A Quantum Approach for Implementing Fixed-Point Arithmetic in Solving Ordinary Differential Equations

José E. Cruz Serrallés*, Oluwadara Ogunkoya†, Doğa Murat Kürkçüoğlu†, Nicholas Bornman†, Norm M. Tubman‡, Silvia Zorzetti†, and Riccardo Lattanzi*
* Center for Biomedical Imaging, Department of Radiology, New York University Grossman School of Medicine, New York, NY.
† Superconducting Quantum Materials and System Center (SQMS), Fermi National Accelerator Laboratory, Batavia, Illinois 60510, USA.
‡ NASA Ames Research Center, Moffett Field, CA, 94035, USA.

***Abstract*—Differential equations (DEs) serve as fundamental tools in mathematical modeling across scientific disciplines, yet classical numerical solvers face limitations with large-scale or computationally intensive problems. This study explores a quantum-inspired approach to solving DEs, combining quantum-inspired techniques with classical methods. It focuses on fixed-point arithmetic on quantum circuits, utilizing basic quantum gates to manipulate DE solutions. We expand upon the techniques introduced by Zanger et al. [Quantum, 5, 502 (2021)] by offering a precise computation for a fixed-point signed multiplication scheme, while also presenting a quantum circuit capable of executing the fixed-point division algorithm. We demonstrate the feasibility of our approach through the simulation of a linear Ordinary Differential Equation (ODE), where initial conditions and parameters are encoded into quantum circuits using fixed-point representation. By executing sequences of quantum gates mimicking numerical integration steps, we obtain approximate solutions to the ODE with specified fixed-point precision.**

I. Introduction

Differential equations (DE) feature prominently in scientific modeling across various domains, including physical and biological systems. Due to their analytical complexity, numerical, rather than exact analytical, methods are commonly employed to obtain solutions. While classical approaches to modeling these problems are well-established, they are not without limitations, such as high computational costs and precision inaccuracies. A comprehensive overview of classical modeling techniques can be found in [1]–[4]. Quantum approaches to solving specific types of differential equations have demonstrated asymptotic scaling advantages compared to classical methods [5]–[8]. In [9], the Harrow-Hassidim-Lloyd (HHL) algorithm was introduced to demonstrate potential exponential speedup compared to classical algorithms for specific linear systems, particularly those characterized by sparse matrices with few non-zero entries. However, drawbacks of the algorithm, and indeed, any practical implementation of a quantum algorithm on current Noisy Intermediate-Scale Quantum (NISQ) era devices [10], includes the large qubit overhead and vulnerability to noise and errors in quantum computers' qubits and control electronics, among other deleterious phenomena. A related algorithm, the Quantum Singular Value Transformation (QSVT) [11], shares similarities with HHL. QSVT exhibits versatility in addressing a broader spectrum of linear systems, encompassing those featuring exponential terms. However, its practical application is hampered by the substantial quantum resources it demands, owing to the intricate nature of the associated quantum circuit. Alternative algorithms for time-independent matrix linear differential equations, particularly those involving non-Hermitian matrices, are explored in [12], [13].

In this study, we explore a hybrid quantum-classical methodology akin to that proposed in [14]. Initially, we provide a comprehensive overview of the techniques outlined in [14], refine the definition of multiplication within a fixed-point system involving binary points, and subsequently generalize these findings to encompass the reciprocal operator. To validate our approach, we apply the developed algorithm to simulate a problem analogous to that addressed in [14].

II. Theory

A. Definitions

In a fixed-point number representation system and within a quantum computing context, fundamental arithmetic operations can be simulated on a quantum circuit by leveraging well-chosen combinations of Hermitian operators (usually present in the Hamiltonian of a system). By encoding the operands and the desired arithmetic operation into the quantum circuit, the evolution of the system under the Hamiltonian can simulate addition, subtraction [15], and multiplication [16]. For an introduction to quantum computing at the logical level, the interested reader is directed to [17].

Given a continuous-time quantum system with the Hamiltonian set to the momentum operator $\hat{P}$, the position operator satisfies

$$\frac{\mathrm{d}\hat{X}}{\mathrm{d}t} = 1 \tag{1}$$

as a consequence of the canonical commutation relation $\left[\hat{X}, \hat{P}\right] = i\hbar$. An implication of this relationship is that given

a continuous-time quantum state $|a\rangle$, the operation $e^{-ic\hat{P}}|a\rangle$ results in a shift, or addition, of state $|a\rangle$ with constant c or equivalently $|a + c\rangle$ [14].

A single qubit encodes a superposition of two states, denoted $|0\rangle$ and $|1\rangle$, which are basis states lying in two-dimensional Hilbert space of the qubit. Extending this, a set of n qubits can encode a superposition of 2^n states (via tensor products of the underlying Hilbert space), each with a unique binary string representation. As such, every unsigned integer from 0 to $2^n - 1$ can be encoded encoded in a quantum system via its corresponding binary representation. In the subsequent discussion, we refer to these n–qubit sets containing a superposition of binary strings as n-qubit registers, with each binary string representing an unsigned integer with values ranging from 0 to $2^n - 1$. Hence, in qubit representation, a number $a = \sum_{j=0}^{n-1} 2^j a_j$ is equivalent to the state $|a_{n-1}, \ldots a_1 a_0\rangle$.

We give a brief review of the basic arithmetic operations as already shown in [14]. Eq. (1) holds when considering discrete operators [18]. Hence, a discretized Hamiltonian operator approach is employed to carry out addition and multiplication involving n-qubit registers [14]. More specifically, defining the discrete position operator $\hat{X}$ as

$$\hat{X} = \sum_{j=0}^{2^n-1} j|j\rangle\langle j|, \tag{2}$$

the discrete momentum operator is computed by diagonalizing with the Quantum Fourier Transform, as shown in the following equation.

$$\hat{P} = \hat{F}^\dagger \hat{X} \hat{F} \tag{3}$$

The discrete Quantum Fourier Transform $\hat{F}$ is defined as

$$\hat{F} = \sum_{k=0}^{2^n-1} e^{-i2\pi jk/2^n} |k\rangle\langle j|. \tag{4}$$

Note that this definition uses $e^{-j\omega}$ in the forward operator, which can differ from other definitions of the QFT that use $e^{j\omega}$, the conjugate.

B. Addition of an Unsigned Integral Register and a Constant Integer

For addition of two unsigned integers $|a\rangle$ and c, a is encoded in a n-qubit register while c is encoded as a rotation parameter. Since, there are 2^n basis states, addition is hereby considered to be $\mod 2^n$.

$$|a + c \bmod 2^n\rangle = e^{-i2\pi c\hat{P}/2^n}|a\rangle = \hat{F}^\dagger e^{-i2\pi c\hat{X}/2^n} \hat{F}|a\rangle \tag{5}$$

To translate to single qubit circuit gates, we make further expansions:

$$\begin{aligned}
e^{-i2\pi c\hat{X}/2^n} &= \sum_{a=0}^{2^n-1} e^{-i2\pi ca/2^n} |\, a \,\rangle\langle\, a \,| \\
&= \sum_{a=0}^{2^n-1} \prod_{j=0}^{n-1} e^{-i2\pi ca_j/2^{n-j}} |\, a \,\rangle\langle\, a \,| \\
&= \sum_{\substack{a_j \in \{0,1\} \\ j=0,\ldots,n-1}} \prod_{j=0}^{n-1} e^{-i2\pi ca_j/2^{n-j}} |\, a_{n-1} \ldots a_0 \,\rangle\langle\, a_{n-1} \ldots a_0 \,| \\
&= \bigotimes_{j=0}^{n-1} \sum_{a_j \in \{0,1\}} e^{-i2\pi ca_j/2^{n-j}} \big|\, a_j \,\big\rangle\big\langle\, a_j \,\big|
\end{aligned} \tag{6}$$

For interpretation in circuit gates, (6) corresponds to z–rotations by an angle $\theta = {}^{2\pi c}/_{2^{n-j}}$ on every jth qubit in the qubit register. In general, we define a $c-$angle $Z-$rotation as follows.

$$\hat{R}_m(c) = \begin{bmatrix} 1 & 0 \\ 0 & e^{i2\pi c/2^m} \end{bmatrix}, \qquad m \in \mathbb{N} \tag{7}$$

Therefore, the total number of gates required scales linearly as the number of qubits. In particular, only $\hat{R}_Z(\theta)$ gates are need for the addition of an unsigned integral register to a constant integer.

C. Addition of Two Unsigned Integral Registers

Given two qubit registers $|a\rangle$ and $|b\rangle$, each encoding unsigned integers and of width n, the addition of the two registers can be achieved using the same logic as in the case of adding an integer to a qubit register, except that the discrete Hamiltonian is modified to be $\hat{X} \otimes \hat{P}$.

$$e^{-i2\pi\hat{X}\otimes\hat{P}/2^n}|a, b\rangle = |a, (a + b) \bmod 2^n\rangle \tag{8}$$

We diagonalize the momentum operator using the Fourier Transform, denoting the identity operator by $\hat{I}$.

$$e^{-i2\pi\hat{X}\otimes\hat{P}/2^n} = \left(\hat{I} \otimes \hat{F}^\dagger\right) e^{-i2\pi\hat{X}\otimes\hat{X}/2^n} \left(\hat{I} \otimes \hat{F}\right) \tag{9}$$

We once again proceed to transform the inner diagonal operator until we obtain an expression that can be implemented with controlled single qubit gates, as follows.

$$e^{-i2\pi\hat{X}\otimes\hat{X}/2^n} = \sum_{a=0}^{2^n-1} \sum_{b=0}^{2^n-1} e^{-i2\pi ab/2^n} |\, a, b \,\rangle\langle\, a, b \,| \tag{10a}$$

$$= \sum_{a=0}^{2^n-1} \sum_{b=0}^{2^n-1} \prod_{k=0}^{n-1} \prod_{j=0}^{n-1} e^{-i2\pi a_k b_j/2^{n-j-k}} |\, a, b \,\rangle\langle\, a, b \,| \tag{10b}$$

$$= \sum_{a=0}^{2^n-1} \sum_{b=0}^{2^n-1} \prod_{k=0}^{n-1} \prod_{j=0}^{n-k-1} e^{-i2\pi a_k b_j/2^{n-j-k}} |\, a, b \,\rangle\langle\, a, b \,| \tag{10c}$$

The upper limit of the innermost product in (10c) was reduced because the exponential term equals one when $j + k \geq n$. In equation (10c), the values of the $Z-$rotations in the qubit register encoding $|b\rangle$ depends on the values of qubit register encoding $|a\rangle$. Notice also that the exponential term is present

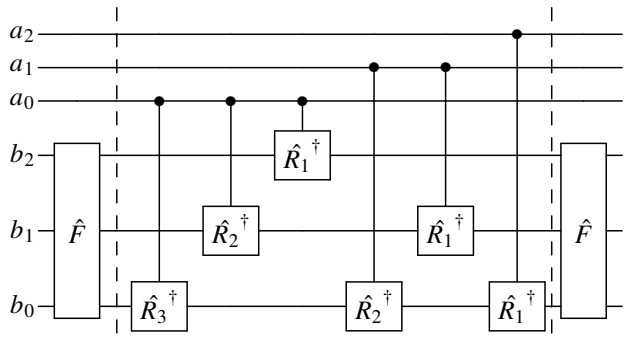

Fig. 1. Quantum circuit for implementing addition of two unsigned integral registers with $n = 3$ as seen in (10c).

only when $a_k = 1$. Therefore, the dynamics generated by $2\pi\hat{X} \otimes \hat{X}/2^n$ can be implemented with controlled $\hat{R}_m$ gates. For each index $k \in \{0, \ldots, n-1\}$ in (10c), there are $n-k-1$ controlled $\hat{R}_m$ gates. Therefore, there is a total of $\frac{n}{2}(n+1)$ controlled $\hat{R}_m$ gates. Since the number of gates for the Quantum Fourier Transform scales as $O(n^2)$, the overall cost of this operation scales as $O(n^2)$ two-qubit gates.

D. Fused Multiplication-=Addition of Three Unsigned Integral Registers

For the multiplication operation, we modify the argument in [14] by providing a more accurate mathematical description of the multiplication operation, providing exact results for performing multiplication at the cost of increased qubit overhead

Consider three qubit registers encoding unsigned integers $|a\rangle$, $|b\rangle$, and $|c\rangle$, each of width n.The fused multiplication-addition of the three registers (modulo 2^n) can be achieved by with the discrete Hamiltonian set to $\hat{X} \otimes \hat{X} \otimes \hat{P}$.

$$e^{-i2\pi\hat{X}\otimes\hat{X}\otimes\hat{P}/2^n}|a,b,c\rangle = |a,b,ab+c \bmod 2^n\rangle \tag{11}$$

We once again diagonalize the operator with the Quantum Fourier Transform.

$$e^{-i2\pi\hat{X}\otimes\hat{X}\otimes\hat{P}/2^n} = \left(\hat{I}\otimes\hat{I}\otimes\hat{F}^\dagger\right)e^{-i2\pi\hat{X}\otimes\hat{X}\otimes\hat{X}/2^n}\left(\hat{I}\otimes\hat{I}\otimes\hat{F}\right) \tag{12}$$

The procedure for deriving the simplified expression is essentially the same as the one for addition, except that instead we factor out the sums over $|a\rangle$ and $|b\rangle$, and the outermost outer product is the outer product over $|a,b\rangle$. The resulting expression is given by the following equation.

$$e^{-i2\pi\hat{X}\otimes\hat{X}\otimes\hat{X}/2^n} = \sum_{a,b,c=0}^{2^n-1}\prod_{l,k,j=0}^{n-1} e^{-i2\pi a_l b_k c_j/2^{n-j-k-l}}|\,a,b,c\,\rangle\langle\,a,b,c\,| \tag{13}$$

Note that the exponential term in the last equation equals 1 when $n \leq j+k+l$. The cases of interest are when $n \geq l+k+j+1$, which implies that $j \in \{0, \ldots, n-(l+k+1)\}$. Consequently, $l+k \in \{0, \ldots, n-1\}$. Therefore, for a fixed n, there are $(l-k-j)$ doubly controlled $\hat{R}_m$ gates where $l \in \{0, \ldots, n-1\}$, $k \in \{0, \ldots, n-(l+1)\}$, and $j \in \{0, \ldots, n-(l+k+1)\}$. Thus, the total number of doubly controlled gates is $\frac{n}{6}(n^2+3n+2)$.

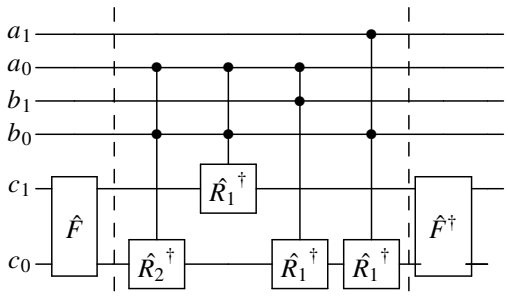

Fig. 2. Quantum circuit for implementing fused multiplication–addition of three unsigned integral registers with $n = 2$ as seen in (10c).

E. Signed and Fractional Encodings

1) Two's Complement Signed Integral Encoding: The discussion so far has focused on unsigned integer encoding. Two's Complement encoding scheme is used to represent superpositions of signed integers on quantum registers. The pertinent details of this encoding scheme are as follows. Given an unsigned integer of the form

$$a = \sum_{k=0}^{n-2} a_k 2^k \equiv a_{n-2}\ldots a_0 \tag{14}$$

the corresponding Two's complement signed integer has a value given by the following equation.

$$b_{n-1}b_{n-2}\ldots b_0 \equiv -b_{n-1}2^n + \sum_{k=0}^{n-2} b_k 2^k \tag{15}$$

where

$$b_{n-1}b_{n-2}\ldots b_0 = (\tilde{a}_{n-1}\tilde{a}_{n-2}\ldots\tilde{a}_0) + 1. \tag{16}$$

The operation '$\sim$' flips the bit value, while the $(n-1)th$ bit is the sign bit (initially set to zero before the flip operation). This encoding scheme is beneficial because addition and multiplication of Two's Complement signed integers corresponds exactly to the addition and multiplication of unsigned integers composed of the same sequence of bits.

2) Fixed-Point Fractional Encoding: To represent fractional numbers, a fixed point scheme is adopted. More precisely, this involves scaling an unsigned n-bit fixed-point number by a factor of 2^{-f} as shown in the following equation.

$$2^{-f}(a_{n-1}, a_{n-2}\ldots a_0) = a_{n-1}a_{n-2}\ldots a_f \cdot a_{f-1}\ldots a_0 \tag{17}$$

Here, f denotes the number of bits that compose the fractional part of the number, or the number of bits after the binary point. Henceforth, we denote n-bit fixed-point numbers with f fractional bits as (n, f)-bit fixed-point numbers.

3) Two's Complement, Fixed-Point Fractional Encoding: Signed fractional numbers are represented using the Two's Complement representation of the fixed-point scheme. The value of an n-bit Two's Complement, fixed-point number with f fractional bits, is given by the following equation, which

Registers: input $|q\rangle \in (n, f)$; output $|r\rangle \in (n, f)$;
ancillae $|s\rangle, |t\rangle \in (n, f)$, $|a\rangle \in (f, f)$.
Let $|r'\rangle = |r, a\rangle \in (n, 2f)$ and $|s'\rangle = |s, a\rangle \in (n, 2f)$

$|q, r, s\rangle \leftarrow \textsc{Init}(|q, r, s\rangle)$ ▷ See Alg. 4
for $i \leftarrow 1$ **to** L **do**
  $|s, a\rangle \leftarrow |1, 0\rangle$ ▷ Reset & Copy
  $|q, r, s'\rangle \leftarrow |q, r, s' - q\, r\rangle$ ▷ FMA† #1
  $|r, t, a\rangle \leftarrow |r, r, 0\rangle$ ▷ Reset & Copy
  $|r', s, t\rangle \leftarrow |r' + s\, t, s, t\rangle$ ▷ FMA #2
end for

Alg. 1. Proposed reciprocal gate for L Newton Iterations.

involves a minor modification of the expression for the value of Two's Complement integers in (15).

$$a_{n-1} \dots a_f \cdot a_{f-1} \dots a_0 \equiv -a_{n-1} 2^{n-1-f} + 2^{-f} \sum_{k=0}^{n-2} a_n 2^k \quad (18)$$

Similar to integers, one can think of signed fixed-point fractional numbers as signed integers composed of the same sequence of bits, scaled by a factor of 2^{-f}.

F. Fused Multiplication, Addition and Scaling of Three Unsigned Fixed–Point Registers

Addition and multiplication are quite straightforward operations for unsigned fixed–point numbers. However, the number of qubits necessary to store the output to full precision increases linearly with respect to the fractional width of fixed–point numbers. The authors of [14] imposed a rounding condition to deal with this issue. As we will demonstrate in the following section, this choice results in significant excessive rounding, thereby compromising the accuracy of the multiplication of fixed–point registers.

III. Methods

A. Proposed Algorithm for Fused Multiplication–Addition of Three Fixed-Point Registers

We start by noting that rounding up the multiplication of two unsigned $(n, f)-$bit fixed–point numbers a and b can be expressed by scaling the integer value by a power of two and using the floor function, as follows. Suppose the unsigned integers $\tilde{a} = 2^{-f} a$ and $\tilde{b} = 2^{-f} b$ are represented in the $(n, f)-$bit fixed point register, then the direct product $c = \tilde{a}\tilde{b}$ has $2f$ bits after the dot. To represent c in the same register requires discarding the bits following the first f bits after the dot. Note that there are still $n - f$ bits before the dots since the arithmetic operations are represented modulo 2^{n-f}. This can be achieved in binary representation by shifting the bits to the right $f-$places as done in [14], i.e.

$$\tilde{c} = \tilde{a}\tilde{b} \gg f = \left\lfloor \frac{ab}{2^f} \right\rfloor. \quad (19)$$

The operation '$\gg f$' denotes the bit shift to the right by f bits or the integer division by 2^{-f}.

We can express the floored product as the convolution of the bits composing a and b, and can then split the resulting sum into two parts: the integral part and the floored fractional part.

$$\begin{aligned} \tilde{c} &= \left\lfloor \sum_{l=0}^{n-1} \sum_{k=0}^{n-1} a_l b_k 2^{l+k-f} \right\rfloor \\ &= \sum_{\substack{l=0 \\ k+l \geq f}}^{n-1} \sum_{k=0}^{n-1} a_l b_k 2^{k+l-f} + \left\lfloor \sum_{\substack{l=0 \\ k+l \leq f-1}}^{n-1} \sum_{k=0}^{n-1} a_l b_k 2^{k+l-f} \right\rfloor \end{aligned} \quad (20)$$

The term that involves the summation constrained by $k + l \leq f - 1$ is precisely the term that was set to 0 after rounding in [14], but in general, this term is not equal to 0, requiring more careful treatment of the multiplication operation.

We propose instead to augment the output register $|c\rangle$ with f ancilla qubits on the right, and then to discard or reuse the ancilla qubits after the fused multiplication–addition. More precisely, we define a new $(n + f, 2f)-$bit fractional output integer $\bar{c}$ whose value is given by the following

$$\begin{aligned} \bar{c} &= c_{n-1} c_{n-2} \dots c_f \cdot c_{2f-1} \dots c_f c_{f-1} \dots c_0 \\ &\equiv \sum_{\substack{l=0 \\ k+l \geq f}}^{n-1} \sum_{k=0}^{n-1} a_l b_k 2^{k+l-f} + \sum_{\substack{l=0 \\ k+l \leq f-1}}^{n-1} \sum_{k=0}^{n-1} a_l b_k 2^{k+l-f} \end{aligned} \quad (21)$$

Note that there are exactly $2f$ bits in the second sum in (21). This is made possible by appending additional f ancilla bits to represent the bits $\{c_{f-1} \dots c_0\}$. Then, multiplication involving this new augmented output register involves a slight modification of (13), as follows.

$$\begin{aligned} e^{-i2\pi \hat{M}_f \otimes \hat{X}/2^n} = &\sum_{a=0}^{2^n-1} \sum_{b=0}^{2^n-1} |\, a, b \,\rangle\langle\, a, b \,| \\ &\bigotimes_{j=0}^{n-1} \sum_{c_j \in \{0,1\}} \prod_{k=0}^{n-1} \prod_{l=0}^{n-1} e^{-i2\pi a_l b_k c_j / 2^{n-j-k-l+f}} \big|\, c_j \,\big\rangle\big\langle\, c_j \,\big| \end{aligned} \quad (22)$$

where M_f is the newly defined multiplication operator.

B. Proposed Algorithm for Calculating Reciprocal of Fixed-Point Registers

The only remaining elementary operation that is necessary for numerical linear algebra over superpositions of fixed-point values on quantum registers is division. As a first step for calculating division, we propose applying Newton's Method

Registers: input $|q\rangle = |q_{n-1} \dots q_0\rangle \in (n, f)$;
output $|r\rangle = |r_{n-1} \dots r_0\rangle \in (n, f)$.

$|q, r\rangle \leftarrow |q, q\rangle$ ▷ Reset & Copy
for $n \leftarrow (n - 1)$ **to** 0 **do**
  $|r_n\rangle \leftarrow \hat{X}|r_n\rangle$ cont. by $|q_{n-1}\rangle$ ▷ Cont. X
end for
$|r\rangle \leftarrow |r + 1\rangle$ cont. by $|q_{n-1}\rangle$ ▷ Cont. Add

Alg. 2. Proposed absolute value gate for Two's Complement fixed–point registers.

Registers: input $|q\rangle = |q_{n-1} \dots q_0\rangle \in (n, f)$;
output $|r\rangle = |r_{n-1} \dots r_0\rangle \in (n, f)$.

$|r_{n-1}\rangle \leftarrow \hat{X}|r_{n-1}\rangle$ cont. by $|q_{n-1}\rangle$ ▷ Cont. X
for $k \leftarrow (n-2)$ **to** 0 **do**
  $|r_{n-1}\rangle \leftarrow \hat{X}|r_{n-1}\rangle$ ▷ Flip MSB
  $|r_k\rangle \leftarrow \hat{X}|r_k\rangle$ cont. by $|q_k, r_{n-1}\rangle$ ▷ Cont. X
  $|r_{n-1}\rangle \leftarrow \hat{X}|r_{n-1}\rangle$ ▷ Undo flip
  $|r_{n-1}\rangle \leftarrow \hat{X}|r_{n-1}\rangle$ cont. by $|r_k\rangle$ ▷ Save
end for
$|q_{n-1}\rangle \leftarrow \hat{X}|q_{n-1}\rangle$ ▷ Temp. flip MSB
$|r_{n-1}\rangle \leftarrow \hat{X}|r_{n-1}\rangle$ cont. by $|q_{n-1}\rangle$ ▷ Flip if needed
$|q_{n-1}\rangle \leftarrow \hat{X}|q_{n-1}\rangle$ ▷ Undo flip
for $n \leftarrow 0$ **to** $\lfloor n/2 \rfloor$ **do**
  $|r_n, r_{n-1-n}\rangle \leftarrow |r_{n-1-n}, r_n\rangle$ ▷ Reverse
end for

Alg. 3. Proposed gate for rounding to a power of 2 and calculating the inverse of this power of 2 via bit reversal.

to find the reciprocal of fixed-point values by finding the roots of the following function, as proposed in [19] in the classical computing context.

$$f(x) = \frac{1}{x} - D \tag{23}$$

D here is the value of the number for which we would like to find the reciprocal. The gradient is given simply by

$$f'(x) = -\frac{1}{x^2}. \tag{24}$$

Consequently, the Newton update step is given by the following equation.

$$x_{k+1} \leftarrow x_k - \frac{f(x_k)}{f'(x_k)} = x_k + x_k(1 + Dx_k) \tag{25}$$

This update step can thus be implemented with two fused multiplication–addition operations, and the process can be repeated until we reach the desired error tolerance. Because quantum computations are not branching–friendly, we opt to fix the number of iterations L, while ensuring that the initial guess is close to the desired reciprocal. Algorithm 1 contains more details on the operations of this gate.

Finding an initial guess for the Newton iterations that is close enough to the eventual result is crucial, because otherwise the method might not converge. We find the initial guess for the reciprocal operation by formulating yet another algorithm that scans the input register qubit-by-qubit while keeping track of the most significant qubit that has been found. However, before we describe this procedure, we must first describe the absolute value operation for fixed-point registers: If the input is a signed register, we first calculate the absolute value of the signed register, which we detail in Alg. 2. The most significant bit of the input register $|q\rangle$ controls all of the operations, namely addition with constant 1 and inversion via $\hat{X}$ gates. Because the chosen representation is Two's Complement, this operation is self-adjoint. Additionally, when we state that addition is controlled by qubit $|q_{n-1}\rangle$, we mean

Registers: input $|q\rangle \in (n, f)$; outputs $|r\rangle, |s\rangle \in (n, f)$.

$|q, r\rangle \leftarrow$ ABS$(|q, r\rangle)$ ▷ Alg. 2
$|r, s\rangle \leftarrow$ INVRECIP2$(|r, s\rangle)$ ▷ Alg. 3
$|r, s\rangle \leftarrow |s, r\rangle$ ▷ Swap
$|q, r\rangle \leftarrow$ ABS$^\dagger(|q, r\rangle)$ ▷ Undo Alg. 2

Alg. 4. Proposed procedure for the initialization of the reciprocal gate.

Registers: input $|q\rangle \in (n, f)$; input/output $|r\rangle \in (n, f)$.
Let $\mathcal{I} = [0, 2^n)$ if unsigned or $\left[-2^{n-1}, 2^{n-1}\right)$ otherwise.

for $k \in \mathcal{I}$, $l \in \mathcal{I}$ **do**
  $|q, r\rangle \leftarrow \left|k2^{-f}, l2^{-f}\right\rangle$ ▷ Reset & Copy
  $|q, r\rangle \leftarrow |q, r + q\rangle$ ▷ Addition
  $s \leftarrow$ MEASURE$(|s\rangle)$
  $\hat{s} \leftarrow (k + l) \cdot 2^{-f}$
  COMPARE$(s, \hat{s})$
end for

Alg. 5. Proposed scheme for the exhaustive validation of the addition gate that was originally proposed in [14].

that all of the operations for addition are controlled by this qubit.

Given the absolute value of the input register $|q\rangle$ that is stored on register $|r\rangle$, we now proceed to describe the proposed approach for finding the initial guess of the reciprocal operation. For simplicity, we assume there are $n = 2f + 1$ total qubits and f fractional qubits. The algorithm can be extended to the case where the two sets about the decimal point have unequal lengths when disregarding the sign qubit.

Given input register $|q\rangle$ and output register $|r\rangle$, we first ensure that the output register is in state $|0\rangle$ with a reset, and then apply a $C\hat{X}$ gate to the most significant qubit $|r_{n-1}\rangle$, controlled by the most significant or sign qubit of the input $|q_{n-1}\rangle$. Qubit $|r_{n-1}\rangle$ stores whether we have encountered a non-zero qubit in the input. Then, for $k = n - 2$, we invert $|r_{n-1}\rangle$ with an $\hat{X}$ gate, apply a $CC\hat{X}$ gate to $|r_k\rangle$, controlled by $|q_k, r_{n-1}\rangle$, apply another $\hat{X}$ gate to $|r_{n-1}\rangle$ to undo the inversion, and then apply a $C\hat{X}$ gate once again to $|r_{n-1}\rangle$, controlled by $|r_k\rangle$, effectively storing whether we encountered a non-zero qubit. The process is then repeated for $k \leftarrow n-3$ to 0. At the end of this loop, we flip qubit $|r_{n-1}\rangle$ if qubit $|q_{n-1}\rangle$ was not active to begin with by inverting $|q_{n-1}\rangle$ with an $\hat{X}$ gate, inverting $|r_{n-1}\rangle$ with a $C\hat{X}$ gate, controlled by $|q_{n-1}\rangle$, and apply an $\hat{X}$ gate to $|q_{n-1}\rangle$ to undo the inversion. With the power of 2 stored on output register $|r\rangle$, we reverse the order of the qubits so as to obtain the reciprocal of said power of 2. Algorithm 3 describes this process in more detail.

Finally, we have all of the necessary gates and operations for initializing the reciprocal operation. We calculate the absolute value of the input, obtain the nearest power of 2 and invert it by reversing the order of the qubits, undo the absolute value operation, and then perform swaps and resets to ensure that the registers are initialized properly. Algorithm 4 provides more details on this procedure.

Registers: inputs $|q\rangle, |r\rangle \in (n, f)$; input/output $|s\rangle \in (n, f)$.
Let $\mathcal{I} = [0, 2^n)$ if unsigned or $\left[-2^{n-1}, 2^{n-1}\right)$ otherwise.

for $k \in \mathcal{I}$, $l \in \mathcal{I}$, $m \in \mathcal{I}$ **do**
  $|q, r, s\rangle \leftarrow \left|k2^{-f}, l2^{-f}, m2^{-f}\right\rangle$ ▷ Reset & Copy
  $|q, r, s\rangle \leftarrow |q, r, s + q\, r\rangle$ ▷ FMA
  $s \leftarrow \textsc{Measure}(|s\rangle)$
  $\hat{s} \leftarrow \lfloor \left((m \cdot 2^f) + k\, l \right) \cdot 2^{-f} \rfloor \cdot 2^{-f}$
  $\textsc{Compare}(s, \hat{s})$
end for

Alg. 6. Proposed scheme for the exhaustive validation of the fused multiplication–addition (FMA) gate.

C. Assessment of Accuracy of Addition and Multiplication Schemes

We assess the correctness of the implemented addition scheme and our proposed multiplication scheme by exhaustively simulating the gate under consideration for all possible combinations of inputs. More specifically, for addition, we implemented the procedure described in Alg. 5, whereby we iterated over the outer product of the domain of each input to the addition gate, checking the correctness of each output, a process that scales as $O\left(2^{2n}\right)$ for n-bit inputs. Similarly, for fused multiplication–addition, we implemented the procedure described in Alg. 6, once again checking the output over the outer product of all three inputs, which is a process that scales as $O\left(2^{3n}\right)$ for n-bit inputs.

D. Assessment of Accuracy of Proposed Division Scheme

The number of qubits required for the implementation of the division operation is higher due to the necessary number of ancilla qubits. Additionally, the procedure is not exact as it is an iterative algorithm. Consequently, we opted to sample the domain of the inputs and generate statistics based on the observed errors in the outputs, taking into account the effect of rounding due to fixed-point arithmetic. This differs from the approaches used for addition and multiplication, which are exhaustive and exact.

E. Simulation of Rudimentary Ordinary Differential Equation on a Simulated Quantum Computer

In order to validate our approach, we solved the same system of ordinary differential equations that is featured in [14]. Namely, we solved the following system of equations.

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{u} \quad \text{s.t. } \mathbf{u}\left(0^+\right) = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \tag{26}$$

The analytical solution of this system subjected to the prescribed initial conditions is given by

$$\mathbf{u}(t) = -\begin{bmatrix} \sin(t) \\ \cos(t) \end{bmatrix} u(t), \tag{27}$$

where the function $u(t)$ denotes the Heaviside step function. We solved the system with the trapezoidal rule –as opposed to Forward Euler as in [14]– for a variety of qubits after the decimal point. We fixed $n = f + 2$ and use Two's Complement registers so as to be able to represent ± 1 exactly.

TABLE I
SUMMARY OF STATISTICS OBSERVED WHEN VALIDATING FUSED ADDITION–MULTIPLICATION OVER UNSIGNED REGISTERS USING THE ROUNDING SCHEME PROPOSED BY ZANGER ET AL. [14].

n	f	failure rate	mean error	σ of error	max. |error|
4	1	23.438%	-0.9375	1.991	7.5000
4	2	46.875%	-0.9375	1.242	3.7500
4	3	64.453%	-0.6445	0.646	1.8750
5	1	24.219%	-1.9375	4.073	15.500
5	2	48.438%	-1.9375	2.533	7.7500
5	3	66.602%	-1.3320	1.312	3.8750
5	4	78.711%	-0.7871	0.643	1.9375

IV. Discussion

A. Error in Addition, Multiplication, and Division

Following the procedures for validation of addition and fused addition-multiplication that are described in Algs. 5 and 6, we tested exhaustively for all possible combinations of register widths (n, f), where $n \in [2, 3, 4, 5, 6]$ and $f \in [2, \ldots, n]$, and obtained 100% accuracy for the addition scheme that was originally proposed in [14] and 100% accuracy for the fused multiplication-addition scheme that we proposed. However, the fused multiplication–addition routine with the rounding condition that was proposed in [14] exhibited large error. Table I summarizes the observed statistics of the error for several representative widths (n, f) of unsigned registers. A "failure" was defined when the output of the FMA gate does not match the expected output exactly. Due to the modulo 2^{n-f} arithmetic, the error in the output can be quite large. Even when there was no overflow, the error rate of this gate was still unacceptably high for scientific computation. The error rate increased as the number of fractional qubits f was increased, matching our assessment of the deficiencies of this rounding approach.

For the proposed division algorithm, we simulated the division operation for 100 samples, for $L = 10$ Newton iterations, and for fractional qubits f of 6, 7, and 8. Figures 4a, 4b, and 4c show the distribution of the error in the output for 6, 7, and 8 fractional qubits, respectively. We observed a spread of the error about 0, with the majority of non-zero error at $\pm 2^{-f}$, as expected. As the number of fractional qubits was increased, we observed that the standard deviation also decreased. For f of 6, 7, and 8, we observed that the means of the error were $7.8125 \cdot 10^{-4}$, $1.0938 \cdot 10^{-3}$, and $-1.5625 \cdot 10^{-4}$, respectively, and that the standard deviations of the error were $1.02676 \cdot 10^{-2}$, $5.2107 \cdot 10^{-3}$, and $3.3276 \cdot 10^{-3}$, respectively, with the standard deviation decaying approximately as $O\left(2^{-f}\right)$.

B. Simulation of Ordinary Differential Equation

We simulated the same system of ordinary differential equations as in [14] with initial conditions $\mathbf{u}(0^+) = \begin{bmatrix} 0 & -1 \end{bmatrix}^{\mathrm{T}}$ for register widths

$$(n, f) \in \{(10, 8), (11, 9), (12, 10), (13, 11), (14, 12)\},$$

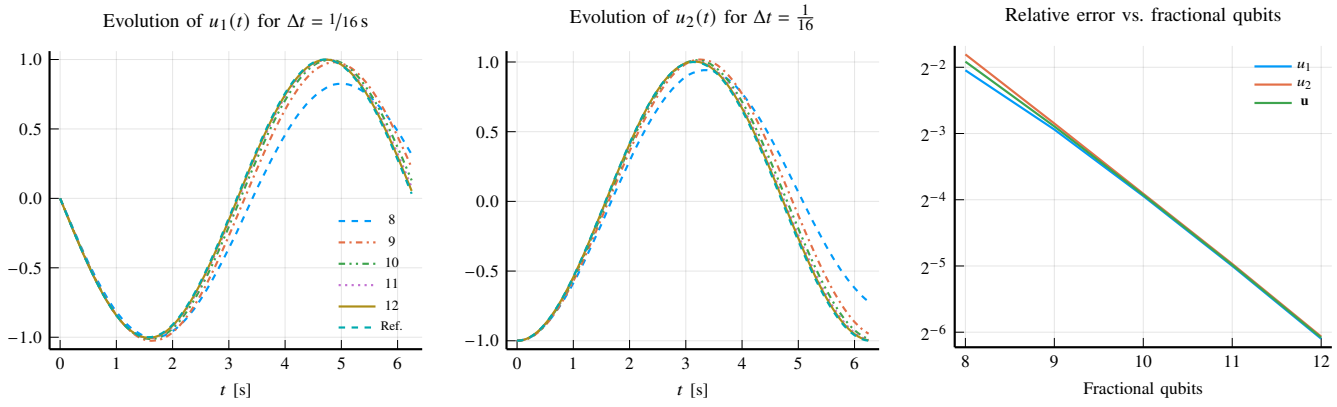

Fig. 3. Results from the solution of the tested system of ordinary differential equations for several fractional width f, with total width $n = f + 2$. Figs. (a) & (b) show the evolution over time of u_1 and u_2, respectively, for the tested values of f. Fig. (c) shows the relative error with respect to the analytical solution as a function of fractional width f.

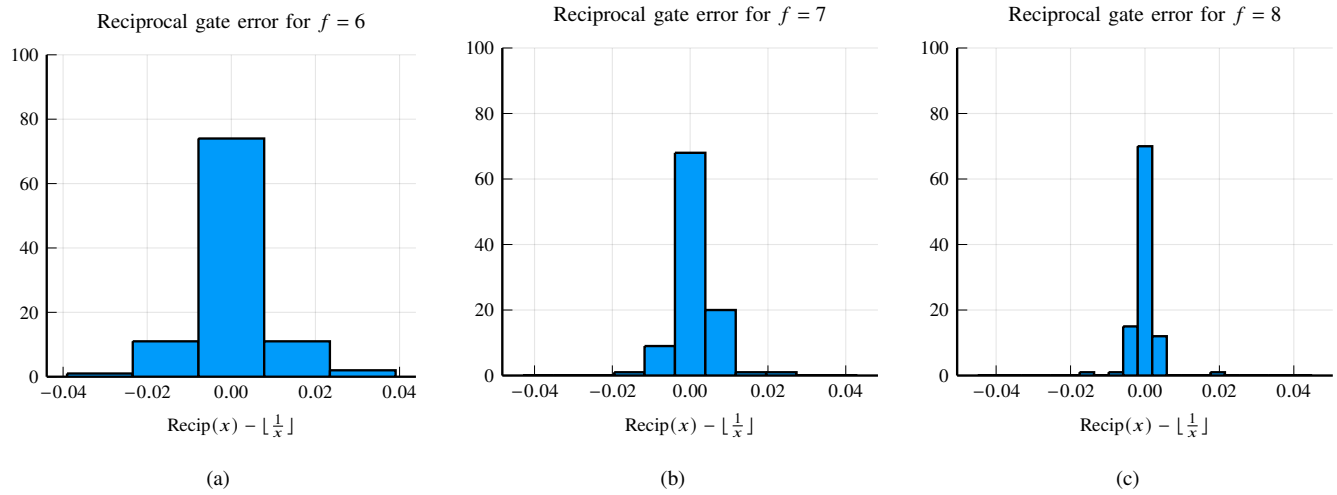

Fig. 4. Error of output of reciprocal gate when compared with expected output. The total number of qubits was constrained to $n = 2f + 1$ for f fractional qubits. Each gate was tested with 100 samples of a normal distribution $\mathcal{N}(0, 5)$ and for $L = 10$ Newton iterations. Samples that did not have a reciprocal that is representable after flooring were discarded, such as 0 and 2^{-f}.

while using a step size of $\Delta t = 1/16$ s. The explicit trapezoidal rule update step is given in the following equation.

$$\mathbf{u}_{k+1} \leftarrow \frac{1}{1 + \frac{\Delta t^2}{4}} \begin{bmatrix} 1 - \frac{\Delta t^2}{4} & \Delta t \\ -\Delta t & 1 - \frac{\Delta t^2}{4} \end{bmatrix} \mathbf{u}_k \tag{28}$$

When evaluated at the prescribed step size of $\Delta t = 1/16$ s, this equation reduces to

$$\mathbf{u}_{k+1} \leftarrow \frac{1}{1025} \begin{bmatrix} 1023 & 64 \\ -64 & 1023 \end{bmatrix} \mathbf{u}_k \tag{29}$$

Figure 3 demonstrates the evolution of the two components over 100 time steps or 6.25 s. We saw uniform convergence to the analytic solution as the number of fractional qubits per register was increased. Figure 3c shows the relative error in the L_2 sense of the solution when compared to the analytic solution in Eq. 27. The decay of the error as a function of f behaves asymptotically as $O(2^{-f})$ when compared to both the analytic solution and the solution that was obtained using the trapezoidal rule on a classical computer with 64-bit floating-point arithmetic. We expect that given enough fractional qubits, the error with respect to the analytic solution would saturate due to the use of the trapezoidal rule.

V. Conclusion

While fixed-point arithmetic is more limited than floating-point arithmetic in terms of accuracy as a function of register bit widths, fixed-=point arithmetic is significantly less complex to implement in terms of gate operations. In this work, we demonstrated that with the appropriate number of ancilla qubits, multiplication can be implemented accurately

for signed fixed=-point numbers using Hadamard and z-rotation gates. Building on this, we showed that given enough qubits to achieve the desired accuracy, solving an ordinary differential equation with fixed-point arithmetic is feasible, and we showed that extending the set of available operations to include division is also feasible.

Future work would involve exploring ways of rendering these operations robust to noise with error correction strategies. Improving the initialization procedure by obtaining a better initial guess for the iterative reciprocal gate would also be a fruitful future direction. Additionally, exploring the implementation of floating–point arithmetic with these basic fixed–point operations would also be an interesting avenue to explore. Another possible and motivating application of our approach is to explore the use of quantum computing to simulate the dynamics of bulk magnetization in Magnetic Resonance Imaging (MRI), which is used extensively in medical imaging.

The physics of MRI depends on applying an external, strong magnetic field B_0 to a tissue (on the order of 0.5–7 T), which aligns the nuclear spins of the tissue and induces a net bulk magnetization $\mathbf{M}$ that precesses about the direction of the static field at the Larmor frequency. The longitudinal and transverse components of $\mathbf{M}$ exhibit relaxation times T_2 and T_1, respectively, which dictate the decay rates of the components under only B_0. These relaxation times are parameters to the governing Bloch Equations. Obtaining a large set of solutions to the Bloch Equations with different parameters, such as T_1 and T_2, is a computationally demanding task on a classical computer. In future work, we aspire to leverage quantum parallelism to simulate the Bloch Equation for many different sets of external parameters simultaneously.

Acknowledgements

This material is based upon work supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science Research Centers, Superconducting Quantum Materials and Systems Center (SQMS) under the contract No. DE-AC02-07CH11359.

References

[1] C. Runge and M. W. Kutta, "Beiträge zur numerischen Lösung partieller Differentialgleichungen," *Mathematische Annalen*, vol. 46, no. 2, pp. 167–178, 1895.

[2] J. C. Adams and F. Bashforth, "On the numerical integration of functions," *Cambridge Philosophical Transactions*, vol. 10, pp. 200–210, 1855.

[3] L. Euler, "Methodus invariantibus integralium differentiarum aequationes," *Commentationes Arithmeticae*, vol. 9, pp. 193–228, 1768.

[4] L. F. Shampine and M. W. Reichelt, "Implementation of rosenbrock methods," *ACM Transactions on Mathematical Software (TOMS)*, vol. 12, no. 1, pp. 58–78, 1986.

[5] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM journal on computing*, vol. 26, pp. 1484–1509, 1997.

[6] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, pp. 505–510, 2019.

[7] D. R. Simon, "On the power of quantum computation," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1474–1483, 1997.

[8] L. K. Grover, "Quantum computers can search arbitrarily large databases by a single query," *Phys. Rev. Lett.*, vol. 79, pp. 4709–4712, 1997.

[9] A. W. Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical Review Letters*, vol. 103, no. 15, p. 150502, 2009.

[10] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, Aug. 2018. [Online]. Available: https://doi.org/10.22331/q-2018-08-06-79

[11] A. M. Childs, R. Kothari, and R. D. Somma, "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision," *SIAM J. Comput.*, vol. 46, pp. 1920–1950, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:3834959

[12] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, "Efficient quantum algorithms for simulating sparse hamiltonians," *Communications in Mathematical Physics*, vol. 270, pp. 359–371, 2005. [Online]. Available: https://api.semanticscholar.org/CorpusID:37923044

[13] D. W. Berry, A. M. Childs, and R. Kothari, "Hamiltonian simulation with nearly optimal dependence on all parameters," in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, 2015, pp. 792–809.

[14] B. Zanger, C. B. Mendl, M. Schulz, and M. Schreiber, "Quantum Algorithms for Solving Ordinary Differential Equations via Classical Integration Methods," *Quantum*, vol. 5, p. 502, Jul. 2021. [Online]. Available: https://doi.org/10.22331/q-2021-07-13-502

[15] S. Lloyd and S. L. Braunstein, "Quantum computation over continuous variables," in *Quantum Information with Continuous Variables*. Springer, 1999, pp. 9–17.

[16] J. Alvarez-Sanchez *et al.*, "A quantum architecture for multiplying signed integers," *Quantum Information Processing*, vol. 10, no. 6, pp. 775–789, 2011.

[17] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

[18] G. Verdon, J. Pye, and M. Broughton, "A universal training algorithm for quantum deep learning," *arXiv preprint arXiv:1912.08278*, 2019.

[19] U. Kucukkabak and A. Akkas, "Design and implementation of reciprocal unit using table look-up and newton-raphson iteration," in *Euromicro Symposium on Digital System Design, 2004. DSD 2004.*, 2004, pp. 249–253.