# Qubit Optimized Quantum Implementation of SLIM

H. O. Cildiroglu[1,2,3*] and O. Yayla[3]

[1]Physics Department, Boston University, Commonwealth Ave, Boston, 02100, MA, USA.
[2]Department of Physics Engineering, Ankara University, Dögol st, 06100, Ankara, Turkiye.
[3]Institute of Applied Mathematics, Middle East Technical University, Universiteler, Ankara, 06800, Ankara, Turkiye.

*Corresponding author(s). E-mail(s): hoc@physics.bu.edu;
Contributing authors: oguz@metu.edu.tr;

**Abstract**

The advent of quantum computing has profound implications for current technologies, offering advancements in optimization while posing significant threats to cryptographic algorithms. Public-key cryptosystems relying on prime factorization or discrete logarithms are particularly vulnerable, whereas block ciphers (BCs) remain secure through increased key lengths. In this study, we introduce a novel quantum implementation of SLIM, a lightweight block cipher optimized for 32-bit plaintext and an 80-bit key, based on a Feistel structure. This implementation distinguishes itself from other BC quantum implementations in its class (64–128-bit) by utilizing a minimal number of qubits while maintaining robust cryptographic strength and efficiency. By employing an innovative design that minimizes qubit usage, this work highlights SLIM's potential as a resource-efficient and secure candidate for quantum-resistant encryption protocols.

**Keywords:** SLIM, Block Ciphers, Quantum Implementation

# 1 Introduction

Classical cryptography methods primarily hinge on exploiting computational inefficiencies within mathematical problems such as prime factorization and discrete logarithms to uphold the security of encrypted data. The potential of quantum computers to efficiently solve these problems, which pose significant challenges for classical computers, using quantum algorithms fundamentally threatens the security of existing encryption protocols [1–7]. Therefore, there is a burgeoning interest in crafting algorithms resilient to quantum computing methods, ensuring secure communication and data protection [8–14]. In this context, it is crucial to construct quantum algorithms for classical encryption methods, strengthening them against potential threats and thoroughly assessing their resilience to quantum attacks [15–23].

Block ciphers (BC) are still considered to be quantum-resilient algorithms employed for encrypting data blocks or fixed-length bit groups. Most BCs are designed to encrypt data in predefined blocks of either 64 or 128 bits in size [24–39]. The Feistel structure is a fundamental design element of symmetric BC algorithms, enabling the partitioning of blocks, the use of subkeys, round-based operations, and an iterative structure to perform both encryption and decryption processes with the same key [24, 25]. Designed to enhance encryption security, it provides a flexible framework that can be utilized in many encryption algorithms. SLIM, a novel variant of the Feistel-based BCs with 32-bit block size, is designed for lightweight applications from contemporary RFID technologies to the Internet of Things (IoTs), striking a balance between security and efficiency [40]. In the construction of SLIM, four 4x4 S-boxes are utilized to function as a non-linear component of the cipher, executing a non-linear operation on a 16-bit word. In this regard, SLIM is proposed to be secure, despite its low bit usage compared to other BCs.

In this work, we present a quantum implementation of SLIM with the aim of developing encryption methods that are robust against emerging quantum technologies. Achieving an optimized quantum cost, our implementation employs the smallest number of qubits compared to other BC quantum implementations in existing literature. To achieve this, we conduct a comprehensive analysis of the SLIM algorithm in the context of quantum circuits, focusing on its Key Addition-Substitution-Permutation (KSP) layers. By strategically reversing the KSP structure of SLIM and avoiding the use of ancilla qubits, we successfully construct for the first time the algorithm with a total of 112 qubits and a quantum cost of 27220. This approach lays the foundation for a practical implementation in current technology, providing an opportunity to test and enhance its resilience in contrast to quantum attacks.

This paper is organized as follows. In Section II, we present the KSP structure of the SLIM, along with the notation designed for quantum circuits. In Section III, we perform the quantum implementation of SLIM using quantum gates. The fourth and final section is dedicated to calculating the quantum costs associated with the implementation of SLIM and discussing it in comparison with other BC implementations.

## 2 The structure of SLIM

The development of quantum technologies poses a significant threat to the security of existing cryptographic algorithms, encouraging the exploration of quantum-resilient (or post-quantum) cryptography. In this regard, BC stands out with its simple but resilient design. However, some BC algorithms may be solved by quantum computers due weakness in their design, eg Grover search attack on their differential characteristics [41]. On the other hand, SLIM, a recently introduced variant of BC, promises to be a quantum-resistant algorithm with its lightweight design and a well-balanced Feistel structure in terms of security and efficiency [40].

SLIM is a symmetric encryption algorithm which has Feistel structure that uses the same key for both encryption and decryption. The only difference between these processes is the use of decryption sub-keys in reverse order. SLIM integrates both confusion and diffusion principles. Confusion is efficiently handled through a compact 4-bit S-box, using the same structure of PRESENT [31]. Simplicity is a crucial aspect of SLIM, evident in the compact size of the S-box (Fig. 1) and the use of inherently straightforward operations. Operating with an 80-bit key, SLIM is designed to encrypt and decrypt 32-bit plaintext and ciphertext blocks.

| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S(x) | C | 5 | 6 | B | 9 | 0 | A | D | 3 | E | F | 8 | 4 | 7 | 1 | 2 |

**Fig. 1** S-box of SLIM

The rearrangement phase of SLIM utilizes a crucial permutation layer, generating a 16-bit output from 16-bit inputs through a meticulously selected rule specified in Fig. 2. This choice as mentioned in its proposal [40] arises from a need for securing against both linear and differential cryptanalysis. In linear cryptanalysis, the permutation rule is designed to resist patterns, creating substantial confusion to hinder adversaries from identifying patterns within the cipher. Simultaneously, the rule takes into account differential cryptanalysis, disrupting systematic relationships between input and output differentials to fortify the cipher's resilience against such attacks. Hence, these techniques play a role in thwarting various cryptanalytic approaches, illustrating the algorithm's effectiveness and suitability for secure information transmission.

| x | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P(x) | 7 | 13 | 1 | 8 | 11 | 14 | 2 | 5 | 4 | 10 | 15 | 0 | 3 | 6 | 9 | 12 |

**Fig. 2** P-box of SLIM

The 32 sub-keys (16-bit each), required for 32 rounds, are generated from the 80-bit encryption key in SLIM. NIST guidelines recommend a key length of at least 80 to resist exhaustive search attacks [42]. The initial five sub-keys $\{K_1, \ldots, K_5\}$ are directly derived from the original key ($K = M_0 M_1 \ldots M_{39} L_{39} \ldots L_1 L_0$). Specifically, $K_1$ corresponds to the first least significant 16 bits, $K_2$ to the subsequent 16 bits, and

so forth (up to the fifth round). The 80-bit key is then divided by a splitter, yielding two 40-bit values as the most and least significant bits (MSB and LSB). Each half is subsequently processed individually (See Fig. 3). In each round, the LSB undergoes a left cyclical shift (LCS) of two bits, followed by an XOR operation with the MSB. The output of this XOR process is then passed to a substitution layer. The round sub-key is created by manipulating the output of the S-boxes and the rotated MSB, achieved through an LCS of 3 bits using the XOR technique.

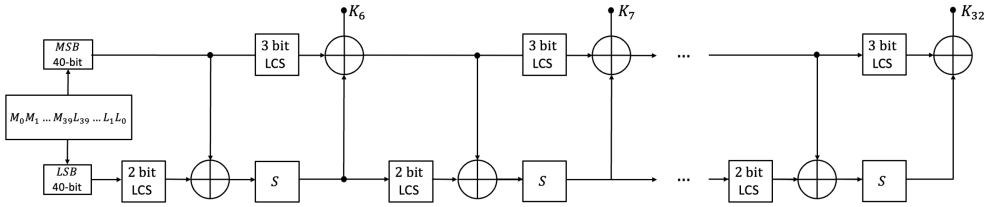



**Fig. 3** Key Generation of SLIM ($i > 5$).

In the SLIM's encryption algorithm, the input splits into up and down parts, which go through 32 rounds of processing along with the generated sub-keys. The interior structure of SLIM is given in Fig. 4. According to this, the 32-bit input is split into two equal sixteen-bit halves as $U_i$ and $D_i$, where $U_i = D_{i-1}$ and $D_i = U_{i-1} \oplus P(S(K_i \oplus D_{i-1}))$.

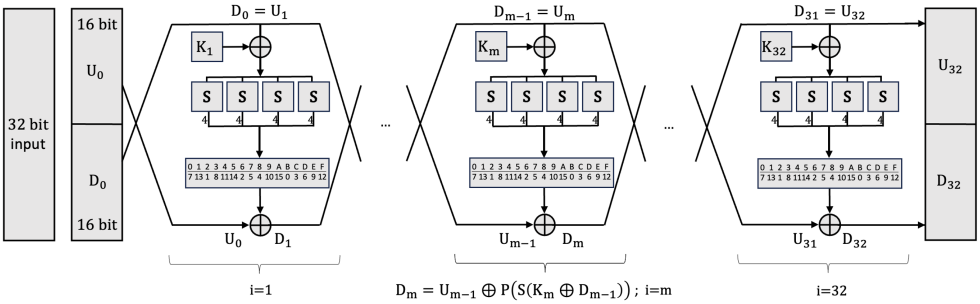



**Fig. 4** P-box of SLIM

## 3 Quantum Implementation of SLIM

In quantum mechanics, the state of a system is described by wave functions ($|\psi\rangle$), defined in the Hilbert space ($\mathcal{H}$)-a vectorial complex inner product space. These wave functions represent the probability amplitude of a particle that exhibits a specific property (e.g. position or spin) upon measurement. Thus, a quantum state is a complex

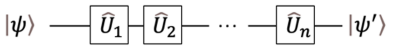
linear superposition of possible substates within this space. Single-particle quantum systems with two subparts, such as horizontal $|H\rangle$ and vertical $|V\rangle$ polarization, spin-up $|\uparrow\rangle$ and spin-down $|\downarrow\rangle$, or more broadly $|1\rangle$ and $|0\rangle$, are referred to as quantum bits (or simply qubits). Qubits are the quantum counterparts of classical bits, the fundamental units of classical computation. The general state of a single qubit is expressed as:

$$|\psi\rangle = a|1\rangle + b|0\rangle; \quad |\psi\rangle \in \mathcal{H}_2; \quad a, b \in \mathbb{C}; \quad a^2 + b^2 = 1. \tag{1}$$

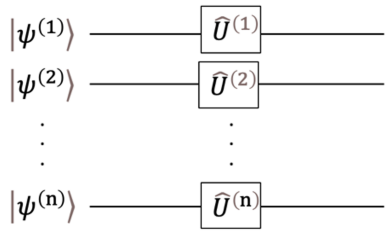
Composite systems are formed by combining two or more discrete and separately prepared qubits ($|\psi^{(i)}\rangle \in \mathcal{H}^i; i = 1, \ldots, n$), which reside in the tensor product space $\mathcal{H}^{\otimes n}$, can be given as:

$$|\Psi\rangle = |\psi^{(1)}\rangle \otimes \cdots \otimes |\psi^{(n)}\rangle \quad (\text{or simply } |\psi^{(1)} \ldots \psi^{(n)}\rangle), \tag{2}$$

where $|\Psi\rangle \in \mathcal{H} = \mathcal{H}^{(1)} \otimes \cdots \otimes \mathcal{H}^{(n)}$. Besides, due to the causality of the theory, the evolution of the system can be controlled by unitary operators such as $|\Psi'\rangle = \hat{U}|\Psi\rangle$. If the system undergoes a stepwise evolution, its state is transformed sequentially by a series of unitary operators. The final output state of the system is $|\Psi'\rangle = \hat{U}_n \ldots \hat{U}_2 \hat{U}_1 |\Psi\rangle$ or, equivalently,




On the other hand, the transformed state of the $n$-qubit system is $|\Psi'\rangle = \left[\hat{U}^{(1)} \otimes \cdots \otimes \hat{U}^{(n)}\right] \left[|\psi^{(1)}\rangle \otimes \cdots \otimes |\psi^{(n)}\rangle\right]$ or equivalently,




Here, the evolution operator $\hat{U}^{(i)}$ acts on the ket $|\psi^{(i)}\rangle$. These representations are suitable for constructing quantum implementations of the S, P, and K layers for the encryption algorithm given in the previous section. To achieve this, some quantum gates such as the X (or simply NOT), Hadamard ($H$) gate, CNOT, CCNOT (or Toffoli), and SWAP gates need to be introduced (See Fig. 5).

To implement SLIM as a quantum circuit, the 32-qubit input is first divided into two equal parts: the upper half ($U$) and the lower half ($D$). The 16 qubits ($D_0$) from the D-box are combined with the first round key ($K_1$), derived by dividing the 80-bit key into five equal parts. Next, the S-box is executed four times in parallel, starting from the least significant qubits (LSQ). The resulting outputs are then passed through

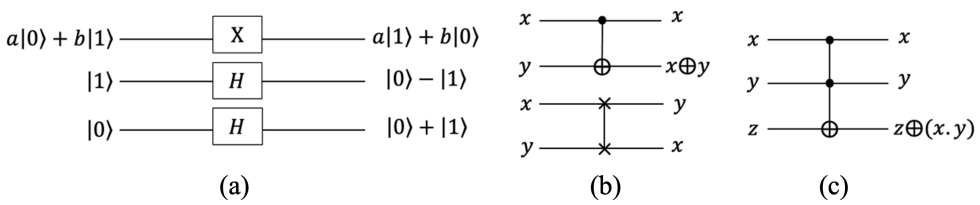



**Fig. 5** Quantum gates: (a) X: NOT, H: Hadamard; (b-top) CNOT: Controlled-NOT, (b-bottom) SWAP; (c) CCNOT. Here $a, b \in \mathbb{C}$, $x, y, z \in \{0, 1\}$, and $\oplus$ is the bitwise XOR.

the P-box. For simplicity, this sequence of operations of key addition, substitution and permutation layers is referred to as KSP. Following the KSP process, the results are manipulated using CNOT gates with the 16 qubits ($U_0$) from the U-box, and the first round is completed by obtaining $D_1$.

At this stage, it is essential to ensure the feasibility of implementing the second and subsequent rounds of SLIM. When KSP is applied on $D_0$, it transforms into $U_1$. While this transition is straightforward from a classical perspective, it introduces challenges in quantum paradigm, as it necessitates duplicating the $D_0$ packet, which is a task that inherently requires additional qubits. This duplication involves adding and applying CNOT gates to a set of ancilla qubits, which are initialized to the $|0\rangle$ state, matching the size of the packet (16 qubits). Moreover, to complete all rounds of SLIM, new ancilla qubits must be introduced before each round, in addition to the initial set. In experimental setups where an increase in qubit count is negligible, such as idealized scenarios, this approach proves advantageous, as it significantly reduces the number of quantum gates required. However, given the limitations of current quantum computing technology, such a method is impractical due to the cost of qubit resources.

Instead, we employ a more efficient and a novel strategy that avoids excessive qubit usage. By leveraging the Feistel structure of SLIM, we retrieve the original qubits for subsequent rounds through the reverse application of the KSP process ($\text{KSP}^{-1}$), which consists of $\text{P}^{-1}$, $\text{S}^{-1}$, and $\text{K}^{-1} = \text{K}$, applied in reverse order from the lower to upper branches. This approach effectively prepares the system for the second round and allows $U_i$ to be reused as $D_{i-1}$ in each subsequent round. Now, let us realise the quantum implementations of the K, S and P layers and their inverses respectively.

**Key-Addition Layer:** SLIM has 80 qubit keys and runs 32 rounds in total. In this context, for the first 5 rounds, the 80-qubit key is directly divided by five according to the LSQ and added as 16-qubit keys (See Fig. 6). Then, the 80-qubit key goes through a divider to create two 40-qubit keys, labeled MSQ and LSQ. The resulting 40-qubit keys are then treated separately. Accordingly, in each round, the LSQ undergoes a two-qubit circular left shift operation (blue colored) and then the output produced is CNOTed with the MSQ (purple colored). The output of the CNOT process is forwarded to a substitution layer (yellow colored). The result of the S-boxes, which undergoes a three-qubit circular left-shift operation (green colored), is manipulated

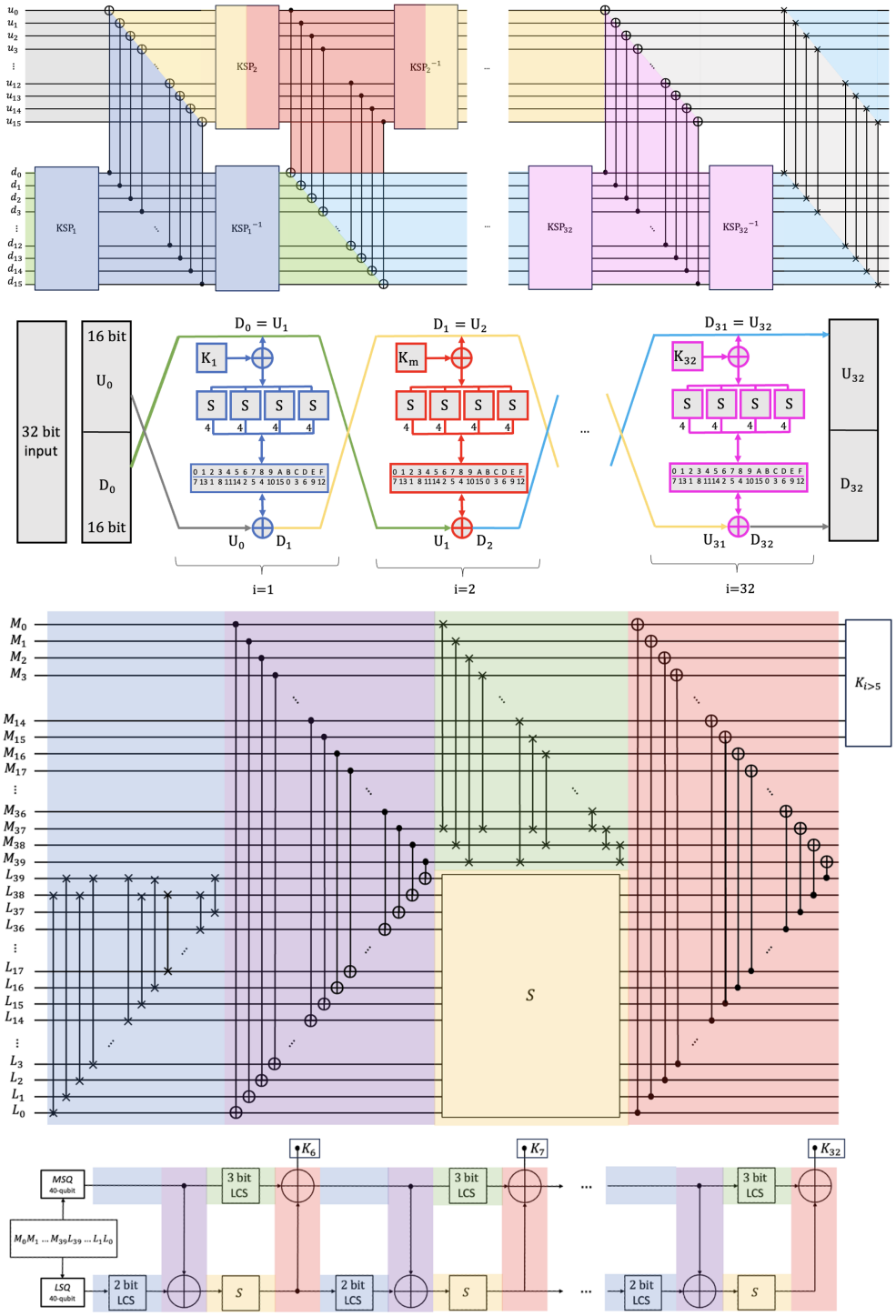



**Fig. 6** Quantum implementation of 32 rounds of SLIM and the key schedule for the rounds $i > 5$

using the CNOT operations to generate the subkey (red colored). The first 16 qubits are then taken and used as the key. This cycle continues until the 32nd round.
The Feistel structure of SLIM inherently ensures that the inverse of the key addition layer, $K^{-1}$, is equivalent to the key addition layer itself, K. This property arises because the operations performed within the K layer of CNOT, substitution, and circular shift are inherently reversible when applied in the same sequence. Consequently, for the decryption process, $K^{-1}$ is directly implemented as K, eliminating the need for additional computational overhead or resources to reverse the key schedule. This characteristic is a fundamental advantage of the Feistel structure, significantly simplifying the implementation of $KSP^{-1}$ during decryption.

**S-box:** The S-box used in SLIM, as given in (3), is identical to the one designed for the PRESENT algorithm [31]. Its quantum implementation has been optimized using the LIGHTER-R framework, a specialized tool for reversible circuit synthesis [16]. LIGHTER-R is particularly well-suited for this task because it eliminates the need for ancillary qubits and minimizes garbage output while efficiently optimizing quantum gates. This tool provides an end-to-end solution for reversible S-box construction, offering significant advantages in terms of gate cost and resource efficiency. As such, the quantum circuits for both the S-box and its inverse, $S^{-1}$-box, are carefully designed in this work to align with these optimizations . To reconstruct the S-box as a quantum circuit with variable output, we use the quantum gates described in the preceding section (NOT, CNOT, and CCNOT gates). The Boolean functions that define the logical operations necessary for constructing the S-box transformations are as follows.

$$\begin{aligned}
x_0 &\rightarrow x_3'x_2'x_0 \oplus x_3'x_1x_0 \oplus x_3x_2'x_0' \oplus x_3x_1x_0' \oplus x_3x_2x_1'x_0 \oplus x_3'x_2x_1'x_0' \\
x_1 &\rightarrow x_3'x_2'x_1 \oplus x_3x_2x_0 \oplus x_3'x_1x_0' \oplus x_3x_2'x_1' \oplus x_3x_2'x_0' \\
x_2 &\rightarrow x_3x_2x_1' \oplus x_2'x_1x_0' \oplus x_3'x_2x_1x_0 \oplus x_3'x_2'x_1' \oplus x_2'x_1'x_0 \\
x_3 &\rightarrow x_3x_2'x_1 \oplus x_3x_2'x_0 \oplus x_3'x_1'x_0' \oplus x_3'x_2x_1 \oplus x_3'x_1x_0
\end{aligned} \tag{3}$$

Here, the notation, e.g., $x_3'x_2'x_0$, indicates [(NOT $x_3$) AND (NOT $x_2$) AND $x_0$]. Using these functions and the LIGHTER-R framework (NCT-gc), the optimized quantum implementation of the SLIM S-box is constructed as shown in Fig. 7.

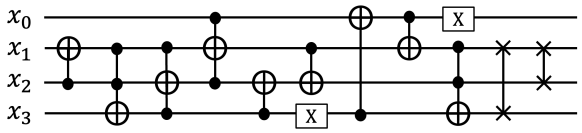



**Fig. 7** Quantum implementation of S-box

Similarly, the Boolean functions for the $S^{-1}$-box are as follows [43]:

$$x_0 = S_3'S_2'S_0' + S_2'S_1'S_0' + S_2S_1'S_0 + S_3'S_2S_0 + S_3S_2S_1S_0' + S_3S_2'S_1S_0$$

$$
\begin{aligned}
x_1 &= S_3'S_2'S_1'S_0 + S_3'S_1S_0' + S_3S_2S_0 + S_3S_1S_0 + S_3S_2'S_0' \\
x_2 &= S_3'S_2'S_1' + S_3'S_1'S_0' + S_3S_1'S_0 + S_2'S_1S_0' + S_3'S_2S_1S_0 \\
x_3 &= S_3'S_2'S_0 + S_3'S_2'S_1 + S_2S_1S_0 + S_3S_2S_1 + S_3S_2'S_1'S_0' + S_3'S_2S_1'S_0'
\end{aligned} \tag{4}
$$

These functions, when implemented using the LIGHTER-R framework, allow for an efficient reversible construction of the $S^{-1}$-box. The resulting circuit (Fig. 8) is optimized to balance quantum gate costs and qubit usage, ensuring compatibility with the overall $KSP^{-1}$ structure.

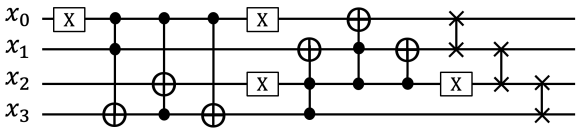



**Fig. 8** Quantum implementation of $S^{-1}$-box

After the key addition layer processes the 16 bits of the input block $U_i$, the S-box operates in parallel for four 4-qubit segments, starting from the LSB. These transformed outputs are then routed through the P-layer, where the qubits are permuted according to a predefined mapping.

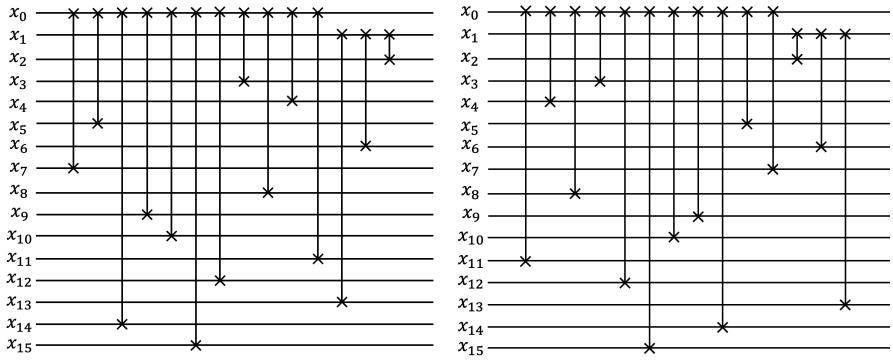



**Fig. 9** Quantum implementations of (a) P-box and (b) $P^{-1}$-box.

**P-box:** The permutation layer given in Fig. 2 can also be implemented in a quantum circuit. For this, it is sufficient to use the SWAP gates with no quantum cost. In this context, the permutation operation (0 7 5 14 9 10 15 12 3 8 4 11)(1 13 6 2) can be directly implemented in a 16-qubit quantum circuit (See Fig. 9-a). The inverse of the P-box, referred to as the $P^{-1}$-box, undoes the permutations performed by the P-box. As shown in Fig. 9-b, it is implemented in a similar manner using SWAP gates with zero quantum cost. The permutation operation for the inverse mapping is (0 11 4 8 3 12 15 10 9 14 5 7)(1 2 6 13), which corresponds directly to the reverse mapping of the original P-box operation.

**KSP:** We combine the key addition, substitution, and permutation layers into a single cohesive unit for constructing the KSP structure, which plays a critical role in the implementation of the SLIM cipher. For one encryption round, the KSP acts on the 16 qubits coming from the D-box ($D_i$) is given in Fig. 10.

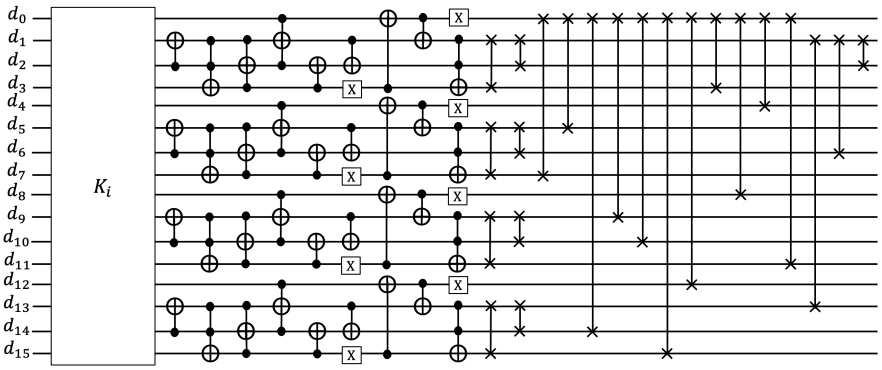



**Fig. 10** Illustration of one round of implementation of KSP

First, the 16 qubits from $D_i$ are CNOTed with the round key $K_i$, derived from the key schedule (Fig. 6). Following this, the transformed qubits are divided into four 4-qubit blocks, each of which is processed in parallel through the quantum S-box circuits. These S-boxes, optimized for quantum implementation, introduce the required non-linear transformations (Fig. 7). The outputs of the four parallel S-box operations are then routed through the P-layer, which performs a predefined permutation of the qubits (Fig. 9). This layer is implemented using quantum SWAP gates, ensuring zero quantum cost for the permutation operation. Upon completing the P-layer, the KSP structure for the current round is finalized, preparing the qubits for interaction with the U-box in the subsequent step.
This modular design of the KSP not only ensures efficiency but also allows for straightforward reversibility, leveraging the Feistel structure of SLIM. A similar process can be constructed for the inverse operation, $\text{KSP}^{-1}$, which includes the inverse permutation layer ($\text{P}^{-1}$), the inverse substitution layer ($\text{S}^{-1}$), and the reverse key addition layer ($\text{K}^{-1}$ =K). These components work together to efficiently return the qubits to their initial state, ensuring seamless integration into the overall quantum implementation of SLIM.

# 4 Quantum resources for SLIM

The total quantum cost of implementing the SLIM algorithm is evaluated by calculating the number of quantum gates required to construct its circuits. Specifically, the cost is determined using standard gate metrics: a NOT gate has a cost of 1 unit, a CNOT gate also costs 1 unit, and a CCNOT gate (Toffoli gate) has a cost of 5 units [44]. The SLIM algorithm operates over 32 rounds in total. As described in Section 3,

the first 5 rounds involve the direct use of 80 qubits in the K layer without requiring additional quantum gate operations for key scheduling. Consequently, these rounds incur no extra gate cost. However, starting from the 6th round, the K layer introduces a significant computational overhead due to the quantum gates required for the key scheduling process. Therefore, to calculate the total quantum resource requirements for SLIM, the first 5 rounds and the subsequent rounds should be considered separately (See Table 1).

According to the quantum algorithm of SLIM (Fig. 6), each round consists of three main steps: the first half of the qubits undergo KSP, followed by CNOT operations with the second half of the qubits, and then the qubits pass through $\text{KSP}^{-1}$. For simplicity, the cost of the CNOT operations between KSP and $\text{KSP}^{-1}$ is included in the cost of KSP. The total cost in the first 5 rounds is 1300, which is sum of the costs of KSP and $\text{KSP}^{-1}$ as given at 4th and 5th rows in Table 1. In the subsequent rounds, each K step includes 20 NOT, 130 CNOT, and 40 CCNOT. However, the cost of KSP and $\text{KSP}^{-1}$ for the remaining 27 rounds is 25920. As a result, the total cost required for the quantum implementation of SLIM is 27220.

**Table 1** Quantum resources requirement for the proposed quantum implementation of SLIM

| LAYER | NOT | CNOT | CCNOT | TOTAL | COST |
|---|---|---|---|---|---|
| S | 2 | 5 | 4 | 11 | 27 |
| $\text{S}^{-1}$ | 4 | 2 | 4 | 10 | 26 |
| $K_{i>5}$ | 20 | 130 | 40 | 5130 | 9450 |
| $KSP_{i\leq 5}$ | 8 | 52 | 16 | 380 | 700 |
| $KSP^{-1}_{i\leq 5}$ | 16 | 24 | 16 | 280 | 600 |
| $KSP_{i>5}$ | 28 | 182 | 56 | 7182 | 13230 |
| $KSP^{-1}_{i>5}$ | 36 | 154 | 56 | 6642 | 12690 |
| SLIM | 1848 | 9452 | 3184 | 14484 | 27220 |

Besides, the depth of a quantum circuit (simply quantum-depth), measures to the number of sequential steps required to execute all operations in the circuit. Unlike classical depth, which refers to the maximum number of gates along a single left-to-right path through the circuit, quantum depth accounts for the layering of quantum gates, where parallelizable gates within a layer are considered as one step. This distinction highlights the unique structure of quantum circuits and their reliance on parallelism to optimize execution. Moreover, certain gates, such as the Toffoli gate, play a critical role in determining quantum depth. While a single Toffoli gate has a T-depth of 1, its full decomposition requires seven layers and four additional ancilla qubits [15]. This metric is crucial as it reflects both the circuit's computational complexity and the time required for execution. Deeper circuits often indicate more intricate computations but may also introduce challenges in maintaining coherence.

In the quantum implementation of SLIM, the depth of the circuit is determined by the different components used in each round. The depth of the S-box is 33, while its inverse, the $\text{S}^{-1}$box, has a depth of 32. For the K layer in rounds $i > 5$, the depth is 35. For the first 5 rounds ($i \leq 5$), the combined depth of KSP and $\text{KSP}^{-1}$ is relatively small, totaling 340. However, in the subsequent 27 rounds ($i > 5$), the depth increases

as KSP and $KSP^{-1}$ together contribute a total depth of 3,726. Adding these values together, the total depth of SLIM across all 32 rounds is 4,066. On the other hand, the absence of ancilla qubits notably increases the quantum cost. If ancilla qubits were utilized, the need for $KSP^{-1}$ operations would be eliminated. In this scenario, the total cost would comprise 13,930 units from the KSP operations, along with an additional 496 units attributed to the CNOT operations required for managing the ancilla qubits. This adjustment would bring the total quantum cost down to 14,426 units.

A detailed analysis of SLIM, particularly in comparison to other block ciphers, requires an evaluation of its total quantum cost, circuit depth, and overall qubit utilization (see Table 2). Within the domain of quantum computing, where efficiency and resource optimization are paramount, SLIM demonstrates a well-balanced approach, achieving notable depth efficiency while maintaining competitive performance and resource demands.

**Table 2** Cost and qubit comparison of quantum implementations with other BCs

| CIPHER | QUBIT | BLK | KEY | NOT | CNOT | CCNOT | TOTAL | DEPTH | COST | REF |
|---|---|---|---|---|---|---|---|---|---|---|
| SIMON | 192 | 64 | 128 | 1216 | 7396 | 1408 | 10020 | 2643 | 15652 | [17] |
| SIMON | 256 | 128 | 128 | 4224 | 17152 | 4352 | 25728 | 8427 | 43136 | [17] |
| RECTANGLE | 144 | 64 | 80 | 567 | 4964 | 2000 | 7531 | 226 | 15531 | [18] |
| LBLOCK | 144 | 64 | 80 | 877 | 16747 | 14280 | 31904 | 1740 | 89024 | [19] |
| LiCi | 192 | 64 | 128 | 379 | 12900 | 8624 | 21903 | 1210 | 56399 | [19] |
| PUFFIN | 192 | 64 | 128 | 620 | 3136 | 3584 | 7340 | 353 | 21676 | [20] |
| PRINT | 128 | 48 | 80 | 154 | 3840 | 2304 | 6298 | 336 | 15514 | [20] |
| SM4 | 260 | 32 | 128 | 8750 | 134912 | 26624 | 170286 | 20480 | 276782 | [21] |
| SLIM | 112 | 32 | 80 | 1848 | 9452 | 3184 | 14484 | 4066 | 27220 | This work |

## 5 Conclusion

In this study, a quantum implementation for SLIM has been proposed for the first time, offering a lightweight BC design optimized for quantum cost. SLIM distinguishes itself as the algorithm with the lowest qubit requirement (112 qubits) compared to other BCs in the literature (see Table 2). This compact qubit utilization, combined with a well-balanced quantum cost of 27,220, positions SLIM as a resource-efficient solution in quantum cryptography. Its unique design eliminates the need for ancilla qubits, simplifying its implementation while maintaining its robustness.

The quantum implementation of SLIM aligns with the broader goal of developing encryption algorithms that are resilient to future quantum threats. By thoroughly analyzing the S, P, and K layers, this study provides the first comprehensive evaluation of SLIM's behavior in the quantum domain. The results reveal a total gate depth of 4,066, highlighting its efficiency compared to other block ciphers, as many alternatives require significantly higher depths for similar levels of security. SLIM's efficient gate distribution is evidenced in Table 2, where critical layers such as KSP and $KSP^{-1}$

are shown to manage their computational demands effectively without relying on additional quantum resources.

What sets SLIM apart is its balance between resource requirements and performance. While other quantum implementations, such as SIMON, RECTANGLE, and LBLOCK, achieve varying degrees of success, they require higher qubit counts or impose larger quantum costs (see Table 1). In contrast, SLIM minimizes its quantum footprint while maintaining competitive cryptographic security. Its low qubit count, optimized quantum cost, and acceptable depth make it a strong candidate for lightweight encryption in the era of quantum computing.

In conclusion, this research highlights SLIM's strengths as a quantum-resistant lightweight block cipher. It marks an important step toward addressing the vulnerabilities of classical cryptographic methods in the face of advancing quantum computing. By achieving a notable balance between qubit efficiency, quantum cost, and circuit depth, SLIM provides a promising framework for secure encryption protocols in a quantum-enabled future. Continued research will further enhance its resilience, paving the way for robust cryptographic solutions to secure sensitive data against the dynamic landscape of quantum threats.

## Data Availability

All data generated or analysed during this study are included in this published article and its supplementary information files.

**Funding.** Not applicable

**Conflict of interest/Competing interests.**

**Ethics approval and consent to participate.**

**Consent for publication.**

## References

[1] Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J Comput **26**(5), 1484–1509 (1997)

[2] Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proc of the 28th Annual ACM Symposium on Theory of Computing (STOC), ACM, (1996)

[3] Yamamura, A., Ishizuka, H.: Quantum cryptanalysis of block ciphers (Algebraic systems, formal languages and computations). RIMS Kokyuroku **2000**(1166) (2000)

[4] Simon, D.: On the power of quantum computation. In: Proceedings of the 35th IEEE Symposium on the Foundations of Computer Science (FOCS), pp. 116–123 (1994)

[5] Dowling, J.P., Gerard, J.M.: Quantum technology: the second quantum revolution. Phil. Trans. R. Soc. A **361**, 1655–1674 (2003)

[6] Polkovnikov, A., Sengupta, K., Silva, A., Vengalattore, M.: Colloquium: Nonequilibrium dynamics of closed interacting quantum systems. Reviews of Modern Physics **83**(3), 863–883 (2011)

[7] Harrow, A.W., Montanaro, A.: Quantum computational supremacy. Nature **549**(7671) (2017)

[8] Daemen, J., Rijmen, V.: Specification for the Advanced Encryption Standard (AES). FIPS 197 (2001)

[9] Li, Z., Cai, B., Sun, H., Liu, H., Wan, L., Qin, S., Wen, Q., Gao, F.: Novel quantum circuit implementation of advanced encryption standard with low costs. Science China Physics, Mechanics and Astronomy **65**(9), 290–311 (2022)

[10] Huang, Z., Sun, S.: Synthesizing Quantum Circuits of AES with Lower T-depth and Less Qubits. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology – ASIACRYPT 2022, pp. 614–644. Springer, Cham (2022). Springer

[11] Almazrooie, M., Samsudin, A., Abdullah, R., Mutter, K.N.: Quantum reversible circuit of AES-128. Quantum information processing **17**, 1–30 (2018)

[12] Langenberg, B., Pham, H., Steinwandt, R.: Reducing the cost of implementing the advanced encryption standard as a quantum circuit. IEEE Transactions on Quantum Engineering **1**, 1–12 (2020)

[13] Luo, Q.-b., Yang, G.-w., Li, X.-y., Li, Q.: Quantum reversible circuits for multiplicative inverse. EPJ Quantum Technology **9**(1), 24 (2022)

[14] Langenberg, B., Pham, H., Steinwandt, R.: Reducing the cost of implementing the advanced encryption standard as a quantum circuit. IEEE Transactions on Quantum Engineering **1**, 1–12 (2020)

[15] Selinger, P.: Quantum circuits of T-depth one. Physical Review A—Atomic, Molecular, and Optical Physics **87**(4), 042302 (2013)

[16] 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), pp. 260–265 (2019). IEEE

[17] Anand, R., Maitra, A., Mukhopadhyay, S.: Grover on SIMON. Quantum Inf. Process. **19**, 1–17 (2020)

[18] Saravanan, P., Jenitha, J., Aasish, S., Sanjana, S.: Quantum circuit design of RECTANGLE lightweight cipher. In: 2021 25th International Symposium on VLSI Design and Test (VDAT), pp. 1–4 (2021). IEEE

[19] Jing, X., Li, Y., Zhao, G., Xie, H., Wang, Q.: Quantum circuit implementation and resource analysis of LBlock and LiCi. Quantum Information Processing **22**(9), 347 (2023)

[20] Paramasivam, S., Jenitha, J., Sanjana, S., Haghparast, M.: Compact quantum circuit design of PUFFIN and PRINT lightweight ciphers for quantum key recovery attack. IEEE Access **11**, 66767–66776 (2023)

[21] Luo, Q.B., Li, Q., Li, X.Y., Yang, G.W., Shen, J., Zheng, M.: Quantum circuit implementations of SM4 block cipher optimizing the number of qubits. Quantum Information Processing **23**(5), 177 (2024)

[22] Chun, M., Baksi, A., Chattopadhyay, A.: DORCIS: Depth optimized quantum implementation of substitution boxes. Cryptology ePrint Archive (2023)

[23] Lee, W.-K., Jang, K., Song, G., Kim, H., Hwang, S.O., Seo, H.: Efficient implementation of lightweight hash functions on GPU and quantum computers for IoT applications. IEEE Access **10**, 59661–59674 (2022)

[24] Feistel, H.: Cryptography and Computer Privacy. Scientific American **228**(5), 15–23 (1973)

[25] FIPS: Data Encryption Standard. NBS **46-3** (1999)

[26] FIPS: Advanced Encryption Standard. NBS **197** (2001)

[27] Jaques, S., Naehrig, M., Roetteler, M., Virdia, F.: Implementing Grover oracles for quantum key search on AES and LowMC. In: Advances in Cryptology–EUROCRYPT 2020: 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10–14, 2020, Proceedings, Part II 30, pp. 280–310 (2020). Springer

[28] Schlieper, L.: In-place implementation of Quantum-Gimli. arXiv 2020. arXiv:2007.06319

[29] Pal, O., Jain, M., Murthy, B., Thakur, V.: Quantum and Post-Quantum Cryptography, pp. 45–58. Wiley Online Library (2022)

[30] Chauhan, A.K., Sanadhya, S.K.: Quantum resource estimates of Grover's key search on ARIA. In: Proc. Int. Conf. Secur. Privacy Appl. Cryptogr. Eng., pp. 238–258 (2020)

[31] Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J., Seurin, Y., Vikkelsoe, C.: PRESENT: An ultra-lightweight block cipher. In: Cryptographic Hardware and Embedded Systems-CHES 2007: 9th International Workshop, Vienna, Austria, September 10-13, 2007. Proceedings 9, pp. 450–466 (2007). Springer

[32] Hong, D., Sung, J., Hong, S., Lim, J., Lee, S., Koo, B.-S., Lee, C., Chang, D., Lee, J., Jeong, K., Kim, H., Kim, J., Seongtaek, C.: HIGHT: A new block cipher suitable for low-resource device. In: Cryptographic Hardware and Embedded Systems-CHES 2006: 8th International Workshop, Yokohama, Japan, October 10-13, 2006. Proceedings 8, pp. 46–59 (2006). Springer

[33] Knudsen, L.R., Leander, G., et.al.: PRINT-cipher: A block cipher for ic-printing. In: Proc. 12th Int. Workshop in Lecture Notes in Computer Science, vol. 6225. USA (2010)

[34] Guo, J., Peyrin, T., et.al: The PHOTON family of lightweight hash functions. In: Rogaway, P. (ed.) Proc. 31st Annu. Int. Cryptol. Conf. (CRYPTO), in Lecture Notes in Computer Science, vol. 6841, pp. 222–239. Springer, Santa Barbara, CA, USA (2011)

[35] Beaulieu, R., Shors, D., et.al: The SIMON and SPECK lightweight block ciphers. In: Proc. 52nd Annu. Design Autom. Conf., p. 175 (2015)

[36] Zhang, W., Bao, Z., et.al.: RECTANGLE: A bit-slice ultra-lightweight block cipher suitable for multiple platforms. In: Proc. IACR, p. 84 (2014)

[37] Cheng, H., Heys, H.M., et.al: PUFFIN: A novel compact block cipher targeted to embedded digital systems. In: Proc. 11th EUROMICRO Conf. Digit. Syst. Des. Archit. Methods Tools, pp. 383–390 (2008)

[38] Wu, W., Zhang, L.: LBlock: a lightweight block cipher. In: App. Cryp. Net. Sec. 9th Int. Conf. ACNS 2011, Nerja, Spain, pp. 327–344 (2011)

[39] Patil, J., Bansod, G., Kant, K.S.: LiCi: A new ultra-lightweight block cipher. In: Int. Conf. on Emerging Trends and Innov ICT, pp. 40–45 (2017)

[40] Aboushosha, B., Ramadan, R.A., Dwivedi, A.D., El-Sayed, A., Dessouky, M.M.: SLIM: A lightweight block cipher for internet of health things. IEEE Access **8**, 203747–203757 (2020)

[41] Yadav, T., Kumar, M., Kumar, A., Pal, S.K.: A practical-quantum differential attack on block ciphers. Cryptography and Communications, 1–21 (2023)

[42] NIST: Project for Post-Quantum Cryptography Standardization (2016)

[43] Mohanapriya, R., Kumar, N.: Optimized Implementation of S-box and Inverse S-box for PRESENT Lightweight Block Cipher. In: 2023 2nd International Conference on Vision Towards Emerging Trends in Communication and Networking Technologies (ViTECoN), pp. 1–5 (2023). IEEE

[44] Shende, V.V., Markov, I.L.: On the CNOT-cost of TOFFOLI gates. Quantum Information and Computation **9**(5), 461–486 (2009)