A Novel and Efficient square root Computation Quantum Circuit for Floating-point Standard

Gayathri S S[a], R.Kumar[a,*], Majid Haghparast[b], Samiappan Dhanalakshmi[a]

[a]*Department of Electronics and Communication Engineering*
College of Engineering and Technology, Faculty of Engineering and Technology
SRM Institute of Science and Technology
SRM Nagar, Kattankulathur 603203,
Kanchipuram, Chennai TN, India
[b]*Faculty of Information Technology, University of Jyväskylä, P.O. Box 35, FI-40014*
University of Jyväskylä, Jyväskylä, Finland

Abstract

It is imperative that quantum computing devices perform floating-point arithmetic operations. This paper presents a circuit design for floating-point square root operations designed using classical Babylonian algorithm. The proposed Babylonian square root, is accomplished using Clifford+T operations. This work focuses on realizing the square root circuit by employing the bit Restoring and bit Non-restoring division algorithms as two different approaches. The multiplier of the proposed circuit uses an improved structure of Toom-cook 2.5 multiplier by optimizing the T-gate count of the multiplier. It is determined from the analysis that the proposed square root circuit employing slow-division algorithms results in a T-count reduction of 80.51% and 72.65% over the existing work. The proposed circuit saves a significant number of ancillary qubits, resulting in a qubit cost savings of 61.67 % When compared to the existing work.

Keywords: Quantum arithmetic circuits, T-count, T-depth, Floating-point square root, Babylonian square root, Quantum Computing, Integer division

*Corresponding Author: R.Kumar, SRM Institute of Science and Technology; kumarr@srmist.edu.in

1. Introduction

Quantum computing is one of the most promising emerging and upcoming computing constructs, with applications in cryptography[1, 2, 3, 4], image processing [5, 6, 7] scientific information processing [8, 9, 10, 11], and in finding the solution of system of linear equations [12, 13, 14]. Quantum square root circuit can be used to solve Hamiltonian equations, quadratic congruence, Poisson equations, and to solve trigonometric functions [15]. As the qubits are inherently unstable, physical quantum computing devices are highly susceptible to noise errors. [16, 17, 18, 19, 20]. In the circuit model, fault-tolerant Clifford+T quantum gates or quantum error correction codes can be used to handle noise-intolerant quantum circuits. A spy quantum circuit is required to build circuits that use quantum error correcting codes, which can result in a more complicated overall circuit [21, 7]. The overhead of quantum T gates and number of qubits is accompanied by a noise-tolerant quantum circuit built with fault-tolerant quantum gates [22, 23, 21]. The performance of a fault-tolerant quantum circuit can be analysed using three distinct parameters namely,

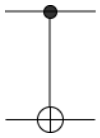
1. Qubits: The total number of qubits including the ancillary qubit employed in the quantum circuit

2. rredT-depth: The count of quantum T-gate layers in the quantum circuit

3. redT-count: The number of T-gates utilized in total to implement the unitary

Because of the increased cost of recognising the T gate, T-count has become an important performance measure for fault tolerant quantum circuit design. Furthermore, existing machines have few qubits, and humongous machines are hard to fathom. As a necessary consequence, the total amount of qubits and T-count required by a quantum circuit is an essential measurable statistic.A critical aspect of gaining computing benefit from a quantum computing machine is reducing the amount of resources required to implement a quantum algorithm. This work aims to reduce the physical cost associated with T-gates and qubits in implementing quantum algorithms by using Clifford+T gates [24, 25]. The list of fault-tolerant Clifford+T logic gates is provided in Table 1. Clifford+T quantum gates are fault-tolerant gates, but quantum CCNOT (Controlled-controlled NOT) or Toffoli gates are excluded.

Table 1: Gates for Clifford +T quantum systems

S.No	Gate	Matrix	Symbol
1	Pauli-x	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$	X or $\bigoplus$
2	Hadamard	$\frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	H
3	CNOT (Controlled-Not)	$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$	
4	T-Gate	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$	T
5	T-Gate Hermitian Transpose or T †	$\begin{bmatrix} 1 & 0 \\ 0 & e^{-i\pi/4} \end{bmatrix}$	T^{-1}
6	Phase Gate	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$	S
7	Phase Gate Hermitian Transpose	$\begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$	S^{-1} or S †

Considering the application of quantum CCNOT gates in quantum circuits, decomposing the quantum Toffoli gate into Clifford +T gate sets has gained widespread interest in the literature [26, 27, 28, 29]. In addition to the many decomposition schemes available to decompose the Toffoli gate [26, 30, 31], the Clifford+T solution is better realized as it has a reduced T- gate count of 4 and a reduced T depth of 3 in Gidney's adder (GA) [32] is shown in Figure 1.

The Clifford+T gate set is a universal set, as the T gate is expensive to fault-tolerantly implement, it's crucial to minimize the T gates usage in a fault-tolerant quantum system. Compared to integer arithmetic [33, 34, 22, 35, 36, 37, 38] the design of quantum circuits for floating-point arithmetic has received less importance in the literature [39, 23, 37]. Dutta et.al [40] proposed a quantum circuit model to realize square root for both fixed point and

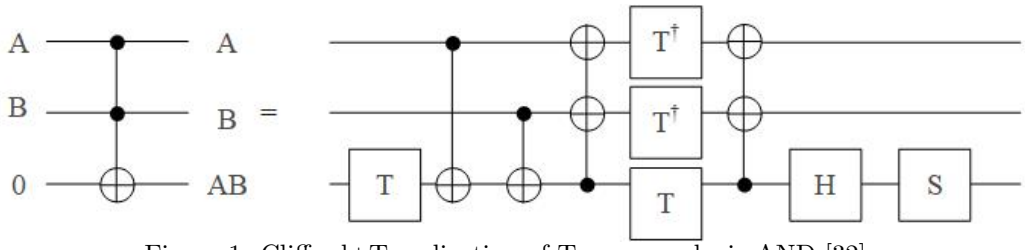


Figure 1: Clifford+T realization of Temporary logic AND [32]

floating-point number using an iterative method that employs two variables. The existing design employs high T-costs because the adder used in the design is the Cuccaro adder (CUA) [41], which has a high T-count. This work focuses on developing a quantum circuit to perform square root operations on single-precision floating-point numbers using the classical Babylonian algorithm. The proposed quantum circuit employs a division circuit in two ways: Restoring division algorithm (RDA) and Non-restoring division algorithm (NRDA). Section 2 describes the preliminary stages of a square root computing circuit using the traditional Babylonian approach. In Section 3, the resource estimation calculation for the proposed quantum circuit model is discussed and compared to previous work. The conclusion of the proposed work is presented in section 4.

2. IEEE- 754 standard Floating-point square root Computation

According to IEEE-754 standard, floating-point numbers can be represented as either single precision or double precision according to the bits size. Figure 2 illustrates a typical representation of a single-precision floating-point number. As complex values are not allowed in IEEE-754 floating-point num-

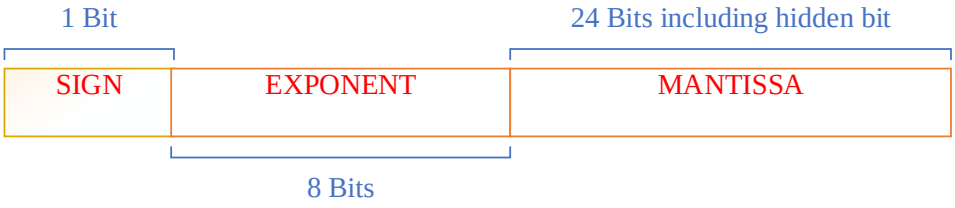


Figure 2: Single-precision IEEE 754 Floating-point standard

bers, the square root algorithm is exclusively designed for positive floating-point numbers. If the exponent is even, the exponent is decreased to its half

to perform the square root operation. If the exponent is odd, exponent must be increased by one and accordingly mantissa is aligned. The square root is calculated on the fraction and the exponent is adjusted to match the bias. The algorithm to solve square root problem is shown in Algorithm 1.

Algorithm 1: Floating-point square root algorithm

Input: Input $S_a E_a M_a$
Result: square root of Input $S_r E_r M_r$
$S_r = S_a$
if $E_a = odd$ **then**
    $I_{Er1} = E_a + 1$
    $I_{Er} = I_{Er1}/2$
    Align Mantissa
else
    $I_{Er} = E_a/2$
$E_r = I_{Er} + Bias$
$Mr = \sqrt{M_a}$

The quantum circuit model performs floating-point square root operation as directed by the Algorithm 1 is shown in Figure 3.

2.1. Babylonian square root Algorithm

One of the oldest ways for determining the square root of a number is the Babylonian square root algorithm. Despite its age, the technique is one of the most efficient for computing square roots, and it is used by many modern computing system [42]. The Babylonian procedure for computing the square root of an integer assumes an initial guess of a positive number that is much greater than the square root of the input. The iteration begins after assuming the initial guess, and the hardware of each iteration includes two integer division circuits, one adder circuit, and one multiplier circuit. To ensure the accuracy of the result, the iteration is repeated until a very low relative error is obtained between the current output and the previous output state. The procedure to perform square- root computation using Babylonian method is illustrated in Algorithm 2. The top level overview quantum circuit for computing integer square root using Babylonian algorithm is shown in Figure 4. Each iteration involves the use of two quantum division and three quantum addition circuits. The quantum addition circuit can be used

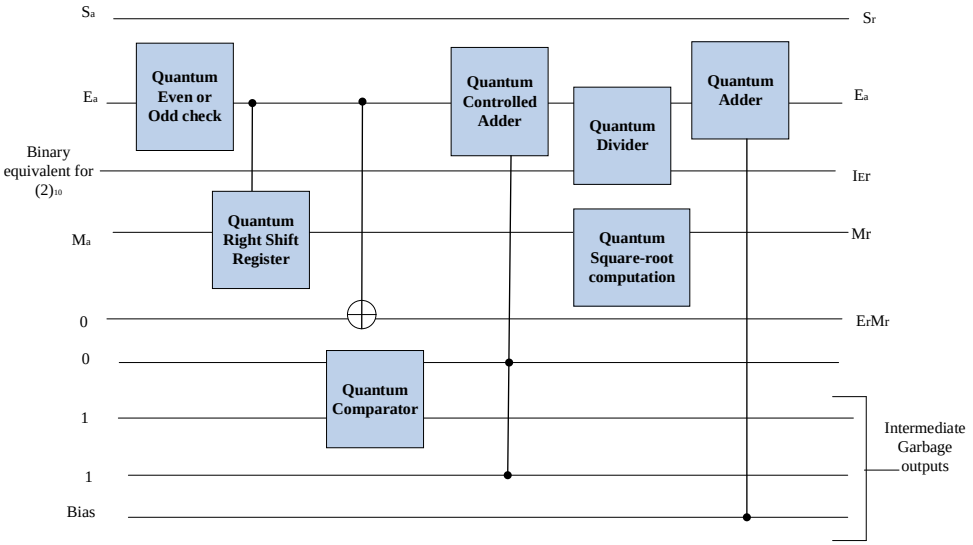


Figure 3: Top level overview of quantum floating-point square root circuit

Algorithm 2: Babylonian algorithm to compute square root

Input: Input fractional number M_x
Result: square root of the number M_r
Assign $G_1 = b > \sqrt{M_x}$
for $i = 1toN$
$G_{i+1} = \frac{1}{2} * (G_i + \frac{M_x}{G_i})$
$\epsilon = G_{i+1} - G_i$
if $\epsilon = 00000000000000000000000011$
$M_r = G_{i+1}$
else
$i = i + 1$
end
end

to perform subtraction and comparisons between the two input qubits. In each iteration, the first quantum division block computes the quotient of M_x/G_i, the following quantum adder circuit computes $G_i + M_x/G_i$, and the second division circuit reduces the adder circuit's result to half its value. The quantum subtractor circuit computes the difference in output between the current and previous output. The quantum comparator circuit aids in the

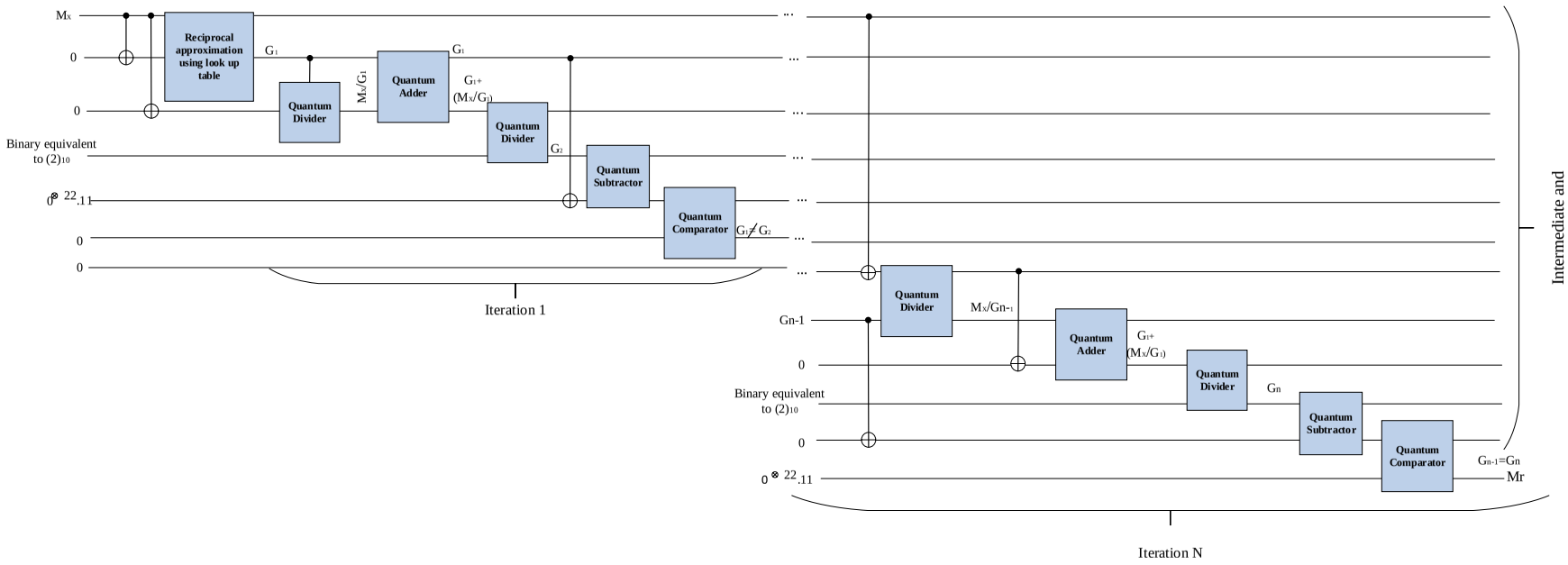


Figure 4: High level overview of integer square root computation using Babylonian algorithm

termination of iterations until the desired accuracy is achieved.

2.2. Proposed Quantum circuit to identify even and odd exponent

To calculate the resultant exponent of square root computation circuit, an even or odd identifying circuit is required. The algorithm for determining the exponent is even or odd is shown in the Algorithm 3. The output R_x is generated by an array of CCNOT gates, by performing logical And operation on the input exponent qubit and constant qubits $|00000001\rangle$. If the output R_x generated is $|00000001\rangle$ it indicates the exponent is odd else the exponent input is concluded as an even exponent. Figure 5 depicts the quantum circuit used to determine whether an exponent is even or odd. Table 2 summarizes the resource use of the proposed even or odd identifying quantum circuit that is built for eight qubit input . The uncomputation circuit of the temporary And gate does not contain a T-gate, hence the restoration circuit does not serve to the count of excess T gates in the proposed circuit [32].

Algorithm 3: Algorithm to Identify Even or Odd Exponent

Input: Input E_x
Result: E_x is even or odd
Assign $X = 0 to 7$
Assign $Y_X = 00000001$
$R_X = Y_X \& 00000001$
$X_X = \neg(R_X \bigoplus 00000001)$
$Equal = E_X \& E_{X-1} \& E_{X-2} \& ... \& E_0$
if $Equal = 1$
$E_X => odd$
else
$E_X => even$
end

Table 2: Estimation of resources in the proposed Odd or Even number identification circuit

Design	T-count	T-depth	Ancilla
Proposed Circuit	52	26	14

2.3. T-count optimized Toom-Cook 2.5 multiplier

Many researchers have previously investigated quantum multiplication using a variety of algorithms with a time complexity of $O(n^2)$ [22, 23, 43, 44].

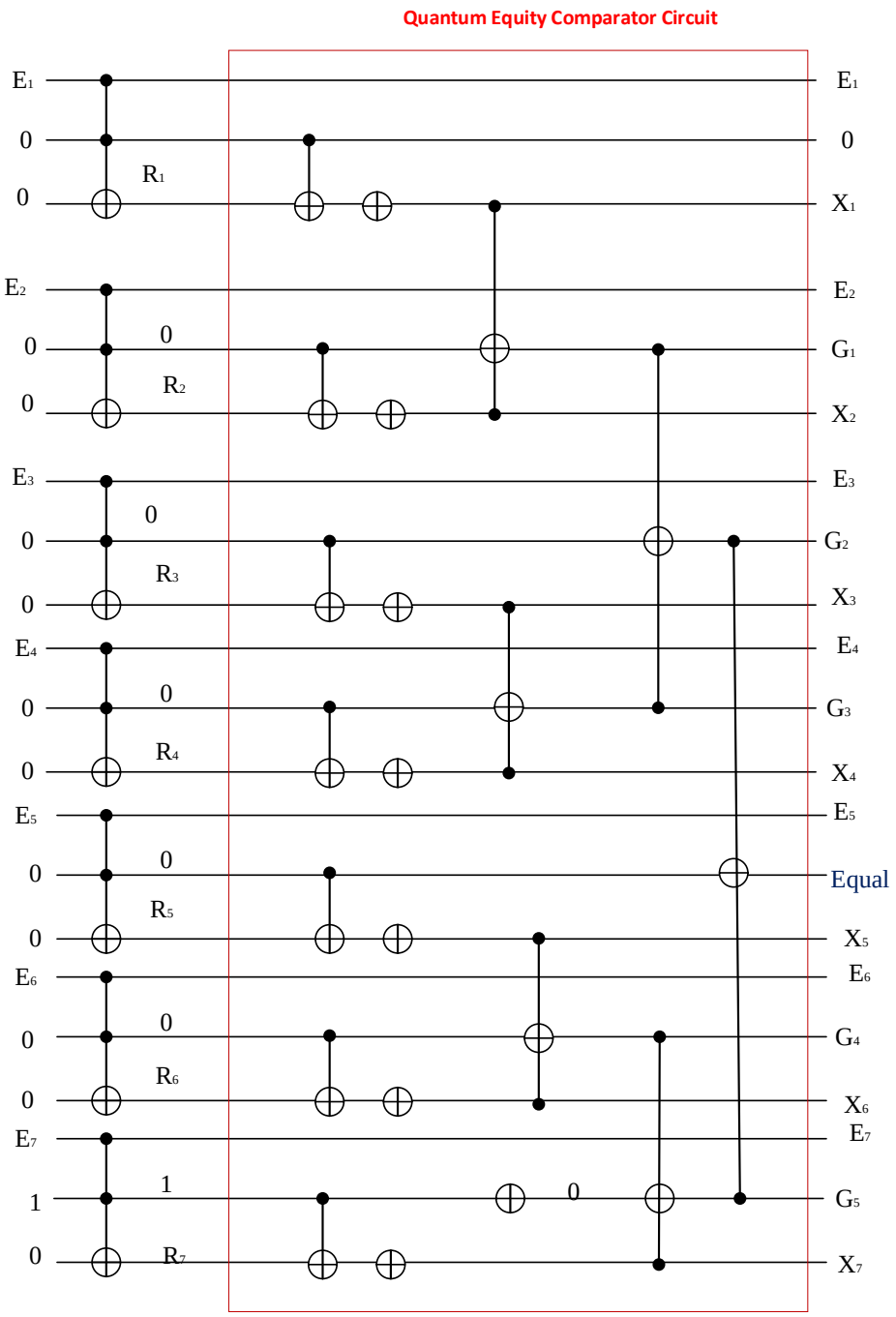


Figure 5: Quantum circuit to find even and odd exponent qubits

Multiplication that has a less time complexity is Toom-cook multiplier that reports a complexity of $O(n^{1.465})$. The procedure to perform multiplication of two integers using Toom-cook 2.5 algorithm is shown in Algorithm 4. The final product computation requires only four multiplications and each multiplications is performed between bit-width of $\frac{l}{2}$ and $\frac{l}{3}$ where l is the length of the input bit sequence. This technique must be performed recursively so that we have smaller problems of size $\frac{l}{6}$ after the second stage. The existing Toom-cook multiplier [40] quantum circuit model uses CUA, whose T-count is $14n - 7$, [41]. Here in this paper,a Toom-cook multiplier is implemented utilising GA, which reported a lower T-gate count of $4n$ for n qubit addition [32].

Algorithm 4: Toom-cook 2.5 multiplication algorithm

Input: Multiplier M and Multiplicand N
Result: Product P
Decompose M and N into two or three equal parts
Assign k=2.5
Assign $x = max\{\lceil \frac{\lceil log_2 M \rceil}{K} \rceil, \lceil \frac{\lceil log_2 M \rceil}{K} \rceil\}$
Assign $N = N_2 2^{2x} + N_1 {}^{2x} + N_0$
Assign $M = M_1 2^{x} + M_0$
Assign $A = M_0 N_0$
Assign $B = (M_0 + M_1)(N_0 + N_1 + N_2)$
Assign $C = (M_0 - M_1)(N_0 - N_1 + N_2)$
Assign $D = M_1 N_2$
Assign $E = D$
Assign $F = A + \frac{1}{2}B + \frac{1}{2}C$
Assign $G = \frac{1}{2}B - \frac{1}{2}C - D$
Assign $H = A$
P = $E2^{3x} + F2^{2x} + G2^{2x} + H$
end

3. Estimation of the resource requirements for the proposed quantum floating-point Babylonian square root unit

The number of qubits and T gates for each stage of the floating-point square root circuit are used to determine resource efficiency of the proposed

quantum circuit. Total T-count and qubits are calculated by adding the T-count and qubit usage at each level of the quantum circuit. The proposed floating-point square root unit design uses the same multiplication circuit for both the approaches, hence the resource usage of the proposed Toom-cook multiplier is evaluated.

3.1. Resource estimation of Modified Quantum Toom-cook 2.5 multiplier

Toom-cook multiplier is realized using $4l$ multiplications and l bit addition for the computation of product of two l bit numbers. To calculate the intermediate outputs A,B,C, and D from Algorithm 4 four recursive calls has to be made on Toom-cook multiplier that computes the product of bits each of length $\frac{l}{2}$ and $\frac{l}{3}$ respectively. To compute the intermediate outputs four adders of size $\frac{l}{2}$ and six adders of size $\frac{l}{3}$ are required. Finally, to compute A,B,C, and D four $5\frac{l}{6}$ adders are required. Based on the recursive analysis, the Toffoli gate count to implement the Toom-cook multiplier is shown in the following Equations 1,2,3,4,5.

$$Toffoli_{countl} = 16Toffoli_{\frac{l}{6}} + 40Adder_{\frac{l}{6}} + 22Adder_{\frac{l}{6}} + 4Adder_{\frac{l}{2}} + 4Adder_{\frac{5l}{6}} \tag{1}$$

The base case multiplication of two one bit numbers can be done by a Toffoli gate and hence the value of $Toffoli_{\frac{l}{6}}$ is equated to one. There are totally $log_6 n$ terms in each summation. On evaluating the summation using Geometric progression and doubling the cost that is consumed in the uncomputing section the $Toffoli_{count}$ is evaluated as,

$$\begin{aligned} Toffoli_{countl} = 16^{\log_6 l} Toffoli_{count1} &+ 40(Adder_{\frac{l}{6}} + 16Adder_{\frac{l}{36}} + ...) \\ &+ 22(Adder_{\frac{l}{3}} + 16Adder_{\frac{l}{18}} + ...) \\ &+ 4(Adder_{\frac{l}{2}} + 16Adder_{\frac{l}{12}} + ...) \\ &+ 4(Adder_{\frac{5l}{6}} + 16Adder_{\frac{5l}{36}} + ...) \end{aligned} \tag{2}$$

$$Toffoli_{countl} = 2(16^{\log_6 l} + 23.2l[((\frac{16}{6})log_6 l - 1] \tag{3}$$

$$Toffoli_{countl} \leq 49l^{\log_6 16} \tag{4}$$

The T-count of a Toffoli gate that performs addition is 4 in the computation section and T-count of the uncomputation section is zero. Thus, the T-count

and T-depth of the entire multiplication circuit is calculated as shown in Equations, 5, 6,

$$T_{countl} \leq 196l^{\log_6 16} \tag{5}$$

$$T_{depthl} \leq 98l^{\log_6 16} \tag{6}$$

Along with the input qubits, the auxiliary qubits are included in the total number of qubits. The number of Toffoli counts corresponds to the number of ancilla qubits in the proposed multiplier, as shown in Equations 6 and 7.

$$Qubits \leq 49l^{\log_6 16} + l \tag{7}$$

3.2. Resource estimation of RDA

The resource usage estimation of the quantum circuits model using RDA is discussed in this section. The algorithm to perform division using RDA is shown in Algorithm 5. The quantum circuit model to construct a two qubit restoring division is shown in Figure 6 [22]. The quantum subtractor and quantum controlled addition circuit are two crucial components of an RDA division circuit. The value of $Z = A - Ay$ is computed using the subtraction

Algorithm 5: Algorithm to perform RDA [22]

Input: Two input numbers A_x and A_y
Result: Quotient Q, Remainder R_x
Assign $Q = A_x$
Initialize $R = 0_{x-1}0_{x-2}...0_10_0$
for $i = 1tox - 1$
$Z = R_{x-1-i}R_{x-2-i}...R_1R_0Q_{x-1}...Q_{x-i}$
$Z = Z - Ay$
if $Y < 0$
$Z = Z + A_y$
end
$R_{x-i} = Not(R_{x-1-i})$
end
$Q = Q - A_y$
if $Q < 0$
$Q = Q + A_y$
end
$R_0 = Not(Q_{x-1})$

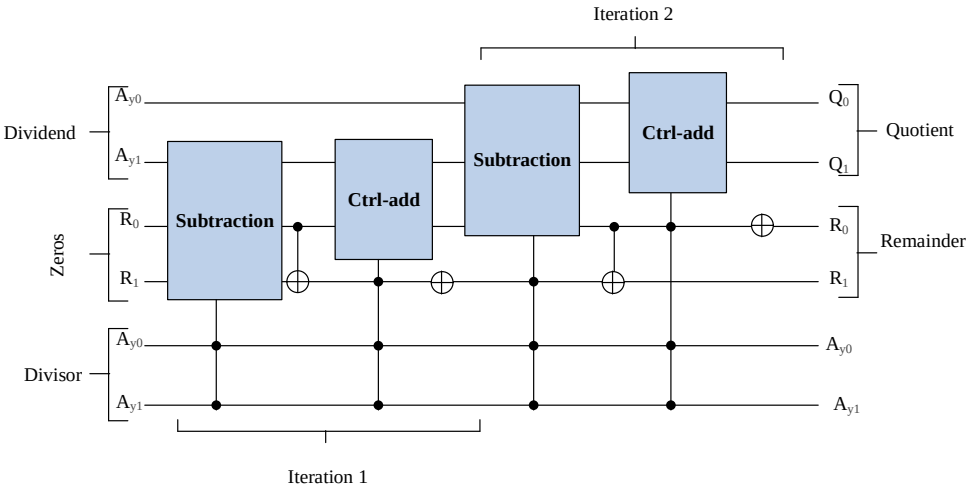


Figure 6: Two qubit RDA circuit

circuit. The value of Q is computed using a quantum controlled addition circuit. By calculating the inverse of the quotient output, the Pauli-X gate generates the remainder of the divider inputs. This method is performed n times for n input qubits, and the quotient and remainder outputs are stored in registers Q and R, respectively, after the nth iteration. The T-count and qubit of the quantum RDA circuit for a single iteration(SI) for n input qubits is calculated as shown in Equation 8,9, 10 and 11 respectively,

$$Resource - count_{SI} = Resource - count_{Subtraction} + Resource - count_{ctrl-add} \tag{8}$$

$$T - count_{SI} = 4n + 18n \tag{9}$$

$$T - depth_{SI} = 2n + 8n \tag{10}$$

$$Qubits_{SI} = 2n_{Subtraction} + 4n_{Ctrl-add} + 2n_{ancilla} \tag{11}$$

3.3. Resource estimation of NRDA

The NRDA is less complex than the RDA because quotient restoration is not included. The algorithm to perform division using NRDA is shown in Algorithm 6. The quantum circuit model to generate quotient and remainder output using NRDA is shown in Figure 7. The quantum subtractor circuit is responsible for computing Q, the Adder-Subtractor circuit computes the

Algorithm 6: Algorithm to perform NRDA [22]

Input: Two input numbers A_x and A_y
Output: Quotient Q, Remainder R_r
Initialize $R = 0_{x-1}0_{x-2}...0_10_0$
Initialize $Q = 0_{x-1}0_{x-2}...0_10_0A_{Xn-1}A_{Xn-2}...A_{X1}A_{X0}$
$Q = A - M_y$
for $i = 1tox - 1$
$Q_{x-i} = Not(Q_{x-i})$
$Y = Q_{x-1-i}...Q_0$
if $Q_{x-i} = 0$
$Z = Z + A_y$
else $Z = Z - A_y$
end
end
If $R < 0$
$R = R + A_y$
end
$Q_0 = Not(Q_0)$

output Z based on the value of Q. The remainder output is calculated using the final adder-subtractor circuit.

The T-count and qubit estimation of the NRDA circuit for single iteration is shown in Equations 12, 13, 14, and 15 respectively,

$$\begin{aligned} Resource - count_{SI} = Resource - count_{Subtraction} + \\ Resource - count_{Adder-Subtractor} \end{aligned} \tag{12}$$

$$T - count_{SI} = 4n + 4n \tag{13}$$

$$T - depth_{SI} = 2n + 2n \tag{14}$$

$$Qubit_{SI} = 4n + 1 + 2n_{ancilla} \tag{15}$$

3.4. Resource estimation of proposed Babylonian square root algorithm

Every iteration of the Babylonian square root algorithm uses two division circuits, three adder circuits, a multiplier circuit, and a quantum comparator circuit. The number of iterations needed to compute the square root of a given number using the Babylonian square root algorithm depends on the

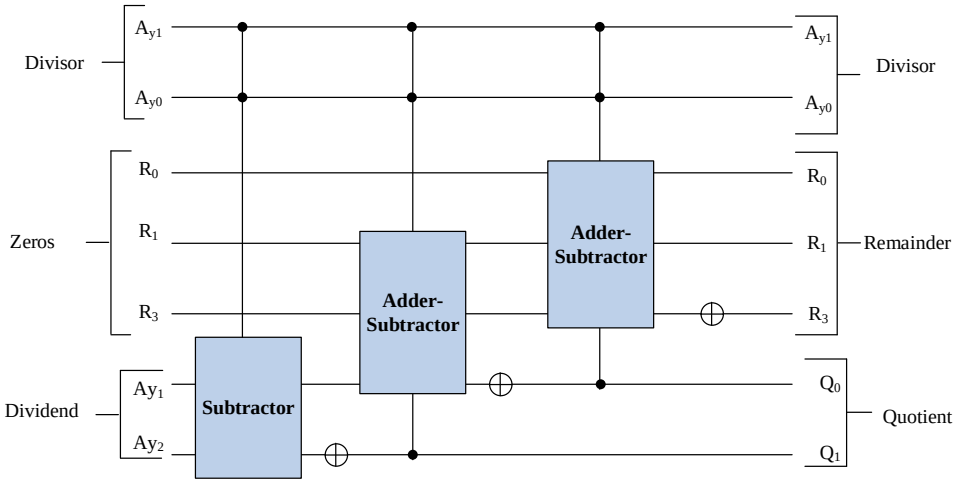


Figure 7: Two qubit NRDA circuit

input number and the initial guess. Typically, the resource-count of a single iteration is used to calculate the entire resource estimation of a square root circuit, and this can be extended to any number of qubits and is shown in Equation 16,

$$\begin{aligned} Resource - count_{SI} = Resource - count_{2*Divider+3*Adder} + \\ Resource - count_{Multiplier+Comparator} \end{aligned} \tag{16}$$

The resource estimation of the proposed Babylonian square root algorithm using RDA and NRDA algorithm for a single iteration is shown in Table 3, The proposed single precision Babylonian floating-point quantum square root circuit is compared with the existing floating-point square root quantum circuit design [37]. An even or odd identification circuit, two adders for exponent qubits, a divider to calculate exponent of the input to its half, and a square root computation circuit on mantissa are used in the resource utilisation of square root computation on a floating-point number as shown in Equation 17,

$$\begin{aligned} Resource - Count = Resource - count_{odd/evencircuit+Divider+ShiftRegister} + \\ Resource - count_{2*Adders} + \\ Resource - count_{squarerootcircuit_{SI}} \end{aligned} \tag{17}$$

The Table 4 shows the T-count, T-depth, and ancilla estimation of the proposed Babylonian square root algorithm to compute the square root of a

Table 3: Resource estimation of Proposed Babylonian square root algorithm on the mantissa part of the single precision floating-point number for single iteration using slow division algorithms

Design	T-count	T-depth	Ancilla
NRDA[23]	4416	2208	1104
RDA[23]	11638	5520	1104
Adder[32]	96	48	24
Modified Toom-cook multiplier	26068	13034	6517
Comparator[23]	96	48	24
Proposed Babylonian square root(NRDA)	35284	17642	8821
Proposed Babylonian square root(RDA)	49728	24266	8821

single precision floating-point input.

In Table 5, the proposed quantum circuit to determine the square root of a floating-point input using the Babylonian square root method is compared to the existing work [37]. When employing the NRDA and RDA algorithms, the proposed quantum square root computing circuit for IEEE-754 standard input saves a T-count of 80.51% and 72.65%, respectively. When compared to the existing circuit model [37], the proposed circuit demonstrates a 61.67% gain in conserving qubits.

4. Conclusion

An efficacious quantum circuit realization with less resource necessity is a major challenge for solving real-time problems in scientific computing. A fault-tolerant floating-point square root architecture for 32 qubits width is proposed in this paper. The presented work outperforms state-of-the-art result in a variety of performance metrics, including the number of T-gates used and the number of qubits. The reduction of ancillary quantum bits without increasing the size and complexity of the circuits would be an intriguing future challenge. The proposed work is anticipated to get utilized in quantum algorithms where reduction T-Count and T-depth are of primary concern.

Table 4: Resource estimation of Proposed Babylonian square root algorithm on 32 Qubit floating-point number for single iteration using RDA and NRDA

Design	**T-count**	**T-depth**	**Ancilla**
Odd or Even Check Circuit(8 Qubits)	52	26	14
NRDA(8 Qubits)	448	224	112
RDA(8 Qubits)	1232	560	112
Shift Register[45]	Nil	Nil	24
Adders(8 Qubits)	32	16	8
Mantissa square root circuit(NRDA)	35284	17642	8821
Mantissa square root circuit(RDA)	49728	24266	8821
Proposed Floating-point square root(NRDA)	35848	17642	8987
Proposed Floating-point square root(RDA)	50292	24884	8987

Table 5: Comparison of Proposed Babylonian square root algorithm with the existing circuit

Design	**T-count**	**T-depth**	**Qubits**
Floating-point square root circuit[37]	183946	NA	23535
Proposed Floating-point square root(NRDA)	35848	17642	9019
Proposed Floating-point square root(RDA)	50292	24884	9019

Funding

Not applicable

Acknowledgement

Not applicable

Data Availability Statement

The data used for this work are available from the corresponding author upon reasonable request.

Conflicts of interest

The authors declare no conflicts of interest

References

[1] A. P. Bhatt, A. Sharma, Quantum cryptography for internet of things security, Journal of Electronic Science and Technology 17 (2019) 213–220.

[2] R. J. Hughes, D. M. Alde, P. Dyer, G. G. Luther, G. L. Morgan, M. Schauer, Quantum cryptography, Contemporary Physics 36 (1995) 149–163.

[3] D. M. Nguyen, S. Kim, Quantum key distribution protocol based on modified generalization of deutsch-jozsa algorithm in d-level quantum system, International Journal of Theoretical Physics 58 (2019) 71–82.

[4] C. H. Bennett, Logical reversibility of computation, IBM journal of Research and Development 17 (1973) 525–532.

[5] F. Yan, A. M. Iliyasu, P. Q. Le, Quantum image processing: a review of advances in its security technologies, International Journal of Quantum Information 15 (2017) 1730001.

[6] Y. Cai, X. Lu, N. Jiang, A survey on quantum image processing, Chinese Journal of Electronics 27 (2018) 718–727.

[7] H. Thapliyal, E. Muñoz-Coreas, Design of quantum computing circuits, IT Professional 21 (2019) 22–26.

[8] H.-S. Li, P. Fan, H. Xia, H. Peng, G.-L. Long, Efficient quantum arithmetic operation circuits for quantum image processing, SCIENCE CHINA Physics, Mechanics & Astronomy 63 (2020) 1–13.

[9] C. Monroe, Quantum information processing with atoms and photons, Nature 416 (2002) 238–246.

[10] F. Flamini, N. Spagnolo, F. Sciarrino, Photonic quantum information processing: a review, Reports on Progress in Physics 82 (2018) 016001.

[11] R. Babbush, D. W. Berry, I. D. Kivlichan, A. Y. Wei, P. J. Love, A. Aspuru-Guzik, Exponentially more precise quantum simulation of fermions in second quantization, New Journal of Physics 18 (2016) 033032.

[12] M. Reiher, N. Wiebe, K. M. Svore, D. Wecker, M. Troyer, Elucidating reaction mechanisms on quantum computers, Proceedings of the National Academy of Sciences 114 (2017) 7555–7560.

[13] M. Zamboni, M. Graziano, P. D. G. Turvani, L. Raggi, Arithmetic circuits for quantum computing: a software library (2020).

[14] L. Sousa, Nonconventional computer arithmetic circuits, systems and applications, IEEE Circuits and Systems Magazine 21 (2021) 6–40.

[15] M. K. Bhaskar, S. Hadfield, A. Papageorgiou, I. Petras, Quantum algorithms and circuits for scientific computing, arXiv preprint arXiv:1511.08253 (2015).

[16] S. Bravyi, A. Kitaev, Universal quantum computation with ideal clifford gates and noisy ancillas, Physical Review A 71 (2005) 022316.

[17] V. Belavkin, Quantum filtering and control in example: Unstable qubits, in: AIP Conference Proceedings, volume 1110, American Institute of Physics, 2009, pp. 57–62.

[18] M. Amy, D. Maslov, M. Mosca, Polynomial-time t-depth optimization of clifford+ t circuits via matroid partitioning, IEEE Transactions on

Computer-Aided Design of Integrated Circuits and Systems 33 (2014) 1476–1489.

[19] W.-J. Huang, W.-C. Chien, C.-H. Cho, C.-C. Huang, T.-W. Huang, S. G. Tan, C. Cao, B. Zeng, C.-R. Chang, Phase analysis on the error scaling of entangled qubits in a 53-qubit system, Scientific reports 11 (2021) 1–9.

[20] A. Paler, I. Polian, K. Nemoto, S. J. Devitt, Fault-tolerant, high-level quantum circuits: form, compilation and description, Quantum Science and Technology 2 (2017) 025003.

[21] H. Thapliyal, T. Varun, E. Munoz-Coreas, Quantum circuit design of integer division optimizing ancillary qubits and t-count, arXiv preprint arXiv:1609.01241 (2016).

[22] S. Gayathri, R. Kumar, S. Dhanalakshmi, B. K. Kaushik, M. Haghparast, T-count optimized wallace tree integer multiplier for quantum computing, International Journal of Theoretical Physics (2021) 1–13.

[23] S. Gayathri, R. Kumar, S. Dhanalakshmi, G. Dooly, D. B. Duraibabu, T-count optimized quantum circuit designs for single-precision floating-point division, Electronics 10 (2021) 703.

[24] T. Häner, S. Jaques, M. Naehrig, M. Roetteler, M. Soeken, Improved quantum circuits for elliptic curve discrete logarithms, in: International Conference on Post-Quantum Cryptography, Springer, 2020, pp. 425–444.

[25] H. Thapliyal, Mapping of subtractor and adder-subtractor circuits on reversible quantum gates, in: Transactions on Computational Science XXVII, Springer, 2016, pp. 10–34.

[26] L. Biswal, D. Bhattacharjee, A. Chattopadhyay, H. Rahaman, Techniques for fault-tolerant decomposition of a multicontrolled toffoli gate, Physical Review A 100 (2019) 062326.

[27] M. Saeedi, M. Pedram, Linear-depth quantum circuits for n-qubit toffoli gates with no ancilla, Physical Review A 87 (2013) 062318.

[28] C. Jones, Low-overhead constructions for the fault-tolerant toffoli gate, Physical Review A 87 (2013) 022328.

[29] M. Mosca, P. Mukhopadhyay, A polynomial time and space heuristic algorithm for t-count, Quantum Science and Technology 7 (2021) 015003.

[30] P. Selinger, Quantum circuits of t-depth one, Physical Review A 87 (2013) 042302.

[31] M. Amy, D. Maslov, M. Mosca, M. Roetteler, 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 (2013) 818–830.

[32] C. Gidney, Halving the cost of quantum addition, Quantum 2 (2018) 74.

[33] E. Muñoz-Coreas, H. Thapliyal, Quantum circuit design of a t-count optimized integer multiplier, IEEE Transactions on Computers 68 (2018) 729–739.

[34] H. Thapliyal, E. Munoz-Coreas, T. Varun, T. Humble, Quantum circuit designs of integer division optimizing t-count and t-depth, IEEE Transactions on Emerging Topics in Computing (2019).

[35] E. Munoz-Coreas, H. Thapliyal, T-count and qubit optimized quantum circuit design of the non-restoring square root algorithm, ACM Journal on Emerging Technologies in Computing Systems (JETC) 14 (2018) 1–15.

[36] A. Khosropour, H. Aghababa, B. Forouzandeh, Quantum division circuit based on restoring division algorithm, in: 2011 Eighth International Conference on Information Technology: New Generations, IEEE, 2011, pp. 1037–1040.

[37] S. Dutta, Y. Tavva, D. Bhattacharjee, A. Chattopadhyay, Efficient quantum circuits for square-root and inverse square-root, in: 2020 33rd International Conference on VLSI Design and 2020 19th International Conference on Embedded Systems (VLSID), IEEE, 2020, pp. 55–60.

[38] H. Thapliyal, E. Muñoz-Coreas, V. Khalus, Quantum circuit designs of carry lookahead adder optimized for t-count t-depth and qubits, Sustainable Computing: Informatics and Systems 29 (2021) 100457.

[39] T. Haener, M. keneken, M. Roetteler, K. M. Svore, Quantum circuits for floating-point arithmetic, in: International Conference on Reversible Computation, Springer, 2018, pp. 162–174.

[40] S. Dutta, A. Suau, S. Dutta, S. Roy, B. K. Behera, P. K. Panigrahi, Quantum circuit design methodology for multiple linear regression, arXiv preprint arXiv:1811.01726 (2018).

[41] S. A. Cuccaro, T. G. Draper, S. A. Kutin, D. P. Moulton, A new quantum ripple-carry addition circuit, arXiv preprint quant-ph/0410184 (2004).

[42] O. Kosheleva, Babylonian method of computing the square root: Justifications based on fuzzy techniques and on computational complexity, in: NAFIPS 2009-2009 Annual Meeting of the North American Fuzzy Information Processing Society, IEEE, 2009, pp. 1–6.

[43] I. L. Markov, M. Saeedi, Constant-optimized quantum circuits for modular multiplication and exponentiation, arXiv preprint arXiv:1202.6614 (2012).

[44] S.-M. Cho, A. Kim, D. Choi, B.-S. Choi, S.-H. Seo, Quantum modular multiplication, IEEE Access 8 (2020) 213244–213252.

[45] J.-w. Lee, E. K. Lee, J. Kim, S. Lee, Quantum shift register, arXiv preprint quant-ph/0112107 (2001).