Implementing Quantum Error Correcting Codes on the IBM Melbourne Quantum Computer

Ali Hindy

Abstract

We study the performance of different circuits implementing the Shor code circuit, a quantum error correcting code that can correct arbitrary single qubit error. Assuming unbiased noise, we implement and compare the performance of two different circuits implementing the Shor Code , one with an intermediate syndrome measurement and one without. We show that the Shor code without intermediate syndrome measurements can provide sufficient protection against arbitrary single qubit errors. Implementing the Shor Code using IBM Qiskit, we calculated the efficacy of both circuits. We also derived an upper bound for the probability of error occurring across the circuit and optimized the circuit in order to reduce this upper bound.

1 Introduction

Quantum Error Correcting Codes is a sub-field of quantum computing generally related to correcting the error that real quantum computers make when executing computations. Quantum computers have the potential to analyze data much faster than classical computers, and to discover new medicines, predict stocks, improve cybersecurity, and simulate complicated chemical equations [8]. However, quantum computers must run under very low temperatures. If the temperature rises, the computer may make a small error, which can eventually snowball into massive errors that completely ruin results [1].

A quantum circuit is similar to a classical circuit in the sense that quantum bits (qubits) pass through gates (matrix operations) analogous to the classical gates NOT, AND, OR. One such circuit is a quantum error correcting code called the Shor Code, which encodes a single qubit state into a nine qubit state [9].

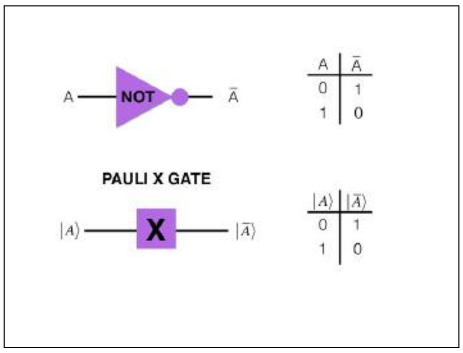


Fig. 1: A Pauli X gate, which is analogous to the classical NOT gate

Instead of 0s and 1s (classical bits), quantum computers operate off of $|0\rangle$s and $|1\rangle$s (qubits) and generally rely on quantum entanglement in order to operate. $|0\rangle$ and $|1\rangle$ (Dirac notation) are the basis vectors for two-dimensional complex space. Quantum entanglement is the property of two or more qubits being highly correlated, meaning they are in a quantum state that generally can not be written as a product state of independent single qubits. An example of an entangled state is the 2 qubit $|\Phi^+\rangle$ Bell State.

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

In order to construct this Bell State, a quantum gate called a CNOT gate, can be applied to two qubits. Quantum gates are matrices that can be applied to one or more qubits. With multiple quantum gates, a quantum circuit can be formed. Quantum entanglement allows multiple qubits in an entangled state to be acted upon simultaneously, which is not possible in classical computing as classical bits can only have one state at a time. A physical qubit is a qubit on hardware while a logical qubit is an abstract (not necessarily hardware or software) qubit. One such application of quantum entanglement is with the Shor Code, which encodes one logical qubit onto nine physical qubits so that if an error on a single qubit occurs, we can figure out what the nine qubit state looked like.

Quantum error correcting codes work by using ancilla (helper) qubits in order to add redundancy to the original data so that if some data gets corrupted, the original data can still be recovered. Our implementation of Quantum Error Correction works in three steps: a single qubit is encoded onto multiple other qubits, then random error is simulated, and finally the error is simultaneously corrected and decoded to recover the original state. In the case of the Shor Code, $|0\rangle$ and $|1\rangle$ are encoded as follows:

$$|0_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle)$$

$$|1_L\rangle = \frac{1}{2\sqrt{2}}(|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle)$$

In both of these equations, a single logical qubit is encoded onto nine physical qubits, which is the first step of quantum error correction. The states are encoded using tensor ($\otimes$) products. Each of the $(|000\rangle + |111\rangle)$ terms refers to the groups of qubits (1,2,3), (4,5,6), and (7,8,9), which is used to correct X errors. With these nine qubits, if one mistake occurs, we can correct it using the decoding circuit. The next step of quantum error correction is to simulate error by doing some arbitrary random operation. The Shor Code can correct any arbitrary single qubit error, which can be composed by an X, bit flip error, and a Z, phase flip error. An X error turns a $|0\rangle$ into a $|1\rangle$ and a $|1\rangle$ into a $|0\rangle$, while a Z error turns a $|0\rangle$ into a $|0\rangle$ and a $|1\rangle$ into a $-|1\rangle$ These two types of error correspond to two Pauli matrices:

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

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

Every error matrix can be written as a linear combination of I,X,Z,and XZ, and, the Shor Code can hence correct every type of arbitrary single qubit error.

1.1 The Project

The goal of this project is to implement the Shor Code on a real quantum computer, the IBM Melbourne, using Python packages such as IBM Qiskit in order to measure if the theoretical efficacy is the same as the practical efficacy. We plan on contributing to IBM Ignis, an open source Python framework for quantum computing that does not currently include the Shor Code. There also exist in the literature other quantum error correcting codes including the Repetition Code and the Bacon-Shor Code, but the only one in the IBM Ignis documentation is the Repetition Code.

Figure 2 depicts the Shor Code circuit, which includes the encoding, random error (E), and decoding. This circuit does not include intermediate measurements, and only measures the first qubit at the end of all the operations. As seen in Figure 2, a quantum circuit is comprised of matrix operations such as the CNOT gate and the Hadamard gate.

The IBM Melbourne is a fifteen qubit quantum computer in Melbourne, Australia. The quantum computer was made publicly available only recently, which has led to a spike in the number of research papers that practically implement quantum algorithms. The IBM Melbourne has the greatest number of qubits that IBM publicly offers. Novel quantum technology called Noisy Intermediate State Quantum (NISQ) (50-100 qubit devices) is not publicly available; however, it has the potential to completely change the field of quantum computing.[7] .[4] IBM computers like the IBM Rochester have powerful capabilities for the future of quantum computing. Furthermore, quantum error correcting codes like the Shor Code

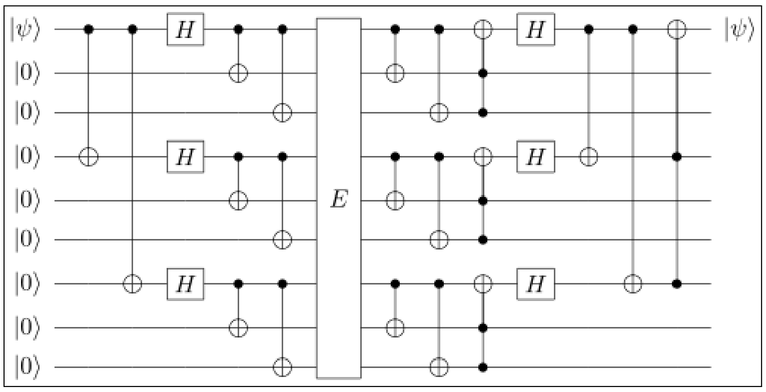


Fig. 2: The Shor Code circuit

will become significantly more important in reducing noise on a quantum circuit. NISQ technology will facilitate novel cybersecurity methods, the discovery of new medicines, and much more. By implementing the Shor Code on a fifteen qubit device, progress will be made towards making quantum computations more precise.

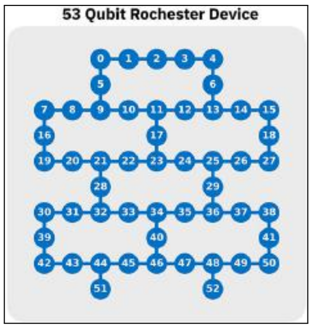


Fig. 3: The IBM Rochester, IBM's largest quantum computer (not publicly available)

Noise models are a useful tool for simulating the error on real quantum computers. We consider noise models such that qubits in the code block are subject to both bit flip (X) errors and phase (Z) errors, where the bit flips occur with probability p_X and phase errors occur with probability p_Z. We assume that the noise acts independently on each qubit,and that the X and Z errors are uncorrelated. We also assume that each error happens with equal probability, which is known as unbiased noise. However, on the real IBM Melbourne, there are other types of error such as crosstalk error that cannot be accurately accounted for in a simulator.

2 Literature Review

Quantum error correction is a relatively new field of science (from the past three decades) and new papers are habitually produced that make progress in the field. However, there are some limitations in the field, which include lacking practical results.[5] Most papers only theorize the quantum error correcting codes instead of actually implementing them, since quantum computers were not widely available until recently. The advent of IBM Qiskit has allowed for an exponential increase in the number of research papers related to practical implementation of previously theoretical ideas.

The IBM Melbourne's architecture provides for connectivity-3 operations, which is ideal for the Shor Code. A connectivity-3 operation is an operation involving connections with 3 qubits, which the Melbourne provides in its base architecture.[4] The IBM Melbourne also comes with a quantum transpiler, which compiles and transforms the circuit in order to minimize the number of gates before it runs on the Melbourne.[3]

For example, the Shor Code has a connectivity-4 qubit on qubit 0, which is not possible on the Melbourne given the architectural constraints. However, the transpiler avoids this issue by adding multiple gates and using extra gates in order to avoid this issue.

We read papers that ran other quantum error correcting codes such as the Bacon-Shor Code on a simulator.[**?**] Such papers numerically calculate the performance of the codes and at best use a simulator. However, in order to test the real performance of quantum error correcting codes, one must run code on a real quantum computer, as opposed to a simulator or theorizing the accuracy.

3 Purpose

1. Learn more about quantum computing / encode a real circuit
2. Contribute to the open source IBM Qiskit Project
3. Successfully encode, run a random noise generator, and decode on the IBM Melbourne

4 Methods

For programming, we used Python coupled with scientific computing packages like Numpy, as well as plotting packages like Matplotlib. Additionally, we used IBM Qiskit, a python library that allows interfacing with real quantum computers, in order to code a quantum circuit. In terms of studying the theory of quantum computing, we read the Nielsen-Chuang textbook, lecture notes from graduate classes at Harvard and the University of Waterloo, as well as more recent papers in quantum computing [2]. We first implemented the Shor Code on the IBM Melbourne with an encoding circuit, error generator, and decoding circuit all in that order. With an input of 0, we measured the probability of each bit occurring. If the input is a 0 and the output is a 1, then we know that the circuit failed to correct the error. Since the IBM Melbourne was down for maintenance for 2 weeks, we used a noise model and simulator for the Melbourne and compared the results.

4.1 Connectivity

Although the Melbourne is a powerful quantum computer, it is limited in the number of connections that certain qubits have with each other. There is a limited number of connections for 2 qubit operations, with some qubits like qubit 7 only being able to execute a 2 qubit operation with qubit 8.

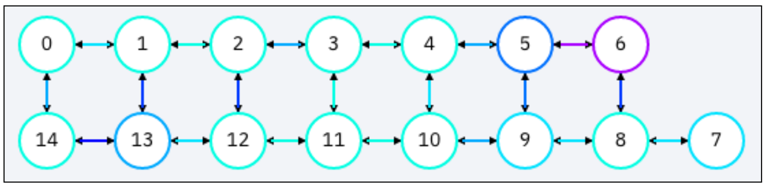


Fig. 4: The Architecture of the IBM Melbourne, with each circle representing a qubit and each line representing a connection for a possible 2-qubit operation

As seen in Figure 7, 3 qubit operations are not possible on the IBM Melbourne. When the transpiler receives a 3 qubit operation such as the Toffoli gate, it must decompose the gate into a series of 2 qubit operations. The Toffoli gate, as given by

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

can be deconstructed into 6 CNOT gates and a combination of H, T, and $T^\dagger$ gates [1]. However, as previously mentioned, the CNOT gate has a high probability of error and as such makes the circuit less efficient in correcting error.

4.2 Circuits

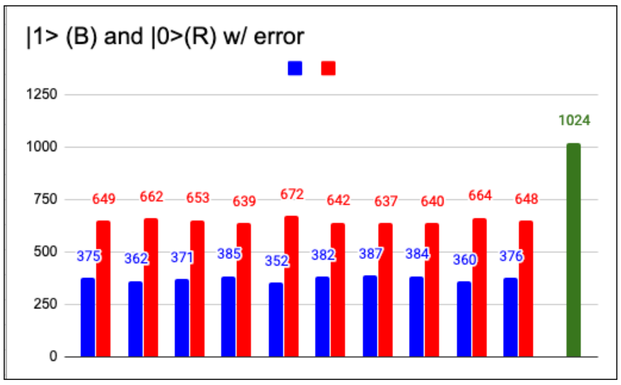
Since the current architectures that IBM provides do not allow for intermediate measurements, we applied the Principle of Deferred Measurement, which states that intermediate measurements can always be moved to the end of the circuit. [6] However, on the simulator, intermediate measurements can be made with the same noise model as the Melbourne. 7 Different experiments were executed:

1. Shor code without added error (no intermediate measurement) on a simulator
2. Shor Code with added error (no intermediate measurement) on a simulator
3. Optimized Shor Code circuit with the added error on a simulator
4. Shor Code with an intermediate measurement without added error
5. Shor Code with an intermediate measurement with added error on a simulator
6. Shor Code without added error on the IBM Melbourne
7. Shor Code with added error on the IBM Melbourne

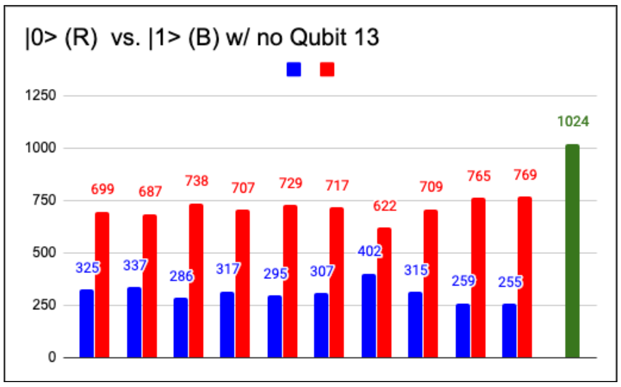
5 Results and Discussion

5.1 Results

For each circuit, we conducted multiple trials, with each trial consisting of 1024 iterations of the circuit, and graphed the results. Additionally, the upper bound for the probability of error (p_e) was manually calculated.



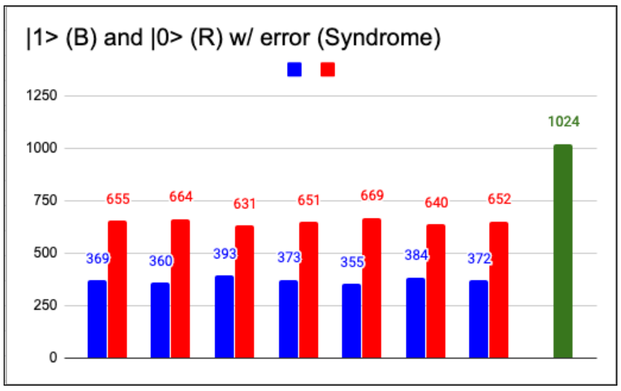
(a) Results with an Error Generator



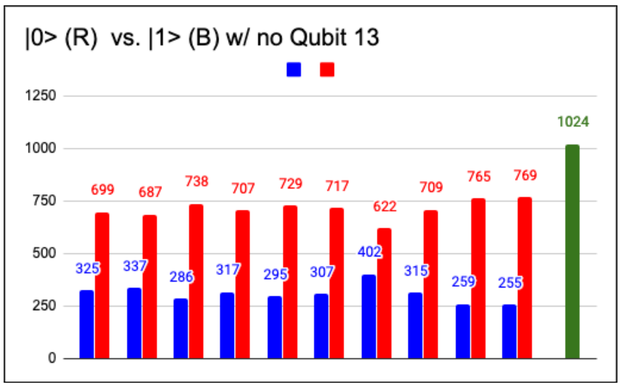
(b) Results with an Optimized Circuit

Fig. 5: Comparison of Non-Optimized vs. Optimized Circuit Performance

As seen in Figure 4, the results with the optimized circuit on the simulator are better than without the optimized circuit. After 1024 iterations of each circuit, the number of $|0\rangle$s and $|1\rangle$s were measured, with the red bar showing the number of $|0\rangle$s and the blue bar showing the number of $|1\rangle$s. Ideally, the output of the circuuit should all be $|0\rangle$ since the Shor Code is equivalent to the identity matrix for Qubit 0. Each pair of bars represents one experiment, where the circuit was ran 1024 times.In order to optimize the circuit, certain qubits with a high CNOT error rate, such as Qubit 13, were replaced with qubits with a lower CNOT error rate, such as Qubit 8. The probability of error for the non-optimized circuit was 36.45 percent while the probability of error for the optimized circuit was 30.25 percent. The green bar shows the ideal performance, where all error is corrected. However, the Shor Code only corrects for single qubit errors, and other errors such as crosstalk add noise that the Shor Code can not correct.



(a) Results with an Optimized Syndrome Circuit



(b) Results with an Optimized Deferred Circuit

Fig. 6: Comparison of Syndrome vs. Deferred Circuit Performance

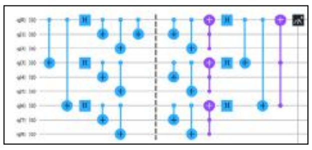
As seen in Figure 5, the results with the optimized circuit without an intermediate measurement surpass the results with the optimized circuit with an intermediate measurement. The intermediate syndrome measurement occurs directly after the error occurs, while the deferred measurement happens at the end, which looks like Figure 2. This higher accuracy is most likely due to the higher error rate that measurement operations have on the Melbourne. Additionally, the syndrome measurement circuit has 85 CNOT gates, while the non-syndrome measurement circuit has only 78. CNOT gates have a high probability of error, so higher amounts of CNOT gates decreases the performance of the circuit. The probability of error for the syndrome measurement circuit was 43.8 percent while the probability of error for the optimized non-syndrome circuit was 30.25 percent.

5.2 Dedicated Use / Maintenance

One setback I faced was the dedicated use of the Melbourne for other researchers. More prominent researchers at IBM have first priority to the Melbourne, and during Week 4 they queued hundreds of circuits to the Melbourne, essentially blocking my access to the computer. In order to overcome this issue, I experimented with other circuits such as the 5-qubit stabilizer code and used the noise model and simulator. The noise model captured the error parameters of the IBM Melbourne and applied that to each qubit in the circuit. Additionally, the Melbourne was down for 2 weeks during my research, and as a result I had to turn to the simulator in order to continue getting results. No other publicly avaliable quantum computer has more than 5-qubit capability, meaning that my two options were to use a simulator or try a 5-qubit stabilizer code. However, the 5-qubit stabilizer code requires 4 ancilla qubits, so I could not use any other publicly avaliable IBM architecture.

5.3 Transpiler

On IBM Qiskit, every circuit that runs through a real quantum computer is sent through the transpiler, which changes the circuit in order to match the qubits of the IBM Melbourne. The transpiler is not optimal for running the circuit, as it merely tries to match the circuit to the qubits of the Melbourne without taking into consideration the number of CNOT gates. Since CNOT gates have a high probability of error, the upper bound for the probability of a non-optimized circuit is 92.48 percent.



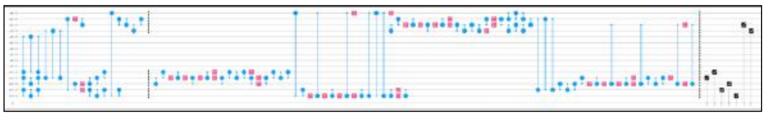


Fig. 7: Initial Circuit (top) vs Transpiled Circuit (bottom)

As seen in Figure 4, the initial circuit has 18 CNOT operations and 4 Toffoli operations, whereas the transpiled circuit has 78 CNOT operations, which drastically increases the probability that error occurs. Additionally, the transpiler uses some qubits (for example, Qubit 13) that have a high CNOT error percentage relative to other qubits. As a consequence, we had to optimize the circuit by trying variations of the transpiled circuit as well as using different qubits with a lower CNOT error rate.

5.4 Architecture Limitations

Although the IBM Melbourne is a state of the art quantum computer, it does not support intermediate syndrome measurements. As such, we could not run the intermediate measurement circuit on the IBM Melbourne and had to use the simulator.

Tab. 1: Error Parameters for the IBM Melbourne

Qubit	Measurement Error	Single Qubit Error	CNOT Error
Q0	2.75E-02	9.59E-04	cx$0_1 : 2.433e-2, cx01_4 : 4.758e-2$
Q1	5.15E-02	1.10E-03	cx$1_0 : 2.433e-2, cx1_2 : 1.088e-2, cx1_13 : 6.045e-2$
Q2	2.05E-02	8.02E-04	cx$2_1 : 1.088e-2, cx2_3 : 2.008e-2, cx2_12 : 3.508e-2$
Q3	6.60E-02	4.49E-04	cx$3_2 : 2.008e-2, cx3_4 : 1.775e-2, cx3_11 : 3.060e-2$
Q4	3.40E-02	1.70E-03	cx$4_3 : 1.775e-2, cx4_5 : 2.513e-2, cx4_10 : 2.430e-2$
Q5	6.40E-02	2.54E-03	cx$5_4 : 2.513e-2, cx5_6 : 6.955e-2, cx5_9 : 5.338e-2$
Q6	2.55E-02	4.18E-03	cx$6_5 : 6.955e-2, cx6_8 : 1.662e-1$
Q7	5.10E-02	2.85E-03	cx$7_8 : 3.683e-2$
Q8	2.70E-01	7.11E-04	cx$8_6 : 1.662e-1, cx8_7 : 3.683e-2, cx8_9 : 3.717e-2$
Q9	5.15E-02	3.18E-03	cx$9_5 : 5.338e-2, cx9_8 : 3.717e-2, cx9_10 : 5.028e-2$
Q10	2.30E-02	1.15E-03	cx$10_4 : 2.430e-2, cx10_9 : 5.028e-2, cx10_11 : 2.536e-2$
Q11	4.25E-02	6.11E-04	cx$11_3 : 3.060e-2, cx11_10 : 2.536e-2, cx11_12 : 1.826e-2$
Q12	2.90E-02	5.15E-04	cx$12_2 : 3.508e-2, cx12_11 : 1.826e-2, cx12_13 : 2.821e-2$
Q13	1.09E-01	1.74E-03	cx$13_1 : 6.045e-2, cx13_12 : 2.821e-2, cx13_14 : 4.870e-2$
Q14	3.45E-02	6.98E-04	cx$14_0 : 4.758e-2, cx14_13 : 4.870e-2$

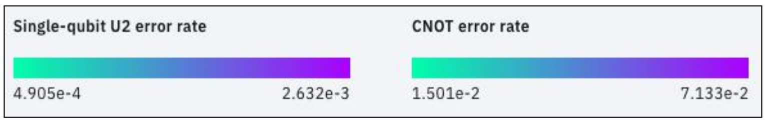


Table 1 shows the Measurement Error, Single Qubit Error, and CNOT Error Calibrations for the IBM Melbourne. The probability of a Single Qubit Error is much lower than both the Measurement Error and CNOT Error, and as a result, CNOTs and measurement operations are the most likely to cause an error. As such, the goal of optimizing the circuits was to minimize the number of CNOT operations. Since there was only 1 final measurement on Qubit 0 in all of the circuits that were passed through the Melbourne, minimizing the number of CNOTs was the priority. While the transpiler drastically increased the number of CNOT operations, applying CNOT operations on certain qubits (such as between qubit 1 and 0) will yield less error than between qubits 8 and 6, which has a 0.1662 percent chance of error. However, some qubits have more possible connections than others, which yields the issue of connectivity.

6 Conclusion

We have studied the performance of multiple versions of Shor Code on the IBM Melbourne and on the simulator against unbiased noise for single qubit errors, concluding that it is possible to optimize the circuit for the given quantum computer. Additionally, using the principle of deferred measurement to minimize the number of measurements also decreases the probability of error occuring.

Further study is required to test the performance of other quantum error correcting codes. Using real hardware instead of a simulator raises a variety of problems, including connectivity issues and other types of error such as crosstalk. Optimizing circuits for different computers requires knowledge of the computer's architecture, and further study is required to learn whether there is a general solution for minimizing the number of gates and hence the probability of error. We expect, however, that this analysis of the performance of the Shor Code will help guidance towards fault tolerant architectures.

7 Acknowledgements

I would like to thank my mentor, Abhijit Mudigonda, for his constant support of my project. Without him, I could never have done this project. Only with his clear explanations of concepts that made no sense to me and his motivation for me to continue my research was I able to complete my project!

References

[1] Panos Aliferis and Andrew W. Cross. Subsystem fault tolerance with the bacon-shor code. *Physical Review Letters*, 98, 05 2007.

[2] Daniel Chiu, Franklyn Wang, and Scott Duke Kominers. Generalization by recognizing confusion. *arXiv:2006.07737 [cs, stat]*, 06 2020.

[3] Cramer J. Repeated quantum error correction on a continuously encoded qubit by real-time feedback, 2016.

[4] N. David Mermin. Shor's 9-qbit error-correcting code. *Quantum Computer Science*, pages 207–209.

[5] Mikio Nakahara and Tetsuo Ohmi. *Quantum computing : from linear algebra to physical realizations*. Crc Press, 2008.

[6] Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge Cambridge University Press, 2019.

[7] R. Padma Priya and A. Baradeswaran. An efficient simulation of quantum error correction codes. *Alexandria Engineering Journal*, 57:2167–2175, 09 2018.

[8] S. A. Selesnick. Foundation for quantum computing ii. *International Journal of Theoretical Physics*, 46:984–1002, 12 2006.

[9] Sixia Yu, Qing Chen, C. H. Lai, and C. H. Oh. Nonadditive quantum error-correcting code. *Physical Review Letters*, 101, 08 2008.