Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.037336
Article

Temperature-Triggered Hardware Trojan Based Algebraic Fault Analysis of SKINNY-64-64 Lightweight Block Cipher

Lei Zhu, Jinyue Gong, Liang Dong* and Cong Zhang

Communication and Electronics Engineering Institute, Qiqihar University, Qiqihar, 161006, China
*Corresponding Author: Liang Dong. Email: dongliang@163.com
Received: 31 October 2022; Accepted: 03 March 2023

Abstract: SKINNY-64-64 is a lightweight block cipher with a 64-bit block length and key length, and it is mainly used on the Internet of Things (IoT). Currently, faults can be injected into cryptographic devices by attackers in a variety of ways, but it is still difficult to achieve a precisely located fault attacks at a low cost, whereas a Hardware Trojan (HT) can realize this. Temperature, as a physical quantity incidental to the operation of a cryptographic device, is easily overlooked. In this paper, a temperature-triggered HT (THT) is designed, which, when activated, causes a specific bit of the intermediate state of the SKINNY-64-64 to be flipped. Further, in this paper, a THT-based algebraic fault analysis (THT-AFA) method is proposed. To demonstrate the effectiveness of the method, experiments on algebraic fault analysis (AFA) and THT-AFA have been carried out on SKINNY-64-64. In the THT-AFA for SKINNY-64-64, it is only required to activate the THT 3 times to obtain the master key with a 100% success rate, and the average time for the attack is 64.57 s. However, when performing AFA on this cipher, we provide a relationship between the number of different faults and the residual entropy of the key. In comparison, our proposed THT-AFA method has better performance in terms of attack efficiency. To the best of our knowledge, this is the first HT attack on SKINNY-64-64.

Keywords: SKINNY-64-64; lightweight block cipher; algebraic fault analysis; Hardware Trojan; residual entropy

1 Introduction

The research on lightweight block ciphers (e.g., LED, Piccolo, PRESENT, Midori, HIGHT, SIMON, SKINNY [1], etc.) provides a solution to the problem of security for resource-constrained IoT devices. However, whether the security of lightweight block ciphers themselves meets the requirements has become a major research topic in the field of cryptography.

Fault attacks are widely used in security studies of block ciphers as one of the practical threats of modern cryptographic implementations. However, existing fault attack methods (changing device voltage [2–4], laser irradiation [5,6], X-ray irradiation [7], electromagnetic and clock scrambling [8,9],

etc.) make it difficult to perform precise fault injection into running hardware devices in a low-cost manner. As the application of lightweight block ciphers continues to increase in proportion to hardware implementations, the security of cryptographic circuits is under increasing threat from HTs. A HT is a malicious circuit that is maliciously inserted into the integrated circuit (IC) and has destructive capabilities (e.g., changing the functionality of the original circuit, suspending services, etc.), which can be implemented by directly modifying the field-programmable gate array (FPGA) configuration bitstream [10] or by modifying the original design of the circuit at the register transfer level (RTL) [11], etc. HTs are extremely capable of attacking not only certain embedded devices [12,13] and neural networks [14,15], but also disrupting network-on-chip (NoC) communication [16], among others.

With the global production of hardware devices and the influence of the cost budgets and time-to-market of products, third-party intellectual property (IP) cores are widely used, which makes the possibility of implanting HTs in certain cryptographic IP cores much more likely [17,18]. Through investigation and research, it was found that temperature can be used as a trigger condition for HT attacks. Usually, HTs mainly consist of a trigger and payload. Hutter et al. [19] used a manually adjustable heating plate to heat the microcontroller ATmega162 at a high temperature to achieve the acquisition of exploitable faults for the RSA encryption algorithm. In [20], Kumar et al. were able to achieve accurate fault injection with a success rate of approximately 0.1% by varying the operating temperature of the hardware device. In addition, Dash et al. [21] changed the internal temperature of a computer by increasing the utilization of the central processing unit (CPU) of the computer, thus enabling the activation of a HT. However, this additional CPU activity can easily trigger CPU workload monitor alerts. In [22], Ghandali et al. imposed more than the maximum operating temperature on the hardware device to achieve a certain path of the extended combinational circuit, thus maliciously manipulating the true random number generator (TRNG) design. Clearly, it is feasible to use a reasonable temperature as a trigger condition when performing actual HT attacks.

Motivation. In order to enable precise fault injection in a low-cost manner, we design a HT based on temperature triggering. The HT can be activated using only a heat gun or a household hair dryer. From the attacker's point of view, using temperature as the trigger condition for the HT is ideal. First, the attacker can activate the HT without physically destroying the hardware device. Second, a temperature-based HT will be smaller and quieter than the combinatorial and timing circuits traditionally used to trigger HTs, and it has a lower hardware overhead area and power consumption. To better analyze the security of SKINNY[1], we combine the designed HT with AFA [23] and propose the THT-AFA. It has the features of being non-intrusive, low-cost, and with high fault injection accuracy. Compared with the traditional AFA, this analysis method significantly improves the efficiency of SKINNY attacks.

Contribution. We design a THT based on the 28 nm ZYNQ-7010 FPGA [24] development board and combine it with AFA. The main contributions of this paper are as follows.

(1) For SKINNY, some properties are given that arise during the propagation of the fault, which can be used not only as a precondition to predetermine the inserted position of the THT, but also to discriminate the position of the injected random nibble fault.
(2) An evaluation method of the residual entropy of the key after different numbers of fault injections is proposed, which evaluates the security of SKINNY against fault attacks under different fault models. Under the random nibble fault injection model, we perform an AFA on

[1]For convenience, SKINNY-64-64 is represented by SKINNY for the rest of the paper.

SKINNY and give the equation for the relationship between the residual entropy of the key and the number of faults.

(3) A low-cost THT was designed, which was implemented on a 28 nm ZYNQ-7010 FPGA development board. We first implemented the SKINNY circuit (benchmark circuit) based on the FPGA development board and then inserted the designed THT into the circuit. The SKINNY circuit with THT displays only a 0.179% increase in flip-flop usage compared to the benchmark circuit.

(4) A THT-AFA method is proposed and applied to the security study of SKINNY. The results show that after activating the THT 3 times (i.e., 3 single-bit fault injections), all data bits of the master key can be obtained with a 100% success rate, and the average time to solve is 64.57 s.

The rest of this paper is structured as follows. In Section 2, we briefly introduce the SKINNY cipher. The fault model and fault propagation analysis are given in Section 3. In Section 4, we propose the THT. Algebraic fault equations are established, and SKINNY attack experiments are presented, in Sections 5 and 6, respectively. Finally, Section 7 presents the conclusions.

2 The SKINNY Block Cipher

In this section, we give some of the notations used in the paper, followed by a brief description of the design specifications of SKINNY.

2.1 Notations

$\oplus$: The XOR operation.

$\|$: The bit splice notation.

$P = m_0 \| m_1 \| \ldots \| m_{15}$: The plaintext data.

$MK = tk_0 \| tk_1 \| \ldots \| tk_{15}$: The master key data.

$C = C_0 \| C_1 \| \ldots \| C_{15}$: The correct ciphertext data.

$C^* = C_0^* \| C_1^* \| \ldots \| C_{15}^*$: The erroneous ciphertext data.

$\Delta C = C \oplus C^*$: The differential value of the output ciphertext.

X_r^i: The i-th bit at the input state of SubCells at round r.

$X_{r,i}$: The i-th basic arithmetic cell at the input state of SubCells at round r.

2.2 The Design Specifications of SKINNY

For SKINNY, its internal state (IS) is loaded row-wise, similar to the way in which tweakey states are loaded; see [1] for details of its design.

2.2.1 The Encryption Process of SKINNY

The number of rounds of encryption for SKINNY is 32. Similar to many other block ciphers, the internal state of SKINNY is repeatedly updated. The encryption process for SKINNY is shown in Fig. 1. It is important to note that the round key used by SKINNY in encryption is derived from the key schedule based on the master key. The SKINNY round function contains five different operations: SubCells (SC), AddConstants (AC), AddRoundTweakey (ART), ShiftRows (SR), and MixColumns (MC). SC^{-1}, AC^{-1}, ART^{-1}, SR^{-1}, and MC^{-1} denote the inverse of SC, AC, ART, SR, and MC.

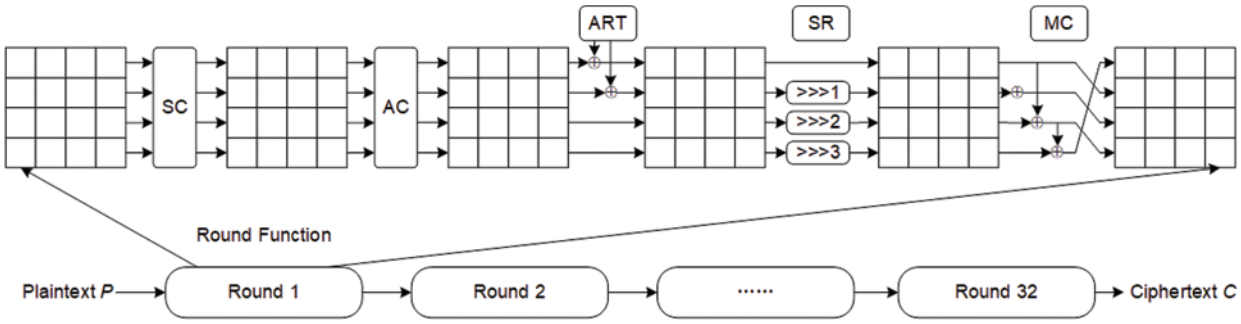

Figure 1: The encryption process of SKINNY

SC. The 4-bit S-box S_4 is applied to every cell of the internal state of SKINNY, and the details of this S-box for hexadecimal data substitution are shown in Table 1.

Table 1: The 4-bit S-box used in SKINNY

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S_4[x]$	c	6	9	0	1	a	2	b	3	8	5	d	4	e	7	f

AC. The round constants are generated using a 6-bit affine linear feedback shift register (LFSR) with states denoted as (rc5, rc4, rc3, rc2, rc1, rc0) (rc0 is the least significant bit) and are XOR with the intermediate states of the cipher.

ART. The round key that has been key-programmed needs to have the first 32 bits of data extracted and then be XOR with the internal state of the cipher.

SR. The basic operator cells in the second, third, and fourth rows of the SKINNY cipher state cell array are cyclically shifted to the right by 1, 2, and 3 positions, respectively.

MC. Each column of the cipher internal state array needs to be left-multiplied by the binary matrix M.

2.2.2 The Key Schedule of SKINNY

The key schedule of SKINNY is implemented by permutation P, where P = [9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7]. The permutation P is applied to the cell positions of the array of internal state cells of the round key: for all $0 \leq i \leq 15$, there is $IS_i \leftarrow IS_{P[i]}$. The key schedule of SKINNY is shown in Fig. 2.

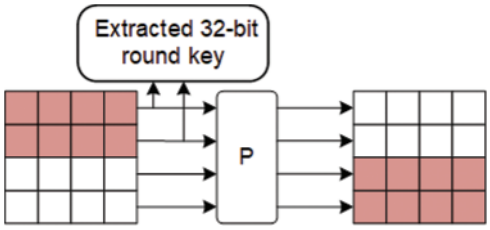

Figure 2: The key schedule of SKINNY

3 The Fault Model and Fault Propagation Analysis of SKINNY

To present a clearer picture of the SKINNY fault attack and the associated issues involved in the propagation of the fault, the following definitions are given.

Definition 1. Suppose that the number of encryption rounds of a cryptographic algorithm A is R, and an attacker injects a fault into the intermediate state data IS in round r ($1 \le r \le R$) of A at some point, and its corresponding output error ciphertext is C^*. The number of bits of the fault acting on IS is said to be the fault width, i.e., f_w; let f_d be the attack depth of the fault, which represents the total number of rounds of fault propagation between the fault injection location and the error ciphertext output, $f_d = R - (r-1)$. Let $\varphi(K)$ denote the residual entropy of the master key after fault injection, i.e., the $\log_2$ result for the key search space. In addition, let N denote the number of fault injections under a complete fault attack.

3.1 Fault Models

3.1.1 Random Nibble Fault Injection Model

In this paper, random nibble faults are injected into the SKINNY cipher using software simulation and this model is applied to the study of AFA for this cipher, as detailed in Section 6.1.

(1) The cipher uses the same master key in its operations and the attacker has access to the correct ciphertext data C and the error ciphertext data C^* corresponding to the same plaintext data P.
(2) A random fault with fault width $f_w = 4$ is injected at the 28th SC input of the SKINNY, the fault location f_l and the fault value f_v are unknown, and the fault depth $f_d = 5$.

3.1.2 Specific Single-Bit Fault Injection Model

This model was implemented based on the THT and combined with AFA to study the security of SKINNY, as detailed in Section 6.2.

(1) See (1) in Section 3.1.1 for details.

(2) A single-bit fault of fault width $f_w = 1$ is injected at the SC input of round 27 of SKINNY, with known fault location f_l and fault value f_v, and fault depth $f_d = 6$.

3.2 Analysis of the Fault Propagation Process

The length of the basic arithmetic cell of SKINNY is 4 bits, and the differential fault propagation path when $(f_d, f_l) = (5, 0)$ is shown in Fig. 3. The blank part of the diagram indicates that there are no faulty differentials and the part marked in yellow is a faulty collision (i.e., faults meet).

Definition 2. Let f_e be the extension of the fault, which represents the number of associated master keys involved in the propagation of the fault. $W(f_d, f_l) = \{i \mid tk_i$ is the master key involved in the fault propagation process, $0 \le i \le 15, 1 \le f_d \le 32, 0 \le f_l \le 15$, where $i, f_d, f_l \in Z\}$, and $f_e(f_d, f_l) = |W(f_d, f_l)|$. From Fig. 3, $f_e(5, 0) = 13$, and by the same token, when $f_d = \{5, 6\}$, $0 \le f_l \le 15$, the corresponding f_e is shown in Table 2. Clearly, $f_e(f_d, f_l)$ is maximum when $8 \le f_l \le 11$, i.e., the key information involved in the fault propagation process is more complete and therefore they are more likely to be selected for a fault attack under the specific single-bit fault injection model.

During SKINNY fault propagation, the output differential value ΔSC^R of the SC operation of the R-th round can be calculated directly from the differential value ΔC of the output ciphertext.

$$\Delta SC^R = \left(\mathrm{AC}^{-1}\left(\mathrm{ART}^{-1}\left(\mathrm{SR}^{-1}\left(\mathrm{MC}^{-1}(\Delta C)\right)\right)\right)\right) \tag{1}$$

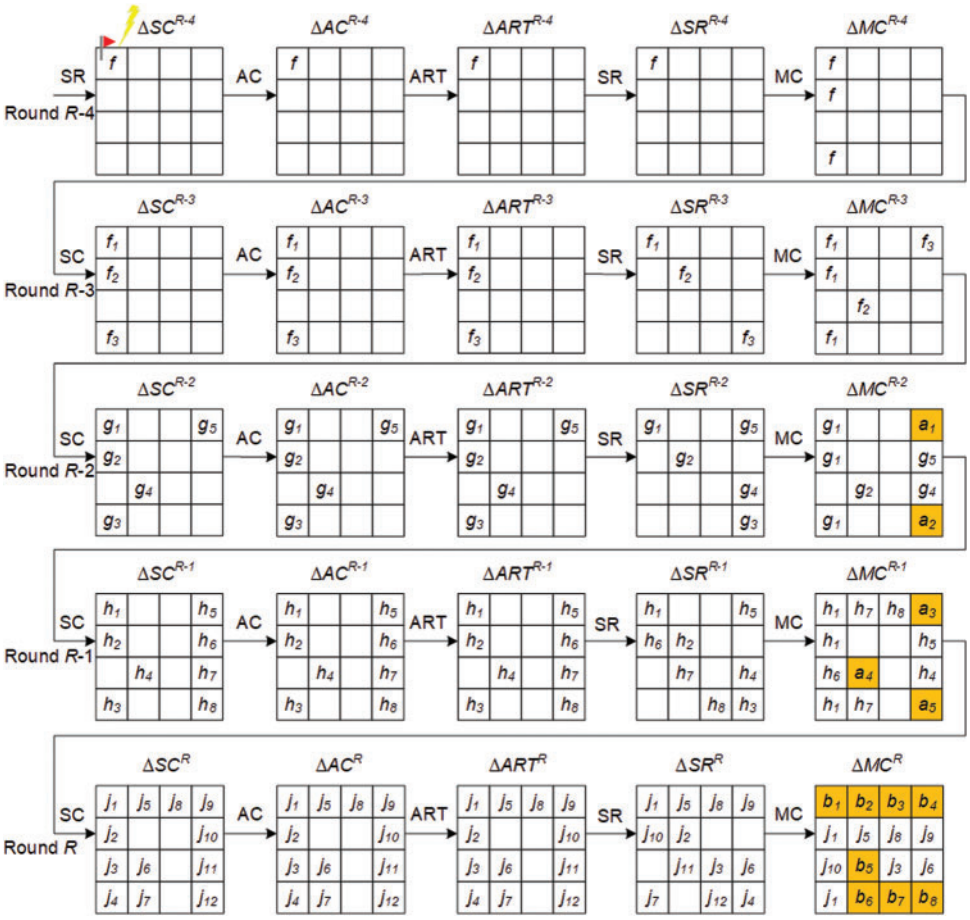

Figure 3: Differential fault propagation process for SKINNY

Table 2: Index values of the master key involved in the fault propagation process

f_l	$W(f_d, f_l)$	
	$f_d = 5$	$f_d = 6$
0	{1 2 4 5 6 7 8 9 10 11 12 14 15}	{0 1 2 4 5 6 7 8 9 10 11 12 13 14 15}
1	{0 2 3 6 7 8 9 10 11 12 13}	{0 2 3 4 6 7 8 9 10 11 12 13 14 15}
2	{0 3 4 5 6 8 9 10 11 12 13 14}	{0 2 3 4 5 6 7 8 9 10 11 12 13 14 15}
3	{0 1 4 5 7 8 9 10 11 12 14 15}	{0 1 2 4 5 6 7 8 9 10 11 12 13 14 15}
4	{0 1 4 7 8 9 10 11 13 14 15}	{0 1 2 3 4 5 7 8 9 10 11 12 13 14 15}
5	{0 2 4 6 7 8 9 10 11 12 14 15}	{0 1 2 4 5 6 7 8 9 10 11 12 13 14 15}
6	{0 2 3 7 8 9 10 11 12 13 15}	{0 1 2 3 4 6 7 8 9 10 11 12 13 14 15}
7	{0 2 4 5 8 9 10 11 12 13 14}	{0 2 3 4 5 6 7 8 9 10 11 12 13 14 15}
8	{0 2 3 4 5 6 7 8 9 10 11 12 13 14 15}	{0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15}
9	{0 1 2 3 4 5 7 8 9 10 11 12 13 14 15}	{0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15}
10	{0 1 2 4 5 6 7 8 9 10 11 12 13 14 15}	{0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15}

(Continued)

Table 2: Continued

f_l	$W(f_d, f_l)$	
	$f_d = 5$	$f_d = 6$
11	{0 1 2 3 4 6 7 8 9 10 11 12 13 14 15}	{0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15}
12	{1 4 7 9 10 11 14 15}	{0 1 4 5 7 8 9 10 11 12 14 15}
13	{2 4 6 7 8 10 11 12 14 15}	{1 2 4 5 6 7 8 9 10 11 12 14 15}
14	{0 2 3 8 9 11 12 13}	{0 2 3 6 7 8 9 10 11 12 13}
15	{0 4 5 6 9 10 12 13 14}	{0 3 4 5 6 8 9 10 11 12 13 14}

For $(f_d, f_l) = (5, i)\,(0 \le i \le 15)$, there are all corresponding differential fault propagation diagrams for which the output differential value of the SC for the R-th round is calculated separately, and the 16 cases in Fig. 4 occur. In particular, it should be noted that the white cells within the diagram represent a value of 0 for the SC output differential value ΔSC^R of the R-th round of the cipher.

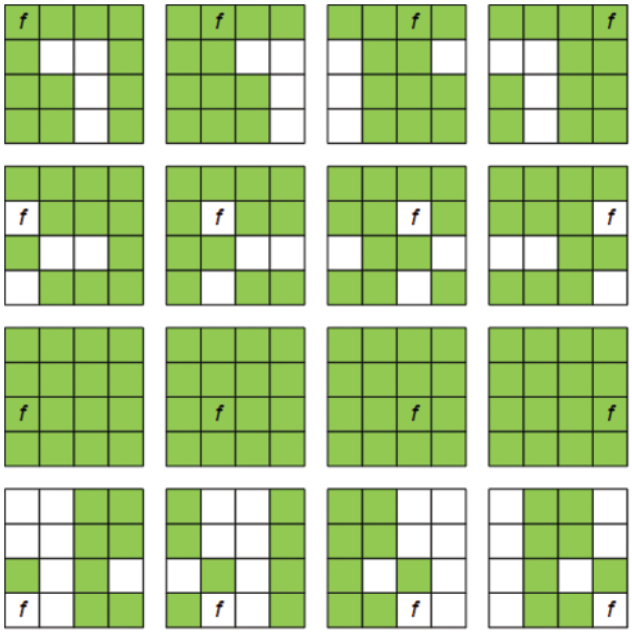

Figure 4: The fault propagation diagram for each fault location in round R-4 for SKINNY

In theory, if a fault can be propagated to all bits of the master key, this key can be fully recovered. As shown in Table 2, when $f_d = 5$, it is possible to obtain the master key of SKINNY by injecting two faults, while, for $f_d = 6$, one fault is possible. Let us set $f_l \in U$; when $f_d = 5$, $\sum_{f_d \in U} f_e(f_d, f_l) = 16$; then, there is $|U|_{\min} = 2$; at this point, the set U has 30 elements, $U \in \{(0, 1), (0, 2), (0, 6), (0, 8), (0, 9), (0,11), (0,14), (1, 3), (1, 9), (1, 10), (2, 9), (2, 10), (2, 11), (3, 8), (3, 11), (4, 8), (5, 9), (6, 10), (7, 11), (8, 9), (8, 10), (8, 11), (8, 12), (9, 10), (9, 11), (9, 13), (9, 15), (10, 11), (10, 14), (11, 15)\}$. Moreover, when $f_d = 6$, $\sum_{f_d \in U} f_e(f_d, f_l) = 16$; then, there is $|U|_{\min} = 1$; at this point, the set U has 4 elements, $U \in \{8, 9, 10, 11\}$.

Interestingly, when a fault is injected ($f_w = 1$ or 4) into $X_{28,i}\,(0 \le i \le 7, 12 \le i \le 15)$, a graph of the distribution of differential values of the SC output that can be uniquely identified appears, as shown

in Fig. 4 (the f in the graph indicates the presence of a fault in this basic arithmetic cell). In response, this feature can be used to reduce the search space at the fault location during the attack on SKINNY, thereby reducing the solving time of the master key.

4 The Proposed THT

Typically, a HT is designed with two parts: the trigger logic (TL) and the payload logic (PL). Of these, the TL is used to determine if the activation condition is met by a signal line and state value that references the attacker's pre-defined signal line and state value. Once the activation condition is met, the PL performs the corresponding attack (e.g., denial of service, alteration of the original circuit function, or disclosure of secret information, etc.).

4.1 Prerequisites for the Implementation of THT

In this paper, the premises for the design and implementation of the HT circuit are as follows.

(1) SKINNY is implemented as cryptographic IP with certain protections (e.g., sensors from untrusted IP core vendors). The prototype of this cryptographic IP was implemented on a 28 nm Xilinx ZYNQ-7010 FPGA development board. The fact that physical sensors (e.g., temperature sensors) are deployed together with cryptographic IP is a common practice in industrial designs—for example, the LKT4202U [25] security chip. The LKT4202U not only integrates various secure encryption modules (e.g., RSA, DES, AES, etc.) on-chip, but also deploys various physical sensors such as temperature and voltage sensors.
(2) The attacker can insert a HT by directly modifying the design input of the RTL or specific logical elements of the Netlist. In addition, they can only access the Xilinx design language (XDL) [26] file and not the design phase.
(3) Under specific temperature conditions, the HT is activated to flip a specific bit of the intermediate state of the cipher. In this case, the temperature can be changed by a heat gun or a household hair dryer.

4.2 Design of the THT

The design of the THT is shown in Fig. 5. We have chosen to insert the THT at X_{27}^{32}, which has a deeper depth for fault attacks, making it easier for the THT to evade detection, based on certain characteristics generated by the fault during propagation (see Section 3.2 for details).

The left part of Fig. 5 shows the encryption process of SKINNY, while the right part shows the designed THT, which consists of two parts: ***Trigger*** and ***Payload***. When the temperature detected by the temperature sensor reaches the threshold value set by the attacker, a "1" stored in flip-flop FF_1 activates the THT. The payload of the THT consists of two parts: ***Payload* (A)** and ***Payload* (B)**. The three input signals a, b, c of the XOR gate $F_1(a, b, c)$ are connected to X_{27}^{32}, the trigger state of THT and R_flag, respectively. R_flag is the state information of the SKINNY encryption round, which is mainly responsible for comparing whether the current round is the target round in real time. The signal output by R_flag is "1" when and only when the current round of the SKINNY encryption operation is the same as the target round. When both input signals b and c of AND gate $F_2(b, c)$ are "1", CE is "0" and ***Payload* (B)** temporarily disables the loading of new plaintext data during the next round of SKINNY encryption, so that the same plaintext can be the pair of correct and incorrect ciphertexts that is obtained. The attacker only needs to perform ciphertext-only analysis on the ciphertext to recover the master key.

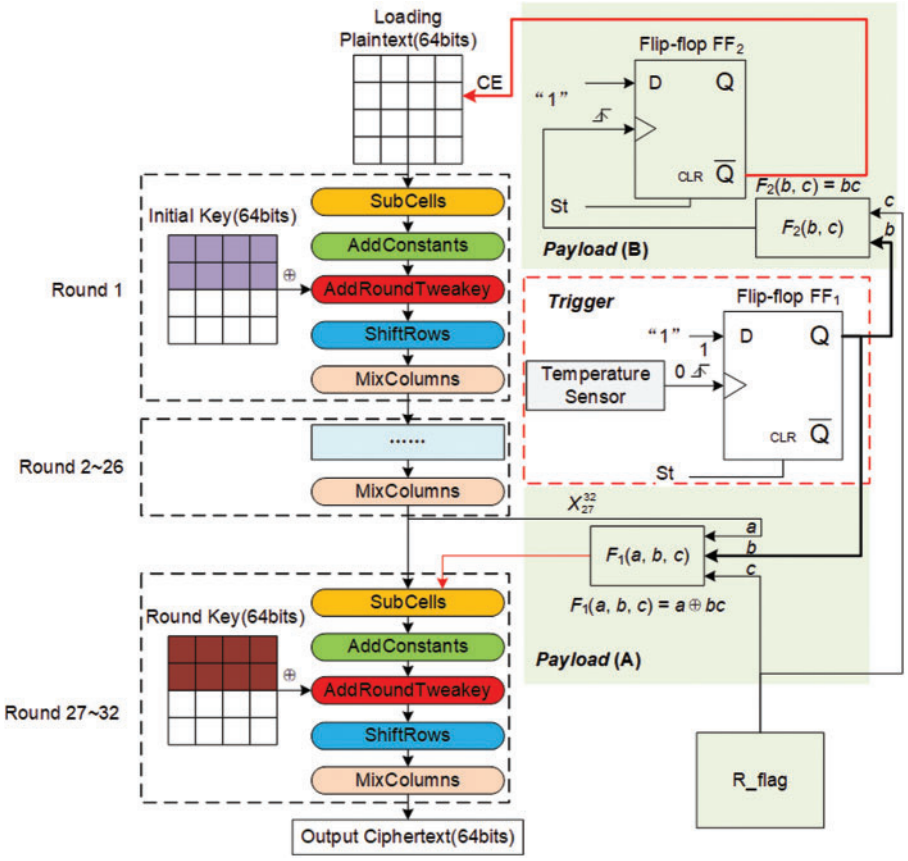

Figure 5: The design of THT

4.2.1 Design of the TL

In this paper, the THT that we have designed is implemented on a 28 nm Xilinx ZYNQ-7010 FPGA (xc7z010clg400-1), which has an internal integrated temperature sensor that generates a 12-bit temperature output in the range [−273°C, + 230.7°C], e.g., 977 h indicates that the current temperature is +25°C. We directly use the rising edge of the summed 8th- and 9th-bit temperature outputs as the trigger signal for the THT, i.e., the THT is triggered when the on-chip temperature is above the threshold value of +73.34°C (b00 h). An inexpensive heat gun can trigger the THT, and the test platform is shown in Fig. 6, with (1) the 28 nm Xilinx ZYNQ-7010 FPGA development board; (2) the air outlet of the heat gun; and (3) the temperature variations within the chip. The hardware overhead required for the TL implementation is one flip-flop (FF_1).

4.2.2 Design of the PL

Suppose, in the k-th encryption process, that E_k and P_k denote the current encryption and the plaintext loaded, respectively. For the same plaintext, a pair of correct and incorrect ciphertexts (C, C^*) are obtained by the attacker. As shown in Fig. 5, PL consists of two parts: ***Payload* (A)** is used to inject a bit-level fault at X_{27}^{32} during E_k, while ***Payload* (B)** is used to temporarily disable the loading of new plaintext at E_{k+1}. The THT is activated by the "1" stored in FF_1 when the on-chip temperature of the FPGA reaches the threshold set by the attacker. In addition, the R_flag signal indicates whether the

current round is the target round of 27. When both conditions are met, the fault is injected into the SKINNY circuit.

Figure 6: The THT test platform

4.3 The Hardware Overhead of THT

Cryptographic hardware is mostly provided in the form of third-party IP cores, for which we have designed and packaged our own IP cores for the SKINNY circuit. The basic logic unit of the 28 nm Xilinx ZYNQ-7010 FPGA is the Slice, which contains four 6-input look-up tables (LUTs). We can search the XDL for used LUTs, some of which have three or fewer used inputs, and $F_1(a, b, c)$ and $F_2(b, c)$ in Fig. 5 can be merged with them, simply by modifying the corresponding Slice instance on the XDL. In summary, the only hardware resources required to implement THT are the two flip-flops (FF_1 and FF_2). After the THT was inserted into the SKINNY circuit, it was synthesized and implemented using the Vivado 18.3 development tool and verified using the FPGA development board. Compared to the benchmark circuit, the SKINNY circuit with THT inserted only increases the number of flip-flops used by 0.179% and the on-chip power (Dynamic: 1.277 W, Static: 0.112 W) is the same for both, with the hardware resource consumption detailed in Table 3.

Table 3: Hardware overhead for circuits

Circuits	Resource parameters				
	LUT	LUTRAM	Flip-flop	BUFG	Power
SKINNY	718	60	1116	1	1.389 W
THT SKINNY	718	60	1118	1	1.389 W
Overhead	0	0	+0.179%	0	0

5 The AFA for SKINNY

When performing AFA on a cipher, building a system of decryption equations from the ciphertext can effectively reduce the solution time. In this section, we give not only the construction of the system of equations for the decryption of SKINNY, but also the algebraic system of equations representation of the fault information.

5.1 Construction of Systems of Algebraic Equations for SKINNY

5.1.1 Algebraic Equation Representation of SC^{-1}

Assuming that the inputs and outputs of a 4-bit S-box are (x_1, x_2, x_3, x_4) and (y_1, y_2, y_3, y_4), respectively, where x_4 and y_4 are denoted as the lowest significant bits of the SC inputs and outputs, respectively, the SC^{-1} can be expressed as Eq. (2).

$$\begin{aligned} x_1 &= y_1 + y_2 + y_3y_4 + y_2y_3 + y_1y_3 + y_1y_4 + y_1y_2y_3 \\ x_2 &= y_2 + y_3 + y_4 + y_1y_4 + y_1y_2 \\ x_3 &= 1 + y_1 + y_2 + y_4 + y_1y_2 \\ x_4 &= 1 + y_2 + y_3 + y_4 + y_3y_4 + y_2y_4 + y_1y_3 + y_1y_2y_4 + y_1y_2y_3 \end{aligned} \tag{2}$$

5.1.2 Algebraic Equation Representation of AC^{-1}

Let $(z_5, z_4, z_3, z_2, z_1, z_0)$ be a vector of round constants for the r-th round of the cipher, x_i denote the input to AC, and y_i denote the output data of AC, where $0 \le i \le 63$; then, we can derive Eq. (3).

$$\begin{aligned} y_i &= x_i + 1, \quad i = 34 \\ y_i &= x_i + z_5, \quad i = 18 \\ y_i &= x_i + z_4, \quad i = 19 \\ y_i &= x_i + z_3, \quad i = 0 \\ y_i &= x_i + z_2, \quad i = 1 \\ y_i &= x_i + z_1, \quad i = 2 \\ y_i &= x_i + z_0, \quad i = 3 \\ y_i &= x_i, \qquad \text{Others} \end{aligned} \tag{3}$$

5.1.3 Algebraic Equation Representation of ART^{-1}

In the ART operation, suppose that x_i and y_i denote the 1-bit input state and round key, respectively, and let z_i denote the output data of ART. ART^{-1} can be expressed as Eq. (4).

$$\begin{aligned} \mathrm{x_i} &= y_i + z_i, \ 0 \le i \le 31 \\ x_i &= z_i, \qquad \text{Others} \end{aligned} \tag{4}$$

5.1.4 Algebraic Equation Representation of SR^{-1}

Suppose that x_i and y_i $(0 \le i \le 63)$ denote one input and output of SR, respectively. Let the vectors SP^{-1} denote the shifted nibble indices of SR^{-1}, and SP^{-1} [16] = [0, 1, 2, 3, 5, 6, 7, 4, 10, 11, 8, 9, 15, 12, 13, 14]. Then, SR^{-1} can be expressed as Eq. (5). Where % indicates the mod function.

$$x_i = y_{\left(4\times \mathrm{SP}^{-1}[i/4]\right)+i\%4} \tag{5}$$

5.1.5 Algebraic Equation Representation of MC^{-1}

First, in this paper, the inverse matrix M^{-1} can be derived from the binary matrix M.

$$M^{-1} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Assuming that the 1-bit inputs and outputs of the MC are x_i and y_i, respectively, the MC^{-1} can be expressed as Eq. (6).

$$\begin{aligned} x_i &= y_{i+16} \\ x_{i+16} &= y_{i+16} + y_{i+32} + y_{i+48} \\ x_{i+32} &= y_{i+16} + y_{i+48},\ i \in \{0, 1, \ldots, 15\} \\ x_{i+48} &= y_i + y_{i+48} \end{aligned} \tag{6}$$

5.2 Construction of Systems of Algebraic Equations for Fault Information

In the correct encrypted state, let X be 64 bits of intermediate state data; then, we have $X = (x_0 \parallel x_1 \parallel \ldots \parallel x_{63})$. Moreover, let X^* denote the 64 bits of incorrectly encrypted data following the injection of a fault, $X^* = \left(x_0^* \parallel x_1^* \parallel \ldots \parallel x_{63}^*\right)$. Let Z be the fault differential between X and X^*. Then, we can derive Eq. (7).

$$Z = z_0 \parallel z_1 \parallel \ldots \parallel z_{63},\ z_i = x_i \oplus x_i^*, 0 \le i \le 63 \tag{7}$$

When $f_w = 4$, Z can further be divided into 16 4-bit variables, i.e., $Z = (Z_0||Z_1|| \ldots ||Z_{14}||Z_{15})$, and we have Eq. (8).

$$Z_i = z_{4\times i} \parallel z_{4\times i+1} \parallel z_{4\times i+2} \parallel z_{4\times i+3}, 0 \le i \le 15 \tag{8}$$

Depending on whether the location l_k, l_u $(0 \le l_k \le 63, 0 \le l_u \le 15)$ at which the fault is injected is known to the attacker, Z can be represented by different algebraic equations.

5.2.1 Known Fault Location

In the actual attack environment of cryptographic algorithms, it is difficult for an attacker to determine the exact location of the fault injection. However, the THT that we have designed is capable of precisely achieving bit-level fault injection. In Eq. (9), z_i denotes that there is no fault at the i-th location.

$$z_i = 0, 0 \le i \le 63, i \ne l_k \tag{9}$$

5.2.2 Unknown Fault Location

In this paper, we use software simulation to perform a random nibble fault injection into SKINNY. In addition, we indicate whether Z_i has been injected with a fault by introducing 16 variables u_i.

$$u_i = \prod_{i=4\times l_u}^{4\times l_u+3} (1 + z_i), 0 \le i \le 15 \tag{10}$$

If $u_i = 0$, then Z_i is the location of the injected fault. Since $u_0, u_1, \ldots, u_{15}$ has and only has a zero, we can derive Eq. (11).

$$(1 + u_0) \vee (1 + u_1) \vee \ldots \vee (1 + u_{15}) = 1, u_i \vee u_j = 1, 0 \leq i < j \leq 15 \tag{11}$$

According to Eqs. (10) and (11), Z needs to be represented by 96 variables and 272 conjunctive normal form (CNF) equations.

6 The Attack Experiments of SKINNY

In this paper, the software program for SKINNY was written using the C language. In addition, we have used Bosphorus [27] and CryptoMiniSat 5.8.0 [28] for the solution of the algebraic systems of equations, both running on a laptop with an Intel® Core (TM) i5-6300HQ, 2.30 GHz, 8 G RAM and Ubuntu 20.04.2 LTS 64-bit operating system.

6.1 The AFA of SKINNY

We perform random nibble fault injection at the SC input of round 28 of SKINNY. The fault injection in this subsection is implemented through software simulation. For different numbers of fault injections, we conducted 1000 separate attack experiments, the results of which are shown in Fig. 7. When the number of fault injections is $N = 1$, the residual entropy of the key $\varphi(K)$ can be reduced to 25.751 bits on average, and 2 fault injections are expected to reduce it to a small value. When the number of fault injections reaches 6, the residual entropy of the key is reduced to 0.058 bits on average, the success rate of the attack is 97.3%, and 94.8% of the attacks can be solved within 1 h.

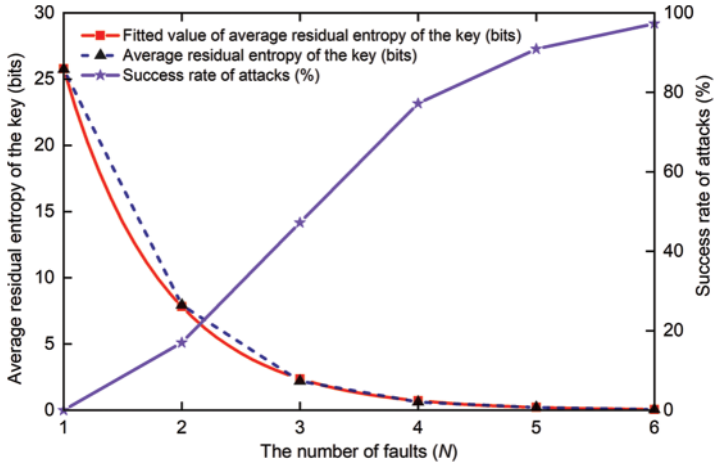

Figure 7: Results of the AFA of SKINNY

In addition, at a confidence level of 95%, we give the results of the fit (shown in the red curve in Fig. 7) based on the actual fault attack results, which is informative and can be represented by Eq. (12).

$$\varphi(K) = 85 \times e^{-1.193N},\ N \in \{1, 2, \ldots\} \tag{12}$$

6.2 The THT-AFA of SKINNY

First, the THT is inserted into the SKINNY circuit at X_{27}^{32}. The SKINNY circuit is deployed on a 28 nm Xilinx ZYNQ-7010 FPGA development board, and we activate the THT by using a hot air gun

to attack the main chip of the board. When the temperature of the main chip of the FPGA development board reaches the threshold that we set, the THT is triggered and the bit data at X_{27}^{32} are successfully flipped, and we can obtain the corresponding ciphertext pair (C, C^*). Due to the extremely sparse diffusion layer for SKINNY, there is only a limited number of data bits that can affect the cipher after a single fault injection, and the efficiency of attack is not ideal. In response, we gradually increased the number of attacks until a better result emerged. When $N = 2$ and 3, 1000 sets of experiments were conducted on them, respectively, and the results of the attacks are shown in Fig. 8. Two single-bit fault injections can obtain the master key with a success rate of 94.3%, and the average time for their attack is 44.78 s. By increasing the number of fault injections to 3, the success rate of cracking the key reaches 100% and the average time of its attack rises to 64.57 s.

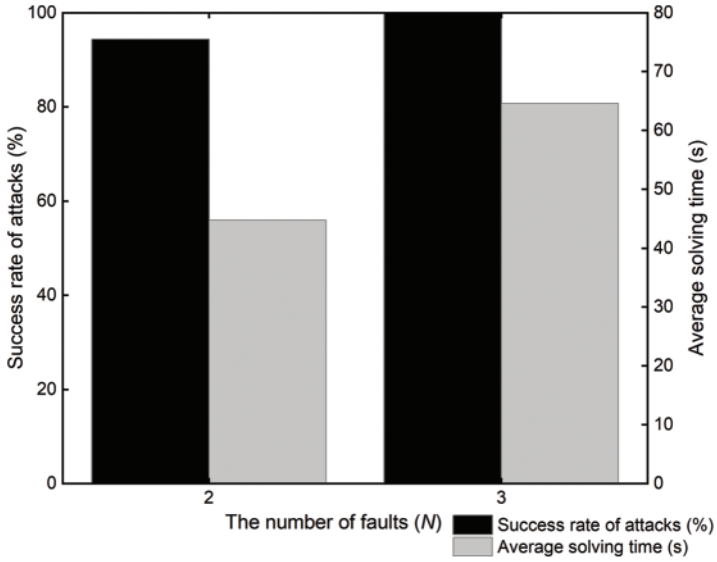

Figure 8: The THT-AFA for SKINNY

The experimental results show that the success rate of the cipher being broken gradually increases as the number of times that the THT is activated increases. However, as the number of algebraic equations required then increases, its solution time also increases. In summary, the optimum number of fault attacks is 3, i.e., the THT is activated 3 times.

6.3 Analysis and Discussion

The complexity of our proposed THT-AFA method is mainly reflected by the number of fault injections and the solving time of the master key. For different cryptanalysis methods, the efficiency of their attacks is determined by the following factors:

(1) The higher success rate of the attack;
(2) The smaller solving time of the master key;
(3) The deeper the f_d and the narrower the f_w.

In fact, using fewer fault injections would be more relevant to the actual attack environment. Table 4 shows the existing works on instance attacks on SKINNY and provides the relationship between the precondition, the fault models, the number of faults, the average solution time, and the success rate for various methods. Vafaei et al. [29] successfully performed differential fault analysis

(DFA) on SKINNY using approximately 10 faults (2–3 faults injected at each of 4 different locations). Xu et al. [30] performed an enhanced persistent fault analysis (EPFA) of SKINNY with the required number of faults of 1500–1600. In [31], the authors not only performed algebraic persistent fault analysis of SKINNY based on known plaintexts (KP-APFA), but also combined S-box decomposition with constraint-based algebraic fault analysis (SD-APFA) to achieve a security analysis of SKINNY. Although some attacks were as low as 26.17 s, they used five times as many faults as we did. In a comprehensive comparison, our proposed THT-AFA can recover the master key of SKINNY with the least number of faults and less time.

Table 4: Comparison with existing works

Methods	Precondition	Fault models	Faults	Average time	Success rate	References
DFA	Known plaintext/ciphertext	$f_d = 5, f_w = 4$	Approx. 10	—	—	[29]
EPFA	Ciphertext only	$f_d = 32$, $f_w = 4$	1500–1600	—	100%	[30]
KP-APFA	Known plaintext/ciphertext	$f_d = 4, f_w = 4$	11	Minimum 2000 s	—	[31]
SD-APFA	Ciphertext only	$f_d = 6, f_w = 4$	16	26.17 s	100%	[31]
		$f_d = 8, f_w = 4$	14	83.42 s	100%	
		$f_d = 10$, $f_w = 4$	12	956.65 s	86%	
AFA	Known plaintext/ciphertext	$f_d = 5, f_w = 4$	6	1180.11 s	97.3%	This paper
THT-AFA	Ciphertext only	$f_d = 6, f_w = 1$	3	64.57 s	100%	This paper

7 Conclusion

In this paper, a THT-AFA method is proposed for the SKINNY cipher. We design a THT, which is implemented on a 28 nm Xilinx ZYNQ-7010 FPGA development board with a hardware overhead of only 2 flip-flops. We can change the temperature inside the main chip of this FPGA development board with a hot air gun. The THT is triggered when the temperature inside the chip reaches the threshold that we set, thus performing a bit-level fault injection into SKINNY. In addition, we performed an AFA of the SKINNY cipher under the random nibble fault injection model. When the number of random nibble faults reached 6, the residual entropy of the key for this cipher was reduced to 0.058 bits on average, and the success rate of its attack was only 97.3%. The master key of SKINNY can be recovered with a 100% success rate by flipping the bit at X_{27}^{32} only 3 times, with an average time to attack of 64.57 s. In this work, we show that lightweight block ciphers are vulnerable to AFA and that HT is a significant potential threat to secure chips.

Acknowledgement: We would like to thank the handling editor and the anonymous reviewers for their careful reading and helpful remarks.

Funding Statement: This paper was supported in part by the Natural Science Foundation of Heilongjiang Province of China (Grant No. LH2022F053), in part by the Scientific and technological development project of the central government guiding local (Grant No. SBZY2021E076), in part by the Postdoctoral Research Fund Project of Heilongjiang Province of China (Grant No. LBH-Q21195), in part by the Fundamental Research Funds of Heilongjiang Provincial Universities of China (Grant No. 145209146), and in part by the National Natural Science Foundation of China (NSFC) (Grant No. 61501275).

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi *et al.,* "The skinny family of block ciphers and its low-latency variant mantis," in *Proc. CRYPTO 2016*, Santa Barbara, CA, USA, pp. 123–153, 2016.

[2] A. Barenghi, G. Bertoni, E. Parrinello and G. Pelosi, "Low voltage fault attacks on the RSA cryptosystem," in *Proc. FDTC 2009*, Lusanne, Switzerland, pp. 23–31, 2009.

[3] A. Barenghi, G. M. Bertoni, L. Breveglieri, M. Pellicioli and G. Pelosi, "Low voltage fault attacks to AES," in *Proc. HOST 2010*, Anaheim, CA, USA, pp. 7–12, 2010.

[4] C. H. Kim and J. J. Quisquater, "Fault attacks for CRT based RSA: New attacks, new results, and new countermeasures," in *Proc. WISTP 2007*, Heraklion, Crete, Greece, pp. 215–228, 2007.

[5] J. Rodriguez, A. Baldomero, V. Montilla and J. Mujal, "LLFI: Lateral laser fault injection attack," in *Proc. FDTC 2019*, Atlanta, GA, USA, pp. 41–47, 2019.

[6] T. Korak, "Investigation of parameters influencing the success of optical fault attacks," in *Proc. FPS 2013*, La Rochelle, France, pp. 140–157, 2013.

[7] N. Theißing, D. Merli, M. Smola, F. Stumpf and G. Sigl, "Comprehensive analysis of software countermeasures against fault attacks," in *Proc. DATE 2013*, Grenoble, France, pp. 404–409, 2013.

[8] J. Breier, D. Jap, X. Hou, S. Bhasin and Y. Liu, "SNIFF: Reverse engineering of neural networks with fault attacks," *IEEE Transactions on Reliability*, vol. 71, no. 4, pp. 1527–1539, 2022. https://doi.org/10.1109/TR.2021.3105697

[9] A. Barenghi, L. Breveglieri, I. Koren and D. Naccache, "Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3056–3076, 2012. https://doi.org/10.1109/JPROC.2012.2188769

[10] R. S. Chakraborty, I. Saha, A. Palchaudhuri and G. K. Naik, "Hardware trojan insertion by direct modification of FPGA configuration bitstream," *IEEE Design & Test*, vol. 30, no. 2, pp. 45–54, 2013. https://doi.org/10.1109/MDT.2013.2247460

[11] J. Zhang and Q. Xu, "On hardware trojan design and implementation at register-transfer level," in *Proc. HOST 2013*, Austin, TX, USA, pp. 107–112, 2013.

[12] S. Mal-Sarkar, A. Krishna, A. Ghosh and S. Bhunia, "Hardware trojan attacks in fpga devices: Threat analysis and effective counter measures," in *Proc. GLSVLSI '14*, Houston, Texas, USA, pp. 287–292, 2014.

[13] T. Hoque, X. Wang, A. Basak, R. Karam and S. Bhunia, "Hardware trojan attacks in embedded memory," in *Proc. VTS 2018*, San Francisco, CA, USA, pp. 1–6, 2018.

[14] J. Clements and Y. Lao, "Hardware trojan design on neural networks," in *Proc. ISCAS 2019*, Sapporo, Japan, pp. 1–5, 2019.

[15] J. Ye, Y. Hu and X. Li, "Hardware trojan in FPGA CNN accelerator," in *Proc. ATS 2018*, Hefei, China, pp. 68–73, 2018.

[16] M. H. Khan, R. Gupta, J. Jose and S. Nandi, "Dead flit attack on NoC by hardware trojan and its impact analysis," in *Proc. NoCArc '21*, Athens, Greece, pp. 10–15, 2021.

[17] S. Bhasin, J. L. Danger, S. Guilley, X. T. Ngo, L. Sauvage *et al.,* "Hardware trojan horses in cryptographic IP cores," in *Proc. FDTC 2013*, Los Alamitos, CA, USA, pp. 15–29, 2013.

[18] D. Knichel, T. Moos and A. Moradi, "The risk of outsourcing: Hidden SCA trojans in third-party IP-cores threaten cryptographic ICs," in *Proc. ETS 2020*, Tallinn, ESTONIA, pp. 1–6, 2020.
[19] M. Hutter and J. M. Schmidt, "The temperature side channel and heating fault attacks," in *Proc. CARDIS 2013*, Berlin, Germany, pp. 219–235, 2013.
[20] R. Kumar, P. Jovanovic and I. Polian, "Precise fault-injections using voltage and temperature manipulation for differential cryptanalysis," in *Proc. IOLTS 2014*, Platja d'Aro, Spain, pp. 43–48, 2014.
[21] P. Dash, C. Perkins and R. M. Gerdes, "Remote activation of hardware trojans via a covert temperature channel," in *Proc. SecureComm 2015*, Dallas, TX, USA, pp. 294–310, 2015.
[22] S. Ghandali, D. Holcomb and C. Paar, "Temperature-based hardware trojan for ring-oscillator-based TRNGs," arXiv preprint arXiv:1910.00735, 2019.
[23] N. T. Courtois, K. Jackson and D. Ware, "Fault-algebraic attacks on inner rounds of DES," in *Proc. E-Smart '10*, Montreuil, France, pp. 1–59, 2010.
[24] Internet, "7 Series FPGAs and Zynq-7000 SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter User Guide (UG480)," [Online]. Available: https://docs.xilinx.com/r/en-US/ug480_7Series_XADC/7-Series-FPGAs-and-Zynq-7000-SoC-XADC-Dual-12-Bit-1-MSPS-Analog-to-Digital-Converter-User-Guide-UG480. (accessed on 28 September 2022).
[25] Internet, "LKT4202U," [Online]. Available: http://www.bjlcs-tech.com/article/226.html. (accessed on 30 December 2022).
[26] C. Beckhoff, D. Koch and J. Torresen, "The Xilinx Design Language (XDL): Tutorial and use cases," in *Proc. ReCoSoC 2011*, Montpellier, Francepp, pp. 1–8, 2011.
[27] D. Choo, M. Soos, K. M. A. Chai and K. S. Meel, "Bosphorus: Bridging ANF and CNF solvers," in *Proc. DATE 2019*, Florence, Italy, pp. 468–473, 2019.
[28] Internet, "Cryptominisat 5.8.0," [Online]. Available: https://github.com/msoos/cryptominisat/releases. (accessed on 31 December 2022).
[29] N. Vafaei, S. Saha, N. Bagheri and D. Mukhopadhyay, "Fault attack on SKINNY cipher," *Journal of Hardware and Systems Security*, vol. 4, no. 4, pp. 277–296, 2020. https://doi.org/10.1007/s41635-020-00103-z
[30] G. Xu, F. Zhang, B. Yang, X. Zhao, W. He *et al.,* "Pushing the limit of PFA: Enhanced persistent fault analysis on block ciphers," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 6, pp. 1102–1116, 2021. https://doi.org/10.1109/TCAD.2020.3048280
[31] X. Fang, H. Zhang, D. Wang, H. Yan, F. Fan *et al.,* "Algebraic persistent fault analysis of SKINNY_64 based on S_Box decomposition," *Entropy*, vol. 24, no. 11, pp. 1508, 2022. https://doi.org/10.3390/e24111508