my wonderful family and friends who have shown me unconditional love and support through this process, my time at Bard, and throughout my life.

1
Introduction

1.1 Fundamental Principles of Quantum Mechanics

Quantum mechanics is a branch of physics that describes the behavior of matter, light, and energy at an atomic and subatomic level. It accounts for the properties of photons, molecules, and atoms and their constituents: electrons, protons, and neutrons. It is a fundamental theory that is woven into many areas of modern physics.

Quantum mechanics is based on the principles of wave-particle duality [6], meaning that particles can exhibit both wave and matter-like behavior depending on how they are observed. The definitions of waves and matter were quite distinct until the introduction of this concept. The traditional examples of waves were light and sound, while electrons, protons, neutrons, and other small atoms were regarded to be matter. The double slit experiment is a famous experiment that illustrates the wave-particle duality that is so prominent in quantum mechanics.

The experiment consists of a barrier with two slits in it, a stream of particles, and a screen to record the pattern of the particles after passing through the slits. Both the classically defined matter particles and the photons produce an interference pattern on the screen as a result. This suggests that the particles have a wave-like character. As they pass through the slits, these particles interfere with each other and create an interference pattern that can only be explained by an inherent wavelike quality.

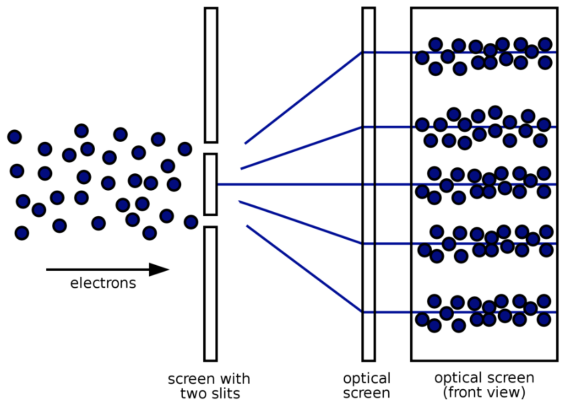


Figure 1.1.1: The double-slit experiment result for a stream of electrons is illustrated above.

Interestingly, when the experiment is repeated with detectors to determine which slit a particle has gone through, the interference pattern disappears and the particles behave as as individual particles. This demonstrates the observer effect [2], which is the idea that making an observation affects how a particle behaves.

The wave-like nature of particles along with the implication of the observer effect have been fundamental to the development of quantum mechanics, and have resulted in a mathematical framework to describe the behavior and interactions of small particles. These principles permit the occurrence of seemingly counterintuitive phenomena, such as the capacity of a particle to exist in multiple states simultaneously and the ability of particles to entangle and interact with one another at great distances.

The study of Quantum mechanics has resulted in a new fundamental understanding of nature, and has opened the doors for considerable scientific development. Harnessing the wave-like nature of small particles has resulted in significant technological advancement including that of quantum computers. This counter intuitive theory for the behavior of small particles is now yielding tangible and comprehensible results.

1.2 Quantum + Computing = Quantum Computing

Computing is the process of performing mathematical or logical operations on an input to get an output. Computation can take many forms, including numerical calculations, logical operations, data analysis, and simulation. It involves using hardware and software to input, process, and output data and information. There are various types of computation, each requiring their own unique set of tools and techniques.

In classical computing, computation is performed using binary digits, or bits. Bits are the fundamental units of information that are represented by one of two states, 0 or 1. These values represent the two possible states of an electronic switch. A quantum computer uses qubits, which in contrast, can exist in multiple states simultaneously due to the properties of quantum mechanics [8]. Both bits and qubits are used to represent data and operations that can be processed in their respective devices.

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

Figure 1.2.1: This figure illustrates two potential qubit states and their vectorial definitions [9].

In order to transform an input, a computer must possess the ability to perform certain operations. In a classical system, a computer will use logic gates to manipulate bits and perform computations. The central processing unit is responsible for executing instruction and coordinating the various components of a classical computation. A quantum computer uses quantum gates to perform operations on, superpose, and entangle qubits. They can be combined to create complex quantum circuits that can then perform specific quantum computations.

$$\boldsymbol{X}|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

Figure 1.2.2: The quantum X gate transforms a qubit in the state $|0\rangle$ to the state $|1\rangle$ [9].

Every computer has a physical system in which fundamental units of information exist and are manipulated. A classical computer uses transistors [7] to perform computations. Transistors are small on-off switches that can be controlled by an electrical signal. They are arranged in circuits to create logic gates that then act on bits. Quantum computers rely on quantum processors, which are made up of entangled qubits and are manipulated by quantum gates. Quantum processors are made of various physical systems, a superconducting circuit [10] being one that is more common. These circuits are made from superconducting materials that can be used to create qubits which are then controlled by microwave pulses.

Classical computers have built in mechanisms that assist in detecting and correcting errors within the data. Due to the fragile nature of qubits, quantum computers are prone to errors, and require advanced techniques for error correction, some of which include quantum error correction codes, fault-tolerant quantum computations, quantum teleportation and quantum annealing [6]. Progress in error correction is essential for the development of practical quantum computers.

While classical computers have proven to be incredibly useful, and have become a staple of the modern world, they are limited by the number of bits they can process simultaneously. This limits their ability to solve certain types of problems, particularly those involving large amounts of data or complex optimization problems. Quantum computers have the potential to solve these problems, as the properties of quantum mechanics they operate on allow for an exponential speed up of data processing [5]. Though quantum computers are still in the early stages of development, and are not yet capable of outperforming classical computers for most tasks, they have the potential to have a significant impact on the future of science and technology.

1.3 IBM Quantum Systems

IBM's quantum computers work based on the principles of quantum mechanics. IBM is a leading producer of quantum hardware and processors, utilizing the power of superconducting circuits to produce qubits [1]. Compared to a classical computer, an IBM quantum processor is not much larger than that of a classical processor. However, a quantum hardware system is approximately the size of a car. The hardware is mainly composed of cooling systems to keep the superconducting processor at an appropriate operational temperature.

To function properly, quantum processors need to be very cold–about a hundredth of a degree above absolute zero. To achieve this state, IBM uses cooled superfluids to create superconductors. The low temperature allows the material within quantum processors to exhibit the ideal traits for functionality. The cold temperatures allow for electrons to move through the materials without resistance, as per the definition of a superconductor. As electrons pass through the superconductors, they match up to form "cooper pairs" which can then carry charge across barriers or insulators through a process known as quantum tunneling.

IBM's quantum computers utilize Josephson junctions as superconducting qubits. Josephson junctions are simply two superconductors placed on either side of an insulator. The superconducting qubits are arranged in a lattice-like structure and are controlled using microwave photons. These are used to get qubits to hold, change and read out individual units of quantum information.

IBM Quantum is an online platform allowing for public access to cloud based quantum computing services, including access to a set of IBM's prototype quantum processors. The service can be used to run algorithms and experiments around the possibilities of quantum computing [1]. The system allows for the study of the quantum Fourier transform along with the efficacy of quantum computing in its early stages.

2 Mathematical Basics of Quantum Computing

2.1 Representing Single Qubit States

2.1.1 Qubit Formalization

Qubits are the basic unit of information used in quantum computing. What makes a qubit special is its ability to harness the principles of quantum mechanics. A classical bit will always have a defined state, either 0 or 1, while a qubit doesn't require one until the act of measurement.

The two possible states defined in equation 1.2.1 have a special relationship ensuring their definition upon measurement. The vectors defined by the $|0\rangle$ and $|1\rangle$ state are orthonormal, meaning they are orthogonal and normalized [3]. When measured, this property will ensure that the qubit will have an output of either $|0\rangle$ or $|1\rangle$. Because the vectors are linearly independent, they can be used to describe any vector in 2D space using vector addition and scalar multiplication. These properties imply that $|0\rangle$ and $|1\rangle$ form an orthonormal basis.

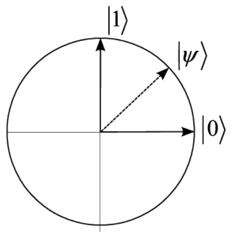


Figure 2.1.1: The $|0\rangle$ and the $|1\rangle$ state are perpendicular to one another and can be used to represent any vector in 2D space [9].

The properties of orthonormality and linear independence allow for the state of a qubit to be represented as a linear combination of the two states $|0\rangle$ and $|1\rangle$. This type of combination is described as superposition and can be representative of a qubit's state vector.

$$|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \tag{2.1.1}$$

[9]

Given a qubit can be in the the $|0\rangle$ state or the $|1\rangle$ state, or a superposition of the two, there are an infinite amount of states a qubit can be in. A qubit can be any linear combination of $|0\rangle$ and $|1\rangle$ as long as their corresponding amplitudes are normalized.

2.1.2 Measurement

Measurement is a fundamental concept in quantum computing. It is the process of extracting information from a quantum system. This operation causes a quantum system to collapse into one of its possible states with a probability determined by the amplitude of each respective state.

Probabilities

When a qubit state is in superposition, there is a simple way to measure the probability of finding the qubit's state vector, ψ in a particular state x.

$$p(|x\rangle) = |\langle x|\psi\rangle|^2 \tag{2.1.2}$$

[9]

This process is the absolute value of an inner product between two vectors squared. The inner product is a generalization of the dot product and always results in a scalar value, which in this case, is the probability of the state vector ψ being in a state x. The state x is representative of any possible qubit state.

The implementation of equation 2.1.2 on our qubit state 2.1.1 to find the probability of the qubit being measured in the $|0\rangle$ state yields a fifty percent chance of the qubit being in either the $|0\rangle$ state, or the $|1\rangle$ state.

$$\begin{aligned}\langle 0|\psi\rangle &= \frac{1}{\sqrt{2}}\langle 0|0\rangle + \frac{1}{\sqrt{2}}\langle 0|1\rangle \\ &= \frac{1}{\sqrt{2}}(1) + \frac{1}{\sqrt{2}}(0) = \frac{1}{\sqrt{2}} \\ p(|0\rangle) &= |\langle 0|\psi\rangle|^2 = \frac{1}{2}\end{aligned}$$

[9]

The result is simply the amplitude of the corresponding state squared.

Normalization

Normalization is the process of ensuring that the quantum state of a qubit is properly scaled so that it's total probability is equal to one.

$$\langle\psi|\psi\rangle = 1 \tag{2.1.3}$$

[9]

The qubit state vector ψ is represented by a linear combination of the vector states $|0\rangle$ and $|1\rangle$ with corresponding complex coefficients α and β. The total probability of measuring any outcome is the sum of the squared magnitudes of the probability amplitudes for each possible outcome.

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{2.1.4}$$

$$\langle\psi|\psi\rangle = 1$$

$$|\alpha|^2 + |\beta|^2 = 1$$

Normalization is then achieved by dividing the qubit's state vector by the square root of the sum of the squared magnitudes of the probability amplitudes. This ensures the total probability of measuring any outcome is equal to one.

Normalization is a fundamental requirement for the validity of quantum mechanics and allows for meaningful predictions and measurements.

Global Phase

The global phase is an overall phase factor that can be applied to a quantum state vector without affecting any observable physical quantities, such as probabilities or measurement outcomes. This is because any observable quantity in quantum mechanics depends only on relative phases between components of the quantum state rather than on the global phase. Generally, any factor γ, for which $|\gamma|$=1 is defined as a global phase.

$$|\langle x|(\gamma|a\rangle)|^2 = |\gamma\langle x|a\rangle|^2 = |\langle x|a\rangle|^2 \tag{2.1.5}$$

[9]

Using equation 2.1.5, it is possible to differ this state $|1\rangle$ by a factor of i and observe the result.

$$i|1\rangle = \begin{bmatrix} 0 \\ i \end{bmatrix}$$

$$|\langle x|(i|1\rangle)|^2 = |i\langle x|1\rangle|^2 = |\langle x|1\rangle|^2$$

After applying the rules of measurement, the factor of i disappears and its effect is completely independent of the measured state. This implies that the states $|1\rangle$ and $i|1\rangle$ are equivalent in all ways that are physically relevant due to the global phase principle.

The Observer Effect

The observer effect is a phenomenon where the act of observing a system can affect the system being observed [9]. Because of this, once a qubit is measured, it is known with certainty what the state of the qubit is. If a qubit in superposition is measured to be in the state $|0\rangle$, there is a 100% chance that another measurement will yield the same result. This implies the act of measurement changes the state of the qubit.

This is often referred to as collapsing the state of the qubit. If the state of a qubit were to be measured at each point of a computation, it would always be well defined. This would

$$|q\rangle = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \xrightarrow{\text{Measure } |0\rangle} |q\rangle = |0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

Figure 2.1.2: After a qubit is measured, it collapses to the measured state [9].

make a qubit no different than a classical bit, and the computation could easily be replaced by a classical computation. To achieve quantum computation, qubits must be allowed to explore more complex states, which is only possible in the absence of observation. Measurements are therefore only used when it is necessary to extract an output.

2.1.3 Bloch Sphere

The Bloch sphere is a useful geometric representation of the state space of a qubit. It is a sphere with a diameter of one, where the north and south poles represent the two orthogonal states of the qubit, $|0\rangle$ and $|1\rangle$ [9]. The other points on the sphere correspond to the superpositions of these states.

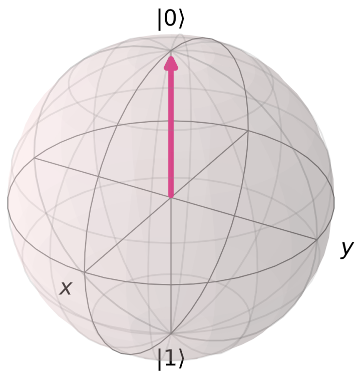


Figure 2.1.3: The $|0\rangle$ qubit state represented on the Bloch Sphere [9].

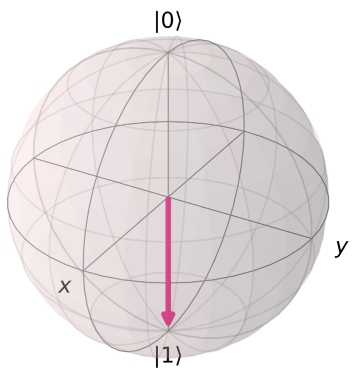


Figure 2.1.4: The $|1\rangle$ qubit state represented on the Bloch Sphere.

The general state of a qubit can be represented as a linear combination of orthogonal vectors with complex coefficients as in 2.1.2. Because global phase is immeasurable, it is only possible to measure the difference in phase between the states $|0\rangle$ and $|1\rangle$. By confining the coefficients α and β to be real numbers, and adding the term $e^{i\phi}$, we are able to isolate the relative phase between the two terms.

$$|\psi\rangle = \alpha|0\rangle + e^{i\phi}\beta|1\rangle \tag{2.1.6}$$

$$\alpha, \beta, \phi \in \mathbb{R}$$

[9]

With the knowledge that a qubit must be normalized, as well as some trigonometric substitutions, it is possible to describe the real numbers α and β in terms of the variable θ. Using this and the additional phase difference term, the state of any qubit can be described using the variables ϕ and θ.

$$\sqrt{\alpha^2 + \beta^2} = 1, \qquad \sqrt{sin^2x + cos^2x} = 1 \quad \longrightarrow \quad \alpha = \cos\frac{\theta}{2}, \quad \beta = \sin\frac{\theta}{2}$$

$$|\psi\rangle = \cos\frac{\theta}{2}|0\rangle + e^{i\phi}\sin\frac{\theta}{2}|1\rangle \tag{2.1.7}$$

[9]

This general qubit state can be interpreted and plotted with the use of spherical coordinates. The radius is R=1 since the magnitude of a qubit state vector is always 1. Using this formula, any single qubit can be plotted on the surface of the Bloch sphere.

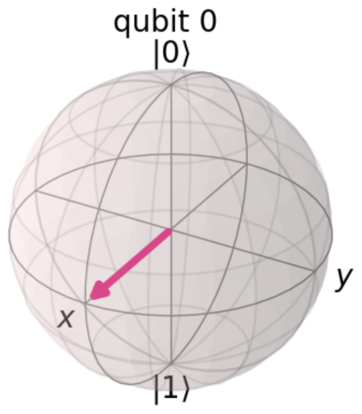


Figure 2.1.5: A superposed qubit state represented on the Bloch sphere [9].

The Bloch sphere provides a convenient way to visualize and analyze the state of a qubit. Any unitary operation on a qubit can be represented as a rotation on the Bloch sphere and any measurement of a qubit can be represented as a projection onto one of the poles of the sphere. The Bloch sphere provides an insight into the various basis states and makes it easier to understand the behavior of quantum systems.

2.2 Single Qubit Gates

Single qubit gates are quantum logic gates that operate on a single qubit to change its state. These gates can be represented as both matrices and rotations around the Bloch sphere.

2.2.1 Pauli Gates

The Pauli gates consist of three basic gates: Pauli-X, Pauli-Y and Pauli-Z, which correspond to rotations around the X, Y and Z axes of the Bloch sphere, respectively [9]. The Pauli-X gate is also known as the NOT gate, referring to the classical logic gate. It flips the state of a qubit from $|0\rangle$ to $|1\rangle$ or vice versa. This can be thought of as a rotation by π radians around the x-axis of the Bloch sphere.

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = |0\rangle\langle 1| + |1\rangle\langle 0|$$

$$X|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = |1\rangle$$

[9]

The Pauli-Y gate corresponds to a rotation of π around the Y axis of the Bloch sphere and is represented by the matrix below.

$$Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} = -i|1\rangle\langle 1| + i|1\rangle\langle 0|$$

[9]

The Pauli-Z gate corresponds to a rotation around the Z axis of the Bloch sphere by π radians. It is represented by the matrix below.

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = -|0\rangle\langle 0| - |1\rangle\langle 1|$$

[9]

Any single-qubit operation can be expressed as a combination of rotations around the X, Y and Z axis. The Pauli gates hence form a basis for any single qubit operation making them essential to quantum computing.

2.2.2 Different Basis States and the Hadamard Gate

In quantum computing, a basis is a set of states that can be used to represent a system's quantum state.

Applying the Pauli-Z gate to our defined states $|0\rangle$ and $|1\rangle$ seems to have no effect on a qubit while it is in either of these two states. This is because $|0\rangle$ and $|1\rangle$ are two eigenstates of the Pauli-Z gate. The states $|0\rangle$ and $|1\rangle$ form the computational basis, which is the standard basis in quantum computing [5]. Any single-qubit state can be written as a linear combination of these two states.

However, there are several different bases that are commonly used in quantum computing. Another popular basis is the X-basis, formed by the eigenstates of the Pauli-X gate. This basis is useful because it is a superposition of the computational basis states and it can be used to create entangled states. This basis is also referred to as the Hadamard basis, as the implementation of the Hadamard gate creates the aforementioned superposition of the computational basis. This can also be thought of as a rotation of $\frac{\pi}{2}$ around the Z axis of the Bloch sphere.

To create a basis, one simply need two orthogonal vectors, making there an infinite number of bases. The eigenvectors of both Hermitian and unitary matrices form a basis for their respective vector spaces. Due to this property, it is certain that the Pauli-X gate and the Pauli-Y gate also form an alternate basis for single qubit states.

2.2.3 Other Single Qubit Gates and their Properties

There is an infinite number of possible gates that can act on a qubit. Some of the more common single qubit gates are tabulated below.

Gate Name	**Corresponding Matrix**	**Description**
P-gate	$\mathrm{P}(\phi) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\phi} \end{bmatrix}$	The P-gate, also known as the phase gate, performs a rotation of ϕ around the Z-axis.
I-gate	$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	This is the Identity gate, and it has no effect on the qubit state.
S-gate	$S = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{2}} \end{bmatrix}$	The S-gate is a P-gate with $\phi = \frac{\pi}{2}$. It does a quarter turn around the Bloch sphere.
T-gate	$T = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{i\pi}{4}} \end{bmatrix}$	The T- gate is the P-gate with $\phi = \frac{\pi}{4}$.
U-gate	$U(\theta, \phi, \lambda) = \begin{bmatrix} \cos\frac{\theta}{2} & e^{-i\lambda}\sin\frac{\theta}{2} \\ e^{i\phi}\sin\frac{\theta}{2} & e^{i(\phi+\theta)}\cos\frac{\theta}{2} \end{bmatrix}$	The U-gate is the most general form of all single qubit quantum gates. Any gate can be specified as $U(\theta, \phi, \lambda)$.

Single qubit gates are reversible, meaning it is possible to apply a second gate to the qubit that undoes the effect of the first gate. This property is extremely useful in quantum computing because it ensures that no information is lost during a computation.

$$XX|0\rangle = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = |0\rangle$$

[9]

Single qubit gates are also Hermitian. This means that they are equal to their own conjugate transpose. This property ensures that the probability of measuring a particular outcome is equal to the probability of measuring the corresponding outcome with the complex conjugate of the gate applied [5].

By virtue of these two traits, it follows that single qubit gates are also unitary. A matrix is called unitary when the matrix multiplied by its transposed complex conjugate equals the Identity matrix. This enables the operation to preserve the length of the quantum state vector and ensures the probabilities of possible outcomes always add up to one.

$$UU^\dagger = U^\dagger U = I \tag{2.2.1}$$

[5]

These properties make single qubit gates effective tools for quantum computation, and allow for control over the quantum state of a qubit. This allows for the construction of complex quantum algorithms.

2.3 Multi-Qubit systems

While single qubits have some interesting properties, the true power of quantum computing is realized through the interactions between qubits.

2.3.1 Representing Multi-Qubit States

A single qubit has the potential to be in two possible states at once, a superposition of $|0\rangle$ and $|1\rangle$. Introducing an additional qubit to the system would increase the amount of possible states to four, those being $|00\rangle, |01\rangle, |10\rangle$ and $|11\rangle$. Describing the state vector of the two qubits would then require four complex amplitudes for each corresponding state [9].

$$|a\rangle = a_{00}|00\rangle + a_{01}|01\rangle + a_{10}|10\rangle + a_{11}|11\rangle = \begin{bmatrix} a_{00} \\ a_{01} \\ a_{10} \\ a_{11} \end{bmatrix}$$

[9]

If there are n qubits in a system, the system has the possibility of being in 2^n possible states, with 2^n corresponding amplitudes. The terms of the state vector will grow exponentially with the number of qubits in a system. For a system of three qubits, the corresponding state vector will have eight terms.

$$|cba\rangle = \begin{bmatrix} c_0b_0a_0 \\ c_0b_0a_1 \\ c_0b_1a_0 \\ c_0b_1a_1 \\ c_1b_0a_0 \\ c_1b_1a_1 \\ c_1b_1a_0 \\ c_1b_1a_1 \end{bmatrix}$$

[9]

The rules of measurement and normalization will still hold regardless of the number of qubits within a state.

$$p(|00\rangle) = |\langle 00|a\rangle|^2 = |a_{00}|^2 \quad , \quad |a_{00}|^2 + |a_{01}|^2 + |a_{10}|^2 + |a_{11}|^2 = 1$$

[9]

The collective state of two separated qubits can be described using the Kronecker product. This process can be used to describe the collective state of any number of qubits.

$$|a\rangle = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}, \quad |b\rangle = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}$$

$$|ba\rangle = |b\rangle \otimes |a\rangle = \begin{bmatrix} b_0 \times \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \\ b_1 \times \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} b_0a_0 \\ b_0a_1 \\ b_1a_0 \\ b_1a_1 \end{bmatrix}$$

[9]

2.3.2 Multi-Qubit Operations

Single Qubit Gates on Mult-Qubit States

Similarly, it is possible to represent the operations of single qubit gates using the Kronecker product to create matrices that correspond to the 2^n entries of a state vector.

$$X|q_1\rangle \otimes H|q_1\rangle = (X \otimes H)|q_1q_0\rangle$$

$$X \otimes H = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \otimes \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 1 \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \\ 1 \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} & 0 \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \end{bmatrix}$$

$$= \frac{1}{\sqrt{2}} \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \\ 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}$$

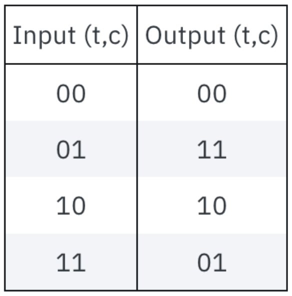
[9]

This matrix can now be applied to the 4D statevector, $|q_1 q_0\rangle$. If there is only a single qubit gate acting on one qubit within the system, the Kronecker product of said gate with the identity matrix will provide the appropriate operation.

Multi-Qubit Gates

Multi-qubit gates are gates that act on two or more qubits in a quantum circuit. These gates are used to manipulate the joint state of qubits, and are a primary example of how qubits interact with each other

The CNOT gate is a conditional gate that performs an X-gate on the second qubit, if the state of the first qubit is $|1\rangle$. The first and second qubit under this operation are often referred to as the control and target qubit respectively. When the qubits within the system are not in superposition, the CNOT gate is very simple to understand, and its operation is represented in the truth table below [9].



Input (t,c)	Output (t,c)
00	00
01	11
10	10
11	01

Figure 2.3.1: The operation of the CNOT gate based on the input states of the qubits in the system.

When acting on a 4D state vector, the CNOT gate has the below matrix representation. It will swap the amplitudes of the $|01\rangle$ and $|11\rangle$ state in the system state vector. This gate is sometimes referred to as a controlled X-gate, as it performs the X-gate on a target qubit depending on the state of the control qubit.

$$\text{CNOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

[9]

More generally, it is possible to find a controlled gate for any desired operation. For a matrix

$$\text{U} = \begin{bmatrix} u_{00} & u_{01} \\ u_{10} & u_{11} \end{bmatrix}$$

[9]

the Controlled-U gate can be found using the operation below.

$$\text{Controlled-U} = \begin{bmatrix} I & 0 \\ 0 & U \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & u_{00} & u_{01} \\ 0 & 0 & u_{10} & u_{11} \end{bmatrix}$$

A SWAP-gate is a two qubit gate that swaps the state of two qubits. It is useful for rearranging qubits in a quantum circuit. The SWAP-gate can be constructed using a combination of CNOT gates. The first CNOT gate acts on qubits 1 and 2 with qubit 2 as the control and qubit 1 as the target. The second CNOT gate acts on the same set of qubits with qubit 1 as the control and qubit 2 as the target. The third CNOT gates acts in the same way as the first. This set of operations flips the state of qubit 2 if qubit 1 is in the state $|1\rangle$, effectively swapping the state of qubit 1 and 2. [9]

$$\text{SWAP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

[9]

The Toffoli-gate is a three-qubit gate with two controls and one target. It performs an X-gate on the target only if both controls are in the state $|1\rangle$. It can also be thought of as a controlled-controlled-NOT-gate. The final state of the target is equal to either the AND or the NAND logic gate operations of the two controls depending on whether the initial state of the target was $|0\rangle$ or $|1\rangle$ [9] [5].

$$\text{Toffoli} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

[9]

2.3.3 Entanglement

Entanglement is a fundamental concept in quantum mechanics, which results from the interaction of qubits following particular multi-qubit operations. The use of a CNOT gate on a qubit in the state $|0+\rangle$ creates a state vector in which each individual state cannot be described independently of one another. This state is known as the Bell State [9].

$$\text{Bell State} : \text{CNOT}|0+\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

[9]

There is a 50% chance of the Bell State being measured in the state $|00\rangle$ and a 50% chance of it being measured in the state $|11\rangle$, but interestingly, a 0% chance of being measured in the states $|01\rangle$ or $|10\rangle$. The act of measurement then determines the states of both qubits in the system upon the collapse of it's superposition. If the first qubit was measured to be in the state $|1\rangle$, the collective state of the two qubit system would change automatically determining the state of the second qubit to be $|1\rangle$ as well.

$$\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) \longrightarrow |11\rangle$$

The correlation between two entangled qubits hold regardless of the distance between them, allowing for the quick transmission of information in a quantum system.

The mathematical foundations of quantum computing are complex. They draw on many different branches of mathematics to describe the behavior of quantum systems. Quantum circuits and algorithms are designed to take advantage of the unique properties of quantum systems as mentioned in this chapter.

3
Defining Quantum Circuits

Quantum circuit diagrams are a method of representing quantum operations on a system of qubits. They are used to design and implement quantum algorithms on quantum computers. Circuits as computational models are typically acyclic circuits, and will represent a finite sequence of operations that cannot contain feedback loops.

3.1 Quantum Circuit Diagrams

In IBM's qiskit, an open-source software development kit for working with quantum computers, the default names for qubits are q_0, q_1, q_2, etc. When there is only a single qubit, the default name for a qubit is q. For conventional purposes going forward, the topmost qubit in a circuit will have the index zero, and corresponds to the rightmost position in a tensor product. The second-to-top qubit has the index one, and corresponds to the second from right position in a tensor product. This pattern continues down to the final qubit in the system having the highest index and corresponding to the leftmost position in a tensor product.

$$|q_1\rangle \otimes |q_0\rangle = |q_1 q_0\rangle$$

Quantum circuits often have all qubits initialized in the state $|0\rangle$, but there can be instances where it is convenient to initialize the input qubits to different states. In the quantum circuit model, horizontal wires represent qubits and gate blocks represent operations acting on the qubits. The flow of information is read from left to right. In general, quantum circuits can contain any number of qubit wires. It is also possible to include classical bit wires, which are typically indicated by double lines. Quantum circuit diagrams often include measurement gates, which measure the quantum state of one or more qubits and output a classical bit [9].

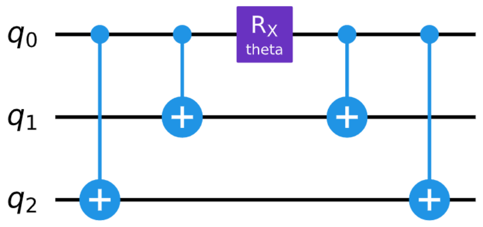


Figure 3.1.1: A representation of a three qubit circuit diagram [9].

Following are some symbols for common gates. Single qubit gates are generally shown as blocks indicating which operation is being used [9].

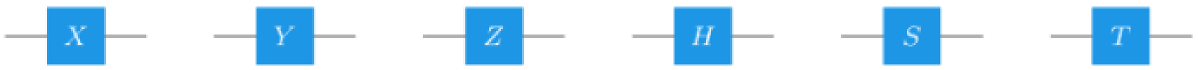


Figure 3.1.2: Circuit diagram representation of single qubit gates [9].

NOT gates, also known as X gates, are also sometimes denoted by a circle around a plus sign.

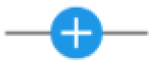


Figure 3.1.3: Alternate circuit diagram notation for an X gate [9].

Swap gates are denoted as follows.

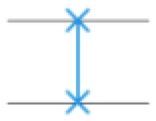


Figure 3.1.4: Circuit diagram representation for the SWAP gate [9].

Controlled gates that describe controlled unitary operations are denoted by a filled in circle (indicating the control) connected by a vertical line to the operation that is being controlled. Controlled NOT and Toffoli gates are denoted as follows.

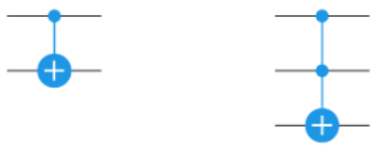


Figure 3.1.5: Circuit diagram representation of the controlled-NOT and Toffoli gate respectively [9].

Arbitrary unitary operations on multiple qubits may be viewed as gates. They are depicted by rectangles labeled by the name of the unitary operations. Below is a depiction of an unspecified unitary operation, U as a gate, along with a controlled version of the gate.

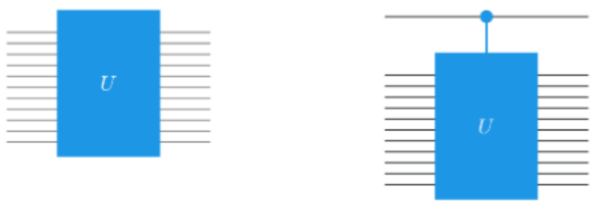


Figure 3.1.6: Circuit diagram representation of an arbitrary unitary operation [9].

3.2 Data Encoding

Data representation is crucial for the success of quantum algorithms. The problem is how to represent and efficiently input data into a quantum system, so that it can be processed by a quantum algorithm. This is typically referred to as data encoding.

Amplitude Encoding

A convenient method of encoding data is amplitude encoding. This process encodes data into the amplitudes of a quantum state. It represents a normalized classical N-dimensional data set, $\mathscr{X}$, as the amplitudes of a n-qubit quantum state $|\psi_x\rangle$ where N=2^n, x_i is the i^{th} element of $\mathscr{X}$ and $|i\rangle$ is the i^{th} computational basis state. Consider a classical dataset $\mathscr{X}$ consisting of M samples, each with N features, where $x^{(m)}$ is an N dimensional vector for m=1,..., M. To encode the dataset $\mathscr{X}$, the data points within the data set must be consolidated into an amplitude vector [9].

$$\alpha = A_{norm}(x_1^{(1)}, \ldots, x_N^{(1)}, \ldots, x_1^{(m)}, \ldots, x_N^{(m)}, \ldots, x_N^{(m)})$$

[9]

A_{norm} is a normalization constant such that $|\alpha|^2 = 1$. This allows for the data set $\mathscr{X}$ to be represented in the computational basis as:

$$|\mathscr{X}\rangle = \sum_{i=1}^{N} \alpha_i |i\rangle$$

[9]

Where α_i are the elements of the amplitude vector and $|i\rangle$ are the computational basis states.

The number of amplitudes to be encoded is $N \times M$. As a system of n qubits provides 2^n amplitudes, the process of amplitude encoding will require that n $\geq \log_2(NM)$ qubits.

The advantage of amplitude encoding is that it only requires $\log_2(NM)$ qubits to encode. This is useful in minimizing noise and extracting more precise results from a quantum computation.

For example, consider the amplitudes of an single cycle sinusoidal function, in this case, a sin wave.

Angle (θ)	Amplitude
0	0
$\pi/4$	$1/\sqrt{2}$
$\pi/2$	1
$3\pi/4$	$1/\sqrt{2}$
π	0
$5\pi/4$	$-1/\sqrt{2}$
$3\pi/2$	-1
$7\pi/4$	$-1/\sqrt{2}$

The eight data points can be represented by a three qubit system. To encode the amplitudes of the sin wave using the amplitude encoding method, we will have to first normalize the dataset.

$$A_{norm} = \frac{1}{\sqrt{(0)^2 + (1/\sqrt{2})^2 + (1)^2 + (1/\sqrt{2})^2 + (0)^2 + (-1/\sqrt{2})^2 + (-1)^2 + (-1/\sqrt{2})^2}} = \frac{1}{2}$$

$$\alpha = \frac{1}{2}\left(0,\ \frac{1}{\sqrt{2}},\ 1,\ \frac{1}{\sqrt{2}},\ 0,\ \frac{-1}{\sqrt{2}},\ -1,\ \frac{-1}{\sqrt{2}}\right)$$

Using the computational basis for three qubits, we can then represent the data set as a sum of possible states with the corresponding encoded amplitudes.

$$|\mathscr{X}\rangle = \frac{1}{2}\left(0|000\rangle + \frac{1}{\sqrt{2}}|001\rangle + 1|010\rangle + \frac{1}{\sqrt{2}}|011\rangle + 0|100\rangle + \frac{-1}{\sqrt{2}}|101\rangle + (-1)|110\rangle + \frac{-1}{\sqrt{2}}|111\rangle\right)$$

Extracting the data following a computation is then only a matter of multiplying the resulting amplitudes by the inverse of the normalization constant.

4
Quantum Fourier Transform

The Quantum Fourier transform (QFT) is a quantum algorithm that performs a Fourier transform on a quantum state. The Fourier transform is a mathematical operation that converts a function in the time domain into its frequency domain representation. The QFT is a quantum analogue of the classical Fourier transform, but operates on quantum states rather than on classical signals.

4.1 Discrete Fourier Transform

The discrete Fourier transform (DFT), the classical analog of the QFT, operates to convert a discrete sequence of data points in the time domain into a discrete spectrum of frequencies in the frequency domain.

The DFT takes a finite sequence of discrete data points as inputs and produces a finite sequence of complex numbers as outputs, which are representative of the sinusoidal components at different frequencies in the Fourier series [4].

Mathematically, given a discrete set of data points $[x_0,\ x_1,\ \ldots, x_{N-1}]$, the DFT computes the discrete spectrum of frequencies $[y_0,\ y_1\ \ldots, y_{N-1}]$ as follows:

$$y_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i k n}{N}} \tag{4.1.1}$$

[4]

N is the total number of data points in the input sequence. x_n is the input data point with an index n, where n is an integer between 0 and N-1. y_k is the complex-valued output at frequency k, where k is an integer between 0 and N-1. The complex exponential $e^{\frac{-2\pi ikn}{N}}$ is a function that represents the sinusoidal component at frequency k in the Fourier series.

The DFT will compute the amplitudes and phases of the sinusoidal components at different frequencies in the input data sequence and will output complex numbers in the output sequence y_k. With this data, the frequency content of the input signal can be extracted, and the original signal can be reconstructed in the time domain.

Consider a simple sinusoid, a 1 Hz sin wave with an amplitude of 1 V. For eight sample points, the sampling frequency will be 8 Hz. Because the Fourier transform decomposes a signal into a set of sinusoids, it is expected that there will be one amplitude with a magnitude of 1V corresponding to the 1 Hz frequency bin.

x_n	Value
x_0	0
x_1	$1/\sqrt{2}$
x_2	1
x_3	$1/\sqrt{2}$
x_4	0
x_5	$-1/\sqrt{2}$
x_6	-1
x_7	$-1/\sqrt{2}$

For the term x_0, the k value will be zero, and the DFT will simply be a sum of the terms. For the 0^{th} frequency bin, the result of the DFT is 0.

$$y_0 = \sum_{n=0}^{N-1} x_n = 0$$

For the first frequency bin, expanding the sum and using Euler's formula to write the DFT as a series of sines and cosines will yield the following result.

$$y_1 = \sum_{n=0}^{N-1} x_n e^{\frac{-i2\pi kn}{N}} = 0 \cdot e^{\frac{-i2\pi(1)(0)}{8}} + \frac{1}{\sqrt{2}} \cdot e^{\frac{-i2\pi(1)(1)}{8}} + 1 \cdot e^{\frac{-i2\pi(1)(2)}{8}} + \ldots.$$

$$= 0 + \frac{1}{\sqrt{2}}[\cos\left(\frac{-\pi}{4}\right)] + i\sin\left(\frac{-\pi}{4}\right)] + 1[\cos\left(\frac{-\pi}{2}\right) + i\sin\left(\frac{-\pi}{2}\right)] + \ldots.$$

$$= 0 + (0.5 - 0.5i) + (-i) + (-0.5 - 0.5i) + (0.5 - 0.5i) + (-i) + (-0.5 - 0.5i) = -4i$$

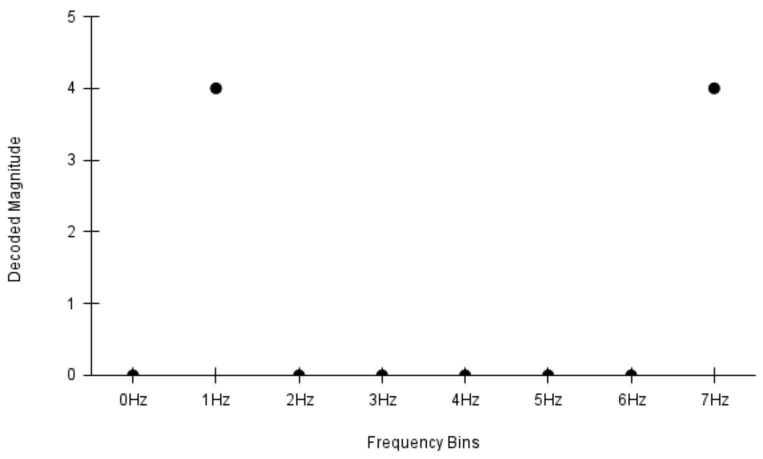
The process can be repeated for the remaining frequency bins to find the following results.

x_n	Fourier Coefficients
x_0	0
x_1	$0 - 4i$
x_2	0
x_3	0
x_4	0
x_5	0
x_6	0
x_7	$0 + 4i$

Only the first and seventh frequency bins have non-zero Fourier coefficients. Taking the square root of the complex number values squared will result in the magnitude of the term.

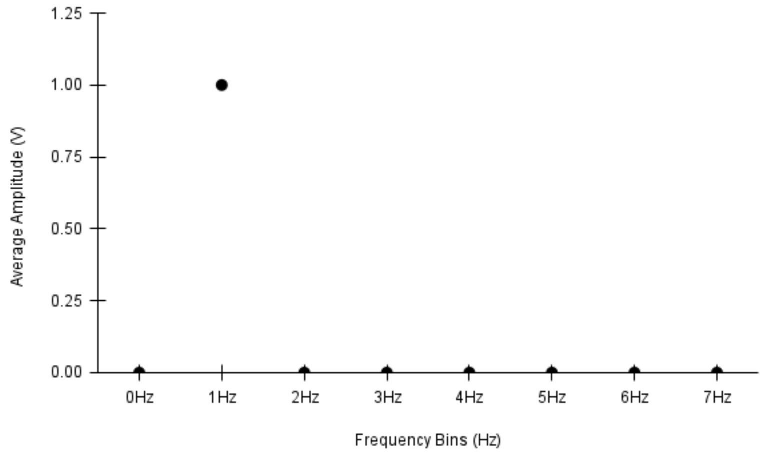
$$Magnitude = \sqrt{A_k^2 + B_k^2} = \sqrt{0^2 + (\pm 4)^2} = 4$$

A plot is useful in understanding the results. The x-axis represents the y_k terms, and the y-axis represents their corresponding amplitudes. Because the sampling frequency is 8 Hz, amd there are 8 points, the frequency resolution is 1 Hz, meaning each frequency bin increases by 1 Hz.



The above shows a two sided frequency plot. The Nyquist limit is the sampling frequency divided by two, in this case, 4 Hz. Any frequency above the Nyquist limit cannot be measured. To compile the results, the values below the Nyquist limit must be doubled.

The amplitude corresponding with the 1 Hz frequency bin is eight. To find the amplitude of the input sample, the resulting amplitude must be divided by the number of data points used to compute the DFT, in this case eight. The result is a sample containing a 1 Hz frequency and an amplitude of 1 V as originally stated.



4.2 Quantum Fourier Transform

The quantum Fourier transform is the quantum implementation of the discrete Fourier transform over the amplitudes of a wave function. The QFT will act on a quantum state $|X\rangle = \sum_{j=0}^{N-1} x_j|j\rangle$ and maps it to the quantum state $|Y\rangle = \sum_{k=0}^{N-1} y_k|k\rangle$ according to the formula

$$y_k = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} x_j w_N^{jk}$$

[9]

Where $w^{jk} = e^{2\pi i \frac{jk}{N}}$, $N = 2^n$, n is the number of qubits and k is the index of the resulting quantum state [9].

Note that only the amplitudes of the state were affected by this transformation. This can also be expressed as the map:

$$|j\rangle \longrightarrow \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} w_N^{jk} |k\rangle$$

[9]

or the unitary matrix:

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} w_N^{jk} |k\rangle\langle j|$$

[9]

Mathematically, the difference between the DFT and QFT is a factor of $\frac{1}{\sqrt{N}}$, which must be considered when decoding a classical data set following the implementation of the QFT.

Consider how the QFT operator as defined above acts on a single qubit state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$. In this case, $x_0 = \alpha,\ \ x_1 = \beta,\ $ and $\ N = 2$.

$$y_0 = \frac{1}{\sqrt{2}} \left(\alpha e^{\frac{2\pi i(0)(0)}{2}} + \beta e^{\frac{2\pi i(1)(0)}{2}} \right) = \frac{1}{\sqrt{2}}(\alpha + \beta)$$

$$y_1 = \frac{1}{\sqrt{2}} \left(\alpha e^{\frac{2\pi i(0)(1)}{2}} + \beta e^{\frac{2\pi i(1)(1)}{2}} \right) = \frac{1}{\sqrt{2}}(\alpha - \beta)$$

Such that the final result is the state:

$$U_{QFT}|\psi\rangle = \frac{1}{\sqrt{2}}(\alpha + \beta)|0\rangle + \frac{1}{\sqrt{2}}(\alpha - \beta)|1\rangle$$

This operation is interestingly exactly the result of applying the Hadamard gate on the qubit. When applying the H operator to the state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, we obtain the new state:

$$\frac{1}{\sqrt{2}}(\alpha + \beta)|0\rangle + \frac{1}{\sqrt{2}}(\alpha - \beta)|1\rangle \equiv \tilde{\alpha}|0\rangle + \tilde{\beta}|1\rangle$$

The Hadamard gate performs the DFT for N=2 on the amplitudes of the state.

For large N, the derivation of the QFT acting on a state $|x\rangle = |x_1 \ldots x_n\rangle$ is as follows [9].

$$QFT_N|x\rangle = \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} w_N^{xy} |y\rangle$$

$$= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{\frac{2\pi ixy}{2^n}} |y\rangle \text{ since } w_N^{xy} = e^{\frac{2\pi ixy}{N}} \text{ and } N = 2^n$$

$$= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} e^{2\pi i(\sum_{k=01}^{n}} \frac{y_k}{2^k} |y_1 ... y_n\rangle \text{rewriting in fractional binary notations } y = y_1 ... y_n, \ \frac{y}{2^n} = \sum_{k=1}^{n} \frac{y_k}{2^k}$$

$$= \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \prod_{k=1}^{n} e^{\frac{2\pi ixy_k}{2^k}} |y_1 ... y_n\rangle, \text{ after expanding the exponential of a sum to a product of exponentials}$$

$$= \frac{1}{\sqrt{N}} \bigotimes_{k=1}^{n} \left(|0\rangle + e^{\frac{2\pi ix}{2^k}} |1\rangle\right) \text{ after rearranging the sum and products and expanding}$$

$$= \frac{1}{\sqrt{N}} \left(|0\rangle + e^{\frac{2\pi i}{2}x}|1\rangle\right) \otimes \left(|0\rangle + e^{\frac{2\pi i}{2^2}x}|1\rangle\right) \otimes \ldots \otimes \left(|0\rangle + e^{\frac{2\pi i}{2^{n-1}}x}|1\rangle\right) \otimes \left(|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right)$$

4.3 QFT Circuit Implementation

The circuit that implements the QFT makes use of two gates, the single qubit Hadamard gate and the two qubit controlled rotation $CROT_k$. The $CROT_k$ gate is a controlled phase rotation gate where $\theta = \frac{2\pi}{2^k} = \frac{\pi}{2^{k-1}}$.

$$CROT_k = \begin{bmatrix} I & 0 \\ 0 & UROT_k \end{bmatrix} \quad UROT_k = \begin{bmatrix} 1 & 0 \\ 0 & e^{\frac{2\pi i}{2^k}} \end{bmatrix}$$

The action of of the $CROT_k$ on a two qubit state $|x_l x_j\rangle$ where the first qubit is the control and the second is the target is given by

$$CROT_k|0x_j\rangle = |0x_j\rangle$$

And

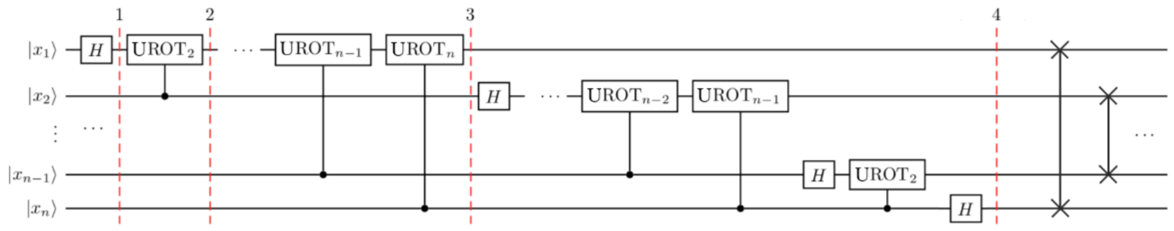
$$CROT_k|1x_j\rangle = e^{\frac{2\pi i}{2^k}x_j}|1x_j\rangle$$

[9]

The circuit implementation is show below:

After the first Hadamrd gate on qubit 1, the state is transformed from the input state to:

$$H_1|x_1x_2 ... x_n\rangle = \frac{1}{\sqrt{2}} \left[|0\rangle + e^{\frac{2\pi i}{2}x_1}|1\rangle\right] \otimes ||x_2x_3 ... x_n\rangle$$



After the $UROT_2$ gate on qubit 1 controlled by qubit 2, the state is transformed to:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^2}x_2 + \frac{2\pi i}{2}x_1}|1\rangle\right] \otimes ||x_2x_3\ldots x_n\rangle$$

After the application of the last $UROT_n$ gate on qubit 1, controlled by qubit n, the state becomes:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^n}x_n + \frac{2\pi i}{2^{n-1}}x_{n-1} + \ldots + \frac{2\pi i}{2^2}x_2 + \frac{2\pi i}{2}x_1}|1\rangle\right] \otimes ||x_2x_3\ldots x_n\rangle$$

Noting that

$$x = 2^{n-1}x_1 + 2^{n-2}x_2 + \ldots + 2^1x_{n-1} + 2^0x_n$$

The above state can be written as

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right] \otimes ||x_2x_3\ldots x_n\rangle$$

After the application of a similar sequence of gates for the remaining qubits, the final state is:

$$\frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^n}x}|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^{n-1}}x}|1\rangle\right] \otimes \ldots \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^2}x}|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[|0\rangle + e^{\frac{2\pi i}{2^1}x}|1\rangle\right]$$

which is exactly the QFT input state as previously derived 4.2 [9].

It is useful to note that only the last qubit depends on the values of all the other input qubits and each further bit depends less and less on the input qubits. This becomes physically important in implementations of the QFT where nearest-neighbor couplings are easier to achieve than distant couplings between qubits.

5
Implementation on the IBM Quantum Computer

5.1 Physical Quantum Computing Systems

IBM has created a platform in which the general public has access to their various quantum processors. This platform allows users to run quantum computing experiments on real IBM quantum devices via the cloud, as well as simulate quantum circuits on classical computers. The five and seven qubit computers are among the IBM quantum devices that have been made public.

The available gates in IBM's graphical interface include [5] :

$$\{I,\ X,\ Y,\ Z,\ H,\ S,\ S^\dagger,\ T,\ T^\dagger,\ U_1(\lambda),\ U_2(\lambda,\phi),\ U_3(\lambda,\phi,\theta),\ CNOT\} \tag{5.1.1}$$

The graphical user interface also provides other controlled gates and operations like measurement and reset. The gates $U_1(\lambda)$, , $U_2(\lambda,\phi)$, and $U_3(\lambda,\phi,\theta)$ are continuously parameterized gates as follows [5].

$$U_1(\lambda) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\lambda} \end{bmatrix}, \quad U_2(\lambda,\phi) = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & e^{-i\lambda} \\ e^{i\phi} & e^{i(\lambda+\phi)} \end{bmatrix}, \quad U_3(\lambda,\phi,\theta) = \begin{bmatrix} \cos\frac{\theta}{2} & -e^{i\lambda}\sin\frac{\theta}{2} \\ e^{i\phi}\sin\frac{\theta}{2} & e^{i(\lambda+\phi)}\cos\frac{\theta}{2} \end{bmatrix}$$

[5]

The gates listed in 5.1.1 are provided for facility and convenience, but are not the gates that are physically implemented. The physical gate set employed by IBM quantum computers is composed of three gates [5]:

$$\{U_1(\lambda),\ R_x(\pi/2),\ CNOT\} \tag{5.1.2}$$

.

Where $R_x(\pi/2)$ is a rotation by an angle $\frac{\pi}{2}$ of the qubit about it's X-axis.

$$R_x\left(\pi/2\right) = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & -i \\ -i & 1 \end{bmatrix}$$

[5]

The gates in 5.1.1 are decomposed into the physical physical gates listed in 5.1.2 and then implemented by the quantum computer.

Most algorithms are typically written for fully connected hardware, meaning an n qubit gate can act on n qubits. In practice, real quantum computers may not have full connectivity. Fortunately there are ways to effectively generate connections through particular gate sequences. For example, a CNOT gate with the qubit j as the control and qubit k as the target can be reversed through a process called phase kickback. This would make the j qubit the target and k qubit the control. This is possible by applying a Hadamard gate on each qubit both before and after use of the CNOT gate.

$$CNOT_{jk} = (H \otimes H)CNOT_{jk}(H \otimes H)$$

. [5]

Similarly, there exists a gate sequence to make A CNOT between qubits j and l if one has connections between j and k, and k and l as follows.

$$CNOT_{jl} = CNOT_{kl}CNOT_{jk}CNOT_{kl}CNOT_{jk}$$

. [5]

These processes can make up for a lack of connectivity at the expense of using additional gates [5].

When implementing a quantum algorithm, it is important to consider the sources of noise in the computer. The two main sources of noises are typically gate infidelity and decoherence. Gate infidelity refers to the fact that the use-specified gates do not precisely correspond to the physically implemented gates. Gate infidelity is usually worse for multi-qubit gates than for single-qubit gates, so it is ideal to minimize the number of multi-qubit gates in an algorithm. Decoherence refers to the fact that gradually over time, the quantum computer begins to act more like a classical object. This effect is more prominent in systems with a larger amount of qubits. After decoherence has fully occurred, the computer can no longer take advantage of quantum effects. As the quantum algorithm advances in time, this gradually adds more noise to a system [5] .

5.2 Programming the QFT on a Quantum Computer

Qiskit is an open source quantum computing library. It allows users to write and run programs on either IBM's quantum processors or on a local simulator without the use of the graphical interface. Qiskit can be viewed as a python library for quantum circuit execution.

A basic Qiskit code has two parts, designing the circuit and running it. In the circuit design phase, a quantum circuit is created with the required number of qubits and classical bits. Then gates and measurements are added to the blank circuits. After the circuit has been designed, a backend needs to be chosen to run the circuit. This can be either one of IBM's processors or a simulator.

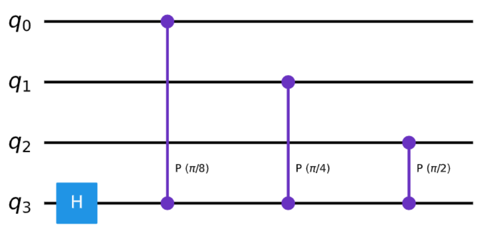
When building a circuit that implements the QFT it is easier to have the qubits upside down and then swap them afterwards. The first step is to create a function that appropriately rotates the qubits.

```python

def qft_rotations(circuit, n):
    if n == 0: # Exit function if circuit is empty
        return circuit
    n -= 1 # Indexes start from 0
    circuit.h(n) # Apply the H-gate to the most significant qubit
    for qubit in range(n):
        # For each less significant qubit, we need to do a
        # smaller-angled controlled rotation:
        circuit.cp(pi/2**(n-qubit), qubit, n)
```

[9]

On a circuit of 4 qubits, qft_rotations() can be represented in the following way:



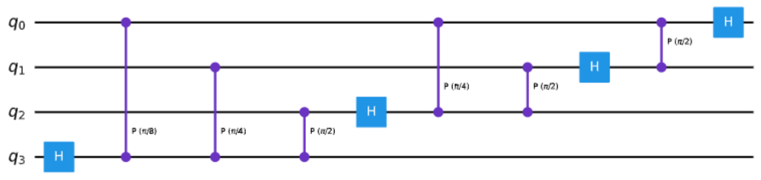
[9]

Now that the most significant qubit has been correctly rotated, the same process must be repeated for the second most significant qubit, the third, and so on. When the qft_rotations() function comes to an end, we can use the same code on the following n-1 qubits. This process is called recursion.

```

def qft_rotations(circuit, n):
    """Performs qft on the first n qubits in circuit (without swaps)"""
    if n == 0:
        return circuit
    n -= 1
    circuit.h(n)
    for qubit in range(n):
        circuit.cp(pi/2**(n-qubit), qubit, n)
    # At the end of our function, we call the same function again on
    # the next qubits (we reduced n by one earlier in the function)
    qft_rotations(circuit, n)
```

[9]



[9]

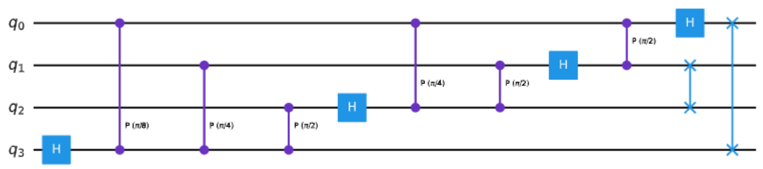
To match the definition of QFT, swap operators must be added to the end of the function. The final function is labeled qft().

```python

def swap_registers(circuit, n):
    for qubit in range(n//2):
        circuit.swap(qubit, n-qubit-1)
    return circuit

def qft(circuit, n):
    """QFT on the first n qubits in circuit"""
    qft_rotations(circuit, n)
    swap_registers(circuit, n)
    return circuit
```

[9]



This is the generalized circuit for the quantum Fourier transform [9].

5.3 Performance of the Quantum Fourier Transform

To enact the QFT, one must define a quantum circuit for it to act on. We will consider a single frequency signal as well as a signal with two frequencies and study the accuracy of the QFT in determining the input frequency and amplitude from data extracted in the time domain of each.

5.3.1 Single Frequency Signal

Consider a sinusoid with a frequency of 1 Hz and an amplitude of 1 V. The sample was measured over one second, resulting in a sampling frequency corresponding to number of samples, N, per second, a Nyquist limit of 1/sampling frequency, and a frequency output resolution of the sampling frequency divided by the number of samples, N.

The QFT circuit was implemented on both the simulator as well as a variety of IBM's quantum processors. The sinusoid was divided into 2^n points with n being the number of qubits in the circuit. After using amplitude encoding to encode the data in the quantum system, the circuit was run for 2, 3, 4, 5, 6 and 7 qubit systems.

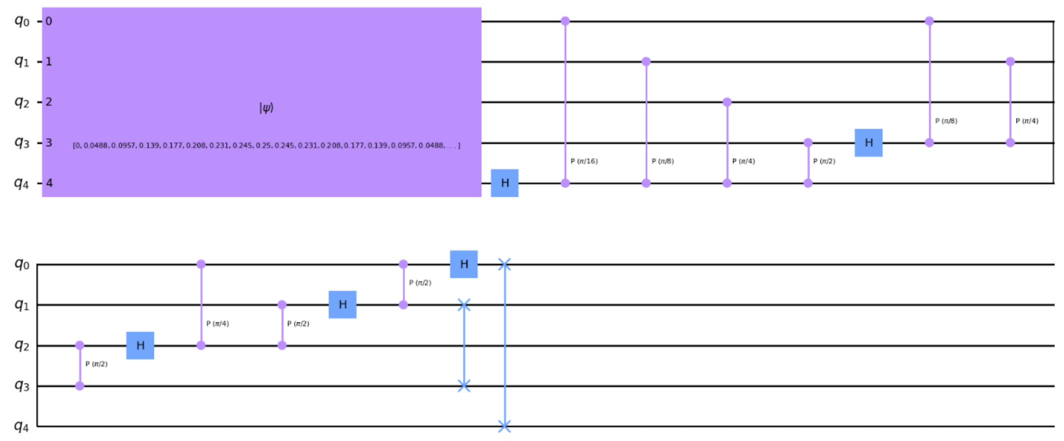
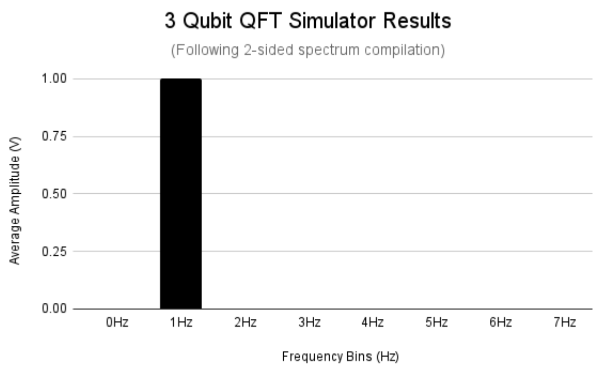


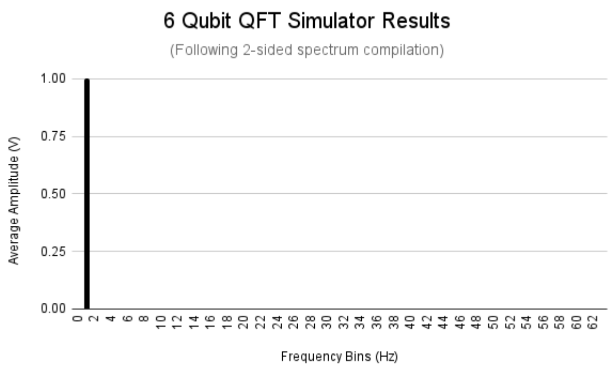
Figure 5.3.1: The above circuit diagram represents the implementation of the QFT on a 5 qubit system [9].

Each setup of n qubits was run on the least busy quantum processor 5 times, each with the maximum number of shots, 20000. The results were averaged and compared to those of the DFT, and QFT simulator.

As seen in figure 4.1 the output of the function should result in a frequency of 1 Hz and an amplitude of 1 V. The results of the QFT verified that prediction.



(a) QFT simulator results for a 3 qubit system.

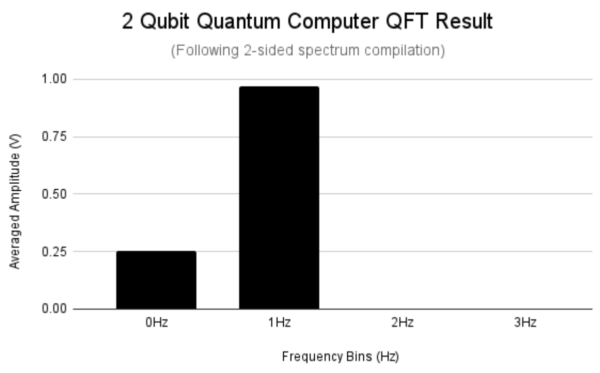


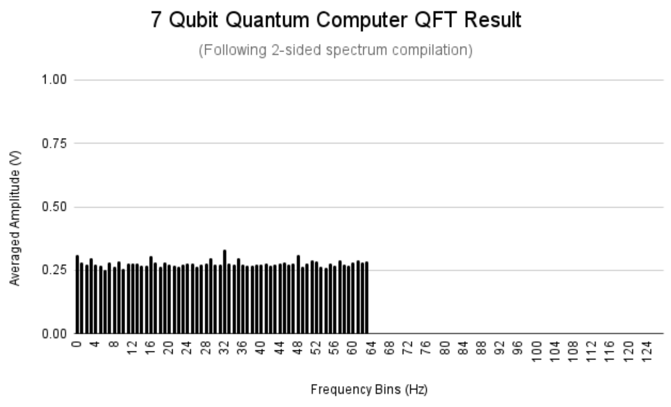
(b) QFT simulator results for a 6 qubit system.

When implemented on IBMs quantum processors, the results compared to the QFT simulator increased in percent error with the number of qubits within the system.

Number of Qubits	Percent Error
2	3.25%
3	16.66%
4	47.45%
5	48.47%
6	64.24%
7	71.89%

The systems containing less qubits resulted in more meaningful outputs compared to the predicted results. The sources of noise within the quantum computer prevented any significant results for systems of 4 qubits or more.





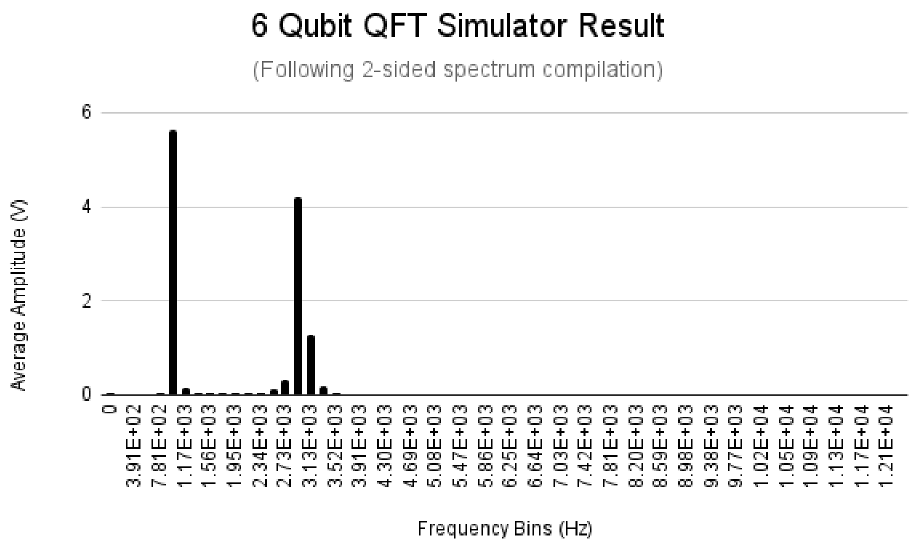
One can see a correlation favoring the 1 Hz frequency bin in a 2 qubit system while in the 7 qubit system there is no significant result.

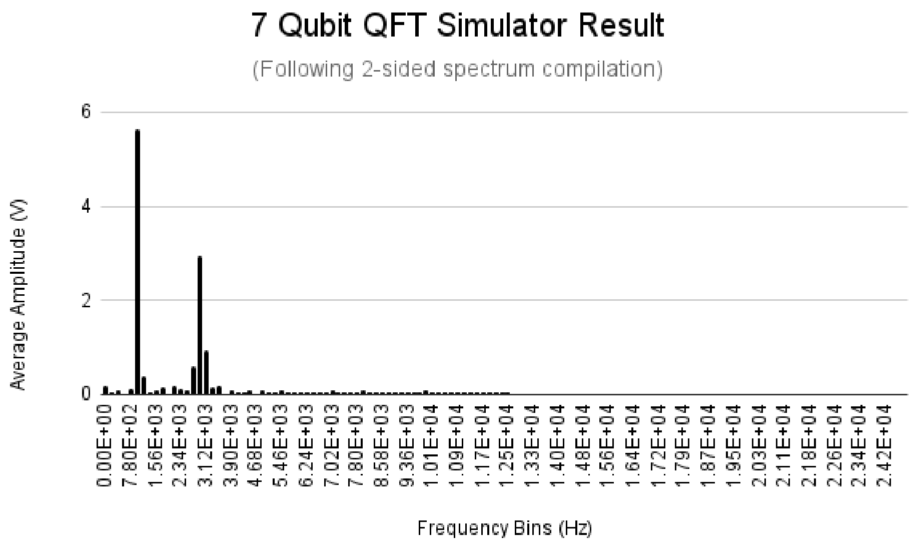
5.3.2 Dual Frequency Signal

Data from a signal containing sinusoids with frequencies of 3 kHz and 5 kHz respectively, both with 5Vpp amplitudes was taken at increments of 5.00E-06 seconds. The initial data set was minimized to correspond to 6 and 7 qubit systems. Less than 64 data points would inaccurately represent the entirety of the signal.

The QFT was run on each n qubit system 10 times, each with 20000 shots, and averaged to compare with the results of the simulator. The sampling frequency, Nyquist limit, and frequency output resolution corresponded to the definitions above. 5.3.1

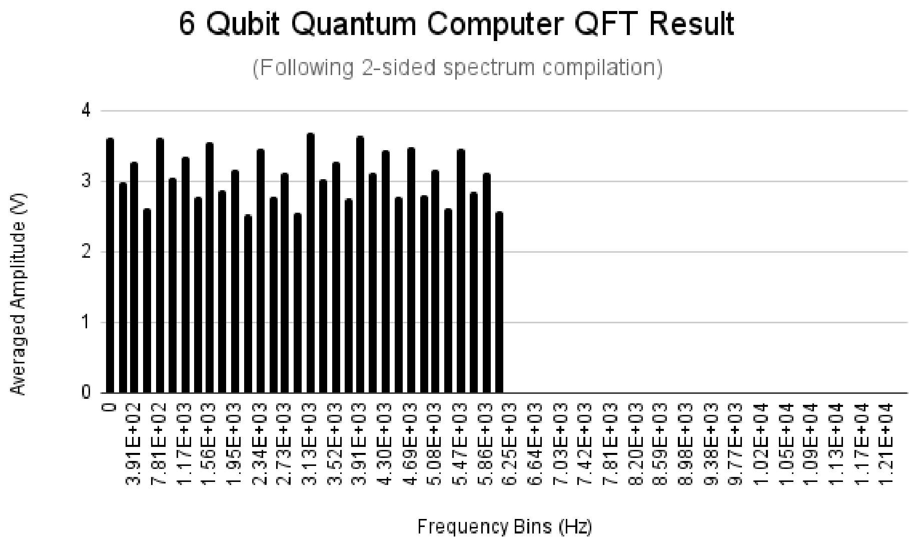
Following implementation of the QFT, the simulator was able to associate the appropriate probabilites with the input frequencies of the sinusoid.

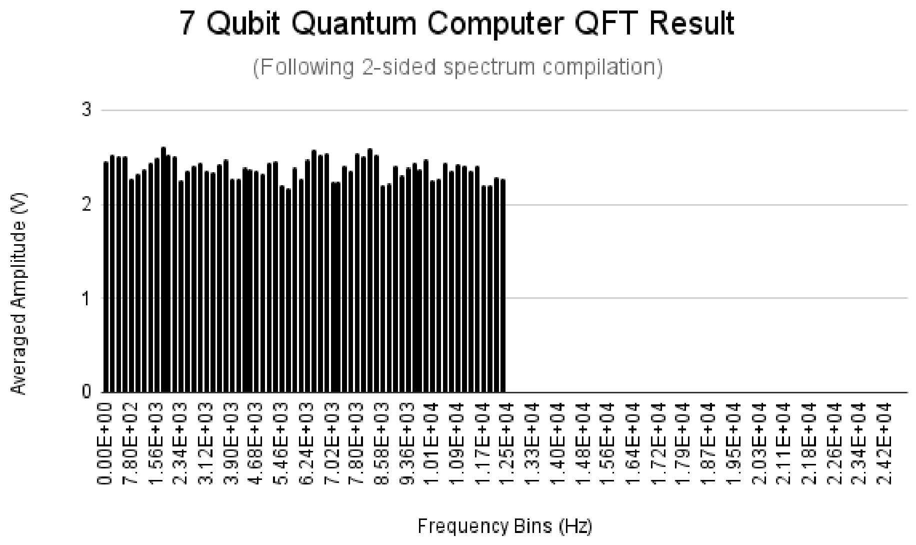




The highest probabilities indicate a frequency of 1 kHz and 3 kHz within the signal. Further analysis could allow for the extraction of these values along with the input amplitudes with more precision.

As seen above, implementation of quantum systems with greater than 3 qubits resulted in significant percent error, and insignificant outputs. The quantum computer was unable to produce any notable results for the 6 and 7 qubit cases representing the dual frequency signal.





The IBM quantum computing simulator was able to accurately demonstrate the QFT for signal analysis and yield significant results consistent with the results from the discrete Fourier transform. For a small number of qubits, results from the quantum processors had small percent errors. As the number of qubits grew within a system, the results of the QFT became insignificant due to interruptions from noise.

Quantum computers are inherently prone to error due to imperfect hardware, and more frequently, environmental interference. The qubits in the measured systems were initialized in a state of superposition to accurately represent the classical data inputted. When a quantum system interacts with its environment, it loses its coherence, or ability to maintain superposition. Decoherence causes errors within the computation, especially when the number of qubits increases due to the exponential increase of complexity within the system.

There are various error correction methods to address this issue that can help mitigate the effects of errors and enable larger scale quantum computation. With the proper error correction technique, the QFT is an effective method of extracting frequencies from a signal.

6
Conclusion

Quantum computation utilizes the principles of quantum mechanics, such as superposition and entanglements, as a basis for computation. With the use of these properties, quantum computers have the potential to have exponential speed up over classical computers. This thesis served as an introduction to the fundamentals of quantum computing and a study of the efficacy of the quantum Fourier transform. The mathematical foundations of the quantum computing model were described to gain a deeper understanding of these properties and the functionality of a quantum computer. The quantum Fourier transform was introduced and implemented on a classical data set using IBM's simulators and quantum processors for a variety of quantum systems. The QFT simulator produced results consistent with the discrete Fourier transform, confirming its potential to be an effective method in extracting frequencies from a data set in the time domain. The quantum processors were able to produce results with small percent error for systems with a small amount of qubits, but increased in error as the number of qubits within the system grew. Quantum computers are inherently prone to error and as the complexity of the system increases, so does the rate for decoherence. Methods of error correction are necessary to account for these sources of error and to enable larger scale quantum computation.

Quantum computing is still within its beginning stages of development, but has the potential to be an increasingly effective method of computation. The ability to perform complex calculations and process large amounts of data in parallel using quantum states could have the ability to transform many industries. Further research is needed for quantum computing to become a practical and widely used technology, but the progress made in recent years is promising, exciting and full of potential.

Bibliography

[1] *Ibm*. Accessed on April 4, 2023.

[2] K. Baclawski, *The observer effect*, 2018 ieee conference on cognitive and computational aspects of situation management (cogsima), 2018, pp. 83–89.

[3] T.S. Blyth and E.F. Robertson, *Basic linear algebra*, Springer Undergraduate Mathematics Series, Springer London, 2002.

[4] Mary L Boas, *Mathematical methods in the physical sciences*, John Wiley & Sons, 2006.

[5] Patrick J. Coles, Stephan J. Eidenbenz, Scott Pakin, Adetokunbo Adedoyin, John Ambrosiano, Petr M. Anisimov, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo N. Djidjev, David Gunter, Satish Karra, Nathan Lemons, Shizeng Lin, Andrey Y. Lokhov, Alexander Malyzhenkov, David Dennis Lee Mascarenas, Susan M. Mniszewski, Balu Nadiga, Dan O'Malley, Diane Oyen, Lakshman Prasad, Randy Roberts, Philip Romero, Nandakishore Santhi, Nikolai Sinitsyn, Pieter Swart, Marc Vuffray, Jim Wendelberger, Boram Yoon, Richard J. Zamora, and Wei Zhu, *Quantum algorithm implementations for beginners*, CoRR **abs/1804.03719** (2018), available at `1804.03719`.

[6] T. L. Dimitrova and A. Weis, *The wave-particle duality of light: A demonstration experiment*, American Journal of Physics **76** (2008), no. 2, 137–142, available at `https://doi.org/10.1119/1.2815364`.

[7] John Earle Gerald A. Maley, *The logic design of transistor digital computers*, Prentice-Hall, 1963.

[8] Roland Omnès, *Consistent interpretations of quantum mechanics*, Rev. Mod. Phys. **64** (1992Apr), 339–382.

[9] Qiskit contributors, *Qiskit: An open-source framework for quantum computing*, 2023.

[10] J. Q. You and Franco Nori, *Superconducting circuits and quantum information*, Physics Today **58** (2005nov), no. 11, 42–47.