applied sciences

MDPI

Article

Quantum Circuit Implementation of Multi-Dimensional Non-linear Lattice Models

René Steijl

James Watt School of Engineering, University of Glasgow, Glasgow G12 8QQ, UK; rene.steijl@glasgow.ac.uk

Abstract: The application of Quantum Computing (QC) to fluid dynamics simulation has developed into a dynamic research topic in recent years. With many flow problems of scientific and engineering interest requiring large computational resources, the potential of QC to speed-up simulations and facilitate more detailed modeling forms the main motivation for this growing research interest. Despite notable progress, many important challenges to creating quantum algorithms for fluid modeling remain. The key challenge of non-linearity of the governing equations in fluid modeling is investigated here in the context of lattice-based modeling of fluids. Quantum circuits for the D1Q3 (one-dimensional, three discrete velocities) Lattice Boltzmann model are detailed along with design trade-offs involving circuit width and depth. Then, the design is extended to a one-dimensional lattice model for the non-linear Burgers equation. To facilitate the evaluation of non-linear terms, the presented quantum circuits employ quantum computational basis encoding. The second part of this work introduces a novel, modular quantum-circuit implementation for non-linear terms in multi-dimensional lattice models. In particular, the evaluation of kinetic energy in two-dimensional models is detailed as the first step toward quantum circuits for the collision term of two- and three-dimensional Lattice Boltzmann methods. The quantum circuit analysis shows that with $O(100)$ fault-tolerant qubits, meaningful proof-of-concept experiments could be performed in the near future.

Keywords: quantum computing; quantum circuits; lattice-based fluid modeling

Citation: Steijl, R. Quantum Circuit Implementation of Multi-Dimensional Non-linear Lattice Models. *Appl. Sci.* **2023**, *13*, 529. https://doi.org/10.3390/app13010529

Academic Editor: Rosario Lo Franco

Received: 29 November 2022
Revised: 20 December 2022
Accepted: 23 December 2022
Published: 30 December 2022

1. Introduction

The application of Quantum Computing (QC) [1] to fluid dynamics simulation represents a relatively new but growing area of research [2–11]. In general, Computational Fluid Dynamics (CFD) covers a wide range of applications of scientific as well as engineering interest. In terms of required computational resources, many applications push the limits of the current high-performance computers. Examples of demanding applications include turbulent flows, combustion, aero-acoustics as well as multi-phase flows. The potential for QC to create a significant speed-up relative to classical approaches forms the main motivation for the recent research interest. Despite notable progress, many important challenges to creating quantum algorithms for fluid modeling remain [4–6,9,10,12]. The non-linearity of the governing equation of fluid dynamics combined with the inherent linearity of quantum mechanics forms one major challenge. Further important challenges relate to efficiently initializing the quantum register of the quantum computer for the considered problem, as well as obtaining classical output at the end of the computation in the quantum computer. The current and near-feature quantum computers, typically referred to as Noisy Intermediate-Scale Quantum (NISQ)-era quantum computers [13], are characterized by a relatively small number of qubits, limited connectivity and a limited amount of quantum error correction. On these machines, the quantum state in the qubit register will stay in a coherent state for only a limited duration, and therefore, current applications on quantum hardware typically involve hybrid classical-quantum approaches with quantum circuits of limited depth (to reduce required coherence time) and a close coupling and frequent exchange of data between quantum and classical part of the simulation.

Thus far, the overhead associated with repeated quantum measurement and quantum-state re-initialization has played an important part in preventing quantum speed-up for fluid dynamics simulations. Recently the available number of qubits on available hardware has grown to $O(100)$. These qubits are still 'noisy', but with the predicted further growth to $O(10^4)$ or $O(10^5)$ physical qubits, the creation of $O(100)$ 'functional qubits' (as termed by IBM, i.e., with a degree of fault tolerance enabling much deeper quantum circuits than on NISQ-era machines) within the next 5–10 years appears realistic.

It is this context that motivated the current work. The development of quantum algorithms for effective three-dimensional flow simulations based on the Lattice Boltzmann method (LBM) with limited exchange of information between the quantum and classical domains in a hybrid quantum/classical approach forms the main aim. The aspect of minimizing the quantum-classical information exchange is driven by the expectation that even for next-generation quantum hardware, the cost of frequent re-initialization and quantum measurement will remain a bottleneck in achieving computational efficiency. Recently, Budinski [11] presented an interesting quantum algorithm for two-dimensional flow simulation based on the LBM. However, its formulation uses a tight coupling of the quantum and classical domains of the simulations introducing significant overhead for each time step performed. In a previous work, Budinski [14] showed a quantum implementation for lattice-based modeling of the linear advection-diffusion equation, where due to the linearity of the problem, some of the expensive re-initialization steps needed for the non-linear LBM could be avoided. An interesting approach to incorporating non-linearity in quantum algorithms based on Carleman linearization of the LBM was presented by Itani and Succi [10]. However, since this approach is based on linearization, it appears that this approach is mainly suited to weakly non-linear low Reynolds number flows. The current approach differs from these existing works in a number of key aspects. The approach followed here uses encoding of data in the quantum computational basis along with a quantum floating-point format previously introduced by the author [7,12]. Non-linear terms are then evaluated using arithmetic for the quantum floating-point format. This avoids restriction to weakly non-linear low-Reynolds numbers flows. Further, based on the used data encoding and floating-point format, the current work aims to perform multiple time steps in a fluid dynamics simulation on the quantum processor, in contrast to approaches where measurement and re-initialization take place at the end of each step. For the chosen approach, it is expected that with a greatly reduced number of bits representing the mantissa as compared to IEEE-754 single precision [15], e.g., 4–8 qubits are suggested here, the required number of qubits in the derived circuits is $O(100)$. This is before 'transpilation' (as termed by IBM) into quantum circuits expressed in terms of native gates on the considered quantum hardware. This clearly creates a requirement for fault tolerance and coherency well beyond the current NISQ-type hardware. However, considering current predictions, quantum hardware capable of executing the derived circuits can be expected this decade.

The main contributions of the present work can be summarized as follows:

- Detailed analysis of quantum-circuit design for the evaluation of non-linear equilibrium distribution function for the D1Q3 Lattice Boltzmann model as well as for the one-dimensional lattice model for the Burgers equation. The modular design for both applications facilitates large-scale re-use of the different components of the circuits;
- A complexity analysis for the one-dimensional models is presented, highlighting the dependency of quantum-circuit width on the number of mantissa qubits in floating-point representation;
- Detailed description of the design process for quantum circuit implementations of non-linear terms in two- and three-dimensional lattice models, including a complexity analysis showing the dependency on the number of mantissa qubits used;
- Demonstration of how a modified shift-and-add-based multiplier can be used to create efficient, modular circuits for kinetic energy evaluation in two-dimensional models;

- Detailed quantum-circuit description for the proposed modular kinetic energy evaluation circuits based on quantum floating-point arithmetic;

For a previously developed quantum-circuit implementation of the D1Q3 lattice model with fewer qubits but larger circuit depth, the analysis and verification using quantum computing simulation techniques were addressed recently by the author and co-workers in Moawad et al. [12]. The systematic evaluation of the quantum circuits presented in this work forms part of ongoing work by the author and co-workers.

The present work is structured as follows. First, non-linear lattice models used in fluid dynamics are reviewed in Section 2. This section also considers the non-linear convection terms in the Navier–Stokes equations to provide context. Key concepts and principles of QC relevant to this work are reviewed briefly in Section 3. Quantum circuits for the evaluation of the non-linear distribution functions of the D1Q3 Lattice Boltzmann model are detailed in Section 4. For a lattice model for the one-dimensional Burgers equation, similar circuits are presented in Section 5. The extension of quantum-circuit design to two- and three-dimensional lattice models are described in Section 6. For two-dimensional kinetic energy evaluation, an efficient, modular quantum circuit is described in detail in Sections 7 and 8. Section 9 summarizes a step-by-step methodology for the evaluation of algorithms. Finally, Section 10 presents the conclusions and directions for future research work.

2. non-linear Lattice modeling of Fluid Dynamics

To highlight the challenges in creating quantum algorithms for fluid dynamics associated with non-linearity of the governing equations, it is convenient to limit the discussion to the incompressible flows of a Newtonian fluid in two-dimensional space. Then, the momentum conservation equations in the Navier–Stokes equations can be written in vector forms as,

$$\frac{\partial}{\partial t}\begin{pmatrix} U \\ V \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} U^2 + P/\rho \\ UV \end{pmatrix} + \frac{\partial}{\partial y}\begin{pmatrix} UV \\ V^2 + P/\rho \end{pmatrix} = \nu\left[\frac{\partial^2}{\partial x^2}\begin{pmatrix} U \\ V \end{pmatrix} + \frac{\partial^2}{\partial y^2}\begin{pmatrix} U \\ V \end{pmatrix}\right] \quad (1)$$

where U and V are two Cartesian components of the velocity vector, and with P, ρ and ν representing the pressure, fluid density and kinematic viscosity, respectively. In terms of deriving effective quantum algorithms to numerically simulate flows governed by the Navier–Stokes equations, the non-linear convective terms on the left-hand side of Equation (1) form a key challenge. Mass conservation is enforced by the continuity equation

$$\frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} = 0 \quad (2)$$

that completes the Navier–Stokes equations for incompressible, iso-thermal flow. The spatial discretization of Equations (1) and (2) using finite-difference, finite-volume or finite-element approaches has been a topic of CFD research for many years. In the present work, an alternative approach to modeling fluid dynamics based on the Lattice Boltzmann Method (LBM) was selected for a number of reasons:

- In the LBM, the non-linear convection terms shown in Equation (1) do not appear in this form. Instead, the convection-related part of the LBM approach becomes linear. However, the inherent non-linearity of modeling fluid dynamics manifests itself as non-linear terms in the collision step, as detailed later;
- Gradient calculations of U, V and P are not used in the LBM. In case quantum-measurement-related noise is present in the hybrid classical-quantum algorithms, then using the LBM avoids increasing this noise level by computing gradients of this noise data;
- The LBM models convection by 'streaming' components of a particle distribution on a regular lattice, as defined later in this section. This proves to be easier to implement than the numerical flux evaluation used in a direct discretization of Equations (1) and (2).

2.1. LBM for Iso-Thermal Two-Dimensional Flows

The Lattice Boltzmann equation defines the governing equation for Lattice Boltzmann modeling. This equation can be derived from a discrete-velocity discretization of the Bhatnagar–Gross–Krook (BGK) equation, such that the discretized particle distribution function is governed by the following equation:

$$\frac{\partial f_a}{\partial t} + \mathbf{e}_a \cdot \nabla f_a = -\frac{1}{\tau}\left(f_a - f_a^{eq}\right) \; ; \; a \in [0, n_{DV} - 1] \tag{3}$$

where n_{DV} denotes the number of discrete velocities in the model. Furthermore, $f_a(\mathbf{x}, t) = f(\mathbf{x}, \mathbf{e}_a, t)$ is the non-equilibrium distribution for discrete velocity $\mathbf{e}_a$ and f_a^{eq} is the corresponding equilibrium distribution function. In the collision term on the right-hand side, τ represents the relaxation time. For iso-thermal LBM models, based on the two-dimensional D2Q9 model, or the three-dimensional D3Q15, D3Q19 and D3Q27 models, the equilibrium distribution function can be written as,

$$f_a^{eq} = \rho w_a \left[1 + \frac{3}{c^2}\mathbf{e}_a \cdot \mathbf{V} + \frac{9}{2c^4}\left(\mathbf{e}_a \cdot \mathbf{V}\right)^2 - \frac{3}{2c^2}\mathbf{V} \cdot \mathbf{V}\right] \tag{4}$$

where w_a and $\mathbf{e}_a$ denote the weighting factor and discrete velocity for direction a, respectively. The lattice speed c is defined as $c = \delta x / \delta t$. Then, in a single time step, discrete values of the distribution function move to the nearest neighbor lattice point (defined by the direction of the discrete velocity that is considered). This movement is typically referred to as 'streaming'. For the D2Q9 model with $N_{DV} = 9$, the following discrete velocities are used,

$$\begin{aligned} &\mathbf{e}_0/c = (0,0)^T \\ &\mathbf{e}_1/c = (1,0)^T \; ; \; \mathbf{e}_2/c = (0,1)^T \; ; \; \mathbf{e}_3/c = (-1,0)^T \; ; \; \mathbf{e}_4/c = (0,-1)^T \\ &\mathbf{e}_5/c = (1,1)^T \; ; \; \mathbf{e}_6/c = (-1,1)^T \; ; \; \mathbf{e}_7/c = (-1,-1)^T \; ; \; \mathbf{e}_8/c = (1,-1)^T \end{aligned} \tag{5}$$

where the weighting factors w_a ($a \in [0, 8]$) for the D2Q9 model are defined as,

$$w_0 = 4/9 \; ; \; w_{1-4} = 1/9 \; ; \; w_{5-8} = 1/36 \tag{6}$$

Based on the 'streaming' from one lattice point to a nearest neighbor lattice point, the evolution of f_a can be written as,

$$f_a\left(\mathbf{x}_i + \mathbf{e}_a \delta t, t + \delta t\right) - f_a\left(\mathbf{x}_i, t\right) = -\frac{\delta t}{\tau}\left[f_a\left(\mathbf{x}_i, t\right) - f_a^{eq}\left(\mathbf{x}_i, t\right)\right] \tag{7}$$

Fluid density and velocity components are related to the discretized distribution functions f_a and f_a^{eq} as follows,

$$\rho = \sum_{a=0}^{n_{DV}-1} f_a = \sum_{a=0}^{n_{DV}-1} f_a^{eq} \; ; \; \rho \mathbf{V} = \sum_{a=0}^{n_{DV}-1} \mathbf{e}_a f_a = \sum_{a=0}^{n_{DV}-1} \mathbf{e}_a f_a^{eq} \tag{8}$$

Lattice Boltzmann methods employ a stream-collide approach to update f_a from time t to $t + \delta t$ in two steps:

- The 'collision step' creates an intermediate update of f_a to f_a^{int} based on the collision term:

$$f_a^{int}\left(\mathbf{x}_i, t\right) = f_a\left(\mathbf{x}_i, t\right) - \frac{\delta t}{\tau}\left[f_a\left(\mathbf{x}_i, t\right) - f_a^{eq}\left(\mathbf{x}_i, t\right)\right] \tag{9}$$

- The convection on the left-hand side of Equation (7), is represented by the 'streaming step', such that based on intermediate update f_a^{int} the final update is computed as,

$$f_a\left(\mathbf{x}_i + \mathbf{e}_a \delta t, t + \delta t\right) = f_a^{int}\left(\mathbf{x}_i, t\right) \tag{10}$$

As a first step toward creating quantum algorithms for D2Q9 and D3Q27, reduced models are considered first. Two non-linear lattice models for one-dimensional flow problems are described in the next sections.

2.2. Modified D1Q3 Model for One-Dimensional Flow

In recent work by the author [12], the D1Q3 model for one-dimensional iso-thermal flow was re-formulated in terms of a D2Q4 model to facilitate quantum-circuit implementation. To summarize, in the original D1Q3 model, the direction vectors $e_i, i \in [0,2]$, density ρ and velocity u (by construction, as a function of f_2 and f_0 for this model) are defined as,

$$e_i = \begin{cases} -1 & \text{for} \quad i = 0 \\ 0 & \text{for} \quad i = 1 \\ +1 & \text{for} \quad i = 2 \end{cases} \; ; \; \rho = \sum_0^2 f_i \; ; \; u = f_2 - f_0 \tag{11}$$

with the collision terms defined as,

$$-\frac{dt}{\tau}\left[\begin{pmatrix} f_0 \\ f_1 \\ f_2 \end{pmatrix} - \begin{pmatrix} \frac{1}{2}\left[\frac{1}{3} - u + u^2\right] \\ \frac{2}{3} - u^2 \\ \frac{1}{2}\left[\frac{1}{3} + u + u^2\right] \end{pmatrix}\right] \tag{12}$$

For quantum-circuit implementation, an even number of discrete-velocity directions was preferred. This was achieved by replacing the original f_1 distribution function with two identical distribution function components, f_1 and f_2, in the modified model (with corresponding duplicate discrete velocity). The corresponding equilibrium distribution functions become,

$$\vec{f}^{eq} = \begin{pmatrix} \frac{1}{2}\left[\frac{1}{3} - u + u^2\right] \\ \frac{1}{2}\left[\frac{2}{3} - u^2\right] \\ \frac{1}{2}\left[\frac{2}{3} - u^2\right] \\ \frac{1}{2}\left[\frac{1}{3} + u + u^2\right] \end{pmatrix} \quad \Rightarrow \quad \vec{f}^{eq}(u=0) = \begin{pmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{6} \end{pmatrix} \tag{13}$$

To further facilitate quantum-circuit implementation, a re-scaled distribution function $\vec{g}$ was defined relative to the 'rest' state (here, 'rest' means $u = 0$) defined in Equation (13), such that the constant factors $1/3$ and $1/6$ are removed,

$$\vec{g} = \vec{f} - \begin{pmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{6} \end{pmatrix} \; ; \; \vec{g}^{eq} = \vec{f}^{eq} - \begin{pmatrix} \frac{1}{6} \\ \frac{1}{3} \\ \frac{1}{3} \\ \frac{1}{6} \end{pmatrix} = \begin{pmatrix} -\frac{u}{2} + \frac{u^2}{2} \\ -\frac{u^2}{2} \\ -\frac{u^2}{2} \\ \frac{u}{2} + \frac{u^2}{2} \end{pmatrix} \tag{14}$$

The density and velocity follow from distribution function components as,

$$\rho = 1 + \sum_0^3 g_i \quad ; \quad u = g_3 - g_0 \tag{15}$$

2.3. Non-Linear Lattice Model for One-Dimensional Burgers Equation

The Burgers equation has been widely used as a model equation in the development of CFD methods. It is a single partial differential equation (in contrast to a system for the Navier–Stokes) representing non-linear dynamics and viscous effects representative of fluid dynamics. Interestingly, the strongly non-linear Burgers equation can be solved using the Cole-Hopf transform as a linear heat equation. The current work considers the discretization of the Burgers equation as a step toward methods for the Navier–Stokes equations for which this type of transform cannot be performed, and therefore, the original,

non-linear form of the Burgers equation is maintained here. The quasi-1D Burgers equation can be written as,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{16}$$

where μ represents the viscosity of the considered medium. In the literature, a range of Lattice Boltzmann models can be found for the Burgers equation. Here, the D2Q4 model, as described by Velivelli and Bryden [16], is used, recast in a notation consistent with that for the modified D1Q3 model used here. Then, this D2Q4 model can be summarized as,

$$\begin{aligned} &e_0/c = (1,0)^T \;;\; e_1/c = (0,1)^T \;;\; e_2/c = (0,-1)^T \;;\; e_3/c = (-1,0)^T \\ &u/c = f_0 - f_3 = f_0^{eq} - f_3^{eq} \end{aligned} \tag{17}$$

with equilibrium distribution function defined as,

$$\begin{aligned} f_0^{eq} &= \frac{1}{4}\left[\frac{u}{c} + \frac{u^2}{c^2}\right] \;\;;\;\; f_1^{eq} = \frac{u}{4c} \\ f_2^{eq} &= \frac{u}{4c} \;\;;\;\; f_3^{eq} = \frac{1}{4}\left[-\frac{u}{c} + \frac{u^2}{c^2}\right] \end{aligned} \tag{18}$$

This highlights that with relatively small changes, i.e., mainly the difference in the 'rest-state' equilibrium state, the quantum algorithm developed here for the D1Q3 model (specifically, in a modified D1Q4 form to ease implementation as a quantum circuit) can be modified into that for the D2Q4 model for the quasi-1D Burgers equation.

3. Quantum Computing Principles

Key concepts and principles are briefly reviewed here. A more detailed introduction is beyond the scope of the present work. Quantum algorithms are represented in the present work as quantum circuits. In the quantum-circuit diagrams used here, the qubits in the quantum register are represented in a vertical arrangement, where the solid horizontal lines represent the time progression. Traversing the quantum circuit from left to right, the quantum state of the qubit registers gets modified by a series of unitary gate operations acting on one or more qubits. For a quantum circuit, its width is defined by the number of qubits, while the horizontal extent, directly related to the number of successive gate operations, represents the circuit depth. The circuit depth that can be executed on a quantum computer is limited by the time the quantum register can stay in a coherent quantum state. The combination of limited feasible circuit depth and width, along with the limited connectivity of qubits in current and near-future quantum computers, formed the basis of the concept of quantum volume [17] to specify quantum hardware capability. For machines with limited quantum volume, a hybrid quantum-classical computing approach termed the Variational Quantum approach [18,19] has been widely used. Shallow, parameterized quantum circuits are used that are tightly coupled in terms of data exchange with classical hardware. The problem considered is typically an optimization problem where the classical computer is tasked with performing a parameter optimization problem in each of a number of iterations. These approaches have their origin in quantum chemistry, but a wider range of practically relevant problems can be tackled with optimization-type approaches [20]. Figure 1 shows an example of the Variational Quantum approach, where the required solution is constructed from a layered network. As shown, multiple layers are used (four in the illustration), each taking multiple qubits as input (six in the example shown). Using depth five in the example, the quantum circuits defined by $U(\lambda)$ involve 13 two-qubit gates. Each of these gates has a parameter $\lambda_i \in [1, 13]$ associated with it. A classical computer creates optimized parameters λ_i employing an iterative approach that takes the measured state of the ancilla qubit as input. A further key part of the approach is the problem-specific Quantum Non-linear Processing Unit (QNPU), shown on the right-hand side of Figure 1.

In the context of the present work, it is particularly interesting that Lubasch et al. [20] published an example of the QNPU for the non-linear Burgers equation. It is important to

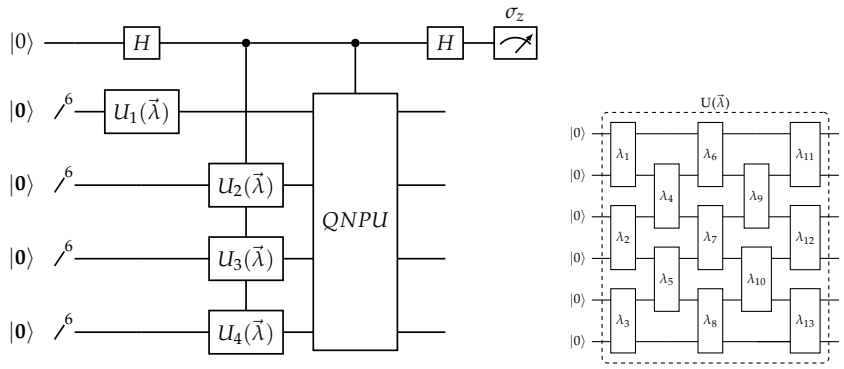


Figure 1. Illustration of the Variational Quantum Computing (VQC) approach, shown here for four layers, each with 6 input qubits and depth 5 for the operators $U(\vec{\lambda})$ with 13 variational parameters defining two-qubit gates. The Quantum Non-linear Processing Unit ($QNPU$) is designed to efficiently create non-linear products using problem-specific quantum circuits. The figure is adapted from Lubasch et al. (2020).

emphasize at this point the contrasting approach used in the present investigation, where quantum circuits with larger circuit depth are developed based on the assumption that more fault-tolerant quantum computers will be available in the near future.

3.1. Data Encoding in Quantum Algorithms

The majority of existing quantum algorithms were designed such that data are encoded in the amplitudes defining the quantum state of the qubit register. This is commonly referred to as amplitude-based encoding or encoding within the amplitudes of the quantum state. An important feature of this encoding technique is that it facilitates the quantum algorithm to benefit from quantum parallelism. Using n qubits, 2^n degrees of freedom are created, with the potential of an exponential saving in terms of memory when comparing the number of qubits and classical bits. Quantum parallelism also enables the simultaneous manipulation of 2^n amplitudes, defining the quantum state by performing a single unitary operator on an n-qubit coherent quantum register. An alternative data-encoding approach is usually referred to as quantum computational basis encoding or encoding within the computational basis of the quantum state. In this approach, the n coherent qubits in the register are used to represent data using a fixed-point representation. A quantum algorithm using this encoding as a first step initializes the quantum state by applying an oracle that transforms the qubit register from the initial zero-state to the fixed-point representation of the required input state, such that in contrast to amplitude-based encoding, only a single of the 2^n possible states has non-zero amplitude. Successive gate operations then effectively change which of the quantum states has the non-zero amplitude. Using this type of encoding, it is less clear than in amplitude-based encoding how quantum algorithms can take advantage of quantum parallelism. However, crucially, it facilitates non-linear operations in terms of arithmetic operations, as previously demonstrated in existing quantum algorithms for arithmetic operations. Encoding within the computational basis has not been widely used in applications related to scientific computing. A notable example is the quantum algorithm for computing the Fourier transform in the computational basis (termed QFTC) by Zhou et al. [21]. Research work covering efficient transformations between the two representation types has, so far, been very limited, e.g., the conversion from amplitude-encoding to computational basis encoding was investigated by Miterai et al. [22], while the reverse transformation was investigated by SaiToh [23]. From previous and ongoing work, it is clear that important and meaningful quantum algorithms using computational-basis encoding can be obtained despite quantum parallelism not being exploited directly as in an algorithm using amplitude-based encoding. However, when algorithms using quantum-basis encoding are used as part of a larger application it is expected that quantum speed-up can still be achieved, particularly if further improvements to the algorithms for

conversion between the two different encoding techniques can be made. The availability of efficient, updatable quantum memory further supports the potential of algorithms using computational-basis encoding.

3.2. Quantum Floating-Point Format

In previous work by the author [7,12], a floating point representation was proposed with reduced precision relative to the IEEE-754 single precision format. The motivation for this representation is the wide range of numbers that can be represented by the quantum-circuit implementations considered in an equivalent fixed-point representation. In the literature, few works deal with quantum circuits for floating-point arithmetic; e.g., Häner and co-workers [24] considered complexity for circuits with IEEE-754 type precision. In the context of quantum annealing, the floating-point division was considered by Rogers and Singleton [25]. The quantum floating-point format used here is explained and demonstrated in Appendix A. A key aspect is the use of an asymmetric bias. This enables the range of numbers that can accurately and reliably be presented to be tailored to the considered problem. For the D1Q3 model, variables represented in floating point format will be small (e.g., velocity components scaled by lattice speed need to be small due to the requirement that the flow speed is much smaller than the speed of sound), leading to a choice of a bias of eight for 3-qubit exponents (in contrast to three as symmetric bias). For the lattice model for the Burgers equation, the low-Mach number restriction is less strict than for the D1Q3 model. To illustrate the dependency of quantum circuits on the employed exponent bias, the floating-point representation used in the Burgers model has an exponent bias of five.

In the current work, computational basis encoding was selected along the quantum floating-point format to facilitate the evaluation of non-linear terms as quantum floating-point arithmetic.

3.3. Ripple-Carry Adders and Reduction-by-Specialization

The quantum circuits developed in this work include quantum full adder and quantum modulo adders. The modular structure of the ripple-carry quantum adder, as originally introduced by Cuccaro [26], forms the main motivation for using this type of quantum adder. A characteristic of the Cuccaro-type quantum adder is the use of majority (MAJ) and un-majority (UMA) operations acting on three qubits. The 'original' quantum circuit, as shown in Figure 2, illustrates the Cuccaro full adder for the addition of two 2-qubit strings. Specifically, $|a1|a0\rangle$ represents the input that, upon completion, is in the original state, while $|r2|r1|r0\rangle$ represents the 3-qubit output string, such that $|r2\rangle = |0\rangle$ before addition and $|r1|r0\rangle$ holds the second 2-qubit input. The left half of the quantum circuits details the two MAJ sub-circuits. The right half similarly shows the two UMA blocks. The $CNOT$ in the center is indicative of the considered full adder. Specifically, removing qubit $|r2\rangle$ and this $CNOT$ operation will transform the circuit into a modulo 2-qubit adder.

The remaining part of Figure 2 shows how the quantum circuit implementation of the 2-qubit full adder can be specialized for the specific cases of input $|a1|a0\rangle = |10\rangle$. Figure 2 shows the step-by-step reduction in the quantum circuit. A complicating factor is that in the Cuccaro-type adders, the qubit input string that is left unchanged at the completion of the addition can have its qubits temporarily change state. This is clear from the 'Step 1' circuit, where two Toffoli gates act with $|a1\rangle$ as a target. In 'Step 2', the NOT gates surrounding the two Toffoli gates are accounted for by changing the conditionals in the Toffoli gates. Then, the actions of qubit $|a1\rangle$ can be removed, as shown in 'Step 3' in Figure 2. Finally, the specific choice of $|a0\rangle = |0\rangle$ is accounted for by first eliminating the $CNOT$s with $|a0\rangle$ as the control qubit. The fact that $|c\rangle = |0\rangle$ will then be maintained in the circuit enables further removal of gate operations. This results in the final state, shown as 'Step 4' in Figure 2. This type of reduction of quantum circuits for specific input qubit states is employed in the present investigation for two purposes. First, in the proposed quantum floating-point arithmetic, specialized adders are often used to apply increments to exponents, as discussed later. The second type of application is in circuit transformations used

to facilitate the evaluation of the quantum circuits using quantum computing simulators on classical computers by reducing the width of the quantum circuit.

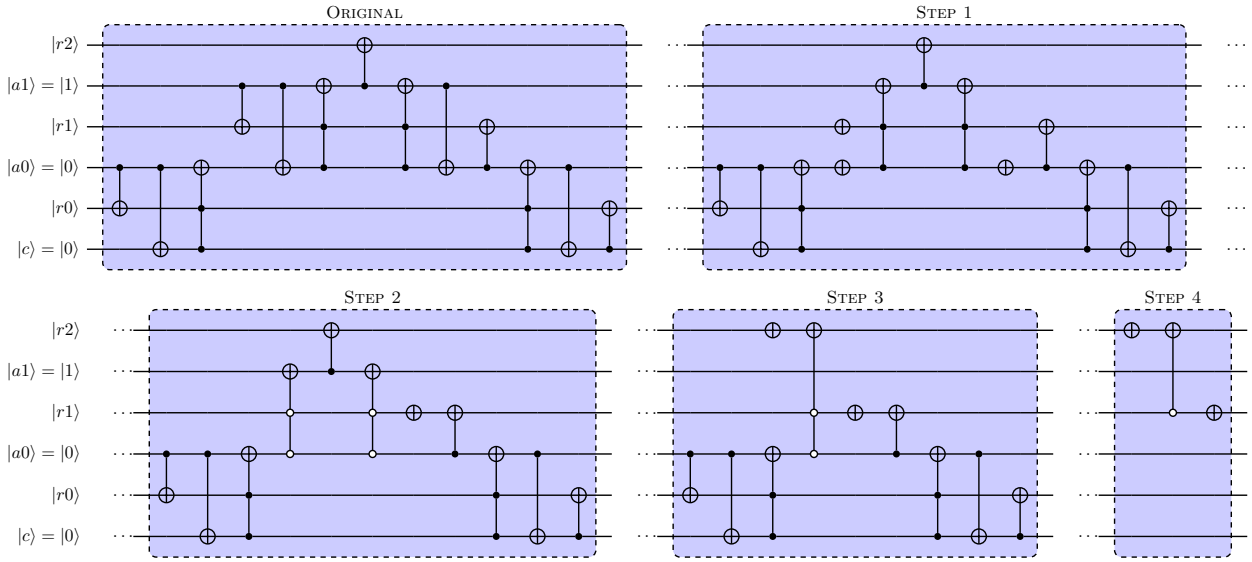


Figure 2. Step-by-step specialization of the 2-qubit full adder. The example shown adds $|a1|a0\rangle = |10\rangle$ into 3-qubit register $|r2|r1|r0\rangle$.

4. Quantum-Circuit Implementation of Modified D1Q3 LBM

In previous work by the author [12], the quantum-circuit implementation of the modified D1Q3 model was analyzed in detail. In that work, a design aimed at minimizing the quantum-circuit width was used, while the main focus of that work was the evaluation and simulation of the circuit using a quantum computing simulator with machine acceleration. In contrast, the design for the modified D1Q3 considered here employs a small number of additional qubits but benefits from a significantly reduced circuit depth. This design trade-off can be summarized as follows:

- The design considered previously [12] ('Design 1') converts the output of the squaring of the u-velocity mantissa (temporarily) in quantum floating-point format, using $N_M - 1 + N_E$ qubits (with 'hidden-qubit' approach);
- In the current design ('Design2'), the output of the squaring of the u-velocity mantissa defined by $2N_M$ qubits is maintained for use in successive steps of the quantum circuit;
- Design 1 requires the squaring and un-computation of squaring to be performed twice, in contrast to Design 2, that only involves a single squaring operation of the u-velocity mantissa. The $2N_M$ qubits defining the squaring are returned to their initial state by a single un-computation of the squaring operation.

A detailed complexity analysis in terms of qubits required shows that although the difference between $N_M - 1 + N_E$ (for Design 1) and $2N_M$ (for Design 2) is needed to store the result of the squared u-velocity, mantissa appears to grow as $N_M - N_E$, and therefore, favors Design 1 for small N_E and increasing N_M. However, Design 1 also needs a sufficient workspace to perform u-velocity squaring, such that for $N_M = 4$ and $N_E = 3$, the difference is reduced to a single-qubit advantage for Design 1. The significant reduction in gate operations needed for un-computation for Design 2 motivates the use of this design approach here. The approach used in Design 2 is also employed for the non-linear lattice model for the one-dimensional Burgers equation in the next section.

In the quantum-circuit implementation, the following qubits are used for $N_M = 4$ and $N_E = 3$:

$$\begin{aligned}
|dv1|dv0\rangle &: \text{two qubits defining the index of discrete velocity}\\
|eu2|eu1|eu0\rangle &: N_E\text{-qbit representation of the exponent of } u\\
|su\rangle &: \text{sign bit for } u\text{-velocity—here a positive value is assumed}\\
|mu2|mu1|mu0\rangle &: (N_M-1)\text{-qbit representation of mantissa of } u\\
|eg2|eg1|eg0\rangle &: N_E\text{-qbit representation of the exponent of } g^{eq}\\
|sg\rangle &: \text{sign bit for the considered component of } g^{eq}\\
|mg2|mg1|mg0\rangle &: (N_M-1)\text{-qbit representation of the mantissa of } g^{eq}\\
|cut\rangle &: \text{qubit-defining truncation to 0 if in state } |1\rangle
\end{aligned}$$

Furthermore, additional qubits ('workspace') are required to perform calculation:

$$\begin{aligned}
|r7|r8|\ldots|r1|r0\rangle &: 2N_M \text{ qubits to store the squared } u \text{ mantissa}\\
|qu3|qu2|qu1|q0\rangle &: N_M \text{ qubits to store a temporary copy of the } u \text{ mantissa}\\
|c\rangle &: \text{'carry' qubit-set here to } |0\rangle\\
|anc\rangle &: \text{ancilla qubit-initialized at } |0\rangle\\
|cr6|cr5|\ldots|cr1|cr0\rangle &: 2N_M-1 \text{ qubits to temporarily store the squaring result}
\end{aligned}$$

Starting from the left-hand side of Figure 3, the steps shown can be summarized as follows:

- Qubit $|cut\rangle$ initialized at $|0\rangle$ is set to $|1\rangle$ in case u^2 gets truncated to 0. For employed quantum floating-point format (exponent bias equals to 8), this occurs for sub-normal u velocity, i.e., $|eu2|eu1|eu0\rangle = |000\rangle$ as well as for velocities with $|eu2|eu1|eu0\rangle = |001\rangle$ or $|eu2|eu1|eu0\rangle = |010\rangle$;
- For $|cut\rangle = |0\rangle$, the mantissa qubits of u get temporarily copied into the workspace qubits in preparation for the squaring operation (termed $SQ4$ here for $N_M = 4$ considered). Once $2N_M$-qubits defining square of u mantissa is set, the temporary copies of u mantissa qubits are removed;
- For the two 'rest-state' discrete-velocity directions (defined by $|dv1|dv0\rangle = |01\rangle$ and $|dv1|dv0\rangle = |10\rangle$) the final state of the qubits defining the equilibrium function in floating-point format can directly be defined from velocity u. This operation is performed at $0110r8$ in the circuit;
- For the discrete velocity pointing left ($|dv1|dv0\rangle = |00\rangle$) and for the discrete velocity pointing right ($|dv1|dv0\rangle = |11\rangle$), the equilibrium distribution function combines a term $\pm u/2$ and $u^2/2$. In the quantum circuit, the operation shown as $0011a2$ prepares this summation for which a 5-qubit modulo adder ($MADD5$) is used for $N_M = 4$ by employing the $2N_M$-qubit mantissa squaring output with u, while introducing shift accounting for the exponents of u and u^2 terms;
- Once the 5-qubit modulo adder has performed the required addition for the two discrete-velocity directions defined by $|dv1|dv0\rangle = |00\rangle$ and $|dv1|dv0\rangle = |11\rangle$, operation $0011b2$ uses this output to define the equilibrium distribution function for these directions in quantum floating-point format.
- Operation $UMA5$ (un-computation of $MADD5$) and subsequent steps shown in Figure 4 un-compute the previously computed u^2 to return the qubits to the bottom half of the circuit (workspace) to their original $|0\rangle$ state. As part of this process, $ISQ4$ un-computed the squaring of the u mantissa.

The squaring of the u-velocity mantissa is based on the shift-and-add principle. Figure 5 shows the controlled 4-qubit full additions ($FA4$) for $N_M = 4$, creating the square with the application of a, output-register shift ($Sh4$) between successive controlled additions. The quantum circuit implementation of the un-computation with $UFA4$ representing

the un-computation of the 4-qubit full additions with reverse shifts ($\hat{Sh}4$) is also shown between successive steps.

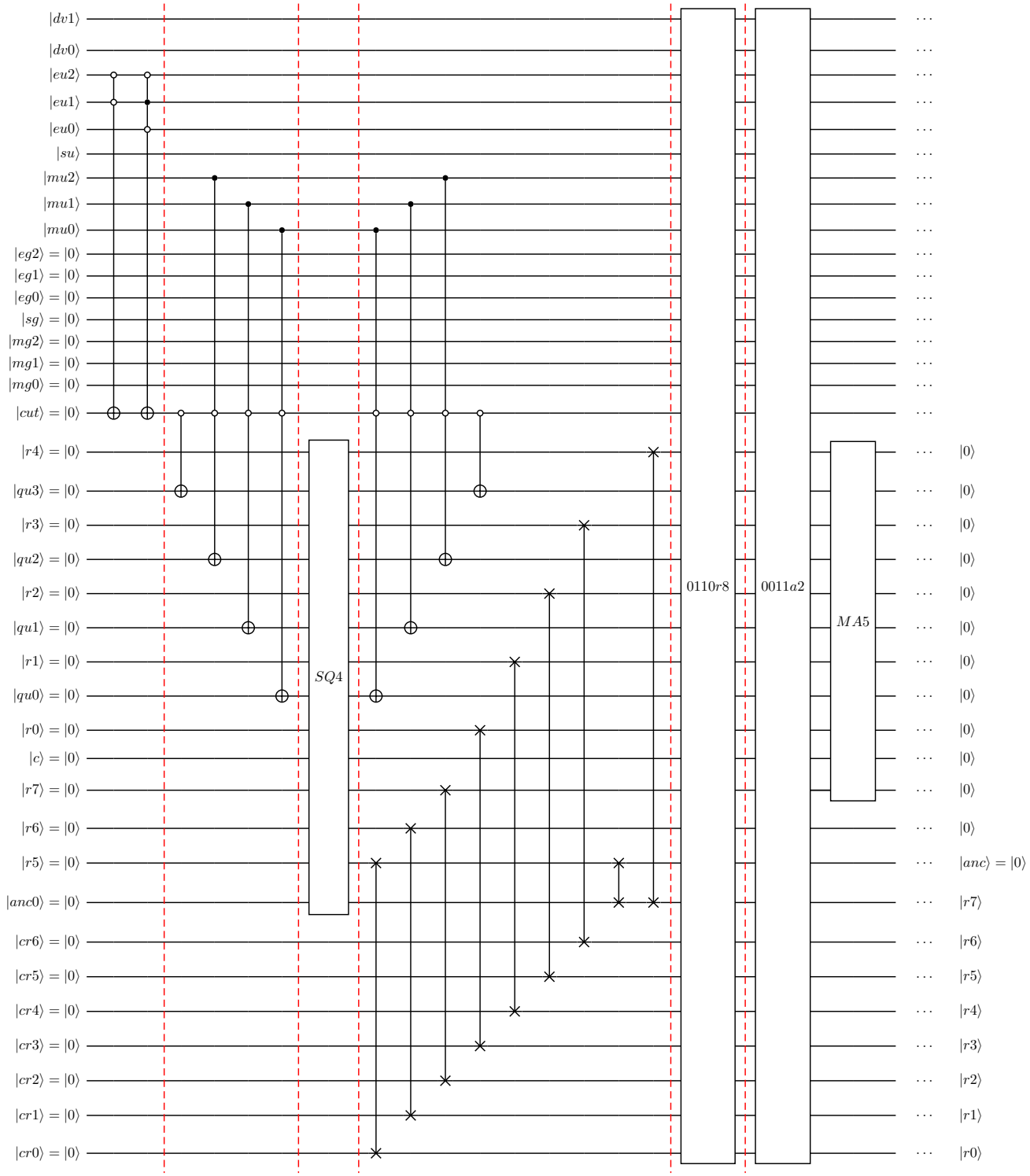


Figure 3. Modified D1Q3 model: quantum circuit design for computing $\bar{g}^{eq}$. Floating-point representation uses four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$), with an exponent bias set to eight. Part 1.

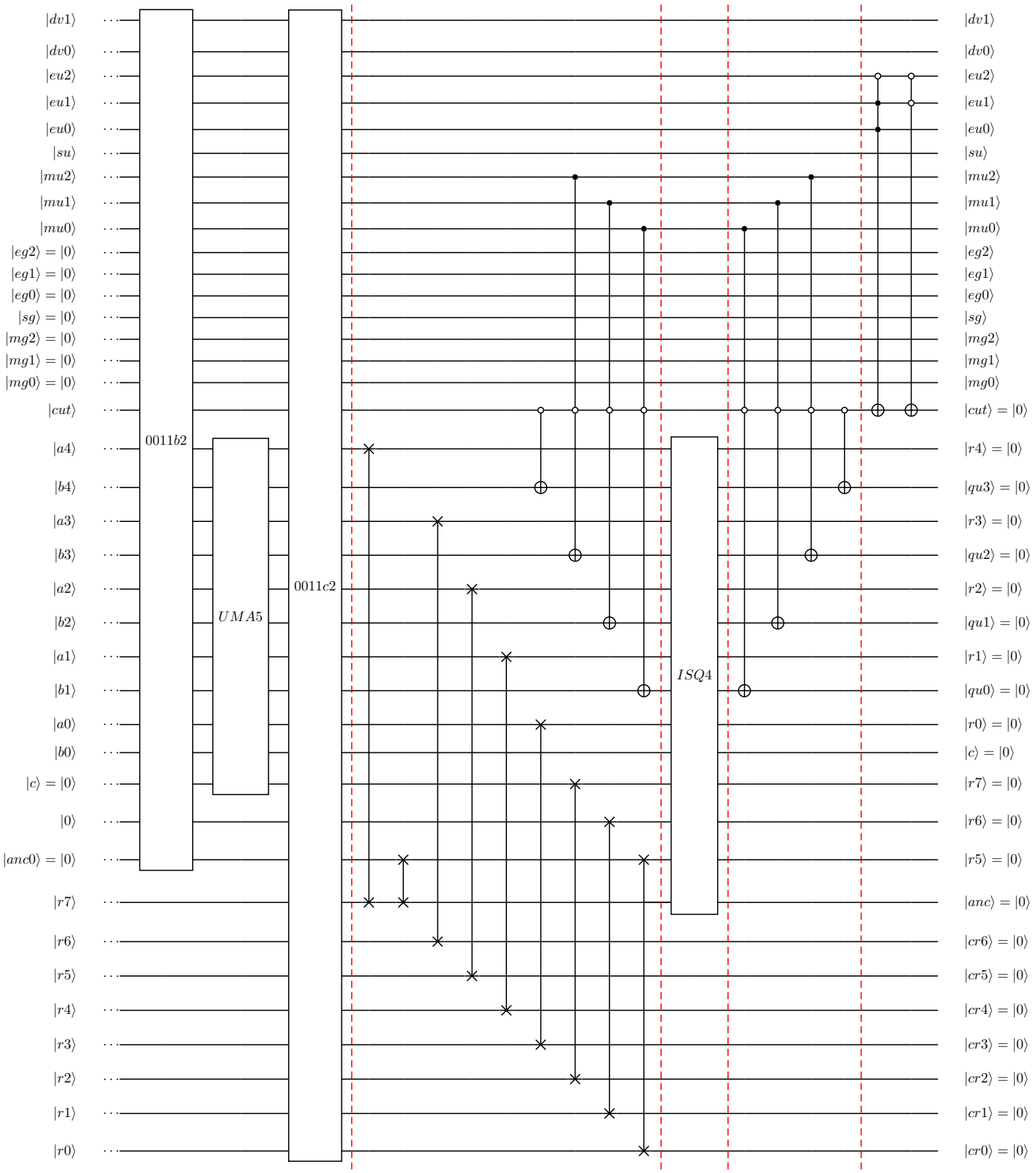


Figure 4. Modified D1Q3 model: quantum circuit design for computing $\vec{g}^{eq}$. Floating-point representation uses four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$), with an exponent bias set to eight. Part 2.

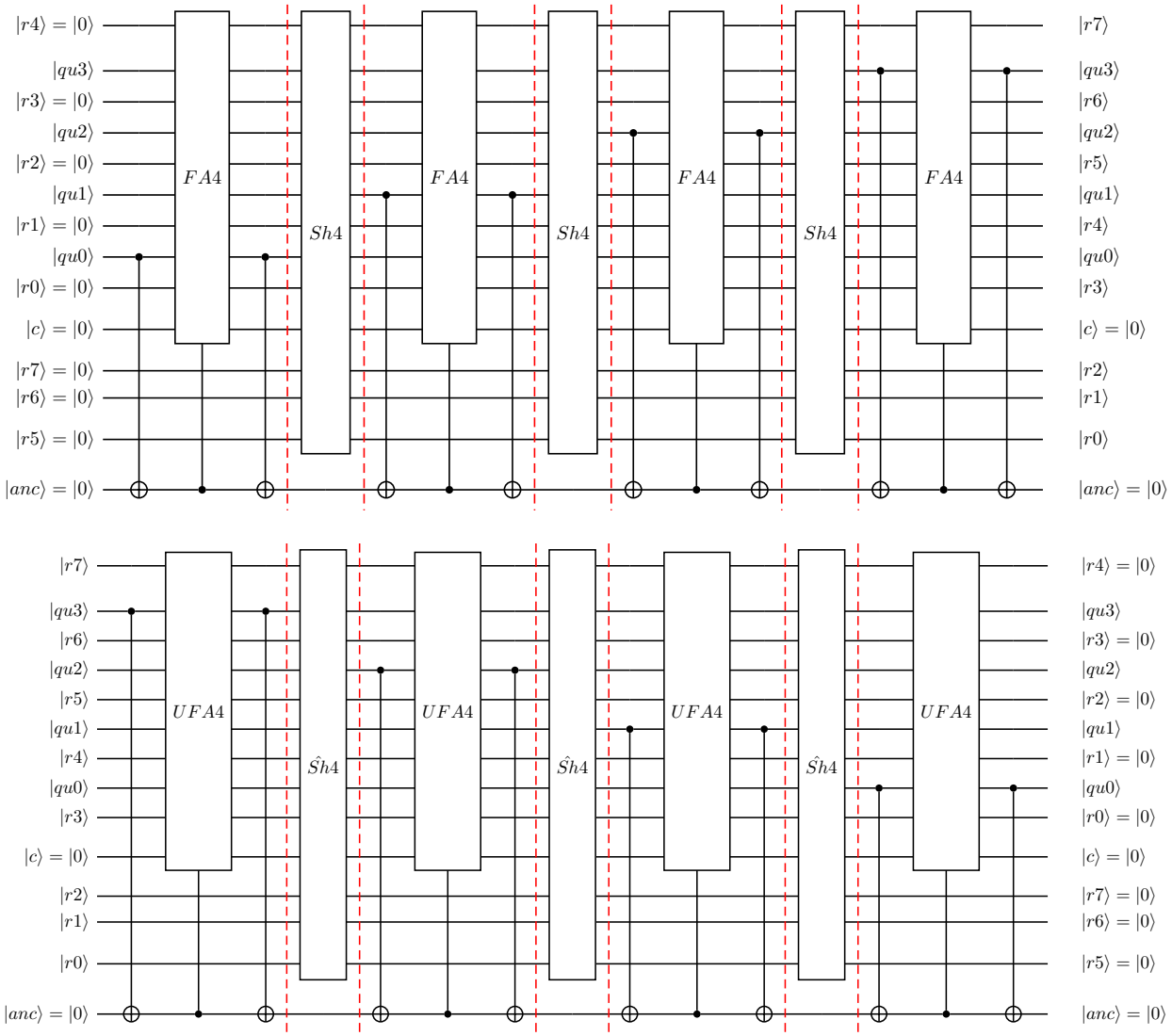


Figure 5. Quantum-circuit implementation of $SQ4$ operation (and its un-computation $ISQ4$) as used in the modified D1Q3 model and in the non-linear lattice model for 1D Burgers equations.

5. Quantum-Circuit Implementation for One-Dimensional Burgers Equation

The quantum-circuit implementation of the non-linear lattice model for the one-dimensional Burgers equation is investigated in this section. In particular, the quantum floating-point-based implementation of its non-linear equilibrium function is employed in the collision step. For $N_M = 4$ and $N_E = 3$ (using an exponent bias of 5), Figures 6 and 7 show the overall design of this quantum-circuit implementation.

In the quantum-circuit implementation, the number of qubits needed is identical to that demonstrated for the modified D1Q3 model in the previous section for the same choice of N_M and N_E. For convenience, the qubits associated with storing the equilibrium distribution function $\vec{f}^{eq}$ are now named with f instead of g. Specifically, here the qubits are named:

$$\begin{aligned} |ef2|ef1|ef0\rangle &: N_E\text{-qbit representation of the exponent of } f^{eq} \\ |sf\rangle &: \text{sign bit for considered the component of } f^{eq} \\ |mf2|mf1|mf0\rangle &: (N_M - 1)\text{-qbit representation of mantissa of } f^{eq} \end{aligned}$$

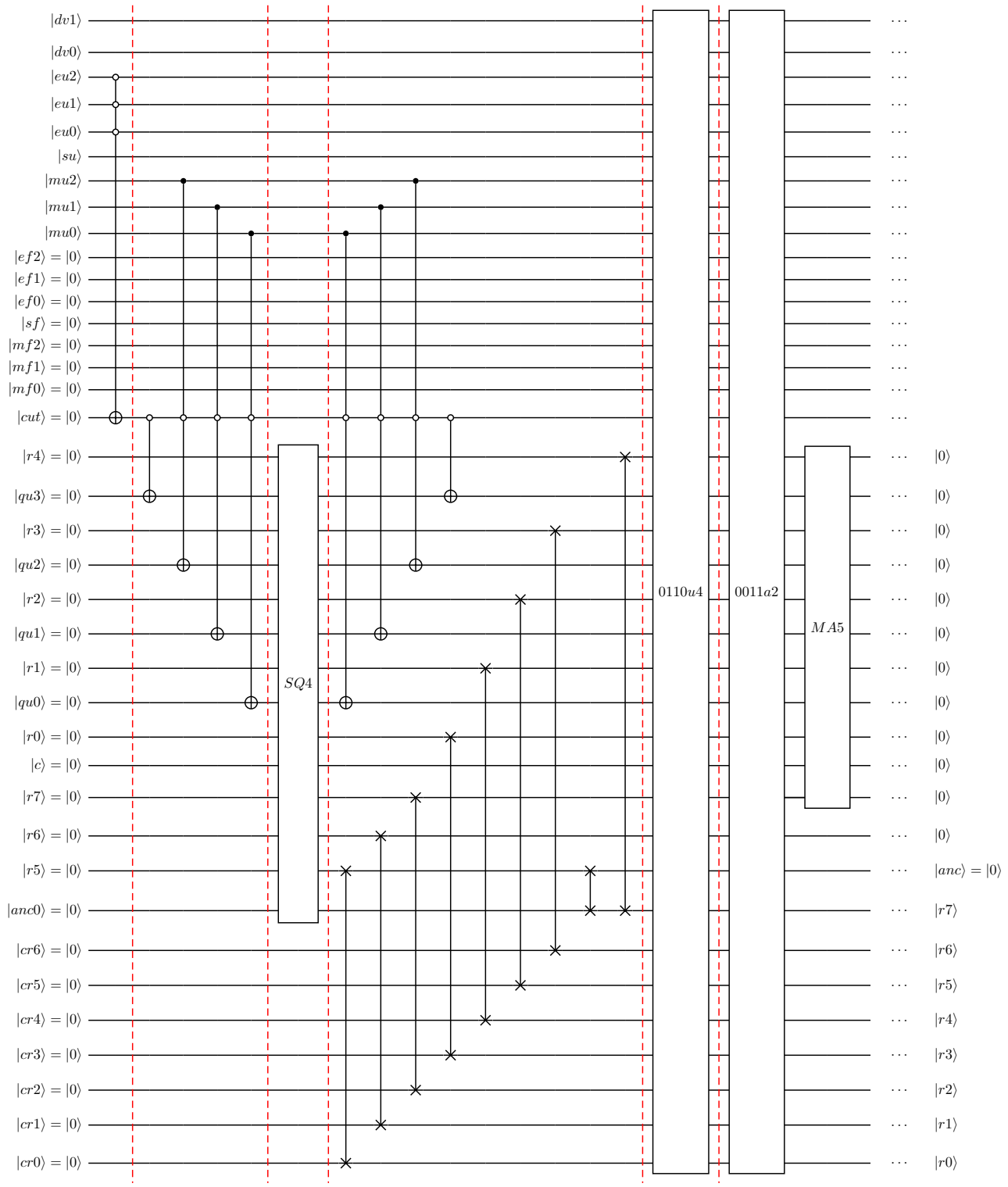


Figure 6. Quasi-1D Burgers: quantum circuit design for computing $\vec{f}^{eq}$ in the lattice model. The floating-point representation uses four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$), with an exponent bias set to five. Part 1.

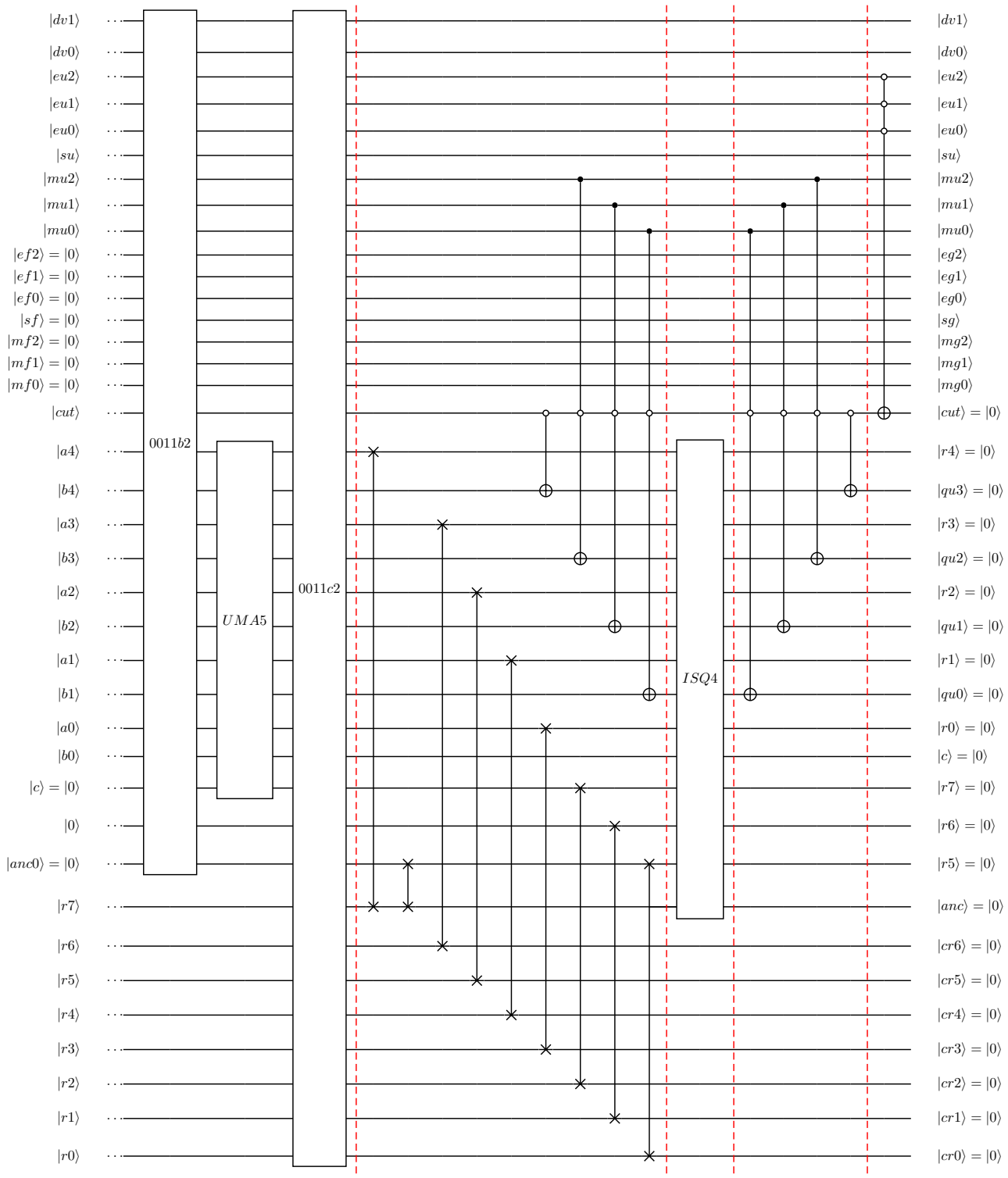


Figure 7. Quasi-1D Burgers: quantum circuit design for the evaluation of $\vec{f}^{eq}$ in the lattice model. The floating-point representation uses four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$), with an exponent bias set to five. Part 2.

Starting from the left-hand side of Figure 6, the steps shown can be summarized as follows:

- Qubit $|cut\rangle$ initialized at $|0\rangle$ is set to $|1\rangle$ in case u^2 gets truncated to 0. For the employed quantum floating-point format (with exponent bias equal to five in contrast

to eight used for modified D1Q3 model), this occurs for sub-normal u velocity, i.e., $|eu2|eu1|eu0\rangle = |000\rangle$;
- For $|cut\rangle = |0\rangle$, the mantissa qubits of u get temporarily copied into the workspace qubits in preparation of the squaring operation (termed $SQ4$ here for $N_M = 4$ considered). Once $2N_M$-qubits defining the square of u mantissa is set, the temporary copies of u mantissa qubits are removed;
- For the two discrete-velocity directions defined by $|dv1|dv0\rangle = |01\rangle$ and $|dv1|dv0\rangle = |10\rangle$, the final state of the qubits defining the equilibrium function in floating-point format can directly be defined from velocity u. This operation is performed at $0110u4$ in the circuit;
- For the two discrete-velocity directions defined by $|dv1|dv0\rangle = |00\rangle$ and $|dv1|dv0\rangle = |11\rangle$, the equilibrium distribution function combines terms $\pm u/4$ and $u^2/4$. In the quantum circuit, the operation shown as $0011a2$ prepares this summation for which a 5-qubit modulo adder ($MADD5$) is used for $N_M = 4$ by employing the $2N_M$-qubit mantissa squaring output with u, while introducing shift accounting for the exponents of u and u^2 terms;
- Once the 5-qubit modulo adder has performed the required addition for the two discrete-velocity directions defined by $|dv1|dv0\rangle = |00\rangle$ and $|dv1|dv0\rangle = |11\rangle$, operation $0011b2$ uses this output to define the equilibrium distribution function for these directions in the quantum floating-point format.
- Operation $UMA5$ (un-computation of $MADD5$) and the subsequent steps shown in Figure 7 un-compute the previously computed u^2 to return the qubits in the bottom half of the circuit (workspace) to their original $|0\rangle$ state. As part of this process, $ISQ4$ un-computed the squaring of the u mantissa.

The similarity of this quantum circuit design with that of the modified $D1Q3$ non-linear lattice model is apparent.

6. Extension to Two- and Three-Dimensional Lattice Models

In the two non-linear lattice models considered so far, i.e., the modified D1Q3 model as well as the D2Q4 model for the quasi one-dimensional Burgers equation, the non-linearity in the equilibrium distribution used in collision term was of the form u^2. LBM models for flow problems in two- or three-dimensional space also contain squared-velocity terms in the respective equilibrium distribution functions. As shown previously in Equation (4), the equilibrium distribution function for 2D and 3D LBM models can generally be written as,

$$f_a^{eq} = \rho w_a \Big[1 + \frac{3}{c^2}\mathbf{e}_a \cdot \mathbf{V} + \frac{9}{2c^4}\left(\mathbf{e}_a \cdot \mathbf{V}\right)^2 - \frac{3}{2c^2}\mathbf{V} \cdot \mathbf{V}\Big] \tag{19}$$

with $a \in [0, 8]$ for the D2Q9 model and $a \in [0, 26]$ for the D3Q27 LBM model. This means that for each of the discrete-velocity directions, a term $-3(u^2 + v^2)/2c^2$ will result in 2D models, and similarly, $-3(u^2 + v^2 + w^2)/2c^2$, for 3D LBM models. In the present work, the extension to multi-dimensional non-linear problems is considered with a particular focus on constructing the non-linear equilibrium distribution function:

- Floating-point arithmetic is used to represent the squared-velocity terms in the equilibrium distribution function;
- As a first step, the focus is on computing $u^2 + v^2$ for two-dimensional problems, where u and v represent the Cartesian velocity components in the $x-$ and $y-$directions scaled by a suitable reference velocity (e.g., lattice velocity in LBM);
- The construction of quantum circuits for the 'full' equilibrium distribution function and the resulting collision term is left for future publications;
- A further motivation for the current focus on evaluating $u^2 + v^2$ is the obvious and direct connection to evaluating kinetic energy in a wider range of lattice models beyond the LBM.

As mentioned in Section 1, the quantum circuits developed are aimed at near-future quantum computers with a relatively small number of 'logical' qubits with a level of fault tolerance facilitating the considered circuit depths. A well-considered trade-off between circuit width and circuit depth is, therefore, essential and will be discussed in the next section.

6.1. Design Considerations for Quantum-Circuit Implementation of $u^2 + v^2$ Calculation

To achieve a term of the form $u^2 + v^2$ in the floating-point format, a number of different approaches can be used. The different approaches considered in the present work are:

- Design *Ekin2D1*, based on the sequential computation of u^2 and v^2 with each term temporarily stored in the quantum floating-point format. This is followed by a floating-point addition to obtain $u^2 + v2$. From a classical computational point of view, this appears to be the most obvious design approach. Figure 8 shows the overall layout of the quantum circuit based on this design. As before, $SQ4$ and $ISQ4$ represent the mantissa squaring operation and its reverse. CC_u2 and CC_v2 are used to define u^2 and v^2 in the floating-point format starting from the $2N_M$ output of squaring. To facilitate the re-use of workspace qubits, the mantissa qubits of u and v are temporarily swapped by SW_{uv}. Based on the difference in the exponent of u^2 and v^2 (evaluated by $DEsq$—result temporarily stored in qubits defining the exponent of v^2), PS prepares the floating-point addition of u^2 and v^2, with $FA4$ and $UFA4$ representing 4-qubit full adder and its reverse. $UDEsq$ returns exponent v^2 to the state before $DEsq$. Not shown here for brevity are the subsequent steps needed to un-compute the calculations of u^2 and v^2 needed to return the qubits involved back to the initial state $|0\rangle$;
- Design *Ekin2D2*, based on the sequential computation of u^2 and v^2 with temporary storage using two separate $2N_M$-qubit registers of the mantissa-squaring operations for u^2 and v^2. Based on these two registers, as well as the exponents of u and v (and their difference), the summation $u^2 + v^2$ in the floating-point format can be performed. The motivation for this design is that compared to Design Ekin2D1, the complexity involved in the setting results in quantum floating-point format can potentially be reduced, as found previously for the 1D lattice implementations. Figure 9 shows the overall layout of the quantum circuit based on this design. Compared to *EKin2D1*, the summation of u^2 and v^2 now uses the difference in the exponent of u- and v-velocity since exponents of u^2 and v^2 are no longer formed explicitly. A key further observation that can be made is the reduction in un-computation steps needed up to the point of setting the desired output. The gate operations to the right of UDE_{uv} represent the first steps of the un-computation used to 'clear' the qubit registers defining the mantissa-squaring output for u^2 and v^2;
- Design *Ekin2D3*, where the term $u^2 + v^2$ is constructed using shift-and-add-based squaring involving both u and v directly. Clearly, the design needs to account for the possibility of u and v having different exponents. The main motivation for this design is that compared to Designs Ekin2D1 and EKin2D2, it can be expected that the quantum-circuit width can be reduced by eliminating the temporary storage of the terms u^2 and v^2 separately. However, the shift-and-add-based squaring of the mantissa will be more complex, as detailed here. A further motivation relates to the potential reduction in rounding and truncation error in defining $u^2 + v^2$. This aspect is considered in more detail in Section 7.

Based on the considerations detailed, as well as the circuit-width analysis shown in the next section, design *EKin2D3* was selected for further detailed analysis. Its quantum circuit implementation is described in Sections 6.3, 7 and 8.

6.2. Analysis of Circuit Width for Different Designs

In required qubits terms, all three designs need at least the following qubits to store input and output data for $N_M = 4$ and $N_E = 3$:

$$
\begin{aligned}
|eu2|eu1|eu0\rangle &: N_E\text{-qbit representation of the exponent of } u \\
|mu2|mu1|mu0\rangle &: (N_M - 1)\text{-qbit representation of the mantissa of } u \\
|ev2|ev1|ev0\rangle &: N_E\text{-qbit representation of the exponent of } v \\
|mv2|mv1|mv0\rangle &: (N_M - 1)\text{-qbit representation of the mantissa of } v \\
|ek2|ek1|ek0\rangle &: N_E\text{-qbit representation of the exponent of kinetic energy} \\
|mk2|mk1|mk0\rangle &: (N_M - 1)\text{-qbit representation of the mantissa of kinetic energy}
\end{aligned}
$$

This means that, more generally, $3N_E + 3(N_M - 1)$ qubits represent the absolute minimum circuit width. In the following, the number of additional qubits required for the different designs is detailed.

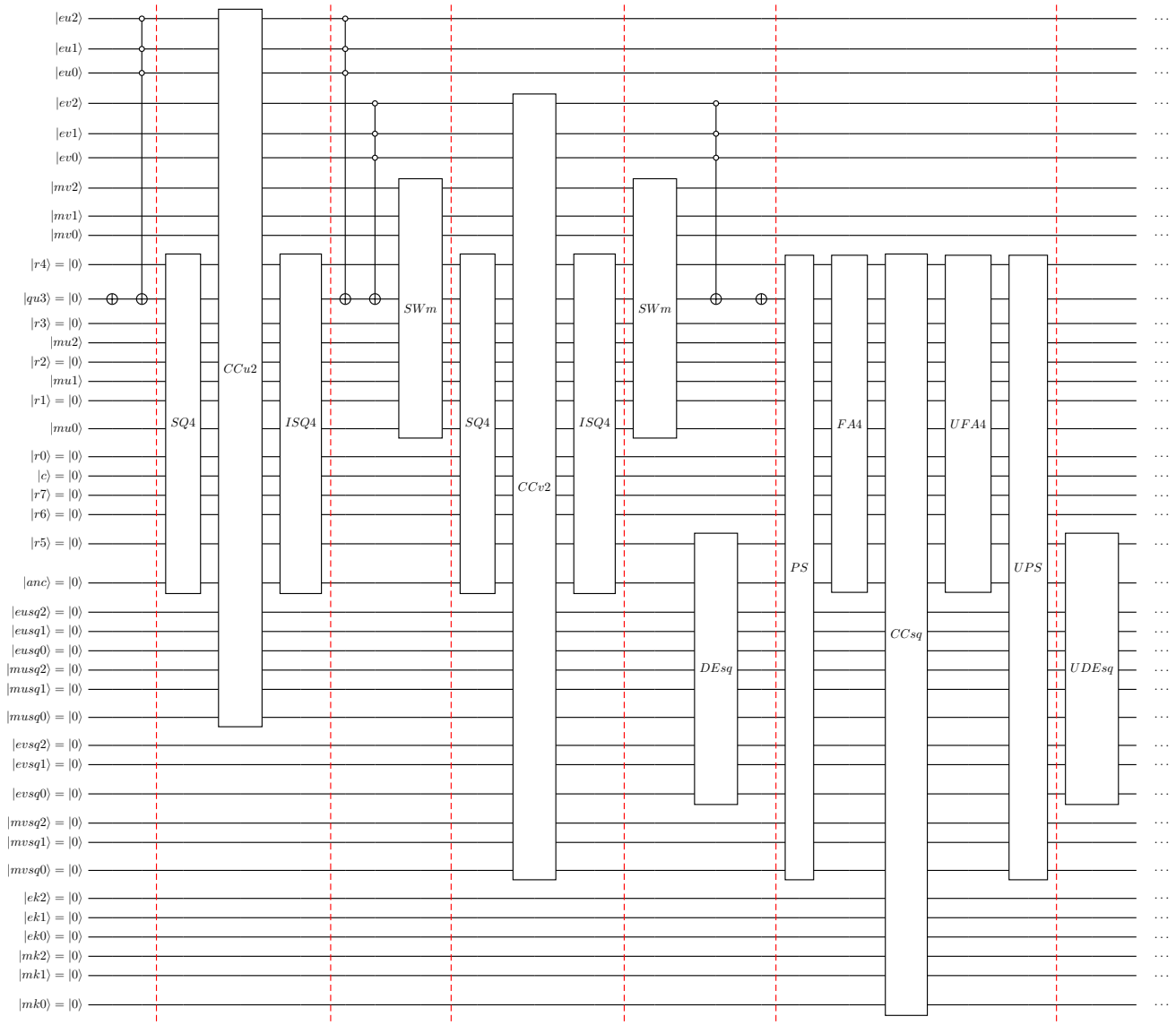


Figure 8. Quantum circuit design *Ekin2D1* for the computation of $u^2 + v^2$. The floating-point representation uses four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$).

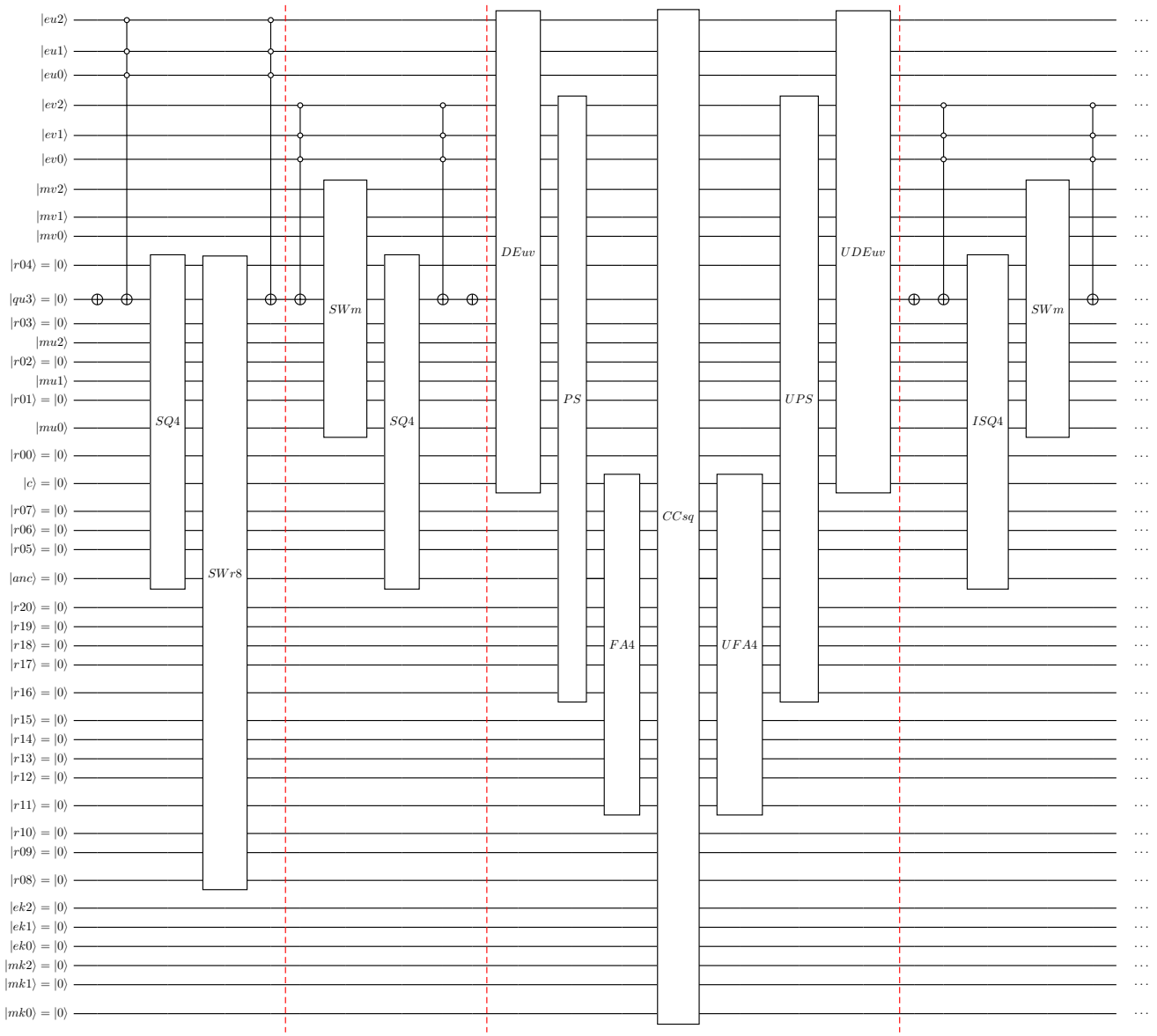


Figure 9. Quantum circuit design *Ekin2D2* for computation of $u^2 + v^2$. The floating-point representation uses four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$).

6.2.1. Design *Ekin2D1*

For Design *Ekin2D1*, the following additional qubits are needed:

$$
\begin{aligned}
|qu3\rangle &: \text{'hidden' mantissa qubit for } u \text{ and } v \text{ for } N_M = 4\\
|r7|\ldots|r0\rangle &: 2N_M\text{-qubit register for mantissa squaring output}\\
|c\rangle &: \text{'carry' qubit—initialized as } |0\rangle\\
|anc\rangle &: \text{ancilla qubit—initialized as } |0\rangle\\
|eusq2|eusq1|eusq0\rangle &: N_E\text{-qbit representation of the exponent of } u^2\\
|musq2|musq1|musq0\rangle &: (N_M-1)\text{-qbit representation of the mantissa of } u^2\\
|evsq2|evsq1|evsq0\rangle &: N_E\text{-qbit representation of the exponent of } v^2\\
|mvsq2|mvsq1|mvsq0\rangle &: (N_M-1)\text{-qbit representation of the mantissa of } v^2
\end{aligned}
$$

Here it is assumed that a number of space-saving measures are used. First, the temporary swapping of u and v mantissa qubits so the same set-up for mantissa squaring of u can be re-used for v. Secondly, the difference in u^2 and v^2 exponents, temporarily needed in working out the quantum floating-point addition, can be stored in $|evsq2|evsq1|evsq0\rangle$ and that after un-computation of this difference, the original exponent of v^2 is restored. Lastly, it can be assumed that the $2N_M$-qubit register $|r7|\ldots|r0\rangle$ can be re-used to perform the addition of the (re-normalized) mantissa qubit-strings of u^2 and v^2. Therefore, the additional qubit count amounts to the total,

$$1 + 2N_M + 2 + 2N_E + 2(N_M - 1) = 1 + 4N_M + 2N_E$$

6.2.2. Design *Ekin2D2*

For Design *Ekin2D2*, the following additional qubits are needed:

$\|qu3\rangle$	:	'hidden' mantissa qubit for u and v for $N_M = 4$
$\|r07\|\ldots\|r00\rangle$	:	$2N_M$-qubit register for mantissa squaring output—u^2
$\|r15\|\ldots\|r08\rangle$	:	$2N_M$-qubit register for mantissa squaring output—v^2
$\|c\rangle$	:	'carry' qubit—initialized as $\|0\rangle$
$\|anc\rangle$	:	ancilla qubit—initialized as $\|0\rangle$
$\|r24\|\ldots\|r16\rangle$	:	$2N_M + 1$-qubit workspace to perform floating-point addition of re-normalized mantissa qubit strings for u^2 and v^2

Here it is assumed that a number of space-saving measures are used. First, the temporary swapping of u and v mantissa qubits so the same set-up for mantissa squaring of u can be re-used for v. Secondly, the difference in u and v exponents, temporarily needed in working out the quantum floating-point addition, can be temporarily stored in $|ev2|ev1|ev0\rangle$ and that after un-computation of this difference, the original exponent of v is restored. Lastly, it is assumed that starting from the two $2N_M$-qubit mantissa-squaring result registers, the exponent difference for u and v, stored temporarily in $|ev2|ev1|ev0\rangle$, can be used to set up two N_M input qubit strings representing re-normalized mantissa qubits of u^2 and v^2 into N_M-qubit full adder, re-using $|c\rangle$ as the 'carry' qubit. Therefore, for this design, the total number of additional qubits is

$$1 + 2N_M + 2N_M + 2 + 2N_M + 1 = 6N_M + 3$$

since, for practical applications, N_M needs to be significantly greater than N_E, it follows that Design *Ekin2D2* has greater circuit width than Design *Ekin2D1*.

6.2.3. Design *Ekin2D3*

For Design *Ekin2D3*, the minimum set of additional qubits can be summarized as,

$\|qu3\|qu2\|qu1\|q0\rangle$	:	'Copy' of mantissa qubits for u and v for $N_M = 4$ (including 'hidden' qubit)
$\|r08\|\ldots\|r00\rangle$	:	$2N_M + 1$-qubit register for mantissa squaring output—$u^2 + v^2$
$\|c\rangle$	:	'carry' qubit—initialized as $\|0\rangle$
$\|anc\rangle$	:	ancilla qubit—initialized as $\|0\rangle$

Here, it should be noted that $2N_M + 1$ is the minimum qubit register size for the combined $u^2 + v^2$ mantissa squaring. For cases with a significant difference in u- and v-exponents, rounding and truncation effects can be reduced by extending this register further. This is discussed in more detail in Section 7. As before, it is assumed that temporarily, the difference in u and v exponents can be stored in $|ev2|ev1|ev0\rangle$, followed by un-computation

to restore the value of the v-exponent. Therefore, the minimum number of additional qubits for Design *Ekin2D3* becomes

$$N_M + 2N_M + 1 + 2 = 3N_M + 3$$

This means that Design *Ekin2D3* can be used to create quantum-circuit implementations with a minimum number of qubits among the three designs considered here. Even if the qubit register size for the combined $u^2 + v^2$ mantissa squaring output is chosen to be larger than $2N_M + 1$. For $N_E = 3$, designs with $2N_M + 3$ and $2N_M + 5$ output registers are analyzed in detail in Section 7. The analysis shows that for $N_M = 4$, extending the output-register size beyond $2N_M + 5$ gives no further reduction in rounding and truncation when setting the final result in the floating-point format.

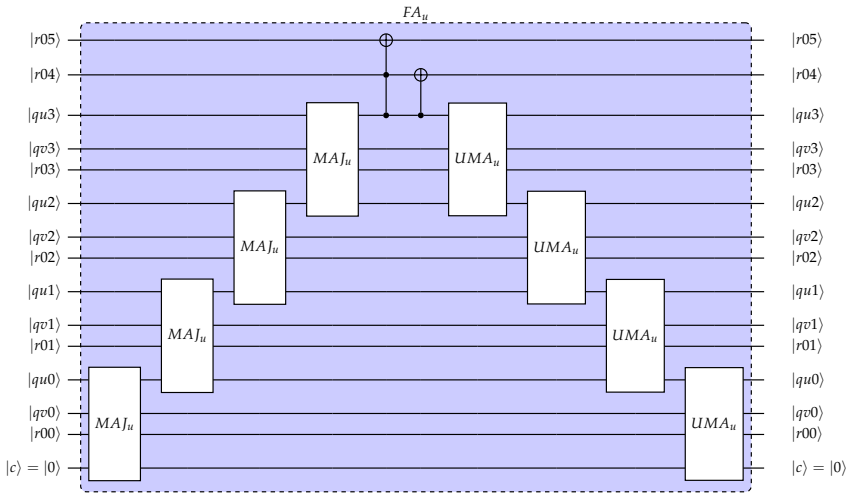
6.3. Modified Full Adders for Combined $u^2 + v^2$ Evaluation

For the combined evaluation of (mantissa of) $u^2 + v^2$, the shift-and-add-based approach is used in a modified form as compared with the evaluation of u^2 for one-dimensional lattice models. As before, quantum full adders, as introduced by Cuccaro (here, controlled by ancilla qubit $|anc\rangle$) form the building block. The clear, modular structure was the motivation to use this type of full adder.

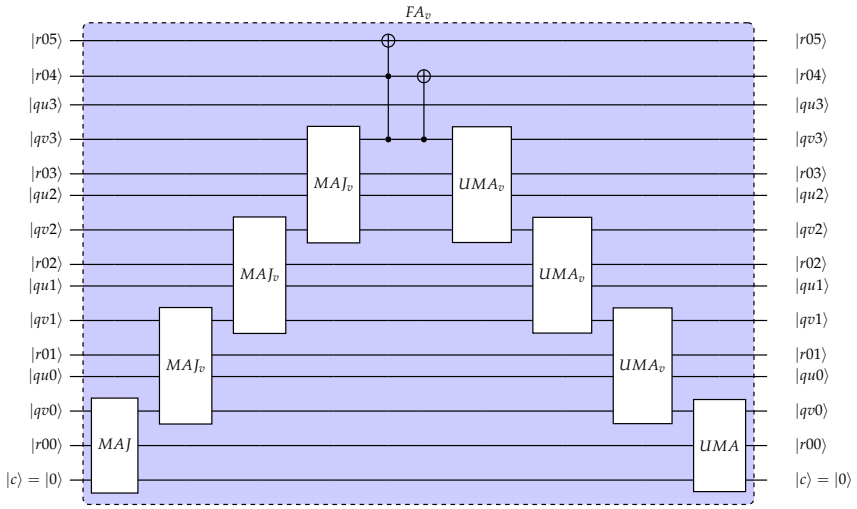
For the combined evaluation of $u^2 + v^2$, a number of modifications to the full adder are introduced that can be summarized as follows:

- For N_M mantissa qubits for both u- and v-velocity, a $(2N_M + 1)-$qubit output register is needed to remove the potential of overflow;
- In the shift-and-add approach, $N_M - 1$ shifts will be used as in the one-dimensional u^2 evaluation;
- In each of the N_M-controlled addition operations for u and N_M-controlled additions steps for v, the controlled additions add N_M mantissa qubits into an $(N_M + 1)-$qubit register;
- For $N_M = 4$, the modified full adders are defined starting from a 5-qubit full adder. The modification is based on a specialization of the $(N_M + 1)$-qubit input register based on the fact that the most significant qubit will be $|0\rangle$;
- A modular structure is maintained by a pair-wise arrangement of the qubits defining u- and v-velocity mantissa, as demonstrated later. This means that now MAJ and UMA blocks appear defined to act covering four qubits (however, one qubit will never be acted on).

Figure 10 shows the quantum-circuit implementation of the modified adders used to add 4-qubit u-input (top of figure) as well as 4-qubit v-input into the 6-qubit register $|r05| \ldots |r00\rangle$. Modified sub-circuits MAJ_u and MAJ_v are derived from MAJ by accounting for the 1-qubit additional spacing introduced by the second set of input qubits interleaved with the first set of input qubits. Similarly, sub-circuits UMA_u and UMA_v can be derived directly from UMA. As can be seen in the circuit for the addition of v-input at the bottom of Figure 10, for the least-significant inputs $|v0\rangle$, the original MAJ and UMA sub-circuits need to be used since only three qubits are involved in these steps. The modified full adders shown here form the basis for the mantissa squaring used in the quantum-circuit design based on quantum floating-point arithmetic described in the next sections.



Quantum circuit adding 4-qubit u-input $|qu3|qu2|qu1|qu0\rangle$ into $|r05|\ldots|r00\rangle$



Quantum circuit adding 4-qubit v-input $|v3|v2|v1|v0\rangle$ into $|r5|\ldots|r0\rangle$

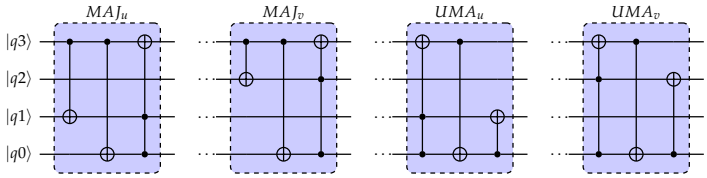


Figure 10. Definition of FA_u and FU_v-modified Cuccaro 5-qubit full adders for use in shift-and-add-based squaring for $N_M = 4$.

7. Detailed Description of Quantum Circuit for $u^2 + v^2$ Evaluation (Design *EKin2D3*)

For the shift-and-add multiplier used in Design *EKin2D3* detailed here, the modified full adder quantum circuits were detailed for the case $N_M = 4$. In this section, $N_M = 4$ will again be used as an example, where it needs to be considered that, in practically relevant applications, N_M will typically be significantly larger due to precision requirements.

Before detailing the quantum-circuit implementation, the key features of Design *EKin2D3* can be summarized as follows:

- Combined evaluation of mantissa squaring of u^2 and v^2 with a minimum of $(2N_M + 1)$ output to avoid overflow;
- Quantum floating-point arithmetic is used with u defined by $N_M - 1$ mantissa qubits $|mu2|mu1|mu0\rangle$ and $|eu2|eu1|eu0\rangle$ (N_E qubits defining exponent), and v is similarly defined by $|mv2|mv1|mv0\rangle$ and $|ev2|ev1|ev0\rangle$;

- In setting up the mantissa squaring, potential differences in the exponents of u and v need to be accounted for. If E_u and E_v are the integer values of the u-velocity and v-velocity exponents, respectively, then for $E_v < E_u$, a floating-point addition needs a 'shift' on the v-input qubits by $\Delta E = E_u - E_v$ placed in the input register of the adder toward a less significant position. For $E_v \leq E_u$ and normalized u-velocity, $|qu3|qu2|qu1|qu0\rangle$ is set to state $|1|mu2|mu1|mu0\rangle$. For $E_u - E_v = 1$, the input qubits $|qv3|qv2|qv1|qv0\rangle$ need to be set to state $|0|1|mu2|mu1\rangle$ (if u is normalized input). Similarly, for larger ΔE, ever larger shifts will be applied;
- The shifts introduced to account for exponent differences of u and v will lead to the elimination of one or more less-significant mantissa qubits for $\Delta E > 0$;
- Improved designs can be derived that employ an extended workspace in the shift-and-add-based multiplier, with the aim of reducing the truncation and rounding effects created for $\Delta E > 0$.

To investigate the effect of extending the shift-and-add-based multiplier in more detail, for $N_M = 4$, three different designs were considered, as detailed in the next sections. For clarity, only the case $E_u \geq E_v$ will be considered.

At the top of the quantum circuits for each of these designs, the qubit mapping can be summarized as follows:

$$\begin{aligned} |eu2|eu1|eu0\rangle &: N_E \text{ qubits defining the } u\text{-velocity exponent} \\ |vsub\rangle &: \text{initialized at } |0\rangle, \text{ in state } |1\rangle \text{ for sub-normal } v \\ |ev2|ev1|ev0\rangle &: N_E \text{ qubits defining the } v\text{-velocity exponent} \\ |mv2|mv1|mv0\rangle &: N_M - 1 \text{ qubits defining the } v\text{-velocity mantissa} \end{aligned}$$

Right at the bottom of the circuit, the qubits defined to store kinetic energy in the floating-point format are allocated:

$$\begin{aligned} |ek2|ek1|ek0\rangle &: N_E \text{ qubits defining the } u^+v^2 \text{ exponent} \\ |mk2|mk1|mk0\rangle &: N_M - 1 \text{ qubits defining the } u^2 + v^2 \text{ mantissa} \end{aligned}$$

The different designs use a varying number of workspace qubits, as detailed in the respective sections. In the quantum-circuit design, the first three steps are identical:

- Set $|vsub\rangle = |1\rangle$ for cases with $|ev2|ev1|ev0\rangle = |000\rangle$ (sub-normal v-velocity);
- Perform operation $Eu - Ev$ that calculates the difference in u- and $v-$velocity exponents ($E_u - E_v$) and set this difference (temporarily) in qubits $|ev2|ev1|ev0\rangle$;
- For $|eu2|eu1|eu0\rangle = |000\rangle$ (sub-normal u-velocity), set $|qu3\rangle = |0\rangle$;

In this design of the quantum circuit, $E_u \geq E_v$ was assumed. For the design with $(2N_M + 1) = 9$-qubit results register, these steps can be seen in the circuit shown in Figure 11. The next step involves the preparation of the mantissa-squaring operation.

7.1. $(2N_M + 1) = 9$*-Qubit Results Register* $|r8|\ldots|r0\rangle$

The minimum-width design employs a $(2N_M + 1) = 9$-qubit results register $|r08|\ldots|r00\rangle$. The $SQ4$ operator used in this design is shown in Figure 12 and employs the modified full adders defined in Section 6.3. As can be seen, N_M steps are used, with a controlled addition for u-velocity and $v-$velocity mantissa qubits during each step. $N_M - 1$ shift is performed on the output register, as previously for one-dimensional lattice models. Figure 11 shows the overall layout of the quantum circuit for $N_M = 4$ and $N_E = 3$, including the set-up of the $SQ4$ operation. The four u-velocity mantissa qubits are set as $|1|mu2|mu1|mu0\rangle$ for normalized u-velocity, and $|0|mu2|mu1|mu0\rangle$ for sub-normal u. The next four blocks of logical operations (separated by dashed lines) define the four qubits defining the mantissa of v-velocity, using a shift where needed. For $|ev2|ev1|ev0\rangle = |000\rangle$ (i.e., $E_u = E_v$), the four qubits $|qv3|qv2|qv1|qv0\rangle$ are set to $|1|mv2|mv1|mv0\rangle$ (for normalized inputs) or $|0|mv2|mv1|mv0\rangle$ for sub-normal v-velocity input. Next, the following blocks

account for the situations with $\Delta E = 1$, $\Delta E = 2$ and $\Delta E = 3$ in succession. As can be seen, for a normalized v-input with $\Delta E = 1$, the circuit defines $|qv3|qv2|qv1|qv0\rangle = |0|1|mu2|mu1\rangle$. Similarly, $|qv3|qv2|qv1|qv0\rangle = |0|0|1|mu2\rangle$ for $\Delta E = 2$. Finally, for $\Delta E = 3$, only the most significant mantissa qubit of v-velocity is used ($|1\rangle$ for normalized and $|0\rangle$ for sub-normal v-input). For $\Delta E \geq 4$, the contribution from v to $u^2 + v^2$ is eliminated fully.

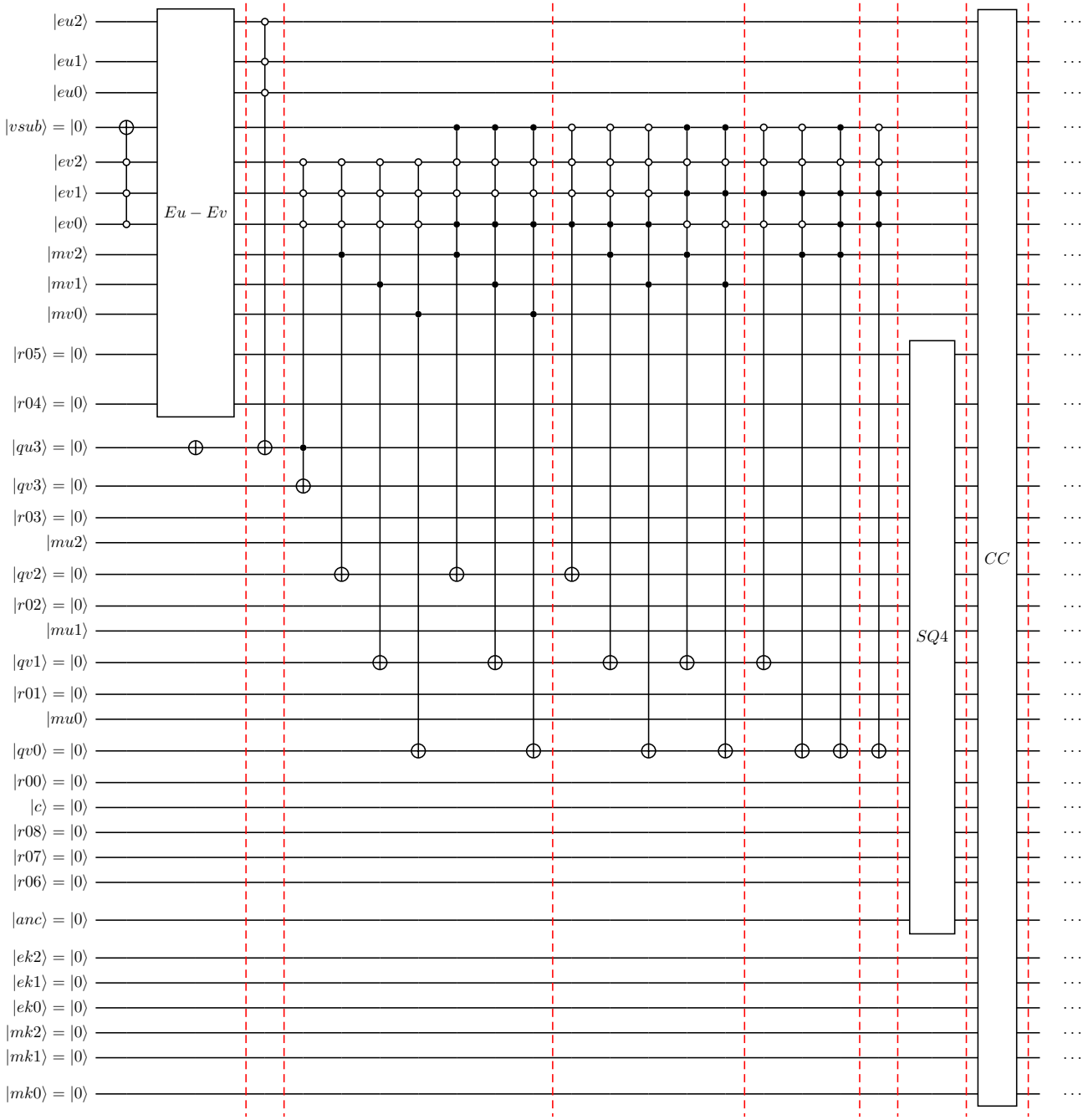


Figure 11. Design *EKin2D3* with $(N_M + 1)-$qubit mantissa squaring output register.

7.2. $(2N_M + 3) = 11$*-Qubit Results Register* $|r10|\ldots|r00\rangle$

In this design, a mantissa-squaring operation termed $SQ5$ sets the result in an 11-qubit output register for $N_M = 4$. For each of the $N_M + 1 = 5$ steps, five u-mantissa qubits and five v-mantissa qubits are added to the results register to employ two controlled, modified 6-qubit full adders. Similar to the modified 5-qubit full adders for the 9-qubit workspace design, the modification is based on the fact that the leading qubit in the original 6-qubit

input is $|0\rangle$. Then, for the mantissa of u and v, $N_M + 1 = 5$ qubits are conditionally added into a $N_M + 3 = 7$-qubit output register for the adder. A total of four shift operations are applied to the 11-qubit output register $|r10|\ldots|r00\rangle$ of the squaring operations, i.e., between each of the controlled-addition steps. This extended squaring operation is detailed in Figure 13.

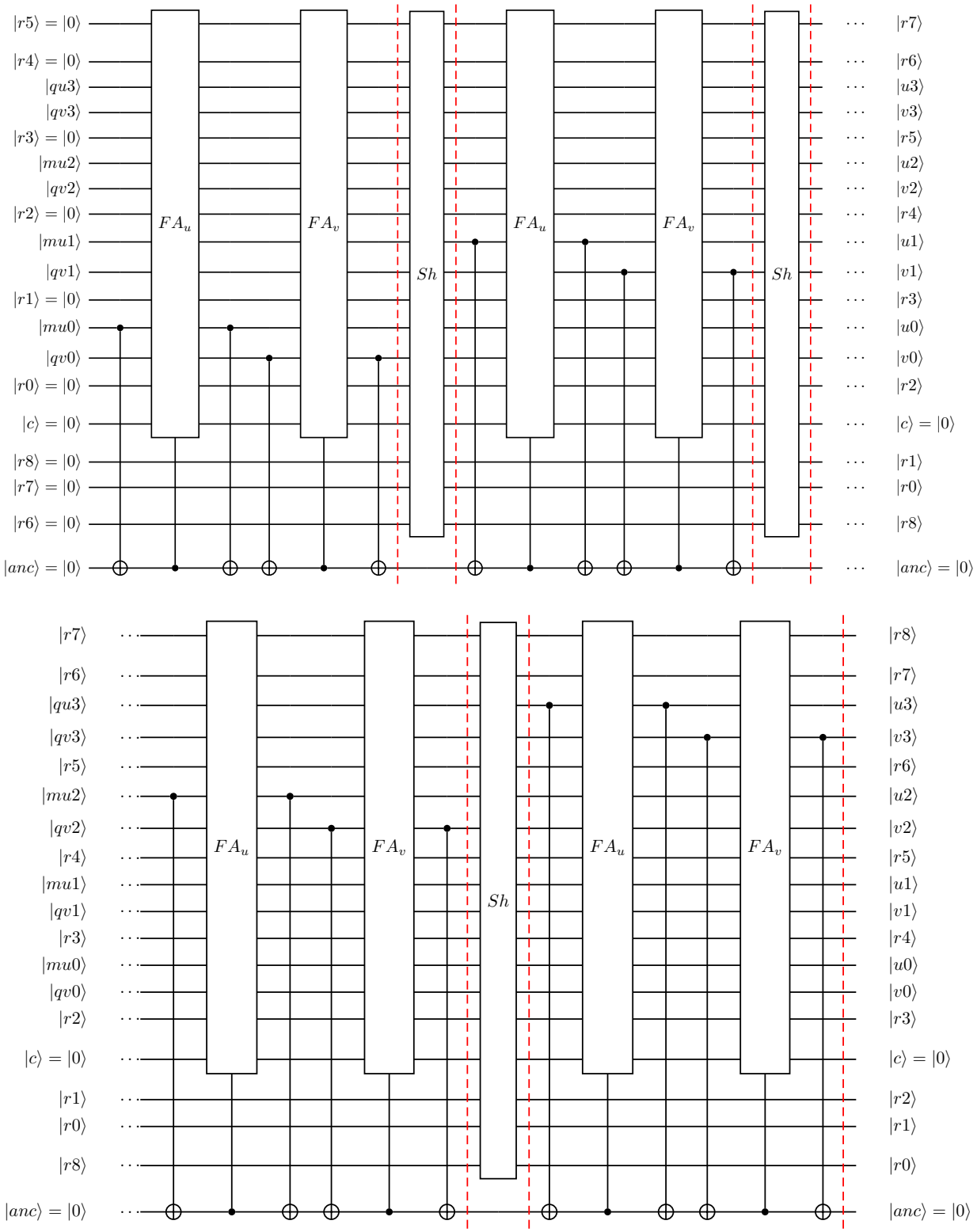


Figure 12. Design *EKin2D3* with $(N_M + 1)-$qubit mantissa squaring output register-*SQ*4 operator.

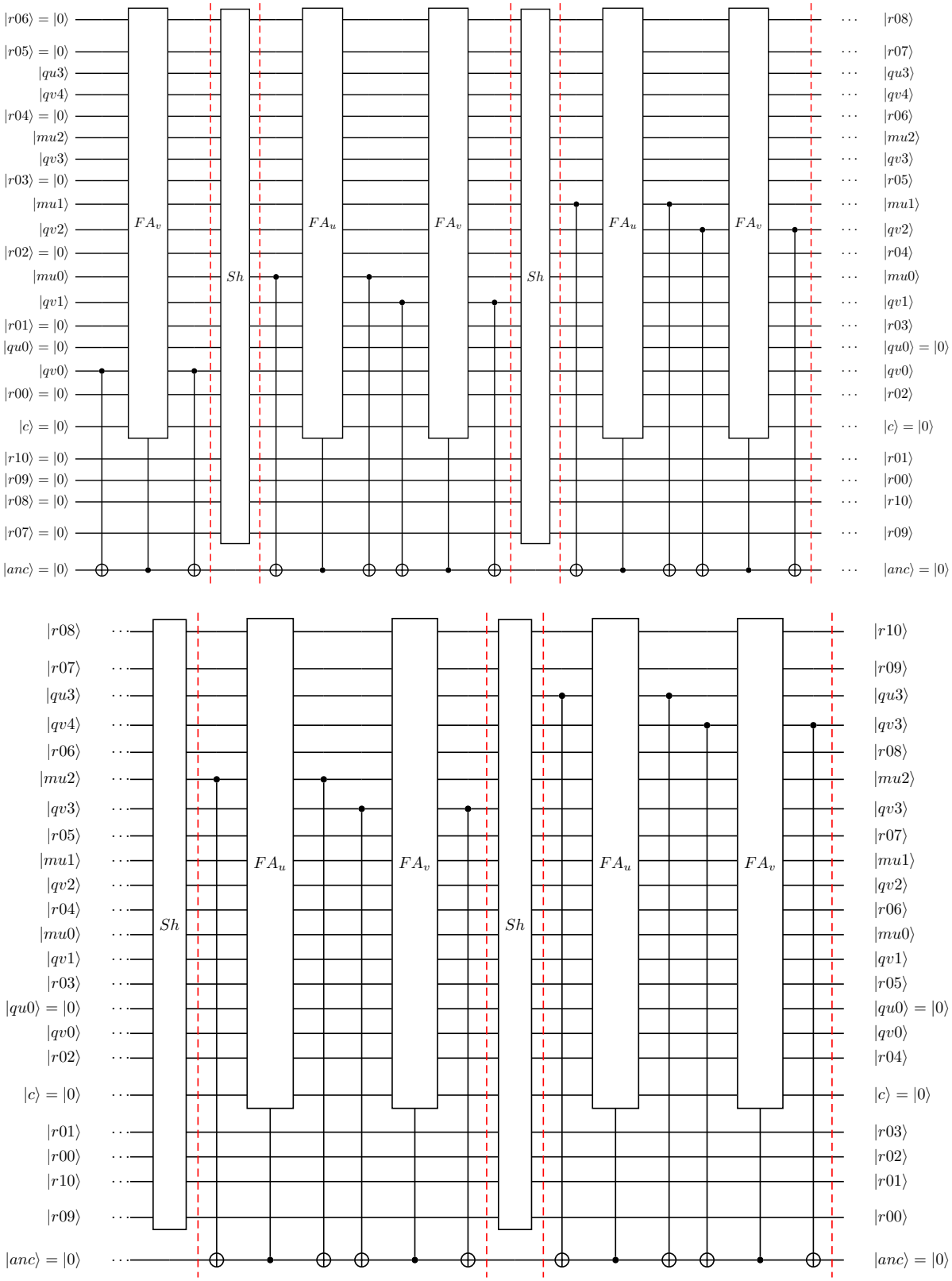


Figure 13. Quantum circuit defining $SQ5$ used in the computation of $u^2 + v^2$. Scaled velocity components u and v are represented as floating-point numbers with four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$). Circuit uses modified 6-qubit Cuccaro full adders (FA_u and FA_v) with 11-qubit results register. *Sh* performs a 'downward' shift of the results register.

Setting up the $SQ5$ operation differs significantly from setting up $SQ4$ in the previous design, as can be seen in Figure 14, where the overall design of the quantum-circuit implementation is shown. Here, five qubits define the mantissa of u-velocity and are set as $|1|mu2|mu1|mu0|0\rangle$ for normalized u-velocity and $|0|mu2|mu1|mu0|0\rangle$ for sub-normal u. For $E_u - E_v = 0$, the five qubits defining the v-velocity are set similarly as $|1|mv2|mv1|mv0|0\rangle$ for normalized v-velocity, and $|0|mv2|mv1|mv0|0\rangle$ for sub-normal v. Next, for $E_u - E_v = 1$, the five v-mantissa qubits are set as $|0|1|mv2|mv1|mv0\rangle$ for normalized v-velocity, and $|0|0|mv2|mv1|mv0\rangle$ for sub-normal v. This means that for $E_u - E_v = 1$, there is no elimination of the least-significant mantissa-qubit for v (as there was in the 11-qubit workspace design). For $E_u - E_v \geq 2$, shifts to the v-matissa qubits will be applied so that one or more less-significant v-mantissa qubits will be removed. The least-significant qubit in the $5-$qubit u-mantissa string is, therefore, always $|0\rangle$, explaining the absence of the full-adder for u in the first step in $SQ5$, shown in Figure 13.

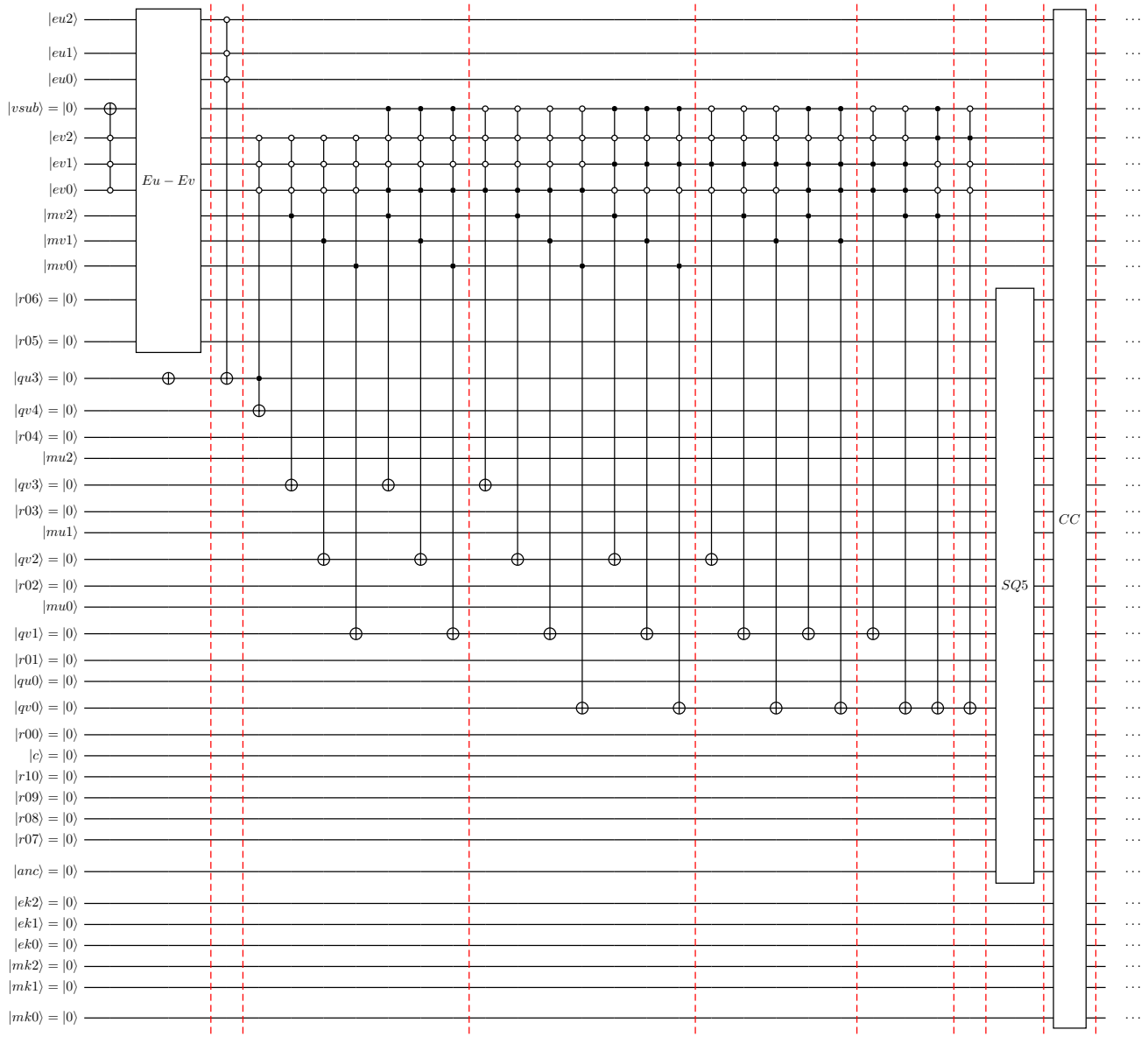


Figure 14. Design *EKin2D3* with $(N_M + 3)-$qubit mantissa squaring output register.

An alternative implementation for $N_M = 4$ using the modified, controlled 5-qubit full adders, as used previously in the 9-qubit workspace implementation, can also be created. In that case, the squaring operation based on the shift-and-add approach employs $N_M + 2 = 6$ steps with $N_M + 1 = 5$ shift operations. In the interest of brevity, this alternative circuit design is not discussed further.

7.3. $(2N_M + 5) = 13$-*Qubit Results Register* $|r12| \ldots |r00\rangle$

In this design, a mantissa-squaring operation termed $SQ6$ sets the result in a 13-qubit output register for $N_M = 4$. For each of the $N_M + 2 = 6$ steps, six u-velocity mantissa qubits and six v-velocity mantissa qubits are added to the results register to employ two controlled, modified 7-qubit full adders. Similar to the modified 5-qubit full adders for the 9-qubit workspace design, the modification is based on the fact that the leading qubit in the original 7-qubit input is $|0\rangle$. Then, for the mantissa of u and v, $N_M + 1 = 6$ qubits are conditionally added to a $N_M + 3 = 8$-qubit output register for the adder. A total of five shift operations are applied to the 13-qubit output register $|r12| \ldots |r00\rangle$ of the squaring operations, i.e., between each of the controlled-addition steps. This extended squaring operation is detailed in Figure 15.

Setting up the $SQ6$ operation differs significantly from setting up $SQ4$ and $SQ5$ in the previous designs. Here, six qubits define the u-velocity mantissa and are set as $|1|mu2|mu1|mu0|0|0\rangle$ for the normalized u-velocity and $|0|mu2|mu1|mu0|0|0\rangle$ for the sub-normal u. For $E_u - E_v = 0$, the six qubits defining the v-velocity mantissa are set similarly to $|1|mv2|mv1|mv0|0|0\rangle$ for the normalized v-velocity, and $|0|mv2|mv1|mv0|0|0\rangle$ for the sub-normal v. For $E_u - E_v = 1$, the six v-mantissa qubits are set as $|0|1|mv2|mv1|mv0|0\rangle$ for the normalized v-velocity, and $|0|0|mv2|mv1|mv0|0\rangle$ for the sub-normal v. Next, for $E_u - E_v = 2$, the six v-mantissa qubits are set as $|0|0|1|mv2|mv1|mv0\rangle$ for the normalized v-velocity, and $|0|0|0|mv2|mv1|mv0\rangle$ for the sub-normal v. This means that for $E_u - E_v \geq 2$ there is no elimination of the least-significant mantissa-qubit for v. For $E_u - E_v \geq 3$, shifts to the v-mantissa qubits will be applied so that one or more less-significant v-mantissa qubits will be removed. The two least-significant qubits in the 6-qubit u-mantissa string are, therefore, always $|0\rangle$, explaining the absence of the full-adder for u in the first two steps in $SQ6$, as shown in Figure 15.

As observed previously for the design employing $(2N_M + 3) = 11$-qubits, a quantum circuit design employing the modified controlled $(N_M + 1)$-qubit full adders, as used previously in the 9-qubit workspace implementation, can also be created, involving $N_M + 4 = 8$ steps with $N_M + 3 = 7$ shift operations. In the interest of brevity, this alternative circuit design is not discussed further.

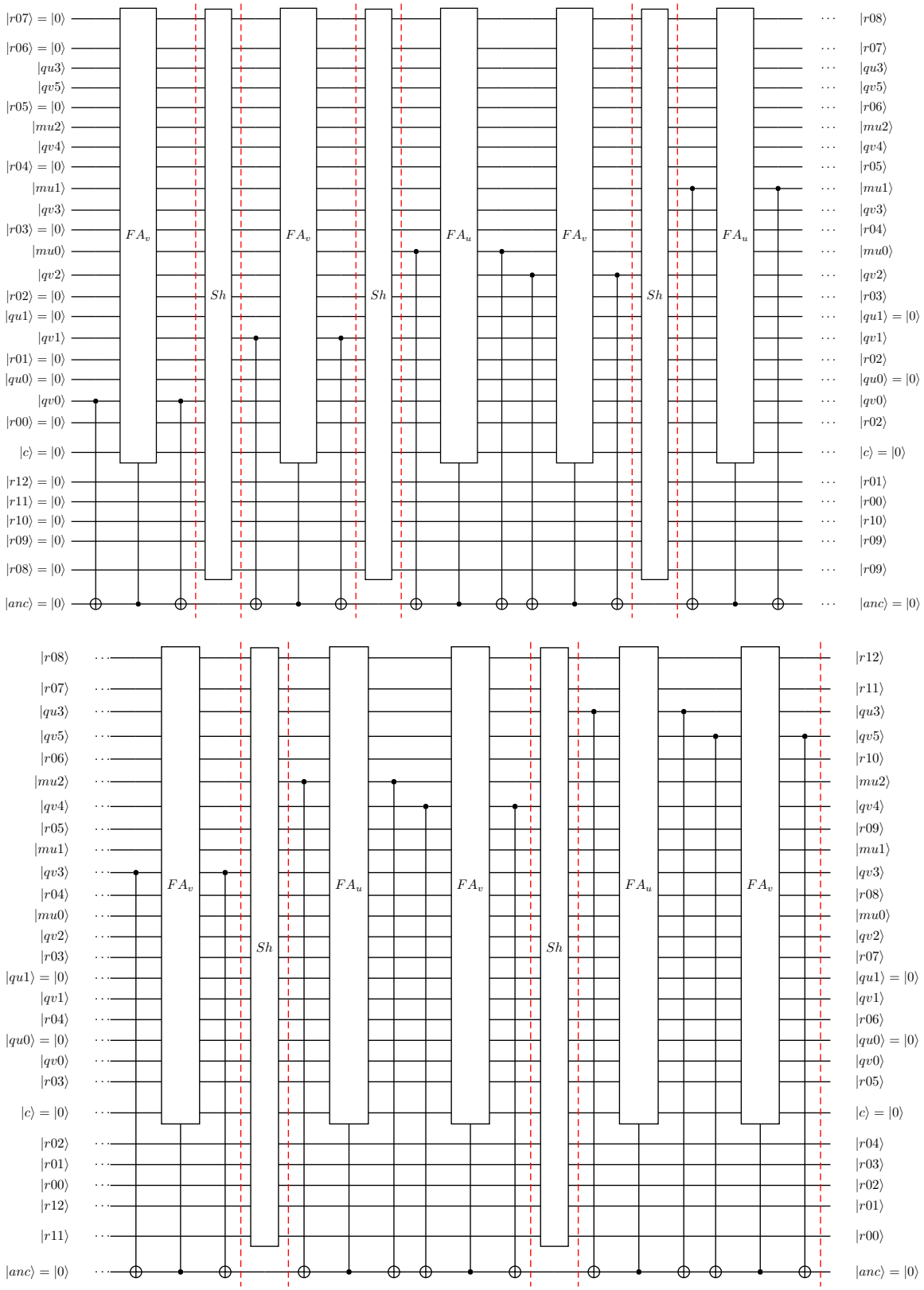


Figure 15. Quantum circuit defining *SQ6* used in the computation of $u^2 + v^2$. The scaled velocity components u and v are represented as floating-point numbers with four mantissa qubits ($N_M = 4$) and three exponent qubits ($N_E = 3$). The circuit uses modified 7-qubit Cuccaro full adders (FA_u and FA_v) with a 13-qubit results register. *Sh* performs a 'downward' shift of the results register.

8. Setting $u^2 + v^2$ in the Floating-Point Format

Figures 16 and 17 show the quantum-circuit implementation of the *CC* operator as used in the kinetic-energy evaluation for an exponent bias of 5. As before, $N_M = 4$, $N_E = 3$ and the assumption $E_v \leq E_u$ were made in developing the circuit. Then, for $|eu2|eu1|eu0\rangle = |000\rangle$, the output $u^2 + v^2$ will always be truncated to 0. For other exponents of u-velocity, Table 1 shows the range of non-zero floating-point numbers for output $u^2 + v^2$.

For the remaining exponent values, the range of outputs can be summarized as follows:

The left-most two gate operations in Figure 16 perform the required operations for $|eu2|eu1|eu0\rangle = |001\rangle$. As can be seen, for $|eu2|eu1|eu0\rangle = |010\rangle$, the output can be sub-normal or normalized. In terms of quantum-circuit implementation, $|r8\rangle = |1\rangle$ indicates that the normalized output occurs for $|eu2|eu1|eu0\rangle = |010\rangle$. In Figure 16, the case $|eu2|eu1|eu0\rangle = |010\rangle$ is implemented as the second block of gate operations (from left). In terms of quantum-circuit implementation, the cases $|eu2|eu1|eu0\rangle = |011\rangle$ and $|eu2|eu1|eu0\rangle = |100\rangle$ are largely identical (apart from setting exponent), as shown in Figure 17. The possibility of overflow for $|eu2|eu1|eu0\rangle = |101\rangle$ and the guaranteed overflow for $|eu2|eu1|eu0\rangle = |110\rangle$ are accounted for in the third block of gate operations in Figure 16.

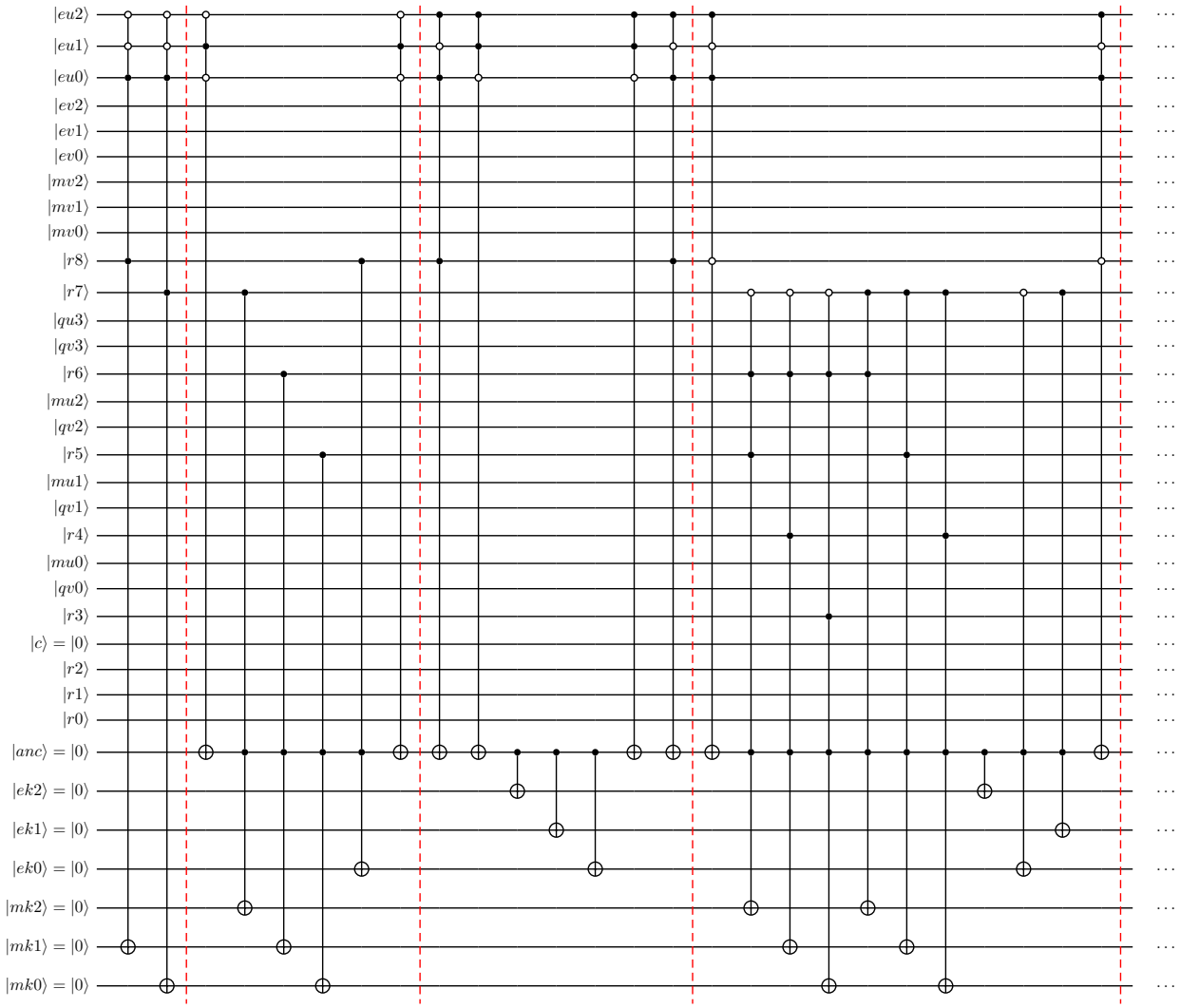


Figure 16. Quantum circuit implementation of *CC* used to set $u^2 + v^2$ in the floating-point format from the 9-qubit work register. $N_M = 4$ and $N_E = 3$. The exponent bias is five. Part 1.

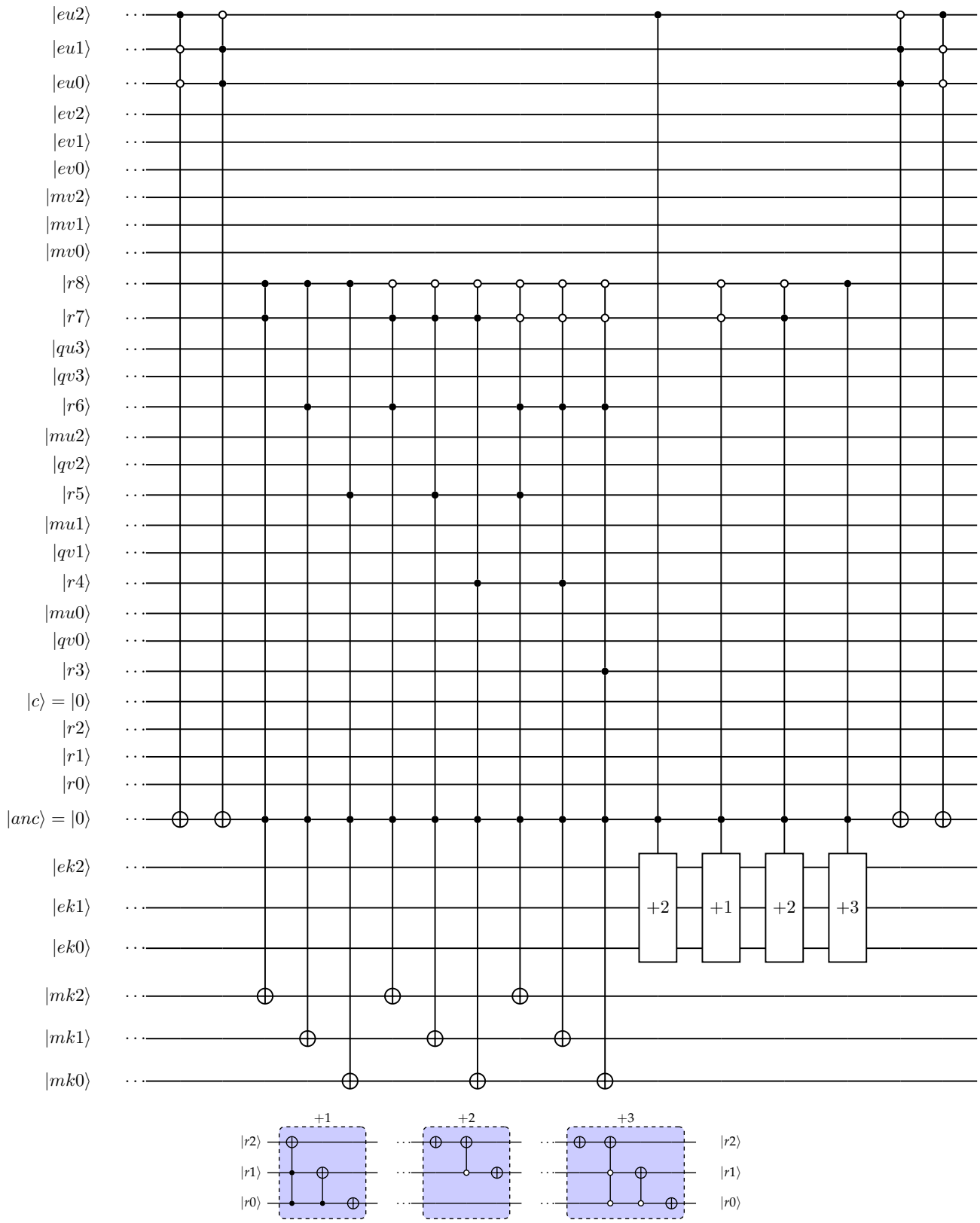


Figure 17. Quantum circuit implementation of *CC* used to set $u^2 + v^2$ in the floating-point format from the 9-qubit work register. $N_M = 4$ and $N_E = 3$. The exponent bias is five. Part 2.

Table 1. Setting $u^2 + v^2$ as floating-point numbers. For different exponents of u-velocity, the range of numbers is shown. The exponent bias is five.

$\|eu2\|eu1\|eu0\rangle$	Minimum	Maximum
$\|001\rangle$	$(8/128)^2 \rightarrow \|000\|000\rangle$	$2 \times (15/128)^2 \rightarrow \|000\|011\rangle$
$\|010\rangle$	$(8/64)^2 \rightarrow \|000\|010\rangle$	$2 \times (15/64)^2 \rightarrow \|001\|110\rangle$
$\|011\rangle$	$(8/32)^2 \rightarrow \|001\|000\rangle$	$2 \times (15/32)^2 \rightarrow \|011\|110\rangle$
$\|100\rangle$	$(8/16)^2 \rightarrow \|011\|000\rangle$	$2 \times (15/16)^2 \rightarrow \|101\|110\rangle$
$\|101\rangle$	$(8/8)^2 \rightarrow \|101\|000\rangle$	$2 \times (15/8)^2 \rightarrow$ overflow

9. Summary of Methodology and Algorithms

The quantum circuits presented in this work were intended as building blocks for future quantum-circuit implementations of full LBM models. To facilitate further developments as well as a wider adoption of the presented circuits, a step-by-step methodology for simulating the algorithm for the modified D1Q3 model is presented in this section. The methodology is based on the evaluation of the circuits on a Quantum Computer simulator. For the Burgers model and kinetic energy evaluation, the methodology is equivalent and not discussed here in the interest of brevity. The step-by-step approach outlined here can be used for two purposes. First, for the relatively small circuits considered here, the intended state of the qubit register at the end of the simulation for a chosen input state can be obtained classically so that the circuits can be verified. A second application involves hybrid quantum/classical simulation approaches where the simulated quantum circuit represents the quantum part of the hybrid simulation.

9.1. Methodology Using Quantum Computer Simulator

For all quantum circuits considered, it is assumed that qubits at the top of quantum circuits shown to act as the most-significant bits in the indexing of amplitudes of the quantum state vector. Furthermore, it is assumed that the employed simulator provides direct access to quantum state amplitudes, so that quantum measurements can be simulated by inspecting amplitudes of the output state. It should be noted that for the number of qubits used in the considered circuits, the simulations employing storage of the full state vector will often exceed limits on available memory. Quantum-circuit reduction steps [12] can then be used to perform a partial evaluation of the circuits. In the interest of brevity, this is not pursued further here, and it is assumed that the original circuit can be simulated directly.

9.2. Algorithm for the Modified D1Q3 Model

At the start of the simulation, it is assumed that velocity u represented in the floating-point format is known (e.g., for a specific example or from a previous step in a hybrid quantum/classical simulation approach) so that $|eu2|eu1|eu0\rangle$ and $|mu2|mu1|mu0\rangle$ are defined (for examples with $N_M = 4$ and $N_E = 3$). For positive u, $|su\rangle = |0\rangle$. The methodology for circuit evaluation involves the following steps:

1. Initialization of initial state of the qubit register $|\psi\rangle_{init}$. Using the qubit arrangement shown in Figures 3 and 4, for each of the four directions defined by qubits $|dv1|dv0\rangle$, a single amplitude of $|\psi\rangle_{init}$ is set to $1/2$ (to define a normalized state vector). The indices of the non-zero amplitude for each of the four directions are:

$$\begin{aligned}
|dv1|dv0\rangle = |00\rangle \quad &: \quad (00|eu2|eu1|eu1|0|mu2|mu1|mu0|\ldots|0)_2 \\
|dv1|dv0\rangle = |01\rangle \quad &: \quad (01|eu2|eu1|eu1|0|mu2|mu1|mu0|\ldots|0)_2 \\
|dv1|dv0\rangle = |10\rangle \quad &: \quad (10|eu2|eu1|eu1|0|mu2|mu1|mu0|\ldots|0)_2 \\
|dv1|dv0\rangle = |11\rangle \quad &: \quad (11|eu2|eu1|eu1|0|mu2|mu1|mu0|\ldots|0)_2
\end{aligned}$$

where dots refer to further qubits in state $|0\rangle$;

2. Evolution of the quantum state of qubit register by performing the quantum gate operations outlined in the quantum circuits shown in Figures 3 and 4. After completion, the output quantum state $|\psi\rangle_{out}$ will have four non-zero amplitudes equal to $1/2$. For non-zero u, the indices of these are different from those in the initial state to reflect the computational work performed;
3. Obtaining output. Assuming that in the output state, the indices of non-zero amplitudes are defined by:

$$(00|eu2|eu1|eu1|0|mu2|mu1|mu0|eg2|eg1|eg0|sg|mg2|mg1|mg0|\ldots|0)_2$$
$$(01|eu2|eu1|eu1|0|mu2|mu1|mu0|eg2|eg1|eg0|sg|mg2|mg1|mg0|\ldots|0)_2$$
$$(10|eu2|eu1|eu1|0|mu2|mu1|mu0|eg2|eg1|eg0|sg|mg2|mg1|mg0|\ldots|0)_2$$
$$(11|eu2|eu1|eu1|0|mu2|mu1|mu0|eg2|eg1|eg0|sg|mg2|mg1|mg0|\ldots|0)_2$$

the state of the qubits $|eg2|eg1|eg0\rangle$, $|sg\rangle$ and $|mg2|mg1|mg0\rangle$ (for $N_M = 4$ and $N_E = 3$) defining the equilibrium distribution function for each of the four directions can be obtained;

Using the presented methodology for circuit verification, the final step would involve checking the obtained shift in indices of non-zero amplitudes with those for the intended output state. For hybrid quantum/classical approaches, the output obtained for the equilibrium distribution function components is used to evaluate the collision term in the 'classical' part of the algorithm. Once the non-equilibrium distribution function and velocity u have been updated in the classical part of the hybrid approach, the steps outlined above can be repeated, starting again from step 1.

10. Conclusions

Quantum-circuit implementations for the evaluation of the non-linear equilibrium distribution function in one-dimensional non-linear lattice models for fluid modeling were presented in detail. A reduced-precision quantum floating-point format was employed with an asymmetric bias optimized for the lattice model considered. For this floating-point representation, the increase in the quantum circuit width and circuit depth with an increasing number of mantissa qubits used was analyzed. The second part of this work dealt with the design of quantum-circuit implementations for the non-linear terms of equilibrium distribution functions in multi-dimensional Lattice Boltzmann models. Specifically, as a first step, the evaluation of the kinetic energy in two-dimensional lattice models was described in detail. The complexity analysis of various quantum circuit designs considered in this study showed that a novel, modular design in which a shift-and-add-based approach directly incorporates contributions from u- and v-velocity components leads to the smallest quantum circuit width. Extending the number of mantissa qubits used in the examples (i.e., $N_M = 4$) to a more realistic scale, e.g., ≥ 8 (to avoid excessive rounding), the quantum circuit width becomes $O(100)$ qubits. For the projected availability of future quantum computers, this means that in the near future, meaningful proof-of-concept experiments could be performed. In future work, the extension to quantum-circuit implementations for the collision terms of the full two- and three-dimensional Lattice Boltzmann methods will be presented as the first next step. Then, quantum algorithms for the full two- and three-dimensional Lattice Boltzmann method will be the main research focus.

Funding: This research received no external funding.

Data Availability Statement: More detailed information on the quantum-circuit implementations, as well as data from complexity analysis, is available from the author upon request by email.

Acknowledgments: The author would like to acknowledge funding received from the University of Glasgow supporting this research.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CFD	Computational Fluid Dynamics
QC	Quantum Computing
LBM	Lattice Boltzmann Method
NS	Navier–Stokes
NISQ	Noisy Intermediate-Scale Quantum

Appendix A. Quantum Floating-Point Representation

An important characteristic of the quantum-circuit development described in this work is the use of a floating-point representation along with computational basis encoding. This quantum floating-point format was introduced by the author in a previous work [6,12] and has the following main features:

- Its design follows the IEEE-754 standard [15] in employing sub-normal numbers and consistent rounding (rounding down to the nearest is used here), as well as the representation of overflow conditions. The maximum value for the exponent possible (considering the number of exponent qubits used) defines overflow, while an exponent value of 0 means that the number represented is sub-normal;
- The number of mantissa and exponent qubits is smaller than the equivalent number of bits in the IEEE-754 single-precision format. The example circuits shown here typically employ four mantissa qubits in the interest of limiting the size and improving the clarity of the circuits;
- Using the 'hidden-qubit' approach, only $N_M - 1$ qubits are stored in a representation with N_M mantissa qubits. N_E qubits represent the exponent;
- The qubit layout for a floating-point number is as follows: the leading qubit represents the sign ($|0\rangle$ for positive numbers), followed by N_E exponent qubits, and finally, $N_M - 1$ mantissa qubits;
- In contrast to the IEEE-754 standard, an asymmetric bias is used here for the exponent. The motivation for this choice was to reduce the number of exponent qubits required to provide a sufficient range for the floating-point numbers for the specific application considered. In the quantum circuits developed here, $N_E = 3$ was selected as a good comprise between range and quantum-circuit width;
- For $N_E = 3$, the maximum exponent value is seven, so the state $|111\rangle$ for exponent qubits indicates overflow. For $N_E = 3$, an exponent bias equal to three is a 'symmetric' bias when following the approach used in the IEEE-754 definition. In the current applications, the floating-point numbers are biased toward smaller numbers by using an exponent bias greater than three.

In Table A1, the sub-normal numbers (and zero) for $N_M = 4$ and $N_E = 3$ are shown for a symmetric bias (3 for $N_E = 3$) and for an exponent bias of eight, selected here for applications of the modified D1Q3 model. As expected, the sub-normal numbers for the asymmetric bias are a factor $2^{(8-3)} = 32$ smaller. Although the floating-point format includes a sign qubit, the arithmetic performed in the quantum circuits typically accounts for a negative sign by temporarily converting the mantissa to 2's complement. The last column in Table A1 shows 2's complement representation of the mantissa for the sub-normal numbers. Starting from the mantissa (including 'hidden qubit'), $N_M + 1 = 5$ qubits are needed to represent 2's complement. As in the IEEE-754 standard, sub-normal numbers are the smallest numbers that can be represented. The best accuracy in using floating-point arithmetic is obtained when normalized floating-point numbers are used. For the D1Q3 lattice example, velocity components (scaled by lattice speeds) are among the variables that need to be represented. Small numbers will result (typically < 0.1 due to scaling used), so an asymmetric bias is used to make the floating-point representation use normalized numbers for these velocities and for the similarly small distribution function components represented in floating-point format (at least for most of the considered computational domain). Further,

for velocity components represented as normalized numbers, computing the square will lead to reduced rounding and truncation compared with velocity components defined as sub-normal numbers for the same N_M. To illustrate normalized numbers as used in the present work, Table A2 shows the floating-point numbers for an exponent value of six. Again, numbers defined for symmetric exponent bias are compared with those for the selected exponent bias of eight. Clearly, the use of the asymmetric bias reduces the maximum (absolute) value that can be represented. However, for the applications considered, this represents no problem with the scaling employed. Furthermore, checks on the occurrence of overflow can be performed using the status of exponent qubits.

Table A1. Sub-normal floating-point numbers with four mantissa and three exponent qubits. The sign is defined by the leading qubit.

bias = 3				
Positive		**Negative**		**2's Complement (Mantissa)**
$\|0\|000\|000\rangle$	0	$\|0\|000\|000\rangle$	0	$\|00000\rangle$
$\|0\|000\|001\rangle$	1/32	$\|1\|000\|001\rangle$	−1/32	$\|11111\rangle$
$\|0\|000\|010\rangle$	2/32	$\|1\|000\|010\rangle$	−2/32	$\|11110\rangle$
$\|0\|000\|011\rangle$	3/32	$\|1\|000\|011\rangle$	−3/32	$\|11101\rangle$
$\|0\|000\|100\rangle$	4/32	$\|1\|000\|100\rangle$	−4/32	$\|11100\rangle$
$\|0\|000\|101\rangle$	5/32	$\|1\|000\|101\rangle$	−5/32	$\|11011\rangle$
$\|0\|000\|110\rangle$	6/32	$\|1\|000\|110\rangle$	−6/32	$\|11010\rangle$
$\|0\|000\|111\rangle$	7/32	$\|1\|000\|111\rangle$	−7/32	$\|11001\rangle$
bias = 8				
Positive		**Negative**		**2's Complement (Mantissa)**
$\|0\|000\|000\rangle$	0	$\|0\|000\|000\rangle$	0	$\|00000\rangle$
$\|0\|000\|001\rangle$	1/1024	$\|1\|000\|001\rangle$	−1/1024	$\|11111\rangle$
$\|0\|000\|010\rangle$	2/1024	$\|1\|000\|010\rangle$	−2/1024	$\|11110\rangle$
$\|0\|000\|011\rangle$	3/1024	$\|1\|000\|011\rangle$	−3/1024	$\|11101\rangle$
$\|0\|000\|100\rangle$	4/1024	$\|1\|000\|100\rangle$	−4/1024	$\|11100\rangle$
$\|0\|000\|101\rangle$	5/1024	$\|1\|000\|101\rangle$	−5/1024	$\|11011\rangle$
$\|0\|000\|110\rangle$	6/1024	$\|1\|000\|110\rangle$	−6/1024	$\|11010\rangle$
$\|0\|000\|111\rangle$	7/1024	$\|1\|000\|111\rangle$	−7/1024	$\|11001\rangle$

Table A2. Example of normalized floating-point numbers with four mantissa and three exponent qubits. Exponent has value of six (110 in binary). The sign is defined by the leading qubit.

bias=3				
Positive		**Negative**		**2's Complement (Mantissa)**
$\|0\|110\|000\rangle$	8	$\|1\|110\|000\rangle$	−8	$\|11000\rangle$
$\|0\|110\|001\rangle$	9	$\|1\|110\|001\rangle$	−9	$\|10111\rangle$
$\|0\|110\|010\rangle$	10	$\|1\|110\|010\rangle$	−10	$\|10110\rangle$
$\|0\|110\|011\rangle$	11	$\|1\|110\|011\rangle$	−11	$\|10101\rangle$
$\|0\|110\|100\rangle$	12	$\|1\|110\|100\rangle$	−12	$\|10100\rangle$
$\|0\|110\|101\rangle$	13	$\|1\|110\|101\rangle$	−13	$\|10011\rangle$
$\|0\|110\|110\rangle$	14	$\|1\|110\|110\rangle$	−14	$\|10010\rangle$
$\|0\|110\|111\rangle$	15	$\|1\|110\|111\rangle$	−15	$\|10001\rangle$
bias=8				
Positive		**Negative**		**2's Complement (Mantissa)**
$\|0\|110\|000\rangle$	8/32	$\|1\|110\|000\rangle$	−8/32	$\|11000\rangle$
$\|0\|110\|001\rangle$	9/32	$\|1\|110\|001\rangle$	−9/32	$\|10111\rangle$
$\|0\|110\|010\rangle$	10/32	$\|1\|110\|010\rangle$	−10/32	$\|10110\rangle$
$\|0\|110\|011\rangle$	11/32	$\|1\|110\|011\rangle$	−11/32	$\|10101\rangle$
$\|0\|110\|100\rangle$	12/32	$\|1\|110\|100\rangle$	−12/32	$\|10100\rangle$
$\|0\|110\|101\rangle$	13/32	$\|1\|110\|101\rangle$	−13/32	$\|10011\rangle$
$\|0\|110\|110\rangle$	14/32	$\|1\|110\|110\rangle$	−14/32	$\|10010\rangle$
$\|0\|110\|111\rangle$	15/32	$\|1\|110\|111\rangle$	−15/32	$\|10001\rangle$

References

1. Nielsen, M.; Chuang, I. *Quantum Computation and Quantum Information: 10th Anniversary Edition*; Cambridge University Press: Cambridge, MA, USA, 2010.
2. Xu, G.; Daley, A.; Givi, P.; Somma, R. Turbulent Mixing Simulation via a Quantum Algorithm. *AIAA J.* **2018**, *56*, 687–699. https://doi.org/10.2514/1.J055896.
3. Steijl, R.; Barakos, G. Parallel evaluation of quantum algorithms for computational fluid dynamics. *Comput. Fluids* **2018**, *173*, 22–28. https://doi.org/10.1016/j.compfluid.2018.03.080.
4. Gaitan, F. Finding flows of a Navier–Stokes fluid through quantum computing. *npj Quantum Inf.* **2020**, *6*, 61. https://doi.org/10.1038/s41534-020-00291-0.
5. Todorova, B.; Steijl, R. Quantum Algorithm for the collisionless Boltzmann equation. *J. Comp. Phys.* **2020**, *409*, 109347. https://doi.org/10.1016/j.jcp.2020.109347.
6. Steijl, R. Quantum Algorithms for Fluid Simulations. In *Advances in Quantum Communication and Information*; Bulnes, F., Ed.; IntechOpen: Rijeka, Croatia, 2020; ISBN 978-1-78-985268-4. https://doi.org/10.5772/intechopen.86685.
7. Steijl, R. Quantum algorithms for nonlinear equations in fluid mechanics. In *Quantum Computing and Communications*; Zhao, Y., Ed.; IntechOpen:London, UK, 2022; ISBN 978-1-83-968133-2. https://doi.org/10.5772/intechopen.95023.
8. Williams, A.; Lind, S. A Quantum Computing Algorithm for Smoothed Particle Hydrodynamics. *arXiv* **2020**, arXiv:2006.06719. https://doi.org/10.48550/ARXIV.2006.06719.
9. Bharadwaj, S.; Sreenivasan, K. Quantum Computation of Fluid Dynamics. *Perspect. Nonlinear Dyn.* **2020**. https://doi.org/10.29195/iascs.03.01.0015.
10. Itani, W.; Succi, S. Analysis of Carleman Linearization of Lattice Boltzmann. *Fluids* **2022**, *7*, 24. https://doi.org/10.3390/fluids7010024.
11. Budinski, L. Quantum algorithm for the Navier–Stokes equations by using the streamfunction-vorticity formulation and the lattice Boltzmann method. *Int. J. Quantum Inf.* **2022**, *20*, 2150039. https://doi.org/10.1142/S0219749921500398.
12. Moawad, Y.; Vanderbauwhede, W.; Steijl, R. Investigating hardware acceleration for simulation of CFD quantum circuits. *Front. Mech. Eng.* **2022**, *8*, 925637. https://doi.org/10.3389/fmech.2022.925637.
13. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. https://doi.org/10.22331/q-2018-08-06-79.
14. Budinski, L. Quantum algorithm for the advection–diffusion equation simulated with the lattice Boltzmann method. *Quantum Inf. Process.* **2021**, *20*, 57. https://doi.org/10.1007/s11128-021-02996-3.
15. Overton, M. *Numerical Computing with IEEE Floating Point Arithmetic*, 1st ed.; SIAM: Philadelphia, PA, USA, 2001.
16. Velivelli, A.; Bryden, K. Domain decomposition based coupling between the lattice Boltzmann method and traditional CFD methods—Part I: Formulation and application to the 2-D Burgers' equation. *Adv. Eng. Softw.* **2014**, *70*, 104–112. https://doi.org/10.1016/j.advengsoft.2014.01.012.
17. Moll, N.; Barkoutsos, P.; Bishop, L.S.; Chow, J.M.; Cross, A.; Egger, D.J.; Filipp, S.; Fuhrer, A.; Gambetta, J.M.; Ganzhorn, M.; et al. Quantum optimization using variational algorithms on near-term quantum devices. *Quantum Sci. Technol.* **2018**, *3*, 030503. https://doi.org/10.1088/2058-9565/aab822.
18. Peruzzo, A.; McClean, J.; Shadbolt, P.; Yung, M.H.; Zhou, X.; Love, P.; Aspuru-Guzik, A.; O'Brien, J. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **2014**, *5*, 4213. https://doi.org/10.1038/ncomms5213.
19. McClean, J.; Romero, J.; Babbush, R.; Aspuru-Guzik, A. The theory of variational hybrid quantum-classical algorithms. *New J. Phys.* **2016**, *18*, 023023. https://doi.org/10.1088/1367-2630/18/2/023023.
20. Lubasch, M.; Joo, J.; Moinier, P.; Kiffner, M.; Jaksch, D. Variational quantum algorithms for nonlinear problems. *Phys. Rev. A* **2020**, *101*, 010301. https://doi.org/10.1103/PhysRevA.101.010301.
21. Zhou, S.; Loke, T.; Izaac, J.; Wang, J. Quantum Fourier transform in computational basis. *Quantum Inf. Proc.* **2017**, *16*, 82. https://doi.org/10.1007/s11128-017-1515-0.
22. Mitarai, K.; Kitagawa, M.; Fujii, K. Quantum analog-digital conversion. *Phys. Rev. A* **2019**, *99*, 012301. https://doi.org/10.1103/PhysRevA.99.012301.
23. SaiToh, A. Quantum digital-to-analog conversion algorithm using decoherence. *Quantum Inf. Process.* **2015**, *14*, 2729–2748. https://doi.org/10.1007/s11128-015-1033-x.
24. Häner, 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. https://doi.org/10.48550/arxiv.1807.02023.
25. Rogers, M.L.; Singleton, R.L. Floating-Point Calculations on a Quantum Annealer: Division and Matrix Inversion. *Front. Phys.* **2020**, *8*, 265. https://doi.org/10.3389/fphy.2020.00265.
26. 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. https://doi.org/10.48550/arXiv.quant-ph/0410184.