Delay Insensitive Encoding and Power Analysis: A Balancing Act

Konrad J. Kulikowski, Ming Su, Alexander Smirnov, Alexander Taubin, Mark G. Karpovsky, Daniel MacDonald
Reliable Computing Laboratory
Department of Electrical and Computer Engineering
Boston University
8 Saint Mary's Street, Boston, MA 02215
{konkul ; mingsu ; alexbs ; taubin ; markkar ; djm}@bu.edu

Abstract

Unprotected cryptographic hardware is vulnerable to a side-channel attack known as Differential Power Analysis (DPA). This attack exploits data-dependent power consumption of a computation to determine the secret key. Dual-rail asynchronous circuits have been regarded as a potential countermeasure to this attack. In this paper, we evaluate the security of asynchronous dual-rail circuits against DPA. Our results show that, unless special precautions are taken, asynchronous circuits are not inherently more DPA resistant than their synchronous dual-rail counterparts. We show that the use of NULL-spaced or Return-To-Zero (RTZ) protocols, used to provide delay-insensitive encoding for asynchronous circuits, can make a DPA attack easier. We present an overview of balancing dynamic implementations of dual-rail fine-grained asynchronous gates that offer a solution for the DPA weakness. We demonstrate the use of asynchronous balanced cells that use RTZ which are not only secure against DPA but also deliver high performance with low design effort through automated pipelining.

1. Introduction

Cryptographic hardware is vulnerable to a variety of attacks that exploit the physical properties of their implementations. One such side-channel attack uses the power consumption of the device during computation to derive the secret key used in the algorithm. The type of power attack of interest in this paper is known as a Differential Power Analysis (DPA) attack [2].

A DPA attack is possible when the power consumption of a circuit is correlated to the data which the circuit is processing. Data-dependent power information can be gathered when a circuit changes its state or switches because of a change in input. Imbalanced gates, glitches, and other switching activity can lead to data-dependent power consumption.

Asynchronous circuits often use 1-out-of-n data encoding to facilitate completion detection. In most applications, 1-out-of-2 or dual-rail encoding is used. Since in this encoding both logical zero and logical one are encoded with code words of the same Hamming weight, 01 and 10 respectively, they are intrinsically more power balanced than their single-rail counterparts. Delay-insensitive encoding is ensured by using a third logic value 00, which separates the data tokens. The use of this NULL value, or spacer, is often called a Return-To-Zero (RTZ) protocol since the gates return to a known state before switching to their next value. This separation of data with a third state has two consequences with respect to DPA:

- Every gate will switch for every data token, which gives information about the state of each gate at each data transition. (Assuming that the gates are in the logical datapath and there are no selective muxes.)
- The complexity of the system in terms of system transitions is reduced from about n^2 to n, where n is the number of possible input vectors to the system. (We further discuss this in Section 5.)

In this paper we show, with the use of simulation results, that the above two consequences of a RTZ protocol can reduce the number of traces needed for a successful DPA attack without the need for increased precision of measurements when compared to a dual-rail implementation without RTZ. We focus our analysis on symmetric key cryptosystems such as the Advanced Encryption Standard (AES) [7] and the Data Encryption Standard (DES) [6], but the results can be generalized to any circuit.

Evaluation of NULL-spaced asynchronous logic for security applications, including DPA, has been performed before with favorable results [1,3,4]. However, these evaluations were performed using microprocessors in which the register switching power was the dominant factor. Little has been done to evaluate the effectiveness of NULL-spaced logic in protecting against a DPA attack when it comes to special cryptographic accelerator hardware. Such hardware, as with implementations of DES and AES, is usually mostly combinational logic. We note that the authors in [5] do consider the use of spacer-based logic for an AES implementation. However, those authors did not perform an actual DPA attack; therefore, it is not clear if their countermeasures are actually effective.

The rest of the paper is organized as follows: In Section 2 we present some key characteristics of the DPA attack, and Section 3 shows our simulation setup and assumptions. In Sections 4 and 5, we present the results of the simulations and analysis respectively. Section 6 contains design for balancing dynamic, asynchronous, dual-rail gates with which the weaknesses of RTZ protocol is resolved. We also present the simulation result of a DPA attack on these balanced gates. Section 7 presents synthesis results of using RTZ protocols and an estimate of the balancing impact on the system performance. Finally, a conclusion is formed in Section 8.

2. DPA attack

A defining characteristic of a secure, well-designed symmetric key algorithm is that by knowing only the input and output (plain text and cipher) of the algorithm, it is not possible to determine the secret key significantly faster than performing an exhaustive search of the key space. As a result, a guess and test method is often used. A key is guessed, and then the guess is checked to determine if it satisfies the input-output relationship of the algorithm. Due to the "diffusion and confusion" principles of these algorithms, it is not possible to guess and verify a couple of the key bits at a time. It is typically an all or nothing guess and test. This aspect ensures the algorithmic security due to the very large total possible key space.

However, the power consumption of the circuit implementing the algorithm can be used to guess and verify a few bits of the key at a time. In many symmetric cryptosystems, such as the AES and DES, encryption (and decryption) contains small nonlinear substitution boxes with a small number of inputs as part of the algorithm. The computation performed by these substitution boxes is directly related to a small part of the secret key.

In the attack, no information about the actual circuit implementation of the algorithm is necessary. DPA uses the known algorithmic functionality of the circuit to predict an event whose occurrence will be manifested by an observable difference in power consumption. Such events can only be predicted accurately if the correct key is known. Observable events can be, for example, the switching of an imbalanced gate from a logical one to a logical zero. For DES and AES, the event that is typically predicted is the transition of an output/input bit of the substitution box.

Even though a difference in power consumption characterizes the targeted event, the difference cannot be directly observed by the power consumption curve since power consumption of other circuitry obscures the data. DPA overcomes this by partitioning the observed power curves into two sets based on the predicted event for each guessed portion of the key. One set contains the power curves for which the event occurred and the other is for the curves for which the event did not occur. If a suitable event (in terms of DPA) was targeted, only a guess of the correct key will predict the occurrence of that event completely accurately. No incorrect key guesses will be able to predict the event correctly. The normalized difference of these two sets for each key guess, known as the differential trace, should have the largest amplitude difference for the correct key portion guess. A guessed key is assumed correct if the magnitude of the differential trace is the largest since that large magnitude corresponds to the correct prediction of the targeted event. The algorithmic procedure for the analysis can be found in [18].

Thus, using DPA an attacker can test and verify a small portion of the key at a time, making cryptanalysis simple.

Because of this statistical nature of the attack, it is not correct to generalize resistance to DPA based on the shape, or relative lack of features, of a power trace of a circuit. That is, while it might be true that a flat power consumption curve makes a circuit resistant to Simple Power Analysis (SPA) [18], it is neither a sufficient nor a necessary condition for a circuit that is DPA resistant. The only sure way to provide resistance to a DPA attack is to ensure there is no correlation between power consumption and the processed data.

3. Simulation setup

To evaluate the effect of a RTZ protocol on DPA we felt that it would be best to perform a simulated DPA attack on a subcircuit of DES. We chose one substitution box (Sbox1) of DES on which to perform the simulations for several reasons. First, a simulated DPA attack requires full analog transistor level simulation. The computational cost of these simulations is very high and could not realistically be performed on a complete DES circuit. Second, a DPA attack on DES has been widely performed and has been fairly well perfected. Since the goal of these experiments was not to test how good our DPA analysis was but how well RTZ logic stands up to existing attacks, we felt that it was best to use well understood and developed procedures for which there is a substantial amount of precedence against which we can compare our results.

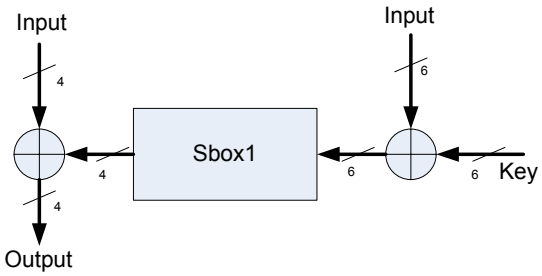

Figure 1. DES simulation subcircuit

The Sbox subcircuit used in the simulation has a 10-bit input text, a 6-bit key input (which is what a DPA attack tries to determine), and a 4 bit output (see Figure 1).

The DES subcircuit was implemented with synchronous single-rail gates, synchronous dual-rail gates (which could function with and without a NULL spacer) and with dual-rail asynchronous request-acknowledge fine-grained NULL-spaced static and dynamic gates.

The goal of designing the experiments was to evaluate the effect of RTZ protocol. RTZ affects how, when, and how often gates switch, all of which can have an impact on DPA since they affect the behavior of the targeted event. Clearly, an asynchronous dual-rail circuit with a RTZ protocol can be implemented in many ways. There are many design choices as to where, how, and with what granularity to do completion detection as well as many other design trade-offs. Our primary concern was to evaluate the effect of the RTZ protocol forcing each gate to return to a known third state between each data vector. Thus, for simplicity and generality we chose to do most of the analysis on a dual-rail implementation without completion detection that can be run with or without a RTZ protocol. This way both RTZ and non-RTZ simulations can be performed on the same gates with the same imbalances and the same circuit structures.

The simulations were also performed on a complete asynchronous dual-rail request-acknowledge NULL-spaced fine-grained pipeline implementation to confirm that the effects observed in the simple dual-rail RTZ implementation can be observed in an asynchronous glitch-free circuit with completion detection circuitry.

Since single-rail CMOS logic is known to be extremely vulnerable to a DPA attack, its simulation served as a baseline against which we could compare the other implementations.

For the synchronous single-rail implementation, the internal logic of the Sbox1 was synthesized from a table specification using Synopsys's Design Compiler and mapped into a simplified library which consisted only of inverters and two input NAND, NOR, and XOR gates. No special optimizations were performed. The synchronous dual-rail implementation was manually created from the synchronous single-rail by replacing each single-rail gate with its dual-rail equivalent. The dual-rail gates were specially constructed so that they would propagate a dual-rail input value of 00 (the NULL input). The asynchronous circuit was synthesized using the Weaver Asynchronous EDA flow from Boston University [8].

All gates were based on 0.18um CMOS technology. The simulations were performed on the transistor level using the Spectre simulation tool from Cadence. All simulations were performed under ideal conditions. That is, there was no noise and all measurements were assumed perfect. The motivation was that if DPA was not successful under ideal measurement conditions then it is highly unlikely that it would be successful under real conditions. Our experimental setup is very similar to that in [9].

We note that although we distinguish between synchronous and asynchronous, there is nothing synchronous about the 'synchronous' implementations. They do not contain registers or a clock. All the circuits are purely combinational. However, in all simulations we assumed that all inputs were synchronized so that all bits changed at the same time when going from one input vector to another.

4. Simulation results

As expected, the CMOS single-rail Sbox implementation quickly revealed the secret key in its differential power trace. The differential trace of the correct key was of the highest amplitude, meaning that the "event" was correctly predicted for that key. After 5000 random inputs, the correct key is clearly distinguishable from the other incorrect 63 keys (see Figure 2). Although there are only 1024 possible input vectors, there can be many more input pairs since any other vector can follow any vector. This creates different transitions in the circuit.

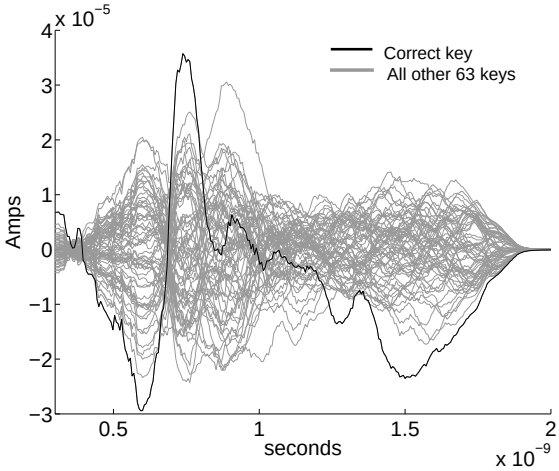

Figure 2. Differential traces of all 64 key guesses for single-rail gate Sbox implementation for 5000 random inputs.

The next simulation was performed on the simple dual-rail implementation without the use of a NULL spacer between the data. Although constant Hamming weight code is used for the dual-rail encoding, there is still a small power imbalance. As Figure 3 shows for a NAND gate, a small difference in energy can still be observed depending on from what state the gate switches.

The obvious improvement in using dual-rail encoding (even when not balanced) can be observed in Figure 4, where after 5000 power traces for random inputs it was not possible to distinguish the correct key (black line) from the other 63 possible keys. That is, the amplitude of

the differential trace for the key was not the maximum when compared to the others. If this differential trace was used to pick a correct key, it would be picked incorrectly since the correct key has amplitude which is far less than other incorrect keys. Clearly, even using a naïve dual-rail implementation is much better against DPA than single-rail. It may still be possible to successfully perform a DPA attack, but significantly more differential power traces would be needed.

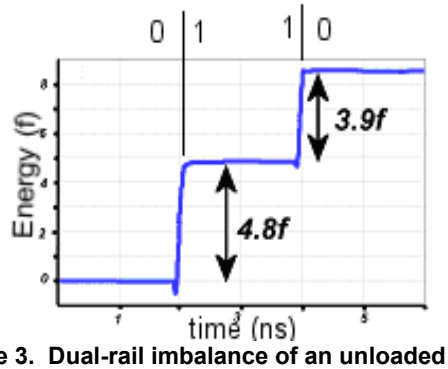

Figure 3. Dual-rail imbalance of an unloaded NAND gate. The current of the gate switching from a logical one to zero and a logical zero to one is integrated to find total energy consumed.

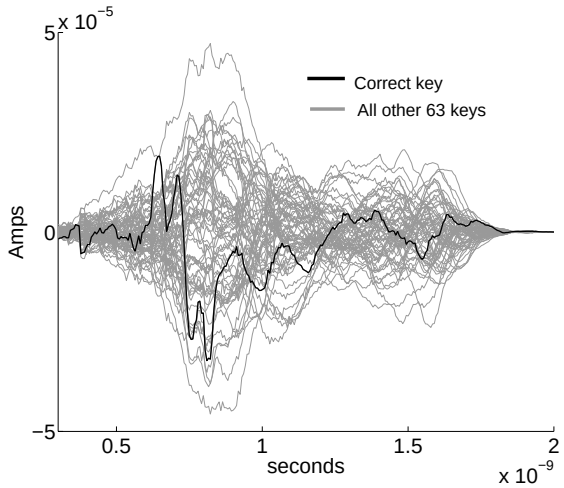

Figure 4. Differential traces of all 64 key guesses for dual-rail non-RTZ Sbox implementation for 5000 random inputs.

The same dual rail circuit was simulated again but this time with the addition of a NULL spacer (00) between input vectors. Unlike in the previous simulation where there were about one-half million possible input transitions, this NULL-spaced system becomes much simpler since each vector returns to a NULL input before the next input vector. There are only as many possible system transitions as there are input vectors, which in this case is 1024. Therefore, there are only 1024 different power signatures of this circuit. Performing DPA on all of the 1024 power traces resulted in a successful DPA attack (see Figure 5).

With only 1024 traces the correct key (black, thick line) is clearly distinguishable from all incorrect keys. Picking the key with the maximum amplitude in the differential trace would result in the correct key selection. Figure 5 shows the differential trace for the DATA to NULL transition. There is also the NULL to DATA transition for every input which can also be used to perform DPA on. We also note that the amplitudes of the successful dual-rail differential traces of DPA are almost identical to that of the completely unprotected single-rail implementation after 5000 power traces.

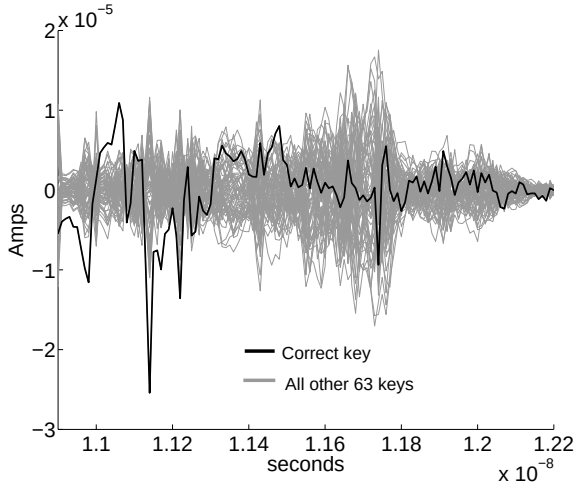

Figure 5. Differential traces of all 64 key guesses for RTZ dual-rail implementation. Shown is the DATA to NULL transition for all 1024 different data inputs.

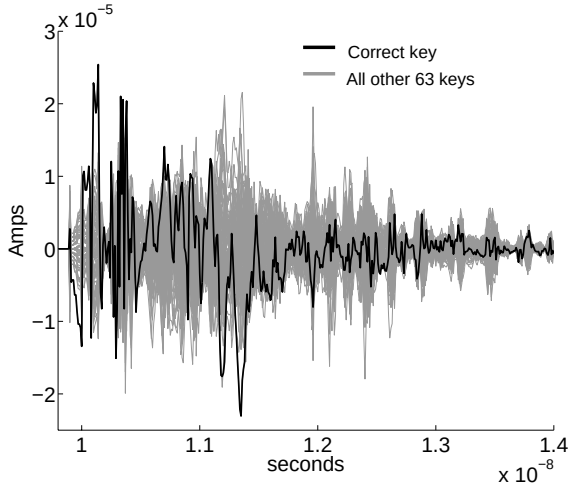

Figure 6. Differential traces of all 64 key guesses for the asynchronous fine-grained dual-rail NULL-spaced implementation. Shown is the DATA to NULL transition for all 1024 data inputs.

With the NULL-spaced dual-rail implementation, it is not necessary to use all 1024 power traces to see the correct key has maximum amplitude. Our simulations show the correct key starts becoming visible using as few as 500 different power traces.

The last simulations were performed on a complete asynchronous dual-rail NULL-spaced implementation to verify that what was observed in the simple dual-rail circuits is still applicable to a real asynchronous glitch-free circuit with completion detection. Performing DPA on the imbalanced standard cell static gate implementation of the circuit proved successful. Once again, the correct key's amplitude clearly distinguishes itself from all other incorrect keys after 1024 different power traces (see Figure 6). In addition, the amplitudes of the differential traces are comparable to those of the single-rail.

5. Analysis

DPA is directly related to the switching characteristics of a system. DPA uses a statistical method to correlate the power consumption, which is dependent on the switching of the circuit and the secret key. Adding an intermediate state between data clearly affects the switching and transitions of the circuit. Taking the DES subcircuit with 10 bits of input data as an example, without the RTZ protocol, a complete characterization of the system with respect to its transitions of input vectors would require $\binom{2^{10}}{2}$, or about half a million, power traces since the system can transition from any vector to any other vector. However, by adding the NULL intermediate state, the complexity of the system, in terms of the number of transitions required to characterize the whole system completely, is only $2^{10} = 1,024$ since the system will always transition to a known NULL state. In a sense, DPA is a characterization of a system by its transitions, hence an RTZ protocol simplifies a system making it easier to analyze.

In addition, not only does the RTZ protocol simplify the system in terms of its transitions, it ensures that all transitions provide useful information for DPA. Because of the third intermediate state all gates will switch at every input, providing information about their logic states which can be correlated with the secret key. Hence, the overall result is that while the transitional complexity of the system is reduced, the amount of information gained per transition is increased. This effect allowed us to perform a successful DPA attack with far fewer power traces on our DES test circuit using the RTZ protocol than in the circuit without the RTZ protocol.

We have shown that the RTZ protocol can be detrimental when considering imbalanced gates. If balanced gates were used then the use of asynchronous circuits with a RTZ protocol has some benefits. One very important benefit stems from the fact that gate imbalances are not necessarily the only sources of data-correlated power information. Since power consumption of a circuit depends on the number of switching gates, events such as glitches and a non-constant number of switching gates can potentially provide enough data correlated power differences to perform analysis. Both of these effects are inherently absent from an asynchronous circuit with a RTZ protocol. Balanced asynchronous NULL-spaced gates have the potential of having equal (but not necessarily constant) power consumption curves for all system transitions. Moreover, since DPA can be thought of as a characterization of a system by its transitions, having an equal power trace for all transitions yields no information and provides no distinguishable DPA events.

We demonstrated that an RTZ protocol can make a circuit more sensitive to power analysis and as a result places a greater importance on the need for balanced gates. We have developed a preliminary version of a balanced asynchronous dynamic dual-rail fine-grained NULL-spaced gate library for use in security applications. We present an overview of some considerations and structures used in the design of the balanced library in the next section.

6. Balanced asynchronous pipeline cells

Several asynchronous pipeline styles exist [12-17]. In this work we have chosen the simplest data-driven style presented in [17] for its minimal and local, or inside the stage, delay assumptions. This style also provides robust, or delay insensitive, inter-stage communication.

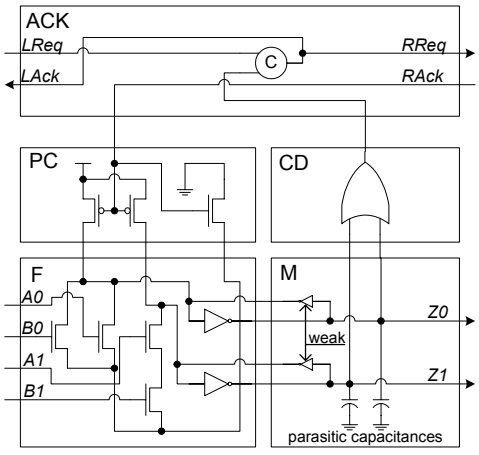

Figure 7. Dual-rail fine-grained pipeline template. F: Functional Block. PC: Pre-Charge Circuitry. CD: Completion Detection. ACK: Acknowledgement. M: Memory.

An example of an asynchronous dynamic fine-grained pipeline implementation of a two input imbalanced AND

gate is shown in Figure 7. Illustrated is the Reduced Stack Precharge Half Buffer (RSPCHB) template from [12]. The only block specifying the gate logical function is *F*. The rest is typical for most of the stages. *LReq* and *LAck* stand for left and *RReq* and *RAck* for right request (*req*) and acknowledgement (*ack*) signals. *ACK* stands for handshake circuitry, *PC* for phase (set/reset) control, *CD* for completion detection and *M* for memory. '*Staticizers*' (or *keepers*) formed by adding weak inverters, as shown in Figure 7, can store the stage output value for an unlimited time which eliminates timing assumptions. At the same time, keepers solve the charge-sharing problem and improve the noise margin of pre-charge style implementations. The *req* line is used to signal data availability to the following stages while the *ack* indicates that the data portion has been consumed. Depending on the communication protocol, some or all of the handshake events can be transmitted over the data lines so the *req* and/or *ack* lines may not even be needed.

To balance the circuit in Figure 7 with regard to power consumption for different data inputs, only the functional block, F, and the completion detection block, CD, need special design. All other blocks are either data independent or, as with the memory block, are already balanced.

6.1. Balancing the functional block

In [10], K. Tiri et al. proposed a design procedure for power-balanced dynamic and differential CMOS logic circuits that can balance the functional block of the gate. The authors present a Sense Amplifier Based Logic (SABL) AND-NAND gate with enhanced special Differential Pull Down Network (DPDN). We use this design to balance the function block of our asynchronous dynamic dual-rail AND gate with one significant change. To convert the SABL AND gate from synchronous to asynchronous, we use the input acknowledgement signal to trigger the pre-charge and evaluate stages, instead of a clock signal.

Figure 8 compares the result of balancing the AND gate. Shown is a plot of the standard deviation of the transient current supply for all four possible dual-rail inputs, for both the balanced (black line) and imbalanced (gray line) AND gate. DATA and NULL inputs were applied for 2ns each, resulting in a 4ns total cycle time for each token evaluation. We consider a gate more balanced when its standard deviation plot is of a smaller magnitude, because a smaller standard deviation plot means the current drawn from the supply was closer to a constant value for all time and for all possible inputs.

The balanced AND gate still has observable spikes just before 2ns, which is during the evaluation stage and just before the input is NULL. As Figure 8 shows, the gate is still has some minor imbalances in time.

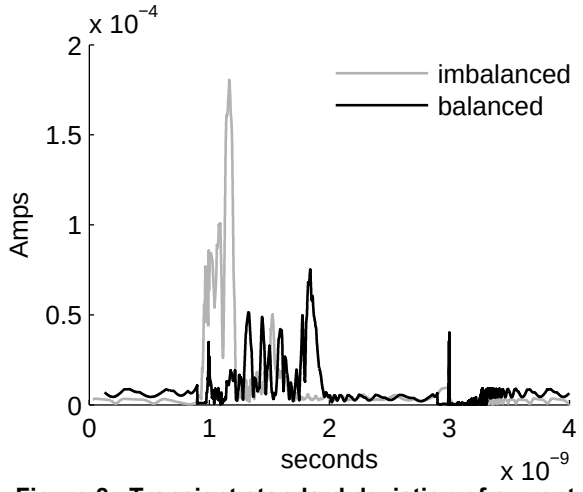

Figure 8. Transient standard deviation of current for all inputs to an imbalanced and balanced asynchronous dynamic dual-rail NULL-spaced AND gate.

We now show a balanced dual-rail XOR function block, which is different from the design style proposed in [10]. Figure 9 depicts an imbalanced dual-rail XOR gate, which includes the pre-charge (PC) logic from Figure 7. Assume all N-transistors in the evaluation stack, except for the footer transistor, have same sizes. The power characteristic of this imbalanced gate is dependent on the input value, since the different amount of capacitance charged and discharged corresponds to the input pattern. After pre-charge, all input signals are logical zero and all capacitors are charged. Suppose now both A1 and B0 switch to logical one. Then C_{L1} and C_1 are discharged while other capacitors are not. If, instead of switching A1 and B0, we switch A0 and B1, capacitors that discharge become C_{L1}, C_2, C_3 and C_4. The overall capacitance discharged in the latter case is different from the former, since according to our assumption, C_1, C_2, C_3 and C_4 are all equal. To overcome this imbalanced behavior, symmetric transistor-level design as in [11] is used. The authors in [11] originally proposed the symmetric design to counter the charge-sharing problem associated with Domino logic circuits, but it also allows an easy method to design a balanced gate. In Figure 10 we show a symmetric dual-rail XOR gate.

As with the imbalanced design, all N-transistors in the evaluation stack, except the footer transistor, have the same size. All transistors are made twice smaller than those in Figure 9. This keeps the circuit speed, area and capacitance nearly the same as in the original design, so roughly speaking, there is no additional cost for this improvement. If all internal parasitic capacitances are equal, there is no power difference between these two different switching activities. The same result holds when we compare the power consumed as different output rails switch. There will be always two internal parasitic capacitors discharged in the rail that stays unchanged during

evaluation. It is the balanced nature of the XOR function that allows us to apply this balancing technique, as opposed to the algorithm developed in [10] and used for the AND gate.

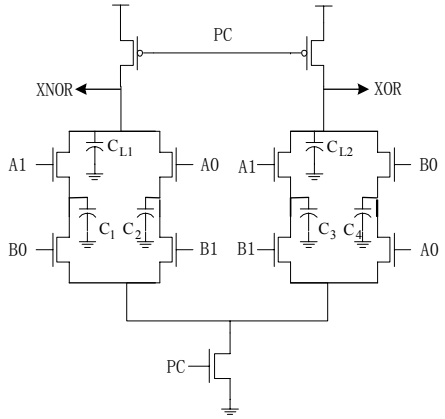

Figure 9. Imbalanced dual-rail XOR gate.

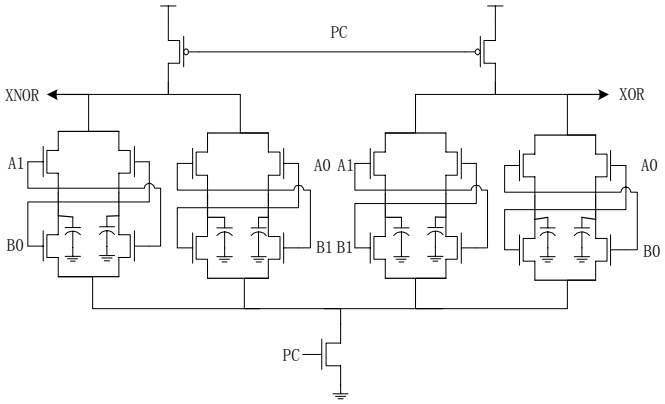

Figure 10. Symmetric dual-rail XOR gate (balanced).

Figure 11 shows the result of using the symmetric balancing technique for the XOR gate. As in Figure 8, the transient standard deviation of current drawn from the supply is plotted for both the imbalanced and balanced XOR gate for all possible dual-rail inputs. A smaller standard deviation magnitude means there was a smaller difference in current drawn for different inputs, so we consider the gate better balanced.

From Figure 11, the imbalanced XOR gate (gray line) has large spikes at 1ns, when the input is DATA and intermediate nodes can charge, and before 2ns, when the function block is evaluating. The balanced XOR gate (black line) greatly reduces all spikes, so it is better balanced. Thus, for all data inputs, the power consumed is nearly the same for all time and therefore reduces the observability of a potential DPA targeted event.

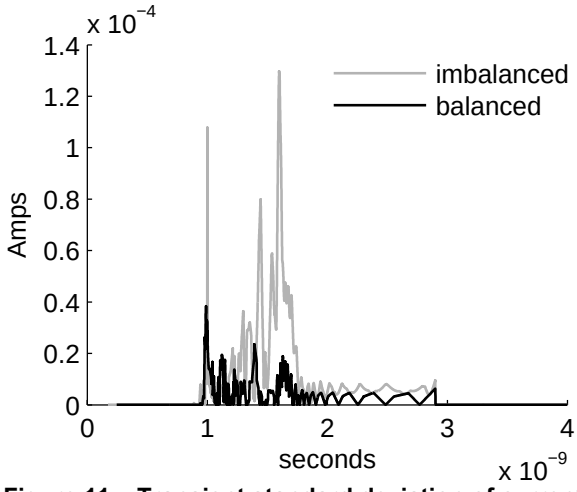

Figure 11. Transient standard deviation of current for all inputs to an imbalanced and balanced asynchronous dynamic dual-rail NULL-spaced XOR gate.

6.2. Balancing the completion detection block

A single OR gate in the completion detection block has different power consumption for different input values. To achieve the balancing goal we duplicate the P-transistor stack of the static NOR gate (see Figure 12) like we did for dual-rail XOR gate. Since for a DATA token, only either D1 or D0 is logical one, one and only one parasitic capacitor is discharged. Once again, P-transistors have twice-smaller sizes than original design.

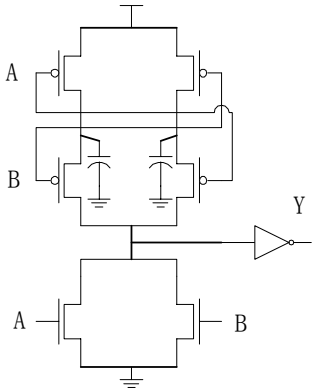

Figure 12. Balanced completion detection

Based on the approach we presented in this section, we developed a set of balanced logic gates. The approach carried out is a standard cell library design based on dual-rail fine-grained pipeline circuits. Each cell consists of an entire pipeline template as shown in Figure 7.

6.3. DPA on balanced asynchronous gates

After developing a library of asynchronous dynamic dual-rail NULL-spaced balanced-power gates, we performed another simulated DPA attack on a DES subcir-

cuit according to the simulation setup described in Section 3. Figure 13 shows the result of the attack, where the black trace is for a correct key guess, and the gray traces are for all other incorrect key guesses, for 64 possible key guesses. We applied all 1024 possible data inputs to generate the various differential power traces.

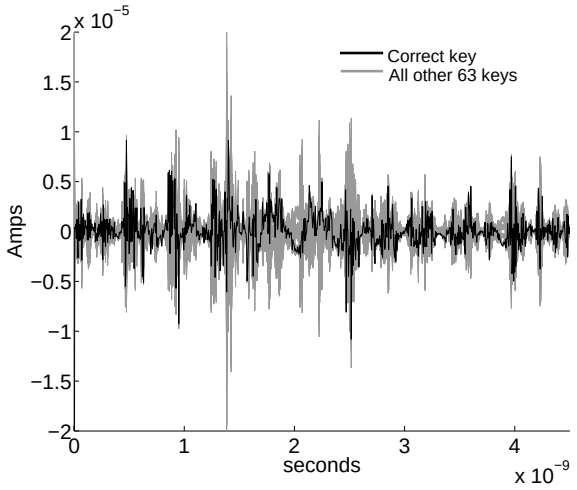

Figure 13. Differential traces of the DES subcircuit for all 64 key guesses using asynchronous dynamic dual-rail NULL-spaced balanced-power gates. Shown is the NULL to DATA transition for all 1024 data inputs.

From Figure 13, the black trace associated with the correct key does not clearly stand out from the rest of the key guesses. There is nothing which distinguishes the correct key from others. The balanced gate library therefore provides protection from a DPA attack. Furthermore, we derive these plots from all possible input combinations, which is possible because we always transition from the known NULL spacer to a DATA vector. Therefore, *applying more data inputs will not make the attack successful because the attacker will not obtain new data-correlated power information.*

We note that our balanced-power gate library successfully defends against a DPA attack despite not being perfectly power-balanced according to the standard deviation plots in Figures 8 and 11. This implies there is a degree of variation in power consumption that a circuit can tolerate and still be DPA resistant. A likely source of such slight variation could stem from variation in transistor fabrication. *Therefore, the difficult task of perfectly matching transistor sizes in a real-world circuit will not be quite as critical to ensure DPA attack protection.*

7. Why RTZ?

RTZ and the cells presented in Figure 7 can be used in so-called data-driven pipelines. The data=driven pipelines have no communication delay assumptions and as a consequence are more robust to variation (process, temperature, etc.). They also allow for pipelining automation capable of quickly producing very fast implementations.

Variation tolerance is possible because only cell-wide transistor variation can have effect on the operation correctness. Given the size of cells, such variation is practically negligible. Neither entire clock domains (as in the case of clocked designs) nor the stages (as in matched delay designs) have to be constrained to offset possible variation. As a result, the entire design performs as fast as its slowest stage under the given conditions (temperature, voltage, etc.).

The delay-insensitive communication allows for pipelining automation [8] and relaxes the placement and routing constraints (place and route constraints still remain important for performance and power balancing). Micropipelining provides very high performance. Our experiments and the experience of other groups from academia [12-16] and industry [17,21] show that the performance of fine-grain pipelined circuits is on the level or exceeds the performance of carefully designed synchronous implementations of the same designs. Micropipelining automation can provide reduction in time-to-market in addition to the performance improvement.

Figure 14 shows the performance of several 10 round implementations of AES, which demonstrate the performance benefits achieved with automatic pipelining using asynchronous delay-insensitive RTZ cells. The Figure 14 compares the performance of synchronous and asynchronous designs. The first four use iit018 – the standard cell library from the Illinois Institute of Technology [19] restricted to flip-flops and cells of up to 2 inputs to match the functional content of the asynchronous library. The implementations are synthesized with the Synopsys Design Compiler DC-Ultra. The AES10c implementation is purely combinational. AES10a is an automatically pipelined implementation of the same HDL code as AES10c using `pipeline_design` DC-Ultra command and constraining the clock period to zero as an option to optimize for performance. AES10mp is a carefully (manually) designed pipelined AES implementation presented in [20] synthesized using the DC-Ultra for the same library.

AES10fgp is the automatically fine-grain pipelined implementation using an asynchronous library based on the modified PCHB [12] template (mpchb018).

Finally the performance estimation of the same implementation using the balanced library blncd018 is shown in the same AES10fgp column.

The graph shows that automated pipelining (also called retiming) for synchronous designs is feasible and improves the design performance from 2.5 to 4 times over the original non pipelined design. Custom design is generally necessary to achieve the performance of 15x and higher over the non-pipelined implementation. The performance of asynchronous fine-grain pipelined designs exceeds that of even manually designed pipelines achiev-

ing 46x and 31.5x increase with unbalanced and balanced libraries respectively over the combinational non-pipelined implementation.

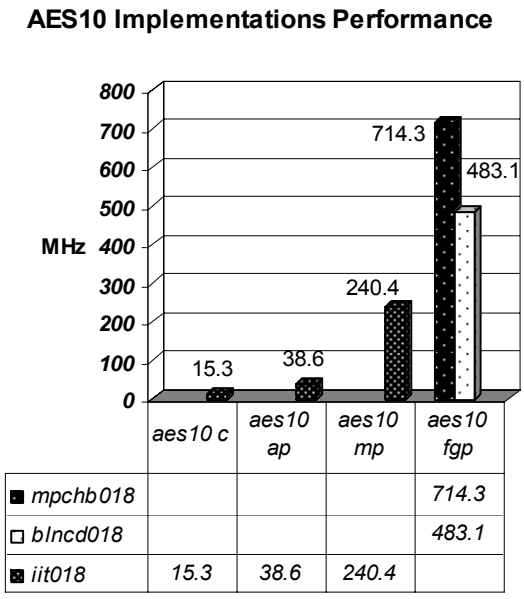

Figure 14. Performance of various 10 rounds AES implementations.

The performance benefits, robustness to variation and low design effort offered by micropipelines come at the price of significant area overhead. Balancing increases the area of unbalanced cells approximately by 20% and slightly decreases their performance. We have not characterized the library yet so the balanced library performance is just an estimate obtained by measuring the cycle time of a simple pipeline composed of cells specified as transistor-level schematics. The performance of the balanced cell designs remains greater than manually pipelined unprotected design.

8. Conclusions

We demonstrated that an RTZ protocol can make a circuit more sensitive to power analysis and as a result places a greater importance on the need for balanced gates. We provided evidence that power balancing for dynamic gates does not have to be a costly proposition. If balanced-power gates are used, then designing with asynchronous circuits using a RTZ protocol provides benefits. Glitches and non-constant gate switching could provide a predictable event for DPA. Asynchronous designs presented are inherently free of such potential problems.

By improving the balance of the gates a DPA attack was no longer successful. Because of the simplification of the number of system transitions due to the RTZ protocol it is easier to exhaustively determine the security of the circuit against DPA.

A balanced asynchronous library with cells which use a RTZ protocol can be used to automatically construct fast and secure implementations of cryptographic algorithms.

9. References

[1] J. Fournier, H. Li, S.W. Moore, R.D. Mullins, G.S. Taylor, "Security Evaluation of Asynchronous Circuits", *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, September 2003

[2] Paul Kocher, Joshua Jae, and Benjamin Jun, "Differential Power Analysis", *Lecture Notes in Computer Science* 1666, CRYPTO '99, pp. 388-397

[3] Z.C. Yu, S.B. Furber, L.A. Plana. "An Investigation into the Security of Self-timed Circuits". *Proceedings of ASYNC'03*, pp 206-215. Vancouver, May 2003

[4] Simon Moore, Ross Anderson, Robert Mullins, George Taylor, Jacques Fournier, "Balanced Self-Checking Asynchronous Logic for Smart Cards", *Microprocessors and Microsystems Journal*, 2003.

[5] Danil Sokolov, Julian Murphy, Alex Bystrov, Alex Yakovlev, "Improving the Security of Dual-rail Circuits". *Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2004

[6] National Bureau of Standards, "Data Encryption Standard", FIPS46, January 1977.

[7] FIPS PUB 197, "Advanced Encryption Standard", http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[8] Smirnov A., Taubin A., Karpovsky M. "Automated Pipelining in ASIC Synthesis Methodology: Gate Transfer Level". *IWLS 2004 Thirteenth International Workshop on Logic and Synthesis*. June 2-4, 2004. Temecula, California, USA.

[9] Kris Tiri and Ingrid Verbauwhede. "Securing Encryption Algorithms against DPA at the Logic Level: Next Generation Smart Card Technology". *CHES 2003. LNCS 2779*, pp. 125–136, 2003.

[10] Kris Tiri, Moonmoon Akmal and Ingrid Verbauwhede. "A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withstand Differential Power Analysis on Smart Cards". *28th European Solid-State Circuits Conference (ESSCIRC 2002)*

[11] Seok-Soo Yoon, Seok-Ryong Yoon, Seon-Wook Kim and Chulwoo Kim "Charge-Sharing-Problem Reduced Split-Path Domino Logic". *VLSI Design*, 2004. Proceedings. 17th International Conference.

[12] Ozdag, R.O. and P.A. Beerel, "High-Speed QDI Asynchronous Pipelines", in *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*. 2002. p. 13--22.

[13] Singh, M. and S.M. Nowick, “Fine-grain pipelined asynchronous adders for high-speed DSP applications”, in *Proceedings of the IEEE Computer Society Workshop on VLSI*. 2000, IEEE Computer Society Press. p. 111—118.

[14] Singh, M. and S.M. Nowick, “High-Throughput Asynchronous Pipelines for Fine-Grain Dynamic Datapaths”, in *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*. 2000, IEEE Computer Society Press. p. 198—209

[15] Sutherland, I. and S. Fairbanks, “GasP: A Minimal FIFO Control”, in *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*. 2001, IEEE Computer Society Press. p. 46—53.

[16] Williams, T.E. and M.A. Horowitz, “A Zero-Overhead Self-Timed 160ns 54b CMOS Divider”. *IEEE Journal of Solid-State Circuits*, 1991. 26(11): p. 1651—1661

[17] Cummings, U., A. Lines, and A. Martin, “An Asynchronous Pipelined Lattice Structure Filter”, in *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*. 1994. p. 126--133.

[18] Paul Kocher, Joshua Jae, Benjamin Jun, “Introduction to Differential Power Analysis and Related Attacks”, 1998. This paper is available at http://www.cryptography.com/resources/whitepapers / DPATechInfo.pdf

[19] J. Grad and J. E. Stine, "A Standard Cell Library for Student Projects," *International Conference on Microelectronic Systems Education*, IEEE Computer Society, pp. 98-99 2003.

[20] Verbauwhede, I., P. Schaumont, and H. Kuoand “Performance Testing of a 2.29-GB/s Rijndael Processor”. *IEEE Journal of Solid-State Circuits* 38(3): p. 569-572.

[21] Fulcrum Microsystems Inc. Web site: http://www.fulcrummicro.com/technology.htm.