Quantifying the Noise Tolerance of Quantum Factoring: A Study of Shor's Algorithm

MSc Optoelectronic and Quantum Technologies Dissertation

Department of Electrical and Electronic Engineering
UNIVERSITY OF BRISTOL

By

Tung Yan YEUNG
Student ID: mh23476

A MSc dissertation submitted to the University of Bristol in accordance with the requirements of the degree of MASTER OF SCIENCE IN OPTOELECTRONIC AND QUANTUM TECHNOLOGIES in the Department of Electrical and Electronic Engineering.

September 5, 2024

DECLARATION AND DISCLAIMER

Acknowledgment

I want to sincerely thank Dr. Imad Faruque and Dr. Jorge Barreto for their excellent advice and assistance with my research endeavor. Their knowledge, support, and enlightening feedback have been crucial in determining the course of this effort.

I am appreciative of the time they took to mentor me and the knowledge they shared. Their efforts have made this journey a wonderful experience and have substantially improved my comprehension of the subject topic.

I appreciate your unconditional support, both of you.

Abstract

The growing dependence on RSA encryption for online security is threatened by the potential of large-scale quantum computers using Shor's algorithm. This dissertation examines how hardware imperfections affect the performance of Shor's algorithm, particularly focusing on depolarising, thermal relaxation, and readout errors in superconducting quantum system. By utilising quantum computing simulator along with the actual IBM quantum processing units (QPUs) backend error rates, the study evaluates the impact of the custom noise models on the algorithm's ability to factor integers. The findings reveal that depolarising noise significantly hinders computational accuracy, suggesting that current quantum devices are unlikely to outperform classical computers in integer factorisation. This research emphasises the need for further exploration of error correction techniques to address these challenges, highlighting that achieving effective RSA-2048 decryption remains a distant goal.

Contents

List of Figures

List of Tables

1 Introduction

RSA is a popular cryptographic scheme that relies on the difficulty of factoring large composite numbers, which are the product of two prime numbers. RSA is most known for ushering the era of secured web browsing. It enabled the widespread transition from the unencrypted Hypertext Transfer Protocol (HTTP) to the Hypertext Transfer Protocol Secure (HTTPS) standard emerging the Secure Socket Layer (SSL) or Transport Layer Security (TLS), which encrypts web traffic and protects user data [1]. RSA encryption has since become essential for numerous digital applications, from secure communications to VPNs, online transactions, and etc. The widespread adoption of RSA-based security has been crucial in building trust and confidence in our daily internet-based activities. However, the potential development of large-scale quantum computers running Shor's algorithm poses a long-term threat to the continued viability of RSA encryption[2].

Noise-free quantum computers hold the potential to provide substantial speedups over classical computing, generating interest across numerous scientific and technological domains. However, achieving quantum speedup for meaningful problem sizes with Shor's algorithm presents a significant challenge. Current noisy and imperfect quantum systems adversely affect algorithm performance. Therefore, understanding the sensitivity of Shor's algorithm to various noise sources will be crucial for assessing the practical feasibility of using quantum computers to break RSA encryption in the future.

This software simulation based research provides insights into how different types of imperfections, known as noises, influence the accuracy of Shor's algorithms on one of the popular Noisy Intermediate-Scale Quantum (NISQ) devices which is the IBM Quantum Processing Units (QPUs), holistically, as well as its ability to successfully factor large numbers - a key capability for breaking RSA encryption. These findings contribute to a better understanding of the practical challenges that must be overcome before quantum computers can be leveraged to pose a credible threat to the security provided by RSA and lay the groundwork for future advancements in quantum cryptography and secure communications.

1.1 Aims

The aims of this dissertation fall into two primary categories: literature review and research phase . The specific aims of literature review are to investigate what hardware imperfections would be able to limit Shor's algorithm's performance, as well as the error rate's range and how the algorithm has been applied to quantum circuits. These aims aid in the creation of a comprehensive conceptual framework that serves as the basis for the ensuing research project. The main aim of the research phase is to use the quantum computing simulator to access locally and through the quantum cloud computing platform in order to run Shor's algorithm by manually introducing noise and conducting experiments using the real QPUs' noise models. This entails looking at the processing capacity of a quantum cloud computing platform to carry out Shor's algorithm. Examining the effect of noise in quantum systems on Shor's algorithm's success rate is another crucial goal. The final aim of the research phase is to propose an outline of how error correction can be done on quantum gate to mitigate noise impacts, along with a corresponding validation method.

1.2 Objectives

The following are the objectives of this research project:

- Model the imperfections of the current superconducting quantum computer architecture in implementing Shor's algorithm.
- Investigate the performance of Shor's algorithm under imperfections and then explore the computational ability achievable for factoring on superconducting quantum simulators

2 Related Work

In the world of cryptography, mathematician Peter Shor's 1994 introduction of Shor's algorithm raised serious concerns. RSA encryption techniques may face difficulties from this factorisation-optimized algorithm [2]. Large-scale quantum computers have the potential to seriously undermine RSA security, according to the paper.

Shor's algorithm works on gate-based quantum computers, but it may not be the quickest way to solve the factorisation problem on quantum devices at the moment. For factorisation workloads, quantum annealing appears to outperform gate-based quantum computing by a large margin. The purpose of quantum annealers, a kind of analogue quantum computer, is to solve certain optimisation issues by the application of Ising Hamiltonians [3]. These optimisation challenges include data processing, chemical similarity in chemistry, search engine ranking, and classification tasks [4]. In 2014, a breakthrough was achieved when a team of researchers factored $N = 143$ with just four qubits. This approach offers an alternative to Shor's algorithm by transforming the factorisation problem into a binary optimisation problem, which is then solved via quantum annealing. As per a study [5], 56153 was the greatest number factored during that period. Using a D-Wave quantum annealer, a genuine physical system, the largest number factored to date is 249919, building on this methodology [6]. Given that gate-based quantum computers require complete error correction in order to factor big numbers and pose a risk to RSA encryption, it is possible that a quantum annealer will be the first to solve the factorisation problem.

The fact that Shor's method can factor big numbers in polynomial time makes it especially important nowadays [2]. Although there are significant differences between the numbers factored using quantum annealers and the ones factored in this work using the shorgpu simulator, 549,755,813,701, remain significantly smaller. [7]. The shorgpu simulator was created expressly to run the iterative Shor algorithm in massively parallel across several GPUs. Even though this effort made use of contemporary computer GPUs, the outcomes give hope for the future use of Shor's algorithmâwhich leverages quantum computationâagainst RSA encryption.

Experiments implemented on small scale quantum-gate based circuits have shown that Shor's method can successfully factor small composite numbers like $N = 15$ and 21 [8, 9]. These continue to be the biggest numbers factored on quantum devices using this method as of right now. In another study [10], it demonstrates that by combining methods from nine different approaches, including Shor's algorithm, 2048-bit RSA numbers can be factored in about eight hours using 20 million noisy qubits.

Shor's algorithm gives rise to concerns over the possible weaknesses of RSA encryption as the field of quantum computing technology develops. Therefore, the purpose of this study is to assess how well-suited modern quantum simulators are for breaking 2048-bit RSA, particularly when using Shor's approach. The purpose of this study is to demonstrate how vulnerable RSA-based cryptography systems are to the strength of quantum algorithms.

3 Methods

In this study, we investigate the performance of Shor's algorithm implemented on an IBM quantum simulator. Quantum simulator provides sandbox environment for iterative development, testing, and debugging of the algorithm before attempting to run on actual high-cost quantum hardware. The selection of the IBM Quantum simulator for this study was primarily driven by availability of large numbers of qubits compared to other quantum cloud computing platform given qubit count is a major consideration factoring larger numbers. The Amazon Braket SV1 simulator, Google QSIM simulator, and IBM Qiskit-Aer simulator can accommodate 34[11], 40[12], and 63 qubits[13], respectively. Additionally, the IBM Quantum platform is widely adopted and established within the quantum computing research community, offering a certain level of standardisation.

3.1 Quantum Simulator

The IBM Quantum simulator utilises the AerSimulator from the Qiskit Aer package at version 0.13.3, running on a Python 3.11.5 environment with the Qiskit library at version 0.45.2 and qiskit-ibmq-provider at version 0.20.2.

3.2 Experimental Setup

The flowchart of the experimental steps is summarised in Figure 1.

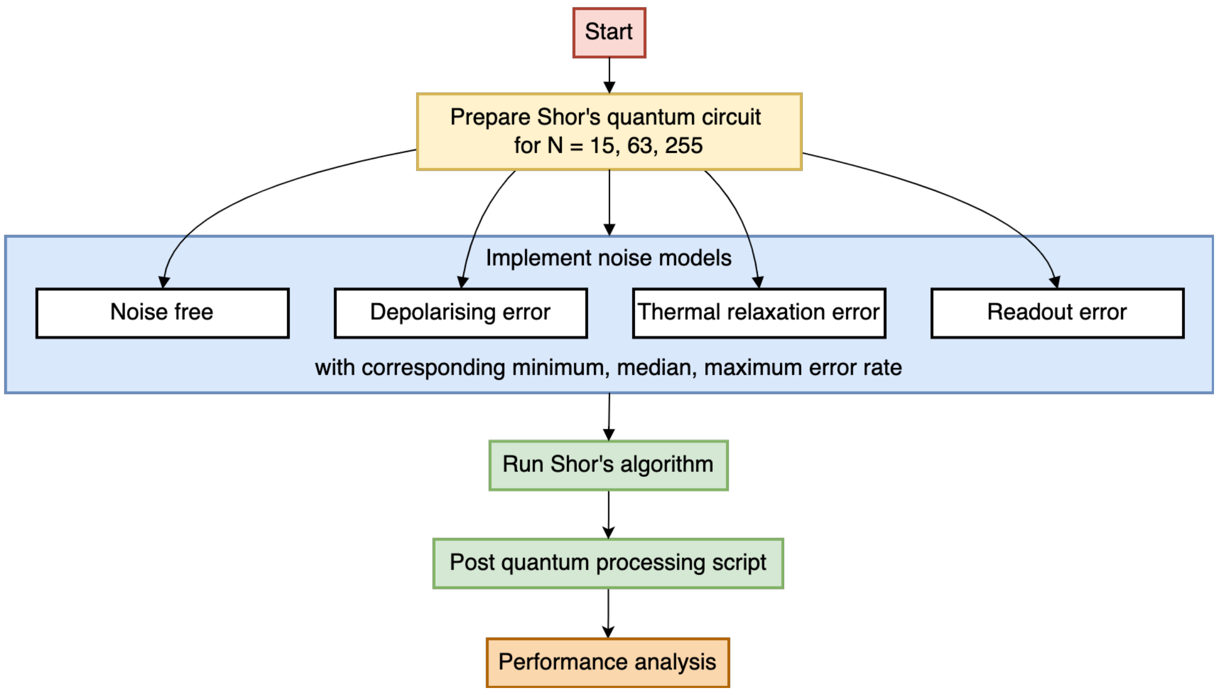
Start
Prepare Shor's quantum circuit for N = 15, 63, 255
Implement noise models
Noise free
Depolarising error
Thermal relaxation error
Readout error
with corresponding minimum, median, maximum error rate
Run Shor's algorithm
Post quantum processing script
Performance analysis

Figure 1: Flowchart of experimental flow

The approach is to attempt factoring the value of N starting from binary: 1111 (decimal: 15) and then incrementing the value by adding double 1s, i.e., binary: 111111 (decimal: 63), and binary: 11111111 (decimal: 255). This uniform step size in the growth of bits for N in binary is expected to maintain consistency in the experimental conditions. A noise-free model is first being implemented to assess the success rate of factoring for each value of N.

Following the construction of Shor's quantum circuit developed for the three values of N, noise-free along with three noises are chosen to study their impacts on the factoring success rate. Specifically, depolarising error, thermal relaxation error, and readout error shall be implemented, and their impacts and trends on the factoring success rate for each value of N will be assessed. The three noise models are built by customising the gate errors based on real quantum device characteristics to create a noisy simulator backend. Next, transpile the circuit for the backend to ensure it is converted to the appropriate noisy basis gate set specific to that backend. Finally, the outcomes are processed using a post-quantum processing procedure and a data processing script to deduce the success rate, enabling a comprehensive analysis of the final organised results.

Since IBM quantum simulator has a maximum shot count of 8192, running simulations with maximum 8192 shots significantly increases the computational time required, in essence, choosing a lower number of shots is a practical compromise between accuracy and efficiency. It allows for a good balance between the quality of the results and the time required to obtain them. To determine the optimal shot count for the developed factoring experiments, a series of trials were conducted using varying shot numbers, i.e. 64, 128, 256, 512, 1024, 2048, 4096 and 8192. Factoring experiments on the numbers 15, 63, and 255 consistently demonstrated that the success rate plateaued at 1024 shots and remained stable up to the maximum of 8192 shots as shown in Figure 2. Consequently, 1024 shots were chosen as the standard for all subsequent experiments in this research project.

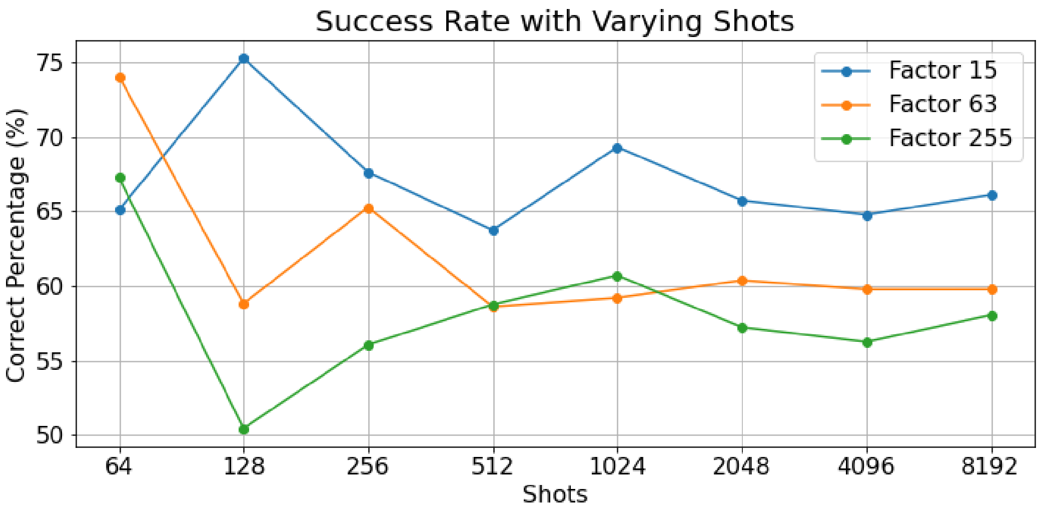

Figure 2: Factoring success rate with varying shots

3.3 Data Preparation and Collection

To implement custom noise models, minimum, median, and maximum values for relevant parameters are required. These parameters include readout assignment error rate, T1 relaxation constant, T2 dephasing constant, gate time and depolarising error rate.

Values of readout assignment error, T_1 relaxation constant and T_2 dephasing constant were obtained from the official IBM public calibration data for 15 QPUs[14], as detailed in Tables 7-8. Within the 15 QPUs datasets, the minimum readout error was observed on ibm_kyiv (0.0010), while the maximum was found on ibm_osaka (0.4931). The minimum T_1 relaxation constant was measured on $ibm\_kawasaki$ (0.7800 μs), with the maximum occurring on $ibm\_brussels$ (629.5500 μs). The minimum T_2 dephasing constant was observed on $ibm\_brisbane$ (0.2100 μs), and the maximum was found on ibm_kyiv (578.2400 μs). To determine the median values, histograms showing the distribution of 15 QPU data were plotted in Figure 3. The median values for readout error, T_1, and T_2, as determined from these histograms, were 0.015, 300 μs, and 135 μs, respectively.

Values of depolarising error rate and gate time were obtained from relevant research findings. In a research paper that demonstrates the relationship between the depolarising noise level and the achievable error using IBM quantum devices, the depolarising noise model has been scaled from 10^-5 to 10^0[15]. Since $10^0 = 1$ means absolute error rate, in this research, depolarising error rate is selected to simulate from 10^-5 to 10^-2. As Dahlhauser and Humble (2021) showed, average depolarising error rates on X gate is 0.0033[16]. This value is selected to be the median depolarising error rate in this research experiments. Another research states IBM quantum devices average one qubit gate time is $0.07\mu s$ and two qubit gate time is $0.559\mu s$[17]. The gate time values are used in this research to implement thermal relaxation error noise model and maintain unchanged.

A table below 1 summarises the chosen values for minimum, median and maximum error rates for the three selected noise channels and the related parameters.

		Minimum	Median	Maximum
Readout Error		0.0010	0.015	0.4931
Thermal Relaxation Error	T1 (μs)	0.7800	300	629.5599
	T2 (μs)	0.2100	135	578.2400
	one qubit gamte time (μs)	/	0.07	/
	two qubit gate time (μs)	/	0.559	/
Depolarising Error		0.00001	0.0033	0.01

Table 1: Summary of error rates and related parameters to implement noise models

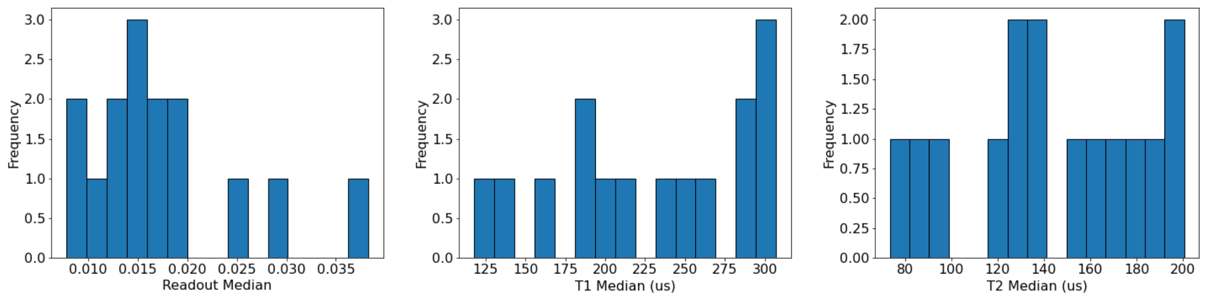

Figure 3: Histograms of median values of readout error, T_1 and T_2 data for IBM QPUs

4 Modelling noise

IBM's noise model represents quantum gate errors with a hierarchical architecture. Single-qubit gates are modeled by first applying a depolarising channel, followed by a thermal relaxation channel. Two-qubit gates are modeled similarly, with a depolarising channel applied to the two-qubit system, followed by independent thermal relaxation channel on each individual qubit. Finally, after all gate errors are applied, the model includes single-qubit readout errors at the stage of measuring the state of a qubit.[18] This hierarchical structure is the default noise models developed for all IBM's QPUs, as a result, in this work, the three different types of errors are chosen to be implemented to analyse the noise impacts are: (1) depolarisation error, (2) thermal relaxation error, and (3) readout error.

4.1 Depolarising Error

Depolarising error is modelled by the following Kraus matricess (Nielsen and Chuang, 2000)[19] representing by noisy quantum channel E_N:

$$E_0 = \sqrt{1-p}I$$

$$E_1 = \sqrt{\frac{p}{3}}X$$

$$E_2 = \sqrt{\frac{p}{3}}Y$$

$$E_3 = \sqrt{\frac{p}{3}}Z$$

where $p \in [0,1]$ is the depolarisation probability and is equally divided in applying all Pauli operations.

The depolarising channel for a single qubit can be described by the following equation:

$$\rho' = (1-p)\rho + \frac{p}{3}(X\rho X + Y\rho Y + Z\rho Z)$$

where:
- ρ is the initial density matrix of the qubit,
- ρ' is the density matrix after the depolarising error.

Bit-flip A flip known as Pauli-X gate, shares similar idea to classical NOT gate changes the classical bit from 0 to 1 and vice versa. To a qubit, it performs amplitudes swapping of $|0\rangle$ and $|1\rangle$ states. For example:

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

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

Phase flip A flip known as Pauli-Z gate, only perform the changes to $|1\rangle$ state. The Pauli-Z gate changes the phase of the qubit state while the basis state remains unchanged. For example:

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

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

Bit-phase flip A flip known as Pauli-Y gate, is a combination of Pauli-X and Pauli-Z gates, which applies bit-flip and phase-flip together to a qubit. For example:

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

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

A noise model only includes depolarising error channel is created and applied to a simple measurement circuit for demonstration. Figure 4 is a circuit includes a quantum register with three qubits and a classical register with three bits. The circuit performs a measurement operation on each qubit. The depolarising error channel is applied to all 3 qubits during measurement operations with a depolarisation probability of 1 (maximum noise). Since the qubit is initialised as $|0\rangle$, the probability for $|0\rangle$ changing the state is $\frac{2}{3}$ by bit-flip and bit-phase flip, and probability for $|0\rangle$ remaining the same state is $\frac{1}{3}$ by phase flip. Figure 5 is the simulation results of the 3-qubits measurement circuit with 8192 shots run. Table 2 explains the theoretical calculations match with the simulated results.

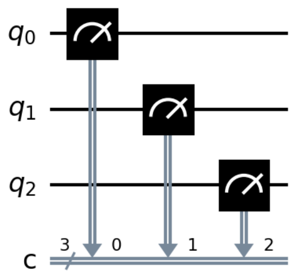

Figure 4: Simple 3-qubit measurement quantum circuit

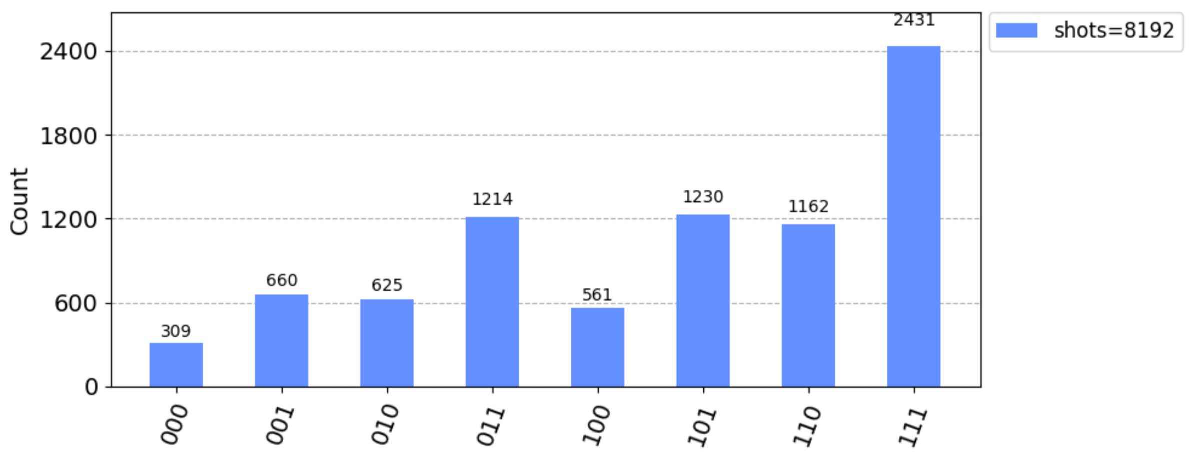

Figure 5: Plotted histogram of counts simulated for the 3-qubit measurement circuit with depolarising error rate $= 1$

4.2 Thermal Relaxation Error

Thermal relaxation channel is parameterised by relaxation time constants T_1, T_2, gate time t, and qubit temperature T, i.e. $thermal\_relaxation\_error(t1, t2, time, excited\ \_state\_population = 0)$.[20] This phenomenon represents the thermalisation of a qubit towards its equilibrium state, influenced by the [21]

States	Counts	Theoretical Calculations	Simulated Results
000	309	$\frac{1}{3} \cdot \frac{1}{3} \cdot \frac{1}{3} = 0.0370$	$\frac{309}{8192} = 0.0377$
001	660	$\frac{1}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} = 0.0741$	$\frac{660}{8192} = 0.0807$
010	625	$\frac{1}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} = 0.0741$	$\frac{625}{8192} = 0.0764$
011	1214	$\frac{1}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} = 0.1481$	$\frac{1214}{8192} = 0.1482$
100	561	$\frac{2}{3} \cdot \frac{1}{3} \cdot \frac{1}{3} = 0.0741$	$\frac{561}{8192} = 0.0686$
101	1230	$\frac{2}{3} \cdot \frac{1}{3} \cdot \frac{2}{3} = 0.1481$	$\frac{1230}{8192} = 0.1503$
110	1162	$\frac{2}{3} \cdot \frac{2}{3} \cdot \frac{1}{3} = 0.1481$	$\frac{1162}{8192} = 0.1421$
111	2431	$\frac{2}{3} \cdot \frac{2}{3} \cdot \frac{2}{3} = 0.2963$	$\frac{2431}{8192} = 0.2970$

Table 2: Summary of theoretical calculations and simulated results for the 3-qubit measurement circuit with depolarising error rate $= 1$

T_1 **Relaxation constant** A constant represents relaxation time until the qubit decays from the excited state to the ground state.

T_2 **Dephasing constant** A constant represents the coherence time, indicating how long the qubit maintains its phase relationship before dephasing occurs.

t_{Gate} **Gate time** A constant represents the duration over which a quantum gate operation is performed on a qubit. The longer the gate time, the higher the likelihood of experiencing relaxation errors, as there is more time for the qubit to transition between its excited and ground states.

Excited state population A constant represents the proportion of qubits that are in an excited state (as opposed to the ground state) at a given time. The environment's temperature is often assumed to be very close to absolute zero ($T = 0$). Therefore, the qubit temperature in this research is assumed to be zero, allowing the effects of temperature on excited state population to be disregarded, i.e. $excited\_state\_population = 0$, simplifying the analysis.

Relaxation error rates are represented by [21]

$$\mathcal{P}_{T_1} = e^{-\frac{t_{\text{gate}}}{T_1}} \quad , \quad \mathcal{P}_{T_2} = e^{-\frac{t_{\text{gate}}}{T_2}}$$

and the probability of a qubit being reset to $|0\rangle$ is given by

$$\mathcal{P}_{\text{reset}} = 1 - \mathcal{P}_{T_1}.$$

According to Qiskit documentation[20], for parameters to be valid T_1 and T_2 must satisfy $T_1 <= 2T_1$, which the parameters being used in this research listed in Table 1 already satisfied. Thermal relaxation error can exhibit two distinct behaviors based on the relationship between T1 and T2.

In the case where $\mathbf{T_2} \leq \mathbf{T_1}$, the thermal relaxation error is described as a probabilistic mixture of a qubit being reset to $|0\rangle$ and dephasing error. The probability of dephasing occurring is defined by

$$\mathcal{P}_Z = \frac{1 - \mathcal{P}_{\text{reset}}}{2} \left(1 - \frac{\mathcal{P}_{T_2}}{\mathcal{P}_{T_1}}\right).$$

The probability of a qubit remaining unchanged is defined by

$$\mathcal{P}_I = 1 - \mathcal{P}_Z - \mathcal{P}_{\text{reset}}.$$

In the case where $\mathbf{T_2} > \mathbf{T_1}$, the thermal relaxation error is defined by a Choi matrix[21]:

$$\begin{bmatrix} 1 & 0 & 0 & \mathcal{P}_{T_2} \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \mathcal{P}_{\text{reset}} & 0 \\ \mathcal{P}_{T_2} & 0 & 0 & 1 - \mathcal{P}_{\text{reset}} \end{bmatrix}$$

The above mathematical representations for the noise apply to one-qubit calculations. For two qubits, the noise models are constructed using the tensor product in Qiskit.

4.3 Readout Error

Readout error[22] means recorded classical bit value being flipped from the true outcome after a measurement. Error is characterised by a list of assignment probability vectors $\mathcal{P}(A|B)$, where A represents the recorded classical bit value and B is the true bit value resulting from the measurement. For a single qubit, this can be expressed as $\mathcal{P}(A|B) = [\mathcal{P}(A|0), \mathcal{P}(A|1)]$.

The $readout\_errors$ has one input parameter, denoted as $\mathcal{P}$, which represents the probabilities for noisy measurement outcomes. This parameter can be substituted into $\mathcal{P}(0|1)$ and $\mathcal{P}(1|0)$. Specifically, $\mathcal{P}(0|1)$ and $\mathcal{P}(1|0)$ are equal to $\mathcal{P}$, while $\mathcal{P}(1|1)$ and $\mathcal{P}(0|0)$ are equal to $1 - \mathcal{P}$.

In the noise model, the readout error is implemented by substituting four values: $\mathcal{P}(0|0)$, $\mathcal{P}(0|1)$, $\mathcal{P}(1|0)$, and $\mathcal{P}(1|1)$. For N-qubit systems, the readout error probabilities are represented as vectors:

$$\mathcal{P}[m] = [\mathcal{P}(0|m), \mathcal{P}(1|m), \ldots, \mathcal{P}(2^N - 1|m)]$$

5 Shor's algorithm

Before introducing Shor's algorithm and its implementation on a quantum circuit, it is recommended to understand the workings of the RSA cryptographic system, as outlined in Appendix 10.1. This section elucidates the significance of integer factorisation in the context of RSA. Subsequently, the following section presents the details of Shor's algorithm, which effectively addresses the factorisation problem.

5.1 Fundamentals of Shor's Algorithm

Any individual who has access to a quantum machine and the corresponding public key can potentially break a message that used RSA encryption. A vital component in this process is the implementation of Shor's algorithm, which plays a key role in compromising RSA. The details in this section draw from the original RSA paper [23]. To effectively launch an attack on RSA, it is necessary to ascertain the private key d. This key can be computed (as the modular inverse of e) if the prime factors p and q are known. Since $p \cdot q = N$, having access to c, e, and N allows us to factor N and recover the original message m.

To solve the prime factors of a number N where $p \neq q$, steps of how Shor's algorithm work are as below:

1. Random pick a value for number a where a must satisfy

   $$1 < a < N$$

   and is co-prime with N

   $$gcd(a, N) = 1.$$

   .

2. Find the order r of function $f(x) = a^r \; mod \; N$ where

   $$r > 0$$

   By incrementing r to find out the smallest condition of r where it satisfies

   $$a^r \equiv 1(mod\; N), \text{ where } r \neq 0 \tag{1}$$

3. Once r has been calculated, the next steps involve the following procedures:

   $$a^r \equiv 1(mod\; N)$$

   $$a^r - 1 \equiv 0(mod\; N)$$

   $$(a^{r/2} - 1)(a^{r/2} + 1) \equiv 0(mod\; N) \tag{2}$$

   The expression $(a^{r/2} - 1)(a^{r/2} + 1)$ represents an integer multiple of n, the number that is to be factored. Both $a^{r/2} \pm 1$ can serve as non-trivial factors of N, provided they are neither 1 nor N itself.

4. Calculate GCD function. $GCD(a^{r/2} + 1, N) \rightarrow p$ and $GCD(a^{r/2} - 1, N) \rightarrow q$.

Below is a **example** for determining the factor of 15, or $N = 15$.

1. Pick the value $a = 2$.
2. Compute order $2^r \equiv 1(mod\; 15)$

   $$2^0\%15 = 1$$

   $$2^1\%15 = 2$$

   $$2^2\%15 = 4$$

$$2^3\%15 = 8$$
$$2^4\%15 = 1$$
$$\rightarrow\ r = 4$$

3. Find
$$GCD(2^{4/2} + 1, 15)\ and\ GCD(2^{4/2} - 1, 15)$$
$$GCD(5, 15) = 5\ and\ GCD(3, 15) = 3$$

5.2 Implement Shor's Algorithm

To grasp the implementation Shor's algorithm, it is important to review the Quantum Fourier Transform (QFT) and Quantum Phase Estimation (QPE) in Section 9. Building on that basis, this part will offer a more comprehensive description of QPE's integration with Shor's Algorithm. The approach employs a quantum circuit that is intended to execute QPE on the controlled operator U [19] in order to solve the order-finding issue. We anticipate that the state will return to $|1\rangle$, as represented by the equation $U^r|1\rangle = |1\rangle$, after completing r iterations.

$$U^r|k\rangle = |a^r k\ mod\ N\rangle \tag{3}$$

With the same example using $N = 15$ and $a = 2$:

$$\begin{aligned} U^1|1\rangle &= |2\ mod\ 15\rangle = |2\rangle \\ U|2\rangle = U^2|1\rangle &= |4\ mod\ 15\rangle = |4\rangle \\ U|4\rangle = U^3|1\rangle &= |8\ mod\ 15\rangle = |8\rangle \\ U|8\rangle = U^4|1\rangle &= |16\ mod\ 15\rangle = |1\rangle \end{aligned}$$

It is possible to compute a superposition of states represented by $a^{rk} \mod N$ by repeatedly executing the U operation. The eigenstate outlined in equation 4 is the important one for U. It can be difficult to separate a particular phase from this superposition, though. A constraint $0 \leq s \leq r - 1$ is applied to an integer s to aid in the selection of an appropriate eigenstate. This makes it possible to derive distinct eigenstates for a range of s values. As a result, this method produces unique eigenstates for every unique value of s, as seen in equation 5.

$$|u_o\rangle = \frac{1}{\sqrt{r}} \sum_{k=1}^{r-1} |a^r k\ mod\ N\rangle \tag{4}$$

$$|u_s\rangle = \frac{1}{\sqrt{r}} \sum_{k=0}^{r-1} e^{-2\pi isk/r} |a^r k\ mod\ N\rangle \tag{5}$$

$$U|u_s\rangle = e^{2\pi is/r}|u_s\rangle \tag{6}$$

An example in cycle ($|u_0\rangle$) with assuming $r = 4$ is provided below:

$$|u_0\rangle = \frac{1}{\sqrt{4}}(|1\rangle + |2\rangle + |4\rangle + |8\rangle)$$

$$\begin{aligned} U|u_0\rangle &= \frac{1}{\sqrt{4}}(U|1\rangle + U|2\rangle + U|4\rangle + |8\rangle) \\ &= \frac{1}{\sqrt{4}}(|2\rangle + |4\rangle + |8\rangle + |1\rangle) \end{aligned}$$

A modulation exponentiation function utilising SWAP gates is developed by Qiskit[24] to act as the controlled-U gate described in Eq. 6 which functions as below:

$$U|1\rangle = |0001\rangle \xrightarrow{\text{SWAP (0,1)}} |0001\rangle \xrightarrow{\text{SWAP (1,2)}} |0001\rangle \xrightarrow{\text{SWAP (2,3)}} |0010\rangle = |2\rangle$$

$$U|2\rangle = |0010\rangle \xrightarrow{\text{SWAP (0,1)}} |0010\rangle \xrightarrow{\text{SWAP (1,2)}} |0100\rangle \xrightarrow{\text{SWAP (2,3)}} |0100\rangle = |4\rangle$$

$$U|4\rangle = |0100\rangle \xrightarrow{\text{SWAP (0,1)}} |1000\rangle \xrightarrow{\text{SWAP (1,2)}} |1000\rangle \xrightarrow{\text{SWAP (2,3)}} |1000\rangle = |8\rangle$$

$$U|8\rangle = |1000\rangle \xrightarrow{\text{SWAP (0,1)}} |0100\rangle \xrightarrow{\text{SWAP (1,2)}} |0010\rangle \xrightarrow{\text{SWAP (2,3)}} |0001\rangle = |1\rangle$$

The measurement of the acting register produces a probability distribution that is centered around about 2^n pairings of s/r after applying the $QFT\dagger$ to it. This is the point at which quantum computation ceases and the traditional post-processing computation step begins. The value of r may be extracted from this distribution by applying the continuous fractions technique of analysis. As a finite continuing fraction, the measured phase θ, which is a positive rational number smaller than one, may be written as follows:

$$\theta = \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots + 1\frac{1}{a_Q}}}} \tag{7}$$

for a given integer Q, where the positive integer coefficients are $a_1, a_2, \cdots, a_Q$. Then, using $s_0/r_0, s_1/r_1, \cdots, s_q/r_q$ to cycle over the cnovergents of $theta$, we make sure that

$$r\ is\ even \tag{8}$$

$$s\ and\ r\ are\ coprime \tag{9}$$

$$a^{r/2}\ mod\ N \neq 1. \tag{10}$$

The factors of N can be derived by the equation $gcd(a^{r/2} \pm 1, N)$, and the solution for the least value of r_q may yield the appropriate order r. On the other hand, the precision of the peaks in the output probability distribution might be impacted if there are not enough qubits. This could lead to an incorrect extraction of the intended order, which would impede the factorisation process.

In this research, quantum circuits for $N = 15, 63$, and 255 are used for the experiments. The value of a is set to 2 across all quantum circuits to ensure a controlled approach to data comparison. This standardised basis for analysis facilitates reliable comparisons across the experiments. Table 3 summarises the setup parameters used to simulate the quantum circuits for Shor's algorithm, along with the expected outputs. Although $N = 255$ has the factor pairs $(3, 85)$, $(5, 51)$, and $(15, 17)$, only the pair $(15, 17)$ is expected to be produced and counted as a success by Shor's algorithm.

N	**a**	**a^r mod 15**	**r**	**gcd(N, $a^{r/2}$ + 1)**	**gcd(N, $a^{r/2}$ - 1)**
15	2	1, 2, 4, 8, 1, ...	4	5	3
63	2	1, 2, 4, 8, 16, 32, 1, ...	6	9	7
255	2	1, 2, 4, 8, 16, 32, 64, 128, 1, ...	8	17	15

Table 3: Summary for Quantum Circuit implementation

5.3 Example of factoring $N = 15$ with $a = 2$

An real implemented experiment example of factoring $N = 15$ utilising the stages outlined in the previously stated parts of Shor's method is provided in this section. These procedures may be roughly divided into three categories: quantum order determination, classical post-processing, and classical pre-processing. Begin with the classical pre-processing procedures:

Step 1: Classical Pre-rocessing To ascertain whether N is odd. Given that 15 mod 2 $= 1$, N must be odd. Next, select a number a at random and ensure

$$1 < a < 15, e.g.\ a = 2$$

$$GCD(2, 15) = 1$$

$$no\ common\ factor\ except\ 1.$$

After that, second stage involves using the quantum order finding technique to discover the order r of function $a^r(mod\ N)$. Figure 6 depicts an example circuit. The further procedures for determining the quantum order may be found in Figure 6.

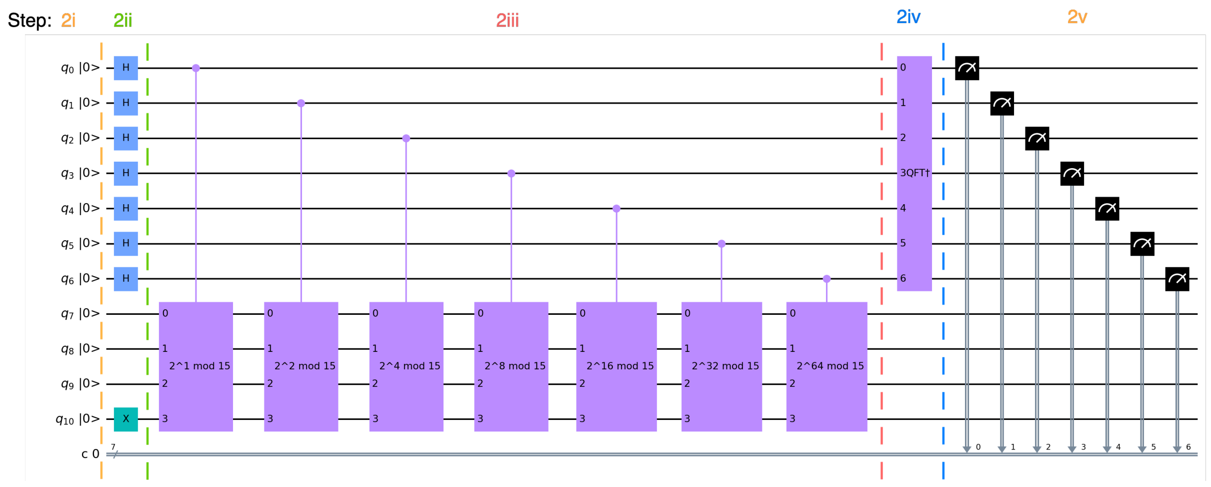

Figure 6: Quantum circuit of Shor's algorithm factoring $N = 15$ using $a = 2$

Step 2: Quantum Order Finding

i: To build QPE circuit for finding the order r of 2 mod 15, the number of bits n for $N = 15$ is determined by the binary representation of N. In this instance, 15 is represented as 1111, which requires 4 bits.

Construct the circuit with $2n + 3 = 11$ qubits [25] in total to achieve certain accuracy by establishing two quantum registers, counting register and acting register. The acting register consists of $n = 4$ qubits where to represent the binary value of N, while the counting register consists of $11 - 4 = 7$ qubits. Additionally, set aside a 7 classical bits in a separate classical register to hold the measurement findings. After that, carry out the first QPE phase, where all qubits are initialised to the ground state $|0\rangle$ so now the circuit can be represented by the mathematical expression as $|0\rangle^{\otimes 7}|0\rangle^{\otimes 4}$.

ii: Apply the X-gate to the first qubit in the acting register and the H-gate to the counting register. The following transformation is produced by this operation:

$$[H^{\otimes 7}|0\rangle^{\otimes 7}]\ |0\rangle^{\otimes 3}X|0\rangle = \frac{1}{\sqrt{2}^7}[|0\rangle + |1\rangle + |2\rangle + \cdots + |127\rangle]|1\rangle$$

iii: Apply the controlled unitary operation using the SWAP gate method $U^r|k\rangle = |a^r k(mod 15)\rangle$ to the qubits located in the acting register, while keeping the control qubits in the counting register. The outcome is as follows:

$$\frac{1}{\sqrt{128}}[|0\rangle|2^0(mod\ 15)\rangle + |1\rangle|2^1(mod\ 15)\rangle + |2\rangle|2^2(mod\ 15)\rangle + \cdots + |127\rangle|2^{127}(mod\ 15)\rangle]$$

Expand the simplify to obtain:

$$\frac{1}{\sqrt{128}}[|0\rangle|1\rangle + |1\rangle|2\rangle + |2\rangle|4\rangle + |3\rangle|8\rangle + |4\rangle|1\rangle + |5\rangle|2\rangle + |6\rangle|4\rangle + |7\rangle|8\rangle + \cdots$$

$$\cdots + |120\rangle|1\rangle + |121\rangle|2\rangle + |122\rangle|4\rangle + |123\rangle|8\rangle + |124\rangle|1\rangle + |125\rangle|2\rangle + |126\rangle|4\rangle + |127\rangle|8\rangle]$$

To further simplify:

$$\begin{gathered}\frac{1}{\sqrt{128}}[(|0\rangle + |4\rangle + |8\rangle + |12\rangle + \cdots + |112\rangle + |116\rangle + |120\rangle + |124\rangle) \otimes |1\rangle + \\ (|1\rangle + |5\rangle + |9\rangle + |13\rangle + \cdots + |113\rangle + |117\rangle + |121\rangle + |125\rangle) \otimes |2\rangle + \\ (|2\rangle + |6\rangle + |10\rangle + |14\rangle + \cdots + |114\rangle + |118\rangle + |122\rangle + |126\rangle) \otimes |4\rangle + \\ (|3\rangle + |7\rangle + |11\rangle + |15\rangle + \cdots + |115\rangle + |119\rangle + |123\rangle + |127\rangle) \otimes |8\rangle]\end{gathered}$$

For instance, by taking measurements on the acting register $|8\rangle$, may result as follows:

$$\left|\frac{1}{\sqrt{128}}\right|^2 (|3\rangle + |7\rangle + |11\rangle + |15\rangle + \cdots + |115\rangle + |119\rangle + |123\rangle + |127\rangle) \otimes |8\rangle$$

iv: Continue with the above example, apply $QFT^\dagger$ on the counting register, we obtain

$$\begin{aligned} QFT^\dagger|3\rangle &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-2\pi i 3\cdot k/128}|k\rangle \\ &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-3/64\cdot\pi i k}|k\rangle \\ QFT^\dagger|7\rangle &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-2\pi i 7\cdot k/128}|k\rangle \\ &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-7/64\cdot\pi i k}|k\rangle \\ QFT^\dagger|11\rangle &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-2\pi i 11\cdot k/128}|k\rangle \\ &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-11/64\cdot\pi i k}|k\rangle \\ QFT^\dagger|15\rangle &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-2\pi i 15\cdot k/128}|k\rangle \\ &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-15/64\cdot\pi i k}|k\rangle \\ &\vdots \\ &\vdots \\ QFT^\dagger|127\rangle &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-2\pi i 127\cdot k/128}|k\rangle \\ &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127} e^{-127/64\cdot\pi i k}|k\rangle \end{aligned}$$

$$
\begin{aligned}
QFT^{\dagger}|c\rangle &= \frac{1}{\sqrt{128}}\sum_{k=0}^{127}[e^{-3/64\cdot\pi ik} + e^{-7/64\cdot\pi ik} + e^{-11/64\cdot\pi ik} + e^{-15/64\cdot\pi ik} + \cdots + e^{-127/64\cdot\pi ik}]|k\rangle \\
&= \frac{1}{\sqrt{128}}[(e^0 + e^0 + e^0 + e^0 + \cdots + e^0)|0\rangle \\
&+ (e^{-3/2\cdot\pi i} + e^{-7/2\cdot\pi i} + e^{-11/2\cdot\pi i} + e^{-15/2\cdot\pi i} + \cdots + (e^{-127/2\cdot\pi i})|32\rangle \\
&+ (e^{-3\pi i} + e^{-7\pi i} + e^{-11\pi i} + e^{-15\pi i} + \cdots + e^{-127\pi i})|64\rangle \\
&+ (e^{-9/2\cdot\pi i} + e^{-21/2\cdot\pi i} + e^{-33/2\cdot\pi i} + e^{-45/2\cdot\pi i} + \cdots + +e^{-381/2\cdot\pi i})|96\rangle] \\
&= \frac{1}{\sqrt{128}}[(1+1+1+1+\cdots+1)|0\rangle + (i+i+i+i+\cdots+i)|32\rangle \\
&+ (-1-1-1-1-\cdots-1)|64\rangle + (-i-i-i-i-\cdots-i)|96\rangle] \\
&= \frac{1}{\sqrt{128}}[32|0\rangle + 32i|32\rangle - 32|64\rangle - 32i|96\rangle]
\end{aligned}
$$

v: At measurement stage, the possible outcomes for the qubits are $|0\rangle$, $|32\rangle$, $|64\rangle$, and $|96\rangle$, each with a probability of:

$$P(k) = \frac{1}{4} \quad \text{for } k \in \{0, 32, 64, 96\}$$

This means each state has an equal probability of being measured.

The experimental noise-free results with shots=1024 is shown in Figure 7. To determine the phase of eigenvalues of operators from the plotted histogram by $\Phi = \frac{\ell}{2^m}$, counting register measurement shows peak ℓ occurs at $|0000000\rangle$, $|0100000\rangle$, $|1000000\rangle$ and $|1100000\rangle$, i.e. $|0\rangle$, $|32\rangle$, $|64\rangle$ and $|96\rangle$ with similar probability of $\frac{1}{4}$.

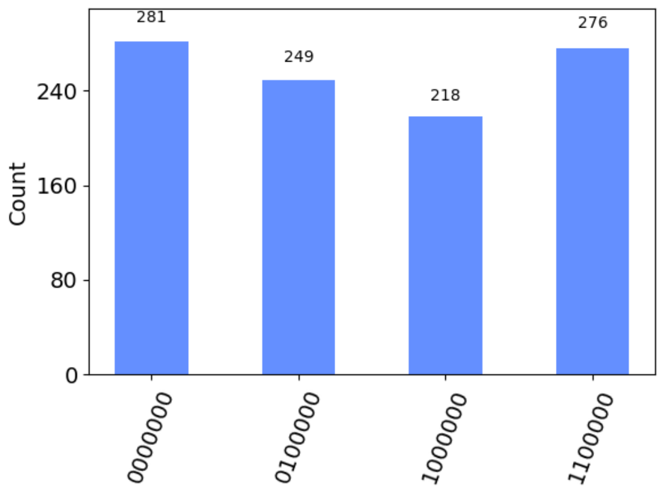

Figure 7: Factor $N = 15$ using $a = 2$ noise free measurement outcomes with $shots = 1024$.

At this point, the order r and the factors will be found using the classical post-processing techniques.

Step 3: Classical Post-processing

i: $|0\rangle$: This result is always be neglected as provides meaningless insights.

ii: $|32\rangle$: Phase $\Phi = \frac{32}{2^7}$. By continued fraction, $\frac{s}{r} = \frac{1}{4}$ where $r = 4$. Now can proceed to the GCD function stage, as r satisfies the requirements 8â10.

$$GCD(3, 15) = 3 \quad and \quad GCD(5, 15) = 5$$

$N = 3 \times 5$ is factored successfully with shots=249.

iii: $|64\rangle$: Phase $\Phi = \frac{64}{2^7}$. By continued fraction, $\frac{s}{r} = \frac{1}{2}$ where $r = 2$. This number will not be proceeded with since r does not match the criteria 10. Factoring failed with shots=218.

iv: $|96\rangle$: Phase $\Phi = \frac{96}{2^7}$. By continued fraction, $\frac{s}{r} = \frac{3}{4}$ where $r = 4$. We have another $N = 3 \times 5$ is factored with shots=276.

Therefore, the probability to get $N = 3 \times 5$ being factored successfully is $\frac{249+276}{249+218+276} = 70.66\%$.

The above example outlines the complete series of quantum circuit and post-quantum processing steps involved in defining and calculating the factoring success rate for Shor's algorithm. Table 4 presents the output generated automatically by the script, which executes all the aforementioned procedures and yields the final percentage of correct results. The complete set of simulated results is available on GitHub.

	Phase	Fraction	Guess for r	Count
0	0.00	0/1	1	281
1	0.25	1/4	4	249
2	0.50	1/2	2	218
3	0.75	3/4	4	276

	Guess for r	Guess 1	Guess 2	Sum by Guess	Repetition Count
0	4	3	5	525	2
1	2	1	3	218	1

Correct result: 525 out of 743 (70.66% correct)

Table 4: Output generated by the post-quantum processing script for $N = 15$ and $a = 2$ in a noise-free simulation, referred to as `Data/fac15_noise-free_data.txt`, available on GitHub.

6 Experiment Results

The parameters mentioned in Section 3.2 are sweeped from minimum and maximum values in a total of 6 incremental steps. This means that each experiment results in 6 data points for the parameters varied from minimum to maximum values.

An example of the plotted count histogram for the depolarising error applied to the factor 15 quantum circuit, varying from the minimum to maximum value of the depolarising error rate, is shown in Figure 8. This illustrates how increasing noise leads to more possible measurement outcomes displayed on the x-axis, resulting in a more spread-out distribution. The rest of the simulated histograms for other noise types applied to different quantum circuits can be found on the Github.

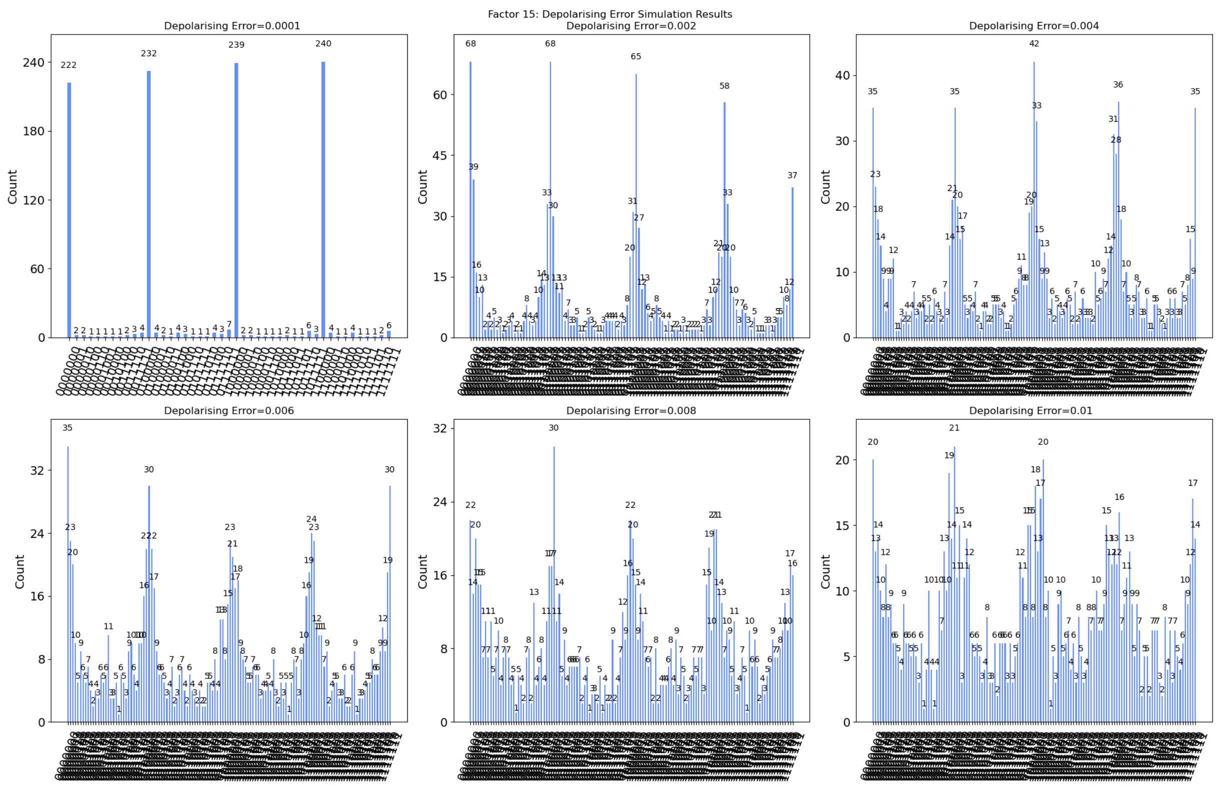

Figure 8: Plotted count histograms simulated for factoring $N = 15$ with sweeping depolarising error rate

6.1 Factoring Success Rates within Varying Error Range

Depolarising error

- Error rate = 1×10^{-4} to 1×10^{-2}
  - Figure 9 demonstrates a steady downward trend in the success rate as depolarising noise increases for $N = 15$.
  - For $N = 63$ and $N = 255$, a steep decline in success rate is observed between 1×10^{-4} and 2×10^{-3}.
  - Based on these observations, an additional experiment was conducted, starting with a smaller depolarising error rate and a narrower range, specifically from 1×10^{-5} to 1×10^{-3}.

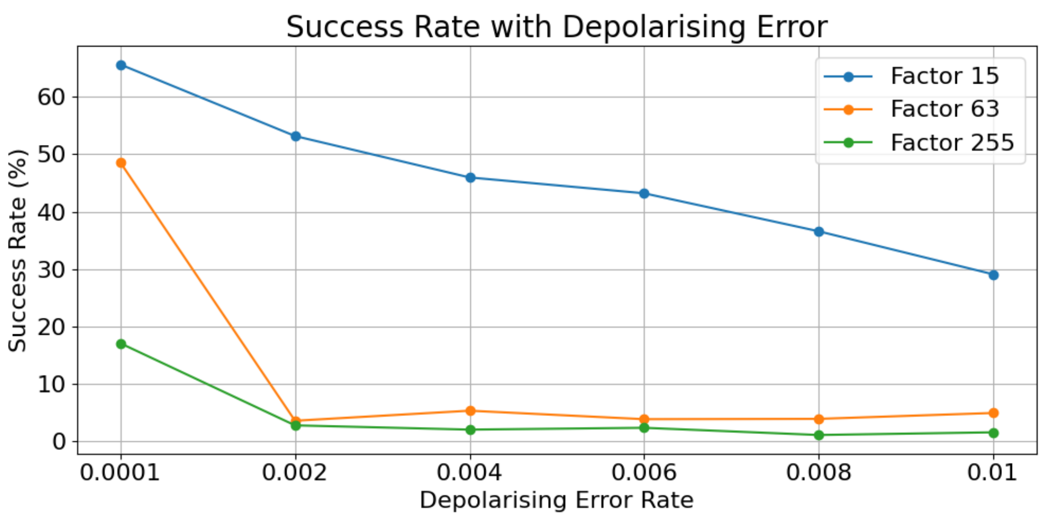

Figure 9: Experimental results for depolarising error

- Error rate = 1×10^{-5} to 1×10^{-3}
  - In Figure 10, $N = 15$'s success rates consistently remain high, ranging between 60% and 68%, showing non-sensitivity to changes in noise.
  - Conversely, for $N = 63$, there is a dramatic decline in success rates from 60.45% to 13.62% as noise increases.
  - For $N = 255$, the success rate starts at 52.72% but drops significantly to 1.82% at a noise level of 0.001.
  - The sensitive range for $N = 63$ and $N = 255$ is identified between 1×10^{-5} and 2×10^{-4}.

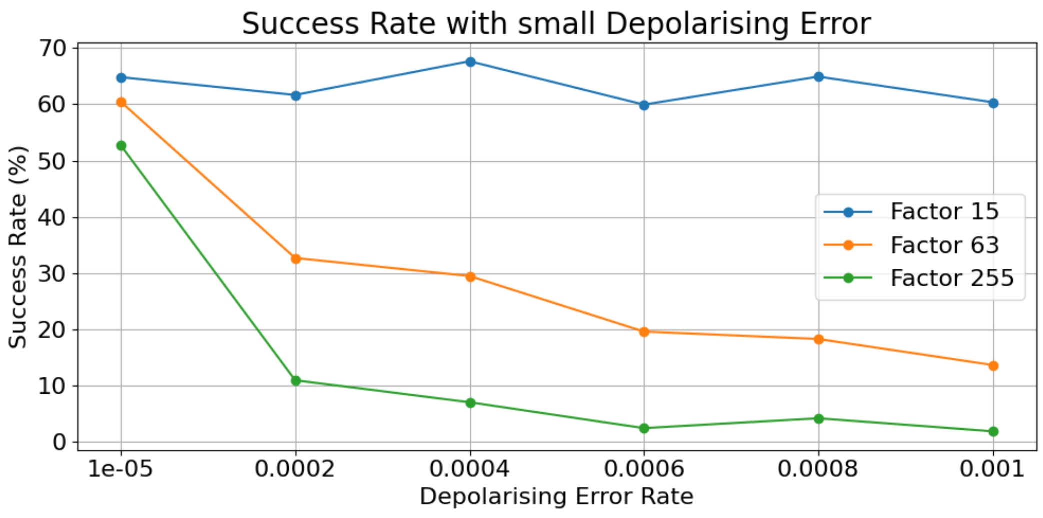

Figure 10: Experimental results for adjusted depolarising error

In summary, as the circuit size and the number of quantum gates increase, the success rate undergoes a steeper decline, becoming increasingly sensitive to even minimal depolarising noise introduced to the gates.

Thermal relaxation error

- Refer to Figure 11, for $N = 15$, the success rates consistently remain above 63% across various error rates, indicating resilience to thermal relaxation errors.
- In contrast, the success rates for $N = 63$ and $N = 255$ exhibit a more pronounced decline:
  - $N = 63$: Rates drop from 50.00% to 26.43%.
  - $N = 255$: Rates decrease from 31.22% to 11.71%.

This decline suggests that larger circuit sizes and increased numbers of quantum gates lead to heightened sensitivity to errors. A notable pattern emerges where success rates for $N = 15$ and $N = 63$ experience a sudden drop until the last simulation data point, corresponding with very small values of T_1 and T_2.

It is anticipated that $N = 255$ may exhibit a similar pattern, although this is not illustrated in the current plot. Given these observations, it can be inferred that if the values of T_1 and T_2 were to become even smaller, $N = 255$ would likely experience a sudden decline in success rates, surpassing the limits of the current simulation range.

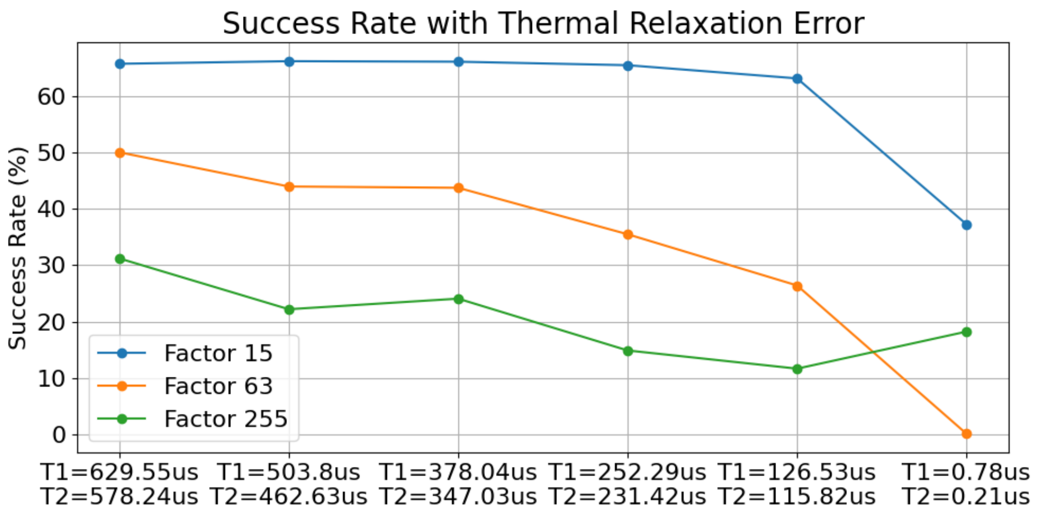

Figure 11: Experimental results for thermal relaxation error

Readout error

- For $N = 15$, the success rates maintain a relatively high level initially but decline steadily as the readout error increases can be observed in Figure 12.
- For $N = 63$, there is a sharper drop in success rates, particularly noticeable at lower error levels.
- $N = 255$ exhibits the most dramatic decline, with success rates falling to critically low levels when the readout error reaches a moderate threshold.

6.2 Factoring Success Rates at Median Error Levels

An experiment was conducted to evaluate the success rates for $N = 15, 63$, and 255 quantum circuits utilised the same AerSimulator as the earlier experiments but employed the real backend $ibm\_brisbane$ noise model. By running 1024 shots for each configuration, the corresponding success rate results are listed in Table 5.

Table 6 presents the factoring success rates for the three custom noise models, utilising median error rate values for each model. The analysis reveals several key insights:

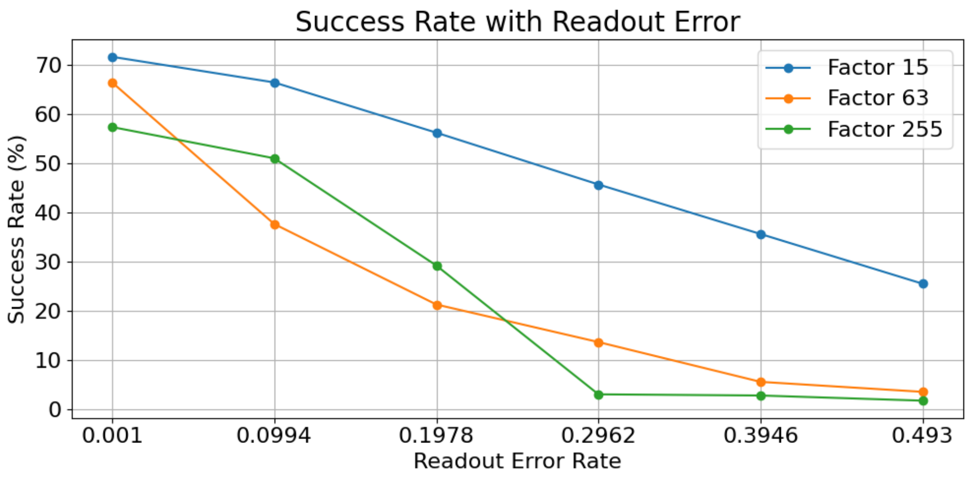

Figure 12: Experimental results for readout error

N	Success Rate
15 (1111)	70.65%
63 (111111)	4.96%
255 (11111111)	2.55%

Table 5: Factor success rates for real QPUs backend ($ibm\_brisbane$)

- Depolarising error significantly impacts the success rates across all factors. Notably, the median depolarising error for $N = 15$ results in a considerably lower success rate compared to the $ibm\_brisbane$ noise model. This suggests that the chosen depolarising error rate of 0.0033 may be excessively high for implementing the median depolarising error noise model. For $N = 63$ and $N = 255$, the corresponding depolarising error rates of 3.47% and 2.40% closely align with those observed in the $ibm\_brisbane$ model. This similarity underscores the dominant influence of depolarising errors, while other error models demonstrate relatively minor effects.
- Thermal relaxation errors for $N = 15$ are minimal within the three circuits and thus negligible. However, for $N = 63$ and $N = 255$, the success rates decrease to 26.99% and 10.49%, indicating that thermal relaxation errors also have a substantial impact on Shor's algorithm performance.
- The readout error for $N = 15$ is also insignificant within the three circuits. For $N = 63$, the readout success rate closely mirrors the noise-free results, suggesting that readout errors exert the least influence among the examined noise types. For $N = 255$, the decrease in success rate relative to noise-free conditions is approximately 5%.

In conclusion, the findings highlight that depolarising errors exert the most significant influence on Shor's algorithm, followed by thermal relaxation errors, with readout errors having the least impact.

Error Type	Factor 15	Factor 63	Factor 255
Noise Free	66.67%	56.44%	59.71%
Depolarising	47.57%	3.47%	2.40%
Thermal relaxation	64.14%	26.99%	10.49%
Readout	65.05%	56.40%	52.92%

Table 6: Factor success rates for the custom noise models applying median error rates

7 Discussion

From Figures 9 to 11, fluctuations such as success rate rebounds and increases along with noise increases or the success rate for $N = 255$ surpasses that of $N = 63$ for certain data points can be seen.

This observation may be attributed to the nature of quantum gates on an actual quantum computer; for instance, Hadamard gates create superpositions of quantum states, ideally when a Hadamard gate is applied to the state $|0\rangle$, it produces a superposition that gives a 50% probability of measuring either $|0\rangle$ or $|1\rangle$. However, in practice, the behavior of real quantum computing systems reflects stochastic characteristics, whereby executing the exact same quantum circuit with the same parameters can yield different measurement outcomes in each execution. The distribution of results will tend to equalise when measured over a larger number of shot trials, such as 8192 times. This is why, in real quantum computers or simulators, researchers adjust shots to a large number, which represents how many times the circuit is measured to obtain a reliable estimate of the quantum state. Such variability can be linked to the nature of quantum gates themselves.

Small imperfections in gate operations are not corrected in the experiments conducted. Consequently, the errors can accumulate over time, leading to disturbances in the state of qubits. This accumulation can compromise the accuracy of calculations, and if significant, may result in incorrect outcomes, ultimately diminishing any potential quantum advantage. It is generally not possible to achieve perfect computational results at this time. Nevertheless, there is a threshold theorem for quantum error correction that says adding additional qubits can lower the overall error rate as long as the implemented circuit's error rate stays below a given critical value [26]. Consequently, future studies can concentrate on reducing the effects of these flaws in order to stop coherence error buildup and the ensuing reduction in quantum computation accuracy.

8 Future Work

An outline of future work has been designed regarding how error correction can be implemented for depolarising errors. Since depolarising error matrices is known and the median depolarising error rate of the quantum machine can be assumed. The proposed approach involves deploying the inverse matrix of the identified depolarising error, applying it in a similar way of how the depolarising error is introduced to each quantum gate and qubit, while considering the corresponding probability of depolarising error. It is anticipated that this inverse matrix could offset the effects of the depolarising error, potentially restoring accuracy to quantum computations. Experiments can be conducted using software simulations of the IBM real quantum computer noise models to validate this methodology.

9 Conclusion

This research paper demonstrates the performance of Shor's algorithm on the IBM quantum simulator, utilising custom noise models, including noise-free, depolarising error, thermal relaxation error, and readout error, alongside the IBM real quantum computer noise model, known as $ibm\_brisbane$. While achieving perfect computational results is not feasible at this stage, future work can be focused on exploring error correction strategies to offset the noise impacts by validating through software simulations of the IBM real quantum computer noise models. In summary, it can be said that quantum computers are unlikely to be as efficient at factoring integers as classical computers due to the effects of imperfections before they can reach the required number of qubits for currently available error-correction methods and thus achieve a high probability of obtaining correct results. The capability to break RSA-2048, which is widely used today, remains a distant goal for Shor's algorithm.

10 Appendix

10.1 RSA cryptography system

Public-key cryptosystems like the RSA algorithm, which was first presented by Rivest, Shamir, and Adleman in 1977, are extensively used in a wide range of digital applications[23]. Your machine creates a public key and a private key each time you visit a "secure site" on the Internet, such as when you read email or make transactions. Subsequently, these keys are employed to guarantee the privacy of delicate data, including credit card numbers and personal information, while it is being transmitted, preventing unauthorised access.

The modulus N and the public exponent e make up the public key in the RSA cryptosystem. It is made up of the private exponent d and the modulus N for the private key.

An RSA public-key/private-key pair can be created by completing the following steps:

1. Generate a collection of big, arbitrary prime numbers, p and q.
2. Calculate N so that
$$N = p \cdot q \tag{11}$$
3. Compute the Euler's totient function
$$\Phi(N) = (p-1) \cdot (q-1) \tag{12}$$
4. Choose a number for d that is in relation to $p-1$ and $q-1$ prime.
$$GCD(d, \Phi(N)) = 1. \tag{13}$$
It is possible to choose any primer number that is smaller than $\Phi(N)$.
5. The multiplicative inverse of d is e, therefore compute the private exponent d from e, p, and q, satisfying
$$e \cdot d = 1(mod\Phi(N)) \tag{14}$$
6. Generate the public key (e, N) and the private key (d, N).

Assume there are two individuals. Bob along with Alice. With (e, N) as her public key and (d, N) as her private key, Alice creates a set of two keys. Bob wishes to $m$send Alice a message. He obtains Alice's public key (e, N) and transmits using the following protocol:

1. First, Bob prepares message m, turning text into a large integer. He uses the public key (e, N) to encrypt his message m. The RSA cryptosystem uses exponentiation to the power of e^{th} modulo N for encryption, producing an encoded message known as the ciphertext that is incomprehensible to humans$c$:
$$c = ENCRYPT(m) = m^e mod\, N. \tag{15}$$
2. He then uses a secure communication route to send Alice the encrypted message c.
3. To decrypt, Alice calculates her exponentiation to the modulo of N at d^{th}:
$$m = DECRYPT(c) = c^d mod\, N. \tag{16}$$

The inverse relationship between the exponents e and d ensures that encryption and decryption processes are compatible, allowing the decryption process to recover the original message m. It is difficult to get m from c if you do not have the prime factors p and q or the private key (d, N). Therefore, the values of N and e can be made publicly known without endangering the system's security, fulfilling a basic criteria of a public-key cryptosystem. Determining d is the only challenging step in cracking the private key, given that N is provided. It is necessary to compute p and q in order to find d. Shor's algorithm can solve because of this.

Below is an **example** of how to encrypt with RSA and breaking it using $N = 15$.

1. Let $p = 3$; $q = 5$; $N = 15$; $e = 7$, in which every value satisfies the conditions given in equations 11â14. Let m to be 4.
2. To get the ciphertext to send out, encrypt the message.
$$c = 4^7 \mod (15)$$
$$ciphertext = 4$$
3. An attacker is now trying to crack the RSA encryption. The attacker knows the public key e, N, and the ciphertext c, but the private key remains unknown. The ciphertext is decrypted by performing the following calculation:
$$m = 4^3 \mod (15).$$
The only thing left to do is determine p and q from N, as this information is what allows one to compute the equation 12. By reversing the equation 13, d can be found once the $\Phi(N)$ is known. This is the reason that in order to break the RSA, N must be factorised.

10.2 Quantum Fourier Transformation (QFT)

Working using the amplitudes of a quantum wavefunction, the Quantum Fourier transform (QFT) is a quantum variation of the Discrete Fourier transform (DFT). It is an important component of many quantum algorithms; two prominent examples are Quantum Phase Estimation (QPE) and Shor's factoring method. In this section, [19] and [27] are the main references. A vector of complex numbers is needed for the DFT $x = (x_0, x_1, ..., x_{L-1})$. The DFT $\tilde{x} = (\tilde{x_0}, \tilde{x_1}, ..., \tilde{x_{L-1}})$ can be described as

$$\tilde{x}_k = \frac{1}{\sqrt{L}} \sum_{j=0}^{L-1} e^{2\pi ijk/L} x_j \ , \tag{17}$$

and the inverse of DFT can be described as

$$x_j = \frac{1}{\sqrt{L}} \sum_{j=0}^{L-1} e^{-2\pi ijk/L} \tilde{x}_k \ , \tag{18}$$

where the vector's length is indicated by the indices $j, k \in \{0, 1, ..., L-1\}$ and L. With n qubits and $L = 2^n$ potential quantum states, the state of quantum state vector $x = (x_0, x_1, ..., x_{L-1})$ is represented by

$$|\Psi\rangle = \sum_{j=0}^{L-1} x_j |j\rangle. \tag{19}$$

A function that transforms between the computational basis and the Fourier basis is what the QFT is commonly understood to be.

$$|State\ in\ Computational\ Basis\rangle \xrightarrow{QFT} |State\ in\ Fourier\ Basis\rangle$$

$$QFT|\Psi\rangle \xrightarrow{QFT} |\tilde{\Psi}\rangle$$

$$|\tilde{\Psi}\rangle = QFT(\sum_{j=0}^{L-1} x_j |j\rangle) = \sum_{j=0}^{L-1} \tilde{x_k} |k\rangle = \frac{1}{\sqrt{L}} \sum_{k=0}^{L-1} e^{2\pi ijk/L} |k\rangle. \tag{20}$$

The transformation that fulfills $QFT \cdot QFT^{\dagger} = I$ is known as the Inverse Quantum Fourier Transform ($QFT^{\dagger}$). It has the exact same definition with a negative sign on the exponents.

$$QFT^{\dagger}|k\rangle = \frac{1}{\sqrt{L}} \sum_{l=0}^{L-1} e^{-2\pi ijk/L} |l\rangle. \tag{21}$$

The standard physical implementation of the QFT circuit is shown in Figure 13. The operations performed on the first qubit, second qubit, and so forth are depicted in the circuit diagram's upper line. The diagram's boxes each correspond to a unitary transformation that has been applied to a particular qubit. The Hadamard gate (shown as H) and the phase-shift gate (represented as R_n) are two examples of the transformations. It's crucial to remember that in order to return the qubits to their initial arrangement at the end of the circuit, $SWAP$ gates are required. This circuit's validity has been demonstrated[28].

QFT has the attribute of unitarity since it is made up only of unitary transformations. The circuit should be undone by applying all operations in reverse order in order to implement $QFT^{\dagger}$. One can acquire the inverse $QFT^{\dagger}$ by traveling left to right in reverse order while crossing the QFT circuit. All phase angles in the circuit are inverted throughout this operation, which starts with the SWAP gates being reversed ($R_n^{\dagger}$ replaces R_n).

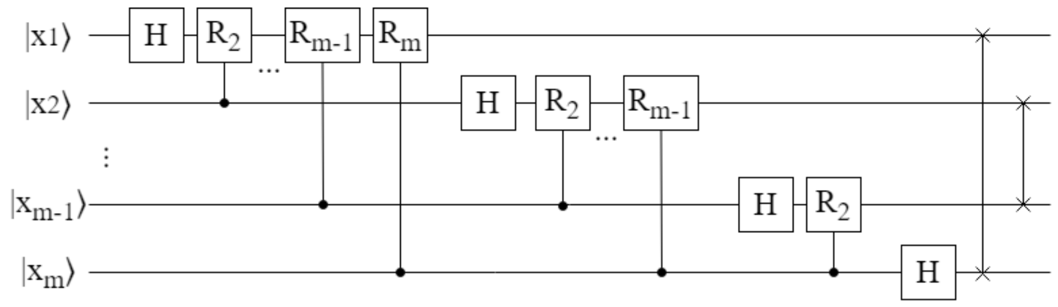

Figure 13: Physical implementation of the standard QFT circuit

10.3 Quantum Phase Estimation (QPE)

The $QFT^{\dagger}$ operation is the second step of the QPE algorithm, sometimes known as the back-end QPE circuit. In this section, [19] and [29] are the main references. Prior to examining the QPE algorithm, one must familiarize themselves with the controlled-unitary (CU) operator's operation. Let U be a unitary operator such that $U|\Psi\rangle = e^{2\pi i\theta}|\Psi\rangle$ and that its eigenvalue is $e^{2\pi i\theta}$. the impact of the CU operator on a two-qubit system with $|\Psi\rangle$ as the second qubit value. The following is how the CU operates:

$$CU(|0\rangle|\Psi\rangle) \rightarrow |0\rangle|\Psi\rangle \ \ and \ \ CU(|1\rangle|\Psi\rangle) \rightarrow e^{2\pi i\theta}|1\rangle|\Psi\rangle \tag{22}$$

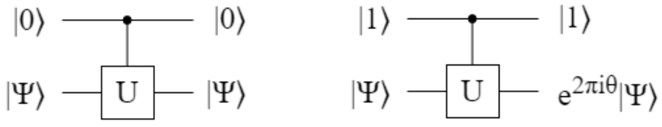

Figure 14: Operation of CU. The target qubit is affected by the gate that is applied in the circuit, while the control qubit is represented by the upper line.

Thus, the first qubit has a superposition of states $|0\rangle$ and $|1\rangle$.

$$CU(\frac{|0\rangle + |1\rangle}{\sqrt{2}}|\Psi\rangle) \rightarrow \frac{|0\rangle + e^{2\pi i\theta}|1\rangle}{\sqrt{2}}|\Psi\rangle \tag{23}$$

A front-end and a back-end component make up the circuit used to implement the QPE algorithm. The two registers that make up the QPE front-end circuit are the acting register (lower register) and the counting register (upper register). The number of bits, m, is determined by the accuracy of the number of digits and the likelihood of success when predicting θ. The following are the steps for QPE.

1. Utilise Hadamard with the register for counting. The quantum state will be

$$\frac{1}{2^{m/2}}(|0\rangle + |1\rangle) \otimes \cdots \otimes (|0\rangle + |1\rangle)|\Psi\rangle. \tag{24}$$

2. Apply CU^{2^j} gate where qubit $m-j$ is th control for $j = 0, \cdot, m-1$.

   - $j = 0$, apply CU^{2^0} where qubit m is the control qubit.

$$\frac{1}{2^{m/2}}(|0\rangle + |1\rangle) \otimes \cdots \otimes (|0\rangle + |1\rangle) \otimes (|0\rangle + e^{2\pi i\theta 2^0}|1\rangle)|\Psi\rangle. \tag{25}$$

   - $j = 1$, apply CU^{2^1} where qubit $m-1$ is the control qubit.

$$\frac{1}{2^{m/2}}(|0\rangle + |1\rangle) \otimes \cdots \otimes (|0\rangle + e^{2\pi i\theta 2^1}|1\rangle) \otimes (|0\rangle + e^{2\pi i\theta 2^0}|1\rangle)|\Psi\rangle. \tag{26}$$

$\vdots$

$\vdots$

- $j = m-1$, apply $CU^{2^{m-1}}$ where qubit 1 is the control qubit.

$$\frac{1}{2^{m/2}}(|0\rangle + e^{2\pi i\theta 2^{m-1}}|1\rangle) \otimes \cdots \otimes (|0\rangle + e^{2\pi i\theta 2^{1}}|1\rangle) \otimes (|0\rangle + e^{2\pi i\theta 2^{0}}|1\rangle)|\Psi\rangle. \tag{27}$$

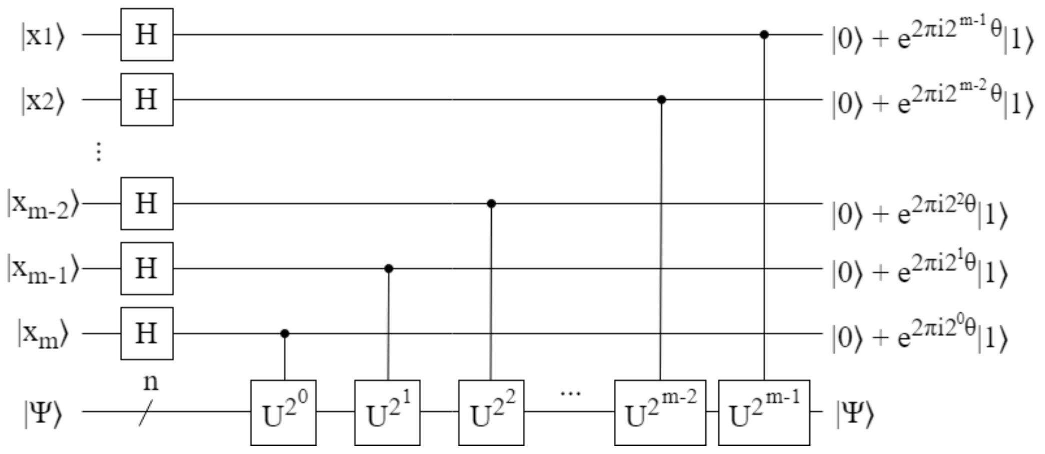

Figure 15: QPE front-end circuit

- For every m qubit in the counting register, the θ has been encoded into the phase of state $|1\rangle$. The front-end's output state formula can be written as

$$\frac{1}{2^{m/2}}\bigotimes_{l=1}^{m}(|0\rangle + e^{2\pi i\theta 2^{m-1}}|1\rangle)|\Psi\rangle$$

$$= \frac{1}{2^{m/2}}\bigotimes_{l=1}^{m}\sum_{k_l=0}^{l} e^{2\pi i k_l\theta 2^{m-1}}|k_l\rangle)|\Psi\rangle$$

$$= \frac{1}{2^{m/2}}\sum_{k_l=0}^{l}\cdots\sum_{k_m=0}^{l} e^{2\pi i\sum_{l=1}^{t}k_l 2^{m-l}\theta}|k_1\cdots k_m\rangle|\Psi\rangle$$

$$= \frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1} e^{2\pi i k\theta}|k\rangle|\Psi\rangle. \tag{28}$$

If θ for some x and $L = 2^m$ has the form $\frac{x}{L}$, or equivalently, if θ can be written with m bits as

$$\theta = 0.\theta_1\theta_2\cdots\theta_{m-1}\theta_m$$
$$= \frac{\theta_1\cdots\theta_m}{N}, \tag{29}$$

the output state of the front-end can be re-written as

$$\frac{1}{2^{m/2}}\sum_{k=0}^{2^m-1} e^{2\pi i kx/L}|k\rangle$$

$$= QFT|x\rangle|\Psi\rangle. \tag{30}$$

3. Remember $|j\rangle$ as the QFT of a basis state.

$$\frac{1}{\sqrt{L}}\sum_{j=0}^{L-1} e^{2\pi ijk/L}|k\rangle. \tag{31}$$

Therefore, the back-end of the QPE circuit consists of an inverse QFT operating on the acting register. The final output state is given by

$$QFT\dagger\,(QFT|x\rangle|\Psi\rangle) = |x\rangle|\Psi\rangle. \tag{32}$$

Then measure $|x\rangle|\Psi\rangle = |\theta_1\cdots\theta_m\rangle|\Psi\rangle$.

In figure 16, the front-end part of QPE circuit is expanded with the connecting of the $QFT^\dagger$ part to show the complete QPE circuit, summarising the previously mentioned processes. The front-end and back-end circuits are divided by the dashed line.

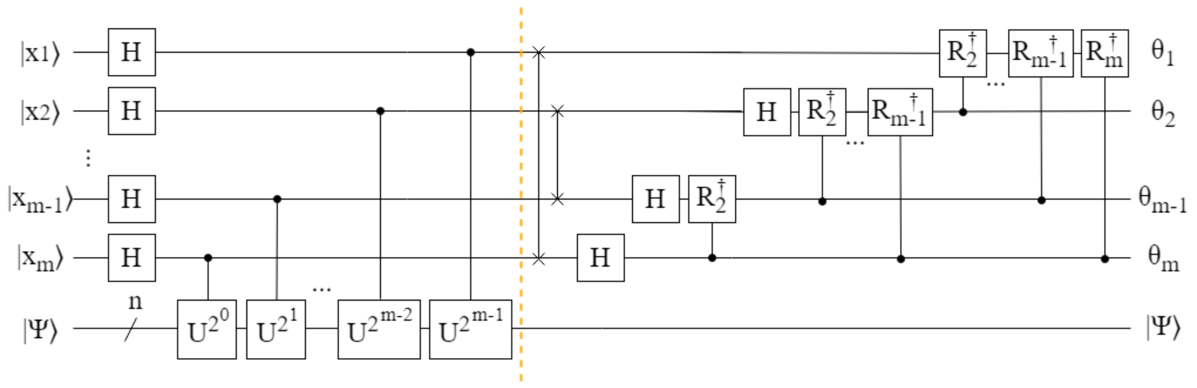

Figure 16: The entire circuit for QPE.

10.4 IBM Quantum Backend Error Rates

	Min	Median	Max
ibm_cusco	0.0067	0.0383	0.4354
ibm_nazca	0.0050	0.0283	0.4900
ibm_osaka	0.0035	0.0248	0.4931
ibm_brussels	0.0033	0.0189	0.2040
ibm_torino	0.0047	0.0187	0.4333
ibm_kyoto	0.0031	0.0175	0.3153
ibm_fez	0.0038	0.0165	0.2108
ibm_strasbourg	0.0037	0.0151	0.4809
ibm_brisbane	0.0049	0.0145	0.3436
ibm_sherbrooke	0.0029	0.0143	0.4256
ibm_cleveland	0.0046	0.0131	0.3237
ibm_kawasaki	0.0045	0.0121	0.3717
ibm_quebec	0.0024	0.0117	0.3830
ibm_rensselaer	0.0027	0.0094	0.3869
ibm_kyiv	0.0010	0.0078	0.2155

Table 7: IBM real quantum computer backend readout error rate

	T1			T2		
	Min	Median	Max	Min	Median	Max
ibm_brussels	5.2100	307.1700	629.5500	0.4600	174.0100	494.8000
ibm_osaka	4.3300	302.1600	606.0200	7.4200	127.2300	483.1000
ibm_quebec	3.1800	295.0700	502.6600	2.2100	200.6300	506.9300
ibm_strasbourg	27.0900	292.3200	617.5200	15.7200	194.1000	552.6100
ibm_kyiv	22.2200	290.8200	595.1000	14.1400	119.0000	578.2400
ibm_sherbrooke	22.3100	257.4800	479.1400	10.2600	184.0300	505.1900
ibm_rensselaer	37.7600	249.2800	551.3600	0.9500	166.3300	443.4600
ibm_brisbane	9.2800	234.1300	417.2600	0.2100	157.6100	535.9500
ibm_kyoto	0.8700	206.9800	400.6500	6.2800	87.9800	313.6400
ibm_cleveland	7.2900	196.2200	350.4900	16.1600	179.6400	512.2000
ibm_nazca	20.0900	192.7100	374.7600	13.0300	135.2600	401.4500
ibm_kawasaki	0.7800	188.7000	373.1200	6.4000	138.9300	439.7300
ibm_torino	6.6500	160.6600	319.1100	11.1700	125.5000	350.7100
ibm_fez	26.1800	131.5100	231.7700	5.3300	95.7500	246.1900
ibm_cusco	21.9400	117.7300	312.4000	0.2800	73.4600	353.9500

Table 8: IBM real quantum computer backend T_1 and T_2 data

10.5 Additional Resources

Visit the GitHub repository at `https://github.com/ansontyyeung/MSc-dissertation-ShorsAlgo` to access the codes, simulation results and plots related to this dissertation.

References

[1] E. Rescorla, *SSL and TLS: Designing and Building Secure Systems.* Addison-Wesley, 2001. [Online]. Available: https://books.google.de/books?id=765zngEACAAJ

[2] P. Shor, “Algorithms for quantum computation: discrete logarithms and factoring,” in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, 1994, pp. 124–134.

[3] L. P. Yulianti and K. Surendro, “Implementation of quantum annealing: A systematic review,” *IEEE Access*, vol. 10, pp. 73 156–73 177, 2022.

[4] P. Hauke, H. G. Katzgraber, W. Lechner, H. Nishimori, and W. D. Oliver, “Perspectives of quantum annealing: methods and implementations,” *Reports on Progress in Physics*, vol. 83, no. 5, p. 054401, May 2020. [Online]. Available: http://dx.doi.org/10.1088/1361-6633/ab85b8

[5] N. S. Dattani and N. Bryans, “Quantum factorization of 56153 with only 4 qubits,” 2014.

[6] S. Jiang, K. A. Britt, A. J. McCaskey, T. S. Humble, and S. Kais, “Quantum annealing for prime factorization,” *Scientific Reports*, vol. 8, no. 1, Dec. 2018. [Online]. Available: http://dx.doi.org/10.1038/s41598-018-36058-z

[7] D. Willsch, M. Willsch, F. Jin, H. De Raedt, and K. Michielsen, “Large-scale simulation of shorâs quantum factoring algorithm,” *Mathematics*, vol. 11, no. 19, 2023. [Online]. Available: https://www.mdpi.com/2227-7390/11/19/4222

[8] A. Politi, J. Matthews, and J. O’Brien, “Shor?s quantum factoring algorithm on a photonic chip,” *Science*, vol. 325, no. 5945, p. 1221, September 2009. [Online]. Available: https://eprints.soton.ac.uk/358548/

[9] E. Martin-Lopez, A. Laing, T. Lawson, R. Alvarez, X.-Q. Zhou, and J. L. OâBrien, “Experimental realization of shorâs quantum factoring algorithm using qubit recycling,” *Nature Photonics*, vol. 6, no. 11, p. 773â776, Oct. 2012. [Online]. Available: http://dx.doi.org/10.1038/nphoton.2012.259

[10] C. Gidney and M. Ekerå, “How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits,” *Quantum*, vol. 5, p. 433, Apr. 2021. [Online]. Available: https://doi.org/10.22331/q-2021-04-15-433

[11] A. W. Services, “Compare simulators - amazon braket,” 2024, accessed: 2024-09-05. [Online]. Available: https://docs.aws.amazon.com/braket/latest/developerguide/choose-a-simulator.html

[12] G. Q. AI, “Quantum simulator | google quantum ai,” 2024, accessed: 2024-09-05. [Online]. Available: https://quantumai.google/qsim

[13] Q. D. Team, “The extended stabilizer simulator - qiskit aer 0.15.0,” 2024, accessed: 2024-09-05. [Online]. Available: https://qiskit.github.io/qiskit-aer/tutorials/6_extended_stabilizer_tutorial.html

[14] I. Quantum, “Ibm quantum platform,” 2024, accessed: 2024-09-05. [Online]. Available: https://quantum.ibm.com/services/resources

[15] T. Tanaka, Y. Suzuki, S. Uno, R. Raymond, T. Onodera, and N. Yamamoto, “Amplitude estimation via maximum likelihood on noisy quantum computer,” *Quantum Information Processing*, vol. 20, no. 9, Sep. 2021. [Online]. Available: http://dx.doi.org/10.1007/s11128-021-03215-9

[16] M. L. Dahlhauser and T. S. Humble, “Modeling noisy quantum circuits using experimental characterization,” *Physical Review A*, vol. 103, no. 4, Apr. 2021. [Online]. Available: http://dx.doi.org/10.1103/PhysRevA.103.042603

[17] C. Duckering, J. M. Baker, A. Litteken, and F. T. Chong, “Orchestrated trios: compiling for efficient communication in quantum programs with 3-qubit gates,” in *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS ’21)*. Association for Computing Machinery, 2021, pp. 375–385.

[18] IBM Quantum, “Noisemodel,” https://docs.quantum.ibm.com/api/qiskit/0.40/qiskit_aer.noise.NoiseModel, 2021, accessed: 2024-09-05.

[19] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[20] Qiskit Development Team, “thermal_relaxation_error,” https://qiskit.github.io/qiskit-aer/stubs/qiskit_aer.noise.thermal_relaxation_error.html, 2024, accessed: 2024-09-05.

[21] N. SÃ¡, I. S. Oliveira, and I. Roditi, “Towards solving the bcs hamiltonian gap in near-term quantum computers,” *Results in Physics*, vol. 44, p. 106131, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2211379722007458

[22] IBM Quantum, “Readouterror,” https://docs.quantum.ibm.com/api/qiskit/0.37/qiskit.providers.aer.noise.ReadoutError, 2021, accessed: 2024-09-05.

[23] R. L. Rivest, A. Shamir, and L. Adleman, “A method for obtaining digital signatures and public-key cryptosystems,” *Commun. ACM*, vol. 21, no. 2, p. 120â126, feb 1978. [Online]. Available: https://doi.org/10.1145/359340.359342

[24] Qiskit, “Shor’s algorithm,” https://github.com/Qiskit/textbook/blob/main/notebooks/ch-algorithms/shor.ipynb, 2024, accessed: 2024-09-05.

[25] S. Beauregard, “Circuit for shor’s algorithm using 2n+3 qubits,” 2003. [Online]. Available: https://arxiv.org/abs/quant-ph/0205095

[26] D. Gottesman, “An introduction to quantum error correction and fault-tolerant quantum computation,” 2009.

[27] S. S. Zhou, T. Loke, J. A. Izaac, and J. B. Wang, “Quantum fourier transform in computational basis,” *Quantum Information Processing*, vol. 16, no. 3, Feb. 2017. [Online]. Available: http://dx.doi.org/10.1007/s11128-017-1515-0

[28] A. J., A. Adedoyin, J. Ambrosiano, P. Anisimov, W. Casper, G. Chennupati, C. Coffrin, H. Djidjev, D. Gunter, S. Karra, N. Lemons, S. Lin, A. Malyzhenkov, D. Mascarenas, S. Mniszewski, B. Nadiga, D. O’Malley, D. Oyen, S. Pakin, L. Prasad, R. Roberts, P. Romero, N. Santhi, N. Sinitsyn, P. J. Swart, J. G. Wendelberger, B. Yoon, R. Zamora, W. Zhu, S. Eidenbenz, A. BÃ¤rtschi, P. J. Coles, M. Vuffray, and A. Y. Lokhov, “Quantum algorithm implementations for beginners,” 2018.

[29] E. Desurvire, *Classical and Quantum Information Theory: An Introduction for the Telecom Scientist*. Cambridge University Press, 2009.