A General Hardware Trojan Technique Targeted on Lightweight Cryptography with Bit-Serial Structure

Yijun Yang[1,2], Liji Wu[1,2(✉)], Ye Yuan[1,2], and Xiangmin Zhang[1,2]

[1] Institute of Microelectronics, Tsinghua University, Beijing 100084, China
lijiwu@mail.tsinghua.edu.cn
[2] Tsinghua National Laboratory for Information Science and Technology, Tsinghua University, Beijing 100084, China

Abstract. Lightweight ciphers have a wide range of applications such as IoT, anti-counterfeiting labels, and passive RFID, which drawing loads of attention in recent years. Obviously, the most significant metric of lightweight cryptography is the area. To implement the smallest area lightweight cipher, to the best of our knowledge, the bit-serial structure is used. However, the bit-serial provides a possible access for the small area occupied hardware Trojan to steal key information at the same time, which makes lightweight ciphers vulnerable to Trojan attack. In this paper, we introduce a general hardware Trojan scheme targeted on ciphers based on bit-serial technique, which can leakage secret key through only one flip-flop at least with ease. This paper will alert cryptographic designers not implement the ciphers only based on design specifications, without taking hardware security into account.

Keywords: Hardware Trojan · Hardware security · Lightweight cryptography · Bit-serial

1 Introduction

The integrated circuit (IC) foundries are now wilder-distributed. As the perplexity of third-party companies and vendors involved increased, hardware security is facing more serious safety threats from various malicious attacks and modifications in terms of Hardware Trojans. A Trojan is usually embedded into a very small part of the circuit from all design phases, such as layout level, gate level and register-transfer level (RTL) and any parts of an untrusted IC supply chain. Additional, avoiding been detected by circuit test technique easily, the hardware Trojan may always-on or be activated by hard conditions or rare signals. Due to the concealment of those malicious circuits, the detection of hardware Trojan is a tricky issue [12–17].

Subsequent paragraphs, however, are indented.

A directly common idea to hunt for hardware Trojan is to relay on the very-large-scale integration (VLSI) test architectures, whose main idea is to run different test patterns and observe the output and behavior of the circuit, since circuit faults are similar to hardware Trojans in some degree. However, this VLSI testing scheme can

J. Li et al. (Eds.): SPNCE 2019, LNICST 284, pp. 647–655, 2019.
https://doi.org/10.1007/978-3-030-21373-2_54

work well for defects detection but not for ingenious hardware Trojan. The reasons for that are concluded as follow: (1) hardware Trojan has no uniform Trojan models as defects' fault models; (2) to enhance target circuit's testability i.e. controllability together with observability, structured Design for Test (DFT) are widely used to control and observe the internal states of circuits, whereas DFT itself can induce potential safety hazard [3]. In addition, traditional automatic test pattern generation (ATPG) algorithm either general Random testing strategy are not effective for Trojan, especially for sequential Trojan activation and detection, cause the sequential testing is extremely time- consuming with low fault coverage rate [8].

In general, Trojan designers may pursuit Trojan area as small as possible, on one hand, the actual spare space of the target chip is always limited, on the other hand, a smaller Trojan inducing weaker side-channel physical parameter information, such as power, time delay, electromagnetic emission and so forth [4], can avoid been detected by side-channel analysis (SCA) based hardware Trojan detection method. Since SCA based detection may be strongly influenced by fabrication variations or measurement errors i.e. sensitive to noise and errors.

In order to cause damage influence leakage key information, hardware Trojan trends to attack all sorts of crypto hardware, especially lightweight ciphers, which are universally employed in both commercial and military applications. For instance, Internet of Things (IoT), which will have 4 billion connected users, anti-counterfeiting, RFID tags, sensor networks, distributed control systems, etc. are increasingly providing convenience and service for our daily life [1]. And those successful working smart devices aforementioned highly depend on lightweight ciphers to provide secure and private communication. Therefore, the research on lightweight cipher's hardware security issue is very valuable and necessary [5, 6].

Considering the requirements for both low cost and security application background, lightweight ciphers have already obtained more attention in the past few years. In general, a lightweight cipher contains a single encryption function or both encryption and decryption function based on either Feistel structure or Substitution-Permutation Network (SPN) to ensure the security [9]. The significant criteria for lightweight cipher contain reliability, cost i.e. chip area, and throughput [9]. A natural idea to reduce area is to limit the data path. In 2017, Jean, Moradi et al. propose the first strategy called bit-sliding, with which the authors first implement the smallest implementation of AES-128 to date [9]. Nevertheless, bit-sliding structure aforementioned can be applied to other lightweight cryptography like PRESENT and SKINNY, without loss of its generality [9]. However, the hardware aspect security issue of bit-sliding was overlooked by its authors. Note that our hardware Trojan technique is just targeted on this state of art structure, revealing its potential hardware security weakness.

2 Preliminaries

In this section, we first use AES hardware implement as an example cipher circuit to clarify the difficulties encountered by hardware Trojan insertion. And then we cover the core idea of bit-sliding techniques designed for state-of-art lightweight ciphers. Next, a

few main metrics of a practical hardware Trojan are presented so as to address the craftiness of our hardware Trojan.

2.1 The Bit-Serial Implementation Strategy

In general, to reach small area occupation desire, cipher designers trend to cut the data path, but that is really tough to make the data path smaller than the Sbox size. The bit-sliding strategy that we mentioned here, make the extremely small bit-serial Application Specific Integrated Circuit (ASIC) implementations of SPN based ciphers come true and the strategy could reduce the data path to a single bit, in the case throughout allows.

The conducting idea of the bit-sliding technique is to serially implement the SPN-based cryptographic algorithms instead of parallel implementing in order to decrease the data path to only one bit and replace scan flip-flop by regular flip-flop, since the scan flip-flop is more about 25% area expensive than regular flip-flop. Here, we just recall the 1-bit-serial implementation, but this strategy also supports other data-path like 2-bit or 4-bit.

Moreover, the overview of the bit-serial data path architecture for AES-128 is shown in Fig. 1 [9], in which the white block indicates regular flip-flop, the scan flip-flop is highlighted by green, black arrows refer to data flow direction and the blue ones represent shift direction during ShiftRows operations. As we know, AES contains KeyExpansion, SubBytes, MixColumns, ShiftRows, and AddRoundKey steps [11]. Under bit-serial architecture, KeyExpansion, AddRoundKey, ShiftRows together with plaintext input step are performed in a bit serial style. In [9], the authors introduce the details of the timing control of the circuit, so for the sake of space, we will not present them here. Note that both the serial form of KeyExpansion and AddRoundKey step bring hidden dangers to hardware security, and our Trojan is aimed the KeyExpansion step exactly to steal the secret key.

2.2 The Bit-Serial Implementation Strategy

First, it should be remarked that a hardware Trojan usually consists of two parts, trigger, and payload [15]. The payload is the action or the damage which a hardware Trojan will do when it is activated [16]. And the trigger refers to the singles or the special condition that could activate the hardware Trojan [16].

In order to hide from being discovered by detection algorithms and achieve damages, a well-designed hardware Trojan should have the following metrics.

Controllability. This characteristic is used to measure whether a hardware Trojan is able to be activated at a desired time and at a presupposed position. From the Trojan designer point of view, we want the Trojan has good controllability.

Concealment. We hope that the insertion of hardware Trojan will make as small as possible changes to the host circuit, no matter at the functional level or back-end layout level. In this case, a hardware Trojan with good concealment can avoid being detected by logic test algorithms, SCA-based detection algorithms easier.

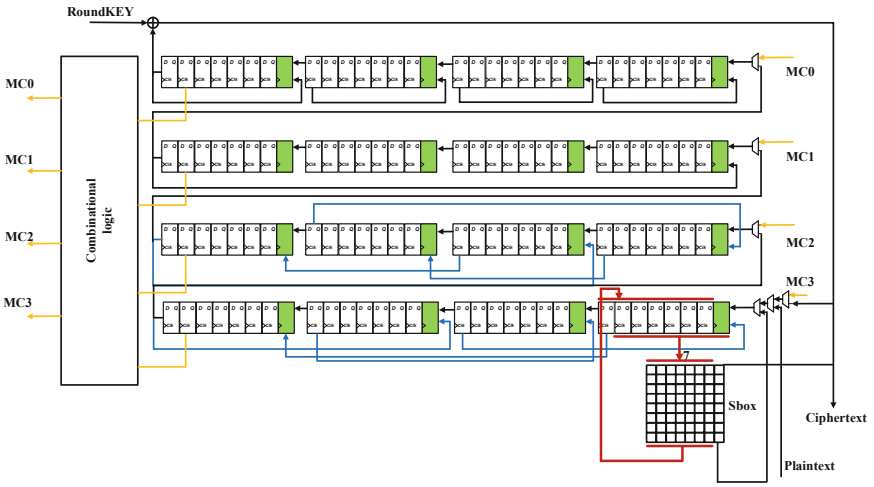

Fig. 1. The data path architecture based on bit-serial technique [9]. (Color figure online)

Rare Trigger Condition. If the trigger condition of the hardware Trojan is too easy to satisfy, then the hardware Trojan is more likely to be found during functional test or ATPG based logic test as we have mentioned in the Introduction section. In this regard, to make the hardware Trojan better hidden, the selection of trigger signals and timing control are crucial issues.

Destructiveness. The implantation of hardware Trojans has special purposes, which are often destructive. Generally, the destructiveness of hardware Trojans can be summarized as the following three aspects [7].

(1) Stealing confidential information or sensitive internal signals to attackers.
(2) Disturbing or interrupting the normal function of the chip, and causing errors during the execution.
(3) Maliciously leaking battery energy to shorten system life or even havocking the whole system directly or indirectly.

Taking into account the malicious nature of hardware Trojan, a powerful hardware Trojan may have as much damage as possible.

It should be noticed that the above four hardware Trojan metrics are often contradictory, and the Trojan designer should make a trade-off according to the actual situation.

3 Trojan Design

Our Trojan design technique is a general strategy targeted on bit-serial structure lightweight ciphers. And the objective of the hardware Trojan is the task of stealing the RoundKey or the secret key directly of the lightweight cryptographic algorithm in the KeyExpansion circuit bit by bit. It is noteworthy that, using the RoundKey, the attacker

can easily obtain the secret key of the lightweight cryptography [11]. Both the trigger theory and the payload implementation details are presented, respectively.

3.1 Mechanism of Trojan Payload

To explain clearly we use the bit-sliding architecture proposed in [9] to illustrate our Trojan approach. The overview architecture of our Trojan scheme is shown in Fig. 2 [9], where the host circuit is the KeyExpansion block, and the Trojans are highlighted in red, the victim flip-flop is highlighted in yellow, and the green blocks indicate the scan flip-flop same to Fig. 1.

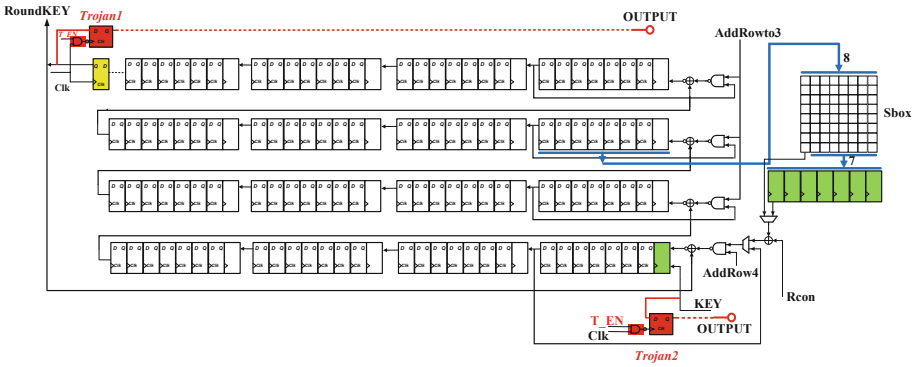

Fig. 2. The mechanism of hardware Trojan architecture based on [9].

First, we will focus on *Trojan1* shown in the top left of Fig. 2. The Trojan payload working process can perform as follows. In this case, during the first 128 clock cycle 128-bit plaintext is shifted in the registers which are illustrated in Fig. 1, at the same time the first round has been generated bit by bit serially per clock cycle based the bit-serial architecture as shown in Fig. 2. In order words, after 128 clock cycles, the first round RoundKey is already prepared and registered in the flip-flops. Next operation is AddRoundKey, which is carried out serially as well. Simultaneously, the Trojan enable signal T_EN is tied to 1, that is to say, the Trojan is triggered.

The main circuit structure of the hardware Trojan is a drop edge triggered D-type flip-flop, which takes the output of the victim flip-flop, highlighted in yellow, as its input and uses the ciphertext output port that is idle during the AddRoundKey encryption process as its output. Note that the area of the Trojan according to manufacturing process, and might be further reduced for the using of back-end layout design strategy Therefore, while the RoundKey is shifted bit by bit to complete the AddRoundKey operation, our hardware Trojan accurately propagate the RoundKey serially to the output port and leaks the key information to the attacker cat likely without disturbing the normal encryption operation.

Note that, it is easy to see that the hardware Trojan can be implanted on any of the 128 flip-flops and any round through the key path indiscriminately, as long as the

correct trigger signal *T_EN* is given in time. Therefore, from a practical point of view, the designer of hardware Trojan can implant the Trojan to a more reasonable place after considering the actual situation, whether space is enough or the routing resource is sufficient, of the physical layout. In this sense, our hardware Trojan scheme has some flexibility. Furthermore, Trojan2 is another example of our Trojan scheme. As shown in Fig. 2, Trojan2 use the secret Key signal as its input directly, so, in this case, to obtain the Key, the *T_EN* needs to be tied to logic 1 during the first 128 clock cycle.

We remark that our Trojan payload contains only one register and one NAND gate, and it does not affect the normal operation of the host circuit during either activated mode or unactivated mode, i.e. it has ideal concealment characteristics. In addition, the security of a cryptography algorithm totally depends on its secret key. However, once our Trojan is activated it has the ability to transfer the secret key to the attacker, which reflects the destructive ability.

3.2 The Mechanism of Trojan Payload

Before proceeding to discuss the details of the trigger mechanism of our Trojan technique, one significant term *"don't-care states"* should be mentioned ahead. The objective of the physical structure of a circuit is to meet the specification and to achieve the goal, the circuit should provide sufficient states, whereas the states a circuit can supply are usually more than its function needed. In other words, the circuit designer always unconsciously introduces some irrelevant states, so-called the "don't-care states". In this sense, those "don't-care states" is really difficult to be detected due to the lack of relevant research and automation test tools [12]. The activation of the "don't-care states" always along with illegal operations, which the customs rarely touch when the circuit system is normally executed. As a result, it would then be beneficial to attempt to use those "don't-care states" as the trigger signals for hardware Trojans, by which the rare trigger condition metric of the hardware Trojan can be dramatically improved.

The trigger signals of our hardware Trojan take advantages of "don't-care states" aforementioned. As for our Trojan trigger mechanism, the hardware Trojan is activated when the last three digits of the input 128-bit plaintext equal to "110" and the plaintext input port has high-level illegal signals within two clock cycles after the end of the plaintext stream input.

The circuit of trigger circuit could be realized in various forms, and Fig. 3 illustrates a simple implementation consisting of host circuit's control logic, to simplify the description, which is abstracted as an 8-bit counter highlighted in blue recording the encryption timing process, and the trigger cone. First, in the process of plaintext feeding, when the 126th clock cycle arrives, G1 outputs a high level that is connected to one input pin of G4. Simultaneously, the other pin of G4 is always monitoring the input port of plaintext, hence if the 127th plaintext bit equals to "1", G4 will supply a pulse through a 4–input OR gate, G8, to the 3bit-Counter illustrated in Fig. 3, and the counter will plus one. Subsequently, during the 128th clock cycle, once the plaintext bit is "1", G5 will carry out a plus to the 3bit-Counter, and the counter plus one again. In the same way, when the 128th plaintext bit equals to "0", the 3bit-Counter will reach "011".

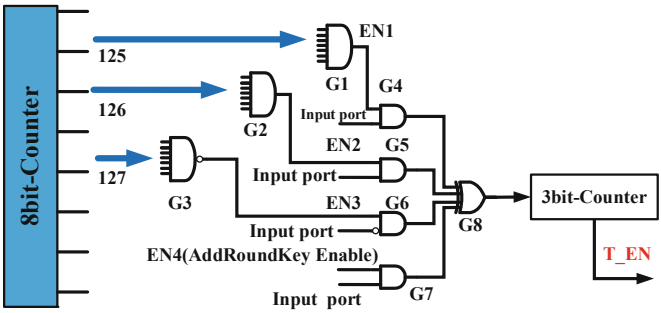

Fig. 3. The trigger mechanism architecture.

So far, the process of plaintext input is completed, and then the lightweight cipher will perform the step of AddRoundKey whose internal control signal EN4 acts as the enable signal of G7. To put it differently, in the next 128 clock cycle, the plaintext input IO port is idle. Therefore, in the first two clock cycles of the AddRoundKey operation, if two logic 1 are continuously fed to the plaintext input port, the 3-bit counter will reach to "101", next then the hardware Trojan enable signal T_EN will turn to logic 1 i.e. the Trojan payload will be activated, nor the 3-bit counter will be reset and waiting for next possible trigger change. Note that, in this example implementation, to obtain the two bits external off-chip trigger signal, we have to lose the first two digits of the RoundKey, but that does not matter since we can recover the complete 128-bit key by the exhaustive method. To emphasize that, since the trigger signals including internal signal and external signal and "don't-care states", and related to sequential circuit implementation, we say that our Trojan has rare trigger condition.

4 Verification

In order to verify the effectiveness of our hardware Trojan technique, we have embedded the hardware Trojan into the key path of AES-128 bit-serial architecture [9], which is an extremely small area lightweight cipher architecture, according to the mechanisms we presented in Sect. 3. It is to be noted that, due to intellectual property reasons, we did not get the implementation detail of its AES-128 control logic in [9], but this does not affect our analysis of the area of our Trojan, by synthesizing the Trojan as a separate module.

Taking the implantation of *Trojan1*, which is highlighted in red in Fig. 2, as an example. Plotted in Fig. 4, the input-output characteristic waveforms of the Trojan circuit displays that the payload part of *Trojan1* could steal the RoundKey bit by bit once it was triggered by the Trojan trigger enable signal *T_EN*. Furthermore, we synthesized Trojan payload circuit with the GSMC 180 nm Mixed-Signal 1P7M process, and from the cell report provided by the synthesizer, we can find that only one register, i.e. 8 GE (Gate Equivalent), is required to achieve the stealing function. In addition, using the data reported in [9], the smallest implementation of AES-128

encryption only 1560 GE, we could get the Trojan's payload area occupancy ratio is about 0.0051%.

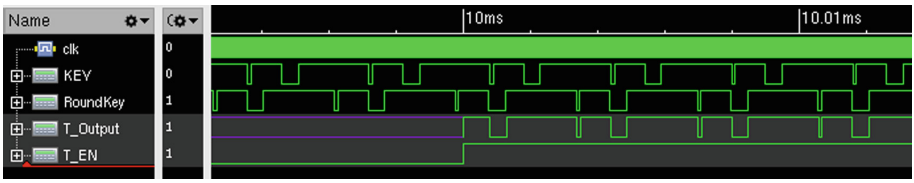

Fig. 4. The input-output characteristic waveforms of Trojan1.

In the light of the mechanism of trigger signals introduced in Sect. 3 plotted in Fig. 3, we have implemented the Trojan trigger cone circuit, and we have synthesized the cone circuit with the GSMC 180 nm Mixed-Signal 1P7M process as well. From the cell report supplied by the synthesizer the whole cone needs 49 GE, so its area occupancy ratio over the lightweight cipher is around 3.04%. Therefore, the total area occupancy ratio of our Trojan is 3.0451%. We have to emphasize that the area of the hardware Trojan is related to the process, and can be further reduced by the choice of the cells and superb layout design skills.

5 Conclusion and Discussion

In this paper, we introduce a general hardware Trojan design technique targeted on lightweight ciphers which are based on bit-serial structure such as SKINNY, PRESENT, AES and so forth. And four main metrics a sophisticated hardware Trojan should be equipped with have been discussed briefly. The function of this Trojan is to steal the RoundKey/Key serially and leak it to the attacker without causing any malfunction of the host circuit. The payload circuit of the hardware Trojan can be constructed by only one register and a NAND gate. As for the trigger signals of the hardware Trojan, we take advantage of the "don't-care states" of the host circuit, in the same time internal logic control wires are utilized to shrink the changes of the host circuit. To put it differently, we use both internal and external signals together to activate the hardware Trojan, which helps the Trojan avoiding been detected by the traditional functional or structural testing techniques. Moreover, though the examples in this paper are targeted on the 1-bit-serial scenario, this Trojan scheme can tackle 2-bit serial cipher easily as well. In order to demonstrate the effectiveness of this Trojan scheme, we inserted our Trojan into the key path of a lightweight cipher and the results are in line with expectations. We hope that through the introduction of this general hardware Trojan technique, we can alert the chip designer to the consideration of hardware security.

Acknowledgement. The authors would like to thank Professor Hu He for early discussions. And we would like to thank Amir Moradi et al. for sharing their SKINNY codes. This work was supported by the National Science and Technology Major Project of China under Grant 2017ZX01030301.

References

1. http://www.rcrwireless.com/20160628/0pinionlreality-check-50b-iot-devices-connected-2020-beyond-hype-reality-tag10
2. Da Rolt, J., Di Natale, G., Flottes, M., Rouzeyre, B.: New security threats against chips containing scan chain structures. In: 2011 IEEE International Symposium on Hardware-Oriented Security and Trust, San Diego CA, p. 110 (2011)
3. Rostami, M., Koushanfar, F., Rajendran, J., Karri, R.: Hardware security: threat models and metrics. In: Proceedings of the International Conference on Computer-Aided Design (ICCAD 2013), Piscataway, NJ, USA. IEEE Press, pp. 819–823 (2013)
4. Kelly, S., Zhang, X., Tehranipoor, M., et al.: Detecting hardware Trojans using on-chip sensors in an ASIC design. J. Electron. Test. **31**(1), 11–26 (2015)
5. Qu, G., Yuan, L.: Design things for the internet of things: an EDA perspective. In: Proceedings of the 2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD 2014), pp. 411–416. IEEE Press (2014)
6. Gao, M., Wang, Q., Arafin, M.T., Lyu, Y., Qu, G.: Approximate computing for low power and security in the Internet of Things. Computer **50**(6), 27–34 (2017)
7. Jin, Y., Kupp, N., Makris, Y.: Experiences in Hardware Trojan design and implementation. In: 2009 IEEE International Workshop on Hardware-Oriented Security and Trust, Francisco, CA, pp. 50–57 (2009)
8. Wang, X., Tehranipoor, M., Plusquellic, J.: Detecting malicious inclusions in secure hardware: challenges and solutions. In: 2008 IEEE International Workshop on Hardware-Oriented Security and Trust, Anaheim, CA, pp. 15–19 (2008)
9. Jean, J., Moradi, A., Peyrin, T., Sasdrich, P.: Bit-sliding: a generic technique for bit-serial implementations of SPN-based primitives. In: Fischer, W., Homma, N. (eds.) CHES 2017. LNCS, vol. 10529, pp. 687–707. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66787-4_33
10. Beierle, C., et al.: The `SKINNY` family of block ciphers and its low-latency variant `MANTIS`. In: Robshaw, Matthew, Katz, Jonathan (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 123–153. Springer, Heidelberg (2016). https://doi.org/10.1007/978-3-662-53008-5_5
11. Stallings, W.: Cryptography and Network Security: Principles and Practice. Pearson, Upper Saddle River (2017)
12. Lv, Y.Q., Zhou, Q., Cai, Y.C., et al.: Trusted integrated circuits: the problem and challenges. J. Comput. Sci. Technol. **29**(5), 918–928 (2014)
13. Zhang, J., Yuan, F., Xu, Q.: DeTrust: defeating hardware trust verification with stealthy implicitly-triggered hardware Trojans. In: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, pp. 153–166. ACM (2014)
14. Zhang, J., Xu, Q.: On hardware Trojan design and implementation at register-transfer level. In: 2013 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST), pp. 107–112. IEEE (2013)
15. Jin, Y., Makris, Y.: Hardware Trojan detection using path delay fingerprint. In: 2008 IEEE International Workshop on Hardware-Oriented Security and Trust HOST, pp. 51–57 (2008)
16. Wolff, F., Papachristou, C., Bhunia, S., et al.: Towards Trojan-free trusted ICs: problem analysis and detection scheme. In: Design, Automation and Test in Europe, pp. 1362–1365. IEEE (2008)
17. Wang, Q., Wang, A., Qu, G., Zhang, G.: New methods of template attack based on fault sensitivity analysis. IEEE Trans. Multi-Scale Comput. Syst. **3**(2), 113–123 (2017)