Quantum Circuit Optimization of Arithmetic Circuits using ZX Calculus

Aravind Joshi, Akshara Kairali, Renju Raju, Adithya Athreya, Reena Monica P

***Abstract**: Quantum computing is an emerging technology in which quantum mechanical properties are suitably utilized to perform certain compute-intensive operations faster than classical computers. Quantum algorithms are designed as a combination of quantum circuits that each require a large number of quantum gates, which is a challenge considering the limited number of qubit resources available in quantum computing systems. Our work proposes a technique to optimise quantum arithmetic algorithms by reducing both hardware resources and the number of qubits, based on ZX calculus. We have utilised ZX calculus rewrite rules for the optimisation of fault-tolerant quantum multiplier circuits, achieving a significant reduction in the number of ancilla bits and T-gates compared to the initially required numbers for fault-tolerance. Our work is the first step in a series of arithmetic circuit optimisations using graphical rewrite tools, paving the way for advancing the optimisation of various complex quantum circuits and establishing the potential for new applications.*

***Keywords**: Circuit Optimization, Quantum Circuit, Quantum Computing, T-count, ZX-calculus.*

I. INTRODUCTION

In classical computing, vast amounts of data cannot be computed simultaneously, making it impractical to model and solve problems of such a data-intensive nature. The large-scale fault-tolerant realisation of quantum computers, on the contrary, is promising due to its ability to compute these amounts of data simultaneously, opening up a wide range of applications, such as prime factorisation of large numbers, molecular structuring to find new drugs, and cybersecurity. Quantum circuit computations are performed using quantum algorithms, which utilise various arithmetic circuits, including subtraction, multiplication, and addition.

Any implementation of these circuits is conditional to keeping the overall resources consumed at an acceptable level, which is achieved by the process of Quantum circuit optimization, usually at a gate level as a collection of gates generally contains all one-bit quantum gates and hence the 2-bit quantum ex-or gate or exclusive-or gate can also be represented as a combination of these gates [1]. Quantum circuit optimisation is a vital topic in the study of quantum circuits, defined as the transformation of given computations into novel circuits using fewer or simpler gates while maintaining their functionality. Since the inception of quantum circuits, which were initially designed using quantum algorithms, research has been conducted on the compilation and optimisation of these circuits. The techniques used target various facets like minimizing the qubit count [2, 3], ancilla bits and garbage bits, reduction in the limitations due to hardware resources [4] and in the number of gates that are more expensive when simulated in error-corrected hardware [5]. For instance, the ancilla constant inputs and outputs are used for computation, but are not helpful since the input or output is garbage bits [6] and therefore the circuit overhead such as the ancilla and garbage outputs need to be minimized. An example of this optimization was the implementation of a reversible quantum integer multiplier that was a garbage output optimized circuit, which achieved an efficiency of sixty to ninety per cent compared to the existing designs [7]. Despite research in this field, the optimizations were costlier to implement in error-corrected hardware error [5] and correction protocols were not investigated in the early work. When assessing optimal circuit constructions, these protocols place constraints on the cost metrics. Many papers have begun to recognise that to implement fault-tolerant circuits, Clifford and T universal gates can be utilised to overcome noise-error-imposed limits. By avoiding uncontrollable errors caused by the interaction of quantum bits, also known as Quantum fault-tolerance, the circuits are more robust and accurate in design. The presented concept underlays the framework for large-scale quantum circuit optimization of fault-tolerant quantum circuits. The techniques like gate substitution, calculation 2 of small (false-) standard forms for distinct families of circuits, and optimization of phase polynomials are the most common approaches for quantum circuit optimization [5]. A novel method of quantum circuit optimisation, formulated from the ZX calculus, is presented here. The circuits are first interpreted as ZX-diagrams, which provide a flexible, low-level language, graphical description of quantum calculations [8].

Manuscript received on 30 December 2023 | Revised Manuscript received on 08 January 2024 | Manuscript Accepted on 15 January 2024 | Manuscript published on 30 January 2024.
**Correspondence Author(s)*

Aravind Joshi, School of Electronics Engineering, Vellore Institute of Technology, Chennai (Tamil Nadu), India.
Akshara Kairali, School of Electronics Engineering, Vellore Institute of Technology, Chennai (Tamil Nadu), India.
Renju Raju, School of Electronics Engineering, Vellore Institute of Technology, Chennai (Tamil Nadu), India.
Adithya Athreya, School of Electronics Engineering, Vellore Institute of Technology, Chennai (Tamil Nadu), India.
Reena Monica P*, School of Electronics Engineering, Vellore Institute of Technology, Chennai (Tamil Nadu), India. Email: reenamonica@vit.ac.in, ORCID ID: https://orcid.org/0000-0002-8202-4027.

Using the rules of ZX-calculus, a strategy for simplification can be devised, demonstrating that the final reduced diagram obtained can be translated back into a quantum circuit. This optimisation method provides a new standard form that is desirable in terms of size and minimises the T-gate count of quantum circuits. For Clifford+T gate circuits, this method enables us to investigate or explore gates that interfere with the Clifford architecture [8]. The above is demonstrated by implementing a fault-tolerant 6-qubit multiplier, followed by the optimisation of this design. The paper is organized as follows. In section 2, the methodology behind the implementation of quantum gates is discussed, followed by a brief overview of ZX calculus.

The section concludes by discussing the implementation of the Multiplier circuit. Section 3 highlights the key steps, execution, and optimisation results. Section 4 discusses the results, and Section 5 provides the conclusion to the work.

II. PRELIMINARIES

Quantum gates act as building blocks to quantum circuits. They operate using qubits, also known as quantum bits, which have quantum information encoded in them. The entanglement property helps the quantum gate avoid information loss, as the qubits are entangled within the gate. The Clifford gate sets are used to implement fault-tolerant circuits. It is already shown that the Gottesman-Knill theorem allows efficient classical simulation of quantum circuits composed of gates in the Clifford set [9]. This makes it possible to simulate circuits with several gates in the order of 1000 [10, 11]. However, this requires a quantum computer that utilises gates outside of the Clifford set to maintain speed. It is commonly noted that most models have a Clifford group with an efficient set of generators, while the non-Clifford group gates require more expensive procedures to implement [12].The most popular non-Clifford gate to use in conjunction with the Clifford set is the T gate. The gates included in the set, along with a few other elementary gates, are described below.

A. Quantum Gates

- *Hadamard Gate:* The Hadamard gate is used to create a superposition of states in a single qubit. It is represented as shown in Fig. 1. A superposition of |0> and |1> states can be visualised as a displacement away from the polar point of a Bloch sphere. The gate can be defined as follows –

$$H = \left({}^{1}\!/_{\sqrt{2}}\right) \times \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (1)$$

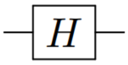

Fig. 1: Hadamard Gate

- *T Gate:* The T gate is used to induce a π/4 phase on a single qubit and can be related to the Phase gate and the fourth root of Pauli-Z. However, it is not a Clifford gate. It is represented as shown in Fig. 2. The gate can be defined as follows –

$$T = \begin{matrix} 1 & 0 \\ 0 & e^{\left(\frac{i\pi}{4}\right)} \end{matrix} \quad (2)$$

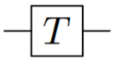

Fig. 2: T Gate

- *Hermitian of T Gate:* The Hermitian of T gate is used to induce a negative π/4 phase on a single qubit and can be related to the S gate as well (as the product of the S gate with itself). The gate can be defined as follows –

$$T^T = \begin{matrix} 1 & 0 \\ 0 & e^{\left(\frac{-i\pi}{4}\right)} \end{matrix} \quad (3)$$

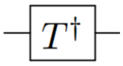

Fig. 3: Hermitian of T Gate

- *Phase Gate:* The Phase gate induces a 90-degree rotation about the Z-axis. It is represented as shown in Fig. 4. The gate can be defined as follows –

$$S = \begin{matrix} 1 & 0 \\ 0 & i \end{matrix} \quad (4)$$

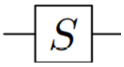

Fig. 4: S Gate

- *Hermitian of Phase Gate:* The Hermitian of the Phase gate induces a 90-degree rotation about the Z-axis. It is represented as shown in Fig. 5. The gate can be defined as follows –

$$S^T = \begin{matrix} 1 & 0 \\ 0 & -i \end{matrix} \quad (5)$$

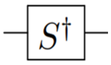

Fig. 5: Hermitian of S Gate

- *NOT Gate:* Similar to a classical NOT gate, the quantum NOT gate also reverses the state of a qubit. It is represented as shown in Fig. 6. The gate can be defined as follows –

$$NOT = \begin{matrix} 0 & 1 \\ 1 & 0 \end{matrix} \quad (6)$$

Fig. 6: NOT Gate

- *CNOT Gate:* The CNOT gate is a 2-qubit gate, where one qubit is a control qubit and the other is a target qubit, upon which the X gate is applied if the control is accurate. It is represented as shown in Fig. 7. The gate can be defined as follows –

$$CNOT = \begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{matrix} \qquad (7)$$

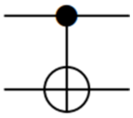

Fig. 7: CNOT Gate

B. ZX-Calculus

Having reviewed the basic quantum gates and their functions, the other concept that requires some background is ZX-Diagrams and their Calculus. ZX-diagrams are essentially a graphical representation of complex matrices of size 2n × 2 m. This notation enables calculations to be performed, which can simplify the effort that would otherwise be required. A ZX-diagram consists of spiders and wires, which constitute the basic building blocks of ZX-Calculus. The thin lines are called wires, where the ones that are present on the left of the dot are inputs, and any wires that are to the right side of the same are outputs [8]. Spiders, also known as generators, are linear maps with the probability of having many inputs or outputs. There are two types of spiders: the Z spider, illustrated as a green dot, and the X as a red dot, where each represents a complementary set of bases. This is illustrated in Fig. 8.

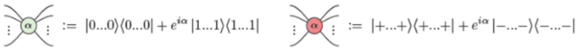
α
$:= |0...0\rangle\langle 0...0| + e^{i\alpha} |1...1\rangle\langle 1...1|$
α
$:= |+...+\rangle\langle +...+| + e^{i\alpha} |-...-\rangle\langle -...-|$

Fig. 8: Z and X Spiders

This linear mapping, built using spiders, is used to model various quantum gates. ZX-Calculus has also been used in simplifying circuits in the past. One popular method combines the ‘sum-of-stabilisers’ method with an automated simplification strategy based on the ZX calculus. They adapted these techniques from the original setting of Clifford circuits with magic state injection to generic ZX-diagrams and show that, by interleaving this” chunked” decomposition with a ZX-calculus-based simplification strategy, stabilizer decompositions can be obtained that are many orders of magnitude smaller than existing approaches [13]. Similarly, the usability of ZX-Calculus for small computations on quantum circuits and states is already discussed along with the Clifford computation and graphical proof of the Gottesman-Knill theorem, and the recent completeness theorems for the ZX-calculus that show that, in principle, all reasoning about quantum computation can be done using ZX-diagrams [14] There are two main differences between ZX diagrams and quantum circuits. ZX diagrams do not need to conform to a rigid topological structure of circuits and can therefore be deformed continuously. The second is that there is a set of rewrite rules for ZX-diagrams, such as the copy rule, Bi-algebra rule, and fusion rules, collectively referred to as the ZX-calculus. An example of simplifying a ZX-diagram is demonstrated below. Let us begin with a ZX diagram featuring three CNOT gates, as shown in Fig. 9.

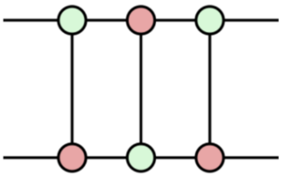

Fig. 9: ZX-Diagram of Circuit with 3 CNOT Gates

We can apply the bi-algebra rule to this.

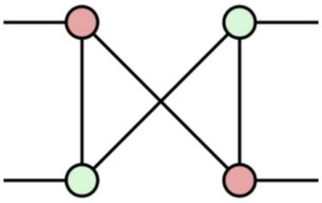

Fig. 10: After the Bi-Algebra Law is Applied

Next, we can pull the gate through the swap as shown in Fig. 11.

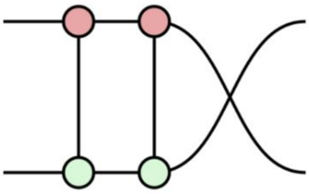

Fig. 11: Circuit After Swap Gate

This is followed by merging the two red nodes and the two green nodes.

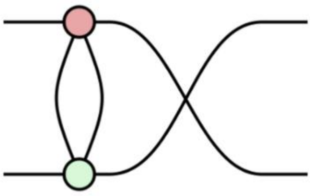

Fig. 12: Merging the Nodes

To this, we can apply the Hopf law as shown in Fig. 13

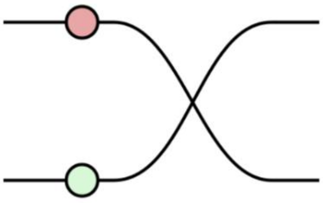

Fig. 13: Circuit after Hopf Law is Applied

Finally, passing this through the swap gate removes the blank spider and gives us the simplified diagram as shown in Fig. 14

Retrieval Number: 100.1/ijitee.B979413020124
DOI: 10.35940/ijitee.B9794.13020124
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

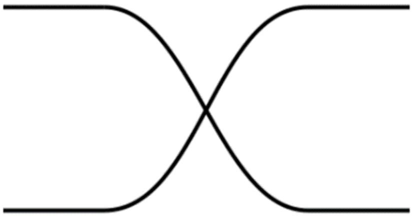

Fig. 14: ZX-Diagram of Simplified Circuit

III. METHOD

The fault-tolerant implementation of quantum arithmetic circuits is gaining significant attention from the research community due to their fundamental applications and the need to compensate for the proclivity of physical quantum computers to introduce noise errors. There are various such implementations, such as Integral multiplication, Integral Divider, and Non-restoring square root. These form a class of fundamental arithmetic circuit implementations. For instance, Thapliyal et al. (2019) present one implementation of the Integral Multiplier. The modules do not produce any garbage outputs and also restore inputs to their starting value; therefore, they are an efficient implementation in terms of Qubit use and T-gates. Other deployments include Galois Squaring and Exponentiation, Bilinear Interpolation and Reversible Square. All of these can be optimized for their T-count or T-depth in their fault-tolerant implementations, as the quantum T gate, which is a non-Clifford gate, is used to obtain a universal gate set

A. Integer Multiplier Circuit

For the implementation of the Quantum Multiplier circuit using the Shift and Add algorithm, conditional Adder circuits are utilised. In a conditional adder circuit, two sets of input signals, A and B, are given, along with the ‘ctrl’ input. Quantum registers are used to store the following qubits. For an n-bit conditional adder, a quantum register for the first input A with n qubits and 2 qubits, a second quantum register for input B with n qubits, and another quantum register with a single qubit for the control input are used. Registers form the basis of a quantum circuit. Quantum operations are forced sequentially on quantum bits to generate the result. The nth qubit in the first Quantum Register is transformed to MSB, and the n qubits in the second quantum register transform to the other bits of the sum. All the other qubits are unchanged at the output. The circuit is as shown in Fig. 15. The conditional Adder circuits are placed in a shifted manner to perform multiplication. For the multiplication of two n-bit numbers, n conditional adders are used. The first stage of the adder can be replaced by an array of Toffoli gates, which helps to reduce the overall T-count in the circuit. The T gate count in a 6-bit conditional adder is 140, whereas the Toffoli gate array has only 42 T gates. There are 25 qubits in this quantum circuit, 6 of which correspond to the first input, 6 again to the second input, and 13 ancilla inputs initialised to zeroes. Each of the 6 qubits in the second input acts as the control input to a corresponding conditional adder. At the end of the computation, 12 ancilla bits contain the 12 product bits, and the final ancilla input is unchanged

B. ZX Optimization

ZX calculus is a graphical language which can be used to create graphical representations of quantum circuits. The calculus works by using Z and X functions, which allows us to modify the quantum circuit model while maintaining the soundness of reasoning. In this way, the properties of circuits, protocols, and entanglement states can be depicted in a visually clear and logically complete manner. ZX optimisation begins with the ZX-diagrams being converted to a graph-like form, where every spider is only of the Z type, which can be achieved using twisting and redefinition. Every spider is associated with some output and no non-zero phases. This is followed by the process of removing as many internal spiders from the graph-like form by pivoting and inverting, as allowed by a set of rules in ZX-calculus. These core rules of ZX-calculus facilitate the easy simulation of the Clifford+T gate set. We can simulate simplified ZX-diagrams that do not have an equivalent quantum circuit, as ZX-diagrams can flexibly portray quantum calculations than any other circuit [7]. The extraction of the circuits is also a direct procedure, where spiders are unfused and replaced with Clifford gates and CZ gates on the inputs and outputs as per requirement. These are not just mere flexible circuits, but contain a rich equational theory, namely the ZX-calculus, and can be deformed arbitrarily. The core parameters of the ZX-calculus provide a thorough theory for Clifford circuits, which can be efficiently simulated classically.

IV. RESULTS

To introduce resistance to noise, in our integer multiplier circuit, we have replaced the Toffoli gate with a Clifford+T gate architecture. The circuit is then converted into a quantum gate using Python commands. The quantum gate is appended to the multiplier code, replacing the Toffoli gate, thereby making the circuit fault-tolerant. The multiplier circuit using the Clifford gate is shown in Figure 15. To perform the ZX calculus, we import the open-source Python library “PyZX” in a Jupyter notebook. Firstly, we change the multiplier code into QASM code and transform it into basic quantum gates. The next step is to identify the gates present in the non-optimised multiplier circuit, followed by converting it into a graph. The graph-like state is a crucial step before simplifying as it allows the optimization to result in a minimum number of internal spiders. The final step involves extracting the optimised circuit and printing the gates that are present in the circuit after optimisation. Ultimately, we have verified the circuit's functionality. To measure the cost-effectiveness of our optimised circuit, we perform a cost analysis that considers the T-gate count and ancilla inputs as primary factors, with the results tabulated in Table 1. The T gate count was reduced from 742 to 488 after optimisation, as shown in Fig. 16.

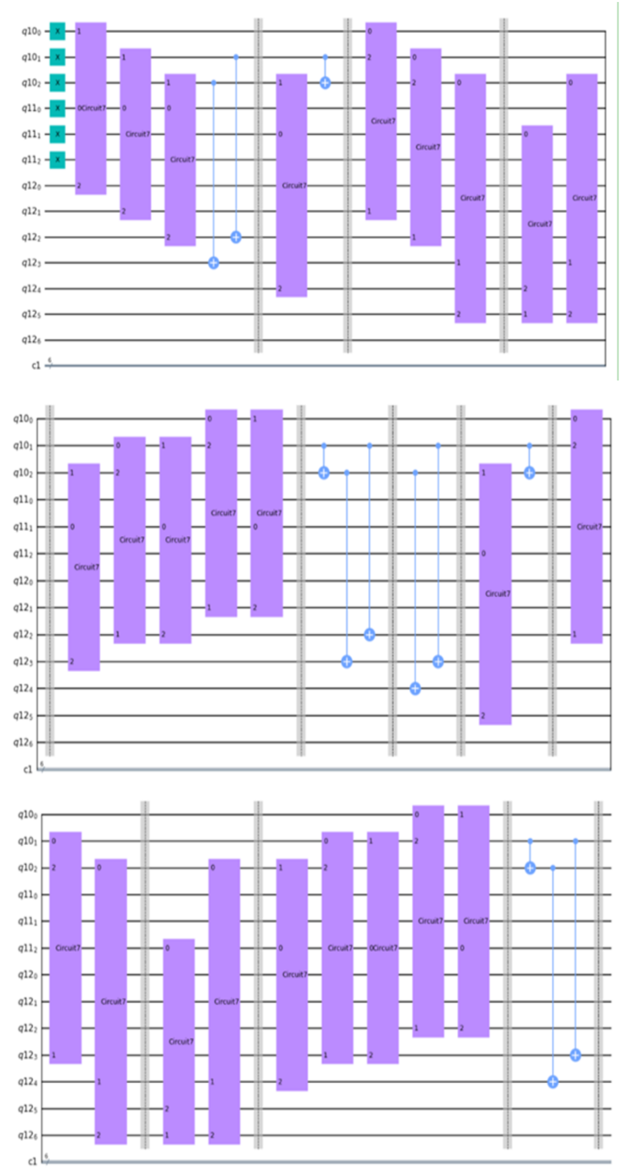

Fig. 15: Integer Multiplier Circuit

Table I: Comparison of Multiplier Circuit Before and after Optimization for 6-Bit Multiplier

Features	Before Optimization	After Optimization
Number Of Qubits	25	25
Number of Gates	1786	2416
T-gate count	742	488
Clifford Gate count	1044	2328

The Clifford gate count increases from 1044 to 2328. Where n is the number of bits for which the adder is designed, in a conditional adder circuit, the T gate count is given by 21n + 14, and the Toffoli gate array uses 7n T-gates. This brings the total T-count to

$$21n^2 - 14 \tag{8}$$

The total ancilla inputs used in the design are given by

$$2n + 1 \tag{9}$$

For a 6-qubit circuit design, the T count is 742, with an ancilla count of 13.

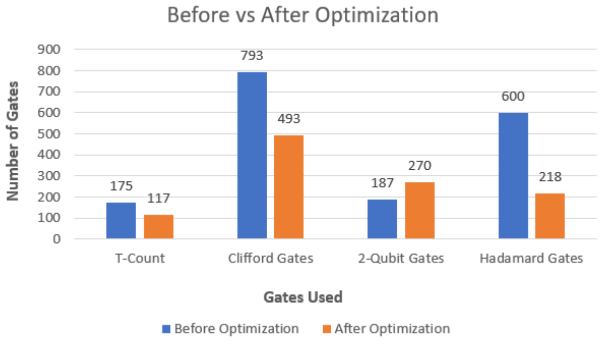

Fig. 16: Optimization Results

V. CONCLUSION

To introduce resistance to noise, we have replaced the Toffoli gate in our integer multiplier circuit with a Clifford+T gate architecture. The circuit is then converted into a quantum gate using Python commands. In this work, a new method for optimising quantum arithmetic circuits using ZX calculus is introduced. In particular, the fault-tolerant implementations of these circuits are considered, and the trade-offs required regarding hardware cost are minimised by using this method. For the fault-tolerant implementation of the 6-qubit multiplier circuit, the optimisation resulted in a 34% reduction in the T-gate count, accompanied by a trade-off in the Clifford gate count. The T-gate count is reduced compared to previous designs, without an increase in the number of qubits, resulting in an overall cost reduction for the hardware implementation of the circuit.

DECALARION STATEMENT

Funding	No, I did not receive.
Conflicts of Interest	No conflicts of interest to the best of our knowledge.
Ethical Approval and Consent to Participate	No, the article does not require ethical approval or consent to participate, as it presents evidence.
Availability of Data and Material/ Data Access Statement	Not relevant.
Authors Contributions	Reena Monica P conceived the idea, proposed, and reviewed the work. Aravind Joshi, Akshara Kairali, Renju Raju and Adithya Athreya carried out the work and documented the initial draft. Adithya Athreya perfected the manuscript based on the reviewers' comments.

REFERENCES

1. Barenco, A., Bennett, C., Cleve, R., Divincenzo, D., Margolus, N., Shor, P., T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary Gates for Quantum Computation," Phys. Rev.. A 52, 3457–3467 (1995) https://doi.org/10.1103/PhysRevA.52.3457
2. Beauregard, S.: Circuit for Shor's algorithm using 2n+three qubits. arXiv preprint quant-ph/0205095 (2002) https://doi.org/10.26421/QIC3.2-8
3. Shende, V.V., Markov, I.L., Bullock, S.S.: Minimal universal two-qubit controlled-not-based circuits. Physical Review A 69(6), 062321 (2004) https://doi.org/10.1103/PhysRevA.69.062321

4. Maslov, D.: Linear depth stabiliser and quantum fourier transformation circuits with no auxiliary qubits in finite-neighbour quantum architectures. Physical Review A 76(5), 052310 (2007) https://doi.org/10.1103/PhysRevA.76.052310
5. Paler, A., Polian, I., Nemoto, K., Devitt, S.J.: Fault-tolerant, high-level quantum circuits: form, compilation and description. Quantum Science and Technology 2(2), 025003 (2017) https://doi.org/10.1088/2058-9565/aa66eb
6. Munoz-Coreas, E., Thapliyal, H.: Quantum circuit design of a t-count optimized integer multiplier. IEEE Transactions on Computers 68(5), 729–739 (2018) https://doi.org/10.1109/TC.2018.2882774
7. Jayashree, H., Thapliyal, H., Arabnia, H.R., Agrawal, V.K.: Ancilla-input and garbage-output optimized design of a reversible quantum integer multiplier. The Journal of Supercomputing 72, 1477–1493 (2016) https://doi.org/10.1007/s11227-016-1676-0
8. Duncan, R., Kissinger, A., Perdrix, S., Van De Wetering, J.: Graph-theoretic simplification of quantum circuits with the ZX-calculus. Quantum 4, 279 (2020) https://doi.org/10.22331/q-2020-06-04-279
9. Wecker, D., Svore, K.M.: Liqui—¿: A software design architecture and domainspecific language for quantum computing. arXiv preprint arXiv:1402.4467 (2014)
10. Gottesman, D.: The Heisenberg representation of quantum computers. arXiv preprint quant-ph/9807006 (1998)
11. Aaronson, S., Gottesman, D.: Improved simulation of stabilizer circuits. Physical Review A 70(5), 052328 (2004) https://doi.org/10.1103/PhysRevA.70.052328
12. Amy, M., Maslov, D., Mosca, M., Roetteler, M.: A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 32(6), 818–830 (2013) https://doi.org/10.1109/TCAD.2013.2244643
13. Kissinger, A., Wetering, J.: Simulating quantum circuits with ZX-calculus reduced stabiliser decompositions. Quantum Science and Technology 7(4), 044001 (2022) https://doi.org/10.1088/2058-9565/ac5d20
14. Wetering, J.: ZX-calculus for the working quantum computer scientist. arXiv 13 preprint arXiv:2012.13966 (2020).

AUTHORS PROFILE

Aravind Joshi holds a Master's in Business Administration from SIBM, Pune, where his focus is on Innovation and Entrepreneurship. He has a bachelor's degree in Electronics and Communication Engineering from the Vellore Institute of Technology, Chennai. His research interests are primarily focused on futuristic technologies, including quantum computing.

Akshara Kairali has a Master's degree in VLSI Design from Vellore Institute of Technology, Chennai. She has a Bachelor's degree in Electronics and Communication Engineering from Peri Institute of Technology, Chennai. She is currently working at Intel Technology India Pvt. Ltd. as an FPGA Silicon Design Engineer. Her research interests include quantum computing and security & privacy in IoT.

Renju Raju has a Master's degree in VLSI Design from Vellore Institute of Technology, Chennai. She has completed a Bachelor of Technology in Electronics and Communication from the Government. Engineering College, Thrissur. She has over 6 years of experience in front-end design and verification of SoCs for high-end automotive, networking, and server applications. She is currently working with Intel Technologies India Pvt. Ltd. as an IP Design Engineer. Her research interests include low-power and security-aware design practices in SoCs.

Adithya Athreya has a Bachelor's degree in Electronics and Communication Engineering from Vellore Institute of Technology, Chennai. He is currently working as an Embedded Software Engineer in the Automotive Industry. His research interests include Quantum Computing and Computer networks.

Reena Monica P holds a Ph.D in Nanoelectronics from Vellore Institute of Technology, Chennai. Her master's degree was in VLSI Design, and she graduated with a degree in Electrical and Electronics Engineering. Her current research interest is in Quantum Computing. Some of the ongoing research includes quantum circuit designs and Optimisation, the Design of a quantum processor integrated into a conventional processor, the creation of a Cryo-CMOS/CNTFET Digital Cell Library for Quantum Computing, Fault-tolerant quantum circuits, Quantum cryptography, and quantum computing for new materials. During her PhD, she worked on the Modelling, Simulation, and Fabrication of Carbon Nanotube Field-Effect Transistors (CNTFETs).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of the Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)/ journal and/or the editor(s). The Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP) and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.