Improving DPA resistance of Quasi Delay Insensitive Circuits using randomly time-shifted Acknowledgment Signals

F. Bouesse, M. Renaudin, G. Sicard
TIMA lAboratory, Concurent Integrated Systems Group
26 av. Félix Viallet, 38031 Grenoble Cedex
fraidy.bouesse@imag.fr

Abstract. The purpose of this paper is to propose a design technique for improving the resistance of the Quasi Delay Insensitive (QDI) Asynchronous logic against Differential Power Analysis Attacks. This countermeasure exploits the properties of the QDI circuit acknowledgement signals to introduce temporal variations so as to randomly desynchronize the data processing times. The efficiency of the countermeasure, in terms of DPA resistance, is formally presented and analyzed. Electrical simulations performed on a DES crypto-processor confirm the relevancy of the approach, showing a drastic reduction of the DPA peaks, thus increasing the complexity of a DPA attack on QDI asynchronous circuits.

1 Introduction and motivations

Nowadays, the possibilities offered by all recent powerful side-channel attacks to access to confidential information, constrain secure systems providers to develop new resistant systems against these attacks. Among these new hardware cryptanalysis attacks, there is the Differential Power Analysis (DPA) which is one of the most powerful and low cost attack. The main idea behind DPA is that there exists a correlation between data processed by the design and the observable power consumption. In 1998 Paul Kocher [1] demonstrated how this correlation can be exploited using statistical means to retrace secret key information.

It is in this context that the properties of Self-timed logic have been exploited in order to propose efficient counter-measures against DPA attacks [2][3].

All results from the analysis of Self-timed logic particularly the Quasi Delay Insensitive asynchronous logic demonstrated the potentiality of this type of logic to increase the chip's resistance [4][5].

However, paper [6] reported that, even if the QDI asynchronous logic increases the resistance of the chip, there still exists some residual sources of leakage that can be used to succeed the attack.

The objective of this paper is to make a DPA attack impossible or impracticable with standard equipment by increasing the complexity of the attack. For doing so, we introduce randomly time shifted (RTS) acknowledgment signals in the QDI asynchronous logic in order to add noise in chip's power consumption. Indeed, the use of a RTS acknowledgement signal in an asynchronous Quasi Delay Insensitive block enables us to desynchronize the data processing time, so as to compute the blocks' output channels at random times. As the DPA attack requires the signals to be synchronized with respect to a fixed time instant for data analysis [1][7], this desynchronization makes the DPA attack more difficult as it is proved in this paper.

We present in the first part of the paper (section 2), the properties of Quasi Delay Insensitive asynchronous logic, especially the properties of the acknowledgment signal. Section 3 first introduces the formal analysis of the DPA attack. It then presents the desynchronization technique based on RTS acknowledgement signals and formalizes its efficiency in terms of DPA resistance. Finally, sections 4 and 5 illustrate the technique using electrical simulations performed on the well known Data Encryption Standard (DES) architecture. Section 6 concludes the paper and gives some prospects.

2 Quasi Delay Insensitive Asynchronous logic: the acknowledgment signal

This section recalls the basic characteristics of an asynchronous circuit, particularly the rule of the acknowledgement signal in the QDI asynchronous logic.

Because this type of circuit does not have a global signal which samples the data at the same time, asynchronous circuits require a special protocol to perform a communication between its modules. The behavior of an asynchronous circuit is similar to a data-flow model. The asynchronous module, as described in figure 1 and which can actually be of any complexity, receives data from its input channels (request signal), processes them, and then sends the results through its output channels. Therefore, a module is activated when it senses the presence of incoming data. This point-to-point communication is realized with a protocol implemented in the module itself. Such protocols necessitate a bi-directional signaling between both modules (request and acknowledgement): it is called handshaking protocols.

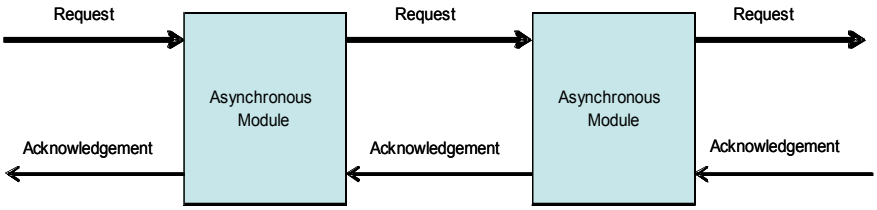

Fig. 1. Handshake based communication between modules.

The basis of the sequencing rules of asynchronous circuits lies in the handshaking protocols. Among the two mains classes of protocols, only the four-phase protocol is considered and described in this work. It is the most widely used and efficiently implemented in CMOS [8].

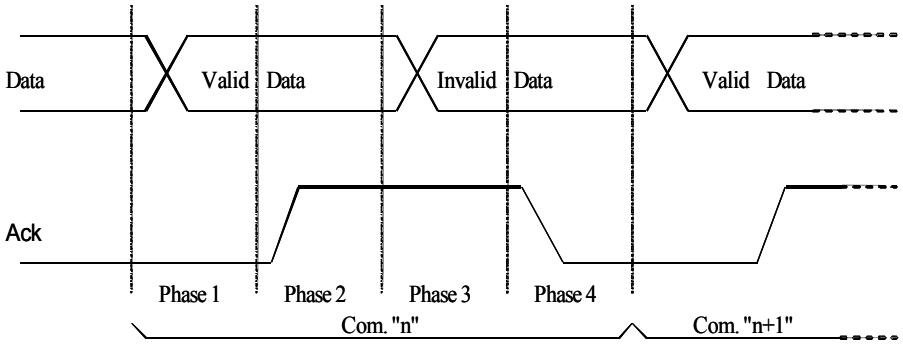

Fig. 2. Four-phase handshaking protocol.

In the first phase (Phase 1) data are detected by the receiver when their values change from invalid to valid states. Then follows the second phase where the receiver sets to one the acknowledgement signal. The sender invalidates all data in the third phase. Finally the receiver resets the acknowledgment signal which completes the return to zero phase.

Dedicated logic and special encoding are necessary for sensing data validity/invalidity and for generating the acknowledgement signal. Request for computation corresponds to data detection and the reset of the acknowledgment signal means that the computation is completed and the communication is finished.

In QDI asynchronous logic, if one bit has to be transferred through a channel with a four-phase protocol, two wires are needed to encode its different values. This is called dual-rail encoding (table 1).

Table 1. Dual rail encoding of the three states required to communicate 1 bit.

Channel data	A0	A1
0	1	0
1	0	1
Invalid	0	0
Unused	1	1

This encoding can be extended to N-rail (1-to-N).

The acknowledgement signal is generated using the data-encoding. The dual-rail encoded outputs are sensed with Nor gates for generating the acknowledgment signal, as illustrated in figure 3.

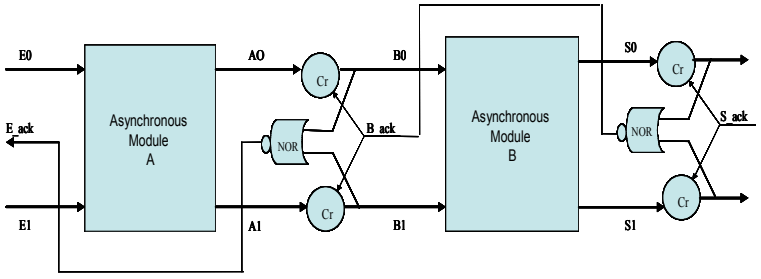

Fig. 3. 1-bit Half-buffer implementing a four-phase protocol
(Cr is a Muller gate with a reset signal)

The Muller C-element's truth table and symbol are given in Figure 4.

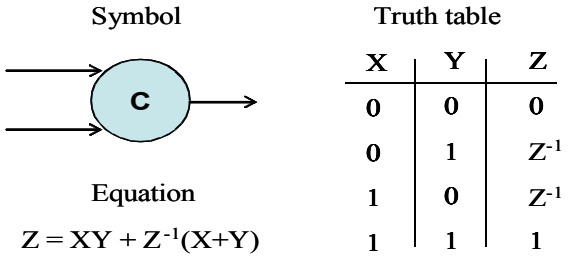

Fig. 4. Truth table and symbol of the C-element.

Figure 3 illustrates the implementation of two asynchronous modules (*A* and *B*) with their memory elements called half-buffer. The half-buffer implements a four-phase protocol. When the acknowledgement signal of module *B (B_ack)* is set, it means that the module is ready to receive data. If a data is transferred from module *A* to module *B*, module *B* computes its outputs and resets its acknowledgement signal (*B_ack*). Module *B* is then ready to receive invalid data from module *A*.

In this operating mode, the acknowledgment signal can be considered as a local enable signal which controls data storage locally. Note that this mechanism does not need any timing assumption to ensure functional correctness; it is simply sensitive to events. Hence, the acknowledgment signal enables to control the activation of the computation in a given module, as well as its time instant.

The technique proposed in this paper, exploits this property by inserting random delays in the acknowledgement signals. It is called Randomly Time-Shifted acknowledgment signals. It basically desynchronizes the power consumption curves making the differential power analysis more difficult as proved in the next section.

3 DPA and RTS acknowledgment signal on QDI asynchronous circuits: Formal Approach

In this section, we formally introduce the basis of the DPA attack [7] and formally analyse the effects of the RTS acknowledgement signal on QDI asynchronous circuits in terms of DPA resistance.

3.1 Differential Power Analysis Attack

The functional hypothesis of DPA attack is the existing correlation between the data processed by the circuitry and its power consumption. There are three main phases for processing the DPA attack: the choice of the selection function *D*, the data collection phase and the data analysis phase.

Phase 1: In the first step, the selection function is defined by finding blocks in the architecture which depend on some parts of the key. Such a function in the DES algorithm for example can be defined as follows:

$$D(C_1,P_6,K_0)= SBOX1(P_6 \oplus K_0)$$

C_1 = *first bit of SBOX1 function.*

P_6 = *6-bit plain-text-input of the SBOX1 function.*

K_0 = *6-bit of the first round's subkey: key to guess.*

SBOX1 = *a substitution function of DES with a 4-bit output.*

Phase 2: The second step consists in collecting the discrete time power signal $S_i(t_j)$ and the corresponding ciphertext outputs (CTO_i) for each of the *N* plaintext inputs (PTI_i). The power signal $S_i(t_j)$ represents the power consumption of the selection function: index *i* corresponds to the PTI_i plaintext stimulus and time t_j corresponds to the time where the analysis takes place.

Phase 3: The right key is guessed in the third phase. All current signals $S_i(t_j)$ are split into two sets according to a selection function *D*.

$$S_0 = \left\{ S_i(t_j) \middle| D = 0 \right\}$$

$$S_1 = \{S_i(t_j) | D = 1\} \tag{1}$$

The average power signal of each set is given by:

$$A_0(t_j) = \frac{1}{|n_0|} \sum_{i=1}^{n_0} S_i(t_j)$$
$$A_1(t_j) = \frac{1}{|n_1|} \sum_{i=1}^{n_1} S_i(t_j) \tag{2}$$

Where $|n_o|$ and $|n_1|$ represent the number of power signals $S_i(t_j)$ respectively in set S_0 and S_1. The DPA bias signal is obtained by:

$$S(t_j) = A_0(t_j) - A_1(t_j) \tag{3}$$

If the DPA bias signal shows important peaks, it means that there is a strong correlation between the D function and the power signal, and so the guessed key is correct. If not, the guessed key is incorrect.
Selecting an appropriate D function is then essential in order to guess a good secret key.

As illustrated above, the selection function D computes at time t_j during the ciphering (or deciphering) process, the value of the attacked bit. When this value is manipulated at time t_j, there will be at this time, a difference on the amount of dissipated power according to the bit's value (either one or zero).

Let's define $d_{0i}(t_j)$ the amount of dissipated power when the attacked bit switches to 0 at time t_j by processing the plaintext input i and define $d_{1i}(t_j)$ the amount of dissipated power when this bit switches to 1.

In reality, the values of $d_{0i}(t_j)$ *and* $d_{1i}(t_j)$ correspond to the dissipated power of all data-paths which contribute to the switching activity of the attacked bit. Each one of these values has its weight in each average power signal $A_0(t_j)$ and $A_1(t_j)$. As the goal of the DPA attack is to compute the difference between these two values, we can express the average power signal of these both sets $A_0(t_j)$ and $A_1(t_j)$ by:

$$A_0(t_j) = \varepsilon_0(t_j) = \frac{1}{|n_0|} \sum_{i=1}^{n_0} d_{0i}(t_j)$$
$$A_1(t_j) = \varepsilon_1(t_j) = \frac{1}{|n_1|} \sum_{i=1}^{n_1} d_{1i}(t_j) \tag{4}$$

Therefore, the DPA signature is expressed by:

$$\varepsilon_0(t_j) - \varepsilon_1(t_j) = \varepsilon(t_j) \quad (5)$$

In order to make an efficient analysis, the amplitude of the DPA signature $\varepsilon(t_j)$ must be as high as possible.

A simple way to guarantee this is to use a significant number of plaintext inputs (N). Indeed, the number of *PTIi* (the number of power signal $S_i(t_j)$) used to implement the attack enables to reduce the effects of the noisy signals and to increase the probability of exciting all data-paths.

- It is well known that the signal-to-noise ratio for the averaged signal increases as the square root of the number of curves.

$$SNR = \sqrt{N}\frac{S_{signal}}{\sigma_{noise}}$$

σ_{noise} is the standard deviation of the noise

- Increasing the number of plaintext inputs (PTIi) allows us to ensure that all data-paths which make switching to 0 or to 1 the attacked bit are excited. The deal here, is to take into consideration all possible quantities dxi(tj) which represent the switching current of the attacked bit. As the probability of exciting all data-paths is proportional to N, bigger the value of N, better the probability to excite all data-paths of the attacked bit is:

$$P(\omega) = \frac{N}{m}$$

m is generally unknown by the hacker and represents the number of data-paths.

Therefore, the knowledge of the implementation which enables to choose the plaintext inputs and the use of high quality instrumentation are assets that improve the DPA attack. In fact, they considerably reduce the number of data (N) required for succeeding the attack.

3.2 The RTS acknowledgement signal

The method we propose in this paper enables the designer to introduce a temporal noise in the design in order to desynchronize the time required for processing the attacked bit. The idea of the approach is to randomly shift in time the current profile of the design. To achieve this goal, we randomize the acknowledgment signal latency of the blocks of the architecture. As illustrated in figure 5, we use a delay element controlled by a random number generator. The design of the random number generator is out of the scope of this paper. True

random number generator (TRNG) design is an important topic and many different types of TRNG implementation exist [9][10].

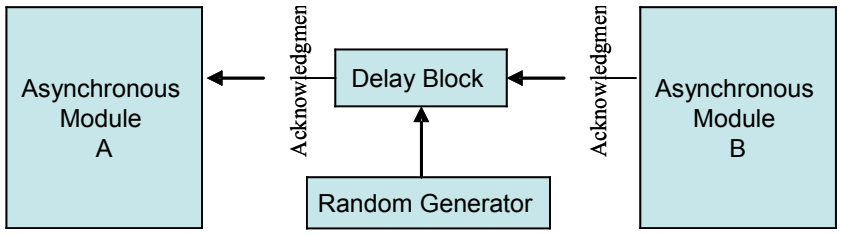

Fig. 5. Implementation of a random acknowledgment signal

Let's denote *n* the number of possible random delays implemented in a given architecture. *n* depends on the number of available acknowledgment signals *(m)* in the architecture and on the number of delays (k_i) implemented per acknowledgment signal. The "*n*" value is computed by the following expression:

$$n = \prod_{i=1}^{m} k_i$$

assuming cascaded modules.

If the acknowledgment signal is randomized *n* times, it means that the value of the attacked bit is computed at *n* different times (t_j). N/n represents the number of times the attacked bit is processed at a given time t_j and $N/2n$ represents the number of times the quantities $d_{0i}(t_j)$ and $d_{1i}(t_j)$ of this bit contribute to set S_0 and S_1 respectively. If we consider that the *N* curves are equally split in both sets $(n_0=n_1=N/2)$, the average power signal of each set is now expressed by:

$$\varepsilon_0(t) = \frac{1}{n_0} \sum_{i=1}^{N/2n} \left(\sum_{j=1}^{n} d_{0i}(t_j) \right)$$
$$\varepsilon_1(t) = \frac{1}{n_1} \sum_{i=1}^{N/2n} \left(\sum_{j=1}^{n} d_{1i}(t_j) \right) \quad (6)$$

The DPA bias signal is then given by the following expression:

$$\varepsilon(t) = \left(\varepsilon_0(t_1) - \varepsilon_1(t_1)\right) + ... + \left(\varepsilon_0(t_n) - \varepsilon_1(t_n)\right)$$

with

$$\varepsilon_0(t_n) = \frac{1}{n_0} \sum_{i=1}^{N/2n} d_{0i}(t_n) \ ;$$

$$\varepsilon_1(t_n) = \frac{1}{n_1} \sum_{i=1}^{N/2n} d_{1i}(t_n) \tag{7}$$

These expressions show that, instead of having a single quantity $\varepsilon_x(t_j)$, we have n different significant quantities $\varepsilon_x(t_n)$ which correspond to n times where the attacked bit is processed. Moreover, it also demonstrates that each quantity $\varepsilon_x(t_j)$ is divided by a factor n as illustrated by the following simplification:

$$\varepsilon_x(t_j) = \frac{1}{n_x}\left(d_{xj}(t_j) + \ldots + d_{xn}(t_j)\right) \cong \frac{d_{xj}(t_j)}{n} \tag{8}$$

$$\text{with} \quad d_{xj}(t_j) \cong \ldots \cong d_{xn}(t_j)$$

It means that, although the number of significant points is increased by n, this approach divides by n the average current peaks variations. It offers the possibility to bring down the level of DPA bias signal closer to circuitry's noise.

3.2 Discussion

Let's for example implement the DPA attack using *1000* plaintext inputs *(N=1000)*. In the standard approach where the attacked bit is processed at a unique given time, we obtain an average of 500 current curves for each of the sets S_0 and S_1.

Using our approach with RTS acknowledgment signals and assuming *n=16* (for example), we obtain 16 different points (in terms of time) where the attacked bit is processed. There are 62 values $d_{xi}(t_j)$ (*N/n* curves) where this bit is processed at time (t_j). Each set then contains 31 curves. When the average power signal of each set is calculated, values $d_{xi}(t_j)$ are 16 times lower than without RTS acknowledgment signals. Hence, the contribution of $d_{xi}(t_j)$ in current peaks variations are reduced by a factor 16.

Therefore, to succeed the attack the hacker is obliged to significantly increase the number of acquisitions *(N)* or to apply a cross-correlation function which is exactly the goal to achieve in terms of attack's complexity. In fact, cross-correlation remains a useful method for synchronizing data. But to be functional, the hacker must identify the amount of current profile of the attacked bit $(d_{xi}(t_j))$ to be used as a reference, and then compute cross-correlations in order to synchronize each of the N curves with the reference. Knowing that, the cross-correlation is applied on instantaneous current curves which contain significant quantity of noise.

To increase the difficulty of this analysis, the value of n can be significantly increased by dealing with the values of m and k.

- The value of m depends on the architecture. Its value can be increased by expanding the acknowledgment signals of the architecture. Each bit or intermediate value of the design can be separately acknowledged. This technique enables also to reduce the data-path latency.

- The values of the delay depend on the time specification to cipher/decipher data. They are bounded by the maximum ciphering/deciphering time.

Consequently, the acknowledgement signals of any asynchronous quasi delay insensitive circuit can be exploited to introduce random delays and therefore increase the DPA resistance of the chips.

4 Case Study: DES Crypto-processor

This section deals with the different possibilities of implementing RTS acknowledgment signals on QDI asynchronous circuits. The DES was chosen as an evaluation vector because the attack on this algorithm is well known.

Figure 6 represents the DES core architecture, implementing a four-phase handshake protocol, using 1-to-N encoded data and balanced data-paths [2]. The architecture is composed of three iterative asynchronous loops synchronized through communicating channels. One loop for the ciphering data-path, the second for the key data-path and the last one for the control data-path which enables the control of the sixteen iterations of the algorithm.

For example let's apply the technique to the five grey blocks of figure 6. Each block has its own acknowledgement signal and the delay inserted in each acknowledgment signal can take four values. Therefore, there are 1024 possible delay values *(n=1024)*. It means that (in terms of DPA resistance) the current peak variations corresponding to $d_{xi}(t_j)$ will be divided by 1024.

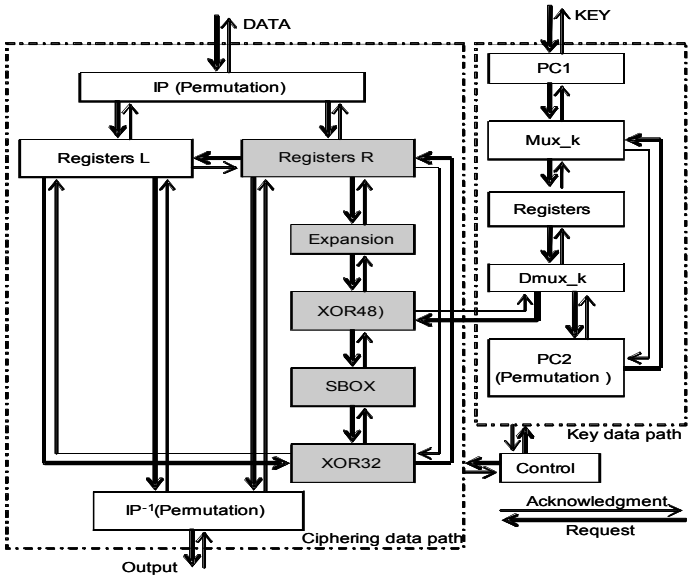

Fig. 6. Asynchronous DES core architecture

5 Results and Analysis: Electrical simulations

Electrical simulations enable us to analyze the electrical behaviour of the design with high accuracy, i.e. without disturbing signal (noise). All electrical simulations are performed with *Nanosim* using the HCMOS9 design kit (0.13µm) from STMicroelectronics.

The architecture used for these electrical analysis implements one acknowledgement signal per block. However, for the needs of illustration only the acknowledgment signal of the inputs of the *SBOX1* is randomly delayed with 8 different delays. The defined selection function, used to implement the attack, is as follows:

$$D(C_n,P_6,K_0) = SBOX1(P_6 \oplus K_0)$$
$$with\ n \in \{1,2,3,4\}$$

The DPA attack has been implemented on the four output bits of the *SBOX1* and on the first iteration of the DES algorithm using 64 plaintext inputs (*N=64*). Figure 7 shows the current profile of the first iteration when the RTS acknowledgment signal is activated and deactivated. When the delay of 13ns is used, the time required for processing an iteration (figure 7-b) corresponds to the time required to process 3 iterations without delays (figure 7-a). Hence the ciphering time is multiplied by a factor 3. This delay is chosen for the sake of illustration only. Given a level of DPA resistance, the delay can be strongly decreased in practice (down to a few nanoseconds with this technology) to reduce as much as possible the timing overhead as well as the hardware overhead caused by the application of the technique

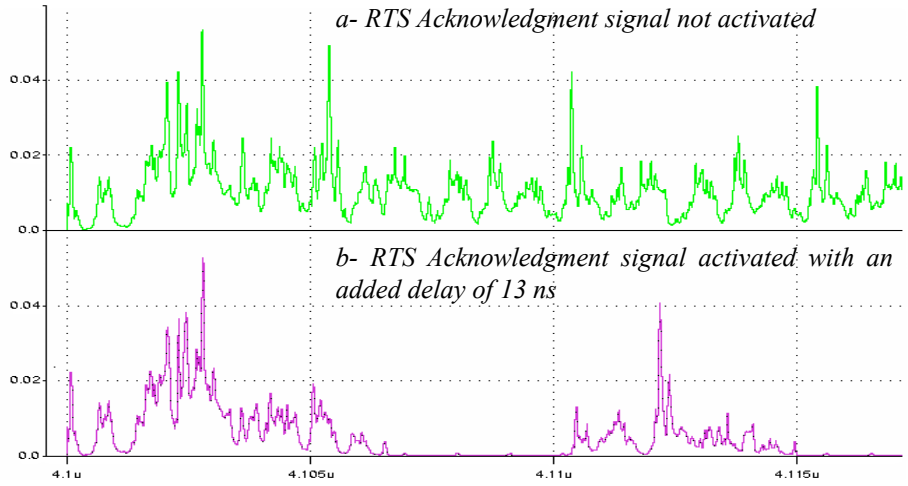

Fig. 7. Current profile of the DES QDI asynchronous architecture.

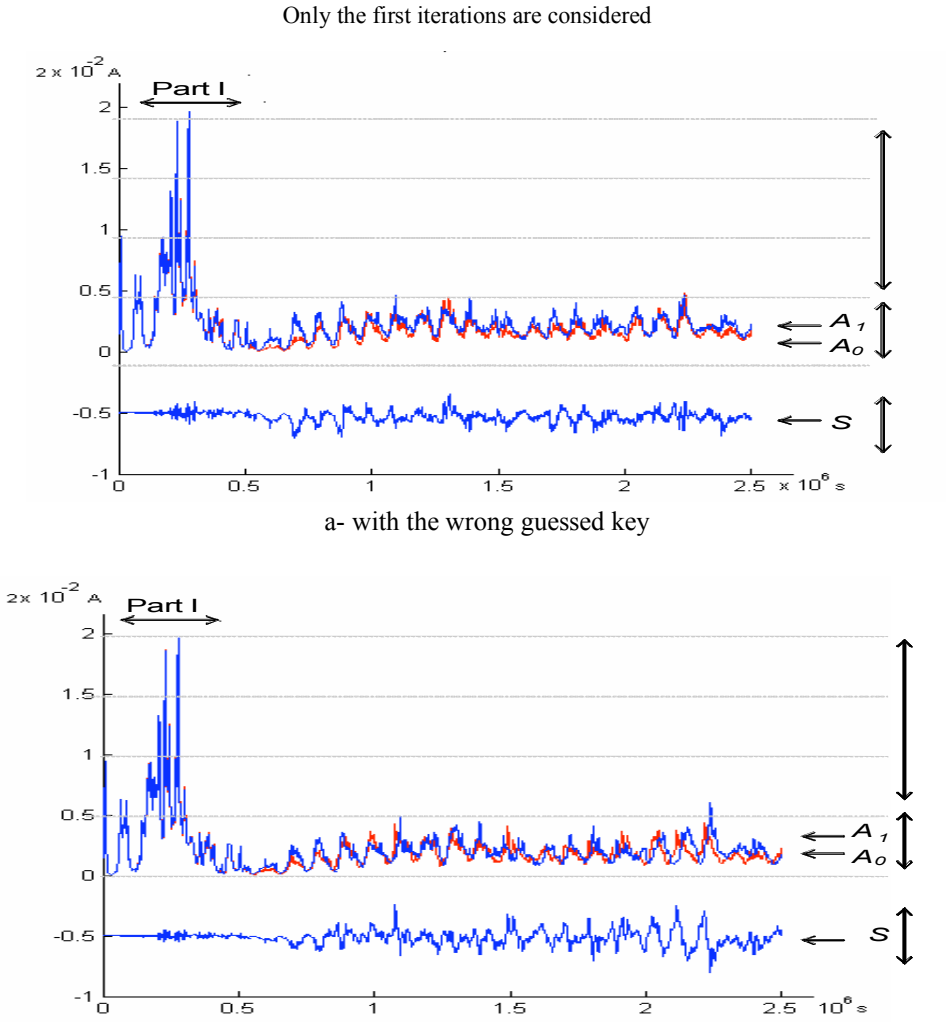

Fig. 8. Electrical signatures when performed DPA attack on bit 4 of the SBOX1. Only the first round is considered and computed using more than 2.100.000 point.

As the *SBOX1* has four output bits encoded in dual-rail, we have 8 data-paths (from outputs to inputs) which enable to compute 8 values of $d_{xi}(t_j)$. Let's recall that, $d_{xi}(t_j)(d_{0i}(t_j);d_{1i}(t_j))$ corresponds to the amount of dissipated power when the attacked bit is processed at time t_j. For example, let's consider the output bit 4 of the *SBOX1*.

Contrary to a standard approach and due to the 8 delay shifts, the values $d_{04}(t_j)$ and $d_{14}(t_j)$ are processed 4 times instead of being processed 32 times, so that their weights are reduced by a factor 8 into sets S_1 and S_0. Each of this set enables us to calculate the average currents $A_0(t_j)$ and $A_1(t_j)$.

Figure 8 shows these average current profiles ($A_0(t_j)$ and $A_1(t_j)$) which are used to compute the DPA bias signal ($S(t_j)$), also shown in figure 8.

Part I of these curves represent the first encryption operations in the first iteration (see figure 8). This part is not affected by the RTS acknowledgment signal which is only applied on *SBOX1*. In fact, before computing the *SBOX* function, the chip first computes IP, Expansion and Xor48 functions (figure 6), so that, in the first iteration, these functions, are not affected by the RTS acknowledgement signal of *SBOX1*. This explains why the amplitude of the average power curve starts decreasing after part I and it clearly illustrates the effect of the RTS signal on the power curves. This can of course be changed by activating the RTS acknowledgement signals of blocks IP, Expansion and/or Xor48.

In the considered example, 64 PTI_i curves are used to implement the attack. In this case, obtaining the key bit from the DPA bias signal is impossible as shown in figure 8. Indeed, there is no relevant peak in the DPA current curves (figure 8-a and 8-b).

6 Conclusion

This paper presented a countermeasure against DPA based on randomly time-shifted acknowledgment signals of asynchronous QDI circuits. The efficiency of the countermeasure was first theoretically formalized and then demonstrated using electrical simulations. The technique principle was illustrated on a DES architecture.

Future works will be focused on the design and fabrication of a DES prototype implementing the RTS acknowledgement signals together with a random number generator.

7 References

1. P. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis," Advances in Cryptology - Crypto 99 Proceedings, Lecture Notes In Computer Science Vol. 1666, M. Wiener ed., Springer-Verlag, 1999.

2. Simon Moore, Ross Anderson, Paul Cunningham, Robert Mullins, George Taylor, “Improving Smart Card Security using Self-timed Circuits”, Eighth International Symposium on Asynchronous Circuits and systems (ASYNC2002). 8-11 April 2002. Manchester, U.K.

3. L. A. Plana, P. A. Riocreux, W. J. Bainbridge, A. Bardsley, J. D. Garside and S. Temple, “SPA - A Synthesisable Amulet Core for Smartcard Applications”, Proceedings of the Eighth International Symposium on Asynchronous Circuits and Systems (ASYNC 2002). Pages 201-210. Manchester, 8-11/04/2002. Published by the IEEE Computer Society.

4. Jacques J. A Fournier, Simon Moore, Huiyun Li, Robert Mullins, and Gerorge Taylor,”Security Evalution of Asunchronous Circuits”, CHES 2003, LNCS 2779, pp 137-151, 2003.

5. F. Bouesse, M. Renaudin, B. Robisson, E Beigne, P.Y. Liardet, S. Prevosto, J. Sonzogni, “DPA on Quasi Delay Insensitive Asynchronous circuits: Concrete Results”, To be published in XIX Conference on Design of Circuits and Integrated Systems Bordeaux, France, November 24-26, 2004.
6. G.F. Bouesse, M. Renaudin, S. Dumont, F. Germain, « DPA on Quasi Delay Insensitive Asynchronous Circuits: Formalization and Improvement », DATE 2005. p.424
7. T. S. Messerges and E. A. Dabbish, R. H. Sloan, “Investigations of Power Analysis Attacks on Smartcards”, USENIX Workshop on Smartcard Technology, Chicago, Illinois, USE, May 10-11, 1999.
8. Marc Renaudin, “Asynchronous circuits and systems: a promising design alternative”, Microelectronic for Telecommunications : managing high complexity and mobility” (MIGAS 2000), special issue of the Microelectronics-Engineering Journal, Elsevier Science, GUEST Editors : P; Senn, M. Renaudin, J, Boussey, Vol. 54, N° 1-2, December 2000, pp. 133-149.
9. Viktor Fischer, M. Drutarovský, True Random Number Generator Embedded in Reconfigurable Hardware, In C. K. Koç, and C. Paar, (Eds.): Cryptographic Hardware and Embedded Systems (CHES 2002), Redwood Shore, USA, LNCS No. 2523, Springer, Berlin, Germany, ISBN 3-540-00409-2, pp. 415-430.
10. V. Fischer, M. Drutarovský, M. Šimka, N. Bochard, High Performance True Random Number Generator in Altera Stratix FPLDs, in J. Becker, M. Platzner, S. Vernalde (Eds.): "Field-Programmable Logic and Applications," 14th International Conference, FPL 2004, Antwerp, Belgium, August 30-September 1, 2004, LNCS 3203, Springer, Berlin, Germany, pp. 555-564.