A GALS Pipeline DES Architecture to Increase Robustness against CPA and CEMA Attacks

Rafael I. Soares[1,2], Ney L. V. Calazans[1], Victor Lomné[3,4], Amine Dehbaoui[3], Philippe Maurine[3], and Lionel Torres[3]

[1] Pontifícia Universidade Católica do Rio Grande do Sul, FACIN, PPGCC, Porto Alegre, Brazil.
[2] Universidade Federal de Pelotas, CDTEC, Pelotas, Brazil.
[3] LIRMM, UMR 5506, Université Montpellier II, CNRS, Montpellier, France.
[4] ANSSI, 51 boulevard de la Tour Maubourg, Paris, France.
e-mail: ney.calazans@pucrs.br

ABSTRACT

Side channels attacks (SCAs) are very effective and low cost methods to extract secret information from supposedly secure cryptosystems. The traditional synchronous design flow used to create such systems favors the leakage of information, which enables attackers to draw correlations between data processes and circuit power consumption, electromagnetic radiation or other sources of leakage. By using well known analysis techniques, these correlations may allow that an attacker retrieves secret cryptographic keys. Differential Power Analysis (DPA) and Differential Electromagnetic Analysis (DEMA) are among the most cited attack types. More accurate types of attacks have been proposed, including Correlation Power Analysis (CPA) that associates power quantities with a specific power model. In recent years, several countermeasures against SCAs have been proposed. Fully asynchronous and globally asynchronous locally synchronous (GALS) design methods appear as alternatives to design tamper resistant cryptosystems. However, according to previous works they use to achieve this with significant area, throughput, latency and power penalties and are not absolutely secure. This paper proposes a new GALS pipeline architecture for the Data Encryption Standard (DES) that explores the trade-off between circuit area and robustness to SCAs. Robustness is enhanced by replicating the DES hardware structure in asynchronously communicating module instances, coupled with self-varying operating frequencies. Designs prototyped on FPGAs with the proposed technique presented promising robustness against attacks, after submitted to differential and correlation analyses. This is true for both power and electromagnetic channels. Additionally the proposed architecture displays throughput superior to previously reported results.

Index Terms: Cryptography, cryptographic attacks, secure cryptography, globally asynchronous locally synchronous.

1. INTRODUCTION

Modern cryptographic algorithms are designed to be secure against known mathematical cryptanalysis techniques. Also, due to the length of the key and current computer processing power, cracking them with brute force may take a quite long amount of time. However, another attack class, side channel attacks (SCAs), can recover sensitive information leaked from physical quantities such as processing time, power consumption and electromagnetic radiation.

Power consumption analysis attacks, a class of SCAs first described by Kocher [1] [2] are powerful, as they do not require expensive resources and as most implementations without specific countermeasures are vulnerable to such attacks. Differential Power Analysis (DPA) is particularly interesting, as it uses simple statistical techniques that are almost independent of the cryptographic algorithm implementation. Differential Electromagnetic Analysis (DEMA) follows the same principle, but measures another physical leakage. Also, the later allow applying multiple measures on specific parts of the attacked circuit. An improvement to DPA attacks is the Correlation Power Analysis (CPA) [3] that relates real power consumption to a power consumption model. The same is applicable to electromagnetic measurements, leading to a technique known as Correlation Electromagnetic Analysis (CEMA) [4] [5].

It is well known that building cryptographic circuits using fully synchronous design methods facil-

itates the task of successfully attacking these with DPA or DEMA. Thus, non-synchronous circuits seem an interesting way to implement countermeasures against SCA, because they may render more difficult to correlate the leaking syndromes to the data flow. This occurs due to the absence of a global synchronization signal. It is easier uniformizing the power consumption profile of an asynchronous circuit than do the same for an equivalent synchronous circuit. This property makes asynchronous design methodologies attractive to designers looking for methods to reduce the vulnerability of cryptographic hardware against DPA/DEMA/CPA/ CEMA attacks. However, asynchronous design often requires custom libraries, usually causes significant area overhead and suffers from the lack of tools to automate its design process.

The globally asynchronous locally synchronous (GALS) design methodology, first proposed by Chapiro [6], allows combining the advantages of asynchronous operation with the convenience of standard synchronous design. In GALS, locally synchronous modules (often called *synchronous islands*) that are designed using mature design tools provide the overall system functionality. A module in a GALS system encapsulates a locally synchronous island within asynchronous interfaces. These interfaces provide mechanisms that govern the communication between GALS modules.

This paper proposes a new architecture to increase robustness of cryptographic hardware to DPA/DEMA/ CPA/CEMA attacks. The architecture is the same described by the authors in [7], but results described in that publication are extended here to encompass new kinds of attacks (CPA/CEMA) and comparison to STTL, an asynchronous logic deemed for increasing robustness to SCAs. The assumptions to achieve robustness in this work are the use of GALS design methods and asynchronous pipeline implementations. The method has been applied to a DES crypto-module, one of the best known cryptographic ciphers. The method can promptly be applied to similar algorithms like AES. The asynchronous pipeline architecture may hide information leakage through side channels like power consumption and electromagnetic radiation. In the approach proposed here a DES round block is replicated and each block constitutes a synchronous island within an exclusive clock domain.

The rest of this paper is organized as follows. Section 2 explores related works on SCAs, proposing a classification for SCAs. Next, Section 3 reviews the DES cryptographic algorithm, while Section 4 describes the proposed architecture. Section 5 presents a proof of concept implementation of the architecture on FPGAs and provides comparisons with other approaches to DPA/DEMA and CPA/CEMA countermeasures. Finally, Section 6 provides conclusions and directions for further work.

2. SCA RELATED WORK

SCAs take advantage of implementation characteristics to recover the secret information involved in computations, such as the cryptographic key, for example. SCAs are therefore less general they need to be adapted to specific algorithm implementations. But they are often more powerful than classical cryptanalysis and considered very seriously by cryptographic designers.

A. SCAs Classification

The literature usually classifies SCAs according to two orthogonal criteria:

- Invasiveness - invasive attacks require unpacking the device to obtain direct access to its components. A non-invasive attack only exploits externally available information such as running timing, power consumption and electromagnetic radiation. Skorobogatov and Anderson [8] proposed an intermediate classification called semi-invasive attacks. They require chip unpacking, but no direct contact to the die surface is necessary.
- Activity - active attacks manipulate the inputs and/or the environment of the cryptographic device to make the device behave abnormally. On the other hand, passive attacks only observe the behavior of the device during its (normal) processing.

Invasive attacks typically require expensive equipments such as sensitive electronic probing stations. An introduction on probing attacks can be found in [9]. On the other hand, most non-invasive attacks can be conducted with relatively inexpensive equipment. Passive non-invasive attacks have received a lot of attention during the last years. These attacks are often referred to as *side channel attacks.* The three most important types of SCA are timing attacks [1], power analysis attacks [2] and electromagnetic attacks [10].

The simple power analysis (SPA) [2] is a visual inspection using only one (or very few) power consumption signals measured during the execution of cryptographic operations. The differential power analysis is a statistical test which examines a large number of power consumption traces to retrieve secret keys. DPA can be developed in different forms. It can be performed by analyzing the intermediate values of one bit [2] or a set of several bits [11]. It can be observed at one instant of time [2] or at some instants of time in what is called High Order DPA [12]. In recent years, the correlation power analysis (CPA) technique, based on associating real power consumption of the device and a power consumption model has been widely studied [3] [4]. It has been demon-

strated that CPA can be conducted as a form of DPA divided by a normalization factor [4].

Electromagnetic radiation signals acquired by dedicated sensors were also successfully used to detect secret information [10]. Similar to power analysis, electromagnetic analysis can be performed using the same technique. In this case, it is called Differential Electromagnetic Analysis (DEMA). Figure 1 depicts an overall classification of most current power and electromagnetic attacks.

The goal of all attacks is to determine the secret key of a cryptographic device, by measuring its execution time, its power consumption or its electromagnetic field.

B. Differential and Correlation Power Analysis

There are many variants of power analyses (PAs). The basic idea behind a PA is to find relationships between power consumption and data processed in a circuit. Power analysis attacks have been developed in many forms. They exploit the dependence between the instantaneous power consumption of a cryptographic device and the data it processes or the operations it conducts.

In the most basic PA, known as Simple Power Analysis (SPA) [1], the attacker monitors the power supply of a crypto processor and correlates the time domain waveform with various operations of the algorithm such as shifts, branches, multiplications, additions and others which can be identified in the power consumption signature. However, SPA is limited and only effective to crack naive implementations.

Differential Power Analysis (DPA) [2] is significantly more powerful. It is effective even for some encryption algorithms that do not require knowledge of the plain data. To perform a DPA, an attacker needs a collection of m power traces $T[i][j]$, with $i=1,..,m$ (and where j is the discrete time index of the values in the trace), and their corresponding ciphertext values $C[i]$. The next step is to define a selection function $D(K_b,C[i])=\{0,1\}$ that can, given subkey K_b consisting of a small subset of b key bits, split the set of m traces and cipher data values in 2 different subsets. The definition of the D function depends on the encryption algorithm, and it is a critical step in a successful DPA attack. On a symmetric key cipher like DES, the selection function D is usually applied on the Sbox output bits. Sboxes are one of the essential processing modules in DES. Section 3 of this work describes DES in some detail.

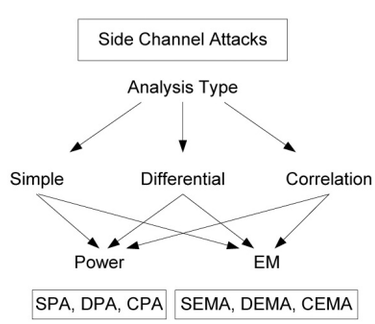

Figure 1. A classification for SCAs.

A DPA proceeds as follows. It assumes a value of the subkey K_b, applies the selection function to partition the power traces $T[i][j]$ in two disjoint subsets, D0 and D1, such that $T[i][j] \in$ D1 if $D(K_b,C[i]) = 1$ and $T[i][j] \in$ D0 if $D(K_b,C[i]) = 0$. Then, it computes the average trace for the two subsets, and their difference $\varnothing[j]$. The attacker then analyzes $\varnothing[j]$, called the differential trace. If $\varnothing[j]$ looks as a mostly horizontal line, then the subkey hypothesis is wrong. If it shows visible peaks, then, with very high probability, the subkey K_b has been found. Once the subkey has been found, one applies the attack on the next subkey until finding all subkeys.

Correlation power analysis (CPA) exploits the relationship between the power consumption P of a device and its power consumption model M [3]. The most common models have linear form, such as the Hamming weight[1] model (HW) or the Hamming distance[2] model (HD). Given a cryptographic key k, the correlation factor between P and M is proportional to the correlation factor between P and HD or HW. Equation (1) gives the correlation factor ρ of CPA.

$$\rho_{PH_k} = \frac{E(P.H_k) - E(P).E(H_k)}{\sigma_P \sigma_{H_k}} \qquad (1)$$

Here, $E(P)$, $E(H_k)$, $E(PH_k)$ represent respectively the expectations of P, H_k (the values of HW or HD estimated for the key k) and $P.H_k$. σ_P and σ_{Hk} are the variances of P and H_k.

Meynard et al. [5] propose a pre-characterized leakage model able to parameterize the electromagnetic radiation from a distance as far as 50 cm. CEMA may be improved to become a more accurate analysis. Réal et al. [13] propose a method to detect a hot spot onto the attacked device. This method is able to find the best positions to locate the electromagnetic probe, to make attacks more easily successful.

C. SCA Countermeasures

Several proposals to counteract DPA attacks are available in the literature. Basically, these countermeasures can be classified in three different

[1] The Hamming weight of a bit vector B is the number of bits '1' in B.
[2] The Hamming distance between two bit vectors B and S of equal length is the number of bit positions where B and S differ.

approaches. The first approach, *uniformization*, proposes to keep power consumption uniform and data independent. Usually, new logic styles are proposed to reach this goal. Some of these are WDDL [14], ADLBL [15] and STTL [16]. All of them are examples of asynchronous dual rail precharge logic (DPL). They employ dual rail encoding and a two-step execution process, precharge and evaluation [17]. These asynchronous circuit approaches have high cost in terms of area and latency and incur in complex design flows. Moreover, they are often very sensitive to the physical place and route steps, which may introduce unbalancing on wire delays. Differentiated delays may cause non-uniform power dissipation, which can significantly decrease robustness [18].

The second approach, *masking*, transforms data and secret keys to hide leakages on cryptosystems. This approach can be applied at different design levels as demonstrates the available literature [19] [20] [21] [22]. Goodwin and Wilson [19] proposed to modify the cryptographic algorithm AES itself. Ghellar and Lubaszewski [20] proposed a new arithmetic data encoding based on finite fields. Golic [21] suggests the use of improvements at gate level design. This author proposes a new structure to synthesize XOR gates to avoid logic unbalancing. Mesquita et al. [22] describe a method based on a reconfigurable architectures to implement the RSA algorithm, dynamically changing the way modular exponentiation is conducted. A clear drawback of this approach is its very high cost in area and latency.

The third approach to countermeasure DPA is *randomization*, i.e. to introduce noise on the power consumption signature of cryptographic applications, by either adding extra hardware or random processing to make impractical applying DPA attacks. Some works suggest specific methods to achieve randomization. Lu et al. [23] proposed to optimize the random delay insertion (RDI) on combinational circuits to improve DPA resistance. This intuitive method may be overcome by specialized DPA attacks such as sliding windows DPA [24] and phase matching DPA [25]. Zafar and Har [26] proposed a similar method. These authors developed a hopper clock generator, able to produce random frequencies for each processed data. This method suffers from the same limitations of optimized RDI, because the whole algorithm is still run with a single frequency.

Kamoun et al. [27] proposed a power noise generator for a cryptosystem using AES. The generator includes the most vulnerable AES functions. The input messages are concurrently processed by AES and the noise generator, but with different keys. Attack results demonstrate the method provides protection only to the first round of AES, the last one remains unprotected.

Standaert et al. [28] proposed the use of a pipeline architecture to counteract DPA attacks. The case studies reveal significant improvements to reduce information leakage, but successful attacks on it have already been reported by the authors themselves, although the results display a much smaller margin of certitude that the breaking of the secret key succeeded. Again, it is possible to suggest that the breaking of the key happens here because the whole encryption operates with a constant clock frequency.

Gürkaynak et al. [29] were the first to propose the use of a GALS methodology to counteract DPA. The authors partitioned an AES round on two blocks, one of which is replicated. Every module has a proper random clock domain and communicates asynchronously with the other. Different countermeasures are proposed including the use of random clocks, out of order execution, parallel execution and fake data executions. The method significantly increases latency and has a high area cost. Also, no attacks have been reported on the proposed architecture, making it difficult to assess its robustness.

This revision shows interesting trends to hide information leakage. The use of fine grain partitioning of the cryptographic algorithm appears as an alternative to counteract DPA/DEMA [28], but the approach is still vulnerable when using a single synchronous domain. References [26] and [29] use several clock frequencies, but each data undergoes encryption with a unique clock frequency, which is still ineffective against DPA. The work proposed in [29] uses fine grain partitioning with a random clock per partition, but the several alternatives to mess up the identification of the power signature present high latency and area costs. The proposal of this paper consists in partitioning the cryptographic algorithm at the round level, coupled to a random frequency choice at each round and for each data to encrypt. Also, successive pieces of data are processed in a pipelined way along the encryption hardware.

3. THE DES ALGORITHM

The Data Encryption Standard (DES) specifies a FIPS approved cryptographic algorithm as required by FIPS 140-3. Tuchman [30] provides a complete description of a mathematical algorithm for encryption and decryption of binary coded information. Encryption converts it to an unintelligible form called *ciphertext*. Decryption converts the data back to its original form, called *plaintext*. The algorithm described in the standard specifies both encryption and decryption operations, which are based on a binary number called *key*. The cryptographic security of the data depends on the security provided to the key used to cipher and decipher the data.

The choice of the DES algorithm is justified, since it has been extensively studied [1] [2] and requires smaller area than its successor, AES, when implemented in hardware. One of the domains where it is extensively used is in smartcards. Figure 2 shows the general structure of DES. It works on 64-bit input data blocks. Each data block passes through 16 rounds of modification. Each round uses one distinct 48-bit key, two 32-bit data inputs in operations that include permutations (P), bit expansion (E), shifting, XORing (+) and substitution functions (SBOXes). After executing the 16 rounds, the algorithm produces a final result, which suffers an inverse permutation and becomes the 64-bit output ciphertext.

The DES algorithm described in Figure 2 can be implemented in either software or hardware. A regular DES implementation creates a single round of the algorithm (in software or hardware) and iterates over it 16 times with adequate parameterization.

Figure 3 depicts a power trace of the synchronous processing using DES. Observing the trace, it is quite simple to identify the operations sequence. First, the initial permutation is executed, followed by 16 rounds, and ending with the inverse permutation. Kocher et al. [2] demonstrate the vulnerability of synchronous implementations and the effectiveness of DPA.

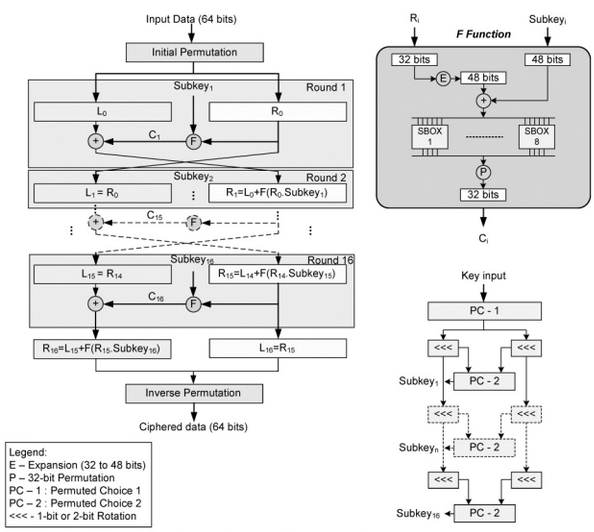

Figure 2. Overall structure of the DES algorithm.

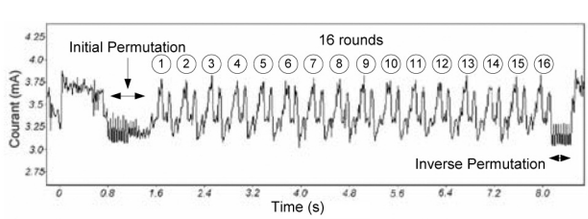

Figure 3. Current trace corresponding to the DES algorithm data processing [1]. Its characteristics are the source of information leakage to avoid or mask for obtaining a robust cryptography.

Clearly, the multiple rounds structure allows implementing the algorithm in pipeline mode. Thus, it is possible to have hardware architectures with from 2 to up to 16 stages to execute the complete algorithm. As an advantage of this pipeline implementation, the cipher data throughput can increase substantially. The downside of the approach is clearly its higher cost in area.

4. PROPOSED ARCHITECTURE

This work proposes a new architecture to build encryption hardware. The basic idea is to employ GALS pipeline implementations using the architecture depicted in Figure 4. This architecture assumes as basis the replication of round hardware or some other *elementary module* of the encryption algorithm. A certain number of consecutive elementary modules are encapsulated inside a synchronous island. The complete encryption hardware is formed by **n** stages. The number of stages **n** is a specific design decision, not a characteristic of the algorithm or of the architecture. The contents of the synchronous islands are also specific design decisions. Synchronous islands may be identical or distinct. When dealing with the DES algorithm, we may naturally opt for a single round as the elementary module. Based on this choice, numerous implementations may be deemed interesting to randomize leakages. For example, we could implement a 3-stage pipeline (**n**=3) where the first stage executes the first 5 rounds, the second stage executes the next 4 rounds and the last stage executes the remaining 7 rounds of DES.

Each pipeline stage is wrapped by an asynchronous interface and managed by a stage-internal finite state machine (FSM) to communicate point to point through a 2-phase handshaking protocol with its neighbor islands and/or the external world. A subsystem external to each pipeline stage (the Clock Subsystem) supplies a random frequency clock signal to the synchronous island. It generates a new clock frequency whenever a stage finishes processing one data instance. The structure of the Clock Subsystem can be constructed in different ways. External crystal

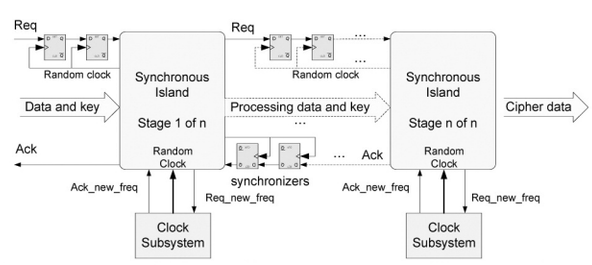

Figure 4. Proposed GALS pipeline architecture to build cryptographic processors.

oscillators and internal ring oscillators are two possibilities. Figure 4 depicts the asynchronous interface scheme. The simple 2-flop synchronizer is used to allow communication between distinct synchronous islands. The main assumption of the synchronizer is that the time reserved for metastability resolution provides a satisfactory mean time between failures (MTBF) [17].

Synchronizers do introduce a timing penalty, which increases the cryptosystem latency. After activating one request signal, two receiver clock cycles are necessary before data processing starts at the next stage. Since each clock subsystem changes its frequency for each new data, the processing instant is randomized at each stage. This in turn increases processing jitter, which makes difficult defining power and electromagnetic circuit signatures.

The cornerstone of the proposed approach is the Clock Subsystem that produces a new frequency for each new data processed. This is intended to improve the security by masking the structure of the cryptosystem information leakages. Some of the previous works discussed in Section 2 have proposed approaches with this same objective. In [23] masking is attempted by inserting random delays in cryptosystems' combinational logic, while in [26] each data is processed with a distinct random clock. The authors of [28] suggest the use of pipelining to implement the cryptosystem, but with a fixed clock. Finally [29] proposes the use of GALS design techniques in cryptosystems for the first time, but without using pipelining. To the knowledge of the authors the present work is the first to use a GALS implementation based on randomized clock frequencies coupled to a hardware pipeline structure.

How the Clock Subsystem is implemented is not a fundamental concern for the architecture this work proposes. However, for the sake of clarity, the next Section discusses an implementation of the complete system, including details of a working Clock Subsystem.

5. PROOF OF CONCEPT

The proposed method requires replicating hardware to counteract SCAs. In fact, the parallel processing of the rounds produces a noisy environment that makes SCAs more difficult. On the other hand, the hardware replication in a pipeline naturally improves the cryptosystem throughput. This Section provides a comparison of the proposed method with some previous works concerning area, throughput and robustness. The results presented herein, together with other results of this same work published elsewhere [7], evidence that this method does improve security.

A. Implementation on FPGAs

Figure 5 depicts a data producer-consumer architecture designed to conduct the robustness experiments and area comparison on FPGAs.

The architecture implemented in an FPGA contains a synchronous island responsible to receive data from and transmit data to an RS-232 serial port. This island feeds the GALS pipeline DES with plain data and receives cipher data from it. This is a synchronous module operating at a fixed frequency of 50MHz. It may use a circular FIFO to keep the pipeline stages full, independent of the serial communication rate. It is possible to conduct experiments with or without the FIFO. The former case allows evaluating the noise yielded during the parallel processing more accurately. Figure 5 also shows a trigger signal (called **Trigger to scope**) produced by this island, which is activated to fire the measurement process by an external oscilloscope for a single data processing action. Without the FIFO, it is possible to evaluate the effect of using the local random clocks in the cryptosystem separately.

The remaining modules of the implementation are the replicated DES rounds. The structure employs an external Switch Control, to turn On/Off GALS operation. When the switch is ON, the architecture behaves like a synchronous pipeline DES. Otherwise, each pipeline stage operates at their own, changing frequencies. This feature allows evaluating the proposed architecture against equivalent pipelined synchronous implementations. It also enables to compare it with other synchronous implementation, using exactly the same floorplan to avoid physical synthesis variations. The general architecture has been prototyped in four versions, respectively with two, four, eight and sixteen stages. These are respectively called GALS PIPE 2, 4, 8 and 16. In the GALS PIPE 2 implementation, the round hardware is replicated twice and each of these executes 8 rounds of the DES algorithm. Similar reasoning applies to the other implementations, mutatis mutandis. For each GALS

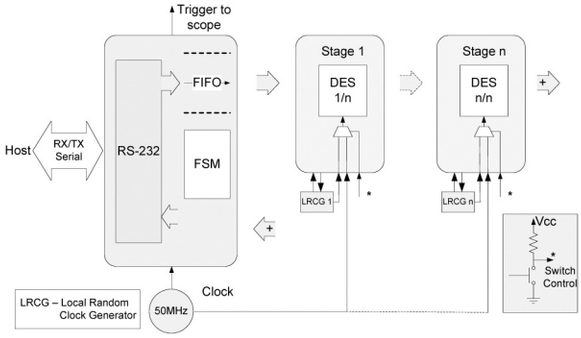

Figure 5. General proof of concept structure to evaluate robustness of GALS pipeline DES architectures on FPGAs.

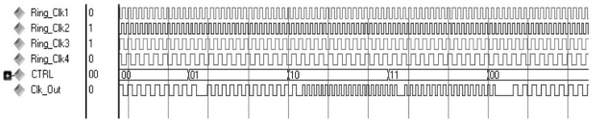

Figure 6. Waveform showing the glitch free clock switching behavior of the random clock generator.

DES X implementation, all stages of the pipeline are identical, which implies a worst case of the architecture concerning robustness, due to the symmetry of the processing.

A local random clock generator drives each stage. It produces always one out of four possible operating frequencies. The clock generator comprises four oscillators that are simply ring delay chains with random size. An essential glitch free multiplexer [31] is used to switch among clock signals. An LFSR module is used to generate pseudo random data to choose every next frequency. Figure 6 shows a timing simulation of the glitch free switching operation of this module.

B. Area comparison

Although the proposed method presents significant area overhead, due to the hardware replication process, this implementation type is compatible and usually better than for example DPL methods, which require much more area to implement circuits on dual rail logic. Besides, the latter use a complex design flow and custom cells.

Table 1 presents a comparison between the prototyped GALS pipeline architectures and STTL [16], a state-of-art DPL style implementing the full DES algorithm. It is observable in the Table that the method proposed here requires less area than STTL in most versions, but suffers an additional area overhead of up to 25 times over a regular, synchronous and lumped DES implementation when compared to the GALS PIPE 16 hardware. Area reports refer to an XC2V4000 Virtex2 Xilinx FPGA.

C. Latency and throughput comparison

Table 2 presents latency and throughput results and a comparison with other approaches. The synchronizers used to communicate data between synchronous islands increase the encryption latency of the cryptosystem. Moreover, there is no buffering inside the stages, which could decrease latency. On the other hand, the more replicated the round hardware is, the greater the achieved improvement in terms of throughput. Also, increased replication produces increased noise to hamper SCAs.

As stated before, the experiments described here employ ring oscillators built with ordinary FPGA devices as frequency generators. Frequencies are chosen so that it is a rare event that two distinct stages operate at the same frequency. In the experiments, the minimum clock frequency is 7.2 MHz and the maximum 21 MHz. Despite the low frequencies used in the experiments, the maximum operating frequency, as estimated by the synthesis tool is around 100 MHz. Thus, to estimate the throughput limits, comparisons are based on the whole architecture executing at minimum and maximum frequencies and at 100MHz, as depicted on Table 2. Hardware replication has showed no significant effect on the maximum operating frequency of all pipeline versions.

D. Measurement setup

To validate the GALS pipeline DES robustness against power and electromagnetic analyses, the employed measurement setup comprises six elements: (1) A Digilent Spartan-3 board with a XC3S1000 Xilinx FPGA, (2) a 500µm magnetic probe, (3) a low-noise amplifier (1GHz bandwidth – 63 db), (4) a positioning XY table, (5) an Agilent Infinium DS80000B Oscilloscope (4GHz – 40 GSa/s) and (6) a PC running MATLAB scripts to control the whole measurement setup.

E. DPA/DEMA results

Initially, only the GALS PIPE 2 architecture version has been submitted to DEMA. In order to perform electromagnetic analyses, EM traces were

Table 1. Area comparison for selected DES implementations, for a XC2V4000 Virtex2 Xilinx FPGA.

Architecture	Slices	Device Area	Overhead w.r.t. Regular DES
Regular DES	267	1 %	1
GALS PIPE 2	935	4 %	3.5
GALS PIPE 4	1830	8 %	6.85
GALS PIPE 8	3605	16 %	13.5
GALS PIPE 16	6614	29 %	24.77
STTL	5130	22 %	19.21

Table 2. Latency and throughput comparison on selected DES implementations. Regular DES is synchronous, lumped.

Architecture	Latency* (cycles)	Throughput (Mbps) f=7.2Mhz	f=21Mhz	f=100Mhz	
Regular DES	17	20.9	61.1	290.9	-
GALS PIPE 2	24	21.8	64	304.7	-
GALS PIPE 4	40	27	79	376.4	-
GALS PIPE 8	61	32.9	96	457	-
GALS PIPE 16	109	35.4	103.4	492.3	-
STTL	NA	NA	NA	NA	14.3
Gürkaynak [29]	Unknown	Unknown	Unknown	Unknown	256**

NA – Not applicable.
* For GALS versions numbers represent minimum values.
** Authors do not detail the measurement setup, but mention the use of three clock domains, one at 190MHz and two others at 250MHz.

collected on the regular DES and on the 2-stage pipelined GALS DES (GALS PIPE 2). More precisely, 100,000 electromagnetic traces have been collected on both implementations.

Next, the multi-bit DEMA described by Bevan and Knudsen [11] has been applied on the two sets of traces. As expected, less than 6,000 traces were sufficient to disclose the entire key of the regular DES. But for GALS PIPE 2 the attack did not succeed, even after having processed the 100,000 traces. Not even a single subkey has been cracked. Figure 7 shows the differential traces computed with 6,000 traces for the regular DES. They correspond to the guess of subkey 1. The differential trace corresponding to the good hypothesis is drawn in black (darker trace), whereas the others are drawn in cyan (lighter traces).

Figure 8 depicts the differential traces computed with 100,000 samples for the GALS PIPE 2 architecture.

As it is possible to observe, the differential trace corresponding to the good hypothesis, drawn in black (darkest trace), is not the differential trace with the highest peak. Another differential trace, drawn in red (medium tone trace), has the greatest peaks, corresponding in fact to a wrong key hypothesis that would be suggested by an attacker as the correct one.

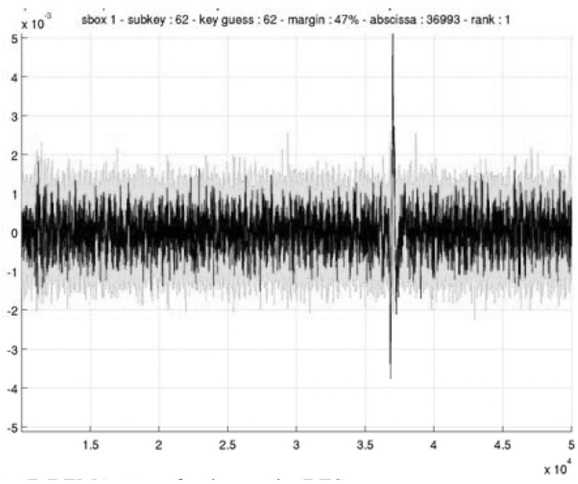

Figure 7. DEMA traces for the regular DES.

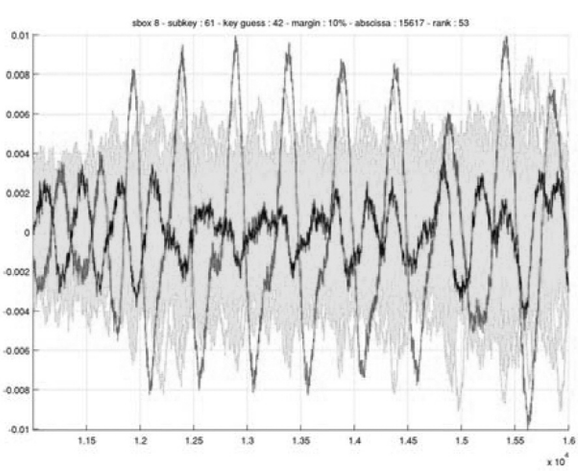

Figure 8. DEMA traces for the GALS PIPE 2 architecture.

Next, the GALS PIPE 4 and 8 implementations have been submitted to DEMA using the same measurement setup. As expected, only the synchronous pipeline architectures were vulnerable to the attacks. None of the GALS architectures have security compromised. Table 3 abstracts the results of the DEMA attacks. The Table shows the number of traces needed to find an entire key for DES, when applicable.

From an attacker's point-of-view, it is possible to explain these results by two reasons: (1) islands work in parallel, which decreases the relation between the processed data and the EM leakage and (2) islands work at distinct frequencies, making attacks succeed less often.

F. CPA/CEMA results

CPA/CEMA attacks have been applied to the same architectures. Table 4 presents the results obtained with this analysis on the synchronous pipeline implementations. Note that all synchronous pipeline architectures show vulnerability to correlation attacks in all subkeys. Generally, a correlation analysis needs fewer traces to find a secret subkey, as Table 4 clarifies, if its results are compared to the overall results of Table 3. However, this kind of analysis requires roughly five times more computation time than a differential analysis. Concerning the GALS pipeline implementations, as expected, it was not possible to find the secret key, even after analyzing 100,000 traces. For this reason, Table 4 does not bring trace numbers for GALS pipeline architectures.

Table 3. DEMA results for synchronous and GALS DES.

	DEMA (#)		
Architecture	**GALS PIPE 2**	**GALS PIPE 4**	**GALS PIPE 8**
Synchronous	49336	16365	88540
GALS	Nf	Nf	Nf

(Nf): Key not found
(#): Number of traces needed to find the secret key

Table 4. Number of traces to find the cryptographic subkeys using CPA and CEMA attacks.

	CEMA			CPA
Subkey	**S-Pipe 2**	**S-Pipe 4**	**S-Pipe 8**	**S-Pipe 2**
1	80	257	2,782	291
2	1,334	2839	8,635	3,986
3	1,326	2084	23,309	6,733
4	992	785	5,021	3,629
5	6,202	6524	71,056	6,447
6	1,785	2667	22,318	3,457
7	1,130	742	3,518	4,109
8	20,025	16285	16,984	7,849

6. CONCLUSIONS

This paper proposes a new GALS pipeline architecture for enhancing robustness to SCA in cryptographic hardware implementations. For the first time, robustness is sought by replicating the structure of elementary modules in cryptographic algorithms in asynchronously communicating pipeline stages, which are supplied with self-varying randomly generated clock frequencies.

The area-robustness trade-off is a main concern of the approach. Compared to a regular (synchronous, non-pipeline) DES implementation, the proposed architecture indeed presents high area overhead. However, compared to state of the art, tamper-resistant asynchronous implementations like STTL, most GALS PIPE versions are small. Besides, attack data on the GALS PIPE architectures display an outstanding resistance to SCAs, definitely better than a regular DES implementation. More sophisticated attacks like CPA/CEMA have been performed, and the results still confirm the robustness of the approach. In another recent publication [32] the authors show that the robustness of the approach is also better than at least one state of the art asynchronous DPL approach.

The proposed architecture also displays higher throughput when compared to non-pipeline implementations. This is true even when accounting for the timing penalties caused by clock domain synchronizers. These figures can be significantly enhanced by the use of more efficient synchronizers like those proposed by Dobkin and Ginosar in [32], in substitution to the employed simple 2-flop synchronizers. These options are currently under investigation.

In conclusion, the proposed architecture provides multiple possibilities to explore the design space of hardware implementation for cryptographic algorithms, adding flexibility to trade area, SCAs resistance and throughput based on specific application constraints.

ACKNOWLEDGEMENTS

This work has been conducted while author Rafael Soares was a PhD student at the PPGCC, PUCRS and author Victor Lomné was a PhD student at the LIRMM, Université de Montpellier 2. It was partially supported by CNPq (under grants PNM 140044/2008-6 and 309255/2008-2), by the CAPES/COFECUB (French-Brazilian Cooperation) under grant no. BEX1446/07-0 and by the ANR – ICTER Project (French National Research Agency), the International "Secure Communicating Solutions" Cluster. Also, all authors acknowledge the work of reviewers of both, the 24th SBCCI and of the JICS journal in improving the overall quality of this article.

REFERENCES

[1] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and other Systems," in *Proceedings of the 16th International Cryptology Conference,* 1996, pp. 104-113.

[2] P. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Proceedings of the 19th International Cryptology Conference,* 1999, pp. 388-397.

[3] E. Brier, C. Clavier and F. Olivier, "Correlation power Analysis with a Leakage Model," in *Proceedings of the International Workshop Cryptographic Hardware and Embedded Systems,* 2004, pp. 16-29.

[4] T. Le Clédière, C. Canovas, C. Servière, J. Lacoume and B. Robisson, "A Proposition for Correlation Power Analysis Enhancement," in *Proceedings of the International Workshop Cryptographic Hardware and Embedded Systems,* 2006, pp. 174-186.

[5] O. Meynard, S. Guilley, J. Danger and L. Sauvage, "Far Correlation-based EMA with a Precharacterized Leakage Model," *in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition,* 2010, pp. 977-980.

[6] D. Chapiro, "Globally Asynchronous Locally Synchronous Systems," PhD Thesis, Stanford University, 1984, 134p.

[7] R. Soares, N. Calazans, V. Lomné, A. Dehbaoui, P. Maurine and L. Torres, "A GALS Pipeline DES Architecture to Increase Robustness against DPA and DEMA Attacks," in *Proceedings of the 23rd Symposium on Integrated Circuits and System Design*, 2010, pp. 115-120.

[8] S. Skorobogatov and R. Anderson, "Optical Fault Induction Attacks," in *Proceedings of the International Workshop Cryptographic Hardware and Embedded Systems,* 2002, pp. 2-12.

[9] R. Anderson, *Security Engineering: A Guide to Build Dependable Distributed Systems*, 2nd Edition, Wiley & Sons, 2001.

[10] K. Gandolfi, C. Mourtel and F. Olivier, "Electromagnetic Analysis: Concrete Results," in *Proceedings of the International Workshop Cryptographic Hardware and Embedded Systems,* 2001, pp. 252-261.

[11] R. Bevan and E. Knudsen, "Ways to Enhance Differential Power Analysis," in *Proceedings of the Information Security and Cryptology,* LNCS 2587, 2003, pp. 327-342.

[12] M. Joye, P. Paillier and B. Schoenmarkers, "On Second-Order Differential Power Analysis," in *Proceedings of the International Workshop Cryptographic Hardware and Embedded Systems,* 2005, pp. 293-308.

[13] D. Réal, F. Valette and M. Drissi, "Enhancing Correlation Electromagnetic Attack using Planar Near-Fiel Cartography," *in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition,* 2009, pp. 628-633.

[14] K. Tiri and I. Verbauwhede, "A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition,* 2004, pp. 246-251.

[15] K. Kulikowski, V. Venkataraman, Z. Wang, A. Taubin and M. Karpovsky, "Asynchronous Balanced Gates Tolerant to Interconnect Variability," in *Proceedings of the IEEE International Symposium on Circuits and Systems,* 2008, pp. 3190-3193.

[16] V. Lomné, P. Maurine, L. Torres, M. Robert, R. Soares and N. Calazans, "Evaluation on FPGA of Triple Rail Logic Robustness against DPA and DEMA," in *Proceedings of the Design, Automation and Test in Europe and Exhibition,* 2009, pp. 634-639.

[17] J. Rabaey, *Digital Integrated Circuits: A Design Perspective,* Prentice Hall, 1996, 702p.

[18] J. Danger, S. Guilley, S. Bhasin and M. Nassar, "Overview of Dual Rail with Precharge logic Styles to thwart Implementation-Level Attacks on Hardware Cryptoprocessors," in *Proceedings of the 3rd International Conference on Signals, Circuits and Systems,* 2009, pp. 1-8.

[19] J. Goodwin, and P. Wilson, "Advanced Encryption Standard (AES) Implementation with Increased DPA Resistance and Low Overhead," in *Proceedings of the International Symposium on Circuits and Systems,* 2008, pp. 3286-3289.

[20] F. Ghellar and M. Lubaszewski, "A Novel AES Cryptographic Core Highly Resistant to Differential Power Analysis," in *Proceedings of the 21st Symposium on Integrated Circuits and Systems Design,* 2008, pp. 140-145.

[21] J. Golic, "Techniques for Random Masking in Hardware," *IEEE Transactions on Circuits and Systems-I,* vol.54, no. 2, February, 2007, pp. 291-300.

[22] D. Mesquita, B. Badrignans, L. Torres, G. Sassatelli, M. Robert and F. Moraes, "A Leak Resistant SoC to Counteract Side Channel Attacks," in *Proceedings of the International Symposium on System-on-Chip,* 2006, pp. 1-4.

[23] Y. Lu, M. O'Neill and J. McCanny, "FPGA Implementation and Analysis of Random Delay Insertion Countermeasure against DPA," in *Proceedings of the International Conference on ICECE Technology,* 2008, pp. 201-208.

[24] C. Clavier, J. Coron, and N. Dabbous, "Differential Power Analysis in the Presence of Hardware Countermeasures," in *Proceedings of the International Workshop Cryptographic Hardware and Embedded Systems,* 2000, pp. 13-48.

[25] S. Nagashima, N. Homma, Y. Imai, T. Aoki and A. Satoh, "DPA Using Phase-Based Waveform Matching against Random-Delay Countermeasure," in *Proceedings of the International Symposium on Circuits and Systems,* 2007, pp. 1807-1810.

[26] Y. Zafar, and D. Har, "A Novel Countermeasure Enhancing Side Channel Immunity in FPGAs," in *Proceedings of the International Conference on Advances in Electronics and Micro-electronics,* 2008, pp. 132-137.

[27] N. Kamoun, L. Bossuet and A. Ghazel, "Correlated Power Noise Generator as Low Cost DPA Countermeasures to Secure Hardware AES Cipher," in *Proceedings of the International Conference on Signals, Circuits and Systems,* 2009, pp. 1-6.

[28] F. Standaert, S. Örs and B. Preneel, "Power Analysis of an FPGA Implementation of Rijndael: Is Pipelining a DPA Countermeasure?" in *Proceedings of the 6th International Workshop on Cryptographic Hardware and Embedded Systems,* 2004, pp. 30-44.

[29] F. Gürkaynak, S. Oetiker, H. Kaeslin, N. Felber and W. Fichtner, "Design Challenges for a Differential-Power-Analysis Aware GALS-based AES Crypto ASIC," *Electronic Notes in Theoretical Computer Science,* vol. 146, no. 2, January, 2006, pp. 133-149.

[30] W. Tuchman, *A Brief History of the Data Encryption Standard. Internet besieged: countering cyberspace scofflaws,* ACM Press/Addison-Wesley Publishing Co., 1997, pp. 275–280.

[31] R. Mahmud, "Techniques to Make Clock Switching Glitch Free," Captured in http://www.eetimes.com/news/design/showArticle.jhtml?articleID=16501239, May, 2009.

[32] R. Soares, N. Calazans, F. Moraes, P. Maurine and L. Torres, "A Robust Architectural Approach for Cryptographic Algorithms using GALS Pipelines," IEEE Design & Test of Computers, 2011. Approved for publication in the Asynchronous Design and Test issue of the journal, to appear in Sep/Oct 2011.

[33] R. Dobkin and R. Ginosar, "Two-phase synchronization with sub-cycle latency," *Integration, the VLSI Journal,* vol. 42, no. 3, June, 2009, pp. 367–375.