Securing Encryption Algorithms against DPA at the Logic Level: Next Generation Smart Card Technology

Kris Tiri and Ingrid Verbauwhede

UCLA Electrical Engineering Department,
7440B Boelter Hall, P.O. Box 951594, Los Angeles, CA 90095-1594
`{tiri, ingrid}@ee.ucla.edu`

Abstract. This paper describes a design method to secure encryption algorithms against Differential Power Analysis at the logic level. The method employs logic gates with a power consumption, which is independent of the data signals, and therefore the technique removes the foundation for DPA. In a design experiment, a fundamental component of the DES algorithm has been implemented. Detailed transistor level simulations show a perfect security whenever the layout parasitics are not taken into account.

1 Introduction

The physical implementation of an encryption algorithm is bound to provide an attacker with important information on top of the plain- and ciphertext used in traditional cryptanalysis. Variations in, among other things, the power consumption of the encryption module and the arrival time of the encrypted data can be observed, and possibly linked to the input data and the secret key. Attacks that use this additional information and link it to the internal state, and hence to the secret key, are referred to as Side Channel Attacks (SCA's) [1].
From these SCA's, Differential Power Analysis (DPA) is the most powerful. It relies on statistical analysis and error correction to extract information from the power consumption that is correlated to the secret key [2]. Many countermeasures that conceal the supply current variations at the architectural or the algorithmic level have been put forward. Yet, they are not really effective or practicable against DPA and/or its derivatives, as the variations actually originate at the logic level.
The fact that the power consumption of a single logic gate, which is the most elementary building stone of the complete encryption module, is controlled by both the logic value and the sequence of its input signals forms the basis of DPA. Using a logic style for which a logic gate has at all times a constant power consumption that is independent of the signal transitions, removes the foundation of DPA and is therefore an effective means to halt DPA.
In this paper, we first present the basics of Differential Power Analysis. Then, we briefly discuss Sense Amplifier Based Logic, which is a logic style with signal independent power consumption. Next, we (1) introduce the design experiment, which consists of securing a module of the DES algorithm against DPA at the logic level; (2)

investigate the effectiveness of the SABL approach; and (3) discuss the effects of the layout parasitics. Finally a conclusion will be formulated.

2 Basics of Differential Power Analysis

Differential Power Analysis has been extensively described in literature. It was first introduced in [2].
A DPA is executed in two phases: data collection and data analysis. During data collection, the power consumption of the device is measured by sampling and recording the supply current for a large number of encryptions. During data analysis, a selection function, which depends on a guess of some bits of the secret key, divides the power measurements into two sets. For each set, a typical supply current is calculated and subsequently a Differential Trace (DT) is generated by computing the difference between the two typical supply currents.
The selection function D consists of predicting a state bit of the encryption module. If the correct subset of the secret key has been predicted, D is correlated with the state bit and hence with the power consumption of the logic operations that are affected by this state bit. The power consumption of the other logic operations and measurement errors however, are uncorrelated. As a result, the DT will approach the effect of the target bit on the power consumption and there are noticeable peaks in the DT. If on the other hand the guess on the secret key was incorrect, the result of the selection function is uncorrelated with the state bit: the DT will approach 0.

3 Sense Amplifier Based Logic: a CMOS Logic Style with Signal Independent Power Consumption

Every logic style can be classified into one of the two existing logic families. If the logic gate continuously draws a current from the supply and measures its state through the path the current takes, the logic style is said to be Current Mode Logic (CML). If on the other hand, the logic gate only draws a current from the supply to change state and measures its state by the amount of charge it stores on a capacitance, the logic style is said to be Voltage Mode Logic (VML).
CML has constant power consumption under the condition that the gate draws a perfectly constant current from the power supply and this independently of the in- and/or output signals. In order to build a current source capable of generating a constant current, special circuit techniques that minimize channel length modulation have to be used. The decisive drawback of CML however, is its static power consumption: even when the logic gate is not processing any data, it continuously burns the current, which makes this logic style impractical for low power applications.
A better alternative is Sense Amplifier Based Logic (SABL) [3]. SABL is a VML style that uses a fixed amount of charge for every transition, including the degenerated events in which the gate does not change state. This means that the logic gate charges

in every cycle a total capacitance with a constant value, even though ultimately different capacitances are switched.
In short, SABL is based on two principles. First, it is a Dynamic and Differential Logic (DDL) and therefore has one and exactly one switching event per cycle and this independently of the input value and sequence. Second, during that switching event, it discharges and charges the sum of all the internal node capacitances together with one of the balanced output capacitances. Hence, it discharges and charges a constant capacitance value. While many DDL-styles exist [4], only SABL (1) controls exactly the contribution of the internal parasitic capacitances of the gate into the power consumption by charging and discharging each one of them in every cycle; and (2) has symmetric intrinsic in- and output capacitances at the differential signals such that it has balanced output capacitances.
In addition to the fact that every cycle the same amount of charge is switched, the charge goes through very similar charge and discharge paths during the precharge phase and during the evaluation phase respectively. As a result, the gate is subject to only minor variations in the input-output delay and in the instantaneous current. This is important since the attacker is not so much interested in the total charge per switching event, as in the instantaneous current and will sample several times per clock cycle in order to capture the instantaneous current.

4 Design Experiment

4.1 Description of Experimental Setup

Goal of the design experiment is to develop design guidelines and to identify possible hurdles for securing an encryption module against DPA at the logic level by simply removing the foundation of DPA. For this purpose, any encryption algorithm could have been chosen. The reason for choosing the DES algorithm [5] is the focus of a great part of contemporary research on how to perform and how to thwart DPA on the DES algorithm.
In order to obtain supply current traces that are as accurate as possible, simulations have been run at the transistor level using HSPICE. Simulating the complete algorithm however, is computationally unfeasible and the algorithm has been stripped-down to a minimum.
The experimental setup, which is shown in Fig. 1, is part of the last round of the DES-algorithm. The module calculates 4 bits of the ciphertext C using a subkey K of length 6 and 4 and 6 bits of the left and right plaintexts L and R respectively. The substitution box is the S1-box. The expansion of the right plaintext R, the permutation of the result of the S-box, and the inverse initial permutation, which are present in the actual DES-algorithm, have been discarded, as they do not change the power measurements: they are hardwired.
The selection function D(C,b,K) consists of calculating bit number b of the left plaintext L, using the known ciphertext C and a guess on the secret key K. The right plain-

text R is also known. In the DES algorithm, R is fed into the inverse initial permutation to form part of the ciphertext C.

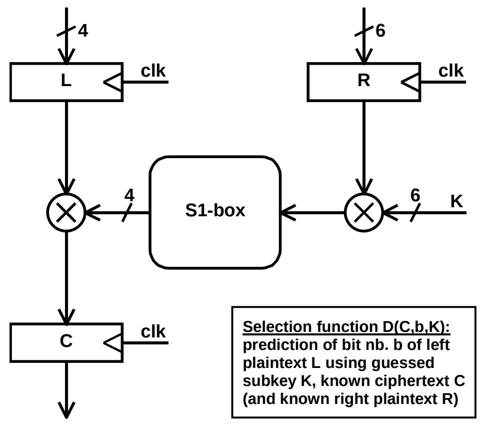
4
L
clk
6
R
clk
4
S1-box
6
K
C
clk
Selection function D(C,b,K):
prediction of bit nb. b of left plaintext L using guessed subkey K, known ciphertext C (and known right plaintext R)

Fig. 1. Experimental setup: DPA on a submodule of the last round in the DES-algorithm

Restricting the experiment to the implementation in Fig. 1 does not simplify the task of putting a stop to DPA. On the contrary, in the implementation of the complete DES algorithm, the power consumption caused by the calculation of the other bits in the same and in the previous rounds, will act as an extra and large noise source on the power measurements. Note also that in this experiment, all measurements are 'perfect measurements'. Aside from the accuracy of HSPICE, there is no quantization error, thermal noise, jitter on the clock of the encryption module, jitter on the sampling moment or any other phenomenon that may introduce a measurement error.
To allow for a comparison, the module has been implemented both in static complementary CMOS logic (SC-CMOS), which is the default logic style in a standard cell library, and in SABL for a 0.18µm, 1.8V CMOS technology. Simulations have been done in HSPICE. In total, the supply current has been captured for 5000 clock cycles with a random input at the plaintext registers L and R, and with a fixed secret key K. The same random input, and the same secret key have been used for both implementations. In order to capture all current variation, the sampling frequency has been set to 100GHz, which corresponds to one sample every 10ps. Note that this very high level of accuracy demands massive simulations. The most time-consuming simulation required 275 hours on a HP Visualize B1000 to complete.

4.2 Effectiveness of the SABL Approach

In a first setup, the simulations are based on a netlist that does not include effects caused by the layout. The parasitic capacitances coming from the intra and inter cell routing of the data signals have been neglected. Fig. 2 shows the transient and statistical properties of the simulated supply current of the SC-CMOS and the SABL implementation of the module presented in Fig. 1.

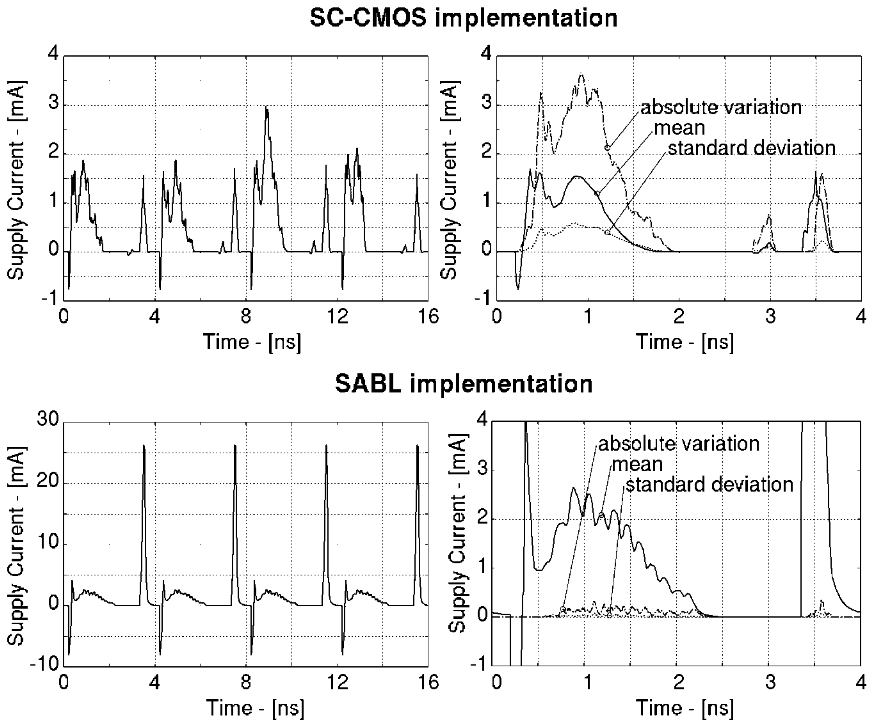

Fig. 2. Simulated supply current: supply current transient of 4 clock cycles (left) and supply current characteristics based on 5000 clock cycles (right) for SC-CMOS (top) and SABL (bottom) implementation

Fig. 2(left) depicts a snapshot of the supply current transient. In total, 4 clock cycles of each 4ns are shown. The supply current of the SABL implementation is very regular and independent of the input signals, whereas the supply current of the SC-CMOS implementation is completely different from cycle to cycle and hence highly dependent on the input signals.

Note that the supply current of the SABL module alternates between a short, high current peak and a time span with a lower current. These events correspond respectively to the precharge phase, in which all gates switch at the same moment, and the evaluation phase, in which each gate switches when its inputs arrive from preceding gates. The current in the evaluation phase is caused by the pairs of static inverters that have been inserted between the SABL gates in order to cascade these dynamic gates according to the domino design rules. We preferred the domino design rules to np design rules for ease of implementation. The pairs of static inverters however, add an extra penalty on the area and the power consumption. The mean energy consumption per clock cycle of the SABL implementation is 11.25pJ compared to 2.70pJ for the SC-CMOS implementation.

Fig. 2(right) depicts the statistical properties of the entire supply current transient. Three curves that describe the typical supply current are shown: they represent the mean supply current, the absolute variation in the mean supply current and the standard deviation on the mean supply current. The curves are generated by first folding the supply current of the 5000 clock cycles on top of each other into 1 clock cycle to generate an 'eye'-diagram and then subsequently calculating the point wise mean, absolute variation and standard deviation. The curves confirm our observations. The mean current of the SABL implementation is a representative switching event for the supply current in every clock cycle. The maximum absolute variation and the maximum standard deviation are 0.37 mA and 89.5 μA respectively. These values correspond to 2% and 4.8% of the mean current at their point of occurrence. The SC-CMOS implementation however, experiences a significant variation in the supply current from clock cycle to clock cycle. The maximum absolute variation and the maximum standard deviation are 3.66 mA and 591.2 μA respectively. These values correspond to 239% and 38.1% of the mean current at their point of occurrence. Table 1 summarizes the numbers.

Table 1. Simulated supply current: variation in the typical supply current based on 5000 clock cycles for SC-CMOS and SABL implementation

Implementation	max(abs. var.) [mA]	*ratio to mean current*†	max(std. dev.) [μA]	*ratio to mean current*†
SC-CMOS	3.66	239%	591.2	38.1%
SABL	0.37	2%	89.5	4.8%

† *At point of occurrence.*

Fig. 3 shows the Differential Traces that have been generated with 8 different key guesses in the selection function. The first bit of plaintext register L has been predicted. Note that in total 64 (=2^6) different guesses of the secret key are possible. Only the DT's of 8 of them are shown for transparency of the figure. The other 56 DT's however, are in accordance with the curves that correspond with the 7 incorrectly guessed keys. The correct secret key is 46.

For the SC-CMOS implementation, the DT of the correct secret key exhibits peaks that are significantly higher than the DT's of the incorrectly guessed keys. All peaks can be brought back to certain precise events. The first peak around 0.5ns corresponds to the rising edge of the clock. At this instant of time, the output of the first bit of the register L becomes equal to the bit that we predicted with the selection function. The second peak around 3ns corresponds to the instant that the input to the first bit of register L changes from the bit that we predicted to a new random input. The last peak around 3.5ns corresponds to the falling edge of the clock.

For the SABL implementation, one can not distinguish which DT is from the correct secret key. Moreover, the DT of the correct secret key would not even be considered as the DT of a possible correct secret key. Contrary to the SC-CMOS module, an analysis of precise events is not possible.

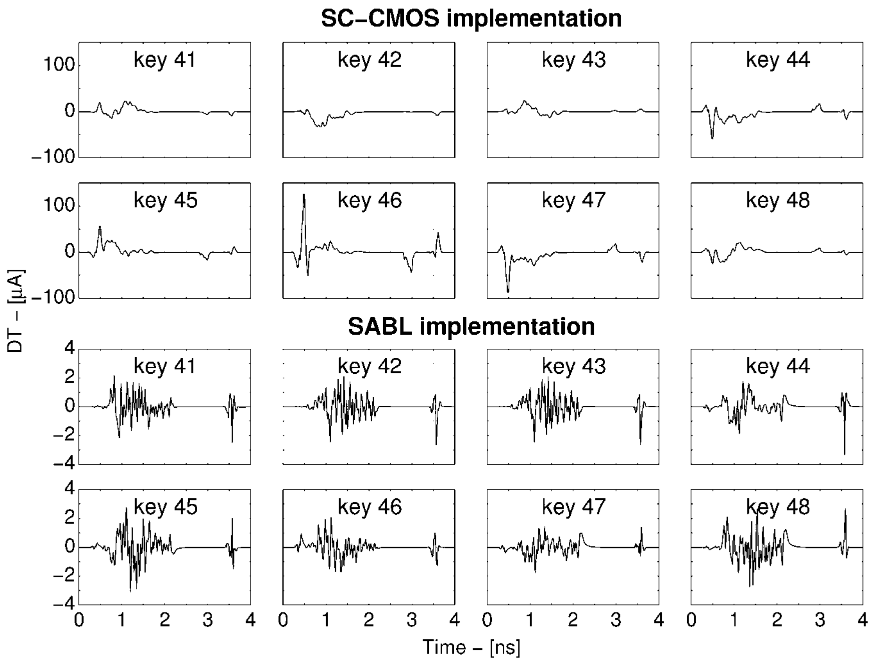

Fig. 3. Differential Traces based on 5000 clock cycles generated by 8 successive key guesses for SC-CMOS (top) and SABL (bottom) implementation. Key 46 is secret key. Please note the different scales for SC-CMOS and SABL implementations

On top of the fact that the DT of the correct secret key does not have any noticeable peaks for the SABL module, the DT's of the SABL module are much smaller than the DT's of the SC-CMOS module. As a result, to determine the DT's of the SABL implementation, the test equipment that captures the supply current in the measurement setup should have a much better accuracy than is necessary to determine the DT's of the SC-CMOS implementation, which are almost 2 orders of magnitude larger. Fig. 4 details the DT's that have been generated with the correct secret key.

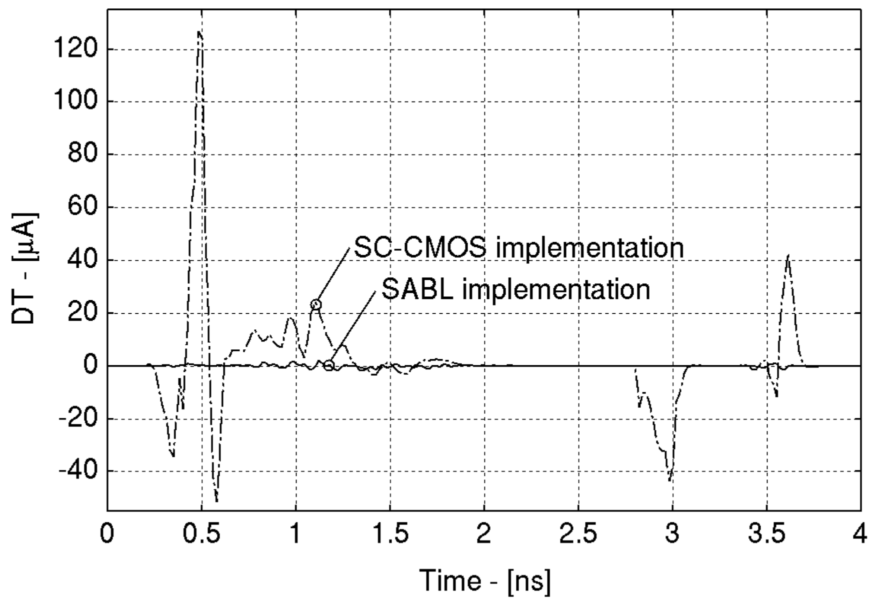

Fig. 4. Differential Traces based on 5000 clock cycles generated by correct secret key for SC-CMOS and SABL implementation

Fig. 5 shows the influence of the data collection on the information content in the DT's. In each plot, the peak-to-peak value (p2p) or the root-mean-square value (rms) of the DT's generated by (1) the correct secret key; (2) an incorrect secret key; and (3) a random bit string as selection function, are shown in function of the number of clock cycles used to generate the DT. The random bit string has been used to avoid any statistical biases of the S-box output. Note the scale difference on the vertical axis between the SC-CMOS implementation and the SABL implementation.
For the SC-CMOS implementation, a mere 200 clock cycles are enough to disclose the correct secret key. For the SABL implementation however, more than 5000 clock cycles have been simulated and the correct secret key does not stand out. It is very unlikely that increasing the number of clock cycles will make the correct secret key stand out. The transient response of the curves has died out and the p2p and the rms are in a steady state response: they are set by the section of the DT, which corresponds to the power consumption of the S1-box. The power consumption of the S1-box is uncorrelated with the selection function, as the bits in the left plaintext have no influence whatsoever on what happens inside the S1-box.
One could argue that occasionally the correct secret key also seems to stand out for the SABL implementation. Fig. 5 however, shows the p2p and rms of only one incorrect secret key. There are DT's of other incorrect secret keys for which the p2p or rms are comparable and/or higher than for the correct secret key.

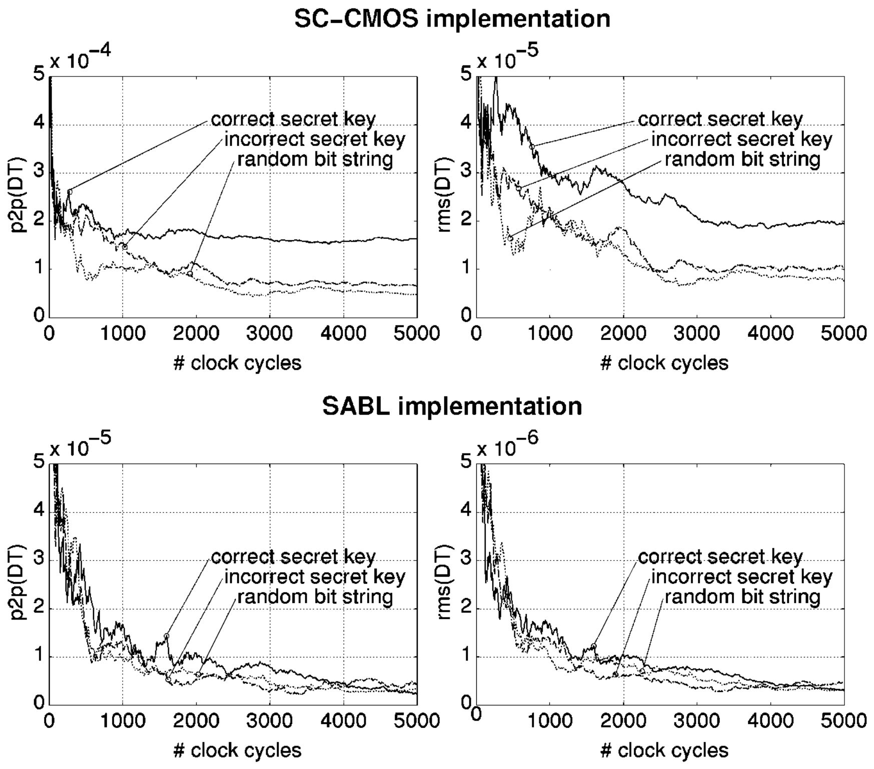

Fig. 5. Influence of data collection: peak-to-peak value (left) and root-mean-square value (right) of the Differential Traces generated by the correct secret key, an incorrect secret key and a random bit string as selection function for SC-CMOS (top) and SABL (bottom) implementation. Please note the different scales for SC-CMOS and SABL implementations

4.3 Effects of Layout Parasitics

The SABL approach has shown to be an effective remedy against DPA when the layout parasitics are not taken into account. In the next setup, the simulations are based on a netlist that does account for the effects of the layout. First, a cell library has been created that contains all cells used in the module. Then, these cells have been used to place and route the module. The complete layout in SABL is shown in Fig. 6.
The parasitic capacitances from the intra- and inter-cell routing will not only result in a performance degradation, in particular in an increase of the input-output delay and of the power consumption, but they will also result in variations in the total charge that is used per switching event if both differential output signals do not see the same parasitic capacitances.

Special attention has been given to the layout of each cell in an effort to balance its intrinsic in- and output capacitances. The inter-cell routing has been addressed by routing the differential lines in the same environment. This assures that the parasitic capacitances to other metal layers are comparable at both interconnects. Further, the cross coupling between long adjacent lines in the same layer has been addressed with shielding. The shielding has a tradeoff with an increase in power consumption and in area.

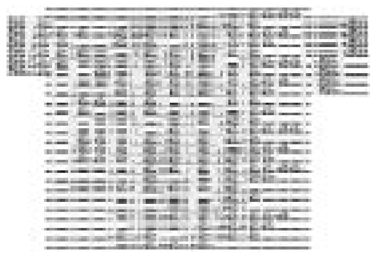

Fig. 6. Layout of SABL implementation of module presented in Fig. 1

Fig. 7, which depicts the transient and statistical properties of the simulated supply current, is in accordance with Fig. 2(bottom). The snapshot of the supply current transient remains very regular and independent of the input signals. Though, compared with the case that the layout was not taken into account, there is approximately a penalty of 100% in the input-output delay and in the power consumption. The mean current of the SABL implementation remains likewise a representative switching event for the supply current in every clock cycle. The maximum absolute variation and the maximum standard deviation are 0.26 mA and 65.8 µA respectively. These values correspond to 13% and 2% of the mean current at their point of occurrence. Note that in spite of the increase in power consumption, the absolute figures have decreased with approximately 30% compared with the case that the layout was not taken into account. The relative figure of the maximum absolute variation however, has increased by a factor of 5. The relative figure of the maximum standard deviation on the other hand, has decreased by a factor of 2.5.

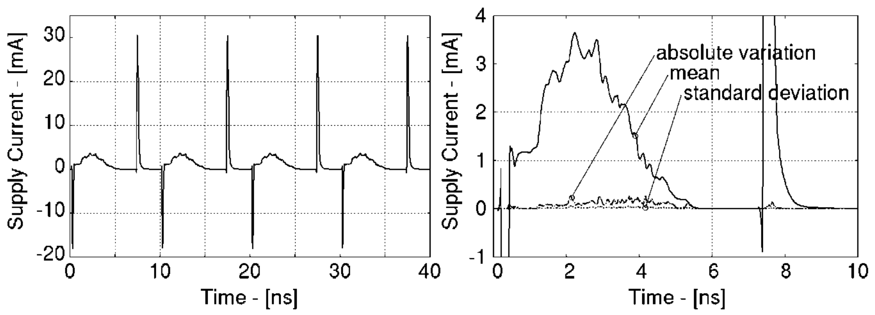

Fig. 7. Simulated supply current: supply current transient of 4 clock cycles (left) and supply current characteristics based on 5000 clock cycles (right)

Even though there does not seem to be a significant difference in the supply current characteristics between the module before layout and the one after the layout phase, the DT of the correct secret key exhibits 2 peaks that are higher than the DT's of the incorrectly guessed secret key as can be seen in Fig. 8. The first peak around 0.5ns corresponds to the rising edge of the clock. At this instant of time, the output of the first bit of the register L changes state. The peak has a value of 10.28 µA, which is a factor of 12.3 smaller than the peak at the rising edge of the SC-CMOS implementation. The latter implementation however, did not include the layout parasitics. Including the layout parasitics into the SC-CMOS implementation will increase this number. The second peak at 7.5ns corresponds to the falling edge of the clock. At this instant, the output of the XOR is read into C.

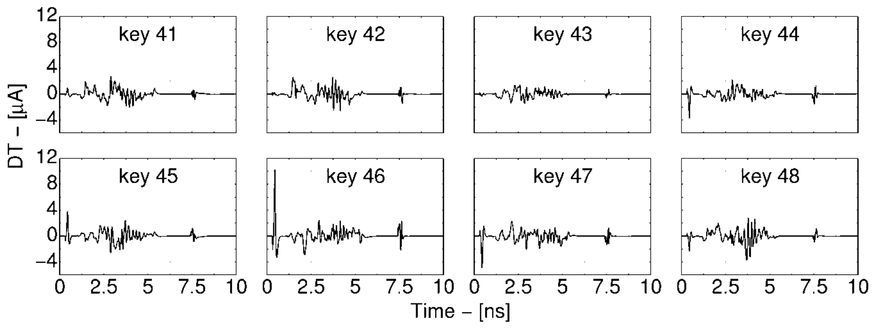

Fig. 8. Differential Traces based on 5000 clock cycles generated by 8 successive key guesses. Key 46 is secret key

5 Conclusions

We have presented a technique to thwart DPA that uses a logic style with data independent power consumption. The technique achieves perfect security whenever the layout parasitics are neglected. In our simulation setup, the secret key has not been exposed and increasing the data collection is very unlikely to help out. With parasitics, DPA is possible. Our simulations however, show that the resulting DT's are more than an order of magnitude smaller than a SC-CMOS implementation. Furthermore in our opinion, improvements are still possible. The resulting increased security will as always come in a tradeoff with some cost. Here, the cost will be an increase in power, area for a more aggressive shielding and an increase in area, initial design time for a perfect symmetric standard cell. It is still unclear however, whether a DPA on an actual product will reveal the secret key or not. The measurement setup will suffer from measurement errors, a larger resolution in the time domain, supply current filtering caused by decoupling, supply parasitics and additional large supply current noise coming from other modules, which for the non-sensitive parts will have the huge supply current variations of SC-CMOS.
In Table 1, we have also presented the minimum variation that seems achievable for any technique at the logic level or higher levels that tries to balance the instantaneous power consumption of a module implementing a logic function. Any actual implementation will suffer from larger variation coming from not only unsymmetrical intra- and inter-cell routing but as well from technology and process variations, over which absolutely no control is possible.

Acknowledgement

The authors would like to acknowledge the support of UC-MICRO (#02-079) and National Science Foundation (CCR-0098361).

6 References

1 Hess, E., Janssen, N., Meyer, B., Schuetze, T.:Information Leakage Attacks Against Smart Card Implementations of Cryptographic Algorithms and Countermeasures – a Survey. Proc. Of EUROSMART Security Conference (2000) 55–64

2 Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. Proc. of Advances in Cryptology, Lecture Notes in Computer Science Vol. 1666 (1999) 388-397

3 Tiri, K., Akmal, M., Verbauwhede, I.: A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards. Proc. Of 28th European Solid-State Circuits Conference (2002) 403-406

4 Bernstein, K., Carig, M., Durham, C., Hansen, P., Hogenmiller, D., Nowak, E., Rohrer, N.: High Speed CMOS Design Styles. Kluwer Academic Publishers (1998) 111-123

5 National Bureau of Standards, “Data Encryption Standard,” Federal Information Processing Standards Publication 46, January 1977