EasyChair Preprint

№ 9075

Optimization of Quantum Read-Only Memory Circuits

Koustubh Phalak, Mahabubul Alam, Abdullah Ash-Saki,
Rasit Onur Topaloglu and Swaroop Ghosh

October 24, 2022

Optimization of Quantum Read-Only Memory Circuits

Koustubh Phalak
CSE Department
Pennsylvania State University
State College, PA, USA
krp5448@psu.edu

Mahabubul Alam
EE Department
Pennsylvania State University
State College, PA, USA
mxa890@psu.edu

Abdullah Ash-Saki
EE Department
Pennsylvania State University
State College, PA, USA
axs1251@psu.edu

Rasit Onur Topaloglu
IBM
Hopewell Junction, NY, USA
rasit@us.ibm.com

Swaroop Ghosh
EECS Department
Pennsylvania State University
State College, PA, USA
szg212@psu.edu

***Abstract*—Quantum computing is a rapidly expanding field with applications ranging from optimization all the way to complex machine learning tasks. Quantum memories, while lacking in practical quantum computers, have the potential to bring quantum advantage. In quantum machine learning applications for example, a quantum memory can simplify the data loading process and potentially accelerate the learning task. Quantum memory can also store intermediate quantum state of qubits that can be reused for computation. However, the depth, gate count and compilation time of quantum memories such as, Quantum Read Only Memory (QROM) scale exponentially with the number of address lines making them impractical in state-of-the-art Noisy Intermediate-Scale Quantum (NISQ) computers beyond 4-bit addresses. In this paper, we propose techniques such as, pre-decoding logic and qubit reset to reduce the depth and gate count of QROM circuits to target wider address ranges such as, 8-bits. The proposed approach reduces the number of gates and depth count by at least 2X compared to the naive implementation at only 36% qubit overhead. A reduction in circuit depth and gate count as high as 75X and compilation time by 85X at the cost of a maximum of 2.28X qubit overhead is observed. Experimentally, the fidelity with the proposed pre-decoding circuit compared to existing optimization approach is also higher (as much as 73% compared to 40.8%) under reduced error rates.**

***Index Terms*—Quantum read-only memory, NISQ, Noise resilience**

I. Introduction

Memory acts as a bridge between the processor and the storage element, where the frequently accessed data is stored in a volatile or non-volatile manner. This saves time from fetching the data from a slower storage element. In the quantum domain, all qubits are initialized to a value such as $|0\rangle$. Therefore, data needs to be loaded in the circuit first before performing a computation. The data loading can be performed via various embedding methods such as amplitude embedding, angle embedding, and hybrid embedding [1]. Each of these methods bring their own set of benefits and challenges. Amplitude embedding for example can accommodate 2^n data into n qubits at higher circuit depth and gate count, possibly degrading the fidelity of computation. Angle embedding, on the other hand, encodes the data as the rotation angle along X/Y/Z axis of a single qubit rotation gate. Therefore, n data points can be loaded onto n qubits relieving the maximum qubit count limitations. One can encode more than one data in a qubit by cascading rotation gates [2] at the cost of increased circuit depth. In quantum machine learning (QML) applications, data loading presents significant training time overhead since the classical dataset needs to be uploaded in quantum domain iteratively and the output sampled in classical domain to determine the gradient and optimize the parameters. Beyond on within these techniques, efficient data encoding is still an active area of research.

In QML applications, quantum memory can simplify the training since the data can be loaded and processed within the quantum circuit without converting to the classical domain. Quantum memory can also store intermediate quantum states during computation to reclaim the qubits. Various circuit-based Quantum Random Access Memory (QRAM) and Quantum Read Only Memory (QROM) [3], [4] circuits using Noisy Intermediate-Scale Quantum (NISQ) computers have been proposed. However, they incur exponential circuit depths and gate counts with the number of address lines degrading the fidelity of the computation. This renders the quantum memory slow and of limited practical use. Optimizations techniques are warranted to address this challenge.

Our work is related to the QROM implementation [4] where a common sub-circuit called 'unary iteration' is used for controlling the data to be sent on the data lines. However, this circuit uses multi-controlled NOT gates which decompos into large number of basis gates increasing the depth as the number of control lines (which is a function of address width in QROM) grows. Moreover, the number of such multi-controlled NOT gates grows exponentially with increasing address lines due to the structure of the QROM circuit. A sawtooth circuit is implemented for optimization where multi-controlled NOT gates are broken down into large number of Toffoli gates

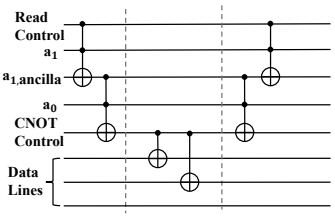

Fig. 1. Sawtooth circuit implementation of QROM as proposed in [4]

with the help of ancilla qubits added between address qubits. However, the count of multi-controlled NOT gates even in the optimized circuit once again is exponential. Thus, the increase of overall gate count is exponential with increasing number of address lines leaving room for more robust optimization. In this paper, we propose optimizations of the unary iteration sub-circuit to reduce the **i)** compilation time, **ii)** gate count, and **iii)** circuit depth of the QROM circuit for faster access.

In the remaining of the paper, Section 2 presents the relevant background details and prior art. Section 3 describes the optimizations performed on the QROM circuit. Section 4 compares the results with the naive QROM implementation. The limitations are also discussed. Section 5 presents a general discussion. Finally, Section 6 concludes the paper.

II. Background and Related Work

A. Quantum computing fundamentals

A deep coverage of this is not possible in this work; we herein introduce most relevant aspects to jump-start the reader.
Qubits: Quantum bits or qubits are the fundamental units of a quantum computer. While classical bits can have two possible values of zero or one, qubits have quantum states denoted using the ket notation $|\psi\rangle$. This state can carry the probability a^2 of it being $|0\rangle$, and the probability b^2, of it being $|1\rangle$, where a and b are the complex numbers with (a^2+$b^2 = 1$). Therefore, a qubit can exist in both states simultaneously. Qubits are also represented in matrix form. For example, $|0\rangle$ is denoted as $\left[\begin{smallmatrix}1\\0\end{smallmatrix}\right]$ and $|1\rangle$ is denoted as $\left[\begin{smallmatrix}0\\1\end{smallmatrix}\right]$.
Qutrits: Qutrits are a ternary version of qubits which can store states of three classical bits and their superpositions. In the context of QRAM, a qutrit has left, right, and wait base states ([5]).
Quantum Gates: A quantum circuit has quantum gates, which perform operations on qubits and change their state. Quantum gates can be represented as a unitary matrix.

The most frequently used categories of gates are single qubit gates, which operate on one qubit, and two qubit gates, which operate on two qubits at once. Hadamard (H), Bit flip (X) and Rotation gate (RX, RY, RZ) are commonly used single qubit gates while Controlled Not (CNOT) is a commonly used two-qubit gate. Gates with more than two qubits also exist, e.g., Contolled Swap, Peres, Toffoli, iToffoli. Every quantum hardware has a set of associated basis gates which every complex gate is mapped onto. For example, the current IBM backends use RZ, ID, X, SX, and CNOT gates as the basis gates. When a quantum circuit is sent to the quantum hardware, the complex multi-qubit gates are decomposed into these basis gates. The gate count and the overall depth of the decomposed quantum circuit depends on the complexity of the available multi-qubit gates.

Quantum Errors: One of the major challenges faced by modern NISQ computers is the presence of errors. Individual qubits are subject to decoherence errors. Decoherence error affects qubit relaxation time, i.e. time it takes for the qubit to drop to a relaxed state from an excited state and dephasing, i.e., deviation from the correct phase. Gate error occurs when a quantum gate gives a wrong computational output due to various systemic and environmental factors. When two gate operations are performed in parallel on neighboring qubits they interfere with each other to corrupt the qubit state. This is called crosstalk error. Readout error or measurement error occurs while measuring a qubit from the quantum to classical state. Decoherence, gate, and crosstalk errors accumulate with circuit size, therefore, shallow and small quantum circuits are preferred for noisy quantum computers.

B. QROM

In Read-Only Memory (ROM), the user can only read the data but cannot write into it. The QROM circuit has a read control signal, address lines, an extra ancilla qubit for CNOT control of data, and data lines. The read control signal provides the read signal to the quantum memory. The user can read the data only if the read signal is in state $|1\rangle$. When the read signal is active, the user will provide valid input on the address lines and get the output on the data lines. The data output depends on the Multi-controlled CNOT gate that gets activated based on the value of address lines. Initially, all the data lines are in $|0\rangle$ state. The naive implementation of the QROM circuit (Fig. 2) contains two address lines and four data lines. To optimize this naive implementation, [4] proposed the sawtooth circuit of the unary iteration. In the sawtooth circuit, ancilla qubits are inserted between the address qubits, and the multi-controlled NOT gate is broken down into Toffoli gates. Assuming two address lines, there will be two Toffoli gates. The first Toffoli gate will be controlled by read control and the MSB (Most Significant Bit: a_1, Least Significant Bit: a_0) a_1 with target at $a_{1,ancilla}$, and the second Toffoli gate will have be controlled by $a_{ancilla}$ and a_0 with target at the CNOT control line. This circuit is shown for a single datapoint in Fig. 1. In this approach, $O(n)$ extra qubits are required for n address lines.

C. Related Work

Our work is closely related to [4] which primarily focuses on usage of quantum computing for quantum physics and quantum chemistry. One of the techniques used in their circuits is ‘unary iteration’, which is a set of control qubit lines to perform control operations. This unary iteration has been used in QROM circuit as one of it’s applications. However, there is little optimization performed on the unary iteration circuit

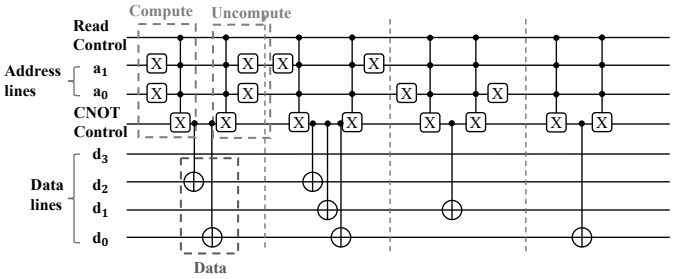

Fig. 2. Naive implementation of QROM circuit with two address lines. In this example, the address and data values are as follows: QROM[00] = 0101, QROM[01] = 0111, QROM[10] = 0010, QROM[11] = 0001.

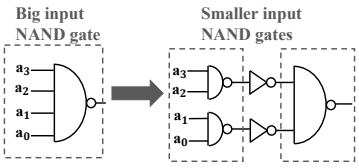

Fig. 3. Classical pre-decoding used in implementing wide input NAND gates (4-input in this example) in memory array decoding logic using small fanin NAND gates. The outputs from first two NAND gates (located in midlogic area) are the pre-decoded signals that are provided to the final NAND gate (located in wordline driver area) for decoding.

resulting in a deep and large gate count overhead over the QROM circuit (details in Section 3).

Other related works involve QRAM development. The quantum version of bifurcation graph-based RAM is proposed in [5] which utilizes qutrits to route the qubits to appropriate memory cells. The bifurcation graph-based RAM can be represented as a full binary tree in which the leaves represent the memory cells, and rest of the nodes are qutrits which assist in routing the qubits to the appropriate memory location. It has both exponential circuit width and circuit depth. Another work [6] analyzes the robustness of this QRAM architecture. Various possible architectures of QRAM are also proposed [7]. A Flip-Flop QRAM architecture is proposed in [3] which store data into qubits in the form of superposition of states. The flip stage (which is the compute stage) loads each data, a register stage stores the data into the register qubit, and a flop stage performs uncomputation on the data lines. The application of FF-QRAM is also extended to continuous amplitudes in [8], which extend their application to loading continuous data instead of only discrete data.

Potentials applications for quantum memories are also studied e.g., usage of Raman quantum memory for optical quantum computing [9]. A detailed explanation on quantum cryptography with integration of quantum memories into quantum repeaters has been presented in [10]. An application of quantum memories in quantum communication has been mentioned in [11].

III. Pre-decoding in QROM circuits

A. Naive Implementation

The naive implementation of the unary iteration sub-circuit of the QROM circuit [4] consists of a read control line, address lines, and a CNOT control line. For every data point, it consists of three stages: compute stage, data read stage, and uncompute stage as marked in three boxes respectively, in Fig. 2. In the compute stage, a multi-controlled not (MCX) gate is used where the controls are on the read control line and address lines, and the target is on the CNOT control line of the data lines. In general, $C^{n+1}X$ (controlled NOT gate having $n+1$ control signals) gates are required for control for n address lines. Moreover, X gates are added prior to the MCX gates to flip address lines in $|0\rangle$ state to $|1\rangle$ state and activate all the controls of the MCX gate pertaining to that particular address only. For example, in Fig. 2, if $a_1 = |0\rangle$ and $a_0 = |0\rangle$, then the first set of X gates will flip the state of both the address lines to $|1\rangle$ state. Assuming that the read control line is also at $|1\rangle$ state, only the first MCX gate of the compute will be triggered. This will flip the CNOT control line to $|1\rangle$ state, and the data lines will read the data using the CNOTs in between the MCX gates. Finally, an uncompute stage is required to flip the CNOT control state back to $|0\rangle$ otherwise the CNOTs designated for other addresses will get triggered corrupting the original data.

This naive structure of MCX gates works well when the number of address lines is small. However, as the number of address lines increases, so does the number of control lines required for the MCX gates which are broken down into basis gates during the compilation process. An MCX gate with $n+1$ controls takes at least 2X more number of basis gates than an MCX gate with n controls for proper decomposition.For example, decomposing an MCX gate with 5 controls will take more number of basis gates than two MCX gates with 4 controls combined. Thus, the decomposition of the MCX gates increases the gate count, and consequently, the overall gate count and depth of the QROM circuit drastically increases as the number of address lines increases. This structure of the naive QROM circuit leads to exponential number of such MCX gates. Suppose a QROM circuit has n address lines which can store 2^n data. For each data, two MCX gates are required, one each for the compute and uncompute stages. Therefore, in total $2*2^n = 2^{n+1}$ MCX gates will be used. Also, let's assume that in the best case, an MCX gate can be decomposed into $O(n)$ Toffoli gates ([12]). Therefore, the total number of Toffoli gates after decomposition becomes $O(n)*2^{n+1} = O(n*2^n)$. Toffoli gates can be broken down into basis gates using $O(1)$ basis gates. Therefore, overall gate count of the circuit will be $O(1)*O(n*2^n) = O(n*2^n)$. Therefore, the increase in gate count is exponential with the address lines. This exponential trend is shown in Fig. 7 which shall be explained later in detail in Section IV.

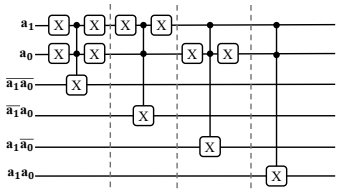

Fig. 4. Pre-decoding of two address lines in QROM circuit to generate 4 signals $\overline{a_1}\overline{a_0}$, $\overline{a_1}a_0$, $a_1\overline{a_0}$ and a_1a_0.

B. Proposed Pre-decoding Implementation

In order to prevent this exponential increase of gate count, we propose pre-decoding of address lines. Similar to pre-decoding performed in classical memory, a subset of address lines are taken and all possible combinations of their signals are generated beforehand prior to providing them as input to the final decoder. For example, it is difficult to realize a 4 input NAND gate inside wordline driver due to large footprint. To overcome this issue, the NAND gate is broken down into 2 input NAND gates by performing pre-decoding. Two pre-decoded signals are generated using the two NAND gates at two pairs of address lines, and a final NAND gate is used with two NOT gates in between on these two pre-decoded address lines. This is shown in Fig. 3. Therefore, design complexity is reduced at the cost of extra pre-decoded signals.

In QROM circuit, the pre-decoded signals are obtained on extra ancilla qubits before sending to the MCX gates. This reduces the number of control operations on the MCX gates, thereby shortening its decomposition. In general, for m address lines pre-decoded together, the m controls are reduced to just one single control. Moreover, multiple subsets of address lines can be pre-decoded separately for further reduction in gate count. Fig. 4 shows the pre-decoding operation on two address lines a_1 and a_0 to generate $2^2 = 4$ pre-decoded signals. Therefore, 4 extra ancilla qubits are required. For pre-decoding of m address lines, 2^m extra ancilla qubits are needed increasing the circuit width. However, this approach reduces the circuit depth and gate count improving the circuit performance under quantum errors. It should be noted though, that the increase in gate count will still be exponential, but it will not be as drastic due to the reduction in number of control signals of MCX gates.

Incorporating the pre-decoding scheme (e.g., Fig. 4) into the naive implementation (e.g., Fig. 2), we get the optimized version of the QROM circuit as shown in Fig. 5. Comparing both the naive and optimized implementations, we note a reduction in the larger C^3X gates (C^3X gates are MCX gates with 3 control lines including the read control. They can be seen in Fig. 2 in compute and uncompute stages and in Fig. 5 in the pre-decoding stage). This is because some address signals were already pre-decoded. We can further reduce the number of multi-controlled NOT gates by replacing the MCX gates in the uncompute stage with a reset gate. A reset gate is a single qubit gate which resets the state of the qubit state back to $|0\rangle$ state. In the uncompute stage, the CNOT control line is required to be reverted back to $|0\rangle$ state. This is because if the state of the CNOT control line is not reset, unwanted CNOT gates corresponding to other address lines may get triggered and output wrong data onto the data lines. A reset gate has a circuit depth of only 1, and does this job in the uncompute state. Therefore, replacing an MCX gate with a reset gate further optimizes the circuit parameters. We calculate the gate count and circuit depth, and compare them with the corresponding values for the naive implementation of the QROM circuit to further quantify the benefit of pre-decoding in QROM circuit.

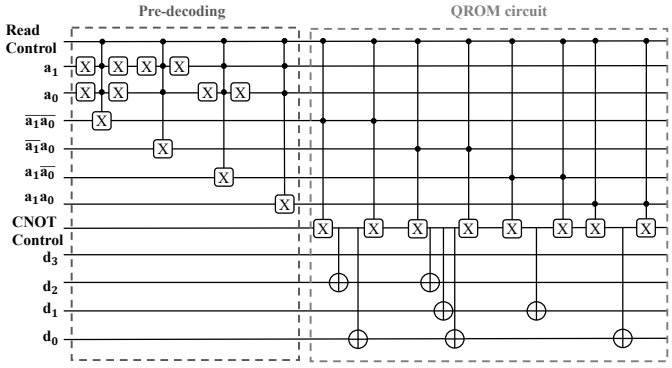

Fig. 5. Optimized QROM circuit. The number of $C^{n+1}X = C^3X$ gates has reduced compared to the naive implementation.

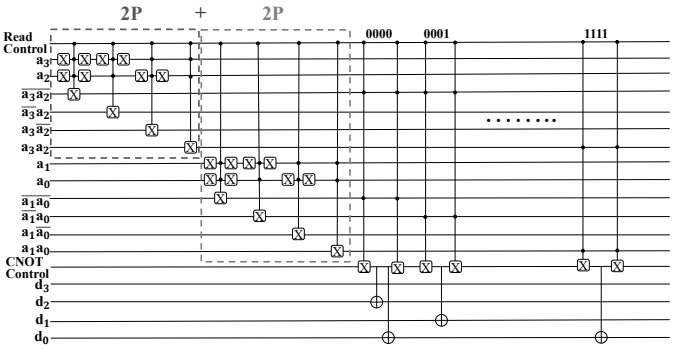

Fig. 6. Pre-decoded QROM circuit with 4 address lines pre-decoded as 2(**P**)+2(**P**).

IV. Results and Limitations

A. Results

Since subsets of address lines are used to pre-decode and obtain the pre-decoded signals, multiple such combinations of subsets are possible to offer space trade-off among circuit depth, gate count, compilation time, and number of extra qubits. For example, a few possible cases for 5 address lines can be,

1) Pre-decode 2 address lines and leave 3 address lines undecoded (2(**P**)+3(**U**))
2) Pre-decode 3 address lines and leave 2 address lines undecoded (3(**P**)+2(**U**))
3) Pre-decode two pairs of 2 address lines and leave the leftover 1 address line undecoded (2(**P**)+2(**P**)+1(**U**))

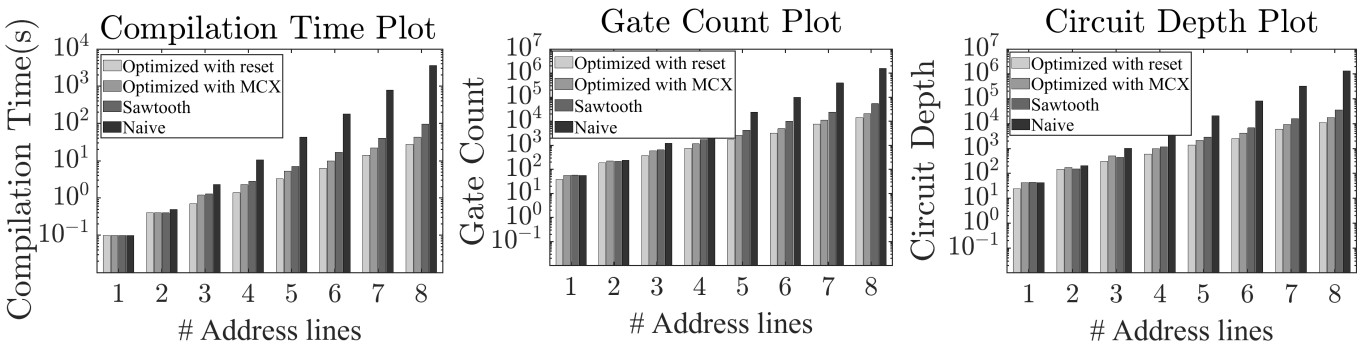

Fig. 7. Comparison of compilation time, circuit depth and gate count QROM circuit for both naive and optimized implementations with varying number of address lines. In the naive implementation, each performance metric value approximately increases $\sim 4x$ for each extra address line. This is due to $4x$ increase in the number of MCX gates for each extra address line.

4) Pre-decode 2 address lines and pre-decode rest 3 address lines (2(**P**)+3(**P**))
5) Pre-decode 4 address lines together, and leave 1 address line undecoded (4(**P**)+1(**U**))
6) Pre-decode all 5 address lines (5(**P**))

We denote the subset of address lines that are pre-decoded by '**P**', and the undecoded subset of address lines by '**U**'. As mentioned previously, various combinations of sets of 'P' and 'U' subsets of addresses are feasible. We obtained the compilation times, gate counts, and circuit depths for all possible combinations for a particular address width. After performing experimentally for different combinations of pre-decoding, it is found that the most optimal results are obtained with $\lceil \frac{n}{2} \rceil$(P)+$\lfloor \frac{n}{2} \rfloor$(P) configuration for n address lines in general. An example is provided in Fig. 6 where 4 address lines are broken down as 2(**P**)+2(**P**). Fig. 7 shows the compilation time, circuit depth, and gate count for the naive and different configurations, including the sawtooth circuit and the two variants of our proposed pre-decoding circuit. On one hand, it shows drastic reduction in all values due to reduction of the control signals required for the MCX gates in the QROM circuit part present inside right box in Fig. 5. On the other hand, the number of controls signals increases in the pre-decoding circuit. These are the MCX gates present in the pre-decoding part of optimized circuit shown in left box in Fig. 5. However, the reduction of control signals of the MCX gates which are present after the pre-decoding circuit are more prominent compared to the increase in pre-decoding circuit. This is because the corresponding drop in gate count after decomposition in the QROM circuit is more than the increase in the gate count after decomposition in the pre-decoding circuit. Thus, the overall gate count reduces, leading to a reduction in circuit depth as well.

Noisy simulations of the optimized QROM circuits are also performed. From the gate count and circuit depth results obtained during compilation, the expected trend is that the fidelity of the pre-decoding circuit should be higher than the fidelity of the sawtooth circuit at iso-address widths because the lower overall circuit depth and gate count will make the predecoded circuit less prone to quantum errors like decoherence, and cross talk in presence of noise. Also, since the circuit depth increases with the number of address lines, the fidelity should also reduce with increasing number of address lines. For the simulations, two different setups were used. In one setup, restricted qubit connectivity is maintained. The connectivity is given according to the coupling map of IBM Mumbai which is one of IBM's quantum computers running on Falcon processor. In the second setup, full qubit connectivity is kept. For both the setups, a noisy Aer simulator from Qiskit is used, with 0 error rate for single qubit gates, and 0.001 error rate for two qubit gates. This two qubit error rate is approximately one tenth of the actual quantum hardware. The error rate is scaled since otherwise the fidelity values are extremely low with the deep QROM circuits and the expected trend is not clearly visible i.e., output becomes random both with and without optimizations. The experiments for both setups are run for 1000 shots. Fig. 8 shows the plots for both setups. The plots follow the expected trends as mentioned above. For restricted connectivity, the fidelity drop in sawtooth circuit (98%-19%) was more than pre-decoding circuit (98%-44%). For full connectivity, the trend was better due to lesser depth (99%-40.8% sawtooth, 99%-73% pre-decoding). For the first scenario of restricted connectivity, error bars have to be used because the fidelity values are volatile and fluctuate a lot. The reason for this fluctuation is due to restricted qubit connectivity leading to an extra step of swap insertion procedure to adhere to the physical qubit mapping, thereby increasing the circuit depth. With this increased circuit depth, there is be more fidelity degradation.

B. Limitations

One should recall that the reduction in circuit depth, gate count and compilation time is at the cost of circuit width i.e., 2^m extra ancilla qubits for every m subset of pre-decoded address lines. We compare the total number of qubits required for both naive implementation and the optimized QROM circuit. The number of qubits required for naive QROM circuit can be calculated as follows: 1 qubit for read control line, n qubits for n address lines, 1 qubit for CNOT control line

and d qubits for d data lines. Therefore, the total number of qubits will be $1 + n + 1 + d = n + d + 2$. In this case, we are keeping d at a constant value of 4. Therefore, the total number of qubits required will be $n + 4 + 2 = n + 6$. Using this as the reference, we calculate the qubit overhead of the optimized QROM circuit for the optimal configuration of $\lceil \frac{n}{2} \rceil$(P)+$\lfloor \frac{n}{2} \rfloor$(P). The results have been plotted in Fig. 9. The general trend observed is that the number of ancilla qubits required in the pre-decoding circuit increases with the number of control signals pre-decoded together ($2^{\frac{n}{2}}$ ancilla qubits for $\frac{n}{2}$ lines pre-decoded together; higher the value of $\frac{n}{2}$, more will be the number of ancilla qubits required). The qubit overhead is therefore more in such cases. There are few minor deviations from this trend. For example, the qubit overhead at 2 qubits is 50%, while that at 3 qubits is 44.44%. This is because the extra qubits needed is same (i.e., 4) while the number of naive qubits needed overall increases from 8 to 9 reducing the % qubit overhead.

From the results obtained, we note as high as around 75X reduction in the circuit depth and gate count, and 85X in the compilation time at the cost of $\approx$ 2.3X extra qubits for 8 address lines. This improvement will further increase as the number of address lines increases. If the qubit overhead is large, one can further break down the optimal configuration into $\frac{n}{4}$(P)+$\frac{n}{4}$(P)+$\frac{n}{4}$(P)+$\frac{n}{4}$(P) to reduce the overhead at the expense of increased circuit depth and gate count.

To get a deeper understanding of the behavior of different configurations of QROM circuits, we performed further analysis of QROM circuits at different configurations of the same number of address lines. Fig. 10 shows the compilation time, gate count, circuit depth, and qubit overhead plots for different configurations of 8 address lines. As mentioned previously, we found that the optimal values are obtained at $\lceil \frac{n}{2} \rceil$(**P**)+$\lfloor \frac{n}{2} \rfloor$(**P**)= 4(**P**)+4(**P**) configuration. This however, comes at the cost of $2^4 + 2^4 = 32$ extra ancilla qubits required in pre-decoding. Another observation is that the values go high when either there are lot of undecoded lines, or when a lot of address lines are pre-decoded together into a single control. Therefore, it is prudent to have a balance of equally pre-decoded address lines and less undecoded lines to keep both qubit overhead and rest of the values as small as possible.

In terms of experiments, it is possible to simulate up to 5 address lines in noisy simulation at higher computational power demand due to increased circuit depth (a limitation). Moreover, as mentioned above, the simulations are performed in reduced noise environment. The noisy simulations also assume full qubit connectivity, which is not the case for real quantum hardware. As a result, while implementing this circuit on real quantum computers, error correction methods like the ones shown in [13]–[15] are required to mitigate the fidelity degradation and get more accurate measurement outputs. Nevertheless introduction of a memory element such as the one proposed herein could revolutionize practical quantum computing as it does not exist currently.

V. Discussion

The proposed optimization reduces the gate count, circuit depth and compilation time at increased circuit width. This approach is still practical as qubit counts are growing over the years with no sight of slowing down. IBM's largest quantum computer has 127 qubits ([16]) with plans to build quantum computers with greater than 1000 qubits by 2023 ([17]). Therefore, sacrificing qubits to improve the fidelity of computation is a viable direction.

One may argue from the experimental results that the QROM circuits are not yet very practical due to fidelity degradation caused by noise in NISQ computers. While this is indeed somewhat true for current NISQ era computers, this issue will eventually die down as improvements are made in quantum computers in general. According to [18], error rates of quantum hardware in the future will reduce significantly, this in turn indicates larger circuits with bigger depths and gate counts and the proposed quantum ROM architecture will run with much higher fidelity, minimizing potential practicality concerns. Along with this, emerging applications may have a need for quantum memories. These will soon increase the demand for proposed quantum memory to be readily available as a building block. Our work targets this anticipated demand in a timely manner.

VI. Conclusion

Quantum memory is an important element that can potentially accelerate applications such as, quantum machine learning. Conventional QROM circuits suffer from high depth, large gate count and higher compilation time for wider address

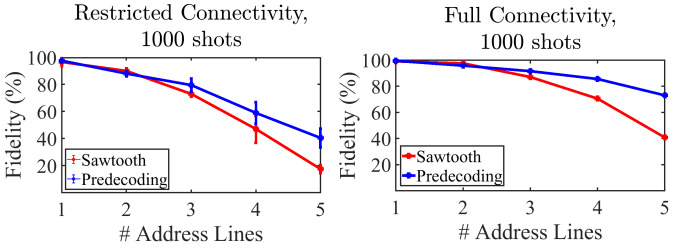

Fig. 8. Experimental simulation results for 1000 shots for optimized QROM circuits. In one scenario, a restricted qubit connectivity was kept. In the second scenario, fully qubit connectivity was maintained.

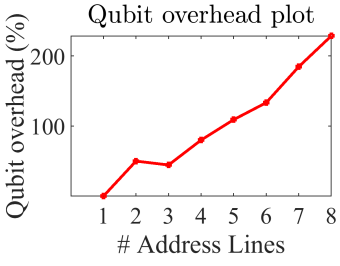

Fig. 9. Qubit overhead in optimized QROM circuit for different address lines for the optimal configuration of $\lceil \frac{n}{2} \rceil$(P)+$\lfloor \frac{n}{2} \rfloor$(P). As the number of controls of MCX gates in the pre-decoding configuration increases, the qubit overhead also increases.

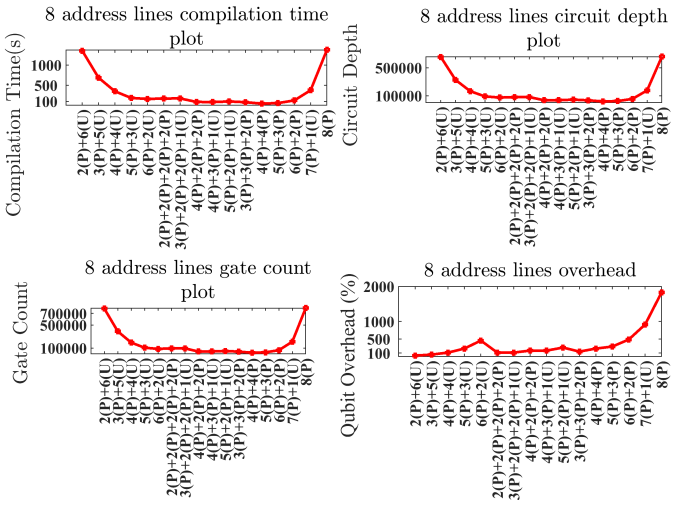

Fig. 10. Compilation time, gate count, circuit depth, and qubit overhead plots for different configurations of 8 address lines.

sizes. We presented a pre-decoding and reset technique to improve the performance of QROM circuits. We noted reduction in circuit depth and gate count as high as 75X and compilation time by 85X at the cost of a maximum of 2.28X qubit overhead. A lesser fidelity drop was also observed in the pre-decoding circuit compared to the sawtooth circuit.

VII. Acknowledgement

The work is supported in parts by NSF (CNS-1722557, CNS-2129675, CCF-2210963, CCF-1718474, OIA-2040667, DGE-1723687, DGE-1821766, and DGE-2113839) and seed grants from Penn State ICDS and Huck Institute of the Life Sciences. We acknowledge the use of IBM Quantum Services for this work. The views expressed are those of the authors, and do not reflect the official policy or position of IBM or the IBM Quantum team.

References

[1] M. Schuld, R. Sweke, and J. J. Meyer, "Effect of data encoding on the expressive power of variational quantum-machine-learning models," *Physical Review A*, vol. 103, no. 3, p. 032430, 2021.

[2] A. Pérez-Salinas, A. Cervera-Lierta, E. Gil-Fuster, and J. I. Latorre, "Data re-uploading for a universal quantum classifier," *Quantum*, vol. 4, p. 226, 2020.

[3] D. K. Park, F. Petruccione, and J.-K. K. Rhee, "Circuit-based quantum random access memory for classical data," *Scientific reports*, vol. 9, no. 1, pp. 1–8, 2019.

[4] R. Babbush, C. Gidney, D. W. Berry, N. Wiebe, J. McClean, A. Paler, A. Fowler, and H. Neven, "Encoding electronic spectra in quantum circuits with linear t complexity," *Physical Review X*, vol. 8, no. 4, p. 041015, 2018.

[5] V. Giovannetti, S. Lloyd, and L. Maccone, "Quantum random access memory," *Physical review letters*, vol. 100, no. 16, p. 160501, 2008.

[6] S. Arunachalam, V. Gheorghiu, T. Jochym-O'Connor, M. Mosca, and P. V. Srinivasan, "On the robustness of bucket brigade quantum ram," *New Journal of Physics*, vol. 17, no. 12, p. 123010, 2015.

[7] V. Giovannetti, S. Lloyd, and L. Maccone, "Architectures for a quantum random access memory," *Physical Review A*, vol. 78, no. 5, p. 052310, 2008.

[8] T. M. Veras, I. C. De Araujo, K. D. Park, and A. J. Dasilva, "Circuit-based quantum random access memory for classical data with continuous amplitudes," *IEEE Transactions on Computers*, 2020.

[9] K. Heshami, D. G. England, P. C. Humphreys, P. J. Bustard, V. M. Acosta, J. Nunn, and B. J. Sussman, "Quantum memories: emerging applications and recent advances," *Journal of modern optics*, vol. 63, no. 20, pp. 2005–2028, 2016.

[10] S. Pirandola, U. L. Andersen, L. Banchi, M. Berta, D. Bunandar, R. Colbeck, D. Englund, T. Gehring, C. Lupo, C. Ottaviani *et al.*, "Advances in quantum cryptography," *Advances in optics and photonics*, vol. 12, no. 4, pp. 1012–1236, 2020.

[11] A. Orieux and E. Diamanti, "Recent advances on integrated quantum communications," *Journal of Optics*, vol. 18, no. 8, p. 083002, 2016.

[12] C. Gidney, "Constructing large controlled nots," 2015. [Online]. Available: https://algassert.com/circuits/2015/06/05/Constructing-Large-Controlled-Nots.html

[13] D. G. Cory, M. Price, W. Maas, E. Knill, R. Laflamme, W. H. Zurek, T. F. Havel, and S. S. Somaroo, "Experimental quantum error correction," *Physical Review Letters*, vol. 81, no. 10, p. 2152, 1998.

[14] J. Chiaverini, D. Leibfried, T. Schaetz, M. D. Barrett, R. Blakestad, J. Britton, W. M. Itano, J. D. Jost, E. Knill, C. Langer *et al.*, "Realization of quantum error correction," *Nature*, vol. 432, no. 7017, pp. 602–605, 2004.

[15] B. M. Terhal, "Quantum error correction for quantum memories," *Reviews of Modern Physics*, vol. 87, no. 2, p. 307, 2015.

[16] M. Sparks, "Ibm creates largest ever superconducting quantum computer," 2021. [Online]. Available: https://www.newscientist.com/article/2297583-ibm-creates-largest-ever-superconducting-quantum-computer/

[17] J. Gambetta, "Ibm's roadmap for scaling quantum technology," 2020. [Online]. Available: https://research.ibm.com/blog/ibm-quantum-roadmap

[18] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.