Journal of Electrotechnology, Electrical Engineering and Management (2017) Vol. 1, Number 1
Clausius Scientific Press, Canada

Clock Glitch Fault Injection Attacks on an FPGA AES Implementation

Yifei Qiao[1,a], Zhaojun Lu[2,b], Hailong Liu[3,c] and Zhenglin Liu[4,d]

[1,2,3,4]Huazhong University of Science and Technology, Wuhan, China

[a]1536464614@qq.com, [b]562576098@qq.com, [C]liuhilon@gmail.com, [d]liuzhenglin@hust.edu.cn

Keywords: AES, Fault injection attacks, Clock glitch, FPGA

Abstract. The Advanced Encryption Standard (AES) algorithm has been widely used to secure communication systems. However, the encryption algorithm is vulnerable to fault injection attacks and various attack methods have been studied. Some methods are just proposed in theory and have not been validated in practice. In this paper, we actualize a fault injection attack on an FPGA AES implementation. We propose a method to generate the highly accurate clock glitch to inject faults in the encryption process. We show that if the frequency of the clock glitch is carefully selected, only 6 faulty ciphertexts are necessary to discover the secret key.

1. Introduction

Cryptographic algorithms and cryptographic devices are being widely used to meet high security requirements. Unfortunately, the devices can leak secret information (such as the secret key) through side channels when performing the encryption. Fault attacks have proven to be an effective type of the side channel attack. Boneh, Demillo and Lipton [1] in 1997 firstly introduced the use of faults occurred during the execution of an encryption algorithm for attacking it and finding the secret key. Biham et. al. [2] proposed the concept of Differential Fault Analysis (DFA) on the Data Encryption Standard (DES). The DFA can discover the key from the analysis of one or multiple couples {correct ciphertext, faulty ciphertext}. AES is the substitute of DES and the fault attack on AES has been a popular research topic. Bloemer et. al. [3] proposed the DFA attack on the AES algorithm by changing a single bit during the first round of the encryption. The single-bit attack can recover the key in principle, but the strict requirement on the fault injection makes it practically infeasible. In [4], Dusart et. al. presented a more general fault model by injecting a single byte fault between the last two round. The multi-byte fault method is proposed by Moradi et. al. [5], however, this attack has not yet been performed in practice.

The fault injection techniques include: power supply voltage variation, injection of glitches in the clock signal, temperature variation, electromagnetic disturbances, or irradiation by a laser beam. The method of clock glitch is low-cost, easier to control the position of the faults, and will not destroy the target device. Endo et.al. [6] presented an on-chip clock-glitch generator with the accuracy of 0.17ns. Although the design of the generator is ingenious, it's not precise enough for our fault injection attacks.

In this paper, we propose a highly accurate clock-glitch generator. Further, we validate the fault attack model in [5] by attacking an AES hardware implementation on an Altera FPGA development board. We show that the key can be discovered using only 6 faulty ciphertexts, if the frequency of the clock glitch is set appropriately.

The rest of this paper is organized as follows: we briefly introduce the AES algorithm and the attack methods.The next section presents our implementation of AES, the clock-glitch generator and the experimental setup. Then, we show the experimental results. Finally, the work is concluded in the last section.

2. Fault Injection Attack on AES

The Description of AES Algorithm. The AES is applied to encrypt or decrypt data blocks of 128 bits by using secret key of 128, 192 or 256 bits. The total number of encryption rounds is decided by the key length. In this paper, the key length is 128 bits and the number of encryption rounds is 10. Except the last round, each round consists of four transformations: *SubBytes* (SB), *ShiftRows* (SR), *MixColumns* (MC), and *AddRoundKey* (ARK). Compared with other rounds, the last round does not execute the *MixColumns* function.

Attack Methods. The attack methods proposed in [5] inject faults at the input of *MixColumns* of the ninth round. The flow of the AES encryption from the last *MixColumns* function (in the ninth round) to the end is exhibited in figure 1. As presented in [5], we could consider each column of the *MixColumns* output in the ninth round independently. So in figure 1, we only consider the first column. Gray cells represent the faulty bytes and their locations in the *MixColumns* output state matrix are (1, 2, 3, 4). In the faulty ciphertext, the locations of faulty bytes become (1, 14, 11, 8). We gather multiple faulty ciphertexts using the same plaintext and key, but injecting different faults. Analysing the correct and faulty ciphertexts, we are able to calculate 4 bytes of the tenth round key (*K10*). Performing this methods for other three columns, all bytes of *K10* will be found. As a result, we can discover the secret key by knowing one round key [4]. According to the number of faulty bytes in one column, the faults we injected can be classified into two models: one byte undisturbed (*Model 1*); all of 4 bytes disturbed (*Model 2*). Obviously, the faults in figure 1 are under *Model 2*.

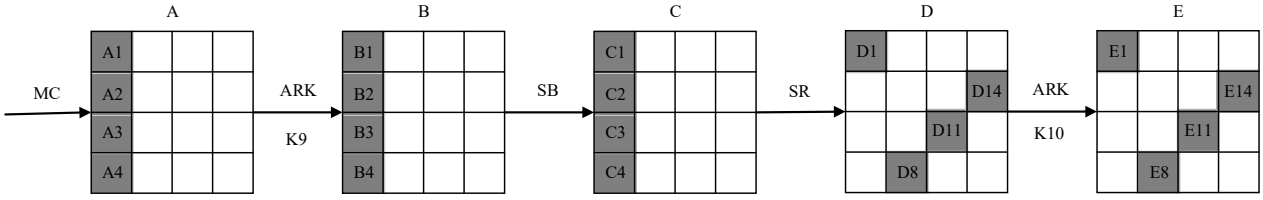

Fig.1 The encryption flow from the last MixColumns to the end

3. The Fault Injection Attack Setup

Implementation of AES. We implement the AES algorithm using Verilog HDL. The architecture of the implementation is given in figure 2. The control module gets the key and plaintext loaded when the 'ld' signal is asserted. When the encryption is completed, the 'done' signal will be asserted. The key expansion module generates each round key and provides it to the round function module. The round function module iterates 10 times to generate the ciphertext. The implementation performs the whole encryption scheme in 11 clock cycles(1 cycle for key expansion and 10 cycles for the encryption,). The design is downloaded on an Altera DE2-115 EP4CE115F29C7 development board.

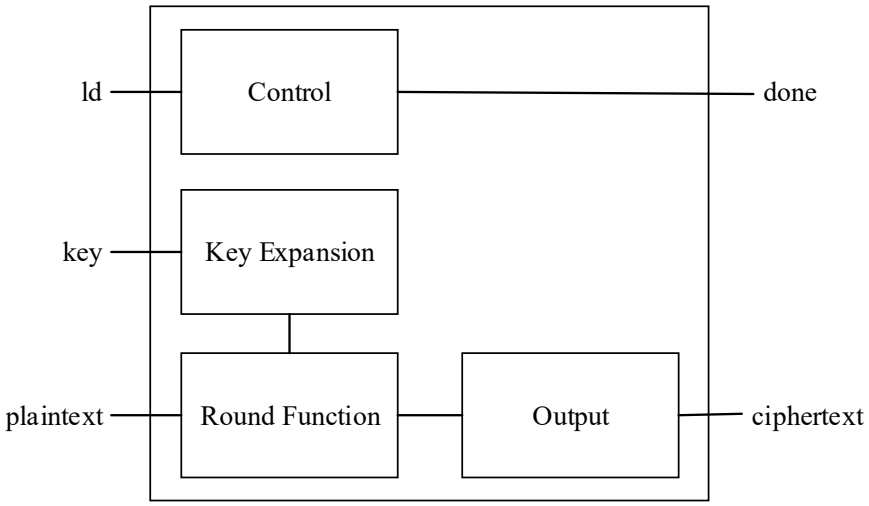

Fig. 2 Implementation of AES

Clock-glitch Generation. The attack methods inject faults in the ninth round of encryption algorithm, so we just need to supply a faster clock for the ninth round and normal clock for other rounds. In order to get the more accurate clock glitch(the faster clock) than that can be generated by using the on-chip PLL circuitry, we use an external clock generated by an Agilent E4438C 6GHz Waveform Generator. The external clock is divided by 4 to generate the normal clock and divided by 2 to generate the faster clock. The output clock of the clock-glitch generator is observed through an oscilloscope and exhibited in figure 3. Figure 3 shows that the fifth clock cycle is the clock glitch and is twice the frequency of other cycles. The position of the clock glitch in the encryption clock can be controlled and is set to the tenth clock cycle in our experiments(one cycle for key expansion before encryption). The clock-glitch generation module is also implemented on the DE2-115 development board.

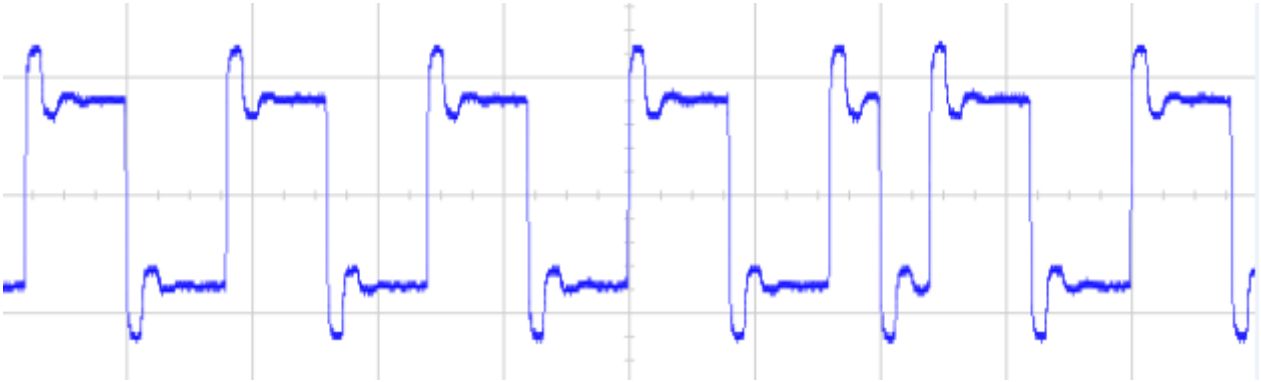

Fig. 3 The output of clock-glitch generation

Experimental Setup. The figure 4 shows our experimental setup. The waveform generator provides an external clock for the FPGA board. The FPGA board performs the clock-glitch generation and the encryption algorithm. Through JTAG UART, the PC(personal computer) downloads the design to the board, sends the plaintext and key to the board, and deals with the data received from the board.

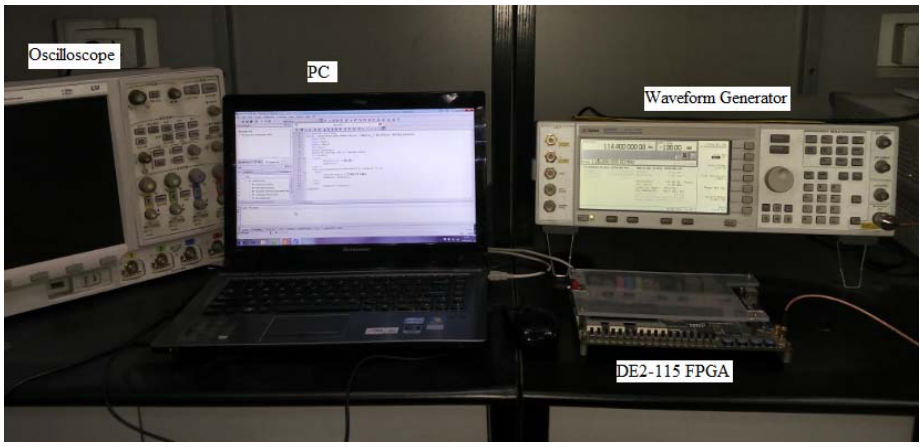

Fig. 4 Experimental setup

4. Experimental Results

The experimental results is given in figure 5. According to figure 5, the maximum operation frequency for our design is 113.7 MHz. There is no faulty state when the frequency of the clock glitch is 113.7 MHz. The frequency of the glitch is increased gradually in steps of 0.1 MHz by rising the frequency of the external clock in steps of 0.05MHz. Simultaneously, we use SignalTap II to record the faulty bytes in the first column of the state matrix at the beginning of the last round. We gather 500 ciphertexts caused by the same frequency of the clock glitch.

In figure 5, with the increase of the clock glitch frequency, the number of fault-free ciphertexts (no fault) is decreasing and the number of faulty ciphertexts under *Model 2* is increasing. From 113.7 MHz to 114.8 MHz, the number of ciphertexts under *Model 1* is increasing. After this, *Model 1* faults decrease and Model 2 faults start to dominate from 114.9 MHz.

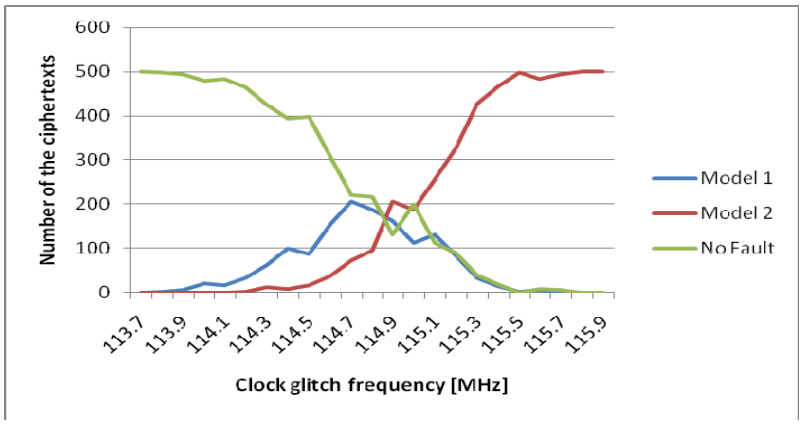

Fig. 5 The distribution of the encryption results

To recover the secret key, *Model 1* only needs 6 faulty ciphertexts, but *Model 2* needs approximately 1500 ciphertexts. In figure 5, *Model 1* faults dominate from 113.7 MHz to 114.5 MHz, during this period, *Model 2* faults hardly occur. We could gather ciphertexts during this period and only 6 faulty ciphertexts are needed to find the key.

5. Summary

This paper proposes a highly precise clock-glitch generator using an external clock. We performed the clock glitch fault injection attack on AES in practice. We injected the clock glitch in the ninth round of encryption. Through adjusting the external clock generator step by step, the maximum operation frequency of our design and the proper frequency period of the clock glitch for fault attacks have been found. When the frequency is set carefully, 6 faulty ciphertexts are enough to recover the

secret key. Experimental results show that the clock glitch fault attack presented before is low-cost, non-intrusive, and effective.

Acknowledgements

This work was financially supported by National Natural Science Foundation of China (61376026).

References

[1] Dj.M. Maric, P.F. Meier and S.K. Estreicher: Mater. Sci. Forum Vol. 83-87 (1992), p. 119

[1] D. Boneh, R.A. DeMillo, and R.J. Lipton: On the importance of checking cryptographic protocols for faults, *Proc. EUROCRYPT* (1997), p. 37–51.

[2] E. Biham and A. Shamir: Differential fault analysis of secret key cryptosystems. *Proc. CRYPTO* (1997), p. 513–525.

[3] J. Bloemer and J.P. Seifert: Fault based cryptanalysis of the Advanced Encryption Standard (AES), *Proc. FinancialCryptogr.* (2003), p. 162–181.

[4] P. Dusart, G. Letourneux, and O. Vivolo: Differential fault analysis on A.E.S., *Appl. Cryptogr. Netw. Security*, vol. 2846 (2003), p. 293–306.

[5] A. Moradi, M. T. M. Shalmani, and M. Salmasizadeh: A generalized method of differential fault attack against AES cryptosystem, *Proc. Int. WorkshopCryptogr. Hardware Embedded Syst.* (2006), p. 91–100.

[6] S. Endo, T. Sugawara, N. Homma, T. Aoki, and A. Satoh: An on-chip glitchy-clock generator for testing fault injection attacks, *Journal of Cryptographic Engineering* (2011), p. 265–270.