Power-Analysis Attack on an ASIC AES implementation

Sıddıka Berna Örs[1] Frank Gürkaynak[2] Elisabeth Oswald[3,4] Bart Preneel[1]
[1]Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC,
Kasteelpark Arenberg 10, B–3001 Leuven-Heverlee, Belgium
[2]Integrated Systems Laboratory ETH Zentrum, ETZ Gloriastrasse 35,
CH-8092 Zürich, Switzerland
[3]Institute for Applied Information Processing and Communications (IAIK),
TU Graz, Inffeldgasse 16a, A–8010 Graz, Austria
[4]A–SIT, Technologiebeobachtung,
Inffeldgasse 16a, A–8010 Graz, Austria
Email:{siddika.bernaors, bart.preneel}@esat.kuleuven.ac.be, kgf@iis.ee.ethz.ch, elisabeth.oswald@iaik.at

Abstract

The AES (Advanced Encryption Standard) is a new block cipher standard published by the US government in November 2001. As a consequence, there is a growing interest in efficient implementations of the AES. For many applications, these implementations need to be resistant against side channel attacks, that is, it should not be too easy to extract secret information from physical measurements on the device. This article presents the first results on the feasibility of power analysis attack against an AES hardware implementation. Our attack is targeted against an ASIC implementation of the AES developed by the ETH Zurich. We show how to build a reliable measurement setup and how to improve the correlation coefficients, i.e., *the signal to noise ratio for our measurements. Our approach is also the first step to link a behavior HDL simulator generated simulated power measurements to real power measurements.*
Keywords: *AES, power analysis attack*

1 Introduction

The block cipher Rijndael [5] has become the Advanced Encryption Standard (AES) [11] in November 2001. The AES will replace the aging DES algorithm [12] in a broad variety of applications. One can anticipate that AES in the next years will become a world-wide the facto standard with an even wider use than the DES.

During the AES selection process, the security of Rijndael was evaluated with respect to all types of attacks. While being resistant to the classical cryptanalytic methods, it turned out that so-called implementation attacks are a serious threat against naive implementations of the Rijndael algorithm. Implementation attacks refer to a new class of cryptanalytic methods, which are aimed against implementations of cryptographic primitives. Power analysis attacks are passive implementation attacks. Kocher et al. have shown they are very effective and relatively cheap to conduct in practice [9]. However, most scientific and public available literature discusses these attacks only for a theoretical view point, such as in [3]. This article compares the different AES candidates with respect to the vulnerability of their key scheduling routines to power analysis attacks. Only [4] and [10] report practical implementations of attacks. Both articles deal with software implementations on a smart card [4] or a micro-controller [10]. To our knowledge, there exists no publication on practical implementations of power analysis attacks on dedicated hardware implementations of the AES.

This article, demonstrates the feasibility of power analysis attacks against hardware implementations of the AES. Our attack is targeted against an ASIC implementation of the AES developed by the ETH Zurich.

The remainder of this article is organized as follows. In Section 2, we briefly introduce power analysis attacks including some mathematical background and a short discussion of the challenges of their practical implementation. In Section 3, we describe the ASIC implementation and the measurement setup. In Section 4, we discuss attacks on simulated measurement data. The same attacks are then performed successfully with real measurement data in Section 5. Based on these results we draw some conclusions in Section 6.

2 Power Analysis Attacks

Traditionally, the main task of cryptographic hardware is the acceleration of operations frequently used in cryptosystems or the acceleration of a complete cryptographic algorithm. In applications, hardware devices are also required to store secret or private keys securely. Hence, a cryptographic device must prevent the extraction and other sensitive information. Active attacks targeting the keys in cryptographic devices are commonly referred to as *tamper attacks*; they have long history in the field of cryptography [2].

Passive attacks were recognized in the cryptographic community as a major threat in 1996, when the first article about timing attacks [8] was published. In a passive attack, the adversary uses the standard functionality of the cryptographic device. The physical and/or electrical effects of the functionality on the device are then used for the attack. There are many different types of effects, such as operation time, power consumption, electromagnetic radiation, etc. If these effects unintentionally deliver information about the key which is used inside the device, then they deliver side-channel information and are called side-channels.

Nowadays, CMOS is by far the most commonly used technology to implement digital integrated circuits. The dominating factor for the power consumption of a CMOS gate is the dynamic power consumption [7]. Two types of power consumption leakage can be observed. The *transition count leakage* gives information about the number of changed bits, while the *Hamming weight leakage* is related to the number of 1 bits being processed simultaneously.

Two types of power analysis attacks are distinguished. In a simple power analysis (SPA) attack, an attacker uses the side-channel information from one measurement directly to determine (parts of) the secret key. In this article, we discuss differential power analysis (DPA) attacks. They use many measurements to filter out noise. While SPA exploits the relationship between the executed operations and the power leakage, DPA exploits the relationship between the processed data and the power leakage.

The first practical implementation of a power analysis attack on the DES was reported in [9]. Since then, some companies and universities have developed the skills to conduct these measurements in practice; these skills include knowledge about statistics, the properties of the attacked cryptographic algorithm, and the measurement setup.

Several countermeasures against SPA and DPA attacks have been proposed so far. For the AES, some rather efficient proposals have been published in [1, 6, 13].

2.1 Theoretical Background

In DPA, an attacker uses a so-called hypothetical model of the attacked device. The quality of this model is dependent on the knowledge of the attacker. The model is used to predict several values for the side-channel output of a device.

These predictions are compared to the real, measured side-channel output of the device. Comparisons are performed by applying statistical methods on the data. Among others, the most popular are the *distance-of-mean test* and the *correlation analysis*. We decided to use the correlation analysis in our attack. For the correlation analysis, the model predicts the amount of side-channel leakage for a certain moment of time in the execution. These predictions are correlated to the real side-channel output. This correlation can be measured using the Pearson correlation coefficient can be used. Let t_i denote the ith measurement data (i.e. the ith trace) and T the set of traces. Let p_i denote the prediction of the model for the ith trace and P the set of such predictions. Then we calculate

$$C(T,P) = \frac{E(T \cdot P) - E(T) \cdot E(P)}{\sqrt{Var(T) \cdot Var(P)}}. \qquad (1)$$

Here $E(T)$ denotes the expectation (average) trace of the set of traces T and $Var(T)$ denotes the variance of a set of traces T. If this correlation is high, it is usually assumed that the prediction of the model, and thus the key hypothesis, is correct.

2.2 Practical Challenges

When conducting power analysis attacks in practice, we need to deal with several technical difficulties. One of the most important issues is how to obtain *good*, *i.e.*, relatively noise free, measurements. The more noisy the obtained measurements are, the worse the statistical evaluation work and the more measurements are needed.

Another practical challenge is the complexity of the measurement setup. Such a setup typically consists of the attacked device, some monitoring tool (*i.e.* the scope) and some tools to operate the attacked device (for example a smart card reader or a chip tester). In addition to the hardware components, we need several software tools that handle the communication between the hardware devices.

Hence, if one performs such an attack in practice, one needs to be sure that, if the analysis fails, this is because there is not enough side-channel leakage and not because there is a bug in one of the components of the measurement setup. Therefore, the first step in evaluating a device against side-channel attacks is to simulate attacks. As they are virtually noise free, and they only involve some of the parts of the complete system, they allow to estimate how difficult a real attack will be.

3 Measurements on the AES chip

The AES operates on 128-bit data blocks and supports three key sizes (128, 192, and 256 bits). The encryption operation consists of four operations: *SubBytes*, *ShiftRows*, *MixColumns* and *AddRoundKey*. These four operations compose one encryption round. For 128-bit keys, the encryption operation starts with a single *AddRoundKey* operation followed by 9 identical encryption rounds. There is a slightly different final encryption round without the *MixColumns* operation.

3.1 Fastcore

Fastcore is an efficient ASIC realization of the AES algorithm in a standard 0.25 μm CMOS process with en/decryption rates in excess of 2 Gb/s.[1] Fastcore contains two separate datapaths for the encryption and decryption operations. Figure 1 shows a simple block diagram highlighting the encryption datapath structure of Fastcore. The encryption operation is performed on 128-bit values in parallel internally, but the external chip interface is limited to 16 bits for plaintext and ciphertext. The input and output buffers are used to store plaintext and ciphertext values and transfer them to/from the chip respectively. Each encryption round requires a round key, that is generated from the encryption key using a key schedule algorithm. The key schedule routine is implemented in the *Key Expansion Unit*. The round keys in Fastcore are generated on-the-fly, parallel to the encryption operation.

In Fastcore, the order of *SubBytes* and *ShiftRows* has been changed and the first *ShiftRows* operation has been moved to the initial *AddRoundKey* operation. The result is a functionally equivalent, but slightly different encryption round structure seen in Figure 1. This transformation allows a more efficient implementation in hardware. The last encryption round shares the *SubBytes* operation with the standard encryption round followed by an additional *AddRoundKey* operation.

3.2 Measurements

The measurement setup consists of an HP83000 test system to provide the chip with the required inputs such as clock, data, key, control, etc., signals and a Tektronix 784C sampling oscilloscope with a Tektronix CT-1 current probe to measure the supply current. Fastcore uses two separate power supplies, a 3.3V supply for I/O and a 2.5V supply for the core cells. Only the core power supply has been measured. In order to reduce switching noise, all measurements were averaged over 16 times.

[1] Fastcore was developed by Dominique Gasser and Franco Hug of the Integrated Systems Laboratory of the ETH Zurich.

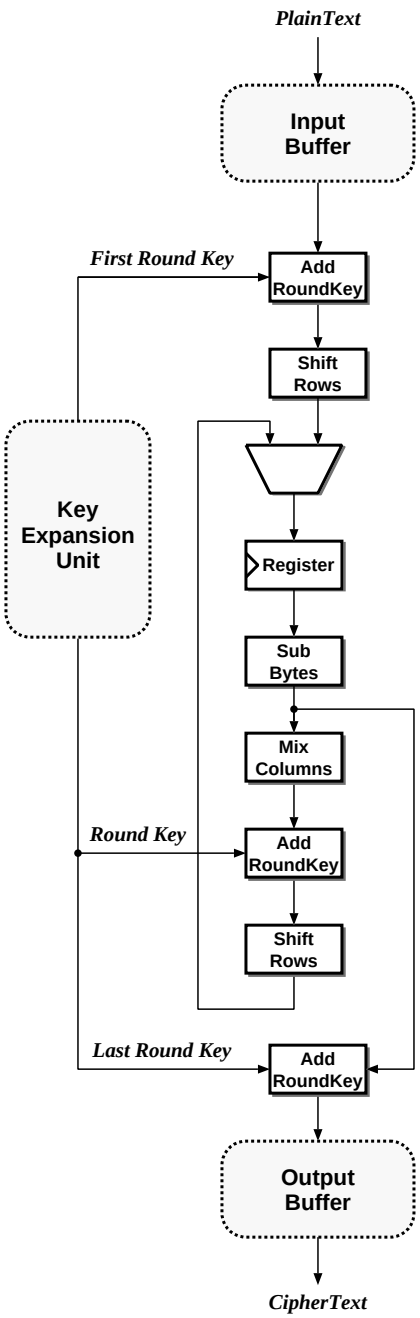

Figure 1. Block diagram highlighting the encryption path of the Fastcore crypto-chip

The HP83000 has been configured to perform an entire test run. Such a test run consists of initializing the cryptochip, loading the encryption key, sending 10 000 plaintexts and comparing the results with expected values. Because, the sampling oscilloscope can not sample the entire test run, the test system has been configured to generate a separate trigger signal at the beginning of each encryption operation. This signal has been used to sample and store the current multiple times for the clock cycles at and around the desired round of the encryption operation.

Fastcore contains a large number of functional blocks that can operate in parallel. Special care has been given to ensure that all unrelated blocks are either idle or compute the same results during the encryption operation. The decryption datapath is stalled and data I/O is not performed during encryption. The only other block that is active during an encryption operation is the key expansion unit. Since the same key is used throughout the measurement, the key expansion unit calculates exactly the same intermediate results for the same encryption round. The test vectors ensure that of the 2 758 flip-flops present in the Fastcore, only 128 flip-flops of the encryption round register have data dependent values and contribute to the difference in the power consumption.

4 A DPA attack using simulated data

The target for our DPA attack were the 8 most significant bits (MSBs) of the `Register` in Figure 1 after the initial key addition operation. Because the key used for this operation is the original key for encryption and the *Shift Rows* operation does not change the position of the 8 MSBs of the result of the *AddRoundKey* operation, we decided to predict the power consumption of during the storage of these MSBs in the `Register`.

We have tried our attack with simulated data before making real measurements. This approach enabled us to estimate the difficulty of a real attack, *i.e.*, an attack using real measurements. To predict the dynamic power consumption of the `Register`, behavioral HDL simulations of Fastcore were used. An advantage of this approach is that it allows to simulate attacks in an early stage of the design flow.

In the first step of this simulated attack, we have produced a so-called simulated power consumption file. For this purpose, we have chosen N random plaintexts and one fixed, but random key. After each encryption round (clock cycle), the simulator has written the total number of bit-changes between the previous and the current values of the `Register` to this file. Hence, the simulator has produced a file which contains a $N \times 10$ matrix ($N = 10\,000$), M_1, with values between 0 and 128.

In the second step, we have chosen the M MSBs of the `Register`. For the same plaintexts and key as in the first step, the simulator has calculated the total number of bit-changes between the previous and the current values of these M MSBs of the `Register` for the initial key addition. This result was stored in a file as an $N \times 1$ matrix, M_2, which contains values between 0 and M. In this particular experiment, we have chosen M as 8.

Then, we have calculated the correlation between all the columns of M_1 and M_2 as follows:

$$c_i = C(M_1(1:N,i), M_2), \tag{2}$$

where $i = 1, \ldots, 10$ and $M_1(1:N,i)$ denotes the ith column vector of matrix M_1.

In step 1 and step 2, the same plaintexts and the same key were used. The only difference between the two steps is the difference in the number of bits taken into account when counting the number of bit-changes. Hence, the values generated in step 2 are a prediction for the values calculated in the initial key addition of step 1. If the calculations are correct, the correlation coefficient of M_2 and the first column of M_1 (which corresponds to the initial key addition) must be significantly higher than the correlation coefficients of M_2 and all other rows of M_1. Figure 2 shows that this is indeed the case.

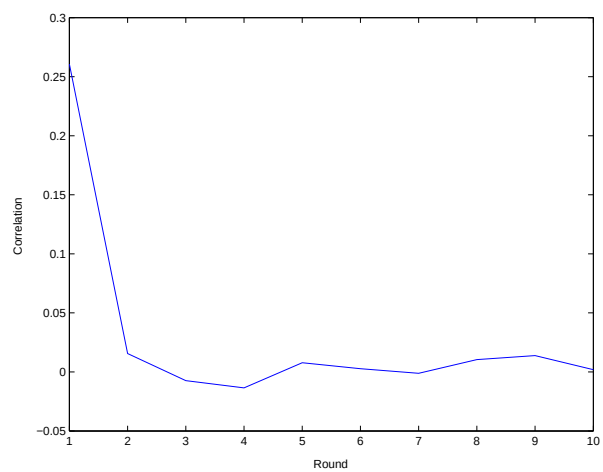

Figure 2. The correlation between all the columns of M_1 and M_2

In the third step, we have repeated the second step with a different value for the key. Hence, we have produced an output file containing the matrix M_3. As in step 2, we calculated the correlation coefficient of M_1 and M_3:

$$c_i = C(M_1(1:N,i), M_3), \tag{3}$$

where $i = 1, \ldots, 10$.

As we used another key to produce M_3, we expect no correlation between the columns of the matrices M_1 and

M_3. Figure 3 shows that this is indeed the case. We conclude that our model, which consist of using predictions of the behavior HDL simulation, makes correct predictions for the real behavior of the Fastcore chip.

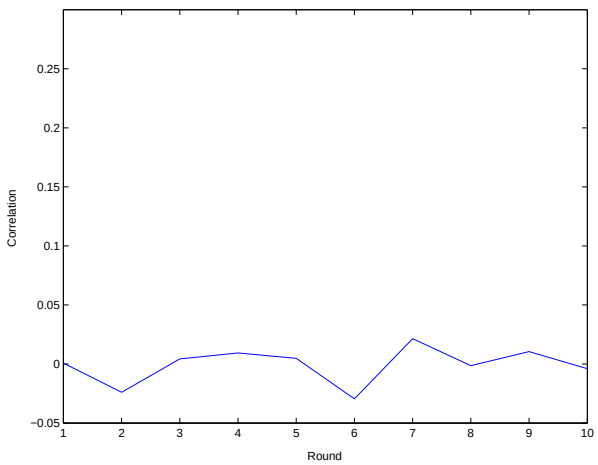

Figure 3. The correlation between all the columns of M_1 and M_3

In the fourth and last step, we have extended the experiments made in step 2 and step 3 in such a way that a full DPA attack on $L = 8$ bits was performed. Hence, we calculated an $N \times 2^L$ matrix M_4. Each column of the matrix M_4 contains the prediction for the bit-changes in the `Register` for a particular guess of the L attacked key bits of the initial key addition. Equation (4) shows how we can calculate the correlation coefficients between the predictions of all the possible keys and the first column of M_1:

$$c_i = C(M_1(1:N,1), M_4(1:N,i)), \tag{4}$$

where $i = 0, \ldots, 2^L - 1$.

From the previous steps we can expect that only one value, corresponding to the correct L key bits, leads to a high correlation coefficient. Figure 4 shows that this expectation is fulfilled.

We have already demonstrated that our attack setup works well together with our model. The only question that remains is how many measurements N are at least needed to determine the correct key. In order to determine this minimum, we calculated the correlation coefficient between M_1 and M_4 for different values of N:

$$c_{i,j} = C(M_1(1:i,1), M_4(1:i,j)), \tag{5}$$

where $i = 1, \ldots, 10\,000$ and $j = 0, \ldots, 2^L - 1$.

As shown in Figure 5, after approximately 400 plaintexts the right L MSBs can be distinguished from the wrong L MSBs. Hence, for the simulated attack, 400 measurements are sufficient to find the correct L MSBs of the key.

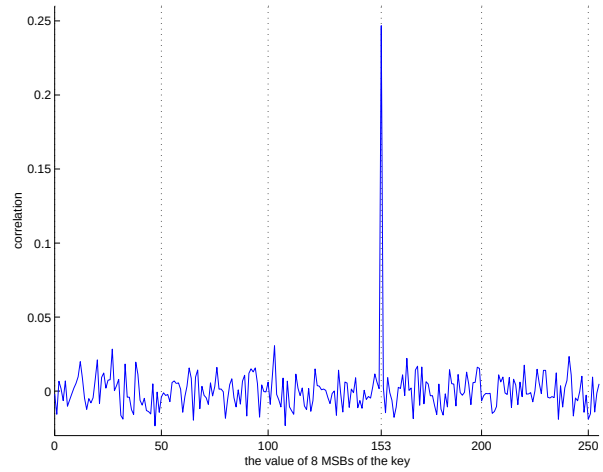

Figure 4. The correlations between the first column of M_1 and M_4

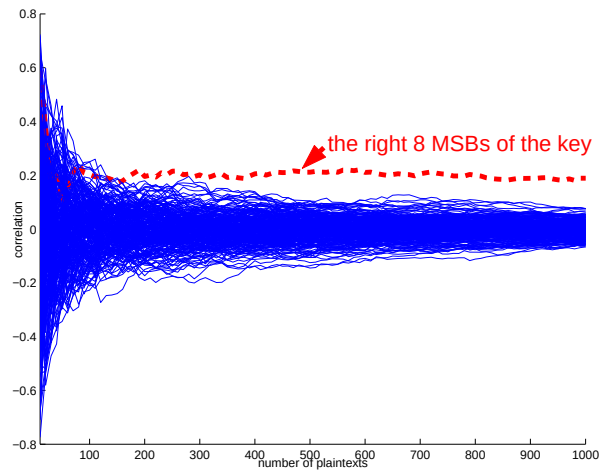

Figure 5. The correlation between the first column of M_1 and all the columns of M_4 for different number of measurements

5 A DPA attack using the measured data

In this section, we present the results of our DPA attacks on Fastcore using real, measured data. We have let Fastcore encrypt the same N plaintexts with the same key as used in the first step of Section 4. The initial key addition operation occurs during the first clock cycle. The result of this operation is written into the `Register` at the rising edge of the second clock cycle. Hence, we have measured the power consumption of Fastcore during the first two clock cycles of the encryption operation. The clock frequency ap-

plied to the chip was *2 MHz* and the sampling frequency of the oscilloscope was *1 GHz*. Hence, 500 samples were acquired per clock cycle. With these measurements, we have produced a $N \times 1\,000$ matrix, M_5. The power trace of one of these measurements, is shown in Figure 6.

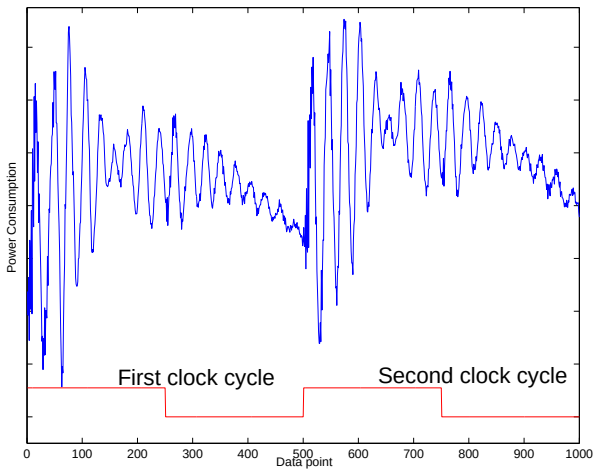

Figure 6. The power trace of the $50th$ measurement

In order to identify the correct L MSBs of the key we have used the correlation coefficient again. We have applied a pre-processing technique to reduce the noise in the acquired measurements and to reduce the amount of measurement data. The pre-processing technique essentially consists of averaging. We have calculated the mean values of the measurement data in the first and the second clock cycles as follows:

$$e_{i,j} = E(M_5(i, D*(j-1)+1 : D*j)), \qquad (6)$$

where $i = 1, \ldots, N$, $j = 1, 2$. D is the number of data points measured during one clock cycle. $M_5(i, D*(j-1)+1 : D*j)$ is the vector which is made of the ith row and the columns between $D*(j-1)+1$ and $D*j$ of M_5. As the mean value of a clock cycle consists of the DC component of the current (which consists of a lot of noise), we can remove parts of the noise by subtracting the mean values of two clock cycles. The matrix $M_6(i) = e_{i,2} - e_{i,1}$, where $i = 1, \ldots, N$, contains the result of this computation. We used these pre-processed measurements as input for our correlation analysis:

$$c_i = C(M_6, M_4(1 : N, i)), \qquad (7)$$

where $i = 0, \ldots, 2^L - 1$.

As shown in Figure 7 the highest correlation occurs at $i = 153$. This value corresponds to *0x99* which are the 8 MSBs of the key.

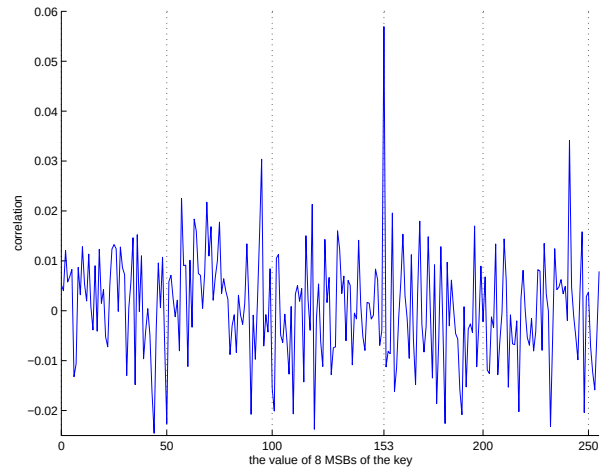

Figure 7. The correlation between the columns of M_4 and M_6

The critical path of Fastcore is around $7\,ns$. Thus, only the first data points of a measurement contain information which is directly related to the attacked operation. Hence, we decided to reduce the amount of data points for the pre-processing step.

In order to determine the minimal number of data points that contain relevant information, *i.e.*, information on the initial key addition, we have calculated the correlation coefficient between pre-processed measurement data (for a varying amount of data points in one clock cycle) and the column corresponding to the correct key of M_4:

$$M_7(i,j) = E(M_5(i, D+1 : D+j)) - E(M_5(i, D+1-j : D)), \qquad (8)$$

where $i = 1, \ldots, N$ and $j = 1, \ldots, D$ and

$$c_i = C(M_7(1 : N, i), M_4(1 : N, 153)), \qquad (9)$$

where $i = 1, \ldots, D$. Figure 8 shows that the correlation is highest when we use 50 data points around the rising edge of the second clock cycle.

Figure 9 depicts the correlation coefficients between all the columns of M_4 and the pre-processed data in column 50 of M_7. This figure shows clearly that the peak corresponding to the correct key becomes higher while the peaks corresponding to the incorrect key guesses stayes constant.

As in Section 4, N was taken as $10\,000$. However, we are interested in the smallest number of measurements that allow for a successful attack. In order to find the minimal number of measurements, we have calculated the following correlation coefficients:

$$c_{i,j} = C(M_7(1 : i, 50), M_4(1 : i, j)), \qquad (10)$$

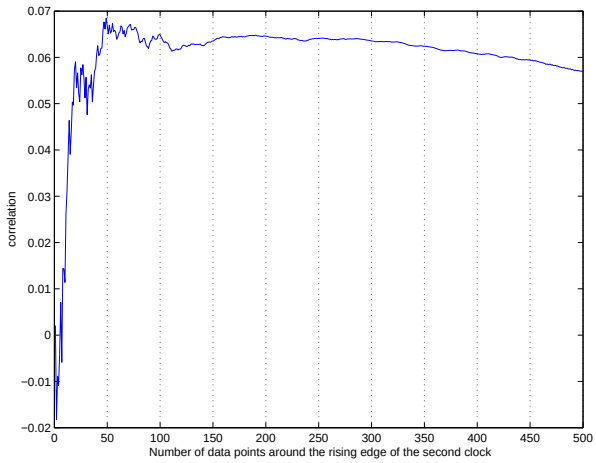

Figure 8. The correlation between $153th$ column of M_4 and all the columns of M_7

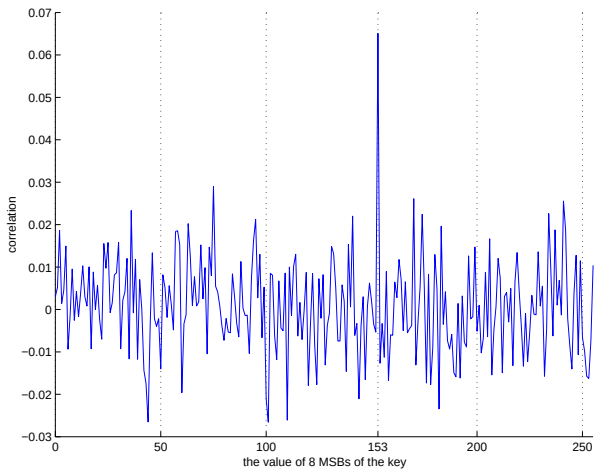

Figure 9. The correlation between all the columns of M_4 and the $50th$ column of M_7

where $i = 1, \ldots, N$ and $j = 0, \ldots, 2^L - 1$.

It is shown in Figure 10 that after approximately $4\,000$ measurements the correct and the wrong 8 MSBs of the key can be distinguished.

6 Conclusions and future work

We have presented the first public implementation of a DPA attack on a hardware implementation of the AES. We have shown how to build a reliable measurement setup and how to improve the correlation coefficients, *i.e.*, the signal to noise ratio for our measurements. Due to the results of

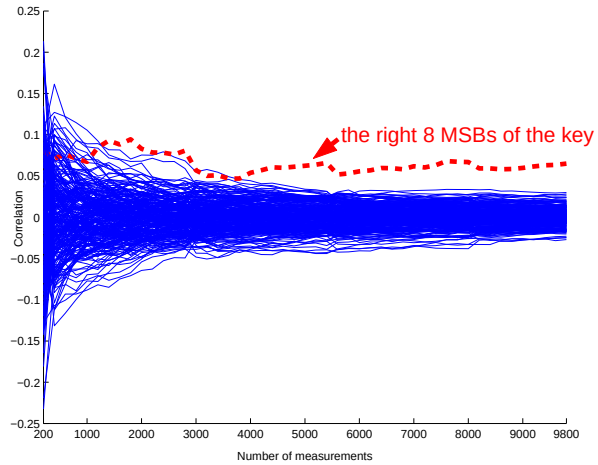

Figure 10. The correlation between all the columns of M_4 and $50th$ column of M_7 for different number of measurements

the simulated attack and the real attack, we conclude that the chip tester, which we used in our measurement setup, introduces a considerable amount of noise in our measurements (we needed 160 times more measurements in the real attack than in the simulated attack). Hence, we plan to develop a less noisy solution to conduct measurements in the near future.

Our approach forms a first step to link real and simulated power measurements. This is very important for designers of cryptographic hardware, as it allows them to estimate the vulnerability to power attacks in a very early stage of the design flow. This can bring important security and cost benefits.

References

[1] M.-L. Akkar and C. Giraud. An implementation of DES and AES, secure against some attacks. In Ç. K. Koç, D. Naccache, and C. Paar, editors, *Proceedings of 3rd International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, number 2162 in Lecture Notes in Computer Science, pages 309–318, Paris, France, May 13-16 2001. Springer-Verlag.

[2] R. Anderson and M. Kuhn. Tamper resistance - a cautionary note. In D. Tygar, editor, *Proceedings of the 2nd USENIX Workshop on Electronic Commerce*, pages 1–11, Oakland, CA, USA, November 18-21 1996.

[3] E. Biham and A. Shamir. Power analysis of the key scheduling of the AES candidates. In *Proceedings of the Second Advanced Encryption Standard (AES) Candidate Conference*, Rome, Italy, 1999.

[4] S. Chari, C. S. Jutla, J. R. Rao, and P. Rohatgi. Towards sound approaches to counteract power-analysis attacks. In

M. Wiener, editor, *Advances in Cryptology: Proceedings of CRYPTO'99*, number 1666 in Lecture Notes in Computer Science, pages 398–412, Santa Barbara, CA, USA, August 15-19 1999. Springer-Verlag.

[5] J. Daemen and V. Rijmen. *The design of Rijndael: AES–The Advanced Encryption Standard*. Springer-Verlag, 2002.

[6] J. D. Golic and C. Tymen. Multiplicative masking and power anaylsis of AES. In B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, editors, *Proceedings of 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, number 2535 in Lecture Notes in Computer Science, pages 198–212, Redwood Shores, CA, USA, August 13-15 2002. Springer-Verlag.

[7] S.-M. Kang and Y. Leblebici. *CMOS Digital Integrated Circuits: Analysis and Design*. McGraw Hill, 2002.

[8] P. Kocher. Timing attacks on implementations of Diffie-Hellman, RSA, DSS and other systems. In N. Koblitz, editor, *Advances in Cryptology: Proceedings of CRYPTO'96*, number 1109 in Lecture Notes in Computer Science, pages 104–113, Santa Barbara, CA, USA, August 18-22 1996. Springer-Verlag.

[9] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In M. Wiener, editor, *Advances in Cryptology: Proceedings of CRYPTO'99*, number 1666 in Lecture Notes in Computer Science, pages 388–397, Santa Barbara, CA, USA, August 15-19 1999. Springer-Verlag.

[10] S. Mangard. A simple power-analysis attack (SPA) attack on implementations of the AES key expansion. In P. J. Lee and C. H. Lim, editors, *Proceedings of 5th International Conference on Information Security and Crytography (ICISC)*, number 2587 in Lecture Notes in Computer Science, pages 343–358, Seoul, Korea, November 2002. Springer-Verlag.

[11] National Institute of Standards and Technology. FIPS 197: Advanced Encryption Standard, November 2001.

[12] National Institute of Standards and Technology. FIPS 46-3: Data Encryption Standard, October reaffirmed 1999.

[13] E. Trichina, D. De Seta, and L. Germani. Simplified adaptive multiplicative masking for AES. In B. S. Kaliski Jr., Ç. K. Koç, and C. Paar, editors, *Proceedings of 4th International Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, number 2535 in Lecture Notes in Computer Science, pages 187–197, Redwood Shores, CA, USA, August 13-15 2002. Springer-Verlag.