Article

T-Count Optimized Quantum Circuit Designs for Single-Precision Floating-Point Division

S. S. Gayathri ^{1,*} , R. Kumar ^{1,*} , Samiappan Dhanalakshmi ¹, Gerard Dooly ² and Dinesh Babu Duraibabu ²

¹ Department of Electronics and Communication Engineering, Faculty of Engineering and Technology, SRM Institute of Science and Technology, College of Engineering and Technology, SRM Nagar, Kattankulathur 603203, India; gayathrs@srmist.edu.in (S.S.G.); dhanalas@srmist.edu.in (S.D.)

² Centre for Robotics and Intelligent Systems, University of Limerick, V94 T9PX Limerick, Ireland; Gerard.Dooly@ul.ie (G.D.); dineshbabu.duraibabu@ul.ie (D.B.D.)

* Correspondence: kumarr@srmist.edu.in

Abstract: The implementation of quantum computing processors for scientific applications includes quantum floating points circuits for arithmetic operations. This work adopts the standard division algorithms for floating-point numbers with restoring, non-restoring, and Goldschmidt division algorithms for single-precision inputs. The design proposals are carried out while using the quantum Clifford+T gates set, and resource estimates in terms of numbers of qubits, T-count, and T-depth are provided for the proposed circuits. By improving the leading zero detector (LZD) unit structure, the proposed division circuits show a significant reduction in the T-count when compared to the existing works on floating-point division.

Keywords: quantum circuits; floating-point division; fault-tolerant; T-count; qubits

Citation: Gayathri, S.S.; Kumar, R.; Dhanalakshmi, S.; Dooly, G.; Duraibabu, D.B. T-Count Optimized Quantum Circuit Designs for Single-Precision Floating-Point Division. *Electronics* **2021**, *10*, 703. <https://doi.org/10.3390/electronics10060703>

Academic Editor: Lucas Lamata
Received: 8 February 2021
Accepted: 14 March 2021
Published: 17 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Quantum integer circuits attracted researchers to implement Shor's factorization algorithm, in order to address fundamental arithmetic problems on a quantum computer and resolve discrete logarithmic problems in a polynomial time [1]. The applications of quantum computing are widespread in the field of quantum chemistry [2] and in solving the linear system of equations [3,4]. However, quantum computers simulate some problems with large-scale qubits that are not available, yet that are also essential in constructing some optimum quantum circuits to assist in the efficient hardware design of future quantum computers [5]. Quantum circuits provide special features, such as a one-to-one mapping between vectors of input and output, as contrasted with traditional classic circuits. As a result of the unique mapping between inputs and outputs, additional input and outputs, called ancilla inputs, and garbage outputs are generated [6]. The garbage generated must be cleared by running the circuit backward, after copying the necessary results to excess auxiliary constants, so that the quantum circuits can be used to implement the complete quantum algorithm. The hardware for physical quantum computing is uniquely susceptible to noise errors [7]. Circuits can be designed using the fault-tolerant Clifford+T gates set to overcome the noise-intolerant behavior of physical quantum computing systems. The implementation of fault-tolerant quantum circuits comes with an additional overhead for T-gates count. The costs that are related to T-gate implementation make T-count and T-depth calculations an essential parameter for the quantum circuit's implementation that permits reliable and scalable computing [8]. A small number of qubits is supported by existing quantum computers, which makes it necessary to create any quantum circuit with fewer qubits. Quantum circuits that are available in literature concentrate on minimizing the qubits and T-Count for fundamental integer arithmetic applications that show a substantial decrease in T-count [9–16]. Quantum Fourier Transform (QFT) is the most potent operation in quantum computing. It helps to extract the periodicity of the quantum states' amplitude

in Shor's factorization algorithm [17,18]. The importance of QFT is estimated in many applications in cryptosystems, like Rivest–Shamir–Adleman (RSA) algorithm and in Elliptic curve cryptography [19]. Single-qubit rotations performed in QFT using R gate are challenging to realize in a fault-tolerant manner [18]. QFT circuits utilize fault-tolerant T gate to propose a fault-tolerant phase rotations, which presents an additional overhead on T-count or ancillary qubits [18,20]. QFT was also used to produce a variety of integer arithmetic circuits, but they are fault-intolerant and, thus, highly susceptible to noise errors [21,22]. Floating-point numbers allow for accurate representations of magnitude that would benefit algorithms of linear systems and quantum machine learning algorithms [4,23,24]. While floating-point arithmetic plays a crucial role in modern, large-scale digital signal processing systems, research on reversible and quantum floating-point arithmetic circuits has received a minimum level of attention than the integer problem solving circuitry [5,25–28]. In a number of the science, multimedia, and adaptive computing sectors, floating-point division is an essential operation [29]. Restoring, non-restoring, and SRT division algorithms are slow division algorithms, whereas fast division algorithms are Goldschmidt and Newton–Raphson algorithm. The slow division algorithms are numerical repeat algorithms, which generate a single bit quotient per iteration and the number of iterations is the same as the input qubits [30]. The fast division algorithm utilizes $\log_2 N$ number of iterations to compute the quotient for N-bit input. The fast division algorithms are functional algorithms that typically compute the input qubit's reciprocal and then perform subtraction and multiplication to compute the input qubits' accurate quotient. In this paper, a quantum division circuit for single-precision floating-point (SPFP) number is proposed using restoring, non-restoring, and Goldschmidt algorithms. The critical goal in developing a quantum division circuit using Restoring, non-restoring, and Goldschmidt algorithm is to minimize the T-count and T-depth of the proposed quantum circuit. Section 2 provides the preliminary literature on floating-point numbers, floating-point division algorithm, quantum Clifford+T gates, the structure of Gidney's adder, quantum adder-subtractor using Gidney's adder [10], and controlled-adder block [15], which is utilized to design the proposed division circuits. Section 3 deals with the design of the proposed quantum leading zero detector, Section 4 deals with the design of a quantum circuit for the proposed single-precision restoring, non-restoring division circuit, and Section 5 deals with the design of the quantum circuit for the proposed single-precision Goldschmidt division algorithm. Section 6 deals with resource utilization discussion of the proposed floating-point division circuits and Section 7 presents the conclusion of the proposed work.

2. Preliminaries

In floating-point depictions, every number is represented in three segments namely sign bit $[S_x]$, exponent bits $[E_x]$, and mantissa, which is the fractional part $[M_x]$. The IEEE754 standard floating-point number portrayal varies in the length of exponent and mantissa bits for the different precision numbers [5,26]. For example, a single-precision floating-point (SPFP) number can be represented, as shown in Equation (1). Algorithm 1 presents the steps for the calculation of the division of two floating-point numbers. The resulting sign bit is initially computed while using an XOR operation on input sign bits. The exponent of the divisor subtracts the dividend exponent and then division takes place on qubits of mantissa. The intermediate exponent and mantissa are normalized in order to obtain the resultant quotient qubits. Figure 1 shows the high-level description of the generic quantum floating-point division circuit demonstrating the steps that are outlined in Algorithm 1.

$$(-1)_x^S * (1.M_{x22}M_{x21}...M_{x0}) * 2^{E_x-127} \quad (1)$$

Algorithm 1: Algorithm for floating-point division.**Input:** Two input floating-point numbers $S_x E_x M_x$ $S_y E_y M_y$ **Result:** Floating-point number $S_r E_r M_r$

$$S_r = S_x \oplus S_y$$

$$I_{Er} = E_x - E_y$$

$$I_{Mr} = M_x / M_y$$

$$E_r M_r = \text{Normalize}(I_{Er}, I_{Mr})$$

Figure 1. High level overview of Quantum floating-point division circuit.

The proposed quantum floating-point division circuit is realized using the Clifford+T gates set listed in Table 1. The performance metric parameter of Fault-tolerant quantum circuit implementation are defined, as follows:

1. T-Count: The number of T-gates employed in the quantum circuit.
2. T-depth: The number of T-gate layers in the quantum circuit that can perform parallel quantum information processing.
3. Qubits: The total number of qubits required to implement the quantum circuit.
4. Circuit size (KQ): T-depth $\times$ No. of qubits.

The proposed quantum floating-point division circuits use CNOT gate, Pauli-x gate, and CCNOT(Toffoli gate) to implement the complete circuit. In terms of fault-tolerant Clifford+T gates set, several researchers worked on the decomposition of the CCNOT gate in order to compute quantum information and estimate the cost of the T-gate [10,31–33]. Among the handful designs of CCNOT gate decomposition, the Toffoli gate implementation by Amy et al. [31] and the Toffoli implementation as temporary logic AND by Gidney [10] shows a better resource utilization. CCNOT decomposition by Amy et al. is widely used for logical operations other than logical AND shows a T-count of 7 and its T-depth is 3, with no excess ancilla qubits [31] is shown in Figure 2. The temporary logic AND decomposition of CCNOT gate shows a T-count of 4 and a T-depth of 2 in the computation section and a T-count of 0 in the uncomputation section [10] is shown in Figures 3 and 4.

Figure 2. Decomposition of Toffoli gate with T-count = 7 and T-depth = 3 [31].

Figure 3. Decomposition of Toffoli gate as temporary logic AND with T-count = 4 and T-depth = 2 [10].

Figure 4. Uncomputation circuit of temporary logic AND with T-count = 0 [10].

Table 1. Clifford+T gates.

S.No	Gate	Symbol	Matrix
1	Pauli-x	X (or) $\oplus$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$
2	Hadamard	H	$\frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
3	CNOT		$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
4	T-Gate	T	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/4} \end{bmatrix}$
5	T-Gate Hermitian Transpose	T^{-1} or $T^\dagger$	$\begin{bmatrix} 1 & 0 \\ 0 & e^{-i\pi/4} \end{bmatrix}$
6	Phase Gate	S	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
7	Phase Gate Hermitian Transpose	S^{-1} or $S^\dagger$	$\begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}$

2.1. Elementary Quantum Circuits for Single-Precision Operands Used in Designing the Proposed Quantum Floating-Point Divider

Craig Gidney's adder has a T-count of $4n$ with a T-depth of $2n$, which is the lowest T-count adder [10] employed to design the subtractor and adder/subtractor components in the proposed quantum floating-point division circuit. A quantum control-add block is required to realize the slow division algorithms on floating-point operands. The control-add unit adapted from the design proposed in 2020 by Haner et al. [15] using two CCNOT gates, four CNOT gates, a temporary logical AND gate, and uncomputation gate is estimated to have a T-count of $18n$ with a T-depth of $8n$ and n ancillary qubits is shown in Figure 5.

Figure 5. Quantum controlled addition circuit with T-count = 18 and T-depth = 8 [10].

The quantum adder/subtractor circuit is constructed with the ripple carry adder [10], which does not lead to the excess usage of T-gate by the circuit and, thus, the T-cost circuit of this quantum circuit remains the same as the quantum n qubit adder with carry input is depicted in Figure 6.

Figure 6. Quantum adder/subtractor for n qubits with T-count = $4n$, T-depth = $2n$ and ancillae = $n + 1$ qubits [10].

The structure of the quantum ripple borrow subtractor for the subtraction of operands in the proposed floating-point division circuits has a T-count of $4n$, T-depth of $2n$, with n ancillary qubits as Gidney's adder [10] is shown in Figure 7.

Figure 7. Quantum Subtractor for n qubits adopted from [10] with T-count = $4n$, T-depth = $2n$ and ancillae = n qubits.

3. Proposed Quantum Leading Zero Detector

A quantum leading zero detector circuit and a quantum normalization unit are required to normalize and pack the floating-point divider result. Here, the Clifford+T gates set are used to construct a quantum leading zero detector unit that counts the number of zeros in the most important bit positions before the first one appears. A four-qubit quantum leading zero detector is proposed and it is scaled for the size of the mantissa. Equations (2)–(4) depict the output expression of the proposed four-qubit quantum leading zero detector by changing the classical Boolean expression. The output variable Q_0, Q_1 calculates the leading zero count and the output variable V becomes high when all of

the input qubits are zeros. Figure 8 shows a quantum circuit for the proposed four-qubit quantum LZD unit, which uses nine CCNOT gates, of which three CCNOT gates are uncomputed for the auxiliary qubit restoration, hence making a T-count of 24, T-depth is 12, and ancillary qubits count is 4. A 32 qubit leading zero detector is necessary for performing normalization and rounding on a floating-point number, as the leading zero detector unit generates $\log_2 N$ bits output for N bit inputs. An input array of 32 qubits is required to normalize the proposed division circuit's output, but the mantissa's size is only 23 qubits. As a required input, it is possible to take the first 23 qubits and consider the remaining nine qubits as ancillary inputs.

$$X'_3 X_2 + X'_3 X'_1 = Q_0 \quad (2)$$

$$X_3 + X_2 = Q_1 \quad (3)$$

$$X_3 + X_2 + X_1 + X_0 = V \quad (4)$$

The proposed LZD is constructed based on the delay effective design that is presented in [34] to adopt it for SPFP input. The mantissa is first divided into nibbles in order to evaluate the number of leading zeros: the nibble containing Most Significant Bit (MSB) is nibble 0 and Least Significant Bit (LSB) is nibble 7. Two output variables, namely V_i and Q_i , are expected to be created by each nibble. Where V_i denotes the logical sum of each nibble as complementary, and Q_i represents the number of leading zeros in each nibble. The output of 8 bit LZD generates the output $VQ'_2Q'_1Q'_0$, where $V = V_1 + V_2$. The output Q'_i is generated as a function with the help of the previous stage inputs Q_i and V_i . The output V is the logical sum of V_i and output $Q'_i = 0$, Q_0Q_1 is produced when $V_1 = 1$ and when $V_2 = 1$ the output $Q'_i = 1$, Q_2Q_3 is produced. The proposed 8 qubit quantum leading zero detector utilizes two 4 bit leading zero detectors with the control logic is shown in Figure 9. The proposed leading zero detector utilizes 19 CCNOT gates and 11 ancillary qubits to generate the output VQ_i . Table 2 shows the resource utilization of the proposed 8 qubit quantum leading zero detector. The proposed 8 qubit LZD is optimized in T-count, T-depth, and ancilla, showing a significant savings of around 32.14%, 29.62%, and 71.05% of the parameters mentioned above, respectively, over the existing work proposed in [26]. The proposed 8 qubit LZD is also compared with the reversible LZD proposed in [35], which is modified to adapt it to the quantum environment. The comparison result shows a significant savings in T-count, T-depth, and ancilla around 45.23%, 65.78%, and 47.61%, respectively, over the existing LZD [35].

Figure 8. Proposed 4-qubit quantum leading zero detector. The outputs are highlighted.

Table 2. Resource utilization of proposed 8 qubit leading zero detector with the existing design.

Parameter	Existing Design [26]	Modified Design [35]	Proposed Design	% of imp. over [26]	% of imp. over Modified [35]
T-count	112	84	76	32.14 %	45.23 %
T-depth	54	63	38	29.62 %	65.78 %
Ancilla	38	21	11	71.05 %	47.61 %

Figure 9. Proposed 8-qubit quantum Leading zero detector.

4. Quantum Circuit Designs for Mantissa Division of the Floating-Point Number Using Restoring and Non-Restoring Division Algorithm

4.1. Quantum Restoring Circuit for Mantissa Division

Algorithm 2 presents the restoring division algorithm for mantissa division.

Algorithm 2: Restoring division algorithm for mantissa of the floating-point number.

Input: Two input numbers M_x and M_y

Result: Quotient Q , Remainder R_n

Assign $Q = M_x$

Initialize $R = 0_{n-1}0_{n-2}\dots0_10_0$

for $i = 1$ to $n - 1$

$Y = R_{n-1-i}R_{n-2-i}\dots R_1R_0Q_{n-1}\dots Q_{n-i}$

$Y = Y - M_y$

if $Y < 0$

$Y = Y + M_y$

end

$R_{n-i} = \text{Not}(R_{n-1-i})$

end

$Q = Q - M_y$

if $Q < 0$

$Q = Q + M_y$

end

$R_0 = \text{Not}(Q_{n-1})$

Figure 10 presents the quantum circuit for calculating the quotient and remainder for the mantissa part of the floating-point number using the restoring division algorithm for a 4 qubit input. This quantum circuit performs n iterations for n qubit inputs. Each iteration uses an n qubit quantum subtractor and n qubit quantum controlled adder. The subtractor computes the output of $Y = Y - M_y$ and the CNOT gate rewrites the location of register R_{n-i} as $R_{n-i} \oplus Y_{n-1}$. The quantum controlled adder circuit computes $Y = Y + M_y$ if the condition $Y < 0$ is true. The quantum Pauli-x gate writes the inverse of Q_{n-1} and produces the output R_0 bit of the division circuit. The subtractor in the final iteration computes the output of $Q = Q - M_y$ and the CNOT gate is applied on R_0 and Q_{n-1} , such that Q_{n-1} is transformed to $Q_{n-1} \oplus R_0$. The controlled adder computes the $Q = Q + M_y$ if the condition $Y < 0$ is true. After performing the aforementioned steps, the quotient output is stored in register Q and the register locations R holds the remainder output.

Figure 10. Quantum restoring division circuit for four qubit input.

4.2. Quantum Circuit for Mantissa Division Using Non-Restoring Algorithm

Algorithm 3 presents the non-restoring division algorithm for mantissa division

Algorithm 3: Non-restoring division algorithm for mantissa of the floating-point number.

Input: Two input numbers M_x and M_y

Output: Quotient Q , Remainder R_r

Initialize $R = 0_{n-1}0_{n-2}\dots0_10_0$

Initialize $Q = 0_{n-1}0_{n-2}\dots0_10_0M_{Xn-1}M_{Xn-2}\dots M_{X1}M_{X0}$

$Q = Q - M_y$

for $i = 1$ to $n - 1$

$Q_{n-i} = \text{Not}(Q_{n-i})$

$Y = Q_{n-1-i}\dots Q_0$

if $Q_{n-i} = 0$

$Y = Y + M_y$

else $Y = Y - M_y$

end

end

If $R < 0$

$R = R + M_y$

end

$Q_0 = \text{Not}(Q_0)$

Figure 11 shows the quantum circuit built for mantissa division using the non-restoring algorithm. In the initial step, the quantum subtractor calculates $Q = Q - M_y$ output and then the iteration starts. The quotient bit Q_{n-i} is reversed in each iteration and the quotient bits are concatenated into a temporary variable Y . By performing controlled addition and subtraction using the quantum controlled adder-subtractor circuit, the quotient output is computed. An addition is performed by the quantum adder-subtractor circuit and the output $Y = Y + M_y$ is generated when $Q_{n-i} = 0$. Similarly, for iterations $1 \leq n - 1$, the circuit performs subtraction and produces the output $Y = Y - M_y$ when $Q_{n-i} = 1$.

Figure 11. Quantum non-restoring division circuit for four qubit input.

5. Quantum Division Circuit Design Proposed for SPFP Number Using Goldschmidt Division Algorithm

The division algorithm of Goldschmidt requires a pair of multiplications and a subtraction unit per iteration. Although the Goldschmidt algorithm involves two multiplications per iteration, the key feature of this algorithm is that both multiplications are independent and, hence, the circuit depth is reduced [36]. Algorithm 4 provides the algorithm for calculating the quotient of two numbers X and Y using the Goldschmidt algorithm.

Algorithm 4: Goldschmidt Division Algorithm for floating-point number with single-precision.

Input: X and Y

Initialize $N_{-1} = X, D_{-1} = Y, F_{-1} = 1/Y$

If $i = 0$ $N_1 = N_0 F_{-1}$

$D_1 = D_0 F_{-1}$

for $i = 1$ to 4

$F = 2 - D_i$

$N_i = N_{i-1} F_{i-1}$

$D_i = D_{i-1} F_{i-1}$

end

$N_i = Q$

$D_i = 1$

Figure 12 shows a high-level description of the SPFP quantum division circuit using the Goldschmidt division. The denominator's reciprocal approximation is determined using a look-up table (LUT) that finds the closest reciprocal approximation of the denominator. A LUT is an array that contains a floating-point number that stores pre-calculated reciprocal values of entire ranges. The denominator is found once the estimated reciprocal and the numerator inputs of the division circuit are multiplied by the denominator's reciprocal value and N_i and D_i are produced. The new factor F_i is computed using the formula $2 - D_i$, and the new factor is multiplied with each N_i and D_i in each iteration. The number of iteration to compute the quotient of two input floating-point number is calculated as $\log_2 N$ for N -bit input.

Figure 12. High-level overview proposed division circuit for a single-precision floating-point (SPFP) number using Goldschmidt division algorithm.

5.1. Quantum Subtractor Circuit Design for SPFP Input

The addition and subtraction of two floating-point numbers is the prerequisite for implementing the floating-point arithmetic logic unit (ALU) of the signal processors and a quantum floating-point divider. Algorithm 5 provides the algorithm for adding and subtracting floating-point numbers. In Figure 13, a high level view of the quantum circuit is depicted for the addition and subtraction of a pair of floating-point numbers.

Algorithm 5: Floating-point addition and subtraction algorithm.

Input: Two input floating-point numbers $S_x E_x M_x$ and $S_y E_y M_y$

Result: Floating-point number $S_r E_r M_r$

$$S_r = S_x \oplus S_y$$

Compare E_x with E_y

$$\delta E = E_x - E_y$$

if $E_x \leq E_y$ **then**

 Swap E_x with E_y

 Swap M_x with M_y

 Adjust Exponent

 Align Mantissa

$$E_x = E_y$$

$$I E_x = E_x$$

$$I M_r = M_x - M_y$$

else

 No swap

 Adjust Exponent

 Align Mantissa

$$E_x = E_y$$

$$I E_r' = E_x$$

$$I M_r = M_x - M_y$$

end

Figure 13. High level overview of proposed quantum subtractor circuit for a SPFP number. Intermediate outputs generated must be set back to the original state by running the compute block backward.

A quantum floating-point subtractor requires a quantum magnitude comparator circuit that is equivalent to the size of the exponent qubits. Several works have been proposed for the realization of a quantum comparator while using serial and tree-based architecture [37–39]. An efficient quantum multi-qubit magnitude comparator was proposed to compute $a < b$ and $a \geq b$ for image binarization in quantum image processing [40]. All of these designs focus on reducing the qubits by reducing ancillary qubits utilized in the circuit. Here, a quantum comparator is used that is designed using Gidney's adder [10] that computes $a \leq b$ and $a > b$, depending on the borrow output of the subtractor. Figure 14 shows the flowchart to design a multi-qubit magnitude comparator using a quantum subtractor with an additional ancilla for two n -qubit inputs A_n and B_n . The final borrow output is copied on the extra ancilla using a CNOT gate. Whenever the borrowed output is high, the input $A_n < B_n$ is calculated, and when the final borrowed output is low, $A_n \geq B_n$. Therefore, the multi qubit quantum magnitude comparator that provides a decreased T-count of $4n$ and uses ancillary qubits of $n + 1$ for qubit inputs of n is shown in Figure 15.

Figure 14. Flowchart to design a magnitude comparator using subtractor.

Figure 15. Multi-qubit string comparator using Gidney's adder.

5.2. Quantum Circuit Design of a Multiplier for Floating-Point Number with Single-Precision

The quantum Goldschmidt division circuit for SPFP number requires two multiplication operations for each iteration. Algorithm 6 presents the algorithm to implement floating-point multiplication.

The mantissa multiplier unit is required to perform quick calculations, as multiplication operations in the Goldschmidt division algorithm are independent operations. Vedic multipliers hold the merit of having less delay, as an increase in bits shows a slow increase in delay and area [41,42] and, hence, the Vedic multiplier is used to realize the floating point multiplication algorithm. Typically a 4×4 vedic multiplier requires 16 CCNOT gates that generate the partial product, seven full adder circuits, eight half adder circuits, and XOR with a total ancilla of 31 qubits. The 4×4 quantum vedic multiplier

can be uniformly scaled with reference to the design that is shown in Figure 16. Figure 17 shows the high-level description of a quantum floating-point multiplier circuit. For various floating-point precision numbers, this structure can be practiced uniformly. Figure 18 shows the line diagram to design a 4×4 multiplier using the vedic multiplication algorithm.

Algorithm 6: Floating-point multiplication algorithm.

Input: Two input floating-point numbers $S_x E_x M_x$ and $S_y E_y M_y$

Output: Floating-point number $S_r E_r M_r$

$$S_r = S_x \oplus S_y$$

$$I_{Er} = E_x + E_y$$

$$I_{Mr} = M_x * M_y$$

$$E_r, M_r = \text{Normalize}(I_{Er}, I_{Mr})$$

Figure 16. General Structure of $N \times N$ Vedic Multiplier.

5.3. $N \times N$ Vedic Multiplier

Each input multiplier and multiplicand is divided into equals of lower and higher order bits to multiply. The $N \times N$ multiplier will require four $N/2$ multipliers and three N -bit ripple adders in order to produce the final product output. Figure 16 shows the proposed General Vedic multiplier structure.

Figure 17. High level overview of quantum multiplication circuit for a SPFP number. Intermediate outputs that are generated must be set back to the original state by running the compute block backward.

Figure 18. Line diagram structure of 4×4 Vedic Multiplier.

6. Resource Utilization of the Proposed Quantum Divider Circuits for SPFP Number

By evaluating the T-count and qubit utilization of each stage in the division algorithms, the T-count and qubit analysis of the proposed floating-point division is calculated. The Total T-count and ancilla are obtained by summing up the T-count at each level of the proposed floating-point division circuit. The proposed 8 qubit leading zero detector is uniformly scaled while using the architecture proposed in [43] to detect the number of zeros for the mantissa and align the mantissas to normalize the floating-point number.

6.1. Resource Analysis of the Proposed Quantum SPFP Divider Circuit Using Restoring and Non-Restoring Algorithms

The resource usage of the proposed quantum restoring division and non-restoring division circuit for SPFP division is calculated for each step.

In order to analyze the resource utilization of the proposed floating-point division circuits using non-restoring division algorithm, the methodology differs from the mantissa division of the circuit. The proposed quantum division circuit uses quantum subtractor and quantum adder-subtractor for the length of the mantissa and, hence, the resource utility varies only in the mantissa division of the restoring division algorithm and is illustrated in Table 3.

For the normalization of floating-point number, a quantum barrel shifter circuit is utilized that can operate in a unidirectional and a bidirectional way. The proposed quantum floating-point division circuits employ the quantum bidirectional shift register proposed in [44] that uses the Fredkin (CSWAP) and Feynman (CNOT) gate.

Table 3. Resource utilization of the proposed quantum SPFP division circuit using restoring and Non-restoring division algorithm.

Component	T-Count	T-Depth	Ancilla
8-Qubit exponent subtractor	32	16	8
24-Qubit controlled adder (Restoring division)	$432 \times 23 = 9936$	$192 \times 23 = 4416$	$24 \times 23 = 552$
24-Qubit adder/subtractor (Non-restoring)	$96 \times 23 = 2208$	$48 \times 23 = 1104$	$24 \times 23 = 552$
24-Qubit subtractor	$96 \times 23 = 2208$	$48 \times 23 = 1104$	$24 \times 23 = 552$
Proposed 32 Qubit LZD	408	46	178
Barrel shifter	1687	2	620
Final exponent adjust	32	16	8
Uncomputation circuit	Nil	Nil	32
Total (Restoring)	14,303	5600	1950
Total (Non-Restoring)	6575	2288	1950

6.2. Resource Analysis of the Proposed Quantum SPFP Division Circuit Using Goldschmidt Algorithm

Table 4 shows the resource count of both floating-point subtraction and floating-point multiplication using vedic multiplier excluding leading zero detector and normalization unit. To perform mantissa multiplication using vedic multiplication algorithm, a 24×24 multiplier is constructed using four 12×12 multiplier and three 24 bit ripple carry adders and a partial product generator that generates 576 partial products, as shown in Figure 16.

Table 4. Resource utilization of the proposed SPFP division circuit using Goldschmidt division algorithm.

Component	T-Count	T-Depth	Ancilla
Proposed quantum SPFP multiplier	$7936 \times 12 = 95,232$	$1448 \times 12 = 17,376$	$1984 \times 12 = 23,808$
Proposed quantum SPFP subtractor	$3965 \times 5 = 19,825$	$82 \times 5 = 410$	$1034 \times 5 = 5170$
Proposed 32 Qubit LZD	408	46	178
Barrel shifter	1687	2	620
Final exponent adjust	32	16	8
Copying intermediate outputs	Nil	Nil	128
Uncomputation circuit	Nil	Nil	32
Total	117,187	17,850	29,944

There are few designs for floating-point division algorithms that have been proposed in the literature that implement slow and fast division algorithms. To adopt those designs for the quantum circuit environment [27,45], the reversible division circuit is modified by analyzing the quantum decomposition of the reversible gates used in the circuit. A comparison of resource utilization of proposed quantum SPFP division circuit is compared with the slow division and fast division algorithms are shown in Tables 5 and 6 respectively.

Table 5. Comparison of proposed single precision slow floating-point division quantum circuits with the existing design.

Designs	Qubits	T-Count	T-Depth
Existing Design [45]	2207	28672	NA
Proposed Restoring divider	2014	14,303	5584
Proposed Non-restoring divider	2046	6575	2288

Table 6. Comparison of proposed single-precision Goldschmidt floating-point division quantum circuits with the existing design.

Designs	Qubits	T-Count	T-Depth
Existing Design (Modified) [27]	29,074	227,920	59,916
Proposed Goldschmidt divider	30,008	117,187	17,850

The proposed quantum circuit for the restoring division algorithm is showing an improvement of 8.74%, 50.11% over the existing design in ancilla and T-count, respectively. Similarly, the proposed non-restoring division quantum circuit shows an improvement of 7.29%, 77.06 % over existing design in ancilla and T-count, respectively [45]. The proposed quantum SPFP division circuit using the Goldschmidt division algorithm is compared with the existing modified reversible floating-point divider [7]. The proposed SPFP quantum division circuit achieves a greater T-count savings of 48.58 % and T-depth savings of 70.20% over the existing design [27].

7. Conclusions

The implementation of a SPFP division circuit using the Restoring, non-restoring, and Goldschmidt algorithm using Clifford + T gates set is discussed in this paper. Several combinations of fundamental arithmetic operators, such as adders, subtractors, multipliers, leading zero detector, and normalization unit, are implemented as efficiently as possible to create the floating-point division circuit. The proposed floating-point division circuits produce garbage outputs. In order to eliminate the garbage output, the quotient is copied on temporary ancilla to run the entire circuit backward. The uncomputation phase of the proposed division circuits doesn't contribute to the T-count or T-depth of the resultant circuit. The proposed quantum division circuit using restoring division algorithm and non-restoring division algorithm shows a better T-count reduction of 50.11% and 77.06 %, respectively, when compared with the existing work. The proposed quantum restoring and quantum non-restoring division circuits also provide savings in ancillary qubits of around 7.29% and 10%, respectively, over the existing design. The proposed Goldschmidt's division quantum circuit shows more significant savings in the T-count of 48.58% and T-depth of 70.20% over the existing work. The proposed floating-point division circuits can be used to design complex circuits where T-gate cost is of paramount concern.

Author Contributions: Conceptualization and Formal analysis: R.K.; Methodology and original Draft writing: S.S.G.; Formal analysis and Investigation: S.D.; writing—review and editing: G.D., D.B.D.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

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 conflict of interest.

References

1. Beauregard, S. Circuit for Shor's Algorithm Using $2n + 3$ Qubits. *arXiv* **2002**, arXiv:quant-ph/0205095.
2. Babbush, R.; Berry, D.W.; Kivlichan, I.D.; Wei, A.Y.; Love, P.J.; Aspuru-Guzik, A. Exponentially more precise quantum simulation of fermions in second quantization. *New J. Phys.* **2016**, *18*, 033032. [[CrossRef](#)]
3. Reiher, M.; Wiebe, N.; Svore, K.M.; Wecker, D.; Troyer, M. Elucidating reaction mechanisms on quantum computers. *Proc. Natl. Acad. Sci. USA* **2017**, *114*, 7555–7560. [[CrossRef](#)]
4. Harrow, A.W.; Hassidim, A.; Lloyd, S. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.* **2009**, *103*, 150502. [[CrossRef](#)]
5. Haener, T.; Soeken, M.; Roetteler, M.; Svore, K.M. Quantum circuits for floating-point arithmetic. In Proceedings of the International Conference on Reversible Computation, Leicester, UK, 12–14 September 2018; pp. 162–174.
6. Bennett, C.H. Logical reversibility of computation. *IBM J. Res. Dev.* **1973**, *17*, 525–532. [[CrossRef](#)]

7. Bravyi, S.; Kitaev, A. Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A* **2005**, *71*, 022316. [[CrossRef](#)]
8. Paler, A.; Polian, I.; Nemoto, K.; Devitt, S.J. Fault-tolerant, high-level quantum circuits: Form, compilation and description. *Quantum Sci. Technol.* **2017**, *2*, 025003. [[CrossRef](#)]
9. Cuccaro, S.A.; Draper, T.G.; Kutin, S.A.; Moulton, D.P. A new quantum ripple-carry addition circuit. *arXiv* **2004**, arXiv:quant-ph/0410184.
10. Gidney, C. Halving the cost of quantum addition. *Quantum* **2018**, *2*, 74. [[CrossRef](#)]
11. Thapliyal, H. Mapping of subtractor and adder-subtractor circuits on reversible quantum gates. In *Transactions on Computational Science XXVII*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 10–34.
12. Muñoz-Coreas, E.; Thapliyal, H. Quantum circuit design of a t-count optimized integer multiplier. *IEEE Trans. Comput.* **2018**, *68*, 729–739. [[CrossRef](#)]
13. Dutta, S.; Bhattacharjee, D.; Chattopadhyay, A. Quantum circuits for Toom-Cook multiplication. *Phys. Rev. A* **2018**, *98*, 012311. [[CrossRef](#)]
14. Thapliyal, H.; Munoz-Coreas, E.; Varun, T.; Humble, T. Quantum circuit designs of integer division optimizing T-count and T-depth. *IEEE Trans. Emerg. Top. Comput.* **2019**. [[CrossRef](#)]
15. Häner, T.; Jaques, S.; Naehrig, M.; Roetteler, M.; Soeken, M. Improved quantum circuits for elliptic curve discrete logarithms. In Proceedings of the International Conference on Post-Quantum Cryptography, Paris, France, 15–17 April 2020; pp. 425–444.
16. Thapliyal, H.; Muñoz-Coreas, E.; Khalus, V. T-count and Qubit Optimized Quantum Circuit Designs of Carry Lookahead Adder. *arXiv* **2020**, arXiv:2004.01826.
17. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Rev.* **1999**, *41*, 303–332. [[CrossRef](#)]
18. Goto, H. Resource requirements for a fault-tolerant quantum Fourier transform. *Phys. Rev. A* **2014**, *90*, 052318. [[CrossRef](#)]
19. Nam, Y.; Su, Y.; Maslov, D. Approximate quantum Fourier transform with $O(n \log(n))$ T gates. *NPJ Quantum Inf.* **2020**, *6*, 1–6. [[CrossRef](#)]
20. Jones, N.C.; Whitfield, J.D.; McMahon, P.L.; Yung, M.H.; Van Meter, R.; Aspuru-Guzik, A.; Yamamoto, Y. Faster quantum chemistry simulation on fault-tolerant quantum computers. *New J. Phys.* **2012**, *14*, 115023. [[CrossRef](#)]
21. Weinstein, Y.S.; Pravia, M.; Fortunato, E.; Lloyd, S.; Cory, D.G. Implementation of the quantum Fourier transform. *Phys. Rev. Lett.* **2001**, *86*, 1889. [[CrossRef](#)] [[PubMed](#)]
22. Ruiz-Perez, L.; Garcia-Escartin, J.C. Quantum arithmetic with the quantum Fourier transform. *Quantum Inf. Process.* **2017**, *16*, 152. [[CrossRef](#)]
23. Clader, B.D.; Jacobs, B.C.; Sprouse, C.R. Quantum algorithm to calculate electromagnetic scattering cross sections. In Proceedings of the Quantum Information and Measurement, Optical Society of America, Rochester, NY, USA, 17–20 June 2013; pp. W6–W26.
24. Lloyd, S.; Mohseni, M.; Rebentrost, P. Quantum algorithms for supervised and unsupervised machine learning. *arXiv* **2013**, arXiv:1307.0411.
25. Jain, J.; Agrawal, R. Design and development of efficient reversible floating point arithmetic unit. In Proceedings of the 2015 Fifth International Conference on Communication Systems and Network Technologies, Gwalior, India, 4–6 April 2015; pp. 811–815.
26. Nguyen, T.D.; Van Meter, R. A resource-efficient design for a reversible floating point adder in quantum computing. *ACM J. Emerg. Technol. Comput. Syst.* **2014**, *11*, 1–18. [[CrossRef](#)]
27. AnanthaLakshmi, A.; Sudha, G.F. A novel power efficient 0.64-GFlops fused 32-bit reversible floating point arithmetic unit architecture for digital signal processing applications. *Microprocess. Microsyst.* **2017**, *51*, 366–385. [[CrossRef](#)]
28. Kamaraj, A.; Marichamy, P. Design of Fault-Tolerant Reversible Floating Point Division. *Inf. MIDEM* **2018**, *48*, 161–172.
29. Daumas, M.; Finot, C. Division of floating point expansions with an application to the computation of a determinant. *J. Univers. Comput. Sci.* **1999**, *5*, 323–338.
30. Schulte, M.J.; Tan, D.; Lemonds, C.E. Floating-point division algorithms for an x86 microprocessor with a rectangular multiplier. In Proceedings of the 2007 25th International Conference on Computer Design, Lake Tahoe, CA, USA, 7–10 October 2007; pp. 304–310.
31. Amy, M.; Maslov, D.; Mosca, M.; Roetteler, M. A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2013**, *32*, 818–830. [[CrossRef](#)]
32. Jones, C. Novel constructions for the fault-tolerant Toffoli gate. *Phys. Rev. A* **2013**, *87*, 022328. [[CrossRef](#)]
33. Welch, J.; Bocharov, A.; Svore, K.M. Efficient approximation of diagonal unitaries over the Clifford+ T basis. *arXiv* **2014**, arXiv:1412.5608.
34. Oklobdzija, V.G. An algorithmic and novel design of a leading zero detector circuit: Comparison with logic synthesis. *IEEE Trans. Very Large Scale Integr. Syst.* **1994**, *2*, 124–128. [[CrossRef](#)]
35. AnanthaLakshmi, A.; Sudha, G.F. Design of an efficient reversible single precision floating point adder. *Int. J. Comput. Intell. Stud.* **2015**, *4*, 2–30. [[CrossRef](#)]
36. Even, G.; Seidel, P.M.; Ferguson, W.E. A parametric error analysis of Goldschmidt's division algorithm. *J. Comput. Syst. Sci.* **2005**, *70*, 118–139. [[CrossRef](#)]
37. Wang, D.; Liu, Z.H.; Zhu, W.N.; Li, S.Z. Design of quantum comparator based on extended general Toffoli gates with multiple targets. *Comput. Sci.* **2012**, *39*, 302–306.

38. Al-Rabadi, A.N. Closed-system quantum logic network implementation of the viterbi algorithm. *Facta Univ. Ser. Electron. Energ.* **2009**, *22*, 1–33. [[CrossRef](#)]
39. Thapliyal, H.; Ranganathan, N.; Ferreira, R. Design of a comparator tree based on reversible logic. In Proceedings of the 10th IEEE International Conference on Nanotechnology, Ilsan, Korea, 17–20 August 2010; pp. 1113–1116.
40. Xia, H.; Li, H.; Zhang, H.; Liang, Y.; Xin, J. Novel multi-bit quantum comparators and their application in image binarization. *Quantum Inf. Process.* **2019**, *18*, 229. [[CrossRef](#)]
41. Vijeyakumar, K.; Elango, S.; Kalaiselvi, S. VLSI implementation of high speed energy-efficient truncated multiplier. *J. Circuits Syst. Comput.* **2018**, *27*, 1850077. [[CrossRef](#)]
42. Vijayan, A.E.; John, A.; Sen, D. Efficient implementation of 8-bit vedic multipliers for image processing application. In Proceedings of the 2014 International Conference on Contemporary Computing and Informatics (IC3I), Mysore, India, 27–29 November 2014; pp. 544–552.
43. Miao, J.; Li, S. A design for high speed leading-zero counter. In Proceedings of the 2017 IEEE International Symposium on Consumer Electronics (ISCE), Kuala Lumpur, Malaysia, 14–15 November 2017; pp. 22–23.
44. Biswal, L.; Bhattacharjee, A.; Das, R.; Thirunavukarasu, G.; Rahaman, H. Quantum Domain Design of Clifford+ T-Based Bidirectional Barrel Shifter. In Proceedings of the International Symposium on VLSI Design and Test, Madurai, India, 28–30 June 2018; pp. 606–618.
45. Jamal, L.; Babu, H.M.H. Efficient approaches to design a reversible floating point divider. In Proceedings of the 2013 IEEE International Symposium on Circuits and Systems (ISCAS), Beijing, China, 19–23 May 2013; pp. 3004–3007.