Improving Security of SDDL Designs Through Interleaved Placement on Xilinx FPGAs

Rajesh Velegalati and Jens-Peter Kaps
ECE Department, George Mason University
Fairfax, VA, USA
Email: {rvelegal, jkaps}@gmu.edu

***Abstract*—Implementations of mathematically secure cryptographic algorithms leak information through side channels during run time. Differential Power Analysis (DPA) attacks exploit power leakage to obtain the secret information. Dynamic and Differential Logic (DDL), one of the popular countermeasures against DPA attacks, tries to achieve constant power consumption thereby decorrelating the leakage with the data being processed. Separated Dynamic and Differential Logic (SDDL), a variant of DDL, achieves this goal by duplicating the original design into Direct and Complementary parts which exhibit constant switching activity per clock cycle and have balanced net delays. Traditionally, on Field Programmable Gate Arrays (FPGAs) both parts are placed side-by-side to ensure symmetrical routing. However, due to process variations both parts will have slightly different delays. This limits the effectiveness of SDDL.**

In this paper we introduce a design flow to achieve interleaved placement of SDDL designs on Xilinx Spartan-3E FPGAs while preserving symmetric routing. We explore several placement configurations with respect to routing and security. The results of our experiments show that a well-balanced placement of SDDL can double the effectiveness of the SDDL countermeasures on FPGAs.

***Keywords*-SDDL for FPGAs; Differential Power Analysis; Interleaved Placement**

I. Introduction

Field Programmable Gate Arrays (FPGAs) are fast becoming a popular choice for a wide variety of applications ranging from digital cameras to aerospace and defense systems. Because of the outstanding feature of combining the programmability of processors with the performance of custom hardware, FPGAs have become an essential part of critical systems. Recent architectural advances of FPGAs are making them an alternative choice for low power applications where Application Specific Integrated Circuits (ASICs) are primarily used. Another hallmark of FPGAs is the ability to implement parallelized architectures efficiently, and they also posses excellent resistance against invasive attacks since the underlying platform is regular and does not reveal information on the actual design content [1],[2]. Because of these features, FPGAs have become attractive hardware platforms for cryptographic implementations.

Unfortunately the hardware implementations of cryptographic algorithms leak information in the form of so called side channels (i.e. power consumption, temperature, electromagnetic radiation, etc). Since Kocher et al. introduced Differential Power Analysis (DPA) [3], which makes use of the power consumption side channel, many countermeasures against it were proposed. These countermeasures can be broadly classified into two categories: *Masking* and *Hiding*. This paper concentrates on *Hiding* countermeasures.

Dynamic and Differential Logic (DDL) [4], a type of *Hiding* countermeasure, obfuscates the data being processed by maintaining constant power consumption for every clock cycle. DDL achieves this by dividing the design into two parts, so called Direct and Complementary parts which exhibit following properties

- For each gate in the original design, either a gate in the Direct part or its corresponding gate in the Complementary part should switch in each clock cycle. This is ensured by "precharging" the outputs of every gate in both parts to logic '0' and subsequently "evaluating" the correct output of the gate.
- Both parts of the DDL design should have symmetrical logic and routing capacitances.

The methodologies to implement DDL can be classified into two types, 1) *All positive logic* used by Wave Dynamic Differential Logic (WDDL) and 2) *positive and negative logic* used by Separated Dynamic Differential Logic (SDDL). SDDL for FPGAs, a variant of the SDDL style, was introduced in [5]. It is specifically designed as a countermeasure against DPA for lightweight or low area implementations on FPGAs. SDDL for FPGAs is still vulnerable to DPA attacks due to glitches and the effect of early precharge and evaluation [6]. Process variations inside the chip also affect the traditional implementation of SDDL on FPGAs where the Direct and Complementary parts are placed in different regions.

A recent study by Maiti et al. [7] shows that ring oscillators using identical resources have a large variation in frequency due to process variations when placed apart. The authors also note that placing such ring oscillators as close as possible will reduce these variations in frequency. Maiti's results suggest that in addition to the two properties of DDL designs mentioned above, it is also important that signals in the Direct part do not propagate with

a different speed then the corresponding signals in the Complementary part. The main contribution of this paper is a design methodology for implementing SDDL for FPGAs with interleaved placement which considers the factor of process variations. In order to achieve this goal we explore different placement configuration for the Direct and Complementary parts of the design. We also evaluate our design methodology against DPA by implementing interleaved SDDL for FPGAs on the Advanced Encryption Standard (AES) block cipher.

II. Placement Configuration

The effect of place and route on security of WDDL designs on FPGAs was studied in [8]. The authors note that constrained placement does not have any effect on the unbalance between the Direct and Complementary parts of the design however, routing does effect it. This is due to fact that WDDL which is placed in interleaved patterns requires cross connections between Direct and Complementary parts to implement negative logic. Hence symmetrical routing is not possible in WDDL style on FPGAs [9]. Different interleaved placement strategies for WDDL designs were explored in [10]. Unlike WDDL, SDDL for FPGAs allows negative logic, hence there is no requirement of any cross connections between Direct and Complementary parts. Thus it is possible to have symmetrical routing between Direct and Complementary parts of SDDL designs on FPGAs.

It is common design practice to separate the two parts of an SDDL design and place them side-by-side as in [5], [9]. Such placement strategy ensures that the two parts would have sufficient resources to obtain symmetrical delays. The relative location of similar components and nets in such placement configurations is rather far and thus have different delays due to process variations. This increases the susceptibility of SDDL designs against DPA. Additionally such placement configurations lead to the possibility of isolating and attacking the Direct part of the design using Electromagnetic Analysis (EMA) [11].

A. Placement Options

The Spartan-3E FPGA fabric consists of Block RAMs, Multipliers and Configurable Logic Blocks (CLBs). The CLBs are arranged in a regular 2D array of rows and columns throughout the FPGA. All CLBs are connected with each other via switch matrices and special interconnects. The availability of these connections has a significant effect on delay, area and routability of a design implemented on an FPGA. Fig. 1 shows the interconnections between a specific CLB and its neighboring counterparts. Each square in Fig. 1 is equivalent to one CLB. The number of connections from the specific CLB (indicated by filled square) to any particular CLB is given by the number present inside the square. Absence of a number indicates that there are no direct connections between the specific CLB to this square.

In order to interleave the Direct and Complementary parts, we explored four different placement configurations

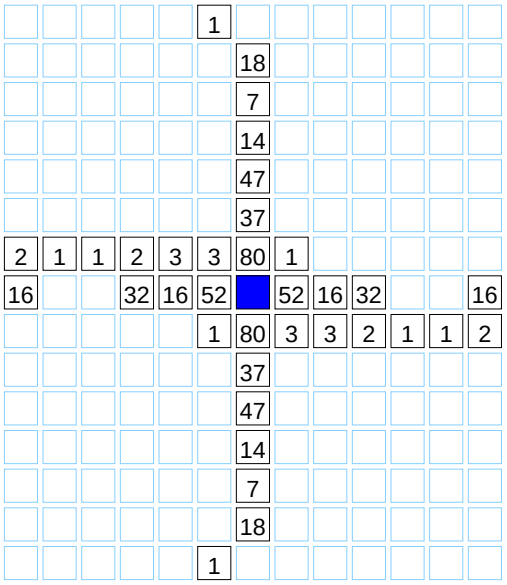


Figure 1. Number of Connections between a CLB and its surrounding counterparts

shown in Fig. 2. Each square in Fig. 2 is equivalent to one CLB. Table I shows the minimum number of connections between CLBs and their closest neighbor in the same part (i.e. either Direct or Complementary) of the SDDL design. Connections can be either in north-south (N-S), east-west (E-W) or diagonal (X) directions.

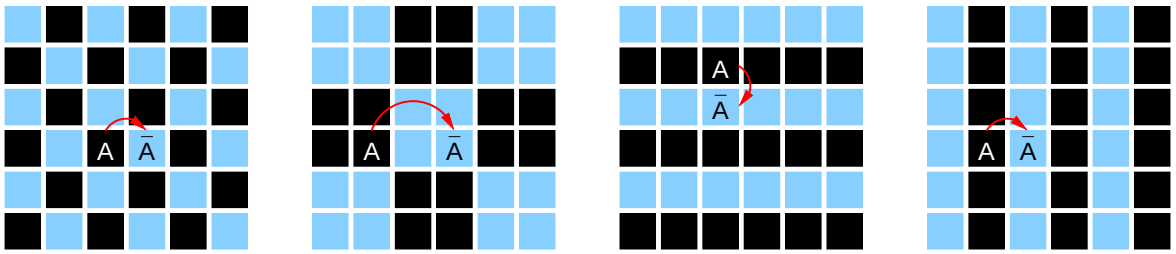
Table I
Minimum Number of Available Connections between CLBs of the Same Part

Patterns	Direction			Distance	% fewer to
	N-S	E-W	X	A $\rightarrow \bar{A}$	No Pattern
[1 x 1]	71	32	8	1 CLB	59.2
[2 x 2]	64	94	8	2 CLBs	39.0
[H x 1]	71	104	0	1 CLB	35.7
[1 x V]	160	32	0	1 CLB	29.4

[1 x 1] Configuration is a patterned like a Chess board in which each square is equivalent to one CLB. For each CLB in the Direct part (A) its complementary CLB ($\bar{A}$) is located in an adjacent CLB shown in Fig. 2a. The number of connections between two adjacent CLBs in each direction is least compared to the other configurations. The distance between the Direct and Complementary CLBs is 1 CLB.

[2 x 2] Configuration is also patterned like a Chess board. However, in this configuration each square is equivalent to four CLBs. For each CLB in the Direct part (A) its complementary CLB ($\bar{A}$) is located in an adjacent CLB shown in Fig. 2b. The distance between the Direct and Complementary CLBs is 2 CLBs and is the highest among the rest of the configurations.

[1 x H] Configuration, uses only even numbered CLB rows to implement the Direct part of the SDDL design. The Complementary part of the design is implemented in odd numbered CLB rows. Compared to [1 x 1] and [2 x 2] configurations, [1 x H] configuration has a higher number



a) Pattern [1 x 1] b) Pattern [2 x 2] c) Pattern [H x 1] d) Pattern [1 x V]

Figure 2. Patterns for placing original (A) and complementary ($\bar{A}$) paths

of connections between two neighboring CLBs. For each CLB in the Direct part (A) its Complementary CLB ($\bar{A}$) is located in an adjacent CLB shown in Fig. 2c. The distance between the Direct and Complementary CLBs is 1 CLB.

[V x 1] Configuration, uses only even numbered CLB columns to implement the Direct part of the SDDL design and the Complementary part is implemented in odd numbered CLB columns. Compared to the other patterns, [V x 1] configuration has highest number of connections between two neighboring CLBs. For each CLB in the direct path (A) its complementary CLB ($\bar{A}$) is located in an adjacent CLB shown in Fig. 2c. The distance between the Direct and Complementary CLBs is 1 CLB.

III. Workflow

We use Xilinx ISE 12.3 and ActivePerl v5.12 to implement interleaved SDDL designs on Xilinx FPGA. Our work flow shown in Fig. 3 is implemented in three phases.

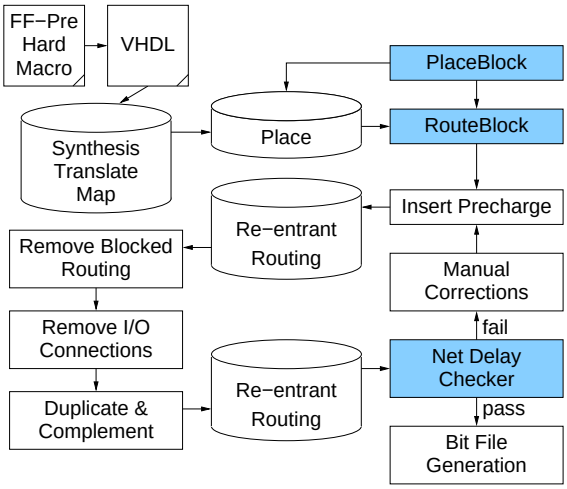


Figure 3. Work Flow

In the first phase, the VHDL description of the cryptographic algorithm with precharged registers is synthesized and mapped to the Xilinx Spartan-3E FPGA. We use a 4-bit Flip-Flop-Precharge hard-macro [12] to precharge the registers used in the design as the precharge circuit must be placed in the same CLB. We block the appropriate CLB positions depending upon the pattern used by using a program called PlaceBlock and use ISE to place the design in the available CLB positions. We then block the routing resources of the blocked CLBs using a program called RouteBlock.

In the second phase logic precharge circuits implemented using the technique described in [9], [5] are inserted into the placed circuit description file obtained from phase 1. Only re-entrant routing is done to obtain a precharged placed and routed design.

In the third phase I/O connections are removed, and the design is copied and duplicated into the appropriate CLB locations and the logic equations are complemented to obtain the complementary design. We duplicate and relocate the original design using the techniques described in [5]. Only re-entrant routing is done to connect the I/O to the Direct and Complementary parts of the design.

We created a verification program called Net delay checker which uses the Reportgen tool provided by the Xilinx ISE to compare the delays of each and every net present in the Direct part of the design to that of Complementary part. We make use of this script to perform a final sanity check before generating a bit file of the Placement Constrained SDDL design. If the Net Delay Checker program fails, it identifies and reports the failed nets. These failed nets are manually corrected and flow is repeated from the second phase until a valid Placement Constrained SDDL design is obtained.

A. Placement Blocking Algorithm

In order to implement the patterned SDDL implementation discussed in Sect II, the PAR tool should be restricted to use only the desired CLBs to implement the original design. The PlaceBlock program is used to generate the information containing the locations of the blocked CLBs depending upon the pattern configuration. The PlaceBlock program requires the pattern configuration and the target area in which we want to implement patterned placement as inputs. It generates an UCF file which contains the CLB prohibit locations depending upon the pattern. The PlaceBlock program uses the algorithm described in Fig. 4.

B. Routing Blocker

It is particularly hard to constrain the PAR tool from using routing resources of a blocked CLB. We require these resources when we have to place the complementary part of the design in these CLBs. Hence, we place a self contained dummy hard-macro in all the blocked CLB positions to prevent the PAR tool from utilizing any of the

Require: Initial CLB-(X,Y) Positions
Require: Final CLB-(X,Y) Positions
Require: [CLB_X_Config x CLB_Y_Config] for e.g.[1 x 1]

```
1: j = 0;
2: while CLB_Y_Initial ≤ CLB_Y_Final do
3:    i = 0;
4:    if j%2 = 0 then
5:       CLB_X_Var = CLB_X_Initial;
6:    else
7:       CLB_X_Var = CLB_X_Initial + (2∗ CLB_X_Config);
8:    end if
9:    CLB_Y_Var = CLB_Y_Initial;
10:   while CLB_X_Initial ≤ CLB_X_Final do
11:      X1p = CLB_X_Var + (2∗ CLB_X_Config);
12:      X2p = CLB_X_Var + (4∗ CLB_X_Config)-1;
13:      Y1p = CLB_Y_Var;
14:      Y2p = CLB_Y_Var + (2∗ CLB_Y_Config)-1;
15:      if i%2 = 0 then
16:         BLOCK CLB from (X1p, Y1p) to (X2p, Y2p)
17:      end if
18:      i++;
19:      CLB_X_Var ← X1p, CLB_Y_Var ← Y1p;
20:   end while
21:   CLB_Y_Initial = CLB_Y_Initial + CLB_Y_Config;
22:   j++;
23: end while
```

Figure 4. Placement Blocking Algorithm

blocked CLB resources. The dummy hard-macro is built using Relationally Placed Macros (RPMs) and Directed Routing (DIRT) Constraints. RPMs lock the driver and load pins of the CLB, whereas DIRT constraints utilize the routing resources of the blocked CLB. The RouteBlock script uses the UCF file generated by the PlaceBlock program to locate the positions of the blocked CLBs and generates an XDL file which contains the description of the dummy hard-macros.

IV. Test Designs

We implemented all our test designs on the Xc3s500efg320-4 FPGA available on Digilent Spartan-3E starter kit. The instantaneous power consumption of the FPGA during encryption was measured using a Tektronics CT-2 current probe and an Agilient 6054A, 500MHz oscilloscope with a sampling frequency of 4G Samples/second. An external power supply was used to power the FPGA core, and the FPGA was clocked at frequency of 200KHz–400KHz. It is an important point to note that one encryption is counted as one measurement irrespective of the number of samples the oscilloscope measures. We use the term Single-Ended (SE) design to denote an unprotected design throughout this paper. Measurement to Disclosure (MTD) is a security metric used to determine the resistance of the designs against power analysis attacks. MTDs are the number of encryptions required to correctly predict the secret key. These MTDs are dependent on factors like the secret key used, target platforms and the order in which the input plaintext is fed to a cryptographic algorithm. Hence we use a more robust metric called security gain (SG) [10] or gain over SE to assess the effect of different interleaved placement configurations against DPA attacks. SG is the ratio of MTDs of protected to MTDs of unprotected designs.

A. Small Test Circuit

The Test Design circuit shown in Fig. 5 consists of an AES S-Box implemented using combinational logic whose input is connected to an 8-bit LFSR and output is XORed with an 8-bit Key. The result is stored in register FF1. The dashed box indicates the part of the test circuit on which SDDL is implemented. The register FF2 is implemented in Input/Output Blocks (IOB) simply to drive the output ports.

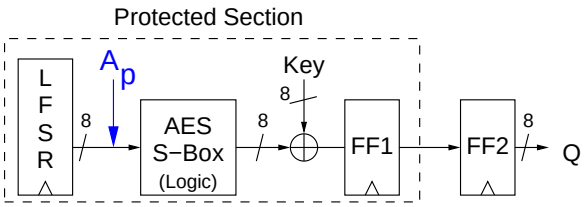


Figure 5. Block Diagram of Test Circuit

B. Attack on Small Test Circuit and Results

We attack the design at the output of the LFSR indicated by the arrow A_p. We use Pearson's correlation to compute the statistical dependence of the instantaneous power consumption with the hypothetical power model i.e the hamming distances [13]. In order to attack the SDDL implementations of the test design we attack the precharged outputs of the LFSR i.e. the precharged inputs to the AES S-Box (Logic). The Hamming Distance (HD) equation for the attack on SE implementations is shown in (1) and for SDDL implementations in (2).

$$P_{est.} = \mathrm{HD}(lfsr_{(i-1)}, \mathrm{SBOX}^{-1}(k_{guess} \oplus Q_i)) \quad (1)$$

$$P_{est.} = \mathrm{HD}(0x00, \mathrm{SBOX}^{-1}(k_{guess} \oplus Q_i)) \quad (2)$$

The post place-and-route implementation results of the SE and SDDL implementations are shown Table II. The test circuit design consumes 49 slices because the AES S-Box is implemented using combinational logic, however this leads to slow designs. The SDDL designs consume 2.3 times more area compared to SE designs because of the extra slices required to precharge the register outputs.

The MTDs of SE and SDDL designs given in Table II indicate the minimum number i.e. the lower bound of MTDs required to obtain the correct key. The Design (5.)

Table II
IMPLEMENTATION RESULTS OF BASIC TEST DESIGN

Design	Slices	FFs/Latches	4 input LUTs	Minimum Delay	Minimum MTD	Gain over SE
1. Test Circuit with Pattern **[1 X 1]**	49	71	27	14.534 ns	1500	1
2. Test Circuit with Pattern **[2 X 2]**	49	71	27	14.412 ns	1500	1
3. Test Circuit with Pattern **[1 X V]**	49	71	27	14.317 ns	1500	1
4. Test Circuit with Pattern **[H X 1]**	49	71	27	14.666 ns	1500	1
5. **SDDL** of Design (1.) **without interleaving**	114	142	86	28.486 ns	23,000	15
6. **SDDL** of Design (1.) **with interleaving**	114	142	86	29.068 ns	50,000	33
7. **SDDL** of Design (2.) **with interleaving**	114	142	86	28.824 ns	50,000	33
8. **SDDL** of Design (3.) **with interleaving**	114	142	86	28.634 ns	50,000	33
9. **SDDL** of Design (4.) **with interleaving**	114	142	86	29.332 ns	50,000	33

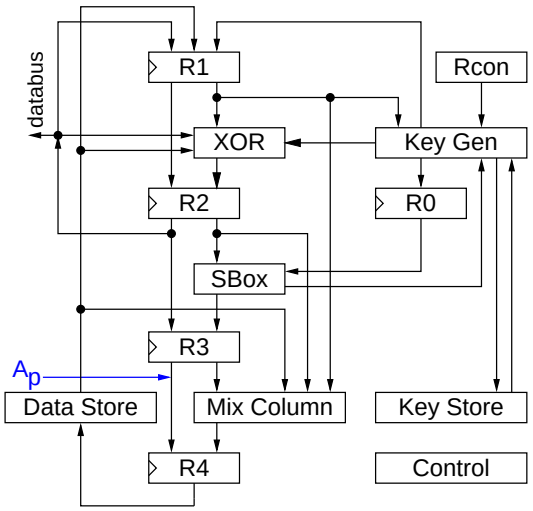


Figure 6. Block Diagram of AES Module

is a non interleaved SDDL implementation of our test circuit using pattern [1 x 1]. The distance between the Direct and Complementary parts of the design was 16 CLBs. Designs (6. to 9.) require more MTDs compared to Design (5.). This conforms our hypothesis that even when the logic and routing of the Direct and Complementary parts are similar, there is a difference in delays due to process variation. It is to be noted that all interleaved designs require a similar number of MTDs irrespective of the pattern configuration. Thus, a user can choose any of the placement patterns show in this paper depending on the design's requirement i.e. availability of routing resources, critical path etc.

C. AES

The Advanced Encryption Standard (AES) [14] is one of the most widely used block ciphers. In this paper we use the AES implementation from [15] which uses a key length of 128 bits. AES applies the same round function ten times to its state during encryption. The round function consists of four different transformations *SubBytes*, *ShiftRows*, *MixColumns* and *AddRoundKey* each changing the state by applying linear, non linear and key dependent transformations.

The data path of the AES implementation is shown in Fig. 6. It is characterized by a pipelined architecture which enables the re-use of registers and minimize the number of internal memory accesses which in turn reduces the number of clock cycles. Five registers R_0, R_1, R_2, R_3, R_4 are used of which R_0 is used exclusively for *RotWord* operation. R_1 is used for key computation and state computation in *MixColumns* operation, R_2, R_3, R_4 are used for state computation. The boxes labeled as *Key store* and *Data store* are 128 bit registers used for Round keys and State Memory respectively.

D. Attack on AES and Results

The point of attack A_p for the AES designs is indicated in Fig.6. Consider the data flow from registers R_2 to R_3. On reset the data in these registers is 0x00. In the first clock cycle, the output of R_3 changes from 0x00 to 0x63 i.e. SBOX(R_2) and the data in R_2 changes to Input data XORed with the key from R_1. In the subsequent clock cycle the data in R_3 changes from 0x63 to SBOX(R_2). This sequence of change in the data of R_3 i.e. $0x00 \rightarrow 0x63 \rightarrow$ SBOX (Key $\oplus$ Input data) occurs every time the AES module is reset. Hence, we can apply the HD model to estimate the power consumption of the register R_3. The power model for the SE cases is given by (3) and for SDDL designs, given by (4).

$$P_{est.} = \mathrm{HD}\left(0x63, (\mathrm{SBOX}((Key_{guess} \oplus Input))_i\right) \quad (3)$$

$$P_{est.} = \mathrm{HD}\left(0x00, (\mathrm{SBOX}((Key_{guess} \oplus Input))_i\right) \quad (4)$$

Table III shows the post place-and-route results of the SE and SDDL implementations. Both AES SDDL designs consume an area 2.92 times larger than the AES SE design. The MTDs from Table III confirm our results from the test circuit namely, that interleaved SDDL is more secure than SDDL without any placement constraints. Also Design (12.) requires 2.3 and 27.5 times more MTDs compared to Design (11.) and Design (10.) respectively.

V. CONCLUSION

One of the vulnerabilities of SDDL for FPGAs is the imbalance of nets in Direct and Complementary paths due to process variation. We explored different placement configurations for SDDL designs to reduce the difference

Table III
SUMMARY OF IMPLEMENTATION RESULTS FROM AES-128 DESIGNS

Design	Slices	FFs	4 input LUTs	Minimum Delay	Minimum MTD	Gain over SE
10. AES SE with Pattern [1 x 1]	371	347	466	15.629 ns	2000	1
11. AES SDDL of Design (10.) **without interleaving**	1086	651	932	30.570 ns	24,000	12
12. AES SDDL of Design (10.) **with interleaving**	1086	651	932	31.258 ns	55,000	27.5

in net delays and propose a new design flow for implementing interleaved SDDL on FPGAs. We implemented the AES block cipher using our design flow and observed that the interleaved SDDL design requires 2.3 times more MTDs than the SDDL design without any placement constraints and 27.2 times more MTDs than AES SE design. For future work, we plan to reduce the effect of early precharge-evaluation and glitches on SDDL for FPGAs and also explore the interleaved placement options on newer technology FPGAs such as Spartan 6.

REFERENCES

[1] T. Huffmire, B. Brotherton, T. Sherwood, R. Kastner, T. Levin, T. D. Nguyen, and C. Irvine, "Managing security in FPGA-based embedded systems," *IEEE Design and Test of Computers*, vol. 25, pp. 590–598, 2008.

[2] S. Trimberger, "Trusted design in FPGAs," in *Proceedings of the 44th annual Design Automation Conference*, ser. DAC '07. New York, NY, USA: ACM, 2007, pp. 5–8.

[3] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Advances in Cryptology - CRYPTO'99*, ser. LNCS, vol. 1666. Berlin: Springer Verlag, Aug 1999, pp. 388–397.

[4] K. Tiri and I. Verbauwhede, "A logic level design methodology for a secure DPA resistant ASIC or FPGA implementation," in *Proc. Design, Automation and Test in Europe (DATE'04)*. IEEE Computer Society, Feb 2004, pp. 246–251.

[5] R. Velegalati and J.-P. Kaps, "DPA resistance for lightweight implementations of cryptographic algorithms on FPGAs," in *Field Programmable Logic and Applications, FPL 2009*, M. Daněk, J. Kadlec, and B. Nelson, Eds. IEEE, Aug 2009, pp. 385–390.

[6] D. Suzuki and M. Saeki, "Security evaluation of DPA countermeasures using dual-rail pre-charge logic style," in *Cryptographic Hardware and Embedded Systems - CHES 2006*, ser. LNCS, L. Goubin and M. Matsui, Eds., vol. 4249. Heidelberg: Springer, 2006, pp. 255–269.

[7] A. Maiti and P. Schaumont, "Improving the quality of a physical unclonable function using configurable ring oscillators," in *Field Programmable Logic and Applications – FPL 2009*, 2009, pp. 703–707.

[8] S. Guilley, S. Chaudhuri, L. Sauvage, T. Graba, J.-L. Danger, P. Hoogvorst, Vinh-Nga, and M. Nassar, "Place-and-route impact on the security of DPL designs in FPGAs," in *Hardware-Oriented Security and Trust, HOST 2008*. IEEE, 2008, pp. 26–32.

[9] P. Yu and P. Schaumont, "Secure FPGA circuits using controlled placement and routing," in *CODES+ISSS '07: Proceedings of the 5th IEEE/ACM international conference on Hardware/software codesign and system synthesis*. New York, NY, USA: ACM, 2007, pp. 45–50.

[10] L. Sauvage, M. Nassar, S. Guilley, F. Flament, J.-L. Danger, and Y. Mathieu, "Exploiting dual-output programmable blocks to balance secure dual-rail logics," *International Journal of Reconfigurable Computing*, vol. 2010, p. 12, Oct 2010.

[11] L. Sauvage, S. Guilley, J.-L. Danger, Y. Mathieu, and M. Nassar, "Successful attack on an FPGA -based WDDL DES cryptoprocessor without place and route constraints." HAL Archives, Sep 2008.

[12] J.-P. Kaps and R. Velegalati, "DPA resistant AES on FPGA using partial DDL," in *IEEE Symposium on Field-Programmable Custom Computing Machines – FCCM 2010*. IEEE, May 2010, pp. 273–280.

[13] E. Brier, C. Clavier, and F. Olivier, "Correlation power analysis with a leakage model," in *Cryptographic Hardware and Embedded Systems – CHES 2004*, ser. LNCS, vol. 3156. Berlin / Heidelberg: Springer, Aug 2004, pp. 135–152.

[14] *Advanced Encryption Standard (AES)*, National Institute of Standards and Technology (NIST), FIPS Publication 197, Nov 2001, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[15] J.-P. Kaps and B. Sunar, "Energy comparison of AES and SHA-1 for ubiquitous computing," in *Embedded and Ubiquitous Computing (EUC-06) Workshop Proceedings*, ser. LNCS, X. Z. et al., Ed., vol. 4097. Springer, Aug 2006, pp. 372–381.