Differential Cryptanalysis on Quantum Computers

Kyungbae Jang, Yujin Oh, and Hwajeong Seo

Division of IT Convergence Engineering, Hansung University, Seoul, South Korea
starj1023@gmail.com, oyj0922@gmail.com, hwajeong84@gmail.com

Abstract. As quantum computing progresses, extensive research has been conducted to find quantum advantages in the field of cryptography. Combining quantum algorithms with classical cryptographic analysis methods, such as differential cryptanalysis and linear cryptanalysis, has the potential to reduce complexity.
In this paper, we present a quantum differential finding circuit for differential cryptanalysis. In our quantum circuit, both plaintext and input difference are in a superposition state. Actually, while our method cannot achieve a direct speedup with quantum computing, it offers a different perspective by relying on quantum probability in a superposition state.
For the quantum simulation, given the limited number of qubits, we simulate our quantum circuit by implementing the Toy-ASCON quantum circuit.

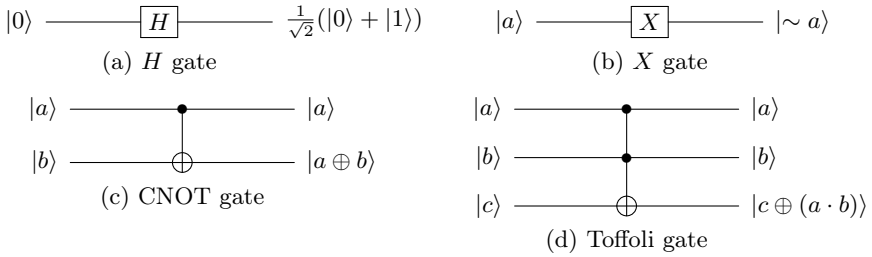
Keywords: Differential Cryptanalysis · Quantum Computer · Toy-ASCON Quantum Circuit.

1 Introduction

Large-scale quantum computers are poised to revolutionize various scientific fields, particularly cryptography. Shor's algorithm [12] makes encryption methods like RSA and Elliptic Curve Cryptography (ECC) ineffective in a post-quantum era. While widespread access to quantum computers may not occur soon, the cryptographic community is already addressing this potential threat. One major concern is the vulnerability of current long-term secrets, which could be compromised if encrypted data is stored until quantum computing becomes feasible. This emphasizes the need for post-quantum cryptosystems, such as those based on robust mathematical problems (against quantum computers) like lattice and code-based cryptography.

Symmetric key cryptography also faces a reduction in security due to quantum computers. Grover's algorithm [5] reduces the complexity of search problems and can be employed to find the secret key of ciphers. Furthermore, combining this quantum algorithm with cryptanalysis methods such as differential cryptanalysis and linear cryptanalysis can achieve a quantum speed-up.

In this paper, we focus on exploring quantum advantages for differential cryptanalysis. Specifically, a method for constructing a differential table using quantum circuits is presented. Unfortunately, this method cannot provide a



(a) H gate (b) X gate (c) CNOT gate (d) Toffoli gate

Fig. 1: Quantum gates

quantum speed-up related to time complexity. However, we approach the problem differently by leveraging the fact that a good differential exhibits the highest amplification in a quantum search.

2 Background

2.1 Quantum Gates

This section explains the fundamental quantum gates used to implement quantum circuits in this work. The H (Hadamard) gate operates on a single qubit and is used to prepare a qubit in a superposition state. The X (NOT) gate also operates on a single qubit and inverts the state of the qubit; $X(a) \rightarrow \sim a$. The CNOT (Controlled-NOT) gate operates on two qubits: one control and one target qubit. It inverts the state of the target qubit only if the state of the control qubit is $|1\rangle$. This quantum gate can replace the classical XOR operation; CNOT $(a, b) \rightarrow (a, a \oplus b)$. The Toffoli gate, also referred to as the CCNOT gate, operates on three qubits: two control qubits and one target qubit. It inverts the target qubit's state only if the two control qubits are in the state $|1\rangle$. This quantum gate can replace the classical AND operation; Toffoli $(a, b, c) \rightarrow (a, b, c \oplus (a \cdot b))$.

The Toffoli gate is the most resource-critical quantum gate introduced so far, as it is implemented as a combination of Clifford gates (H, CNOT, X, etc.) and T gates. There are various methods for implementing Toffoli gates using Clifford and T gates [1,11,6]. This paper adopts one of the methods from [1], which is implemented with 8 Clifford and 7 T gates, and has a T-depth of 4 and a full depth of 8.

2.2 The Grover algorithm

The Grover algorithm measures a target data for the n-qubit search space (in a superposition state) with reduced complexity compared to classical exhaustive search; classical: $O(2^n) \rightarrow$ Grover: $O(\sqrt{2^n})$. The process of Grover's algorithm can be divided into three steps: *Setting input*, *Oracle*, and *Diffusion operator*, as follows.

Setting input Firstly, an n-qubit in a superposition state ($|\psi\rangle$) is prepared using n Hadamard gates; Equation 1. Note that this n-qubit represents the entire space of 2^n cases, and this preparation step can be modified for specific search targets.

$$H^{\otimes n}|0\rangle^{\otimes n} = |\psi\rangle = \left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} |x\rangle \tag{1}$$

Oracle In the *oracle*, a quantum circuit capable of finding a target data in the search space (from the previous step) should be implemented. The solution is satisfied only for the target data and the target data is marked by negating the sign of the state; Equations 2 and 3.

$$f(x) = \begin{cases} 1 \text{ if Circuit}(x) = \text{target output} \\ 0 \text{ if Circuit}(x) \neq \text{target output} \end{cases} \tag{2}$$

$$U_f(|\psi\rangle|-\rangle) = \frac{1}{2^{n/2}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle |-\rangle \tag{3}$$

Diffusion operator The *diffusion operator* amplifies the amplitude of the target data marked by the oracle. The diffusion operator is implemented using a fixed method and is often ignored in the resource estimation for Grover's search as it has negligible cost compared to the oracle [8,2,7,9,4].

Finally, Grover's search iterates a set of the oracle and the diffusion operator approximately $\sqrt{2^n}$ times and measures the target data with high probability.

2.3 Differential Cryptanalysis

Differential cryptanalysis is one of the major methods used in cryptography. When designing ciphers, ensuring non-distinguishability is essential to guarantee robustness against differential attacks. Let Δ_{in} be an input difference of the cipher encryption, and let Δ_{out} represent an output difference such that:

$$\Delta_{out} = Enc(x \oplus \Delta_{in}) \oplus Enc(x). \tag{4}$$

$Prob(\Delta_{in}, \Delta_{out})$ represents the probability of a pair of Δ_{in} and Δ_{out} appearing for the cipher Enc with random plaintext x. For an ideal cipher, assuming an n-bit key is chosen randomly, the probability that a specific input-output pair appears is always $1/2^n$ (i.e., $Prob(\Delta_{in}, \Delta_{out}) = 1/2^n$). However, when the probability is higher than $1/2^n$, the specific input-output pair can be adopted as a differential. That is, if $Prob(\Delta_{in}, \Delta_{out})$ is high, a pair of Δ_{in} and Δ_{out} is denoted as a good differential.

With a good differential that satisfies $Prob(\Delta_{in}, \Delta_{out}) = 1/2^k$, an attacker queries $Enc(x \oplus \Delta_{in}) \oplus Enc(x)$ $/2^{k+1}$ times for a random behavior. The attacker obtains Δ_{out} with high probability, and the attacker can guess that *this random behavior is a cipher encryption*. Subsequently, the attacker can attempt a *last-round attack*.

Using the differential $(\Delta_{in}, \Delta_{out})$ for r-round encryption, an attacker can attempt to recover the round key RK for the last $r+1$-round. The attacker requests encryption queries for a sufficient number of randomly chosen plaintext pairs satisfying the input difference Δ_{in}. This allows them to obtain the corresponding ciphertexts for the last $r+1$-round and then proceed to decrypt the last round by iteratively changing the round key. If the difference of the obtained 1-round decryption results matches the output difference such that:

$$Dec^1(Enc^{r+1}(x \oplus \Delta_{in}), RK) \oplus Dec^1(Enc^{r+1}(x), RK) = \Delta_{out} \tag{5}$$

the attacker counts the corresponding round key RK. The attacker performs 1-round decryption for all ciphertexts, counting the round keys that satisfy the output differential. Finally, the attacker recovers the candidate round key with the highest count.

3 Quantum Circuit for Finding Differential

In this section, a quantum circuit for finding differentials in a superposition state is presented. A similar approach is also proposed in [14], but they finds a differential in a half-superposition state (only the plaintext is in a superposition state, while the input difference is fixed). The quantum circuit presented in this work aims to find a high probability differential in a full-superposition state, where both the input difference and plaintext are in a superposition state.

3.1 Quantum Circuit of Toy-ASCON

To simulate our quantum circuit for finding differentials, we implement a toy version of the ASCON quantum circuit, called toy-ASCON. We utilize the quantum S-box implementation presented in [10] for implementing our toy-ASCON.

S-Box We implement the ASCON S-Box of Equation 6 using quantum gates, and then use it for the Toy-ASCON quantum circuit. Figure 2 shows the quantum circuit of the ASCON S-Box. This quantum circuit allocate 5 ancilla qubits $t_{0\sim5}$ and the output is stored in the input qubits $x_{0\sim5}$.

$$\begin{aligned}
&x_0 = x_0 \oplus x_4, \quad x_4 = x_4 \oplus x_3, \quad x_2 = x_2 \oplus x_1,\\
&t_0 = x_0, \quad t_1 = x_1, \quad t_2 = x_2, \quad t_3 = x_3, \quad t_4 = x_4,\\
&t_0 = \sim t_0, \quad t_1 = \sim t_1, \quad t_2 = \sim t_2, \quad t_3 = \sim t_3, \quad t_4 = \sim t_4,\\
&t_0 = t_0 \cdot x_1, \; t_1 = t_1 \cdot x_2, \; t_2 = t_2 \cdot x_3, \; t_3 = t_3 \cdot x_4, \; t_4 = t_4 \cdot x_0,\\
&x_0 = x_0 \oplus t_1, \quad x_1 = x_1 \oplus t_2, \quad x_2 = x_2 \oplus t_3, \quad x_3 = x_3 \oplus t_4, \quad x_4 = x_4 \oplus t_0,\\
&x_1 = x_1 \oplus x_0, \quad x_0 = x_0 \oplus x_4, \quad x_3 = x_3 \oplus x_2, \quad x_2 = \sim x_2.
\end{aligned} \tag{6}$$

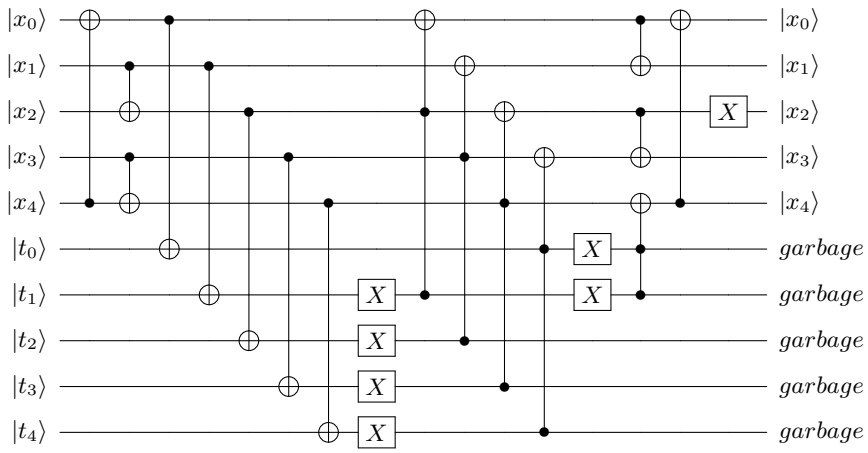


Fig. 2: Quantum circuit of the ASCON S-Box.

Linear Diffusion Layer We assume the linear layer $\Sigma(x)$ operates on 5 qubits and is given by:

$$x \leftarrow \Sigma(x) = x \oplus (x \ggg 1) \oplus (x \ggg 3). \tag{7}$$

The linear layer of Equation 7 can be represented in matrix form, and we can obtain three matrices—permutation; upper, and lower matrices; by using PLU factorization, as shown below.

$$\begin{bmatrix} 1&1&0&1&0 \\ 0&1&1&0&1 \\ 1&0&1&1&0 \\ 0&1&0&1&1 \\ 1&0&1&0&1 \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&0&0&1 \\ 0&0&1&0&0 \\ 0&0&0&1&0 \end{bmatrix} \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&1&1&0&0 \\ 1&1&0&1&0 \\ 1&1&0&0&1 \end{bmatrix} \begin{bmatrix} 1&1&0&1&0 \\ 0&1&1&0&1 \\ 0&0&1&1&0 \\ 0&0&0&1&0 \\ 0&0&0&0&1 \end{bmatrix} \tag{8}$$

Using these matrices, we can implement an in-place quantum circuit for the linear layer $\Sigma(x)$. This implementation method is often adopted in quantum implementations [8,4,3,15,13] to reduce the number of qubits. For details on the in-place implementation of the linear layer using PLU factorization, we recommend readers refer to [15,13]. Figure 3 shows the quantum circuit of the linear layer $\Sigma(x)$ using PLU factorization. Note that quantum swap gates are included in Figure 3 for an intuitive understanding of PLU factorization (strictly speaking, for the permutation matrix). However, in our implementation, this swap operation is achieved by a logical swap that changes the index of qubits without using quantum gates. Therefore, these quantum swap gates are also excluded from the resource estimation.

Constant Addition The constant addition, which involves XOR-ing the constant c with the intermediate value x, is implemented using only X gates. We

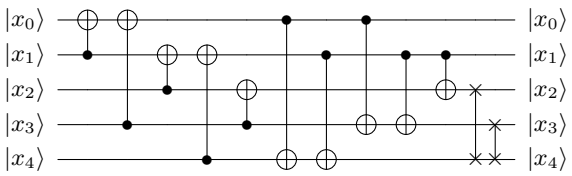


Fig. 3: Quantum circuit (in-place) of the linear layer $\Sigma(x)$.

apply X gates to the bits of x where the corresponding bits of the constant c are 1. We assume the constant is 5 (0b00101).

4 Constructing a Differential Table

Finally, the quantum circuit of Toy-ASCON consists of three steps; S-Box, linear layer, and constant addition. The quantum resources required for the Toy-ASCON quantum circuit are shown in Table 1.

Table 1: Quantum resources required for the Toy-ASCON quantum circuit.

Cipher	#CNOT	#1qCliff	#T	T-depth	#Qubits	Full depth
Toy-ASCON	51	25	35	12	15	38

For the simulation in the quantum state, fewer than 30 qubits are available. Due to this limitation, our Toy-ASCON encrypts 5-bit plaintext, and we operate two Toy-ASCON quantum circuits to encrypt two plaintexts with the input difference, i.e., $Enc(x)$ and $Enc(x \oplus \delta_{in})$. Figure 4 illustrates the quantum circuit for finding the differential (based on the probability) of Toy-ASCON encryption and Table 2 shows the resources required for the quantum differential finding circuit of Figure 4.

Table 2: Quantum resources required for the quantum differential finding circuit.

Cipher	#CNOT	#1qCliff	#Measure	#T	T-depth	#Qubits	Full depth
Toy-ASCON	117	51	10	70	12	25	42

For the preparation step, we prepare the plaintext and input difference in a superposition state (i.e., $\psi(x)$ and $\psi(\delta)$). After that, we copy the plaintext into clean qubits and plus (XOR) the input difference using CNOT gates.

Two Toy-ASCON quantum circuits are operated to generate $\delta_{out} = Enc(x \oplus \delta_{in}) \oplus Enc(x)$. Finally, we measure the input difference and output difference, δ_{in} and δ_{out}.

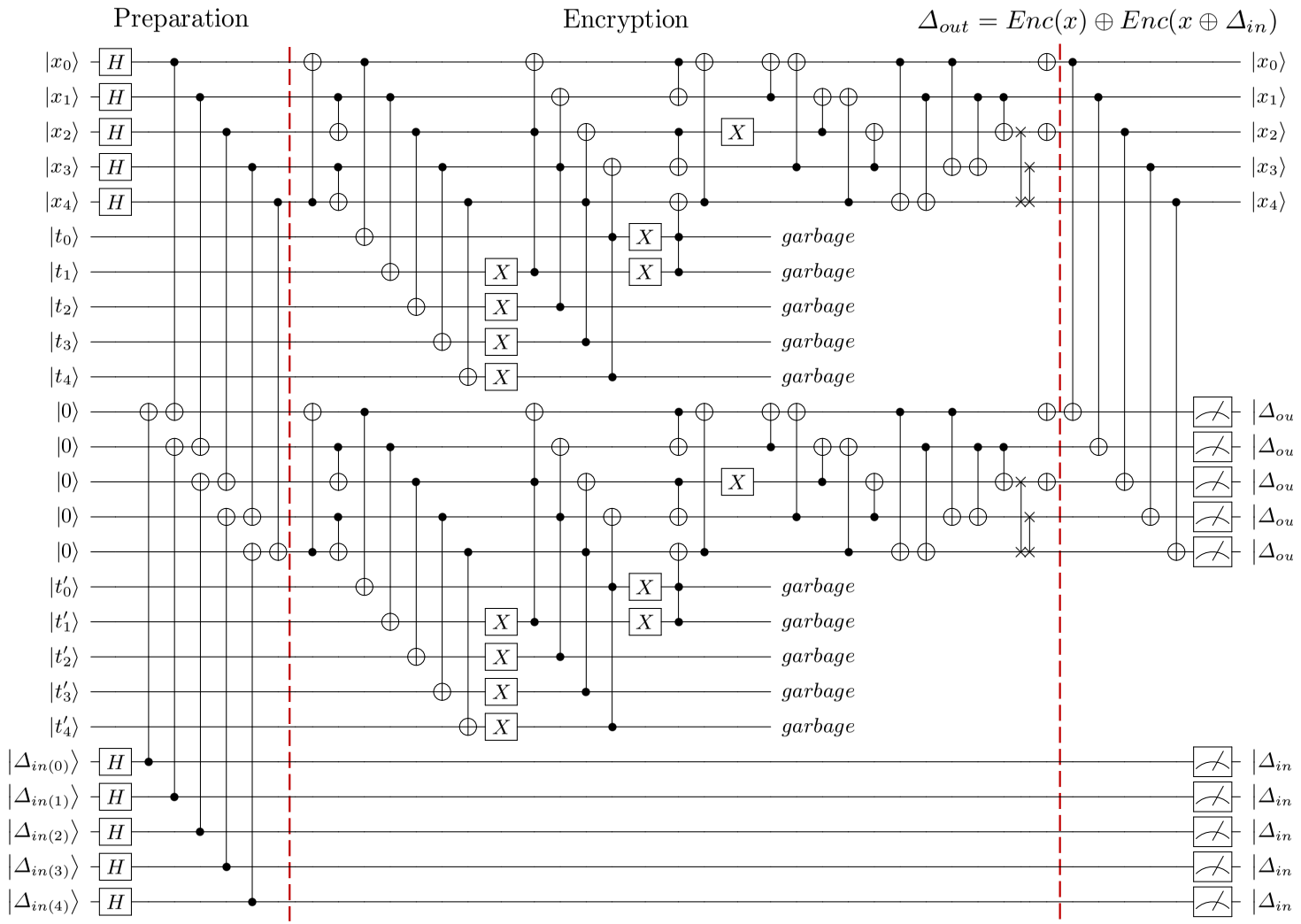


Fig. 4: Caption

The matrix in Equation 9 displays the counting result of operating the quantum differential finding circuit 500 times. Columns and rows represent input and output differences, respectively. In the matrix, the value at the top leftmost position is 0b00000, and the value at the bottom rightmost position is 0b11111. From this matrix, we can infer that the highlighted in red input-output difference pairs are potential candidates for good differentials. Conversely, input-output differences with a value of zero can be considered as impossible differentials.

$$
\begin{pmatrix}
14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & \mathbf{5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 2 & 0 & 1 & 0 & 1 & 0 & 2 & 0 & 0 & 1 & 0 & 3 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{5} & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 3 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 3 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathbf{7} & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 4 & 0 & 0 & 1 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 1 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 2 \\
0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\
0 & 2 & 0 & 1 & 2 & 0 & 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 2 & 0 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 0 & 2 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 1 & 3 & 4 \\
0 & 0 & 2 & 0 & 4 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 2 & 0 & 1 & 1 & 0 & 4 & 0 & 0 & 4 & 0 & 2 & 2 & 0 & 0 & 0 & 0 & 2 \\
0 & 0 & 0 & 1 & 1 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 3 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 4 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 3 & 4 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathbf{5} & 0 & 0 & 2 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 1 & 0 \\
0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 1 & 2 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 3 & 1 & 1 & 3 & 2 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 2 \\
0 & 2 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{5} & 0 & 0 & 0 \\
0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 3 & 0 & 0 & 2 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 0 & \mathbf{5} & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{7} & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 1 & 2 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\
0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 1 & 1 & 2 & 2 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 3 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 2 & 0 & 0 & 2 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 3 & 0 \\
0 & 0 & 1 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 & 4 & 3 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\tag{9}
$$

5 Discussion

In this section, we discuss our quantum circuit for finding differentials in a quantum state. Unfortunately, our quantum differential finding circuit includes an

input difference of zero (which should be excluded), so we need to manually exclude the results for the input difference of zero. The measurement result includes the probability of the differential, $Prob(\delta_{in}, \delta_{out})$. In simpler terms, before measurement, the qubits for δ_{in} and δ_{out} possess all possible cases that can occur in Toy-ASCON encryption. Therefore, we can consider the measurement result as having the probability of a good differential (which has a high probability of appearing in Toy-ASCON encryption). Actually, while this approach does not provide quantum speedup, it offers a different perspective on constructing a differential table.

Additionally, our quantum circuit can be used as a differential distinguisher[1] in the quantum security model, where quantum adversaries can make quantum queries for random behavior (often referred to as the Q2 model).

6 Conclusion

In this work, we focused on constructing a quantum differential finding circuit where the measurement result is based on the probability of a good differential. Due to the simulation limit on the number of qubits, we constructed the Toy-ASCON quantum circuit by implementing an ASCON S-Box and an in-place linear layer. The used implementation methods are generic and can be utilized (or have already been utilized) for quantum implementations.

We explored a new approach for differential cryptanalysis using quantum computers. Although this approach does not provide quantum speedup, it offers a different view for finding differentials based on quantum probability. For future work, we will further develop our approach and provide a concrete analysis in terms of complexity and efficiency.

7 Acknowledgement

This work was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2022R1A6A3A13062701, 100%).

References

1. Amy, M., Maslov, D., Mosca, M., Roetteler, M., Roetteler, M.: A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **32**(6), 818–830 (Jun 2013). https://doi.org/10.1109/tcad.2013.2244643, http://dx.doi.org/10.1109/TCAD.2013.2244643 2

[1] In contrast to the quantum differential finding in our work, this approach provides quantum speedup using Grover's algorithm

2. Baksi, A., Jang, K., Song, G., Seo, H., Xiang, Z.: Quantum implementation and resource estimates for rectangle and knot. Quantum Information Processing **20**(12) (dec 2021). https://doi.org/10.1007/s11128-021-03307-6, https://doi.org/10.1007/s11128-021-03307-6 3
3. Cheung, D., Maslov, D., Mathew, J., Pradhan, D.K.: On the design and optimization of a quantum polynomial-time attack on elliptic curve cryptography. In: Theory of Quantum Computation, Communication, and Cryptography: Third Workshop, TQC 2008 Tokyo, Japan, January 30-February 1, 2008. Revised Selected Papers 3. pp. 96–104. Springer (2008) 5
4. Grassl, M., Langenberg, B., Roetteler, M., Steinwandt, R.: Applying Grover's algorithm to AES: Quantum resource estimates. In: Takagi, T. (ed.) Post-Quantum Cryptography. pp. 29–43. Springer International Publishing, Cham (2016) 3, 5
5. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 212–219 (1996) 1
6. He, Y., Luo, M.X., Zhang, E., Wang, H.K., Wang, X.F.: Decompositions of n-qubit Toffoli gates with linear circuit complexity. International Journal of Theoretical Physics **56**(7), 2350–2361 (2017) 2
7. Jang, K., Baksi, A., Kim, H., Song, G., Seo, H., Chattopadhyay, A.: Quantum analysis of AES. Cryptology ePrint Archive, Paper 2022/683 (2022), https://eprint.iacr.org/2022/683, https://eprint.iacr.org/2022/683 3
8. Jaques, S., Naehrig, M., Roetteler, M., Virdia, F.: Implementing Grover Oracles for quantum key search on AES and LowMC. In: Canteaut, A., Ishai, Y. (eds.) 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. Lecture Notes in Computer Science, vol. 12106, pp. 280–310. Springer (2020). https://doi.org/10.1007/978-3-030-45724-2_10, https://doi.org/10.1007/978-3-030-45724-2_10 3, 5
9. 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 (01 2020). https://doi.org/10.1109/TQE.2020.2965697 3
10. Oh, Y., Jang, K., Baksi, A., Seo, H.: Depth-optimized implementation of ascon quantum circuit. Cryptology ePrint Archive (2023) 4
11. Selinger, P.: Quantum circuits of T-depth one. Physical Review A **87**(4), 042302 (2013) 2
12. Shor, P.W.: Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th annual symposium on foundations of computer science. pp. 124–134. IEEE (1994) 1
13. Van Hoof, I.: Space-efficient quantum multiplication of polynomials for binary finite fields with sub-quadratic Toffoli gate count. arXiv preprint arXiv:1910.02849 (2019) 5
14. Yadav, T., Kumar, M., Kumar, A., Pal, S.K.: A practical-quantum differential attack on block ciphers. Cryptography and Communications pp. 1–21 (2023) 4
15. Yang, Y., Jang, K., Baksi, A., Seo, H.: Optimized implementation and analysis of cham in quantum computing. Applied Sciences **13**(8), 5156 (2023) 5