Quantum Circuit Designs of Point Doubling Operation for Binary Elliptic Curves

Harashta Tatimma Larasati and Howon Kim

School of Computer Science and Engineering
Pusan National University
Busan, Republic of Korea
tatimmaharashta@gmail.com, howonkim@pusan.ac.kr

Abstract. In the past years, research on Shor's algorithm for solving elliptic curves for discrete logarithm problems (Shor's ECDLP), the basis for cracking elliptic curve-based cryptosystems (ECC), has started to garner more significant interest. To achieve this, most works focus on quantum point addition subroutines to realize the double scalar multiplication circuit, an essential part of Shor's ECDLP, whereas the point doubling subroutines are often overlooked. In this paper, we investigate the quantum point doubling circuit for the stricter assumption of Shor's algorithm when doubling a point should also be taken into consideration. In particular, we analyze the challenges on implementing the circuit and provide the solution. Subsequently, we design and optimize the corresponding quantum circuit, and analyze the high-level quantum resource cost of the circuit. Additionally, we discuss the implications of our findings, including the concerns for its integration with point addition for a complete double scalar multiplication circuit and the potential opportunities resulting from its implementation. Our work lays the foundation for further evaluation of Shor's ECDLP.

Keywords: Elliptic curve discrete logarithm problem · Point doubling · Quantum circuit · Quantum cryptanalysis · Shor's algorithm

1 Introduction

Over the decade, there has been a growing interest in Shor's algorithm for solving the elliptic curve discrete logarithm problems (i.e., Shor's ECDLP) [17,18]. Acknowledged to render existing elliptic curve-based cryptosystems (ECC) breakable in polynomial time [16], this algorithm has the potential to accomplish its objective of cracking existing public-key cryptography (PKC) sooner than its more popular counterpart, i.e., Shor's factoring algorithm for cracking RSA, due to its lower quantum resource requirement for the same security level [9,14]. In particular, the advantage of lower key size in ECC is —ironically —the reason why it is in graver danger in the presence of a quantum computer, considering the current development of quantum computing that is still in the early stage, which often favors the number of qubits as the most essential metric.

To date, several works have discussed how to concretely realize Shor's ECDLP for quantum cryptanalysis purposes [16,5,1,3,11], with heavily referenced state-of-the-art works [16,5,1] primarily assessing the implementation for the superconducting qubits architecture as arguably the most prominent quantum hardware platform. Starting from the works by Roetteler et al. [16] and perfected by Haner et al. [5], which both consider prime curves implementation, the landscape then extends to binary elliptic curves by Banegas et al. [1],

All those advancements are based on the pioneering efforts of Proos and Zalka [14], one of the earliest works to translate the high-level Shor's ECDLP algorithm into the description of their possible quantum circuit derivation. Over time, their paper has established itself as the standard reference for subsequent papers in the literature that aims to optimize the quantum circuit implementation of Shor's ECDLP, which has been made easier for testing, verification, and concretely estimating the quantum resource requirement by leveraging reversible circuit and quantum computing simulators that have emerged in the past decade (e.g., RevKit, LIQ$Ui\,|\rangle$, and the more recent ProjectQ, Qiskit, Microsoft QDK/Azure Quantum, and Q-Crypton).

From our observation, these papers preserve the scope provided by Proos and Zalka [14]. That is, for cracking ECC via Shor's ECDLP, the rule can be simplified by considering only the generic case (i.e., for points $P + R$ where $P, R \neq O$, and $P \neq \pm R$) for the elliptic curve group operation [14]. In other words, to achieve the *double scalar multiplication*, the essential components in Shor's ECDLP circuit (see Fig. 1), computation will be done solely by a series of *point addition* operations. Meanwhile, the other operation to perform a more special case where $P = R$, namely the *point doubling* operation, is set aside. The authors of [14] argued that the expected loss of fidelity from the absence of this operation would still be negligible, which was also agreed upon by succeeding papers, e.g., [8].

Nevertheless, when considering the stricter assumption where the occurrence of $P = R$ is more probable during computation and minimum fidelity loss is expected from the construction, point doubling operation will also hold considerable significance. In this case, exploring the point doubling operation, including its quantum circuit construction and the analysis of its quantum resource, will be very beneficial and insightful for more precise resource estimation of Shor's ECDLP.

In this study, we examine the point doubling operation as required for the less relaxed case of Shor's ECDLP, i.e., when the elliptic curve points happen to be the same two points. To the best of our knowledge, this subject, including the possible quantum circuit implementation, has so far been absent in state-of-the-art works in quantum cryptanalysis. For this initial work, we focus on point doubling circuit for binary elliptic curves, whose inherent characteristics make it simpler for tinkering and constructing the operation compared to the prime curves counterpart. To highlight our contributions, we start by analyzing the point-doubling formula and identifying the challenges in its construction with their possible solution. Subsequently, we design the quantum circuits for elliptic curve point doubling to suit several scenarios and analyze its quantum resource

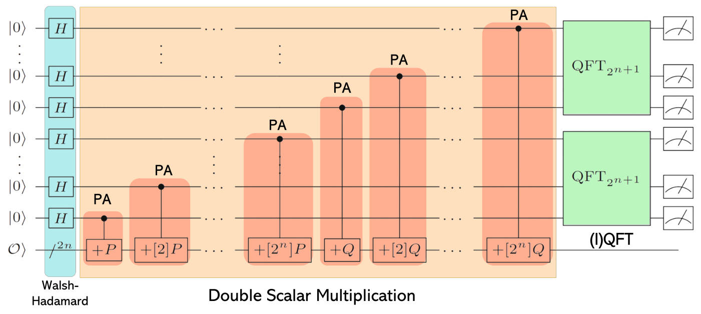


Fig. 1: Quantum circuit of Shor's algorithm for solving the elliptic curve discrete logarithm problem (ECDLP). Figures adapted from [16,10].

cost in a high-level view. Furthermore, we also provide a more detailed discussion of the aspects related to prime curves and the concerns when incorporating the circuit for use in Shor's algorithm.

The contribution of this paper can be summarized as follows:

- We examine the elliptic curve point doubling operation, which is rarely explored in literature. In particular, we discuss the challenges, analyze the formula and the implementation possibility of point doubling circuits for binary elliptic curves.
- We design the corresponding quantum circuit, incorporate several optimization and address the uncomputation, then analyze the high-level quantum resource cost of the circuit.
- We provide an in-depth discussion of our findings and other aspects relevant to point doubling, the concerns when incorporating the circuit with point addition for a complete double scalar multiplication circuit, as well as the open possibilities arising from point doubling implementation.

2 Preliminaries

2.1 Shor's ECDLP

The security of elliptic curve cryptography (ECC) is based on the hardness of the elliptic curve discrete logarithm problem (ECDLP). In this problem, given two points P and Q on an elliptic curve of order r, it is easy to compute the point multiplication $Q = kP$ when the scalar k and the base point P are known. In contrast, the reverse problem of finding the scalar k given both points P and Q is computationally intensive [16] and considered classically intractable.

How it works. Shor's algorithm for solving elliptic curve discrete logarithm problems (Shor's ECDLP) works by essentially running a brute-force attack of computing the scalar multiplication of all states, but intelligently utilizing quantum interference to boost the likelihood of obtaining the desired result while suppressing the undesired value via quantum Fourier transform (QFT). As illustrated in Fig. 1, the algorithm consists of three registers with two $n+1$-sized quantum registers initialized in the state $|0\rangle$ appended with the Walsh-Hadamard (i.e., Hadamard gate on each qubit), which yields the state $\frac{1}{2^{n+1}}\sum_{k,l=0}^{2^{n+1}-1} k, l$. Subsequently, conditional to the state of the register containing k or l, the corresponding multiple of points P and Q are added via the double-scalar multiplication circuit, performing the mapping as in Eq. 1 [16],

$$\frac{1}{2^{n+1}}\sum_{k,l=0}^{2^{n+1}-1} k, l \mapsto \sum_{k,l=0}^{2^{n+1}-1} k, l\,|[k]\,P + [l]\,Q\rangle \tag{1}$$

before appending QFT and measuring the result. Finally, classical post-processing is performed, which theoretically can yield the sought value with high probability. Consequently, this algorithm enables an adversary with a large-scale, full-fledged quantum computer to obtain k by running the algorithm a few times.

Quantum scalar multiplication circuit. In existing works, as previously shown in Fig. 1, the quantum (double) scalar multiplication circuit comprises solely of (controlled) point addition operation, simplifying the operation by making the added point fixed. However, this does not cover the case where both points are the same, which necessitate the use of point doubling operation, therefore may yield incorrect result when doubling the points [14]. Even though it is argued that the fidelity loss from this is small, the stricter case will require the analysis of the point-doubling circuit as well. Therefore, it is beneficial to analyze the point doubling circuit, which we start with this paper.

2.2 Binary Elliptic Curves in the Quantum Realm

From a quantum cryptanalysis perspective, an ordinary binary elliptic curve is often considered instead of other stronger variants such as supersingular [1]. Here, we first describe the theoretical concept of binary elliptic curves. The Weierstrass equation for an ordinary binary elliptic curve is described in Eq. 2,

$$y^2 + xy = x^3 + ax^2 + b \tag{2}$$

where $a \in \mathbb{F}_2$ and $b \in \mathbb{F^*}_{2^m}$ (i.e., the extension field). Then, the points on this elliptic curve, $P = (x, y) \in \mathbb{F}^{\not{2}}{}_{2^m}$, form a set of points that can be computed under the elliptic curve group law comprising *point addition* and *point doubling* operations. In particular, point addition, e.g., $P_1 + P_2 = P_3$, with $P_1 = (x_1, y_1)$, $P_2 = (x_2, y_2) \neq \pm P_1$, and $P_3 = (x_3, y_3)$, can be computed by following Eqs. 3 to 5.

$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \tag{3}$$

$$y_3 = \lambda(x_1 + x_3) + x_3 + y_1 \tag{4}$$

$$\lambda = \frac{y_1 + y_2}{x_1 + x_2} \tag{5}$$

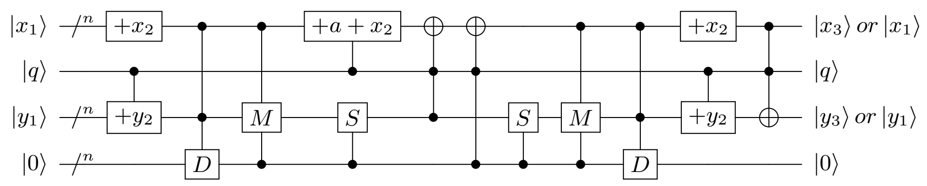


Fig. 2: Point addition circuit for binary elliptic curves proposed by Banegas et al [1].

Meanwhile, the point doubling calculation is as shown in Eqs. 6 to 8. [6,13].

$$x_3 = \lambda^2 + \lambda + a = {x_1}^2 + \frac{b}{{x_1}^2} \tag{6}$$

$$y_3 = {x_1}^2 + (\lambda + 1)x_3 \tag{7}$$

$$\lambda = x_1 + y_1/x_1 \tag{8}$$

Constructing the quantum circuit. From the group law formula above, the corresponding quantum circuit can be constructed. [1] Regarding the quantum point addition circuit, the recent concrete construction is by Banegas et al. [1], which is presented in Fig. 2. As inferred from the figure, the circuit requires three registers of size n in which two serve as input/output registers and one as a clean ancilla register, plus one qubit serving as the control —which in the full scheme of Shor's ECDLP circuit will be associated with the qubit in the upper registers (the ones appended by Walsh-Hadamard). Additionally, the circuit utilizes two multiplications, two divisions, and two squarings —all of which are conditionally controlled, linked to the control qubit and other associated register —and several (controlled) additions and addition by a constant.

Quantum resource cost. In terms of the exact resource count, however, it will greatly depend on the underlying subroutines employed since the aforementioned circuit is still a high-level architecture that will be broken down into its finer-grained components. For instance, choosing to use between two different inversion techniques: greatest common divisor (GCD) [1] or Fermat's Little Theorem (FLT) [1,10,19] for the division subroutines, or between Schoolbook [21] and Karatsuba multiplication [20,15,2,7] will yield quite different performance metrics, including in terms of the total number of qubits (i.e., qubit count or circuit width), circuit depth (i.e., the longest path for the quantum operations to run on the quantum hardware, gate count (i.e., the total number of quantum gates), as well as the more specific terms like Toffoli depth and Toffoli count [4,22].

[1] All classical computation can be simulated on a quantum computer by reversible gates, e.g., Toffoli (the most common), Fredkin, or Barenco gates [23]. However, how to efficiently perform the operation is a whole different topic pursued by researchers.

3 Quantum Circuit Designs of Point Doubling Operation for Binary Elliptic Curves

In this section, we start by elaborating on the challenges in constructing point-doubling operations. Furthermore, we provide three circuits for point doubling to suit different design considerations. In this work, we aim to be clear also for non-expert audiences; therefore, we describe our thought process to develop the resulting circuit.

3.1 Challenges on Quantum Point Doubling Construction

Before going into detail about the point-doubling circuit itself, it would be better to start with the differences between point addition and point doubling from the quantum perspective that we are able to identify. Constructing a point-doubling circuit poses relatively more difficulties than a point-addition circuit. Firstly, to implement point addition, previous works [14] proposed simplification by making one of the two points constant, which is added conditionally depending on the state of the control qubit (which represents each qubit that is appended by Hadamard gates in the upper registers of Shor's ECDLP, see Fig. 1).

With this, the point to be added (i.e., $P_2(x_2, y_2)$) is appended conditionally as a constant; hence can be pre-set and precomputed classically. Furthermore, by making the point a constant, the uncomputation process can be performed with ease since the added point can be immediately subtracted or uncomputed as soon as they are no longer needed in the calculation, making it practical and more efficient. Secondly, as mentioned in [14] and further elaborated in [16], by looking further at the point addition formulas (Eqs. 3 to 5), the value of λ in point addition has a direct, clear relation with both x_1 and x_3, as well as y_1 and y_3 (i.e., x_3 can be obtained from appending x_2 and λ to x_1 with other relevant operations (Eq. 3), and similarly, y_3 can be obtained from appending y_1 and λ to y_1 with other relevant operations (Eq. 4)). Here, we say that the initial state of x_1 and y_1 can be "consumed" to obtain the final desired computation. Then, by intelligently arranging the circuit, we can straightforwardly transform the initial state (x_1, y_1) to the subsequent state (x_3, y_3). As a result, an efficient computation (and uncomputation) can be achieved, and a clear reversibility relationship can be maintained.

On the other hand, the construction of point doubling is relatively tricky. First, we will discuss point doubling in a broader view without restricting ourselves to the case of Shor's algorithm requirement. In the case of point doubling, both points involved in the computation share identical values ($P = R$). Unlike point addition, where it is reasonable to assume that the second point is constant and its value is known in advance, the same assumption does not hold for point doubling. Intuitively, if the point to be doubled were known beforehand, then the whole point doubling operation would serve no purpose.

Hence, the practice of appending the value of the second point, as seen in point addition, is not applicable in this scenario. Consequently, an extra placeholder (register) will be required to store or append the same point, which can be achieved through a "fan-out" or "copy" operation using CNOT gates. Moreover, due to both input points being quantum and the operations being conditionally dependent on the state of a controlled qubit q, many of the operations will ultimately require "elevation,": CNOT becomes CCNOT (controlled-controlled NOT gate a.k.a. Toffoli gate), CCNOT becomes CCCNOT (multi-controlled Toffoli gate), and so on, leading to more complex operations.

Furthermore, examining the point doubling formula in Eqs. 6 to 8, obtaining x_3 from x_1 and y_3 from y_1, is not as straightforward. The term x_1 does not directly evolve into x_3, and similarly for y_1 and y_3. In detail, as inferred from Eq. 6, obtaining x_3 from x_1 requires "copying" x_1 to be squared and then appended (i.e., $x_1^2 + \frac{b}{x_1^2}$), while obtaining it from λ does not require any x_1. Hence, we say that it does not "consume" the initial state. Similarly for y_3 as obtaining it does not make use of y_1 at all. As a consequence, the initial value of y_1 may need to be preserved in the circuit as it can not be erased, hence requiring a placeholder (such as an ancilla register) to hold its value. This makes it challenging to devise an efficient design for its quantum circuit implementation.

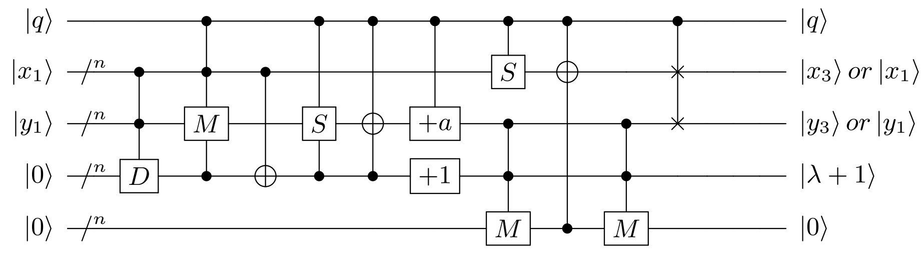
3.2 Proposed Quantum Circuits

Despite the challenges, there are still opportunities from the point-doubling formula that we can leverage to implement the circuit rather efficiently. We observe that there exists an indirect relation that can be taken advantage of. In particular, notice that x_1 has a direct relation to y_3, while y_1 has a direct relation to x_3. By utilizing this correlation, it is possible to transform x_1 and y_1 into y_3 and x_3, respectively. Thereby, a relatively efficient circuit can still be obtained, albeit with a "twisted" input-output relation (i.e., where x_1 maps to y_3 and y_1 maps to x_3 instead of the aligned mapping of x_1 to x_3 and y_1 to y_3).

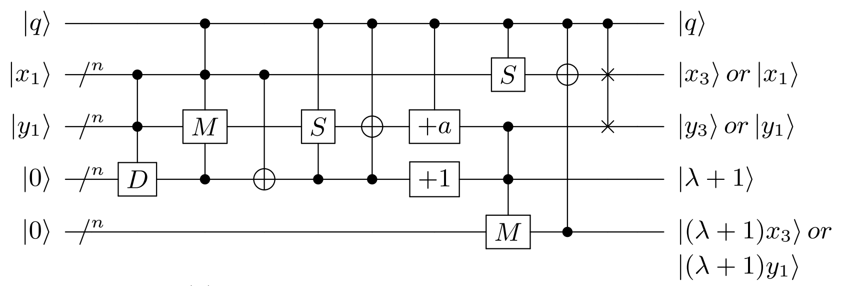
Fundamentally, there is no requirement for the input and the output to be aligned. However, considering the conditional nature of the computation (i.e., if the control qubit q is in the state zero, the doubling does not occur and the value remains as x_1 instead of being transformed into y_3) and the circuit will be incorporated into a larger scheme of scalar multiplication, a direct alignment will be helpful for clarity of the operations, which can be done simply by appending (controlled) swap gates.

Nevertheless, as previously described, the construction of point doubling may necessitate more space (i.e., ancilla registers) than that of point addition. While the latter, as proposed by Banegas et al. [1], requires one ancilla register used as a placeholder for division operation (see Fig. 2), two ancilla registers will be required for performing point doubling. Below, we provide three schemes of point-doubling circuits to suit different implementation preferences.

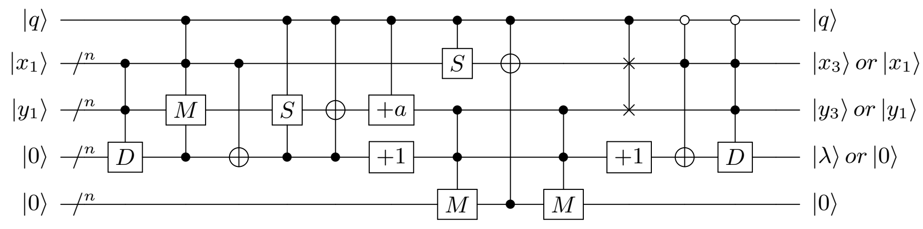
The proposed circuits for performing point doubling are illustrated in Fig. 3. These circuits consist of two n-sized input/output registers, a control qubit q, and two n-sized ancilla registers to store intermediate results. Additionally, the presence of multiple multi-controlled gates throughout the circuit results from the circuit's conditional nature, wherein it remains in the initial state (i.e., x_1 and y_1) when the control qubit is in the state $|0\rangle$. It is important to highlight that our proposal focuses on the high-level structure of the circuit arrangement, whereas the underlying field operations and subroutines (e.g., multiplication, squaring) may employ existing techniques such as Schoolbook or Karatsuba multiplication as proposed in [1,15,7], with necessary adjustments made to accommodate the number of qubits required on each construction. The state change corresponding to these circuits is presented in Table 1. In detail, the complete steps (up to line 15) are for the third scenario (Fig. 3c), while the second scenario (Fig. 3b) and the first scenario (Fig. 3a) terminate at lines 10 and 12, respectively.



(a) Proposed point doubling circuit, with clearing one ancilla register


$|q\rangle$
$|x_1\rangle$
$|y_1\rangle$
$|0\rangle$
$|0\rangle$
n
D
M
S
$+a$
$+1$
$|q\rangle$
$|x_3\rangle$ or $|x_1\rangle$
$|y_3\rangle$ or $|y_1\rangle$
$|\lambda + 1\rangle$
$|(\lambda + 1)x_3\rangle$ or
$|(\lambda + 1)y_1\rangle$

(b) Alternative 1: Without uncomputation



(c) Alternative 2: Fully uncompute when $q = 0$, otherwise leaving one ancilla as λ

Fig. 3: Our proposed point doubling circuits for binary elliptic curves: (a) balanced version that clears one ancilla registers, and two alternatives of (b) without uncomputation for lower depth and lower gate count, and (c) full uncomputation when control qubit $q = 0$, and with a garbage ancilla in state λ when $q = 1$.

Table 1: Point Doubling State Change

Step	$q = 1$	$q = 0$
1	$anc_1 = \frac{y_1}{x_1}$	$anc_1 = \frac{y_1}{x_1}$
2	$y = 0$	$y = y_1$
3	$anc_1 = \frac{y_1}{x_1} + x_1 = \lambda$	$anc_1 = \frac{y_1}{x_1} + x_1 = \lambda$
4	$y = \lambda^2$	$y = y_1$
5	$y = \lambda^2 + \lambda$	$y = y_1$
6	$y = \lambda^2 + \lambda + a = x_3$	$y = y_1$
7	$anc_1 = \lambda + 1$	$anc_1 = \lambda + 1$
8	$anc_2 = (\lambda + 1)x_3$	$anc_2 = (\lambda + 1)y_1$
9	$x = {x_1}^2$	$x = x_1$
10	$x = {x_1}^2 + (\lambda + 1)x_3 = y_3$	$x = x_1$
11	$anc_2 = 0$	$anc_2 = 0$
12	$swap : x = x_3, y = y_3$	$none : x = x_1, y = y_1$
13	$anc_1 = (\lambda + 1) - 1 = \lambda$	$anc_1 = \lambda$
14	$anc_1 = \lambda$	$anc_1 = \lambda - x_1 = \frac{y_1}{x_1}$
15	$anc_1 = \lambda$	$anc_1 = 0$

In the first circuit (Fig. 3a), we propose a balanced approach that strikes a tradeoff between the number of operations and the need to clear the ancilla qubit. While this circuit involves a relatively smaller number of operations, it requires an additional multiplication circuit for performing uncomputation upon one of the ancilla registers. As a result, we obtain one cleared ancilla register that can be used for subsequent computations, and one dirty (i.e., does not revert to its initial state after use) ancilla register in the state $(\lambda + 1)$. This is our favored version because we can secure one clean register with relatively minimal effort.

Alternatively, if circuit depth and gate count take precedence over qubit count, the more suitable circuit would be as illustrated in Fig. 3b. Here, the circuit only performs the expected point doubling operation without considering any uncomputation for ancilla registers. This minimizes depth and the number of subroutines, but we are left with two dirty ancilla registers.

Regarding the third case, our initial goal was to clear all ancilla registers. However, we have not found a more efficient method to fully uncompute them for all possible states of the control qubit ($|0\rangle$ or $|1\rangle$). A complete uncomputation can be achieved when $q = 0$, but the state of λ remains a dirty ancilla when $q = 1$. Note that λ from the previous state (i.e., $x_1 + \frac{y_1}{x_1}$) may not be in the same value as λ in the subsequent operation (i.e., $x_3 + \frac{y_3}{x_3}$); due to this potential differences in value, we should not uncompute it by utilizing x_3 and y_3 when $q = 1$. Had it been the same, it would allow us to obtain two perfectly-uncomputed, clean ancilla registers. This can be done by appending another controlled multiplication circuit and Toffoli gate targeting that ancilla register.

Nevertheless, this third construction is still useful; Evidently, at the time when $q = 0$ indicates that the point doubling does not occur —meaning that most likely a point addition is taking place. This register can be repurposed using a clever arrangement to substitute the ancilla register in the point addition circuit (i.e., the one for performing division operation in Fig. 2). The uncomputation itself can be performed by appending a series of Toffoli gates, a multi-controlled and negative-controlled division operation, and an addition by a constant. This circuit serves as a beneficial alternative construction when circuit width or qubit utilization is the most prioritized quantum resource.

The high-level resource cost of the proposed circuits can be summarized as follows. Compared to point addition, point doubling construction employs one more ancilla register and significantly more controlled and multi-controlled operations due to its non-fixed second point. In detail, for the first scenario (balanced), a total of one division, three multiplications (including multi-controlled version), and two squarings (one multi-controlled) are employed, one controlled swap (i.e., Fredkin gate), as well as several (controlled and multi-controlled) additions, with one clean ancilla registers and one dirty ancilla register. For the second scenario, one less multiplication is employed, with the tradeoff of having both ancilla registers dirty. Additionally, regarding the third alternative of having two clean ancilla registers when the control qubit state is zero, it requires additional subroutines of one negative-control Toffoli series (elevated control from addition via CNOT gates), one negative multi-controlled division, and one addition by a constant. Note that we do not elaborate further on the exact resource since the presence of multi-controlled operations requires a more complex circuit decomposition. Nevertheless, we plan to investigate it further on a quantum simulator in our future work to obtain a more concrete resource estimation.

4 Discussions and Limitations

In this study, we delve into the topic of elliptic curve point doubling circuits, which has yet to be further examined in the literature. After presenting the design and description of our approach in the previous section, we now provide discussions related to the broader implications of our proposal.

Transforming to prime curves. We begin our study from the binary elliptic curves, which are relatively simpler than prime curves. In the case of prime curves, the quantum circuit will be more complicated because it cannot make use of the simplicity of field operation in binary curves. For instance, an addition in the prime fields requires a full adder, whereas binary fields only necessitate one Toffoli gate for each bit. Additionally, FLT-based inversion, which is comparable in performance to GCD-based inversion in quantum binary elliptic curves, has also not been considered to date for its use on prime curves due to the high resource requirements. Similarly, squaring operations are favorable in the binary case due to their relatively efficient construction (i.e., by leveraging a simple LUP decomposition), which are not applicable to prime curves. Even though there is an advantage in prime curves in terms of intuitive verification due to their nature of resembling decimal calculation, it requires more space and operations that are arguably more complicated and resource-intensive.

Relation to scalar multiplication. To realize a quantum elliptic curve scalar multiplication, existing methods rely upon a series of point addition circuits as the sole components. Therefore, the computation is in the form of

$Q = kP = P + P + \ldots + P$ for k times. Considering a more general implementation without limiting its use to Shor's algorithm, performing scalar multiplication by incorporating point doubling alongside point addition can potentially reduce the depth of the circuit and the number of operations. Moreover, the availability of designs for both point addition and point doubling opens up the opportunity to explore various classical elliptic curve point/scalar multiplication (ECPM) techniques (e.g., signed digit method, M-ary method) to be explored in the quantum realm in search of more efficient circuits.

For use in Shor's ECDLP. In order to create a more theoretically accurate and complete Shor's ECDLP circuit, point doubling will need to be integrated into the existing double scalar multiplication (that currently consists entirely of point addition subroutines) to cover the cases when the doubling of points occur. Note that for this algorithm, the input comes from the Walsh-Hadamard, so that the circuit is expected to be able to compute all possible cases (i.e., any combination of zero and one within the circuit). More importantly, the constant value k is unknown. For this reason, a more thorough conditional mechanism is required to control whether point doubling or point addition is in effect during the certain computation phase, resulting in more complex multi-control operations on the circuit. Additionally, in scenarios where both points of interest are quantum values, a comparator circuit may need to be employed to determine whether both values are identical. Note that for the specific use in Shor's ECDLP, both point addition and point doubling may need to be employed altogether to cover all cases, and other implementation requirements (e.g., unique representation for history independence [5], uncomputing garbage outputs to prevent unwanted interference [12]) will need to be taken into account, which will be explored in our future work.

Limitations. Even though our work has provided an initial step to explore further into point doubling, there are still various aspects that require further investigation. This includes how to correctly integrate it with the point addition circuit and whether the previous assumptions taken for Shor's ECDLP regarding the double-scalar multiplication still stand in this case, which is an interesting research problem. We leave these topics for our future work.

5 Conclusions and Future Work

In this study, we have examined the point-doubling operation for binary elliptic curves, which are required in the stricter case of Shor's algorithm. We began by analyzing the point doubling formula, identifying the inherent challenges in its construction, and presenting a possible solution. Subsequently, we designed quantum circuits for elliptic curve point doubling to cater to different scenarios, which shows the need for one more ancilla register compared to point addition, and while they may be comparable in terms of the number of subroutines, more complex multi-controlled operations are required than that of point addition. In addition, we provide a more in-depth discussion of the implications and concerns in incorporating the circuit into Shor's algorithm. To obtain a more detailed resource estimation for point doubling and complete double scalar multiplication for Shor's algorithm, we plan to construct the circuit in the existing quantum computing simulators and run the resource analysis as our future work.

References

1. Banegas, G., Bernstein, D.J., van Hoof, I., Lange, T.: Concrete quantum cryptanalysis of binary elliptic curves. IACR Transactions on Cryptographic Hardware and Embedded Systems pp. 451–472 (2021)
2. Gidney, C.: Asymptotically efficient quantum karatsuba multiplication. arXiv preprint arXiv:1904.07356 (2019)
3. Gouzien, É., Ruiz, D., Régent, F.M.L., Guillaud, J., Sangouard, N.: Computing 256-bit elliptic curve logarithm in 9 hours with 126133 cat qubits. arXiv preprint arXiv:2302.06639 (2023)
4. Gyongyosi, L., Imre, S.: Circuit depth reduction for gate-model quantum computers. Scientific reports **10**(1), 11229 (2020)
5. Häner, T., Jaques, S., Naehrig, M., Roetteler, M., Soeken, M.: Improved quantum circuits for elliptic curve discrete logarithms. In: International Conference on Post-Quantum Cryptography. pp. 425–444. Springer (2020)
6. Hankerson, D., Menezes, A.J., Vanstone, S.: Guide to elliptic curve cryptography. Springer Science & Business Media (2006)
7. Jang, K., Kim, W., Lim, S., Kang, Y., Yang, Y., Seo, H.: Optimized implementation of quantum binary field multiplication with toffoli depth one. In: International Conference on Information Security Applications. pp. 251–264. Springer (2022)
8. Kaye, P.: Optimized quantum implementation of elliptic curve arithmetic over binary fields. Quantum Info. Comput. **5**(6), 474–491 (Sep 2005)
9. Kirsch, Z., Chow, M.: Quantum computing: The risk to existing encryption methods. Retrieved from URL: http://www. cs. tufts. edu/comp/116/archive/fall2015/zkir sch. pdf (2015)
10. Larasati, H.T., Putranto, D.S.C., Wardhani, R.W., Park, J., Kim, H.: Depth optimization of flt-based quantum inversion circuit. IEEE Access (2023)
11. Liu, J., Wang, H., Ma, Z., Duan, Q., Fei, Y., Meng, X.: Quantum circuit optimization for solving discrete logarithm of binary elliptic curves obeying the nearest-neighbor constrained. Entropy **24**(7), 955 (2022)
12. Orts, F., Ortega, G., Combarro, E.F., Garzón, E.M.: A review on reversible quantum adders. Journal of Network and Computer Applications **170**, 102810 (2020)
13. Pornin, T.: Efficient and complete formulas for binary curves. Cryptology ePrint Archive (2022)
14. Proos, J., Zalka, C.: Shor's discrete logarithm quantum algorithm for elliptic curves. arXiv preprint quant-ph/0301141 (2003)
15. Putranto, D.S.C., Wardhani, R.W., Larasati, H.T., Ji, J., Kim, H.: Depth-optimization of quantum cryptanalysis on binary elliptic curves. IEEE Access (2023)
16. Roetteler, M., Naehrig, M., Svore, K.M., Lauter, K.: Quantum resource estimates for computing elliptic curve discrete logarithms. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 241–270. Springer (2017)
17. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. pp. 124–134. IEEE (1994)

18. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review **41**(2), 303–332 (1999)
19. Taguchi, R., Takayasu, A.: Concrete quantum cryptanalysis of binary elliptic curves via addition chain. In: Topics in Cryptology–CT-RSA 2023: Cryptographers' Track at the RSA Conference 2023, San Francisco, CA, USA, April 24–27, 2023, Proceedings. pp. 57–83. Springer (2023)
20. Van Hoof, I.: Space-efficient quantum multiplication of polynomials for binary finite fields with sub-quadratic toffoli gate count. arXiv preprint arXiv:1910.02849 (2019)
21. Vedral, V., Barenco, A., Ekert, A.: Quantum networks for elementary arithmetic operations. Physical Review A **54**(1), 147 (1996)
22. Whaley, B., Karasik, R.: Circuits, randomized computation, deferred measurements (02 2007), https://inst.eecs.berkeley.edu/~cs191/fa07/lectures/lecture9_fa07.pdf
23. Williams, C.P.: Explorations in Quantum Computing. Springer Science & Business Media (2010)