mathematics

IMPACT FACTOR 2.2

CITESCORE 4.6

Article

An Automated Synthesis Framework for Benchmarking Quantum Resource Costs of Symmetric-Key Cryptography

Chanho Choi, Jinseob Oh, SangMan Lee, Geumhwan Cho and Dooho Choi

Special Issue
Recent Advances in Quantum Optimization

Edited by
Dr. Pascal Halffmann

MDPI

https://doi.org/10.3390/math14040719

mathematics

MDPI

Article

An Automated Synthesis Framework for Benchmarking Quantum Resource Costs of Symmetric-Key Cryptography

Chanho Choi [1,2], Jinseob Oh [1], SangMan Lee [1,2], Geumhwan Cho [1,*] and Dooho Choi [1,2,*]

[1] Department of Cyber Security, Korea University, 2511 Sejong-ro, Sejong-si 30019, Republic of Korea; s292513ses@korea.ac.kr (C.C.); jinseoboh@korea.ac.kr (J.O.); robinsmlee@korea.ac.kr (S.L.)
[2] Seqrypton Inc., 413 Industry-Academic Cooperation Building, Korea University, 2511 Sejong-ro, Sejong-si 30019, Republic of Korea
* Correspondence: geumhwan@korea.ac.kr (G.C.); doohochoi@korea.ac.kr (D.C.)

Abstract

Modern information security relies heavily on symmetric-key cryptography such as AES. As quantum computing advances, these classical schemes face increasing pressure from quantum key-search attacks, most notably Grover's algorithm. To evaluate and compare quantum security quantitatively, the core components of symmetric-key algorithms must be implemented and optimized as quantum circuits. Among them, the S-box is a key source of nonlinearity and often dominates the circuit cost. In this paper, we introduce ADOQ (Automatic Depth Optimizer for Quantum circuits), a modular Python (version 3.13.3) framework that automatically synthesizes reversible quantum circuits from S-box specifications and applies a sequence of depth optimization techniques to produce optimized QASM circuits. Our experiments show that ADOQ achieves circuit depths comparable to prior work on 4-qubit S-boxes, and it also supports synthesis for larger S-boxes.

Keywords: quantum computing; quantum circuit synthesis; depth-optimization; automation

MSC: 81P68; 68Q12; 94A60

1. Introduction

Symmetric-key cryptography constitutes a core component of modern cryptographic systems, providing efficient and secure mechanisms to protect large volumes of data. We typically consider these algorithms computationally secure, meaning that an exhaustive key search on classical hardware would require impractically large computational resources. However, the emergence of quantum computing poses a fundamental threat to this security assumption by enabling quantum algorithms that can accelerate the solution of certain computational problems.

Among known quantum algorithms, Grover's algorithm [1] is particularly relevant to symmetric key cryptography, as it provides a quadratic speedup for exhaustive key search, reducing its computational complexity from 2^n to approximately $2^{\frac{n}{2}}$. Consequently, the effective security level of symmetric-key algorithms is reduced to half of their nominal key length under quantum adversaries. For example, AES-128 would offer only 64 bits of security level. This significant reduction in security level motivates a more systematic evaluation of symmetric-key encryption schemes in the quantum setting. In particular, it has become important to evaluate cryptographic algorithms based on their quantum circuit characteristics. However, there is no widely accepted standard for measuring quantum security. Nevertheless, the depth of the quantum circuit is generally recognized as a dominant

Academic Editor: Pascal Halffmann

Received: 20 January 2026
Revised: 11 February 2026
Accepted: 16 February 2026
Published: 19 February 2026

factor in determining the time complexity and execution cost of algorithms on quantum hardware. Hence, the circuit depth serves as a practical metric to compare the circuit-level execution cost of quantum implementations under a fixed cost model. This is because the depth of the circuit directly reflects the time complexity of quantum implementations.

Based on this observation, it is essential to ensure fairness in comparison by applying a consistent depth optimization methodology across different algorithms. However, previous studies [2,3] have employed different optimization strategies, making it difficult to compare their results in a fair and consistent manner. Although previous studies have proposed circuit synthesis approaches and optimization methods, these approaches have limitations. Lighter-R [2] is based on a meet-in-the-middle (MITM) algorithm. This is similar to the well-known width-first search algorithm in the context of graph theory, which is only effective for small functions $S : GF(2^c) \rightarrow GF(2^c)$ (where $c \leq 4$). DORCIS [3] extends LIGHTER-R by incorporating Clifford+T gate decomposition. It enables optimization with respect to both the quantum circuit depth and the T-depth. However, the synthesis of S-boxes with more than five qubits remains challenging due to the large search space. As a result, it is difficult to compare the resulting quantum circuit representations objectively in terms of key metrics such as gate count, circuit depth, and qubit utilization.

To address this challenge, we propose an automated framework named ADOQ (Automatic Depth Optimizer for Quantum circuits) for the quantum circuit optimization of symmetric-key encryption algorithms. The proposed framework also provides a consistent basis for benchmarking circuit-level resource costs relevant to quantum-security assessments by enabling systematic comparisons of circuit-level metrics. It ensures fairness and reproducibility by standardizing the circuit generation process. It automatically synthesizes quantum circuits from truth tables through an integrated translation and depth optimization pipeline. We demonstrate the proposed framework by comparing S-box synthesis results for eight lightweight encryption algorithms.

This work does not propose a new cryptanalytic attack on symmetric-key primitives. Instead, ADOQ provides an automated and reproducible framework for synthesizing and optimizing quantum (reversible) circuit implementations of core components such as S-boxes, enabling systematic resource benchmarking in terms of depth, gate counts, and qubit counts. Hence, ADOQ itself should not be interpreted as introducing a new threat to the security of symmetric-key cryptography. Rather, it can help refine constant-factor resource estimates for implementing known quantum attack models (e.g., quantum-accelerated key search) by providing tighter and reproducible circuit-level cost assessments. For broader discussions on quantum threats and mitigations—including post-quantum cryptography and quantum-secure communication—we refer readers to [4].

To evaluate the proposed framework, we first compare the S-box synthesis results across eight lightweight cryptographic algorithms. These S-boxes are implemented as quantum circuits and evaluated under uniform depth optimization conditions (e.g., the 4-qubit S-box of the DEFAULT-CORE cipher [5]). In addition, we apply the proposed framework to larger S-boxes that were not synthesized in previous work. Specifically, we present synthesis and optimization results for the 5-qubit S-box of ASCON [6] and the 8-qubit S-box of AES [7]. These S-boxes were not addressed in previous work due to limitations of existing approaches [2,3]. We compare the proposed framework against representative prior approaches for quantum S-box synthesis. The experimental results show that the proposed framework achieves a 2.58% improvement in depth optimization compared to LIGHTER-R [2]. While DORCIS [3] reports higher optimization efficiency, it is limited to S-boxes with fewer than five qubits. In contrast, our framework supports scalable synthesis and depth optimization for larger S-boxes, enabling evaluation beyond the scope of existing methods.

Our contributions are summarized as follows:

- We propose ADOQ, an automated framework that synthesizes and depth-optimizes quantum circuits directly from S-box truth tables, enabling consistent circuit generation across different S-box implementations. We demonstrate that the proposed framework can handle larger S-boxes, including the 5-qubit S-box of ASCON and the 8-qubit S-box of AES, which prior work could not address due to methodological limitations.
- We introduce a unified evaluation framework for quantum circuit synthesis and depth optimization. The proposed framework provides a fair and reproducible benchmarking basis by standardizing quantum circuit synthesis and depth optimization across different S-box implementations.

The rest of this paper is organized as follows. Section 2 reviews related work on quantum circuit synthesis and depth optimization for S-boxes. In Section 3, we introduce the bidirectional synthesis algorithm used to generate reversible quantum circuits that satisfy given S-box specifications. In Section 4, we present a set of depth optimization techniques. Section 5 describes the proposed ADOQ framework, which integrates and automates the synthesis and optimization algorithms. The experimental results are reported in Section 6, and we conclude the paper in Section 7.

2. Related Work

Reversible logic synthesis has been extensively studied as a fundamental technique for embedding classical functions into quantum circuits [8–14]. Early foundational work by Shende et al. [8] established the theoretical basis of reversible synthesis, demonstrating that any even permutation can be realized using only NCT (NOT–CNOT–Toffoli) gates without auxiliary lines, whereas odd permutations require exactly one ancilla. This result led to cycle-decomposition-based synthesis frameworks that underpin many subsequent reversible and quantum circuit synthesis approaches.

Building on the foundational theory of reversible logic synthesis, subsequent studies have focused on improving synthesis efficiency through local optimization and structured search techniques. Prasad et al. [9] proposed graph-based data structures that enable the constant-time replacement of equivalent 4-bit reversible subcircuits, thus systematizing local optimization within the synthesis pipeline. Beyond brute-force enumeration, later work explored more scalable algorithmic and algebraic approaches. In particular, Li et al. [10] leveraged bit-wise perfect hashing and compressed state representations to accelerate both search and storage, enabling fast optimal synthesis of 3-qubit reversible circuits and achieving up to a 69.8× speedup over prior methods.

For larger reversible functions, several works explored meet-in-the-middle (MITM)-based synthesis frameworks. Yang et al. [11] combined permutation group theory using GAP software with bidirectional MITM search to synthesize minimal-cost 4-bit reversible circuits under various gate cost metrics. Extending this line of work, Golubitsky et al. [12] introduced symmetry reductions—including simultaneous input/output permutations and negations—to enable optimal synthesis for arbitrary 4-bit reversible functions. The same authors later expanded this framework to enumerate all optimal realizations for a given specification, incorporate linear nearest-neighbor (LNN) constraints, and provide a benchmark corpus of permutation-hard 4-bit functions along with their optimal-size distributions [13].

Collectively, prior studies demonstrate that optimal or near-optimal synthesis has been extensively explored for small-scale reversible and quantum circuits, typically limited to 4-bit or 4-qubit functions. These approaches provide strong optimality guarantees, but they inherently suffer from exponential growth in the search space, which fundamentally limits their scalability to larger specifications. As a result, exhaustive and MITM-based

techniques quickly become computationally infeasible for larger S-boxes or higher bit-width reversible functions.

Moreover, classical reversible synthesis typically optimizes gate count or abstract cost models, which do not always capture hardware-relevant constraints in practical quantum architectures, such as circuit depth, qubit connectivity, and limited coherence time. Although several quantum-aware synthesis approaches have been proposed—particularly for cryptographic S-boxes—these methods are often tightly coupled to specific heuristics, cost models, or gate libraries, making generalization and fair comparison across techniques challenging.

More broadly, existing work largely follows a method-centric evaluation paradigm, where each synthesis or optimization technique is evaluated under its own experimental setup, benchmarks, and metrics. Consequently, it remains difficult to reproducibly compare different approaches and systematically analyze their scalability. In particular, a unified and extensible benchmarking framework that enables a consistent evaluation of depth-optimized quantum circuit synthesis across diverse S-box specifications is still lacking.

As representative examples of such quantum-aware, depth-oriented approaches, several studies have specifically addressed cryptographic S-box synthesis. Several studies have specifically addressed quantum-aware synthesis and depth optimization for cryptographic S-boxes. Dasu et al. [2] proposed LIGHTER-R, an end-to-end tool that generates candidate circuits for 4×4 S-boxes using graph-based meet-in-the-middle (MITM) techniques and selects near-optimal implementations based on user-specified cost metrics over a reversible gate library. Building upon LIGHTER-R, Chun et al. [3] introduced DORCIS, which extends the framework toward depth-first optimization by adopting T-depth as the primary cost metric. DORCIS explicitly accounts for Clifford+T decomposition and demonstrates improved depth performance for 3-bit and 4-bit S-boxes under identical specifications. Recent research on quantum circuit optimization has progressed in several directions, including reducing non-Clifford resources (e.g., Toffoli/T cost), exploiting explicit space–depth trade-offs via ancilla allocation, and in some lines of work, incorporating architecture constraints and fault-tolerant overhead models into resource estimation [15–18]. ADOQ is complementary to these directions by providing an automated, modular pipeline from S-box truth tables to reversible synthesis and depth optimization, enabling systematic benchmarking across cryptographic primitives.

While previous approaches demonstrate promising depth reductions for small S-boxes, they are typically tailored to specific synthesis strategies, cost models, or S-box sizes. As a result, systematic comparisons across different optimization techniques and scalable evaluation beyond limited bit-widths remain challenging. To address this gap, we introduce ADOQ, a unified benchmarking framework designed to systematically evaluate depth-optimized quantum circuit synthesis across a wide range of S-box sizes and optimization strategies.

3. Bidirectional Quantum Circuit Synthesis

In this section, we describe the quantum circuit synthesis mechanism employed in our framework. To efficiently synthesize reversible quantum circuits, our approach adopts a bidirectional quantum circuit synthesis algorithm [19] and extends it with a framework-specific strategy designed for larger S-boxes.

Bidirectional quantum circuit synthesis was originally proposed as a method for constructing reversible logic circuits. This approach systematically transforms a given reversible truth table into a functionally equivalent quantum circuit. The synthesis is performed iteratively by inserting reversible quantum gates (e.g., NOT, CNOT, and Toffoli) that preserve reversibility at each step. Given a reversible *Boolean* function $f : \{0,1\}^n \rightarrow \{0,1\}^n$, the goal of the synthesis algorithm is to derive a sequence of reversible gates for

which their composition realizes f. To achieve this, the algorithm maintains two evolving representations of the target function: a forward representation initialized from the input and a backward representation initialized from the output. During synthesis, gates are applied in the forward and backward directions to progressively reduce the differences between two representations. When the forward and backward transformations reach an identical intermediate state, the synthesis terminates. The final reversible circuit is then obtained by concatenating the forward gate sequence with the inverse of the backward gate sequence.

Existing bidirectional synthesis algorithms [19] were primarily developed for small reversible circuits and have been demonstrated mainly for functions involving up to three qubits. As the number of qubits increases, the search space grows rapidly, limiting scalability and reducing the effectiveness of depth-count and gate-count optimization. As a result, circuits synthesized by these approaches often exhibit non-minimal gate counts and excessive circuit depth. Such characteristics pose practical challenges for near-term quantum hardware, particularly NISQ (Noisy Intermediate-Scale Quantum) devices, where limited coherence time and imperfect gate fidelity impose strict constraints on circuit complexity. To address these limitations, our framework extends existing bidirectional synthesis algorithms to support scalable synthesis of quantum circuits with four or more qubits. In addition, the proposed approach incorporates framework-level optimization mechanisms aimed at reducing circuit depth and gate cost, making the synthesized circuits more suitable for execution on NISQ devices.

3.1. Reversibility and Design Considerations

We focus on reversibility, the use of an elementary gate set, and circuit depth, since these factors directly shape the design and optimization of the synthesis procedure. Quantum circuits must be reversible, meaning that the underlying function defines a one-to-one mapping where each input state corresponds to a unique output state. Consequently, we define our synthesis target as an n-bit reversible function $f : \{0,1\}^n \to \{0,1\}^n$, represented by a reversible truth table. We use a standard set of reversible quantum gates, namely the NOT, CNOT, and Toffoli gates. These gates are commonly used for reversible and quantum circuit synthesis. The elementary quantum gates considered in this work are as follows:

- NOT Gate: Performs a single-qubit inversion.
- CNOT Gate: Performs a conditional inversion on a target qubit controlled by a single control qubit.
- Toffoli Gate: Performs a conditional inversion on a target qubit controlled by two control qubits (controlled-controlled-NOT).

This gate set offers three practical benefits: (i) It preserves reversibility at every intermediate step, (ii) it simplifies the composition of the forward and backward sequences in bidirectional synthesis, and (iii) it enables the consistent evaluation of circuit depth and gate cost. These metrics are critical for assessing the practicality of synthesized circuits on NISQ devices. When a circuit requires more complex multi-controlled operations, such operations can be decomposed into combinations of these basic gates. This allows both the quantum cost and the circuit depth of the synthesized circuit to be evaluated consistently.

In this context, the quantum cost is typically defined as the weighted sum of gate operations, while circuit depth corresponds to the length of the longest sequence of operations that cannot be executed in parallel. Both metrics are critical for assessing the feasibility of quantum circuits on noisy hardware. In NISQ devices, circuit depth directly impacts execution time and must remain within qubit coherence limits. Even circuits with modest gate counts may become impractical if their depth exceeds hardware noise tolerance, whereas

shallower circuits can exploit parallelism. Accordingly, our framework explicitly considers circuit depth as an important optimization objective in bidirectional synthesis.

3.2. Bidirectional Synthesis Procedure

In Algorithm 1, we synthesize a reversible circuit directly from a reversible truth table $T = \{(x, y) \mid y = f(x)\}$ over $\{0,1\}^n$. The procedure processes T in canonical input order and incrementally fixes the mapping for each index $i = 0, \ldots, 2^n - 1$. At iteration i, the i-th row (x_i, y_i) satisfies $x_i = i$, and we enforce $y_i = i$, thereby permanently fixing the i-th mapping before proceeding to the next index.

Algorithm 1 Bidirectional Synthesis Procedure for Reversible Circuits

Input: Reversible truth table $T = \{(x, y) \mid y = f(x)\}$ over $\{0,1\}^n$
Output: Reversible circuit C such that $C(x) = f(x)$
Notation: $d_H(\cdot,\cdot)$: Hamming distance; $S_{\text{in}}, S_{\text{out}}$: gate sequences
Initialize $S_{\text{in}} \leftarrow \langle\rangle$, $S_{\text{out}} \leftarrow \langle\rangle$
Maintain T in canonical input order (row i corresponds to input i)
for $i = 0$ to $2^n - 1$ **do**
    Let the i-th row be (x_i, y_i) where $x_i = i$
    if $y_i = i$ **then**
        continue
    end if
    Find the unique index j such that $y_j = i$
    $d_{\text{out}} \leftarrow d_H(i, y_i)$
    $d_{\text{in}} \leftarrow d_H(i, x_j)$
    if $d_{\text{out}} > d_{\text{in}}$ **then**
        Input-side update
        while $x_j \neq i$ **do**
            Choose $g \in \{NOT, CNOT, Toffoli\}$ that reduces $d_H(x_j, i)$
            $S_{\text{in}} \leftarrow S_{\text{in}} \,\|\, \langle g \rangle$
            Apply g to **Inputs** of T
            Restore canonical input order of T
        end while
    else
        Output-side update
        while $y_i \neq i$ **do**
            Choose $g \in \{NOT, CNOT, Toffoli\}$ that reduces $d_H(y_i, i)$
            $S_{\text{out}} \leftarrow S_{\text{out}} \,\|\, \langle g \rangle$
            Apply g to **Outputs** of T
        end while
    end if
end for
$C \leftarrow S_{\text{in}} \,\|\, S_{\text{out}}^{-1}$ ▷ S_{out}^{-1} denotes the reverse-ordered gate sequence (NOT/CNOT/Toffoli are self-inverse)
return C

At each iteration, we decide whether to apply synthesis from the input side or the output side by comparing two Hamming-distance-based mismatches. We define the output-side mismatch, denoted by d_{out}, as

$$d_{\text{out}} = d_H(i, y_i) \tag{1}$$

which measures how far the current output at row i is from the desired value i. Since f is bijective, there exists a unique index $j = f^{-1}(i)$ such that $y_j = i$. We define the input-side mismatch, denoted by d_{in}, as

$$d_{\text{in}} = d_H(i, x_j) = d_H\left(i, x_{f^{-1}(i)}\right) \tag{2}$$

which measures how far the corresponding input label is from the target index i. We select the input-side update if $d_{\text{out}} > d_{\text{in}}$; otherwise, we select the output-side update as follows:

$$\text{Direction} = \begin{cases} \text{Input-side}, & d_{\text{out}} > d_{\text{in}}, \\ \text{Output-side}, & \text{otherwise}. \end{cases} \tag{3}$$

Table 1 illustrates this selection rule on a 3-qubit truth table. In the input-side update, we iteratively insert gates from $\{NOT, CNOT, Toffoli\}$ on the input labels to reduce $d_H(x_j, i)$ until $x_j = i$, which effectively moves the row-producing output i into the j-th position. After applying an input-side gate, we restore the canonical input order of T so that the j-th row again corresponds to input i. In the output-side update, we analogously insert gates on the outputs to reduce $d_H(y_i, i)$ until $y_i = i$. Finally, we construct the synthesized circuit by concatenating the input-side sequence in order and the output-side sequence in reverse order (i.e., $C = S_{\text{in}} \parallel S_{\text{out}}^{-1}$).

Table 1. Synthesis direction selection using the bidirectional algorithm. (**a**) Synthesis applied from the output side. (**b**) Synthesis applied from the input side.

Inputs **cba**	**Outputs** $c'b'a'$	**Inputs** **cba**	**Outputs** $c'b'a'$
000	000	000	000
001	010	001	001
010	111	010	010
011	100	011	100
100	110	100	101
101	011	101	110
110	101	110	011
111	001	111	111
(**a**)		(**b**)	

For example, in Table 1a, the row with input 001 currently maps to output 010. Since 010 does not match the desired value 001, the algorithm applies an output-side update to transform 010 into 001 while preserving reversibility. In contrast, Table 1b illustrates an input-side update, where the row producing output i is moved into position i by modifying the input labels.

After all indices i have been fixed (i.e., the truth table has been reduced to the identity mapping), the final circuit is constructed by composing the gate sequences synthesized from both directions. Specifically, gates accumulated on the input side are placed on the left in their original order, while gates accumulated on the output side are placed on the right in reverse order. Therefore, the resulting circuit is given by $C = S_{\text{in}} \parallel S_{\text{out}}^{-1}$, which implements the original mapping f.

3.3. Extending Synthesis to Four or More Qubits

The bidirectional synthesis procedure in Section 3.2 serves as a strong baseline for reversible circuit construction. However, existing approaches [19] are mainly effective for small circuits, typically up to three qubits. In modern cryptographic settings, S-box circuits often require four or more qubits, where the synthesis space grows rapidly, and naive extensions become inefficient. For example, PRESENT uses a 4-bit S-box [20], ASCON uses a 5-bit S-box [6], and AES uses an 8-bit S-box [7].

To bridge this gap, our framework extends the bidirectional synthesis procedure to support scalable synthesis for $n \geq 4$ qubits. The key idea is to allow multi-controlled Toffoli (MCT) updates when correcting a mismatched row in the reversible truth table. Given a row (x_i, y_i) with $x_i \neq y_i$, the algorithm selects a target bit position to be flipped and derives a set of control conditions from the remaining bits of the row. This yields a control–target relationship that can be expressed as an MCT operation, enabling direct updates even when more than three control qubits are required. Most NISQ devices are evaluated using a small set of elementary gates. Therefore, we do not leave multi-controlled Toffoli (MCT) gates in the final circuit. Whenever an MCT gate with more than three controls appears, we automatically decompose it into a circuit composed of Toffoli and CNOT gates. While the enhanced procedure supports larger S-boxes, its complexity increases as the number of control qubits in MCT updates grows. Therefore, post-synthesis optimization is crucial to reduce circuit depth and gate cost. In Section 4, we present four optimization techniques that are integrated into our framework to further improve the circuit depth and overall efficiency.

4. Depth-Oriented Optimization Techniques

In this paper, ADOQ targets the synthesis of a reversible circuit implementing a given S-box while minimizing logical circuit depth under the stated gate library and scheduling policy. Functional correctness is enforced during optimization, and gate count and/or algorithmic ancilla are treated as secondary criteria depending on the evaluation setting.

We describe the depth-optimization phase applied after the synthesis of quantum circuits using the bidirectional synthesis algorithm. Although the synthesized circuits satisfy the reversible truth table of the target S-box, they often contain a significant number of MCT gates and redundant or sequentially dependent gate structures. Such characteristics lead to increased circuit depths and gate counts, which directly affect execution time and error rates with respect to NISQ-era quantum circuits.

To address these challenges, we introduce four complementary optimization techniques that target different sources of depth overhead, including gate decomposition, dependency reduction, and global restructuring. These techniques are integrated into the ADOQ framework and can be applied independently or sequentially depending on the optimization objective:

1. MCT decomposition;
2. Bidirectional optimization;
3. Ancilla use optimization;
4. Simulated annealing.

Each optimization technique is discussed in detail in the following subsections.

4.1. MCT Decomposition

MCT gates are essential components for implementing complex Boolean functions in reversible and quantum circuits. However, it often requires many gate operations to be executed sequentially when MCT gates are implemented directly, significantly increasing circuit depth and T-count. In this work, we extend the MCT decomposition algorithm proposed in [21] by explicitly incorporating context-aware decomposition to reduce circuit depths.

Previous work has extensively studied the decomposition of MCT gates into combinations of Toffoli and CNOT gates to reduce implementation complexity and improve hardware compatibility [21]. These approaches primarily focus on minimizing gate count or simplifying gate structures, while applying fixed decomposition patterns regardless of the surrounding circuit context. Figure 1 illustrates a representative example of such a

conventional MCT decomposition, where an MCT gate with three control qubits is decomposed into a sequence of Toffoli gates using one ancilla qubit. As shown in Figure 1b,c, multiple equivalent decomposition patterns exist for the same MCT gate.

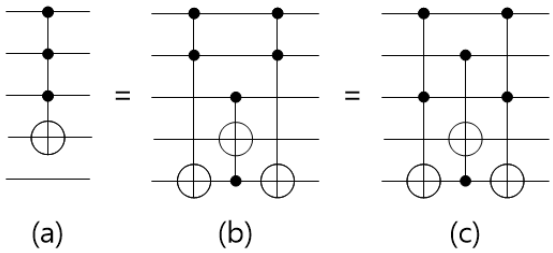

Figure 1. Decomposition of MCT gate. (**a**) A MCT gate with three control qubits. (**b**) Decomposition that applies the first and second control qubit operations to the ancilla qubit. (**c**) Decomposition that applies the first and third control qubit operations to the ancilla qubit. Black dots represent control qubits, empty circles denote target qubits, and the bottom wire corresponds to the ancilla qubit.

Although these decomposition strategies are effective for standardizing complex gates, they do not explicitly address reducing circuit depth. In particular, it can force gates to be executed sequentially when fixed decomposition patterns are used, even though alternative decompositions could allow gate cancellation or parallel execution. In contrast, our approach dynamically selects a decomposition pattern based on the surrounding gate context, with the explicit goal of minimizing the overall circuit depth.

Unlike conventional decomposition methods that apply a fixed pattern regardless of the circuit structure, the proposed algorithm explicitly evaluates the potential for circuit depth reduction when selecting a decomposition strategy. To this end, it estimates the depth impact of each candidate decomposition based on the following factors:

- The number of control qubits;
- The dependency relationships between adjacent gates;
- The potential for gate elimination;
- The potential for concurrent execution with neighboring gates.

If the gates immediately before or after an MCT gate share control qubits, it prioritizes decomposition patterns that allow gate cancellation. Otherwise, it selects a decomposition that maximizes parallel execution with neighboring gates, thereby reducing the overall circuit depth. Figure 2 illustrates how this context-aware decomposition strategy selects different decomposition patterns depending on the surrounding gate context.

Figure 2a shows a context-aware decomposition process for an MCT gate with $q[0]$, $q[1]$, $q[2]$, and $q[3]$ as control qubits and $q[4]$ as the target qubit. Among multiple valid decomposition candidates composed of Toffoli gates, the algorithm selects a pattern that becomes structurally identical to the preceding gate. As a result, the decomposed gate can be canceled, eliminating an entire sequential layer and thereby reducing the circuit depth.

Figure 2b shows a case where the MCT gate cannot be decomposed into a form identical to the preceding gate. In this scenario, it performs a swap with the preceding CNOT gate to expose new opportunities for depth reduction. By decomposing the MCT gate considering gate swapping, the circuit depth is reduced, allowing more favorable gate ordering.

Furthermore, when the gate following the MCT gate is also decomposable, the algorithm evaluates its decomposition candidates and selects the combination that maximizes the overall depth reduction. The MCT decomposition stage plays a crucial role in standardizing the circuit by transforming complex MCT gates into context-aware elementary gate structures. This preprocessing step not only reduces the local circuit depth but also

prepares the circuit for subsequent global depth optimization techniques. After MCT decomposition, we apply the swap-based gate reordering rules described in Section 4.2 to expose parallelization and subsequent depth-optimization opportunities.

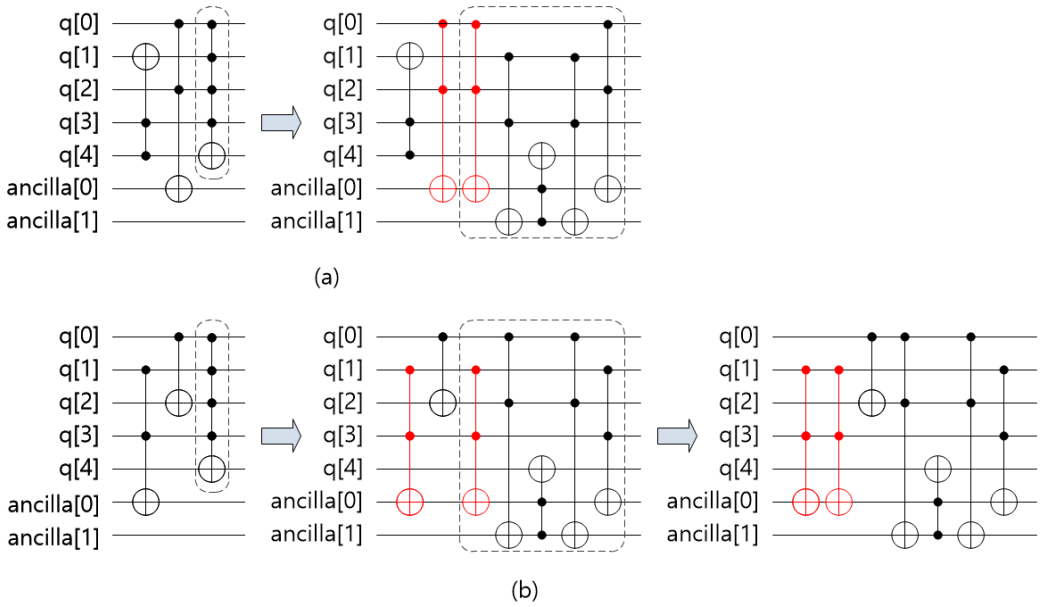

Figure 2. Context-aware decomposition of an MCT gate. (**a**) Decomposition into gates with the same properties as the preceding gate and gate deletion to reduce depth. (**b**) After swapping with the preceding gate, decomposition into gates with the same properties and gate deletion to reduce depth. Red gates placed side by side indicate operations that can be removed during optimization; black dots represent control qubits, empty circles denote target qubits, and dashed boxes highlight local decomposition blocks.

A fixed MCT decomposition applies a predetermined pattern to a k-control MCT gate and therefore incurs $O(k^2)$ work without candidate search. In contrast, our context-aware decomposition performs (i) candidate generation and (ii) candidate scoring with a one-step lookahead. Let A denote the number of available ancillas. Naively, candidate enumeration yields $O(k^2 \cdot A)$ candidates; in our implementation, we restrict the search to a local neighborhood, reducing it to $O(k \cdot A_{\text{local}})$. With this pruning, lookahead scoring costs $O(k^2)$ per iteration, and the decomposition loop runs $O(k)$ iterations, resulting in a worst-case overhead of $O(k^3)$ per MCT gate. Over a circuit containing M MCT gates with control sizes $\{k_i\}_{i=1}^{M}$, this yields $O\left(\sum_{i=1}^{M} k_i\right)$ for fixed patterns versus $O\left(\sum_{i=1}^{M} k_i^3\right)$ in the worst case for the context-aware selection.

4.2. Bidirectional Optimization

After decomposition, the circuit may still contain gate sequences that can be reordered to enable greater parallelism. The bidirectional optimization algorithm [22] reduces circuit depth by scanning the circuit in both forward and backward directions to identify gates that can be executed simultaneously or safely reordered. To identify a set of gates that can be executed in parallel, adjacent gates must not overlap in their control qubit C and target qubit T. This condition can be expressed as follows:

$$\{c_{i-1}, t_{i-1}\} \cap \{c_i, t_i\} = \emptyset \quad (\text{or } \{c_i, t_i\} \cap \{c_{i+1}, t_{i+1}\} = \emptyset)$$

Similarly, adjacent gates can be swapped only when their control and target qubits satisfy the following condition:

$$t_{i-1} \neq c_i \text{ and } c_{i-1} \neq t_i \quad (\text{or } t_i \neq c_{i+1} \text{ and } c_i \neq t_{i+1})$$

When these conditions hold, the gates can be reordered without affecting the circuit's functionality. The algorithm applies a sequence of adjacent gate-swap (commutation) operations to group gates acting on disjoint qubits into the same time layer, thereby increasing parallelism and reducing circuit depth. Building on previous work [22], our approach further incorporates gate deletion during bidirectional reordering. When two identical gates become adjacent during reordering, they are removed to further reduce circuit depth. For example, two consecutive CNOT gates acting on the same qubits can be safely eliminated since they cancel each other.

Figure 3 illustrates the depth optimization process achieved by moving the bidirectional gate position. By first applying bidirectional optimization from the input side, the circuit depth is reduced from 7 to 5. Subsequent optimization from the output side further aligns independent gates into the same execution layers, reducing the depth from 5 to 4.

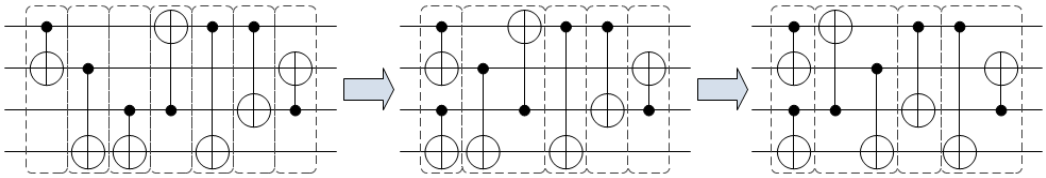

Figure 3. Position movement of gates considering depth optimization. Black dots represent control qubits, open circles denote target qubits, and dashed boxes indicate gates executed in the same depth layer.

Previous work has explored depth optimization through gate movement and reordering. However, beyond simple gate relocation, our approach also identifies pairs of identical gates that become adjacent due to swapping. When two equivalent gates appear consecutively, they can be safely removed without affecting the circuit's functionality. Figure 4 demonstrates this deletion process, where gate movement exposes cancelable gate pairs, leading to additional depth reduction.

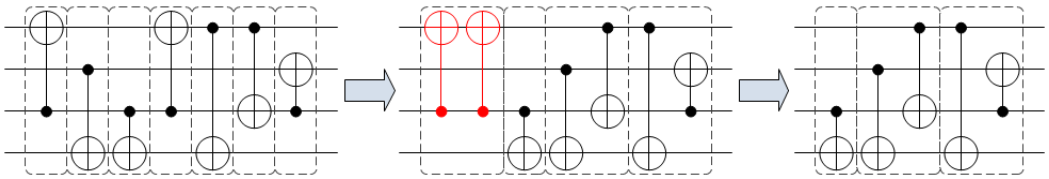

Figure 4. Position movement and depth reduction for gate deletion. Black dots represent control qubits, open circles denote target qubits, red gates indicate operations identified for deletion, and dashed boxes highlight groups of gates executed within the same depth layer.

By combining gate movement, reordering, and elimination, the proposed bidirectional optimization fully exploits latent parallelism in the circuit. This process not only achieves substantial reductions in circuit depth but also aligns the gate structure for subsequent optimization stages.

4.3. Ancilla Optimization

Ancilla qubits enable the temporary storage of intermediate values. By leveraging this capability, gate sequences that would otherwise execute sequentially can be performed in parallel, thereby reducing circuit depth at the cost of additional qubits. The proposed ancilla optimization algorithm scans the circuit to identify consecutive or non-consecutive gates that share identical control qubits and introduces ancilla qubits to decouple these

dependencies. By storing intermediate control states on ancilla qubits, the algorithm restructures the circuit to execute multiple gates in parallel and subsequently restores the original state, preserving functional correctness. In ADOQ, this ancilla optimization is an optional module controlled by an ancilla budget. It explicitly trades increased qubit count for reduced depth; the quantitative trade-offs for ASCON and AES are reported in Section 6.

Compared to previous work [23], our approach extends the usage of ancilla beyond strictly consecutive gate patterns by considering a broader gate context to maximize depth reduction. As illustrated in Figure 5, the algorithm first identifies a candidate gate section that shares a common control qubit. It then expands the optimization region when beneficial and inserts auxiliary CNOT operations for ancilla initialization and cleanup. Among multiple candidates, the algorithm selects the configuration that achieves the greatest depth reduction, achieving a measurable reduction in circuit depth in the given example.

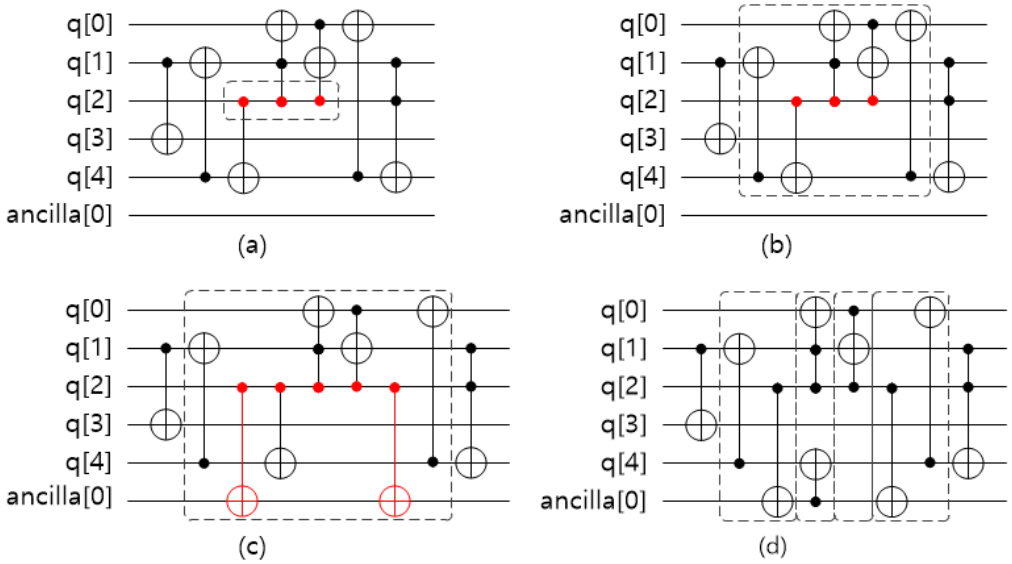

Figure 5. Depth-aware optimization using ancilla qubits. (**a**) Identification of a candidate gate section. (**b**) Expansion of the candidate section. (**c**) Insertion of CNOT gates using ancilla qubits to enable parallel execution. (**d**) Final circuit after depth optimization. Black dots represent control qubits, open circles denote target qubits, and red dots indicate control qubits selected for optimization. Dashed boxes highlight local blocks considered for optimization (including ancilla-assisted updates), and the final arrangement illustrates gates grouped into the same depth layer.

In Figure 5, the highlighted gates indicate a candidate section sharing a common control qubit. By introducing an ancilla qubit and auxiliary CNOT operations, the originally sequential gates are decoupled and executed in parallel, resulting in reduced circuit depth. Although this optimization slightly increases the total qubit count, it provides a practical trade-off in NISQ environments, where limited coherence time and high gate error rates make depth reduction more critical than minimizing qubit usage.

4.4. Simulated Annealing

While the previous optimization stages are deterministic and rule-based, the final stage employs a stochastic global optimization approach based on simulated annealing. Simulated annealing is a probabilistic search algorithm that explores the solution space by occasionally accepting higher-cost solutions. This mechanism helps avoid convergence to local minima and improves the chance of finding near-optimal solutions in complex optimization problems [24].

The algorithm is inspired by the physical annealing process, where a system is gradually cooled to reach a low-energy stable state. At high temperatures, the algorithm explores the solution space aggressively by allowing diverse circuit modifications. As the temperature decreases, the acceptance of higher-cost solutions is gradually restricted, leading the search to converge toward a low-cost circuit.

To modify the circuit, the algorithm randomly adds, replaces, or deletes elementary gates (NOT, CNOT, and Toffoli) at arbitrary locations. Each generated circuit is evaluated using a cost function that combines the Hamming distance, circuit depth, and the number of gates, reflecting both functional correctness and optimization objectives. The simulated annealing optimization procedure used in this work consists of the following steps:

1. Initial Circuit: Set the circuit synthesized by the bidirectional synthesis algorithm as the initial state. Alternatively, the reversible truth table alone can be used as the initial input.
2. Perturbation: Apply random perturbations by adding, replacing, or deleting an elementary gate (NOT, CNOT, or Toffoli) at arbitrary locations in the current circuit.
3. Cost Function: Compute the Hamming distance to evaluate functional correctness with respect to the reversible truth table. In addition, circuit depth or gate count is incorporated into the cost function to reflect the optimization objective.
4. Acceptance Criterion: Let $\Delta = C_{\text{new}} - C_{\text{cur}}$. If $\Delta < 0$, accept the new circuit; otherwise, if the probability is $P(t_{cur}, \Delta) \geq Random(0, 1)$, accept—probability $P(t_{cur}, \Delta)$ is defined in Equation (4):

$$P(t_{cur}, \Delta) = \exp\left(-\frac{\Delta}{K t_{cur}}\right), \tag{4}$$

Here, t_{cur} denotes the current temperature, and K is a scaling parameter controlling the acceptance sensitivity. We use $K = 10$ for 4-qubit S-box synthesis and $K = 30$ for 8-qubit S-box synthesis. These values were selected via a pilot study over candidate settings to minimize the average circuit depth, and then, they are kept fixed across all benchmarks within each bit-width for fair comparability.

A previous work [24] applied simulated annealing to circuit optimization; however, the probabilistic nature of the temperature schedule could cause previously discovered low-cost circuits to be discarded during the search. In this work, we refine the acceptance strategy to preserve the best solutions found so far, thereby improving the probability of retaining the minimum-cost circuit while maintaining effective global exploration.

In this section, four optimization techniques—MCT decomposition, bidirectional optimization, ancilla-based optimization, and simulated annealing—are used together to reduce circuit depth and improve synthesis efficiency. Each technique addresses a different aspect of the optimization problem: MCT decomposition maps complex gates to hardware-compatible primitives, bidirectional optimization exploits parallelism through gate reordering and elimination, ancilla-based optimization enables concurrent execution via temporary qubits, and simulated annealing provides global refinement to escape the local optimum.

Together, these methods form the core of the proposed Automatic Depth Optimizer for Quantum (ADOQ) circuits, enabling fully automated synthesis and the depth optimization of reversible circuits for S-boxes and other symmetric-key cryptographic components.

4.5. Optimization Policy Summary

To avoid ambiguity, we explicitly summarize which optimizations are applied in the reported pipeline. Unless stated otherwise, we apply the following: (i) local gate-level simplifications (e.g., cancellation of adjacent inverses and commutation-enabled reductions), (ii) depth-aware ordering/scheduling during bidirectional optimization (Section 4.2),

(iii) ancilla-aware transformations as described in Section 4.3, and (iv) heuristic refinement via simulated annealing (Section 4.4), which is enabled by default in our experimental configuration but can be disabled as a toggle.

We do not claim exhaustive global Toffoli minimization or globally optimal ancilla reuse; instead, the goal is consistent, reproducible improvements under a fixed set of heuristics and constraints. Throughout optimization, we preserve the fixed I/O convention and do not intentionally introduce additional garbage outputs; in particular, ancillary work qubits introduced by decomposition follow the ancilla clean-up property described in Section 6.

5. Optimization Framework and Case Study

In this section, we present an automated optimization framework named ADOQ. ADOQ integrates the bidirectional synthesis algorithm and the depth optimization techniques described in Sections 3 and 4. Given a reversible truth table as input, the framework first synthesizes a functional quantum circuit and subsequently applies a sequence of depth optimization steps to produce an optimized QASM (Quantum Assembly Language) circuit.

5.1. Design Objectives and Principles

To improve the understanding of our framework, we describe its design principles and the overall architecture of the proposed ADOQ framework. ADOQ is designed to automatically synthesize and optimize circuits corresponding to reversible *Boolean* functions, with a primary focus on minimizing circuit depth for resource benchmarking under a consistent cost model.

5.1.1. Benchmarking Objective and Rationale for Staged Design

Our goal is to provide a standardized and reproducible benchmarking pipeline for S-box quantum circuits—the core nonlinear components of symmetric-key cryptography—so that circuit-level resource metrics (e.g., depth, gate counts, and qubit counts) can be compared under a consistent cost model. To this end, we adopt the following requirements: (1) a uniform input interface (e.g., reversible truth-table/permutation form) that supports S-boxes of varying bit-widths, (2) end-to-end automation from truth-table input to QASM output for reproducible benchmarking, and (3) circuit representations that reduce unnecessary redundancy where possible and remain amenable to consistent depth optimization under shared evaluation rules.

Reversible implementations of the same S-box are not unique: Different embeddings, ancilla policies, and decomposition strategies can yield different depth/width/gate-count trade-offs. For fair and reproducible comparisons across primitives, we therefore fix a benchmarking baseline (input format, I/O convention, and cost model) and evaluate synthesis/optimization results under the same protocol. Consequently, the reported metrics should be interpreted as model-dependent logical-level estimates under the chosen baseline.

Guided by these requirements, we adopt the bidirectional reversible-logic synthesis algorithm (Section 3) as the front-end module to reliably construct functionally correct circuits implementing the given reversible specification. However, reversible synthesis alone does not typically guarantee circuit optimality (e.g., minimum depth). We therefore modularize and chain post-synthesis components (decomposition and depth-oriented optimization modules in Section 4) to systematically improve synthesized circuits under the same cost model. The concrete module composition, input/output artifacts, and the automation boundary of this staged pipeline are detailed in Section 5.1.2 and Figure 6.

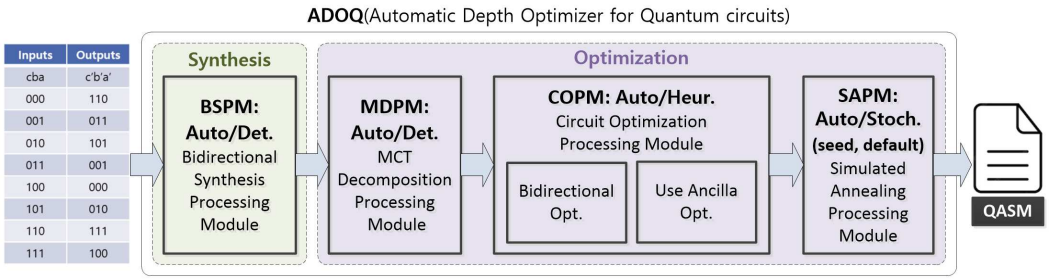

Figure 6. Overall architecture of ADOQ, illustrating the synthesis and multi-stage depth optimization pipeline from a reversible truth table to an optimized QASM circuit (Auto: automatic; Det.: deterministic; Heur.: heuristics; Stoch.: stochastic).

5.1.2. Architecture and Automation Boundary

Figure 6 illustrates the overall architecture of ADOQ as a sequential pipeline from a reversible truth table (e.g., CSV) to an optimized QASM circuit: BSPM (bidirectional reversible synthesis), MDPM (MCT decomposition), COPM (depth-oriented circuit optimization), and SAPM (simulated annealing refinement). To remove ambiguity about intermediate representations, ADOQ passes explicit artifacts between modules (summarized in Table 2).

Table 2. Intermediate artifacts (representations) passed between modules in ADOQ.

Artifact	Symbol	Description/Producer → Consumer
Reversible specification	T	Truth table (CSV); Input → BSPM
Circuit (MCT gates included.)	C_{MCT}	Sequence of gates; BSPM → MDPM
Circuit (MCT gates decomposed.)	C_{Tof}	Sequence of gates; MDPM → COPM/SAPM
QASM output	–	Exported QASM circuit; output

Under the default configuration used in this paper, all stages run automatically without manual circuit editing or manual selection of intermediate results. Users may optionally enable/disable modules or adjust a small set of global hyperparameters (e.g., SA budget or ancilla budget) for sensitivity studies; unless stated otherwise, these settings are fixed across all benchmarks to ensure fair and reproducible comparisons. As indicated in Figure 6, BSPM and MDPM are fully automatic and deterministic given the input and a fixed configuration. COPM is automatic and uses heuristic optimization logic (e.g., scheduling/rewrite choices) while applying predefined rule sets and objective functions uniformly across all instances (no per-benchmark manual tuning). SAPM is automatic but stochastic; it is enabled by default and made reproducible via a fixed annealing schedule and a fixed scaling parameter K across experiments:

1. BSPM constructs an initial reversible circuit satisfying the given truth table using the bidirectional synthesis algorithm.
2. MDPM decomposes MCT gates into combinations of standard Toffoli gates, improving hardware compatibility and preparing the circuit for subsequent depth optimization. In our benchmarking setting, MDPM follows the same minimum T-depth Toffoli decomposition as DORCIS; details and the ancilla clean-up property are specified in Section 6.
3. COPM applies depth-oriented optimization techniques such as gate reordering/elimination and ancilla-assisted parallelization while preserving logical equivalence.
4. SAPM performs global stochastic refinement using simulated annealing to further reduce circuit depth or overall cost by escaping local optima.

Each module can be enabled or disabled depending on the optimization objectives, allowing ADOQ to balance synthesis complexity and depth reduction under the same benchmarking protocol.

5.1.3. Key Design Principles

The design of ADOQ is guided by the following key principles.

- Modularity: Each synthesis and optimization algorithm is implemented as an independent module. This modular design allows individual components to be enabled, disabled, or replaced without affecting the overall framework, facilitating easy integration of new algorithms and future extensions.
- Automation: ADOQ provides end-to-end automation under a fixed default configuration; see Section 5.1.2 for the automation boundary and reproducibility settings (seeded simulated annealing enabled by default).
- Flexibility: The framework supports the selective activation of optimization modules, enabling users to tailor the optimization process according to specific performance objectives, hardware constraints, or available computational resources.
- Focus on depth optimization: While maintaining functional correctness, ADOQ prioritizes the reduction of circuit depth, as shorter execution depth directly translates to improved reliability and execution fidelity on real quantum hardware.

We use the notation in Table 3 consistently across Sections 3–6.

Table 3. Notation and reported metrics used throughout this paper.

Symbol	Meaning
n	Number of input qubits (S-box bit-width)
A	Number of ancilla qubits
G	Number of gates
C	Circuit
T	Truth table
D	Circuit depth (number of parallel layers)
W	Total logical qubits, $W = n + A$
K	Scaling parameter of SAPM
t	Temperature of SAPM (e.g., t_{init}, t_{cur})

We do not assume that mapping a classical primitive to a reversible circuit is unique or canonical. Multiple reversible realizations can implement the same function while exhibiting different trade-offs among ancilla usage, depth, and cost proxies (e.g., Toffoli/gate counts). In this work, we adopt a fixed embedding/decomposition choice primarily for reproducibility and controlled comparisons under a fixed benchmarking baseline; exploring a broader space of reversible decompositions is an important direction for future work.

5.2. Case Study: PRØST S-Box Optimization

To demonstrate the operation of ADOQ, the PRØST cipher's S-box [25] is selected as a case study. Table 4 presents the hexadecimal mapping of the PRØST S-box, which defines the input–output correspondence for 4-bit values.

Table 4. Input–output mapping of the PRØST S-box in hexadecimal notation.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	0	4	8	f	1	5	e	9	2	7	a	c	b	d	6	3

When the PRØST S-box truth table in Table 4 is provided as an input, the BSPM first synthesizes a reversible quantum circuit that satisfies the given input–output mapping. The synthesized circuit is then exported as a QASM file. The resulting circuit is composed of NOT, CNOT, Toffoli, and three controlled Toffoli (C3X) gates.

Although the resulting circuit is functionally correct, it contains MCT gates (e.g., C3X gates), which require further decomposition and optimization to reduce circuit depth and hardware efficiency. Figure 7 illustrates the circuit generated by BSPM.

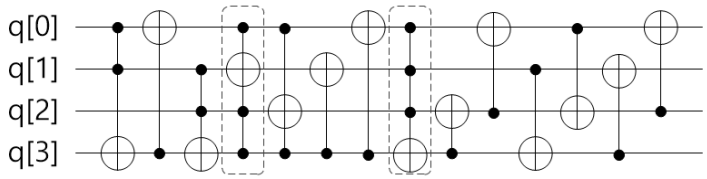

Figure 7. Reversible quantum circuit synthesized by BSPM for the PRØST S-box. Black dots represent control qubits, open circles denote target qubits, and dashed boxes highlight the MCT gates to be decomposed.

The MDPM module then decomposes these C3X gates into sequences of Toffoli gates using ancilla qubits. This decomposition simplifies the circuit structure and reveals opportunities for parallel execution. As a result, certain gates, such as subsequent CNOT operations, can be executed concurrently.

Figure 8 shows the circuit obtained after applying the MDPM module. In this stage, each C3X gate is decomposed into a sequence of standard Toffoli gates using an ancilla qubit. For example, the C3X(q[0], q[2], q[3], q[1]) gate is decomposed into a sequence of Toffoli gates involving the ancilla qubit, which enables subsequent CNOT(q[3], q[1]) operations to be executed in parallel. Similarly, the C3X(q[0], q[1], q[2], q[3]) gate is transformed into an equivalent Toffoli-based construction that exposes additional opportunities for concurrent execution, such as overlapping CNOT(q[3], q[0]) operations. As a result of MDPM, the circuit structure becomes more amenable to parallelization, reducing the overall circuit depth to 14.

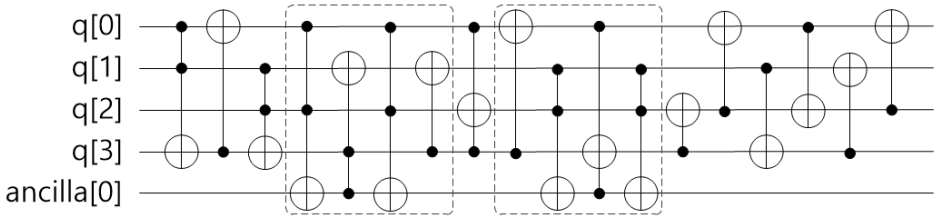

Figure 8. Circuit after MCT decomposition by the MDPM module using ancilla qubits. Black dots represent control qubits, open circles denote target qubits, and dashed boxes highlight the local gate blocks produced by MCT decomposition.

Next, the COPM module performs depth optimization by reordering gates and selectively introducing ancilla qubits. When consecutive gates share the same control qubits, COPM identifies sections that can be parallelized and applies ancilla-assisted transformations to eliminate unnecessary serialization.

Figure 9 illustrates an example of this process. In Figure 9a, COPM detects a circuit segment in which multiple Toffoli gates are executed sequentially due to shared control qubits. As shown in Figure 9b, an additional ancilla qubit (ancilla[1]) is introduced, and CNOT(q[2], ancilla[1]) gates are inserted at both ends of the selected section to preserve functional equivalence.

This transformation converts the Toffoli(q[1], q[2], q[3]) gate into Toffoli(q[1], ancilla[1], q[3]), allowing it to be executed concurrently with the Toffoli(q[0], q[2], ancilla[0]) gate within the same optimization region. As a result of COPM, the circuit depth is reduced from 14 to 13, and the overall cost is further lowered by enabling the parallel execution of these Toffoli gates.

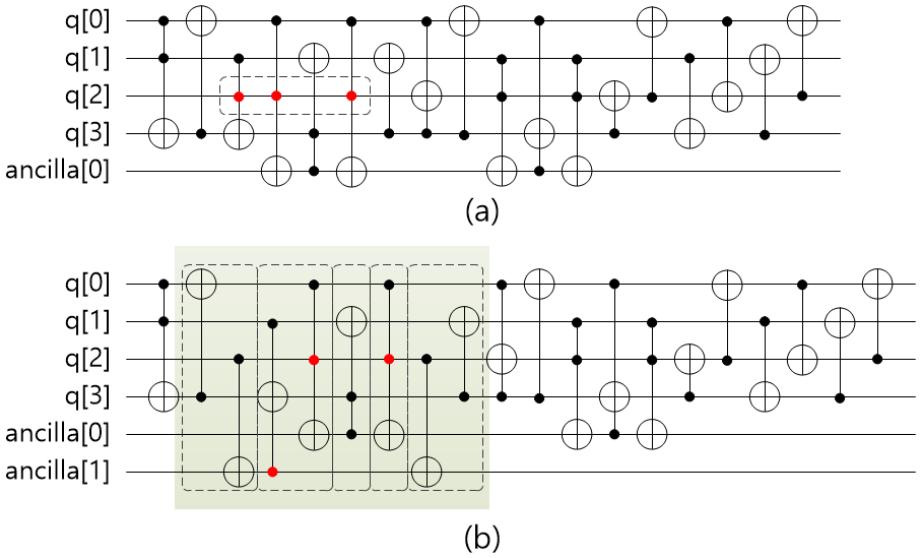

Figure 9. Ancilla-assisted depth optimization by COPM: (**a**) original gate sequence selected for optimization; (**b**) optimized circuit with parallel execution enabled. Black dots represent control qubits, open circles denote target qubits, and red dots indicate control qubits selected for optimization. Dashed boxes and the shaded region highlight the local gate window considered for COPM optimization (including ancilla-assisted updates), and the final arrangement illustrates gates grouped into the same depth layer.

Finally, SAPM is applied to perform global probabilistic optimization via the simulated annealing algorithm. At each iteration, the algorithm applies stochastic transformations—adding, removing, or replacing gates—while continuously evaluating the validity of the circuit using the Hamming distance against the target truth table.

The simulated annealing process aims to escape local minima and identify circuit configurations that further reduce circuit depth and gate count. Figure 10 shows the circuit obtained after applying SAPM. In this experiment, SAPM successfully synthesizes a PRØST S-box circuit with a depth of 7, without introducing any ancilla qubits. Although simulated annealing does not guarantee convergence to the global optimum, this result demonstrates its strong potential to discover compact circuit structures that are difficult to obtain through deterministic optimization methods.

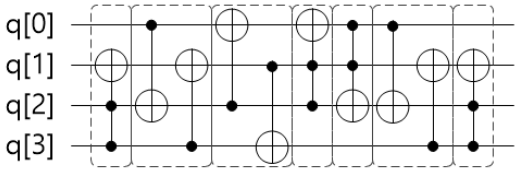

Figure 10. Circuit obtained after probabilistic global optimization using the SAPM module. Black dots represent control qubits, open circles denote target qubits, and dashed boxes indicate groups of gates executed within the same depth layer.

The PRØST S-box case study demonstrates the effectiveness of ADOQ's integrated optimization pipeline. By combining deterministic synthesis, structural decomposition, local depth optimization, and stochastic refinement within a single automated framework, ADOQ is able to generate functionally correct quantum circuits with significantly reduced depth. The resource comparison for the PRØST S-box across optimization stages is summarized in Table 5.

Table 5. Resource comparison for the PRØST S-box at each stage of the ADOQ optimization pipeline. Qubit/ancilla counts, gate counts by type (CX/CCX/C3X), total gates, and depth are reported for baseline and intermediate stages. "–" indicates zero ancilla.

Resources		Baseline (Input)	After MCT Decomposition	After Bi-Directional/ Ancilla Optimization	After SA Optimization
Qubits		4	4	4	4
Ancilla Qubits		–	1	2	–
Gates	CX (CNOT)	9	9	11	6
	CCX (Toffoli)	3	9	9	4
	C3X (MCT)	2	–	–	–
	SUM	14	18	20	10
Depth		12	14	13	7

The modular architecture of ADOQ allows the framework to be readily extended to other S-boxes and cryptographic primitives, supporting systematic and fair comparisons of quantum resistance among symmetric-key algorithms. Moreover, its design enables the future integration of emerging optimization techniques, such as template matching, gate commutation analysis, and learning-based synthesis methods.

6. Experimental Results

We present an experimental evaluation of ADOQ on S-box circuits derived from lightweight encryption algorithms that have been studied in previous works, including LIGHTER-R [2] and DORCIS [3].

The experiments are designed with two primary objectives. First, our goal is to verify the effectiveness of ADOQ's integrated depth optimization pipeline by comparing its results against existing synthesis and optimization methods on 4-bit S-boxes. Second, we demonstrate ADOQ's capability to synthesize and optimize larger S-boxes that are difficult or infeasible to handle using prior approaches.

To ensure fairness in comparison, ADOQ adopts the minimum T-depth Toffoli decomposition method [26], which is also used in DORCIS [3]. As illustrated in Figure 11, this decomposition realizes a Toffoli gate with a circuit depth of 7 and T-depth of 1 using four clean ancillas that are uncomputed back to $|0\rangle$ (hence being reusable and leaving no residual garbage on ancilla qubits). In this paper, the T-depth refers to the number of sequential layers of non-Clifford gates, where gates that can be executed in parallel are counted as a single layer. Based on this definition, ADOQ assigns a depth cost of 7 to a Toffoli gate, while Clifford gates such as H, X, NOT, and CNOT are assigned a unit depth cost of 1.

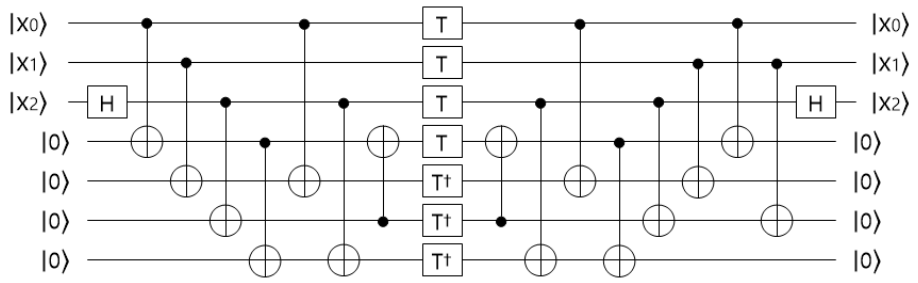

Figure 11. Minimum T-depth Toffoli decomposition used for fair comparison, achieving T-depth of 1 and circuit depth of 7 and requiring four ancilla qubits. Black dots represent control qubits, open circles denote target qubits, H denotes the Hadamard gate, T denotes the phase gate, and T† denotes the conjugate transpose of a T-gate.

6.1. Experimental Setup

We describe the experimental setup used to evaluate the effectiveness of ADOQ. Our experiments focus on the synthesis and optimization of S-box circuits, which are fundamental nonlinear components in symmetric-key cryptography and a primary target in quantum resource estimation.

For 4-bit S-box benchmarks, we compare ADOQ against the results reported in LIGHTER-R [2] and DORCIS [3] on the same benchmark set. To ensure fair comparison, we align (i) the evaluation gate set and counting rules, (ii) the reported metrics (depth, T-depth, and ancilla usage), and (iii) the depth-cost model and Toffoli decomposition rule used to compute ADOQ's results. For larger S-box instances, prior tools do not report results and are not applicable prior tools do not report results and are not applicable due to scalability limitations; thus, we report ADOQ-only results to demonstrate scalability.

We verify functional equivalence of all synthesized and optimized circuits with respect to the target S-box specification. Since our benchmarks are up to 8 bits, we validate circuits by exhaustively evaluating the input–output mapping on all 2^n inputs after key stages (BSPM/MDPM/COPM/SAPM), ensuring that depth/resource numbers are reported only for functionally equivalent circuits.

The runtime of ADOQ depends on the input size and the intermediate circuit structure. In practice, the dominant runtime cost is typically the stochastic refinement stage (SAPM), for which its runtime scales with the chosen budget and the circuit size.

ADOQ is heuristic and may not always yield further improvements under a fixed optimization budget, especially for larger instances. In such cases, ADOQ returns the best circuit found within the budget. We report these outcomes explicitly in our tables.

6.1.1. Benchmarks and Evaluation Protocol

Our evaluation is conducted in two complementary parts. In the first part, we consider 4-bit S-boxes used in lightweight block ciphers that were previously analyzed in the DORCIS framework [3], including S-boxes adopted from LIGHTER-R [2]. These S-boxes have been widely used as benchmark instances in prior quantum synthesis studies, allowing a direct comparison of synthesis and optimization methods under identical problem sizes. Using the same set of 4-bit S-boxes, we applied ADOQ and compared the resulting circuit depth and cost against the reported results of LIGHTER-R and DORCIS. In the second part, we evaluate the scalability of ADOQ by extending the synthesis to larger S-boxes with five and eight qubits. For instance, LIGHTER-R and DORCIS are not applicable, and no experimental results are reported in previous work. Accordingly, this experiment is conducted exclusively using ADOQ to demonstrate its capability to synthesize and optimize larger S-boxes beyond the practical limits of existing tools. The results are reported separately to highlight the scalability and generality of the proposed framework.

6.1.2. Default Configuration (Used Throughout Experiments)

Unless stated otherwise, we use the following default configuration:

- Input: Reversible truth table/permutation in CSV with a fixed I/O convention.
- BSPM: Bidirectional synthesis with fixed parameters.
- MDPM: Minimum T-depth Toffoli decomposition consistent with DORCIS; ancilla clean-up enabled (Figure 11).
- COPM: Bidirectional optimization + ancilla optimization enabled.
- SAPM: Simulated annealing enabled by default; fixed schedule, fixed scaling parameter K, and the optimization budget.
- Reporting: Logical-level depth, logical qubits, and gate counts under the fixed cost model.

We report logical-level resource metrics, including logical qubit counts (data + ancilla) and logical gate-cost proxies such as [Toffoli/T count, T-depth, circuit depth]. We do not include fault-tolerant overheads (e.g., logical-to-physical qubit expansion, syndrome extraction, or code-distance-dependent costs), because these depend strongly on the assumed physical error rates, target logical error, error-correcting code family, and architectural constraints. Accordingly, the reported numbers should be interpreted as model-dependent logical-level estimates intended for relative comparisons under a fixed-cost model, rather than definitive physical-resource projections.

6.1.3. Determinism and Variability

BSPM, MDPM, and COPM are deterministic given the same input and configuration, producing identical outputs across runs. In contrast, SAPM is stochastic due to simulated annealing and its output may vary with the scaling parameter K. To ensure reproducibility, we fix the SAPM scaling parameter K ($K = 10$ for 4-qubit S-box, $K = 30$ for 8-qubit S-box) and the optimization budget in all experiments.

6.2. Results on 4-Qubit S-Boxes

6.2.1. Input Specification and Conversion

Here, we focus on 4-qubit S-boxes used in lightweight symmetric-key cryptographic algorithms and show how different synthesis and optimization methods transform their lookup tables into optimized quantum circuits. Each S-box is originally specified as a hexadecimal lookup table (LUT), which is directly mapped to a reversible truth table by enumerating all 4-bit input values from 0000 to 1111. For example, DEFAULT-CORE [5]'s S-box is defined by the hexadecimal LUT *196F7C82AED043B5*. Interpreting this LUT in order, the *i-th* hexadecimal digit corresponds to the output value for the 4-bit input *i*. Thus, the first digit "1" represents the mapping "0000" $\rightarrow$ "0001." This direct LUT-to-truth-table conversion yields a reversible truth table, which serves as a unified input format for all synthesis and optimization methods evaluated in this section.

6.2.2. Representative Circuit Comparison

Figure 12 compares the quantum circuit implementations of the DEFAULT-CORE S-box across three synthesis frameworks:

- LIGHTER-R synthesizes the S-box using six Toffoli gates, each contributing a depth of 7, along with three Clifford gates of depth 1, resulting in a total circuit weighted depth of 45.
- DORCIS employs four Toffoli gates and five Clifford gates. By identifying Clifford operations that can be executed in parallel, DORCIS reduces the overall circuit weighted depth to 31.
- ADOQ synthesizes the same S-box using four Toffoli gates and eight Clifford gates. Although the number of Clifford operations is higher, ADOQ exploits concurrency among gates to achieve a total circuit weighted depth of 32.

ADOQ and DORCIS differ in how they handle qubit line ordering. ADOQ preserves a fixed one-to-one input–output qubit-line alignment (no wire permutation across the S-box boundary). DORCIS may apply wire permutations (qubit relabeling) to reduce depth, which can be more beneficial in some small 4-qubit instances. We keep a fixed interface to ensure structural consistency when composing S-box modules within larger encryption circuits; otherwise, extra routing (e.g., SWAPs, often realized by three CNOTs) may be needed to match the assumed line ordering.

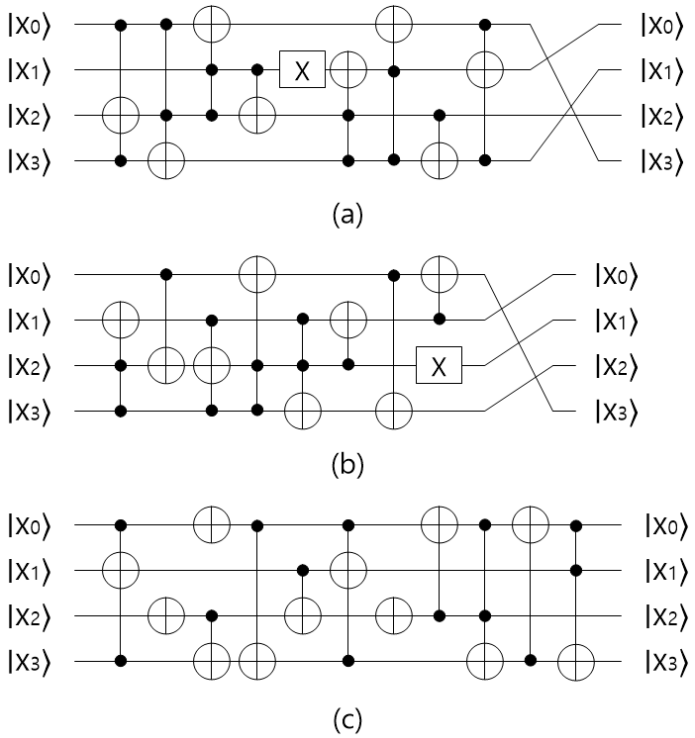

Figure 12. DEFAULT-CORE S-box implementation: (**a**) LIGHTER-R (weighted depth of 45). (**b**) DORCIS (weighted depth of 31). (**c**) ADOQ (weighted depth of 32). Black dots represent control qubits, open circles denote target qubits, and X denotes the Pauli-X gate.

6.2.3. Aggregate Results and Discussion

Table 6 summarizes the synthesis results for multiple 4-qubit S-boxes. Overall, the results show that ADOQ achieves performances comparable to or slightly better than LIGHTER-R, while remaining close to DORCIS in terms of circuit depth, with the added benefit of a more general-purpose and modular synthesis framework.

Table 6. Weighted depth/T-depth comparison for 4-qubit S-box synthesis (LIGHTER-R, DORCIS, and ADOQ) and gate composition of ADOQ outputs. Depth is the number of gate layers under disjoint-qubit parallelization on the final circuit; T-depth is the number of T-gate layers under the same rule. For ADOQ, we also report the counts of X, CX, and CCX gates (SUM: total).

S-Box	LIGHTER-R [2]		DORCIS [3]		ADOQ		ADOQ Gate Composition			
	Depth	T-Depth	Depth	T-Depth	Depth	T-Depth	X	CX	CCX	SUM
DEFAULT-CORE [5]	45	6	31	4	32	4	3	5	4	12
GIFT [27]	32	4	31	4	32	4	2	4	4	10
NOEKEON-GAMMA [28]	32	4	30	4	34	4	2	8	4	14
PICCOLO [29]	32	4	30	4	32	4	3	5	4	12
PRESENT [20]	33	4	32	4	35	4	6	6	4	16
PYJAMASK-4 [30]	32	4	31	4	34	4	2	6	4	12
RECTANGLE [31]	33	4	31	4	33	4	2	5	4	11
SKINNY [32]	32	4	30	4	32	4	2	5	4	11

On average, ADOQ achieves an approximately 2.58% improvement in circuit depth compared to LIGHTER-R while producing circuits that are 6.81% deeper than DORCIS. Concretely, as shown in Table 6, enforcing fixed input–output alignment incurs an overhead of 1–4 additional depth layers (2.25 layers on average, corresponding to 6.81% on average) relative to DORCIS, which permits wire permutations. However, ADOQ offers greater

generality by enabling automatic synthesis and optimization across different S-boxes, which is especially useful for scalable designs and integration into larger cryptographic circuits.

6.3. Results on Larger S-Boxes

6.3.1. Main Results

While most previous studies focused on 4-qubit S-boxes due to the exponential growth of the search space, ADOQ extends applicability to larger S-boxes by integrating automated optimization algorithms that handle increased qubit counts. To demonstrate scalability, we applied ADOQ to synthesize the S-box of the ASCON algorithm [6] (5 qubits) and the AES algorithm [7] (8 qubits). Table 7 shows the comparative results. For these larger S-boxes, prior methods such as LIGHTER-R and DORCIS do not report results due to scalability limitations, and thus, only ADOQ results are shown.

Table 7. Weighted depth comparison of more qubits S-box circuit synthesis of ADOQ with previous works.

S-Box	LIGHTER-R [2]	DORCIS [3]	ADOQ
ASCON [6]-5 qubits	-	-	273
AES [7]-8 qubits	-	-	6093

For the 5-qubit ASCON S-box, the final optimized circuit uses up to 5 additional ancilla qubits (peak ancilla usage). The synthesized circuit consists of 72 quantum gates, including one NOT gate, 28 CNOT gates, and 43 Toffoli gates, resulting in an optimized circuit weighted depth of 273. For the 8-qubit AES S-box, the final optimized circuit uses up to 9 additional ancilla qubits. In this case, the circuit contains a total of 1674 quantum gates, including three NOT gates, 433 CNOT gates, and 1238 Toffoli gates, and the weighted depth of the circuit was optimized to 6093.

The reported weighted depths (e.g., 6093 for the AES S-box) are logical-level metrics computed under our reversible gate library and scheduling rules. They do not include architecture-dependent overheads such as decomposing Toffoli/MCT operations into a device's native 1–2 qubit gate set, routing constraints (e.g., SWAP insertion due to limited connectivity), or fault-tolerant error-correction costs. Accordingly, we do not claim direct executability on near-term NISQ devices; instead, we present these results as reproducible resource benchmarks that can inform architecture-aware compilation and fault-tolerant resource estimation.

6.3.2. Ablation and Sensitivity

Tables 8 and 9 quantify the depth–ancilla trade-off for the ASCON (5-bit) and AES (8-bit) S-boxes and provide a stage-wise ablation of the ADOQ pipeline by reporting intermediate metrics after each stage (baseline/BSPM, after MDPM decomposition, after COPM optimization, and after SAPM refinement). After MCT decomposition, the circuits use 2 (ASCON) and 5 (AES) algorithmic ancilla qubits. The ancilla-assisted stage increases the ancilla usage to 5 (ASCON) and 9 (AES), enabling more parallelism and reducing depths from 54 to 52 for ASCON (3.70%) and from 1011 to 980 for AES (3.07%). The weighted depth shows a similar reduction from 300 to 286 for ASCON (4.67%) and from 6495 to 6134 for AES (5.56%). Finally, simulated annealing further reduces depth without increasing ancilla (ASCON: 52→51; AES: 980→969), illustrating that additional refinement is possible even under a fixed ancilla budget.

Table 8. Resource comparison for the ASCON S-box at each stage of the ADOQ optimization pipeline. Qubit/ancilla counts, gate counts by type (CX/CCX/C3X/C4X), total gates, and depth are reported for baseline and intermediate stages. "–" indicates zero ancilla. "Qubits" denotes data qubits (S-box bit-width), and "Ancilla Qubits" denotes additional algorithmic ancilla.

Resources		Baseline (Input)	After MCT Decomposition	After Bi-Directional/ Ancilla Optimization	After SA Optimization
Qubits		5	5	5	5
Ancilla Qubits		–	2	5	5
Gates	X (NOT)	1	1	1	1
	CX (CNOT)	20	20	28	28
	CCX (Toffoli)	14	43	43	43
	C3X (MCT)	9	–	–	–
	C4X (MCT)	2	–	–	–
	SUM	46	64	72	72
Depth		46	54	52	51
Weighted Depth		–	300	286	273

Table 9. Resource comparison for the AES S-box at each stage of the ADOQ optimization pipeline. Qubit/ancilla counts, gate counts by type (CX/CCX/C3X/C4X/C5X/C6X/C7X), total gates, and depth are reported for baseline and intermediate stages. "–" indicates zero ancilla. "Qubits" denotes data qubits (S-box bit-width), and "Ancilla Qubits" denotes additional algorithmic ancilla.

Resources		Baseline (Input)	After MCT Decomposition	After Bi-Directional/ Ancilla Optimization	After SA Optimization
Qubits		8	8	8	8
Ancilla Qubits		–	5	9	9
Gates	X (NOT)	3	3	3	3
	CX (CNOT)	297	297	473	433
	CCX (Toffoli)	165	1238	1238	1238
	C3X (MCT)	113	–	–	–
	C4X (MCT)	95	–	–	–
	C5X (MCT)	59	–	–	–
	C6X (MCT)	27	–	–	–
	C7X (MCT)	7	–	–	–
	SUM	766	1538	1714	1674
Depth		724	1011	980	969
Weighted Depth		–	6495	6134	6093

Overall, the results demonstrate that ADOQ not only matches the depth optimization performance of prior approaches such as DORCIS on small S-boxes but also extends their applicability to larger S-box constructions.

While DORCIS achieves slightly better depth for 4-qubit S-boxes by allowing line crossings, it is mainly designed for small, fixed-size circuits and does not scale well with respect to larger S-boxes. In contrast, ADOQ is designed to preserve consistent qubit ordering and to integrate heuristic and probabilistic optimization techniques, enabling efficient synthesis and depth optimization for S-boxes with five or more qubits.

By maintaining structural consistency in qubit mapping and supporting a fully automated optimization pipeline, ADOQ provides a reproducible and fair benchmarking framework for evaluating the quantum resistance of symmetric-key algorithms. These properties make ADOQ not only an optimization tool but also a practical platform for analyzing and comparing cryptographic S-boxes in the quantum setting.

Regarding sensitivity, we fix the SAPM hyperparameters (scaling parameter K and optimization budget) as specified in Section 6.1.3; a broader parameter sweep is left as future work.

7. Conclusions

In this paper, we present ADOQ (Automatic Depth Optimizer for Quantum circuits), a modular and automated framework for synthesizing and depth-optimizing reversible quantum circuits for S-box functions. ADOQ takes an S-box lookup table, converts it into a reversible truth table, and generates an optimized QASM circuit through an end-to-end pipeline.

ADOQ first constructs an initial circuit using the bidirectional synthesis algorithm and then applies four depth optimization modules: MCT decomposition, bidirectional optimization, ancilla-assisted optimization, and simulated annealing. In our experiments on multiple 4-qubit S-boxes, ADOQ achieved depth results comparable to existing methods, while providing a unified and reproducible workflow. Importantly, ADOQ also demonstrated scalability beyond 4 qubits by synthesizing and optimizing larger S-boxes such as ASCON (5-bit) and AES (8-bit), which is difficult for prior approaches that focus on small fixed-size settings.

Moreover, circuit cost should not be interpreted as cryptographic security: Security depends on the best-known attack complexity, whereas our results focus on implementation-level resource benchmarking under a fixed reversible specification and cost model. Finally, while we provide empirical scaling results, deriving tight analytical bounds for the full pipeline is beyond the scope of this work due to heuristic and representation-dependent optimization stages.

As future work, we plan to extend ADOQ with additional synthesis strategies and validate the optimized circuits on real quantum hardware.

Author Contributions: Conceptualization, D.C.; methodology, C.C., J.O., G.C. and D.C.; software, C.C.; validation, J.O., S.L., G.C. and D.C.; formal analysis, S.L., G.C. and D.C.; investigation, C.C.; data curation, S.L. and G.C.; writing—original draft, C.C. and J.O.; writing—review and editing, S.L., G.C. and D.C.; supervision, D.C.; project administration, D.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partly supported by the IITP grant funded by the Government of Korea (MSIT) (IITP-2026-RS-2022-00164800) (10%); Q-Crypton (No. 2019-0-00033) (40%); the NRF "Quantum Computing based on Quantum Advantage Challenge" program (RS-2023-00256221) (40%); and the Basic Science Research Program through the NRF funded by the Ministry of Education (No. RS-2025-25411243) (10%). The APC was funded by Q-Crypton (No. 2019-0-00033).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding authors.

Conflicts of Interest: Authors Chanho Choi, SangMan Lee and Dooho Choi are employed by Seqrypton Inc. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Grover, L.K. A fast quantum mechanical algorithm for database search. In *Twenty-Eighth Annual ACM Symposium on Theory of Computing*; Association for Computing Machinery: New York, NY, USA, 1996; pp. 212–219.
2. Dasu, V.A.; Baksi, A.; Sarkar, S.; Chattopadhyay, A. Lighter-r: Optimized reversible circuit implementation for sboxes. In *2019 32nd IEEE International System-on-Chip Conference (SOCC)*; IEEE: New York, NY, USA, 2019; pp. 260–265.
3. Chun, M.; Baksi, A.; Chattopadhyay, A. DORCIS: Depth Optimized Quantum Implementation of Substitution Boxes. *Cryptol. Eprint Arch.* **2023**, *2023*, 1–10.

4. Pan, D.; Long, G.L.; Yin, L.; Sheng, Y.B.; Ruan, D.; Ng, S.X.; Lu, J.; Hanzo, L. The Evolution of Quantum Secure Direct Communication: On the Road to the Qinternet. *IEEE Commun. Surv. Tutor.* **2024**, *26*, 1898–1949. [CrossRef]
5. Baksi, A. *Classical and Physical Security of Symmetric Key Cryptographic Algorithms*; Springer: Singapore, 2022.
6. Dobraunig, C.; Eichlseder, M.; Mendel, F. Ascon v1.2: Lightweight Authenticated Encryption and Hashing. *J. Cryptol.* **2021**, *34*, 33. [CrossRef]
7. *FIPS PUB 197*; Announcing the Advanced Encryption Standard (AES). National Institute of Standards and Technology: Gaithersburg, MD, USA, 2001. Available online: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf (accessed on 18 January 2026).
8. Shende, V.V.; Prasad, A.K.; Markov, I.L.; Hayes, J.P. Synthesis of Reversible Logic Circuits. *IEEE Trans.-Comput.-Aided Des. Integr. Circuits Syst.* **2003**, *22*, 710–722. [CrossRef]
9. Prasad, A.K.; Shende, V.V.; Markov, I.L.; Hayes, J.P.; Patel, K.N. Data Structures and Algorithms for Simplifying Reversible Circuits. *ACM J. Emerg. Technol. Comput. Syst.* **2006**, *2*, 277–293. [CrossRef]
10. Li, Z.; Chen, H.; Xu, B.; Song, X.; Xue, X. An Algorithm for Synthesis of Optimal 3-qubit Reversible Circuits Based on Bit Operation. In Proceedings of the Second International Conference on Genetic and Evolutionary Computing (WGEC 2008), Jinzhou, China, 25–26 September 2008; pp. 455–458. [CrossRef]
11. Yang, G.; Song, X.; Hung, W.N.N.; Perkowski, M.A. Bi-Directional Synthesis of 4-Bit Reversible Circuits. *Comput. J.* **2008**, *51*, 207–215. [CrossRef]
12. Golubitsky, O.; Falconer, S.M.; Maslov, D. Synthesis of the Optimal 4-bit Reversible Circuits. In Proceedings of the 47th Annual Design Automation Conference (DAC '10), Anaheim, CA, USA, 13–18 June 2010; pp. 653–656. [CrossRef]
13. Golubitsky, O.; Maslov, D. A Study of Optimal 4-Bit Reversible Toffoli Circuits and Their Synthesis. *IEEE Trans. Comput.* **2012**, *61*, 1341–1353. [CrossRef]
14. Kliuchnikov, V.; Maslov, D. Optimization of Clifford Circuits. *Phys. Rev. A* **2013**, *88*, 052307. [CrossRef]
15. Feng, J.; Wei, Y.; Zhang, F.; Pasalic, E.; Zhou, Y. Novel Optimized Implementations of Lightweight Cryptographic S-Boxes via SAT Solvers. *IEEE Trans. Circuits Syst. Regul. Pap.* **2024**, *71*, 334–347. [CrossRef]
16. Jeon, Y.; Baek, S.; Kim, J. A Novel Framework to Construct S-Box Quantum Circuits Using System Modeling: Application to 4-Bit S-Boxes. *Comput. Model. Eng. Sci.* **2024**, *141*, 545–561. [CrossRef]
17. Chung, D.; Lee, S. Quantum Implementation of S-Boxes Based on Polynomial Evaluation. *Electron. Lett.* **2025**, *61*, e70337. [CrossRef]
18. Lin, D.; Yang, C.; Xu, S.; Tian, S.; Sun, B. On the construction of quantum circuits for S-boxes with different criteria based on the SAT solver. *Quantum Inf. Process.* **2026**, *25*, 39. [CrossRef]
19. Miller, D.M.; Maslov, D.; Dueck, G.W. A transformation based algorithm for reversible logic synthesis. In Proceedings of the 40th Annual Design Automation Conference, Anaheim, CA, USA, 2–6 June 2003.
20. Bogdanov, A.; Knudsen, L.R.; Leander, G.; Paar, C.; Poschmann, A.; Robshaw, M.J.B.; Seurin, Y.; Vikkelsoe, C. PRESENT: An ultra-lightweight block cipher. In *CHES 2007*; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4727, pp. 450–466.
21. Lee, J.; Kang, Y.; Lee, Y.S.; Chung, B.; Choi, D. MPMCT gate decomposition method reducing T-depth quickly in proportion to the number of work qubits. *Quantum Comput. Eng.* **2023**, *22*, 381.
22. Zhu, C.; Huang, Z. Optimizing the depth of quantum implementations of linear layers. In *International Conference on Information Security and Cryptology*; Springer: Cham, Switzerland, 2022; pp. 129–147.
23. Abdessaied, N.; Wille, R.; Soeken, M.; Drechsler, R. Reducing the Depth of Quantum Circuits Using Additional Circuit Lines. In *Reversible Computation (RC 2013), Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 7948, pp. 221–233. [CrossRef]
24. Shahidi, S.M.; Borujeni, S.E. A new method for reversible circuit synthesis using a Simulated Annealing algorithm and don't-cares. *J. Comput. Electron.* **2021**, *20*, 718–734. [CrossRef]
25. Kavun, E.B.; Lauridsen, M.M.; Leander, G.; Rechberger, C.; Schwabe, P.; Yalçın, T. Prøst v1.1. CAESAR Submission. 2014. Available online: http://competitions.cr.yp.to/round1/proestv11.pdf (accessed on 29 June 2025).
26. Selinger, P. Quantum circuits of T-depth one. *Phys. Rev. A* **2013**, *87*, 042302. [CrossRef]
27. Banik, S.; Pandey, S.K.; Peyrin, T.; Sasaki, Y.; Sim, S.M.; Todo, Y. GIFT: A small present—Towards reaching the limit of lightweight encryption. In Proceedings of the CHES 2017: Cryptographic Hardware and Embedded Systems, Taipei, Taiwan, 25–28 September 2017; pp. 321–345.
28. Daemen, J.; Peeters, M.; Assche, G.V.; Rijmen, V. The Noekeon Block Cipher. 2000. Available online: https://gro.noekeon.org/Noekeon-spec.pdf (accessed on 10 January 2026).
29. Shibutani, K.; Isobe, T.; Hiwatari, H.; Mitsuda, A.; Akishita, T.; Shirai, T. Piccolo: An ultralightweight blockcipher. In Proceedings of the CHES 2011, Nara, Japan, 28 September–1 October 2011; pp. 342–357.

30. Goudarzi, D.; Jean, J.; Kölbl, S.; Peyrin, T.; Rivain, M.; Sasaki, Y.; Sim, S.M. Pyjamask. v1.0. 2019. Available online: https://csrc.nist.gov/CSRC/media/Projects/lightweight-cryptography/documents/round-2/spec-doc-rnd2/pyjamask-spec-round2.pdf (accessed on 15 February 2026).
31. Zhang, W.; Bao, Z.; Lin, D.; Rijmen, V.; Yang, B.; Verbauwhede, I. RECTANGLE: A bit-slice lightweight block cipher suitable for multiple platforms. *Sci. China Inf. Sci.* **2015**, *58*, 1–15. [CrossRef]
32. Beierle, C.; Jean, J.; Kölbl, S.; Leander, G.; Moradi, A.; Peyrin, T.; Sasaki, Y.; Sasdrich, P.; Sim, S.M. The SKINNY family of block ciphers and its low-latency variant MANTIS. In Proceedings of the CRYPTO 2016, Part II, Santa Barbara, CA, USA, 14–18 August 2016; pp. 123–153.