Power Analysis Attacks against FPGA Implementations of KLEIN

Shaohua Tang[1], Jianhao Wu[1], Weijian Li[1] and Zheng Gong[2,3]

[1] School of Computer Science and Engineering, South China University of Technology, China
csshtang@scut.edu.cn, jianhao.wu@foxmail.com, weijianlee@qq.com
[2] School of Computer Science, South China Normal University, Guangzhou, China
[3] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
cis.gong@gmail.com

Abstract. KLEIN is a family of block ciphers proposed by Zheng Gong *et al.* at RFIDSec 2011, and its lightweight features are suitable for resource-constrained devices. However, the original design of KLEIN does not consider the potential attacks by power analysis methods. This paper presents power analysis attacks against a FPGA implementation of KLEIN by the authors of KLEIN. The attacking strategy, attacking point and complexity of our attacks via power analysis against KLEIN are discussed in detail. Besides, the implementation of the attacks is also described, and the experimental data is given. A lot of attacking experiments are launched by this paper, and the experiments confirm that the success probability of our attacks is nearly 100%. Finally, a defensive countermeasure against our attacks is proposed.

1 Introduction

Inevitably there will be some physical information leakage during the operation of a cryptographic equipment. A method for gaining secret keys by collecting and analyzing the physical information leaked during the operation of a cryptographic equipment is called side channel attack (SCA). The side channel attacks mainly include timing attacks, power analysis attacks, and electromagnetic attacks, etc. In 1996, Kocher proposed a timing attack method [4], and then side channel attacks were widespread concerned in the field of cryptography[2,5,6,9]. The power analysis attack is one of the most important and effective side channel attack methods, which was proposed by Kocher *et al.* in 1998 [8].

KLEIN [3] is a lightweight block cipher algorithm proposed by Zheng Gong, *et al.* in 2011. This paper aims to launch power analysis attacks against FPGA implementations of KLEIN; firstly the KLEIN encryption process is analyzed in order to determine an appropriate point of attack, then a model is built based on the attacking point and an attack is successful implemented, and then the complexity of the attack is analyzed; finally, a defensive countermeasure is proposed to correspond to the attack method presented in the paper. The experiments show that our attack method is feasible and efficient.

The rest of this paper is organized as follows. Section 2 describes the KLEIN encryption process and introduces basic principles of the power analysis attacks. Section 3 firstly introduces the attacking strategy, the selection of the attacking point, then describes the implementation process of the attack, and finally analyzes the complexity of the attack and gives the experimental data. Section 4 proposes a defensive countermeasure against the attack method raised in this paper. Finally, Section 5 summarizes this paper.

2 Preliminaries

In this section, we briefly review basic operations of KLEIN and introduce basic principles of power analysis attacks.

2.1 KLEIN

The block length of input plaintext for KLEIN [3] is 64 bits, and the key sizes of KLEIN can be 64, 80, or 96 bits respectively. Corresponding to the 64/80/96-bit key length version of KLEIN, the encryption process iterates 12/16/20 rounds. One round of the encryption process of KLEIN is illustrated in Fig. 1, which consists of four steps, `AddRoundKey`, `SubNibbles`, `RotateNibbles`, and `MixNibbles` Step.

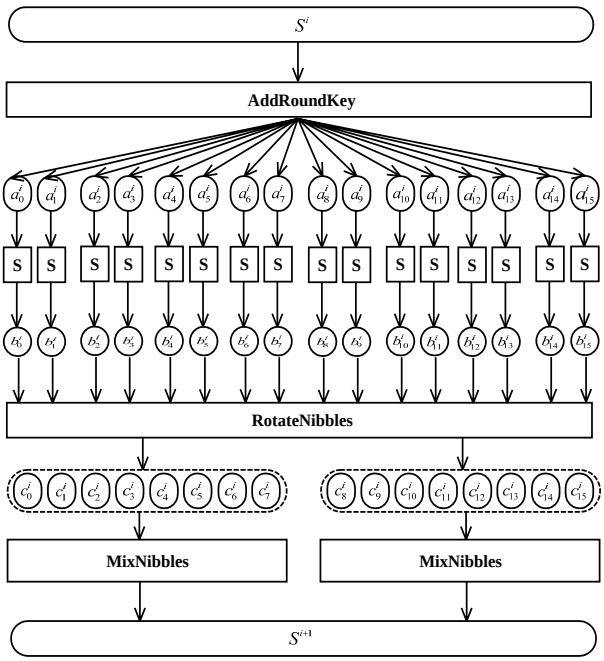

Fig. 1. One round of the encryption process of KLEIN.

Before the first round of the transformation, KLEIN initializes the ciphertext to equal to the plaintext. Then the following steps can be briefly described as follows.

AddRoundKey This step is simply a XOR (Exclusive OR) operation between the current ciphertext and the round key.

SubNibble In this step, the output of `AddRoundKey` is divided into 16 nibbles and each nibble is substituted according to the permutation table of S-box. The outputs of `SubNibble` are 16 nibbles.

RotateNibbles In this step, the outputs of `SubNibble` is rotated left for two bytes, that is, 4 nibbles.

MixNibbles This step merges the outputs of `RotateNibbles`, which are then divided into two tuples; and each tuple can be regarded as a polynomial to multiple the polynomial $c(x) = 03 \times x^3 + 01 \times x^2 + 01 \times x + 02$ modulos $x^4 + 1$. The results of these two tuples will be merged as the initiate ciphertext for the next round of transformation.

2.2 Power Analysis Attacks

ASIC or FPGA electronic devices, no matter cryptographic equipments or other electronic devices, are mostly based on logical circuits. CMOS technology is usually used by a logical circuit during its implementation. The majority of power that a CMOS transistor consumes is dynamic power consumption [7], which is caused when the internal signal or output signal changes. Thus, the power consumption is closely related to the changes of signal, i.e., there is a certain relationship between the power consumption and the data that a cryptographic device processes. Then by measuring the power consumption during its encryption (or decryption) process, an attacker is able to obtain a whole or part of a secret key of a cryptographic device via side channel attack methods, such as power analysis attacks.

In power analysis attacks, an attacker usually hypothesizes part of the secret key, and then calculates a hypothetical intermediate value during the encryption which is related to the secret key and some known data, such as plaintexts. By using the hypothetical keys one by one, the attacker maps the hypothetical intermediate value to hypothetical power consumption, and compares the hypothetical one with the actual power consumption generated by a cryptographic device during its encryption, the attacker can finally distinguish correct secret keys from wrong keys. Note that a set of power consumption measurements taken across a cryptographic operation is usually defined as a trace [8].

An attacker usually observes the dynamic power consumption caused by signal changes during an attack, which is the majority of power consumption. Generally, it is believed that the power consumption $P_{0\to1}$, which stands for a bit of signal changes from 0 to 1, is almost equal to $P_{1\to0}$, which presents a bit of signal changes from 1 to 0, even though they vary slightly indeed.

There are two well-known models to map the hypothetical intermediate value to the hypothetical power consumption, i.e., Hamming Weight Model and Hamming Distance Model. Assuming that $P_{0\to1}$ is equal to $P_{1\to0}$, then a Hamming weight model can be adopted; otherwise, a Hamming distance model can be used to seize the slightly distance between $P_{0\to1}$ and $P_{1\to0}$. Besides, other models can be considered to be extensions of Hamming weight model or Hamming distance model.

3 Attacking KLEIN on FPGA

In this section, attacking strategy, attacking point, and complexity of the attack against FPGA implementations of KLEIN are discussed in detail. Besides, the implementation of the attacks is also presented, and the experimental data is given.

3.1 Attacking Strategy

Based on different methods for analyzing the power consumption, the power analysis attacks can be divided into simple power analysis (SPA) attack and differential power analysis (DPA) attack. For the simple power analysis attack, the available trace are usually derived from one or a limited number of encryption processes. Due to the limitation of the available trace, the power consumption of the attacked device must be significantly related to the secret keys. A typical case is that the sequence of operation is directly dependent on the secret keys. While the differential power analysis attack obtains all or part of the secret keys by analyzing trace during a large number of different data block encryption processes. Accordingly, in differential power analysis attacks, usually only the relationship between the power consumption and the data to be processed is concerned, while the relationship between the power consumption and the sequence of operation is ignored, and the power consumption in the encryption process is deemed as a function of the data to be processed.

For FPGA implementations of KLEIN, the sequence of operation has nothing to do with the secret keys, and there is no significant relationship between the power consumption during the whole encryption process and the secret keys; accordingly, we adopted the differential power analysis method to attack KLEIN. In order to map certain hypothetical intermediate value during the KLEIN encryption process into a hypothetical power consumption value, a function is constructed to achieve the mapping. For the selection of the hypothetical intermediate value, the most important thing is that it should be related to a certain portion of the secret key, so that correct and wrong secret keys could be distinguished based on the correlation between the hypothetical power consumption value mapped by the hypothetical intermediate value and the actual power consumption value. And at the same time, since the number of the hypothetical secret keys required to be tried has exponential relation to the length of the relevant portion of the secret key, the portion of the secret key which is correlated to the hypothetical intermediate value should be as small as possible, so that it need not to try too many hypothetical secret keys to find the correct secret key. In addition, the hypothetical intermediate value may be related to some known data. For example, a plaintext can be used as a parameter of the function in known plaintext attacks.

3.2 Attacking Point

An output value from the first step of the first round transformation of the KLEIN encryption process (`AddRoundKey` step) is selected as a target to make a known plaintext attack. In order to determine whether the signal changes in one bit, each bit of the secret keys is needed to be known. Due to the correlation between the operation of the third step (`RotateNibbles` step) of each round of KLEIN and each bit of the secret key, if a known ciphertext attack is made, the target can only be selected as the input of the fourth step of the last round transformation; while if a known plaintext attack is made, the target can only be selected as the output of the first or second step of the first round transformation. The input of the fourth step (`MixNibbles` step) of the last round transformation is related to a half of the secret key (the first half part of the key or the second half part of the key), and if this step is selected as the target, the complexity of the attack will be very high. The second step (`SubNibble` step) of the first round transformation is a complex non-linear transformation, the design purpose and design principle of which is to resist the linear cryptanalysis and the differential cryptanalysis, the relationship between the output of the step and the secret key is difficult to be analyzed. Accordingly, the output of the first step, hereafter referred to as `R1_AddRoundKey`, of the first round transformation of KLEIN is selected as the target. Similarly, the first step of the second round transformation is referred to as `R2_AddRoundKey`.

For each round of iterative cryptographic algorithm, its input and output should be independent to each other. Since each bit of the plaintext is randomly generated, then after `R1_AddRoundKey`, the value of each bit of all intermediate values is random. Therefore, the output of `R2_AddRoundKey` and the output of `R1_AddRoundKey` are independent to each other, and each bit of the output of `R2_AddRoundKey` is random. That is, the probability that each bit of the output of `R2_AddRoundKey` is 0 or 1 is both 0.5, and the probability is independent of the output of `R1_AddRoundKey`.

Assume that in one of the encryption processes, Hamming weight of certain byte of the output of `R1_AddRoundKey` is HW. Since the probability that each bit of the output of `R2_AddRoundKey` is 0 or 1 is both 0.5, and the probability is independent of the output of `R1_AddRoundKey`, then when the output signal of `R2_AddRoundKey` is generated, the expectation of the number of bits of the signal which flips from 1 to 0 is $0.5 \times HW$, and the expectation of the number of bits of the signal which flips from 0 to 1 is $0.5 \times (8 - HW)$. As mentioned in Section 2, the power consumption caused by the signal change is only be focused, that is $P_{0\to1}$ which stands for the power consumption that is brought by a bit changing from 0 to 1, and $P_{1\to0}$ which represents the power consumption that is brought by a bit changing from 1 to 0. Therefore, the expectation of the power consumption caused by the signal change during the process where the output signal of `R2_AddRoundKey` is from being generated to stabilized is described by (1).

$$\begin{aligned} &0.5 \times P_{1\to0} \times HW + 0.5 \times P_{0\to1} \times (8 - HW) \\ &= 0.5 \times (P_{1\to0} - P_{0\to1}) \times HW + 4 \times P_{0\to1}. \end{aligned} \tag{1}$$

It can be seen that when the output signal of `R2_AddRoundKey` is generated, the expectation of the power consumption caused by signal change of some byte is linearly

related to Hamming weight of the byte in the output of `R1_AddRoundKey`. Therefore, in order to launch the power analysis attack against KLEIN, the Hamming weight model mentioned in Section 2 can be used to build a model for the output values of `R1_AddRoundKey` byte by byte.

3.3 Implementation of the Attack

We are going to attack KLEIN byte by byte, and by comparing the relationship between Hamming weight of the output of the first step of the first round transformation of KLEIN and the trace, judge whether the hypothetical secret key is right. In order to confirm the relationship, a correlation coefficient [1] is used to describe it. The attack process is divided into following three steps.

Step 1 Processing the trace. A random plaintext is generated by a program, and is encrypted with the FPGA implementation of KLEIN; and at the same time, the process of each round transformation of KLEIN is divided into 24 small periods, and the power consumption during each small period is recorded. A trace of KLEIN is illustrated in Fig. 2, which shows a set of power consumption for entire KLEIN operation.

It is needed to ensure that the process during which the output signal of the first step of the second round transformation of KLEIN changes from the output signal of the first step of the first round transformation and stabilizes to the output of the current transformation is within one small period. The smaller the small period is, the higher the likelihood of successful attack is. The trace collected is stored in a matrix, and the number of rows of the matrix is the number of the random plaintexts, the number of columns of the matrix is the number of the small periods during a full KLEIN encryption process. In the matrix, each row stores the trace of the respective small periods during one full KLEIN encryption process. Furthermore, the average values of each column of matrix are recorded in an array, in order to facilitate the following calculation of the correlation coefficients, and reduce repeated calculations. As each byte uses the same trace, this step is only needed to be done once.

Step 2 Calculating the hypothetical power consumption. For each byte, this step needs to be done once. Firstly, a matrix is constructed to store data of hypothetical power consumption, and the number of rows of this matrix are the number of random plaintexts, the number of columns is 2^8, since for different plaintexts and different secret keys, the power consumption is different. With the modeling method mentioned above, for every plaintext, the Hamming weight of the XOR (Exclusive OR) result of the current byte and the hypothetical secret keys from 0 to $2^8 - 1$ is calculated, and is stored in one row of the matrix, and the column number is the value of the hypothetical secret keys. Similarly, an array is used to record the average values of each column of the matrix mentioned above.

Step 3 Calculating correlation coefficients. For each byte, the step also needs to be done once. One column of the matrix which stores the trace is compared with each

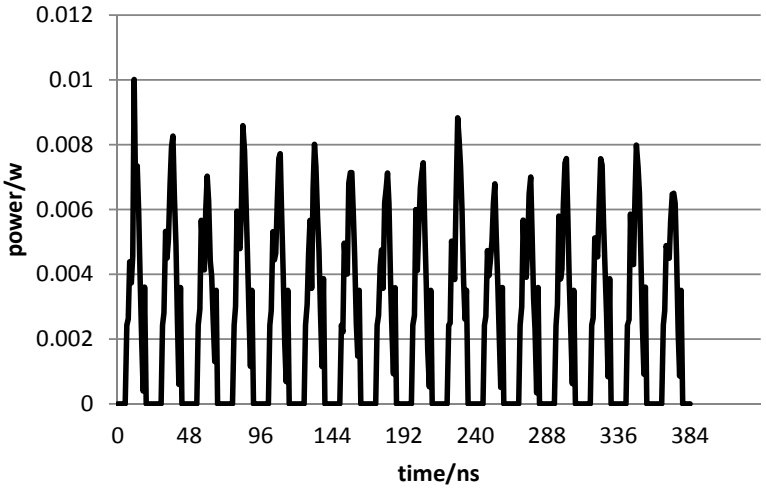

Fig. 2. Trace of KLEIN, a set of power consumption showing entire KLEIN operation.

column of the matrix which stores the hypothetical power consumption, and then the correlation coefficients thereof are calculated; during the calculation the array of the average values mentioned in the above two steps is used to reduce repeated calculation. Each column of the data of trace may be used for calculation for once, or alternatively, only the column of the small period, in which the process from the output signal of the first step of the second round transformation changing from the output signal of the first step of the first round transformation to its stabilizing to the output of the second round transformation happens, is used for calculating the correlation coefficient. In this paper, the latter is adopted, and the column number of the columns mentioned above is obtained by observation before attacks. For the correlation coefficients obtained by the above calculation, they are correspondingly stored in a matrix or a vector, wherein the column number of the matrix is equal to the column number of the trace matrix, and the row number of the matrix is equal to the column number of the hypothetical power consumption matrix, while the element number of the vector also corresponds to the column number of the hypothetical power consumption matrix. The value of the row number of the row which contains the maximum value of the correlation coefficient or the value of the vector number is the secret key obtained by the attack.

In order to facilitate analysis for the complexity of the attack time below, the relevant formula for calculating the correlation coefficient is given. Assumed that one column of the matrix which records the data of actual power consumption is denoted by t_j, one column of the matrix which records the data of hypothetical power consumption is denoted by h_i, the array which records the average values of the data of actual power consumption is denoted by avr_t, and the array which records the average values of data of the hypothetical power consumption is denoted by avr_h, then the calculation formula for the correlation coefficients is described by (2).

$$\frac{\sum_{\mathrm{d}=0}^{\mathrm{D}}\left(\mathrm{h}_{d,i}-avr\_h_i\right)*\left(\mathrm{t}_{d,j}-avr\_t_j\right)}{\sqrt{\sum_{\mathrm{d}=0}^{\mathrm{D}}\left(\mathrm{h}_{d,i}-avr\_h_i\right)^2*\sum_{\mathrm{d}=0}^{\mathrm{D}}\left(\mathrm{t}_{d,j}-avr\_t_j\right)^2}} \tag{2}$$

3.4 Complexity of the Attack

In the three steps mentioned in the previous sub-section, the first step is only executed once, the calculation time thereof is not related to the length of the secret key, and grows linearly with the desired number of random plaintexts. For the second step, the time of the matrix described thereof is calculated, which is related to the number of bytes, and its unit operation includes one byte XOR and calculation of Hamming weight of one byte. The number of bytes of the KLEIN secret key which is attacked in this paper is 8, so the unit operations need to be performed for $2^8 \times 8$ times. For the third step, since only one column is processed for data of the trace, the operation time thereof is equal to the time which is required by calculation of correlation coefficients for 2^8 times. It can be seen from the calculation formula for the correlation coefficients given above that the calculation time for the correlation coefficients grows linearly with the desired number of random plaintexts. The desired number of random plaintext is set as N, and then the time complexity of the third step is $2^8 \times N$. The third step need to be performed 8 times, so the overall time complexity of the third step is $8 \times 2^8 \times N$. In summary, the time complexity of attack method used in this paper is $8 \times 2^8 \times N$. If $N = 4000$, then the time complexity of attack method used in this paper is 2^{23}. The reasonable value of N will be referred to as below.

For storage complexity, the storage required by the first step also grows linearly with the desired number of plaintexts. In the second step, the storage is related to the number of bytes of the secret keys, and its complexity is $2^8 \times 8$. For the third step, its storage is linearly related to the number of hypothetical secret key value for each byte of the secret keys, which is 2^8. In summary, the storage complexity of this paper is the maximum value between $2^8 \times 8 = 2^{11}$ and N, i.e., $\max\left\{2^{11}, N\right\}$.

3.5 Experimental Results

Take the number of random plaintexts as $N = 10,000$. Through the encryption process mentioned above, 10,000 random plaintexts are generated and encrypted; then the trace during the encryption process is recorded, and the hypothetical power consumption is calculated by the hypothetical secret keys and the plaintexts; finally, the two is compared. For each byte, a vector for storing the correlation coefficients is obtained, and the number of the maximum number of the vector is the value of the secret keys obtained by the attack. In fact, the vector is preferably to be visualized, so as to better observe the effect of the correct or incorrect of the hypothetical secret keys on the correlation coefficient. Fig. 3 shows the correlation of the secret keys and the correlation coefficient in the vector of the correlation coefficients which are obtained by attacking the fifth byte, and the hypothetical secret keys are as abscissa. It can be observed from Fig. 3 that the peak in the graph corresponds to the correct secret key, whose value is equal to integer 114.

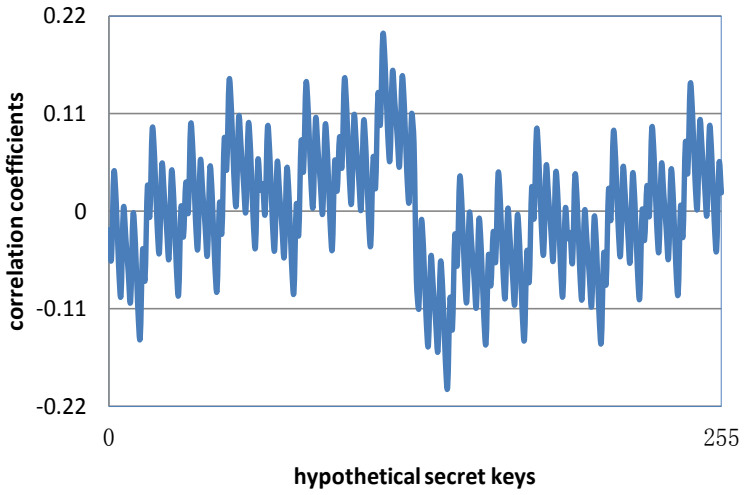

Fig. 3. Effect of the hypothetical secret keys on the correlation coefficients.

For the above-mentioned attack, there's another problem, i.e. the trace of how many times the encryption processes of the random plaintext is desired to attack successfully. N random plaintexts, where $0 \leq N \leq 10000$, are taken to implement the attacks separately, and the value of the correlation coefficient vector which is obtained by attacking the fifth byte with different number of the plaintexts is plotted with the number of the plaintexts N as abscissa. As shown in Fig. 4, the dashed line indicates the change curve of the correlation coefficients corresponding to the correct secret keys. It can be considered that when $N \geq 1000$, power analysis attack could achieve success.

For an attack, the secret key used by different plaintexts is fixed, but for different attacks, the secret keys may be different. In order to quantify the success probability of the attacks, random secret keys are generated to attack the trace obtained during 1000, 2000, 3000, 4000 and 5000 times encryption processes of the random plaintexts for 200 times separately, and for every attack, the secret keys used during the encryption process is randomly generated. The average number of correct bytes during the attack, the attack times of all bytes correct, and the rate of the attack times of all bytes correct are plotted, as illustrated in Table 1. It can be seen that when the number of the random plaintexts reaches 3000, the success probability of the attacks is not less than 95%. Certainly, in order to further improve the success probability of the attacks, the number of the random plaintexts can be taken $N = 4000$, and now the success probability of the attacks is nearly 100%.

4 Countermeasures

For defensive countermeasures for power analysis attacks, it is naturally concerned to make the power consumption during the device encryption process not depend on the intermediate value during operation process. Mask technology is a usual adopted method,

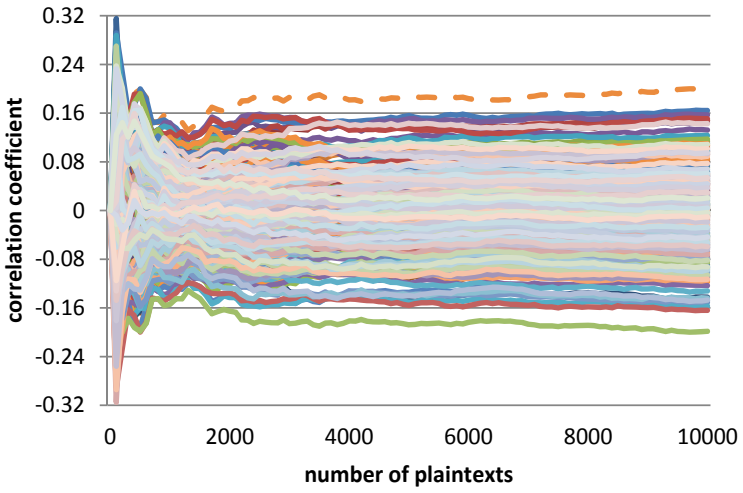

Fig. 4. Variation of the correlation coefficient of different number of plaintexts.

Table 1. The attack data of different number of plaintexts

Number of random plaintexts	Number of times of all bytes correct	Proportion of all bytes correct	Average of correct bytes number	Maximum of correct bytes number	Minimum of correct bytes number
1000	151	25.17%	6.6750	8	3
2000	501	83.50%	7.8200	8	5
3000	578	96.33%	7.9633	8	7
4000	597	99.50%	7.9950	8	7
5000	600	100.00%	8.0000	8	8

and it defends the power analysis attacks by randomizing the intermediate value during the operation process, and it can be designed based on the level of algorithm. For the KLEIN attack process mentioned herein, only the relationship of the power consumption and the Hamming weight of the output of the first step of the first round transformation is used, so the whole encryption process of KLEIN is not needed to be masked, only make sure that the output of the first step is not appear as an intermediate value signal during the encryption process.

The plaintext of the first step of each round of KLEIN is masked, i.e. to make it XOR with a random number to obtain a result, which is made XOR with the secret key, then the result obtained is used as the output of the first step. In order to restore the output of the second step, the second step is rewritten. The output of the second step is selected not only based on the output of the first step but also based on the random number of the mask. Due to the XOR result of the output of the first step and the random number is the original output of the first step, firstly a suitable transposition table is selected based on the random number, and then the output of the first step is used as an input to look up the table to obtain the output of the second step.

As an optimization of the method described above, the first step and the second step of each round of KLEIN are integrated, and the output of the two steps after integration is selected based on plaintexts and the secret keys. First a suitable transposition table is selected based on the plaintexts, and then the secret keys are as an input to look up table to obtain the output of the second step. Thus, lower resource consumption compared with that of the mask method can be obtained under the premise of the same guarantee of the output of the first step is not as the intermediate value signal. Of course, due to 2^8 transposition tables are also needed, the resource consumption certainly will be higher than that without defensive countermeasures.

5 Conclusion

This paper presents power analysis attacks against FPGA implementations of KLEIN, proposes a suitable attacking point of the power analysis attacks against KLEIN, builds a model thereof with Hamming weight model, and performs the attack successfully. When the number of the random plaintexts is 4000, the time complexity of the attack is nearly 2^{23}, the storage complexity is nearly 2^{12}, and the success probability of attack is nearly 100%.

Besides, a defensive countermeasure is proposed to correspond to the attack method presented in the paper. It is suggested to make sure that the output of the internal step is not appear as an intermediate value signal during the encryption process. However, the resource consumption in this way will certainly be higher than that without defensive countermeasures.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (Nos. U1135004, 61170080 and 61100201), and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2011), and Guangzhou Metropolitan Sci-

ence and Technology Planning Project under grant No. 2011J4300028, and High-level Talents Project of Guangdong Institutions of Higher Education (2012), and Guangdong Provincial Natural Science Foundation of under grant No. 9351064101000003.

References

1. Brier, E., Clavier, C., Olivier, F.: Correlation power analysis with a leakage model. In: Cryptographic Hardware and Embedded Systems - CHES 2004, Lecture Notes in Computer Science. vol. 3156, pp. 16–29. Springer-Verlag (2004)
2. Gandolfi, K., Mourtel, C., Olivier, F.: Electro magnetic analysis: Concrete results. In: Cryptographic Hardware and Embedded Systems - CHES, Lecture Notes in Computer Science. vol. 2162, pp. 251–261 (2001)
3. Gong, Z., Nikova, S., Law, Y.: KLEIN: a new family of lightweight block ciphers. In: RFID. Security and Privacy, Lecture Notes in Computer Science. vol. 7055, pp. 1–18. Springer-Verlag (2012)
4. Kocher: Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Advances in Cryptology, CRYPTO 96, Lecture Notes in Computer Science. vol. 1109, pp. 104–113. Springer-Verlag (1996)
5. Messerges, T.S., Dabbish, E.A., Sloan, R.H.: Power analysis attacks of modular exponentiation in smartcards. In: Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science. vol. 1717, 1999, pp. 144–157. Springer-Verlag (1999)
6. Ors, S., Gurkaynak, F., Oswald, E., Preneel, B.: Power-analysis attack on an ASIC AES implementation. In: Proceedings of ITCC. pp. 5–7 (2004)
7. Ors, S., Oswald, E., Preneel, B.: Power-analysis attacks on an FPGA: first experimental results. In: Cryptographic Hardware and Embedded Systems - CHES 2003. vol. 2279, pp. 35–50. Springer-Verlag (2003)
8. P.Kocher, J.Jaffe, B.Jun: Differential power analysis. In: Advances in Cryptology, CRYPTO 99, Lecture Notes in Computer Science. vol. 1666, pp. 388–397. Springer-Verlag (1999)
9. Quisquater, J.J., Samyde, D.: Electro magnetic analysis (EMA): Measures and countermeasures for smart cards. In: Smart Card Programming and Security, Lecture Notes in Computer Science. vol. 2140, pp. 200–210 (2001)