Combined SCA and DFA Countermeasures Integrable in a FPGA Design Flow

Shivam Bhasin, Jean-Luc Danger, Florent Flament, Tarik Graba, Sylvain Guilley, Yves Mathieu, Maxime Nassar, Laurent Sauvage, Nidhal Selmane

Combined SCA and DFA Countermeasures Integrable in a FPGA Design Flow

S. Bhasin, J.-L. Danger, F. Flament, T. Graba, S. Guilley, Y. Mathieu, M. Nassar, L. Sauvage, N. Selmane

***Abstract*—The main challenge when implementing cryptographic algorithms in hardware is to protect them against attacks that target directly the device. Two strategies are customarily employed by malevolent adversaries: observation and perturbation attacks, also called SCA and DFA in the abundant scientific literature on this topic. Numerous research efforts have been carried out to defeat respectively SCA or DFA. However, few publications deal with concomitant protection against both threats. The current consensus is to devise algorithmic countermeasures to DFA and subsequently to synthesize the DFA-protected design thanks to a DPA-resistant CAD flow. In this article, we put to the fore that this approach is the best neither in terms of performance nor of relevance. Notably, the contribution of this paper is to demonstrate that the strongest SCA countermeasure known so far, namely the dual-rail with precharge logic styles that do not evaluate early (EE), happen surprisingly to be almost natively immune to most DFAs. Therefore, unexpected two-in-one solutions against SCA and DFA indeed exist and deserve a closer attention, because they ally simplicity with efficiency. In particular, we illustrate a logic style, called WDDL w/o EE, and a design flow that realizes in practice one possible combined DPA and DFA counter-measure especially suited for reconfigurable hardware.**

***Index Terms*—Side-Channel Analysis (SCA), Differential Power Analysis (DPA), Dual-rail with Precharge Logic (DPL), Early Evaluation (EE), Differential Fault Analysis (DFA), Wave Dynamic Differential Logic (WDDL), Computer-Aided Design (CAD), Field Programmable Gates Array (FPGA).**

I. Introduction

Embedded systems that contain cryptographic modules are becoming commonplace with the generalization of privacy, authentication and integrity in digital communications. The cryptographic hardware is very resource consuming because it relies on complex operations needed to prevent illegitimate users from spying, impersonating or altering the communications. Therefore, many studies focus on the optimization of cryptographic blocks. In parallel, new threats – not of cryptanalytic nature – have shown up: it has been suggested and demonstrated that an attacker can break the logical security conveyed by the cryptography by merely observing or perturbing it on the physical layer. The common point between those two exaction strategies is their aim to defeat the security by retrieving some secret elements (such as keys) from which the security features stem.

One the one hand, observation attacks are also known as side-channel attacks (SCAs [1]), in that they exploit a physical leakage of the device to gain information about its internal secrets. On the other hand, perturbation attacks consist in altering the state of the device so as to retrieve faulted outputs, that together with nominal outputs, can disclose or negate relationships within the secret bits normally concealed into the hardware; these attacks are referred to as differential fault analyses (DFAs [2], [3]). The main strength of SCAs is their furtivity. As they are virtually impossible to detect, an adequate countermeasure must be vigilant each time the cryptographic engine is in use. On the contrary, the first prerequisite for a DFA to be successful is to actually modify the device's state. A detection strategy can thus be enforced to check for the device operations' integrity. However, the careful check of all components of an embedded system is very fastidious and error-prone. In addition, even if any sensitive data is carefully monitored for integrity, the faults coverage remains an issue. Indeed, if detecting one single error (of unitary bit entropy) is easy using simple parity codes, the detection of multiple errors is more difficult to address. In general, the detection logic complexity is exponential with the faults multiplicity, which quickly becomes deterrent in practical applications.

One device can be claimed tamper-resistant only if it is protected, at least to some extent, against both SCA and DFA simultaneously. It must be noticed that the efforts to deploy in protection depends on the threat. To be successful, the best attacks known so far require to garner some thousands of side-channel traces recording (SCAs) but only a couple of faults (DFAs) from an unprotected device. As a consequence, the need for protection is more stringent against DFA than it is against SCA. This asymmetry is one reason for which the countermeasures against DFA and SCA are nowadays studied separately: this partitioning makes it possible for a designer team to tune the countermeasure efficiency according of the threat urgency, while keeping the flexibility to combine them at the final stage of integration. Another reason why countermeasures against DFA and SCA are considered independently is linked with our state-of-art in defense. The protection against DFA is naturally achieved at an algorithmic level, with the introduction of redundancy in data representation and processing. However, the effective protection against SCA is more subtle, since it requires the removal for any source of leakage through physical side-channels. Therefore, the widespread methodology consists in using dedicated logic gates along with *ad hoc* backend steps. As we know how to resist against DFA before the logic synthesis and to resist against DPA after synthesis, it is implicitly considered obvious that the protection against DFA and DPA should be built one on top of each other.

In this article, we advocate that this methodology is neither

natural nor efficient. Basically, we show that a class of strong countermeasures against SCA, namely all variants of dual-rail with precharge logic (DPL) styles which do not suffer from early evaluation (EE), are already protected against the state-of-the-art fault injection techniques. Thus, by subsuming the individual issues of securization against SCA and DFA into a unique problem, we arrive to an original solution that is economic in resources because of its duality w.r.t. both the SCA and the DFA threats. In addition, we show that the countermeasure is all the more efficient as the faults multiplicity is high, which is a property out of reach of traditional protections based on coding theory.

The rest of the article is organized as follows. Section II presents the DPL protection against SCA, and motivates for the preference of DPL without EE. In section III, the protection potential of DPL (w/ or w/o EE) against DFA is explained. The section IV presents a methodology for mapping this protection into FPGAs, and details its performances in terms of resources usage. Finally, conclusions are discussed in section V.

II. Dual-rail with Precharge Logic Styles against SCAs

The goal of a protection against SCAs is to prevent any attacker from the retrieving any information from any internal bit. Various solutions have been proposed to address this requirement. Side-channel *masking* consists in making the activity of sensitive bits random by rewritting the algorithm in such a way that those variables depend on a external entropy source. Side-channel *hiding* adds redundant logic so as to end up with a constant activity when sensitive bits are manipulated. Each solution has its own pros and cons; some logic styles even mix the two for an improved security. Still, the comparison between these securization options is beyond the scope of this article.

In this article, we focus on the hiding styles. Indeed, as will be made clear in Sec. III, those styles combine harmoniously with DFA protection, whereas masking styles do not, as demonstrated in [4]. Information hiding at the bit level can be achieved by a large variety of *ad hoc* encodings and protocols. However, the most convenient ones rely on a so-called dual-rail with precharge representation. Every bit a involved in the algorithm is actually mapped into a couple of wires, named $(a_{\mathrm{F}}, a_{\mathrm{T}})$, and called the 'false' and 'true' halves of the dual-rail variable a. The couple $(a_{\mathrm{T}}, a_{\mathrm{F}})$ alternates between two values:

1) $(0,0)$ or $(1,1)$, called NULL0 or NULL1, and designated as a NULL token, playing the role of spacer, and
2) $(1,0)$ or $(0,1)$, called VALID0 or VALID1, and designated as a VALID token, carrying the value of a.

One DPL computation alternates NULL and VALID tokens, with the remarkable property that exactly one bit toggle occurs in each transition. A pair of gates $(f_{\mathrm{F}}, f_{\mathrm{T}})$ respects the DPL convention if:

- It propagates the NULL values, *i.e.*, if all the inputs are NULL, then $(f_{\mathrm{F}}, f_{\mathrm{T}})$ is also NULL.
- It propagates the VALID values, *i.e.*, if all the inputs are VALID, then $(f_{\mathrm{F}}, f_{\mathrm{T}})$ is also VALID.

Table I
Security features of classical DPL styles.

DPL style + reference	∃ Random?	∃ EE?	Target
WDDL [5]	No	Yes	ASIC and FPGA
MDPL [9]	Yes	Yes	ASIC and FPGA
iMDPL [10]	Yes	No	ASIC and FPGA
DRSL [11]	Yes	No	ASIC
PMRML [12]	Yes	No	ASIC
STTL [13]	No	No	ASIC and FPGA
SecLib [14]	No	No	ASIC
WDDL w/o EE [this article]	No	No	FPGA

Wave dynamic differential logic (WDDL [5]) has been the first logic style to implement these conditions. WDDL has the nice property to be separable, meaning that f_{F} (*resp.* f_{T}) depends only on the false (*resp.* the true) inputs half. However, some other properties have been added afterwards to ensure a secure operation of WDDL. First of all, it has been noticed that on the way from all NULL to all VALID values, glitches could occur if the functions $(f_{\mathrm{F}}, f_{\mathrm{T}})$ were not positive [6]. Afterwards, many authors notice concomitantly that the evaluation time depends on the inputs values [7], [8]. An up-to-date list of known DPLs styles used for side-channel information hiding countermeasure is given in Tab. I.

The salient features of these logic styles are briefly described below:

- WDDL is the less complex DPL style because it is separable, which makes it possible to reduce the overhead of each dual network.
- MDPL adds some logic on top of WDDL to swap randomly the logic interconnect pairs, in a view to balance the routing mismatches. Indeed, this problem is not addressed directly by WDDL but is left to the layouter [15], [16].
- iMDPL fixes the leakage conveyed by data-dependant evaluation and precharge dates in WDDL and MDPL.
- DRSL combines masking and early evaluation protection, and is optimized to be compact using one standard ASIC cell (`AOI222`) and all RSL [17], [18] gates.
- PMRML has similar features as iMDPL and DRSL but a more complex implementation since evaluation signals must be computed and distributed pervasively in the netlist.
- STTL is a non-masked improvement of WDDL style free of early evaluation. STTL is however not balanced in structure, as WDDL, and is limited in speed by the slow validation path, by design longer than the path of the data signal pairs. This limitation seriously impedes the throughput of STTL. Eventually, we underline that STTL requires the routing of three wires per logical signal.
- SecLib is non-masked computation style that fixes the EE issue and features a balanced structure. To be exhaustive, we should also mention the NCL (Null Convention Logic)

Table II
LOOK-UP-TABLE (LUT) MASKS ENCODING FOR 4-INPUT LUTS IMPLEMENTING THE AND FUNCTION IN WDDL W/O EARLY EVALUATION.

	AND_T	AND_F	**Input state in**
a_T a_F b_T b_F	FC80	FAE0	**the DPL protocol**
0 0 0 0	0	0	All NULL0
0 0 0 1	0	0	Transitional from NULL0
0 0 1 0	0	0	Transitional from NULL0
0 0 1 1	0	0	Faulty
0 1 0 0	0	0	Transitional from NULL0
0 1 0 1	0	1	All VALID: $(a, b) = (0, 0)$
0 1 1 0	8	1	All VALID: $(a, b) = (0, 1)$
0 1 1 1	1	1	Transitional from NULL1
1 0 0 0	0	0	Transitional from NULL0
1 0 0 1	0	1	All VALID: $(a, b) = (1, 0)$
1 0 1 0	C	A	All VALID: $(a, b) = (1, 1)$
1 0 1 1	1	1	Transitional from NULL1
1 1 0 0	1	1	Faulty
1 1 0 1	1	1	Transitional from NULL1
1 1 1 0	F	F	Transitional from NULL1
1 1 1 1	1	1	All NULL1

that is a generalization of SecLib albeit deprived from any balance effort.

- WDDL w/o EE is a logic style dedicated to FPGA that removes the EE without computing a rendezvous. Instead, each functional half gate receives the true and false inputs, and decides to output the VALID value only when all the inputs are VALID. This behavior can be achieved by a purely combinatorial gate, as depicted in Tab. II. The detailed rationale behind the "WDDL w/o EE" style is the following:
  - The gate outputs NULL{0,1} when the inputs are NULL{0,1} or transitional from this value.
  - The gate outputs VALID only when all the inputs are VALID.
  - In case of inconsistent values w.r.t. the DPL convention, the gate outputs an arbitrary NULL value.

  This logic does not evaluate early by design, and propagates errors: if any input is stuck to NULL or if the input is out of specifications, then the output always remains to NULL too. In addition, this logic **does not generate glitches** even if the functionality is not positive, and **can be inverting**. Therefore, the synthesis is more optimized than for plain WDDL.

III. POTENTIAL OF DPL W/O EE FOR PROTECTION AGAINST DFAS

A. Fault Model

Most, if not all, fault attacks reported in the literature, use a single perturbation source to generate faults within the FPGA. Basically, the perturbation responsible to place the target device out of specified operating conditions is either global or local. Global perturbations consist in varying one environmental variable, such as the power supply, the clock frequency or the external temperature. The perturbation can be steady or transient. But in either case, the source of faults is not adaptative: the complete circuit is faulted altogether. Local perturbation are more difficult to create, because they require an access to the silicon die surface. This condition means that a mechanico-chemical preparation of the circuit must be done beforehand. Such a step is reserved to advanced laboratories that have access to specialized facilities. Moreover, the preparation cannot be achieved with 100% success probability, which drastically increases the cost of the attack. Nonetheless, even if open samples are available, equipments able to inject a localized fault is often large. For instance, a laser source and its focalization optic limit the minimum distance between two faults.

We would like to underline that it is anyway very difficult if not impossible to resist against coherent multiple faults injection. Any protection mechanism, based on either spacial or temporal redundancy can be abused. Similarly, when a parallel path uses an encoding to check for the data integrity, consistent faults can be injected to change a code word for another one.

However, if we imagine that it is possible with some sophisticated equipment to inject related multiple faults, which has by the way never been published so far, it is not taken for granted that the antinomic bit-flips can be obtained. Indeed, the only way to trick the DPL w/o EE logic is to replace a VALID token (0, 1) by another VALID (1, 0). Now, with two spatially close injectors, it is far from obvious that the faults will not negatively interfere. Indeed, the way to flip a 0 into a 1 is to inject energy at the correct wavelength in a N^+ doped region whereas to flip a bit the other way round, the energy shall be injected in another well possibly at a different wavelength. If we take the example of electromagnetic (EM) injection with micrometric probes, it is expected that opposite fields must be generated to trigger contrary bit flips. However, this also means that the perturbation merely cancels itself due to the proximity of the two regions to excite. In any case, given the lack of literature about this subject and without any proof-of-concept experimental feedback, it is hard to further speculate on the feasibility of such coherent fault injections. Therefore it is safe to consider such a vulnerability as highly implausible, and thus can be ignored in a short to medium term. In summary, we continue our analysis by assuming that multiple faults can be generated locally, but decorrelated one from each other.

B. Early Evaluation Prevention and Faults Transformations

The article [19] shows that WDDL is immune against multiple asymmetric faults such as those caused by setup violations. Basically, the idea is that asymmetric faults turn a VALID token into a NULL one. The NULL token can propagate until the outputs, being even amplified. However, the NULL wave propagation acts as an eraser, which means that the outputs have eventually lost any information about the faulted values. A parallel is done in [19] between asymmetrical faults and the logical propagation of `'U'` value in the 9-valued type `std_ulogic` of VHDL (IEEE standard number 1076).

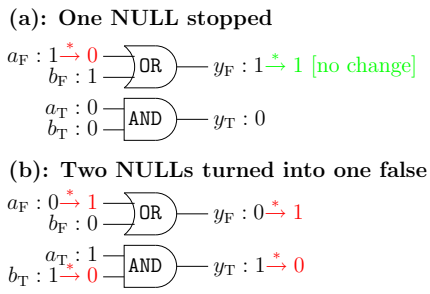

Figure 1. Two DPL w/ EE drawbacks to fight DFAs, one the example of a WDDL AND gate. In this figure and in the subsequent ones, the asterisk character (*) symbolizes the faults.

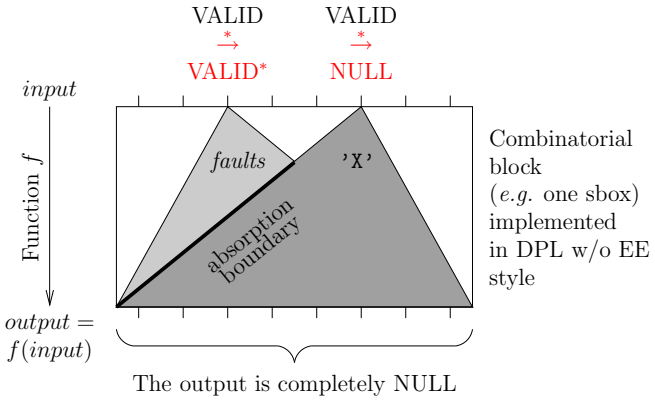

Figure 2. Illustration of the absorption of VALID faults by a salvo of NULL tokens in two interpenetrating logic cones in a DPL w/o EE netlist.

We add in this paper that all dual-rail with precharge logics (DPLs) are actually protected against setup violation attacks. Indeed, they never disclose the faulty result in the presence of a setup violation. Instead, they have two different kinds of behavior:

1) WDDL and MDPL compute results given the inputs, and propagate NULL spacers for the outputs whose values are non decidable. This is the logic behavior of `'U'` in VHDL. One could say that faults in these logics are recessive w.r.t. VALID values.
2) iMDPL, DRSL, PMRML, STTL, SecLib and WDDL w/o EE propagate the NULL on the fault fanout, even if a VALID value could have been deduced. This is the logic behavior of `'X'` in VHDL. Along with the former phenotypic metaphor, faults in this second class of logics are dominant, or rather contaminating, as their propagation is indeed an unexpected avalanche effect.

The implication is that DPL in itself does not provide a good protection against symmetrical faults. As a matter of fact, it can filter out a NULL (see Fig. 1(a)) and generate a faulted VALID from NULL tokens (see Fig. 1(b)). In contrast, the DPL styles that are EE-free propagate the NULL unconditionally; this feature is even part and parcel of the WDDL w/o EE specification. Additionally, the NULL (behaving like an `'X'`) is always absorbing the other VALID faults, as shown in Tab. 2.

C. Propagation of NULL Values Through Substitution Boxes

The fault propagation in logics with EE is exploding in substitution boxes (sboxes). The average number of NULL tokens at the output of various sboxes when one or several NULL tokens of the same type (either NULL0 or NULL1) are at the input has been computed in Tab. III for any logic style subject to EE, such as WDDL or MDPL.

In DPL w/o EE, the propagation is also independent on the implementation. It is also more straightforward as it does not depend on the data: the propagation through a gate occurs iff the output depends on the given input. This is case of all non-trivial gates. Notably, any fault, even single, on the input of an sbox, corrupts the entire sbox output: the propagation is maximal.

Table III
NUMBER OF NULL TOKENS PROPAGATED ON AVERAGE THROUGH THE SBOXES OF AES ($8 \rightarrow 8$) AND DES ($6 \rightarrow 4$) IN DPL WITH EE.

Fault multiplicity	AES Sbox (SubBytes)	DES Sboxes #1	#2	#3	#4	#5	#6	#7	#8
0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
1	4.04	2.48	2.53	2.65	2.46	2.53	2.60	2.63	2.50
2	7.04	3.88	3.90	3.92	3.93	3.91	3.93	3.93	3.91
3	7.94	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4	8.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
5	8.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
6	8.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
7	8.00	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
8	8.00	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.

D. Analysis of the DFA Protection of the Proposed Logic

Single bit faults are inefficient against DPL because they turn a VALID data into a NULL token, that propagates and leads to an unexploitable error since it hides the faulted value. This is the typical scenario described in paper [19]. Highly multiple faults generate randomly a large quantity of NULL values along with some more unlikely but devastating bit-flips. However, as NULL values are systematically propagated, they proliferate very quickly after some combinatorial logic layers traversal. And as they have the nice property to contaminate VALID values, the risky coherent bit-flips (simultaneous $0 \xrightarrow{*} 1$ and $1 \xrightarrow{*} 0$ in one dual-rail couple), they jam their propagation hopefully before they reach the algorithm output. This absorption property is all the more efficient as the number of NULL generated by the multiple faults is high. Therefore, the only way to inject a poisonous fault is to stress the circuit sufficiently enough to have multiple faults, without nonetheless creating too many faults so as to leave a chance for them not to be absorbed during their percolation towards the outputs. But, hopefully, in this opportunity window of low stress (generation of 2, 3, or maximum 4 errors because of the high diffusion of cryptographic algorithms), efficient coding schemes can be used in supplement to the DPL w/o EE protection.

To be more accurate, we present a simple model that provides a convincing proof of our assertion. Let us consider a dual-rail circuit that is attacked with a perturbation that is focalized on $2n$ wires, and that has an intensity sufficient enough to cause $m \leq 2n$ simultaneous faults. We also make the optimistic hypothesis that the m faults are equidistributed over the $2n$ wires, and that the flips are truly symmetrical, *i.e.* it is as likely to flip to a 0 and to a 1. To further simplify the modelization, we also assume that the attacked block has a perfect diffusion: in practice, this is not exactly true for one round of an algorithm, but for two of them. Nevertheless, it helps us grasp more intuitively the idea of the proof without introducing overcomplicated considerations. Therefore, for a fault to successfully propagate through the round, no single NULL shall be generated. Otherwise, the NULL wave catches the fault, because of the perfect diffusion, as already depicted in Fig. 2. The first constatation is that for VALID faults to be generated, m must be even. Indeed, they are generated by pairs. If, on the contrary, m is odd, then at least one NULL (fit-flip of one wire in a pair) is generated, leading to the VALID fault absorption. Then, a VALID fault is generate iff, given a unique fault, a second one occurs in the paired wire. For $m = 2$ faults, this happens with probability $1/(2n-1)$. For more faults, the generation of solely paired faults consists in always pairing the remaining faults. Then, the probability to generate at least one VALID fault that survives until the output is equal to:

$$p(2n, m) \doteq \begin{cases} \binom{n}{m/2} / \binom{2n}{m} & \text{if } m \text{ is even,} \\ 0 & \text{otherwise.} \end{cases}$$

This probability becomes very small starting from a multiplicity of 4 when m increases up to n[1]. This is to be contrasted with schemes involving a coding with error detection. They are basically able to detect:

- all the faults of multiplicity smaller than the error detection capability r[2], but
- only a ratio of $1 - 1/2^r$ faults for $m > r$.

The figure 3 compares the rate of successful faults injection depending on the multiplicity, for an $n = 8$ set of wires, respectively for the proposed scheme based on DPL w/o EE and for a classical integrity check with a linear code detecting $r = 2$ bits of error.

The authors would like to insist that this is the first time that a countermeasure against DFA proves efficient even in the context of a large number of faults. As a matter of fact, usual schemes, based on spatio-temporal or coding, can be defeated with high probability if the number of faults is greater than the detection capacity. Smartly enough, the implementations using DPL w/o EE take advantage of three properties that all contribute to destroy the VALID faults:

[1]When m is too large, starting from n, the probability increases, because of the property: $p(2n, m) = p(2n, 2n - m)$.

[2]Faults of multiplicity $m \leq r$ mutate a code word into a non-code word.

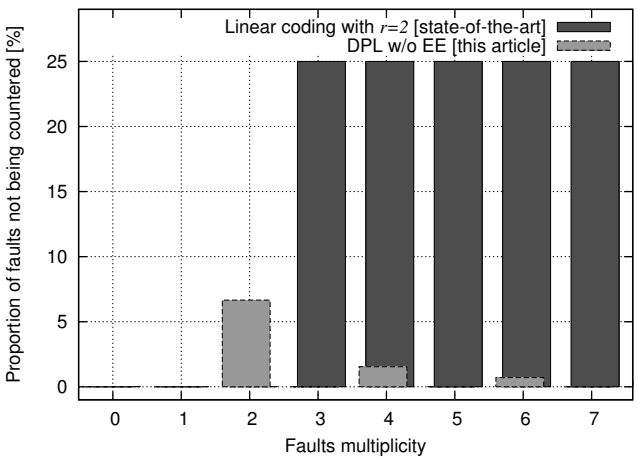

Figure 3. Probability that m faults injected on n wires be innocuous due to the protection conveyed by two different countermeasures: either a *detection* by an informational redundancy scheme or an *annihilation of the faulted data* by one or several VALID $\xrightarrow{*}$ NULL token transformations.

1) faults are very likely to alter only one wire in a pair, especially if the stress is badly localized, thus creating much more NULL tokens than wrong VALID pairs,
2) because of the protection against EE, NULL values win against VALID ones, hereby hiding in particular VALID fault propagation,
3) as the algorithms implement cryptography, they have a high diffusion, which helps the NULL values meet (and thus eat) the possibly faulted VALID values still alive.

IV. CAD Flow for the Proposed Counter-Measure

As every digital system, cryptographic coprocessors can be separated into control and datapath. The datapath contains the secret key related operations. Thus to assure security of the design it is sufficient to secure the datapath only. A design flow to implement a cryptographic coprocessor on an FPGA is shown in Fig. 4. Since DPL designs are redundant by nature, we have to use customised tool for processing. The goal of this synthesis is to remove the unnecessary logic redundancy while keeping the redundancy needed for DPL style. This cannot be achieved by a standard design flow. An ASIC synthesizer is used to synthesize the design with a library containing only those gates which respect the DPL style constraints. Then the output netlist is processed using a custom tool which converts a single-rail netlist into a DPL netlist. The controller is then connected to the datapath using a wrapper. Thereafter, a legacy FPGA vendor tool does synthesis, mapping, placing & routing for the whole design on the FPGA. Although the design flow is shown for Altera FPGAs, it has also been tested apt for Xilinx FPGAs.

As stated earlier, to secure a design against SCA and DFA we can use a DPL style which is free from EE. WDDL is a DPL style most suited for FPGA designs but it is prone to EE. In [19], authors implement a WDDL design in FPGAs using a library containing four-input functions which are positive in

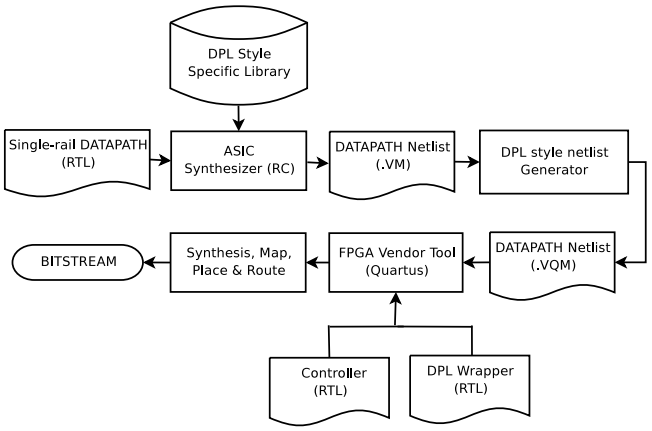

Figure 4. Design-flow for proposed counter-measure.

Table IV
AREA OF AN AES DATAPATH SYNTHESIZED FOR THE STRATIX FPGA.

Logic style	Reference	WDDL w/ EE	WDDL w/o EE
LuT4 count	2,396	11,249	28,569

nature. We use the same methodology in this paper. To make WDDL protected against EE, we limit the library to two-input gates, implemented as per Tab. II.

We have applied these syntheses on an AES datapath. The table IV summarizes the area of an unprotected datapath, the same datapath protected with an EE-prone logic (namely WDDL) and the same datapath protected with an EE-free logic (namely WDDL w/o EE), for the Stratix family of Altera. The implementation size of the "WDDL w/o EE" style is greater than that of original "WDDL", however it is more secure against SCAs and completely secure against any type of DFAs; still, some optimizations will help reduce the overhead.

V. CONCLUSION

This paper shows that, in addition to increasing the resistance against SCAs, the DPL styles also help resist against DFAs. Indeed, single faults consist in turning a VALID token into a NULL one, which conceals the value of the (sensible) data before corruption. The DPL styles that protect against the EE side-channel analysis ensure in addition that the NULL propagation contaminates all the data it crosses in the combinatorial logic cones. Thus, in the case of multiple faults, both VALID faults and NULL tokens are generated, but the NULL tokens destroy the VALID faults prior they arrive at the algorithms inputs. Therefore, we show for the first time that a SCA counter-measure is, as such, already an excellent counter-measure against DFA.

We also introduce WDDL w/o EE, a simple logic style that enhances the plain WDDL style by making it EE-free and having it avoid non-VALID inputs propagation. In addition, the synthesis of WDDL w/o EE is efficient because even non-inverting and positive functions are allowed. We provide a mapping of this new logic into LuT4-based FPGAs.

REFERENCES

[1] P. C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," in *Proceedings of CRYPTO'99*, ser. LNCS, vol. 1666. Springer-Verlag, 1999, pp. 388–397, (PDF).

[2] E. Biham and A. Shamir, "Differential Fault Analysis of Secret Key Cryptosystems," in *CRYPTO*, ser. LNCS, vol. 1294. Springer, 1997, pp. 513–525.

[3] G. Piret and J.-J. Quisquater, "A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD," in *CHES*, ser. LNCS, vol. 2779. Springer, September 2003, pp. 77–88, Cologne, Germany.

[4] A. Boscher and H. Handschuh, "Masking Does Not Protect Against Differential Fault Attacks," in *FDTC, 5th Workshop on Fault Detection and Tolerance in Cryptography, IEEE-CS*, aug 2008, pp. 35–40, DOI: 10.1109/FDTC.2008.12, Washington, DC, USA.

[5] K. Tiri and I. Verbauwhede, "A Logic Level Design Methodology for a Secure DPA Resistant ASIC or FPGA Implementation," in *DATE'04*. IEEE Computer Society, February 2004, pp. 246–251, Paris, France.

[6] S. Guilley, L. Sauvage, J.-L. Danger, T. Graba, and Y. Mathieu, "Evaluation of Power-Constant Dual-Rail Logic as a Protection of Cryptographic Applications in FPGAs," in *SSIRI*. Yokohama, Japan: IEEE Computer Society, jul 2008, pp. 16–23, DOI: 10.1109/SSIRI.2008.31, http://hal.archives-ouvertes.fr/hal-00259153/en/.

[7] K. J. Kulikowski, M. G. Karpovsky, and A. Taubin, "Power Attacks on Secure Hardware Based on Early Propagation of Data," in *IOLTS*. IEEE Computer Society, 2006, pp. 131–138, Como, Italy.

[8] M. Saeki and D. Suzuki, "Security Evaluations of MRSL and DRSL Considering Signal Delays," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E91-A, no. 1, pp. 176–183, 2008, DOI:10.1093/ietfec/e91-a.1.176.

[9] T. Popp and S. Mangard, "Masked Dual-Rail Pre-charge Logic: DPA-Resistance Without Routing Constraints," in *Proceedings of CHES'05*, ser. LNCS, vol. 3659. Springer, Sept 2005, pp. 172–186., Edinburgh, Scotland, UK.

[10] T. Popp, M. Kirschbaum, T. Zefferer, and S. Mangard, "Evaluation of the Masked Logic Style MDPL on a Prototype Chip," in *CHES*, ser. LNCS, vol. 4727. Springer, Sept 2007, pp. 81–94, Vienna, Austria.

[11] Z. Chen and Y. Zhou, "Dual-Rail Random Switching Logic: A Countermeasure to Reduce Side Channel Leakage," in *CHES*, ser. LNCS, vol. 4249. Springer, 2006, pp. 242–254, http://dx.doi.org/10.1007/11894063_20.

[12] K. J. Lin, S. C. Fan, S. H. Yang, and C. C. Lo, "Overcoming glitches and dissipation timing skews in design of DPA-resistant cryptographic hardware," in *DATE '07: Proceedings of the conference on Design, automation and test in Europe*, IEEE Computer Society, Ed. San Jose, CA, USA: EDA Consortium, 2007, pp. 1265–1270, DOI: 10.1109/DATE.2007.364471. Nice, France.

[13] R. Soares, N. Calazans, V. Lomné, P. Maurine, L. Torres, and M. Robert, "Evaluating the robustness of secure triple track logic through prototyping," in *SBCCI'08: Proceedings of the 21st annual symposium on Integrated circuits and system design*. New York, NY, USA: ACM, 2008, pp. 193–198.

[14] S. Guilley, F. Flament, R. Pacalet, P. Hoogvorst, and Y. Mathieu, "Security Evaluation of a Balanced Quasi-Delay Insensitive Library," in *DCIS*. Grenoble, France: IEEE, nov 2008, 6 pages, Session 5D – Reliable and Secure Architectures, ISBN: 978-2-84813-124-5, full text in HAL: http://hal.archives-ouvertes.fr/hal-00283405/en/.

[15] K. Tiri and I. Verbauwhede, "Place and Route for Secure Standard Cell Design," in *Proceedings of WCC / CARDIS*, Kluwer, Ed., Aug 2004, pp. 143–158, Toulouse, France.

[16] S. Guilley, P. Hoogvorst, Y. Mathieu, and R. Pacalet, "The "Backend Duplication" Method," in *CHES*, ser. LNCS, vol. 3659. Springer, 2005, pp. 383–397, August 29th – September 1st, Edinburgh, Scotland, UK.

[17] D. Suzuki, M. Saeki, and T. Ichikawa, "Random Switching Logic: A Countermeasure against DPA based on Transition Probability," 2004, http://eprint.iacr.org/2004/346.

[18] ——, "Random Switching Logic: A New Countermeasure against DPA and Second-Order DPA at the Logic Level," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. E90-A, no. 1, pp. 160–168, 2007.

[19] N. Selmane, S. Bhasin, S. Guilley, T. Graba, and J.-L. Danger, "WDDL is Protected Against Fault Attacks," in *FDTC*. IEEE Computer Society, September 6th 2009, in conjunction with CHES'09, Lausanne, Switzerland. Online version: http://hal.archives-ouvertes.fr/hal-00410135/en/.