Journal of Electronic Testing (2022) 38:261–277
https://doi.org/10.1007/s10836-022-06012-z

Cross-PUF Attacks: Targeting FPGA Implementation of Arbiter-PUFs

Trevor Kroeger[1] · Wei Cheng[2] · Jean-Luc Danger[2] · Sylvain Guilley[2,3] · Naghmeh Karimi[1]

Received: 15 February 2022 / Accepted: 11 June 2022 / Published online: 30 June 2022

Abstract
The hardware primitives known as Physically Unclonable Functions (PUFs) generate unique signatures based on uncontrollable variations which occur during the manufacturing process of silicon chips. These signatures are in turn used for securing Integrated Circuits either as a secret key for cryptographic modules, or as a medium for authenticating devices. Naturally being a security primitive, PUFs are the target for attacks as such it is important to mitigate such vulnerabilities. This paper in particular investigates PUFs' vulnerability to power-based modeling attacks. Here, we expand upon our previous simulation based Cross-PUF attacks by targeting PUFs realized in real-silicon; namely, we consider PUFs deployed in Field-Programmable Gate Array (FPGA) fabrics. In *Cross-PUF* attacks, a model of a reference PUF is used to attack another PUF realized from the same HSPICE simulated design or the same bitstream in FPGA. We also investigate the impact of such attacks on multi-bit parallel PUFs. The HSPICE simulation results are compared vis-a-vis with the FPGA implementation outcome of these attacks confirming the effectiveness of such simulations. Finally we show that a combination of Dual Rail logic and Random Initialization logic, named DRILL, can be effectively used to thwart such power-based modeling attacks.

Keywords Physically Unclonable Function · FPGA Implementation · Side-Channel Attack · Cross-PUF Attack · Machine Learning · DRILL Countermeasure

1 Introduction

Physically Unclonable Functions (PUFs) are hardware security primitives which were first proposed by Pappu et al. [35] and expanded to silicon devices by Gassend et al. [14]. PUFs are the hardware equivalent to one-way functions whose functionality is based on the unpredictable variations occurring in the manufacturing process of the Integrated Circuit (IC) in which the PUF is embedded [16]. Unlike their algorithmic-level relatives, PUFs are unique to the circuitry in which they are embedded as they owe their uniqueness to the physical specifications of the underlying circuit. PUF primitives, similar to the software-based one-way functions, are useful for device authentication, as well as generating keys for cryptographic cores [10]. These primitives impose a much lower resource overhead compared to the cryptographic alternatives when used for device authentication [16]. A PUF is fed with an input word so-called challenge, and produces a unique response per challenge where these pairings are known as Challenge-Response Pairs (CRPs). The cardinality of the set including the CRPs are different for different PUF types. The PUFs with a small set of CRPs are known as *weak PUFs* whereas the ones with an exponential set of CRPs are called *strong PUFs* [16]. The security of PUFs has been formalized normatively in ISO/IEC 20897 [17].

PUFs are considered to be useful for security purposes and therefore have found their way into various different applications. On a small scale they have been integrated into

Responsible Editor: V. D. Agrawal

Trevor Kroeger and Wei Cheng contributed equally to this work.

✉ Trevor Kroeger
trevor.kroeger@umbc.edu

Wei Cheng
wei.cheng@telecom-paris.fr

Jean-Luc Danger
jean-luc.danger@telecom-paris.fr

Sylvain Guilley
sylvain.guilley@secure-ic.com

Naghmeh Karimi
naghmeh.karimi@umbc.edu

[1] CSEE Department, University of Maryland Baltimore County, Baltimore, MD 21250, USA

[2] Télécom Paris, Institut Polytechnique de Paris, Paris, France

[3] Secure-IC S.A.S. ("Think Ahead" Business Line), Paris, France

RFID tags and smart chips [8, 19], owing to their relatively lightweight designs. They are also particularly useful in anti-counterfeiting applications by producing unique device identifiers [21]. These devices even have found industrial applications into more complex systems such as autonomous vehicles [18, 26] and cryptocurrency [31].

Owing to their involvement in highly secure applications, PUFs are the focus of various attacks. The modeling attack characterizes the strong PUF's input/output relationship to predict the PUF response for each challenge, hence focusing on the PUF's CRPs [37] to model the PUF. This attack takes advantage of Machine Learning (ML) algorithms and has been proven highly effective in modeling all the arbiter-PUF's family, even the most robust ones relying on PUF composition like the XOR-PUF, ML-PUF and interpose PUF [39, 43]. Meanwhile, the PUFs' side channels (power consumption or measurements of error probability) have also received attention in recent years as they have been shown to be supporting successfully the modeling attacks [4, 5, 9, 27, 38].

This paper focuses on the power side-channel based attacks in order to directly induce the PUF response from the power traces; not by using CRPs for modeling attacks. In this paper, we expand our previous work on the possibility of the *Cross-PUF* attacks; i.e., the capability of attacking one PUF with a model created from another instance of a similar design [22], and show the feasibility of such an attack on Field-Programmable Gate Arrays (FPGAs) where the same bitstream is used for programming different FPGAs of the same type. Indeed, the previous work on *Cross-PUF* attacks pertained to simulated PUF circuitries, and this paper launches these attacks on the FPGAs implementations of such PUFs. We also investigate how vulnerable the multi-bit response parallel PUF implementations realized in FPGA are, and how our proposed countermeasure [23], so-called DRILL, can resist such attacks in the real-silicon. =

The contributions of this work include:

- Demonstrating the effectiveness of the power-based modeling attacks on the arbiter-PUFs implemented in real silicon;
- Successful launching *Cross-PUF* attacks on arbiter-PUFs' FPGA implementations;
- Investigating the vulnerability of multi-bit response parallel PUFs to power-based modeling attacks using real-silicon data;
- Demonstrating the effectiveness of our countermeasure (say DRILL) against the aforementioned single-PUF and cross-PUF attacks in real silicon.

The rest of this paper is organized as follows. First, relevant background is discussed in Sect. 2. This is followed by the related work on PUF side-channel attacks in Sect. 3. The threat model and attack methodology is discussed in Sect. 4. The DRILL countermeasure is discussed in Sect. 5. The setup for the experiments is relayed in Sect. 6 before getting into the results of our investigations in Sect. 7. Finally, Sect. 8 concludes the paper.

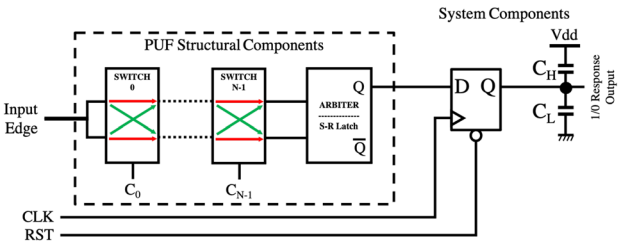


Fig. 1 Structure of an arbiter-PUF [14]. This includes both the PUF Structural Components as well as the Systems Components

2 Background

2.1 Arbiter-PUF

The arbiter-PUF generates a random response based on the delays of cascading multiplexers. As shown in Fig. 1, the delay of two paths is realized through the chain of multiplexers (switches) is compared for each given challenge via an arbitration unit, mainly implemented using an S-R latch. In an arbiter-PUF, each pair of multiplexers are fed with a bit of the input challenge to determine whether the input is connected straight to the output or swapped. The arbiter-PUF is queried with a rising transition, which then propagates through the delay chains with an accumulating delay related to each of the stages on each path. Depending on transitions' arrival time to the two inputs of the arbiter the response would be a '0' or a '1'. Indeed the sign of the difference in these path delays determines the response, not their values *per se*.

Note that in a circuit with an embedded PUF, once generated the PUF response must be stored for eventual use. This usually takes place in a simple register (namely a D-Flip-flop, as shown in Fig. 1). As we will show, these system components induce large leakages which can ultimately reveal the response of the PUF through side-channel analyses.

Arbiter-PUFs are mostly implemented in single-bit versions mainly to avoid high power consumption in time. However, such implementations suffer from low throughput as they only produce a single bit response at a time, and also they are more vulnerable to power-based modeling attacks. To alleviate such shortcomings, multi-bit response parallel bits are used in real circuits; albeit with a few parallel PUFs as parallelization does not come without a cost, and increases

the amount of power and area the PUF instance utilizes [16]. Thus for the sake of complexness, we consider both single- and multi-bit response parallel arbiter-PUFs in this study.

2.2 FPGA-based PUF Implementations

FPGAs are capable of realizing complex digital logic circuitries through a programmable IC. While there are various components on an FPGA, most digital logic is implemented through SRAM-Based Look-Up Tables (LUTs) [11]. The propagation delays between input and output signals of these LUTs contribute to the delay of the whole circuit. As expected, these delays also contribute to determining the response of delay-based PUFs such as arbiter-PUFs implemented on FPGAs.

The placement of the utilized LUTs (in the underlying FPGA) and routing between them also affects the propagation delays, and in turn the arbiter-PUF response. Thereby, implementing arbiter-PUFs in FPGAs needs careful consideration to be able to meet PUF requirements, e.g., randomness, uniformity, uniqueness, and reliability. Indeed, a thorough understanding of the LUTs' structure is required to appropriately route the signals [1, 40] in FPGA implementations of PUFs. It is worth noting that FPGA implementations of PUFs have known issues with their uniqueness property; by increasing the logic delay from the LUTs these uniqueness issues can be mitigated [29]. Also through programmable delay lines, any issues with the PUFs' uniformity can also be mitigated [30]. A full description of our designs' layout realized in an FPGA is discussed in Sect. 6. A *Slice* for the Xilinx Spartan-6 FPGA is shown in Fig. 2. Note that this internal structure consists of two storage components, a LUT and some internal multiplexers. There are four internal structures in a slice. The connections within this structure are unchanged during operations.

The structure of the arbiter-PUF must be carefully placed into these structures to ensure that all of the multiplexers (switches) and the interconnections within the chain are identical. The placement for the switch within the slice is shown in Fig. 3.

3 Related Work

The strong PUFs like arbiter PUFs are the targets of powerful modeling attacks carried out via their CRPs. The attacker gathers a set of CRPs and then uses that set to learn a model by means of ML algorithm which can predict the response to the previously unseen challenges. Many robust PUFs relying on composition of arbiter-PUFs have been devised to resist ML attacks. The most recent one is the Interpose PUF [34], which was claimed to be resilient to modeling attacks. But they do not resist to powerful ML attacks [7, 39, 43].

Moreover, modeling attacks can be supported by monitoring side-channel of the PUF during the runtime to infer the response when it is operating. For instance [5, 9] exploit the reliability of the PUF responses, [38] utilizes power and timing to recover the delay model of the PUF.

Our work differs from these researches as we focus directly on the PUF digitization and storage component, without trying to model the PUF itself, but rather attacking any PUF by using the power model of the reference PUF. We show that this *Cross-PUF* attack undermines the security of the PUF as a whole and leaves PUFs vulnerable to both single-PUF and *Cross-PUF* attacks.

There is also a lot of research in the realm of protecting ICs from side-channel attacks. These researches are mainly concerned with voltage regulation of the device and the addition of noise to make analysis of the IC more difficult [2, 20, 32, 44, 45]. However, the protection presented in this

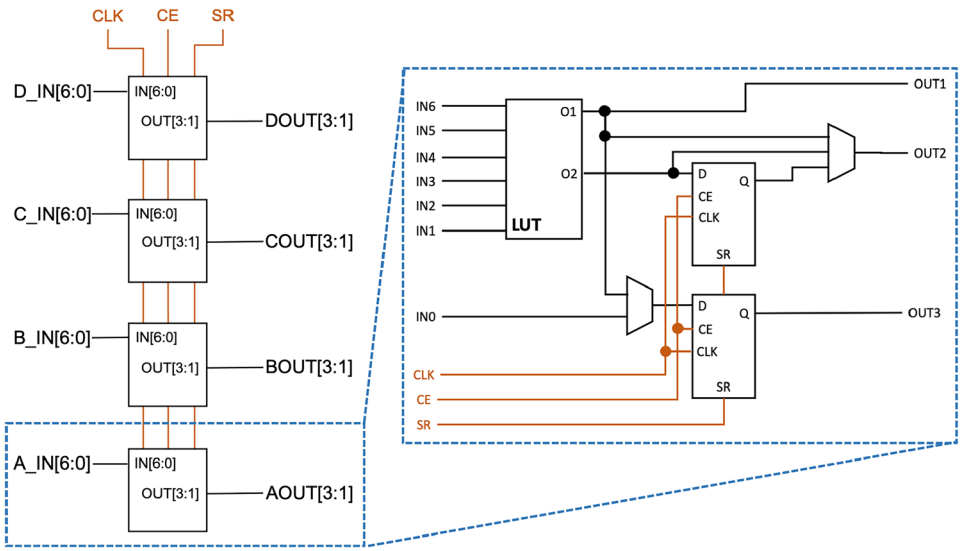


Fig. 2 Structure of Xilinx Virtex-6 Slice. The LUT logic structures use shared CLK, CE, and SR inputs [41]

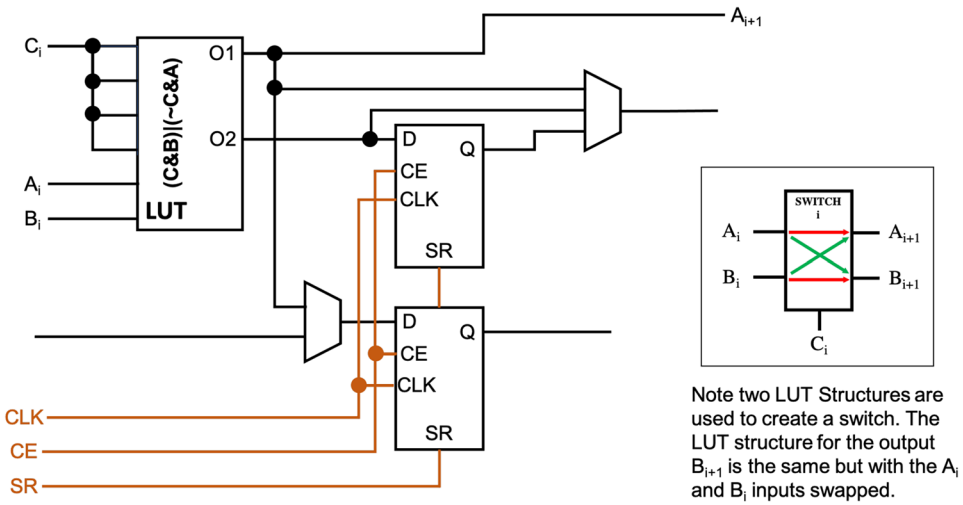


Fig. 3 Placement of a switch structure within the LUT components of a slice

paper is specific to PUFs and differs from these methods as the protection proposed here is integrated into the PUF itself instead of a full chip implementation providing protection for this specific application.

4 Modeling Attacks and Our Threat Model

Being security primitives, PUFs are fruitful targets for the adversaries. Attackers mainly gravitate towards attacks that are based on intercepting the CRPs of a target PUF to model its behavior, and thus predict the response for challenges that have not been used previously [36, 37]. Likewise the behavior of the PUF can be modeled through the IC's power side-channel in a similar fashion [27, 38]. In both of these cases only a single PUF instance is targeted, meaning that the PUF which is modeled is also the PUF that is targeted. This is the fundamental difference between the previous research and what is seen in the *Cross-PUF* attack which uses the power consumption of one PUF to attack another realized using the same blueprint (typically a GDSII file in ASIC and a bitstream in FPGA).

4.1 Threat Model

In this work, we postulate an attacker who has the ability to record the power consumption of an IC in operation. Moreover, the attacker also has the ability to acquire a chip with a reference PUF of the same design as the target PUF. The attacker observes the power consumption of the IC that includes the reference PUF while in operation, and records the power traces with the corresponding responses. These pairings are then used to create a model that can be used to attack the target PUF. This non-invasive "profiling" attack enables the adversary to create a model from the power traces of the reference PUF in order to infer the responses of the target PUF. Such attacks do defeat the requirement about "tamper-resistance" stated in clause 5.5.5 of the Part 1, entitled "Security requirements", of the International Standard ISO/IEC 20897-1:2020 on PUFs [17].

❶ **Self-PUF attacks.** When the adversary leverages the model to predict responses from unseen challenges on the same chip, then the attack is termed "*Self-PUF*".
❷ **Cross-PUF attacks.** When the adversary employs the model on different PUF instances, then the attack is termed "*Cross-PUF*". It is assumed that the attacker acquired a valid IC that has passed the testing required for regular distribution to the end-user.

It is important to point out that the aforementioned *Cross-PUF* attacks cannot be performed by characterizing solely the PUF's CRPs. Contrary to the popular modeling attack, the *Cross-PUF* has no access to the response. It relies on the model of the power consumption of its response, not the relation challenge ⇒ Response. Indeed, the power consumption of the PUF in operation reveals the response due to the underlying similarities in the design instances; more specifically those concerned with the registration of the PUF's response [22]. Hence, the *Cross-PUF* attacks allow the adversary to target all the protected arbiter-PUF against modeling attacks. For instance the controlled-PUF [13] or the composition of arbiter-PUFs like the recent and robust interpose-PUF [34] could be the targets of this profiling side-channel attack.

5 Countermeasure

To thwart the *Cross-PUF* attack we propose the use of the *DRILL* countermeasure [24, 25]. DRILL incorporates two primary techniques: Dual Rail Logic (DRL)

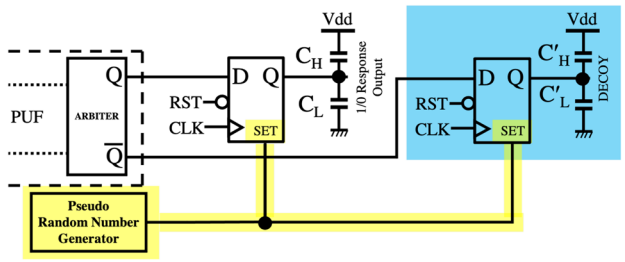


Fig. 4 DRILL countermeasure: consisting in DRL (highlighted in blue) and RIL (highlighted in yellow) implemented on top of an standard arbiter-PUF

and Randomized Initialization Logic (RIL) both of which attempt to reduce the SNR (signal-to-noise ratio) of the power traces and hamper the attacker's ability to successfully model/attack the PUFs. The DRILL countermeasure expands off of the presented arbiter-PUF instance shown in Fig. 1 to create the countermeasure shown in Fig. 4.

The DRL portion of the countermeasure adds two Flip-Flops fed by the arbiter outputs Q and their opposite $\bar{Q}$. This follows the concept presented in [, § 7.3] where the leakage of the Flip-Flop value is mirrored by an equivalent Flip-Flop. The leakage present on the Flip-Flop instances is due to the capacitive loading of those Flip-Flops therefore ideally this loading will be identical. However, such equality in power consumption is admittedly hard to achieve in practice in real silicon [24, 28].

The RIL portion of the countermeasure adds randomness to the setting/resetting mechanism of the Flip-Flops storing Q and $\bar{Q}$. Indeed increasing the randomization is an effective countermeasure as also noted by [42]. With such a random setting mechanism, instead of the Flip-Flop only portraying a single transition (from 0 to 1), it displays the transition for 1 to 0 as well. As previously stated the capacitive loading on the outputs of the Flip-Flops plays a large role in the leakage of the response registration. Balancing such loading within the FPGA requires careful management on placement and routing. In this paper, in our FPGA implementation, to ensure equal loading the decoy bit was handled as an actual data bit from the PUF instance.

6 Experimental Setup

In this work we set up two sets of experiments. One set includes our transistor-level simulations and the other set is based on a real-silicon implementation on a commercial FPGA fabrication.

6.1 Simulation Setup

We used Synopsys HSpice version 2016.06-SP2-2 on a 64-bit Linux System. The simulations were performed at 80°C and Vdd = 1.1 V. The HSpice Simulated experiments were performed for new devices (i.e., no aging). We used the open-source NANGATE 45 nm library in our HSpice simulations [33]. The simulations were preformed pre-placement and routing. The process variations were realized in the target circuits using a Gaussian distribution for:

- transistor gate length L: $3\sigma = 10\%$,
- threshold voltage V_{TH}: $3\sigma = 30\%$, and
- gate-oxide thickness t_{OX}: $3\sigma = 3\%$.

In this paper, five instances of unprotected and the DRILL-protected PUFs were simulated, and their response and power traces were extracted for 12,000 challenges. Similarly two instances of unprotected as well as DRILL-protected Multi-bit Parallel PUFs (more precisely 2-bit Parallel PUFs) were simulated. They are fed with 12,000 challenges; their power traces and responses were collected as well. To make these results closer to a realistic scenario Gaussian noise (N) was added to the extracted traces (X) as follows:

$$Y = X + N \qquad \text{where } N \sim \mathcal{N}(0, \sigma^2).$$

In these investigations, noise levels of σ={2.5e-4, 16e-4, 32e-4, 64e-4} were added to the simulated traces to provide a range of noise levels that were reasonable for real-silicon implemented PUFs. The traces include the current drawn from the voltage source (Vdd) between the time that the transition feeds the PUF and when the response is arbitrated and registered in the storage Flip-Flop at a rate of 1 ps.

6.2 FPGA-based Implementations

We implemented 5 instances of an unprotected PUF and 5 instances of DRILL-protected PUFs on FPGA along with 2 instances of unprotected and DRILL protected 2-bit Parallel PUFs. The target platform for implementation was the Sakura-G FPGA Development board. This board is designed to ease side-channel analyses. It contains a Xilinx Spartan-6 which was programmed with Xilinx ISE 14.7. The board is shown in Fig. 6.

The PUFs were implemented using the hard macro functionality available within the FPGA toolset. This allowed for the components of the PUF to be specifically placed on the device such that the operation of the PUF would be

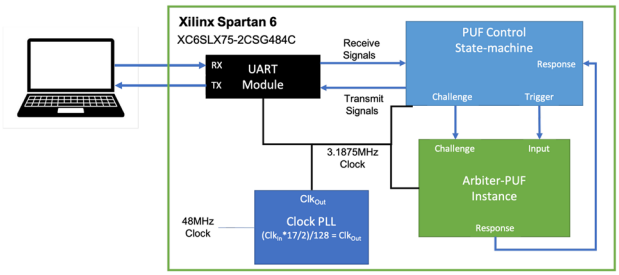


Fig. 5 FPGA PUF Implementation Block Diagram

consistent between implementations. We added modules for providing the challenge to the PUF, triggering the PUF's operation, and sending the resulting response. This is shown in the block diagram of Fig. 5. The system operates such that the PUF produces the response in a single clock cycle when it is triggered.

This development board is specifically designed for performing power-based measurements, and contains circuitry to measure the voltage fluctuation of the device while in operation. This circuitry provides 20dB of amplification to the signal while also filtering out high-frequency noise. We did not remove any of the decoupling capacitors and note that the balancing of the DRILL countermeasure as well as the chip as a whole will have an imbalance based on the location of the capacitors as well as the placement of the PUF within the device. Figure 7 shows the placement of the PUFs within the FPGA. The PUF is implemented as a hard macro. This hard macro is placed in different locations in each bit file, as shown in the figure, to produce each of the 5 PUF variants we investigate. To be clear each PUF variant has its own bit file.

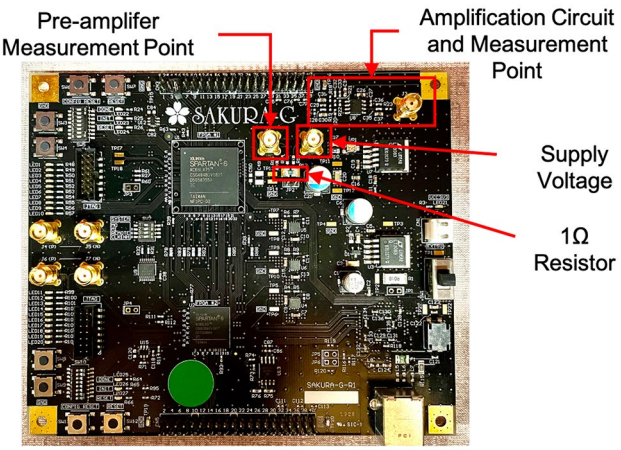


Fig. 6 Sakura-G FPGA development board [15]

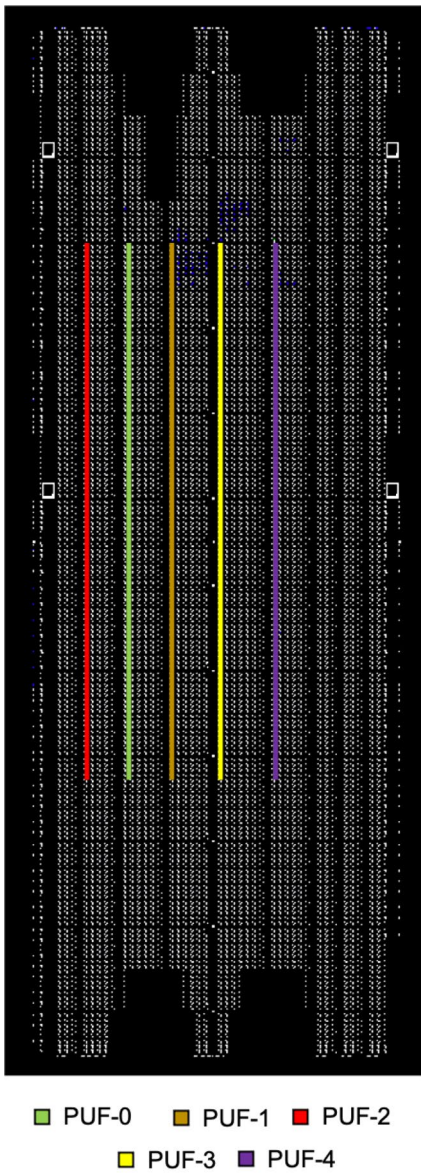


Fig. 7 Placement of the various PUF implementations within the fabric of the Sakura Development Board. The location of PUF-0, PUF-1, PUF-2, PUF-3, and PUF-4 are shown in green, orange, red, yellow, and purple colors, respectively

Power traces from the Sakura board were collected via a Teledyne Lecroy Waverunner 8254M Oscilloscope at 20GS/s. The signals were collected from the rising input edge of the PUF until the response registration in the Flip-Flop, i.e., the same time period as when the simulated traces were collected. To mirror the primary and decoy output bits for the DRILL countermeasure, only a total of 5% of additional slices were added to the overall area of the PUF. These many slices impose negligible additional overall power drawn by the device.

The performance metrics of the implemented PUFs will be discussed in the results section (Sect. 7).

6.3 Modeling Details

We used the Support Vector Machine (SVM) [1] algorithms to model our simulation-based and FPGA-based PUFs. Due to the nature of the PUFs, the Radial Basis Function (RBF) kernel is used to transform the input related to a nonlinear space. This model was chosen due to its effectiveness in modeling PUF behavior. In evaluating the multi-bit PUF results, we utilize an "one vs rest" scheme to provide the highest accuracy in our results.

6.4 Signal-to-Noise Ratio

To evaluate the signals and the PUFs' susceptibility to attack, the SNR is used. As stated in [28, §4.3.2], this is the ratio between inter-variance and intra-variance amongst the power traces and is frequently used in side-channel analysis [12, 28].

$$\mathrm{SNR} = \frac{\mathrm{Var}(\mathit{Signal})}{\mathrm{Var}(\mathit{Noise})} = \frac{\mathrm{Var}([\mathrm{Mean}(\mathcal{L}_0), \mathrm{Mean}(\mathcal{L}_1)])}{\mathrm{Mean}([\mathrm{Var}(\mathcal{L}_0), \mathrm{Var}(\mathcal{L}_1)])}. \quad (1)$$

In this work, we compare the SNR of the simulated PUF instances to their counterparts implemented within the FPGA.

6.5 Modeling Accuracy

The accuracy of the modeling attack, presented in the experimental results, is defined as:

$$\mathrm{Accuracy} = \frac{N_{\mathrm{corr}}}{N_{\mathrm{total}}}, \quad (2)$$

where N_{corr} denotes the number of responses predicted correctly, and N_{total} is for total number of tests.

7 Experimental Results

7.1 PUF Performance Metrics

To ensure that the FPGA implementation of the arbiter-PUF presented in Fig. 1 is valid, we assess the implementation based on standard PUF metrics. These metrics are Randomness, Uniqueness, and Reliability [6]. The results of these metrics tests are shared below along with assessments to their meaning.

[1] For the sake of brevity, we omitted the experimental results of another two algorithms, namely Decision Tree and Random Forest, since SVM provides a higher accuracy than that of those two.

Table 1 NIST Randomness Test Results

Test	Passed Percent
Frequency	97%
Frequency Block	100%
Runs	99%
The Longest Run	97%
Binary Matrix Rank	100%
FFT	100%
Non-overlap. Template	97%
Overlapping Template	100%
Universal	100%
Linear Complexity Test	87.5%
Serial	100%
Approx. Entropy	100%
Cumulative Sums	96%

Randomness Randomness is the ability to predict the value before it is generated. The randomness metric is defined by NIST standards [3]. The randomness test values for all five PUF instances for 100,000 responses are shown in Table 1. Notably, the Random Excursion and Random Excursion Variants tests optimally require over one million input values and therefore are not shown in this table [3]. The uniformity is taken into account by the Frequency Randomness test. This table shows that the PUF implementation meets the requirements of randomness of a correct implementation of the PUF.

Uniqueness Uniqueness is the metric that describes the PUF's unclonability, i.e., whether one PUF is unique from another instance. The results of investigating the uniqueness of the 5 FPGA implemented PUFs with respect to one another are shown in Table 2. Here the average percentage of similar responses between PUF instances is 51.19% which is only 1.19% away from the ideal value 50.00%.

Reliability/Stability For a PUF to be useful it should retain its value for a particular challenge. To assess this metric a comprehensive investigation should be made over a long period of time (to consider device aging impacts) and also at various temperatures. In this investigation, we

Table 2 The normalized Hamming distance between the response of each PUF pairs when fed with the same challenge bits; representing the uniqueness of the generated responses

	PUF-4	PUF-3	PUF-2	PUF-1
PUF-0	37.13%	39.98%	64.09%	56.86%
PUF-1	46.93%	58.43%	54.25%	—
PUF-2	47.88%	47.46%	—	—
PUF-3	58.87%	—	—	—

Table 3 Reliability of 5 PUF FPGA instances replayed twice for 12,000 responses. Percentage shown is the difference between the two runs

	Percent Difference
PUF-0	0.98%
PUF-1	0.88%
PUF-2	1.58%
PUF-3	0.52%
PUF-4	0.99%

observe more temporal reliability based on applying a set of challenges two times to the device. The percentage of differences between these two runs is shown in Table 3.

Note that the instances of the parallel PUF are derived from these instances therefore their performance metrics hold similar results to those shown here.

The takeaway from the metrics assessment is the FPGA implementations of the PUF are valid PUF instances for practical applications.

7.2 HSPICE Simulated PUF Results

The HSPICE simulated PUF instances were attacked using both *Self-PUF* and *Cross-PUF* methodologies to determine the feasibility of these attacks. Figure 8 depicts the average accuracy of the *Self-PUF* and *Cross-PUF* attacks for all 5 PUF instances implemented at the transistor level and simulated using HSpice. It can be observed that the unprotected PUF instance displays a high level of accuracy for both the *Self-PUF* and *Cross-PUF* attacks. Although the modeling accuracies start to decline for both types of attacks when σ = 64e-4, the accuracies are still > 85%. Thus the attack can still be considered successful. On the other hand, the results

Table 4 The maximum SNR for the simulated PUF traces at the various noise levels for unprotected and DRILL-protected PUFs

	σ = 2.5e-4	σ = 16e-4	σ = 32e-4	σ = 64e-4
Unprotected	12.224361	0.299846	0.079410	0.021712
DRILL	0.034492	0.001739	0.000761	0.000970

confirm that the DRILL does have a large impact in decreasing the accuracy. Indeed at noise levels where σ > 16e-4, the attack can be deemed ineffective as the accuracy has fallen to 60% and below. Note that the situation in which there is no noise present in the system is unrealistic. Indeed, there is always some algorithmic noise (i.e., noise arising from the activity of the surrounding logic gates). For noise with σ > 2.5e-4 the SNR is low when observing real PUF implementations as we will discuss in the next section, but it is included here to show how well DRILL decreases the SNR.

To have a clear view of the accuracies shown in Fig. 8, we further assess the SNR levels of the simulated traces. In particular, knowing the SNR of the unprotected simulated PUF traces will allow for a comparison between the simulation PUF instances and those implemented on FPGA. The SNR values are shared in Table 4. From these SNR values it can be seen that the SNR drops significantly if the DRILL countermeasure is used i.e., in simulation it achieves the goal of lowering the SNR.

7.3 Real-Silicon Implementation Power Traces

Before reviewing the results of the attack on FPGA implemented PUFs, observations can be made on their collected power traces. These power traces are shown in Fig. 9. To show the period of time in which the Flip-Flop is

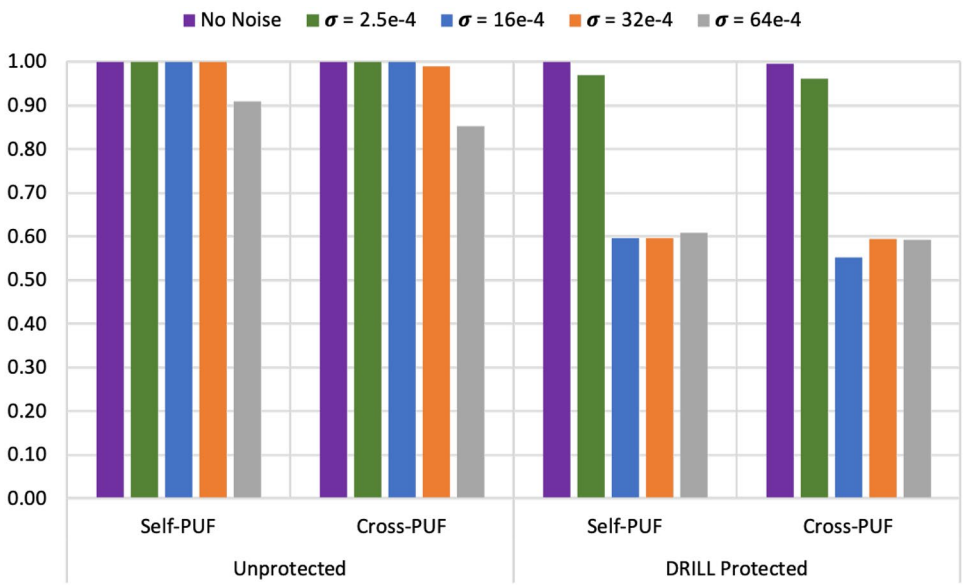
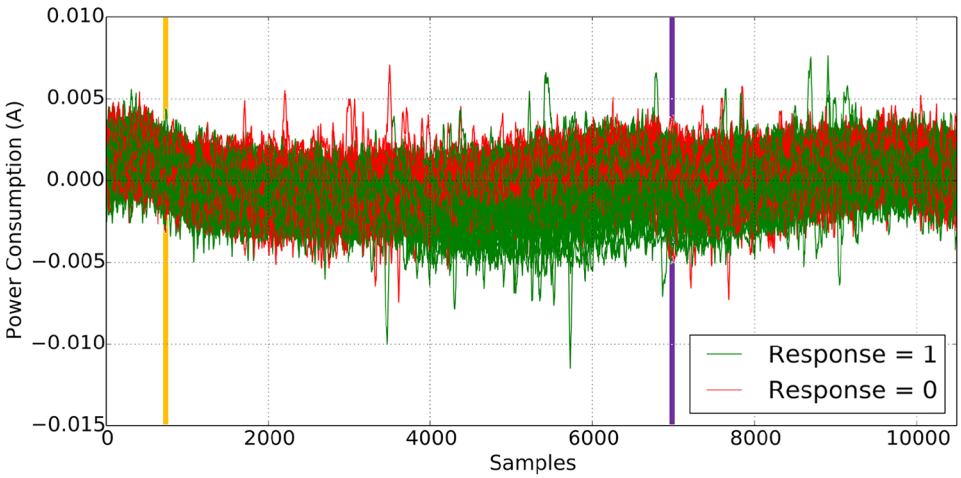
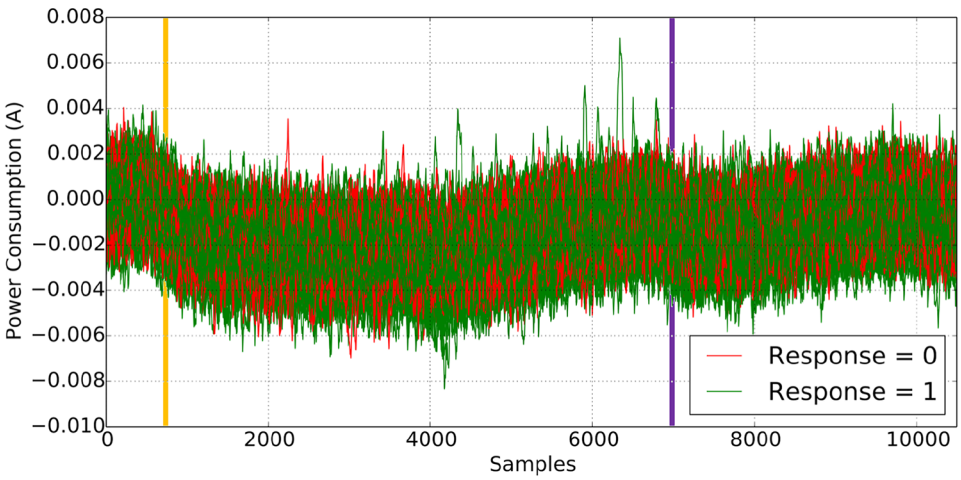


Fig. 8 *Self-PUF* and *Cross-PUF* attack accuracy for simulated PUFs showing both unprotected and DRILL-protected PUF instances for various levels of noise



(a) Traces for Unprotected PUF-0



(b) Traces for DRILL-protected PUF-0

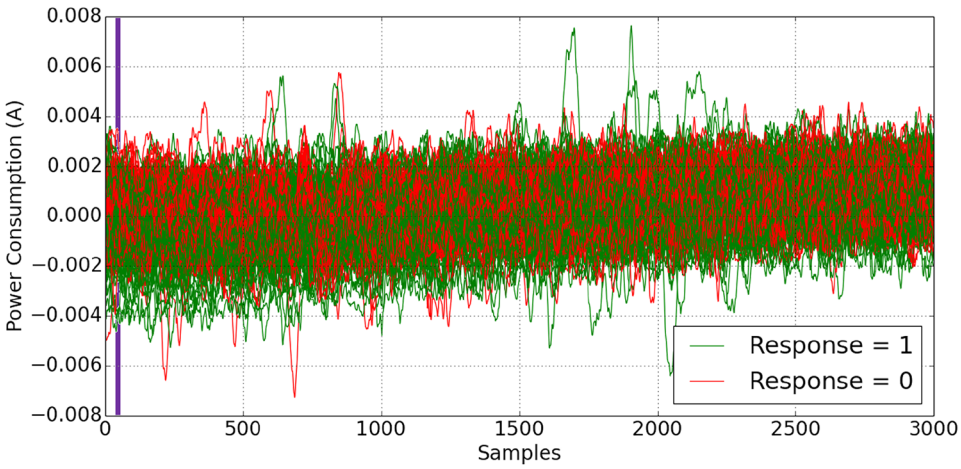
Fig. 9 Superimposing 200 traces from PUF-1 unprotected and DRILL-protected instances implemented on an FPGA fabric. The orange line is when the PUF is triggered with the rising edge, and the purple line is when the response Flip-Flop registers the response

operating, the zoomed traces are shown in Fig. 10. Specifically, Fig. 9(a) (zoomed in Fig. 10(a)) shows the traces for an unprotected PUF for which no discernible pattern can be made to distinguish the responses of 0 from the response that are 1. This is the same for the DRILL-protected traces shown in Fig. 9(b) (zoomed in Fig. 10(b)). The absence of a pattern within these traces means that the noise levels are high enough to hide the phenomenon needed to determine the response, necessitating more advanced means to do so.

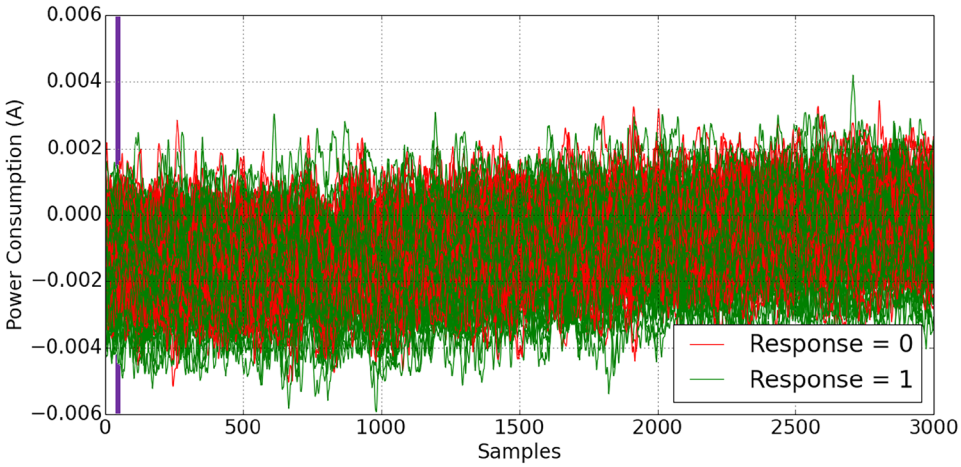
As the noise plays a factor in obscuring the response, it is important to understand the relationship between the noise and the ability of distinguishing the response of the PUF. We therefore extracted the SNR of those PUFs as shown in Table 5 for both unprotected and protected instances. As shown in the table, the peak SNR ranges from ≈ 0.072 to ≈ 0.282 and although the PUFs exhibit similar behavior some PUF's have higher SNR than others. In the simulation, this level of SNR are comparable to those with the added noise levels of $\sigma = \{16e\text{-}4, 32e\text{-}4, 64e\text{-}4\}$, hence we will compare the attack accuracies of the FPGA implementations with the simulation results extracted for the noise levels of $\sigma = \{16e\text{-}4, 32e\text{-}4, 64e\text{-}4\}$. Another important observation from these results is that the implementation of DRILL within the FPGA lowers the SNR, as all of the resulting SNR levels for these traces are below those of the unprotected PUF instances. This confirms the effectiveness of our DRILL countermeasure against power-based modeling attacks.

7.4 *Self-PUF* Results

The first set of results depicts the accuracy of *Self-PUF* attacks on FPGA implementations of the arbiter-PUF. As mentioned earlier, in these attacks the model of a PUF is used to attack that very same PUF, i.e., to predict its responses from any unseen challenges. These attacks are



(a) Traces for Unprotected PUF-0



(b) Traces for DRILL-protected PUF-0

Fig. 10 Zoomed-in traces from PUF-1 for unprotected and DRILL-protected instances implemented on an FPGA fabric. The purple line is when the response Flip-Flop registers the response

used as a basis for any subsequent attacks since they logically have a higher accuracy due to the relevance of the model to the actual PUF. For these investigations, 1,000 traces were used for training and the model was tested against 11,000 traces.

Unprotected PUFs The results of the *Self-PUF* attacks on unprotected PUFs are shown in Fig. 11. In the figure, the simulated traces are shown for the noise levels which correspond to those seen within the FPGA. This figure shows that for each PUF instance the *Self-PUF* attack accuracy ranges from 92.62% to 94.33%, which is a high level of accuracy for predicting the response of the PUF. Moreover, the accuracy is consistent with the simulated PUF results.

DRILL-Protected PUFs The results of attacking the PUF equipped with DRILL are shown in Fig. 12. As depicted, the accuracy is significantly lower than the unprotected PUF shown in Fig. 11. The DRILL-protected PUF experiences an accuracy ranges from 55.86% to 73.68%. This confirms the efficiency of the DRILL protection in real silicon. As illustrated, the accuracy of the attack on PUF-2 is higher than the other DRILL-protected instances. This is likely because its SNR level is also higher than the other PUF instances, as seen in Table 5. This

Table 5 Maximal SNR of the five unprotected and five DRILL-protected PUFs implemented within the FPGA

	PUF-0	PUF-1	PUF-2	PUF-3	PUF-4	Avg. All
Unprotected	0.282	0.228	0.161	0.195	0.072	0.188
DRILL	0.0377	0.0034	0.0549	0.0070	0.0149	0.02358

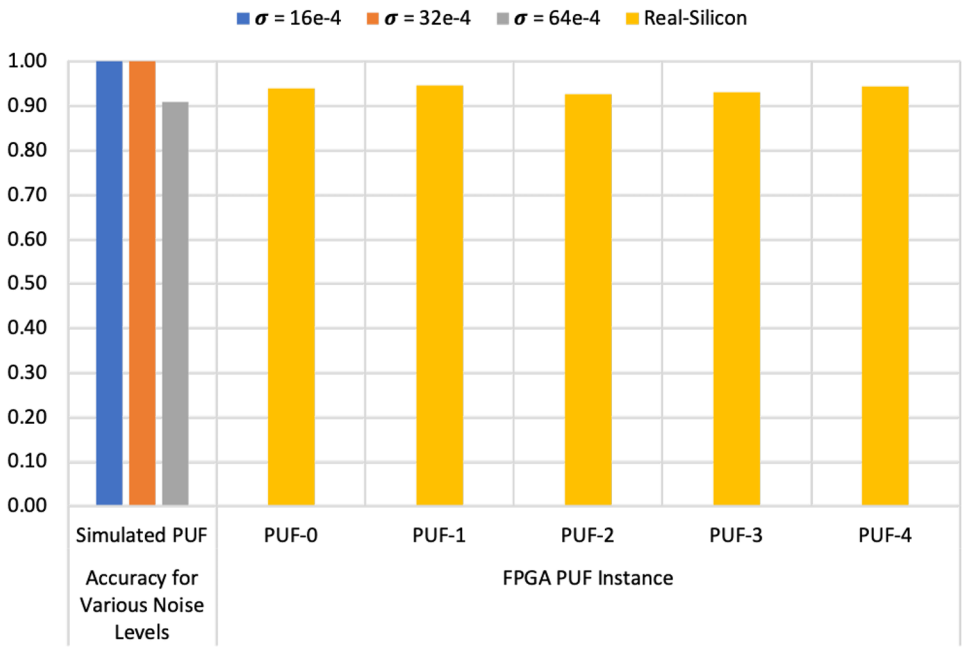


Fig. 11 Accuracy of *Self-PUF* attacks on unprotected PUFs realized in FPGA. Simulation attack accuracies are shown for the noise levels with comparable SNR

means that the placement of FPGA implemented PUFs can affect their vulnerability to attack (recall from Fig. 7 that each implemented PUF is different only in the placement of the hard macro otherwise the overall design and layout within the FPGA are the same). Overall, these attack results are also consistent with the simulated results shown earlier.

The takeaway point from these results is that unprotected PUFs are vulnerable to *Self-PUF* attacks, and DRILL can be used to effectively hinder modeling attacks that aim at inferring the PUFs' response based on their power consumption.

7.5 *Cross-PUF* Results

Cross-PUF attacks, as previously described, relate to the cases in which a PUF model created from a reference PUF is used to attack another PUF. What follows shows the accuracy of such an attack on unprotected and DRILL-protected PUF implementations. The *Cross-PUF* attack investigations were in each case performed where 1000 traces from the reference PUF were used for training and evaluation was performed against 11,000 traces of the target PUF.

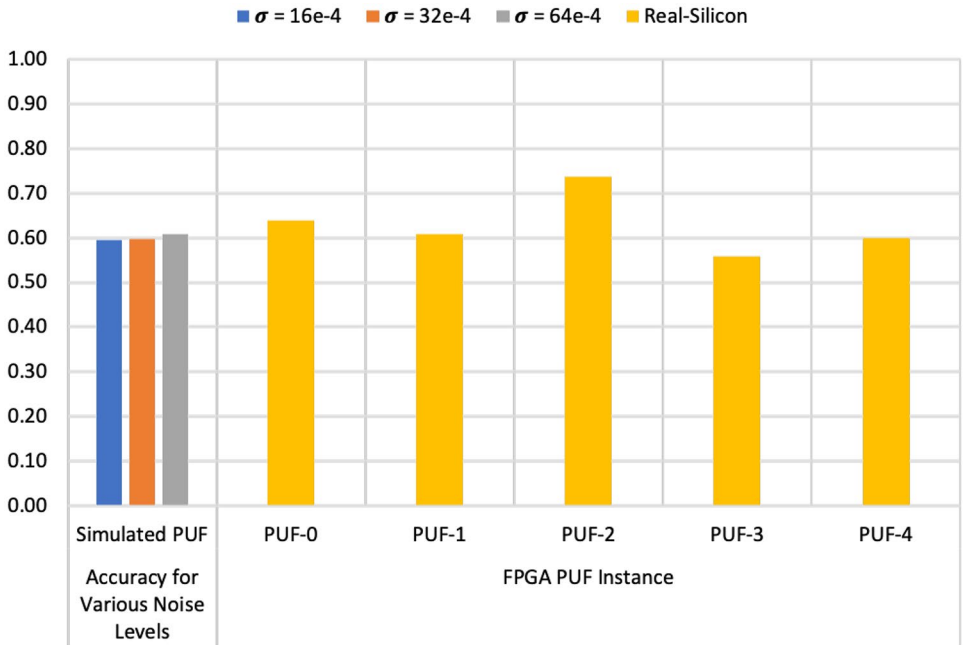


Fig. 12 Results of DRILL-protected *Self-PUF* attacks on FPGA-implementation of PUFs. Simulation attack accuracies are shown for the noise levels with comparable SNR

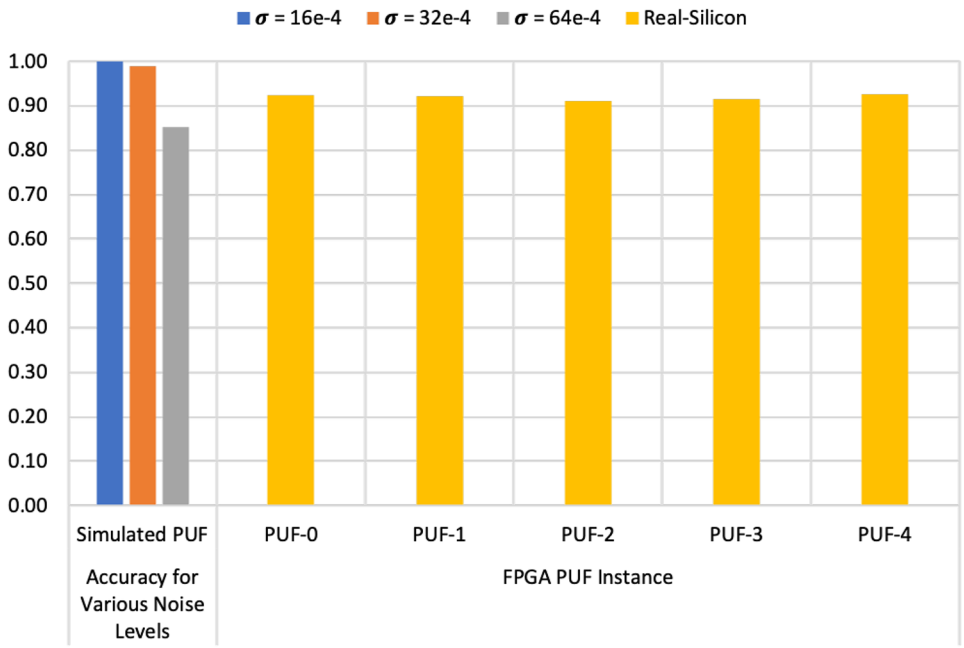


Fig. 13 Results of *Cross-PUF* attacks on FPGA Implemented PUFs. Simulation attack accuracies are shown for the noise levels with comparable SNR

Unprotected PUFs The results of unprotected *Cross-PUF* attacks are shown in Fig. 13. This figure presents the average accuracy of the *Cross-PUF* attack when the stated PUF is used to create the model for the attack. The figure shows that each PUF instance can be used as a reference PUF and still while achieving high accuracy in attacking a different PUF realized from the same design. The accuracy ranges from 90.98% to 92.59% which is only slightly less than the accuracy of the *Self-PUF* attacks. Note that this is also consistent with the simulated PUF results with similar SNR.

DRILL-Protected PUFs Logically, since the *Self-PUF* instances of DRILL were unsuccessful the results of the *Cross-PUF* attacks on DRILL should be unsuccessful as well. This hypothesis is indeed confirmed when looking at the results of the *Cross-PUF* attacks on DRILL-protected PUFs shown in Fig. 14. In this figure, the accuracy of the attack has a maximum of 53.18%, which is very close to the ideal accuracy of 50%. When compared to the *Self-PUF* attack these results are even more promising as it shows that DRILL effectively thwarts the *Cross-PUF* attack as all the accuracies for each PUF model are decreased. This holds

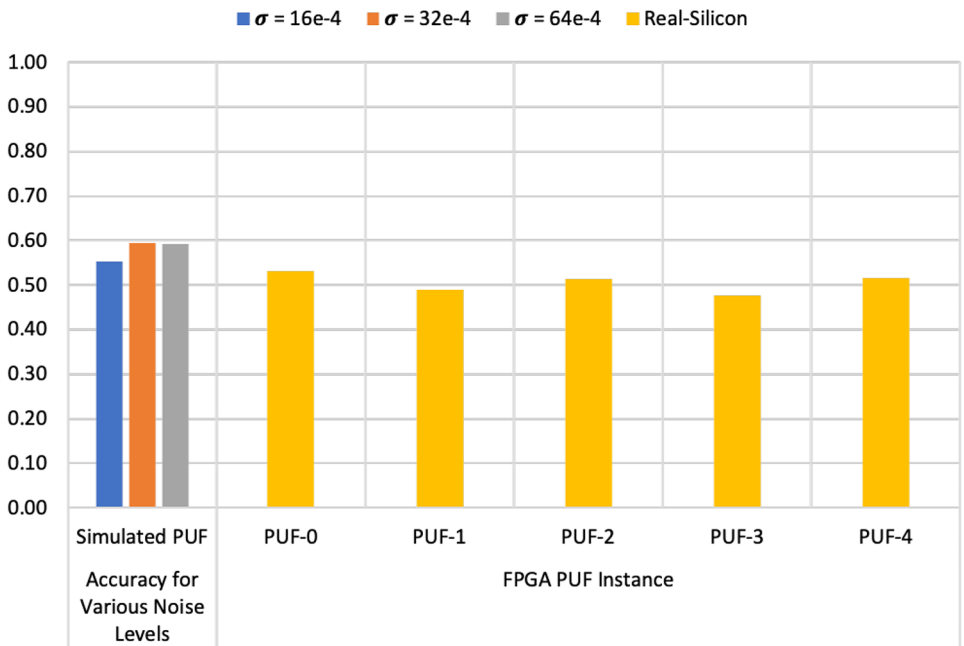


Fig. 14 The attack accuracy of *Cross-PUF* attacks for DRILL-protected PUFs. The simulated PUF accuracies are shown when the noise level has a comparable SNR to the FPGA implemented PUFs

true for PUF-2 as well which displayed a higher level of vulnerability.

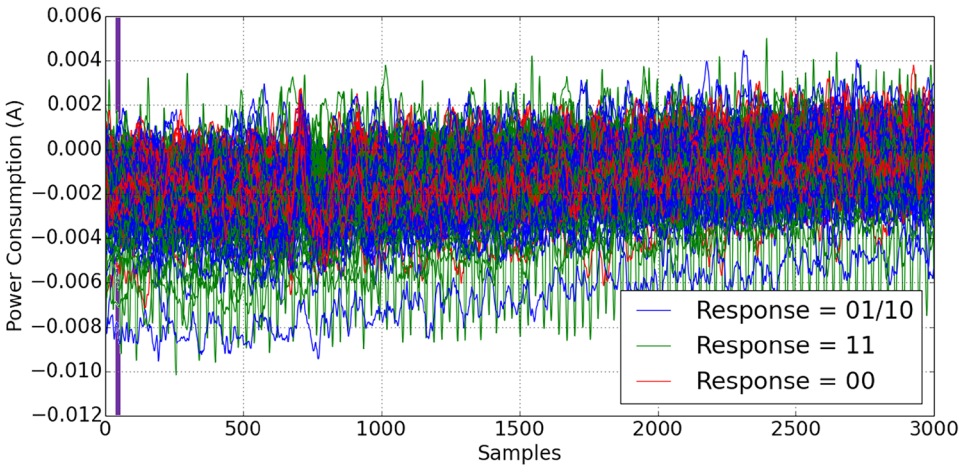
The takeaway from these results is that *Cross-PUF* attacks are a real concern in real-silicon implemented PUFs, and not just an artifact of simulation. Additionally, the results of the *Cross-PUF* attacks on the FPGA implemented PUFs support the simulated results when the SNR levels are comparable.

7.6 Results of Attacks on Multi-bit Parallel PUFs

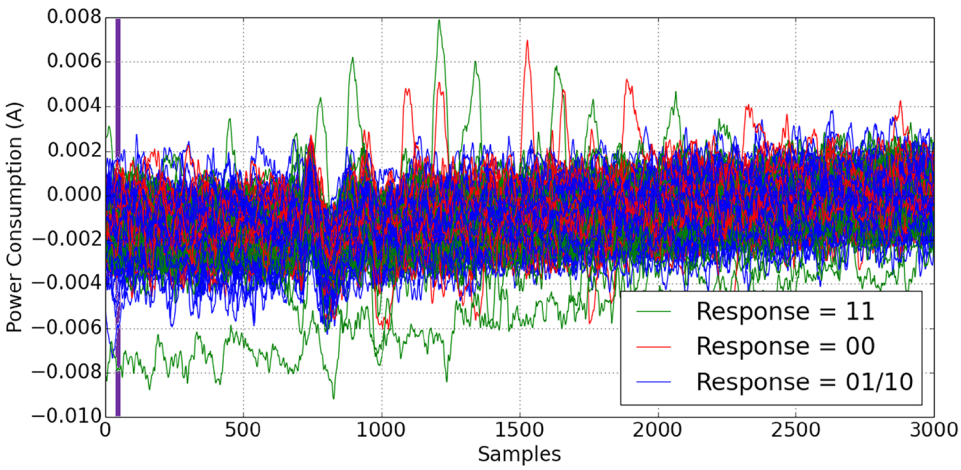
These sets of results investigate whether power-based modeling attacks are effective in Multi-bit Parallel PUFs where the designer has included more than one PUF instance in the circuit, thus generating more than 1-bit of response with each query. In the results presented here, two unprotected and two DRILL-protected 2-bit Parallel PUF instances were targeted. It is important to note that for 2-bit PUF instances, there are 4 values that occur in the responses. Thereby, the ideal accuracy of a protected PUF instance is 25% in this case.

Parallel PUF Power Traces Fig. 15 shows the power traces for a 2-bit response parallel PUF instance within an FPGA, zoomed into the period of time in which the Flip-Flop is active. As shown the traces for both unprotected PUF and the DRILL-protected PUF do not have a clear distinction related to the generated responses. Observing the DRILL-protected instance further it can be seen that all of the traces regardless of the response have the same behavior at ≈ 750 samples, since this phenomenon is present in all of the trace it does not reveal any adversarial information about the response.

Parallel PUF Attack Results The results of attacking the parallel PUFs are shown in Fig. 16. As depicted, the unprotected instances of PUF-0 and PUF-1 can be modeled with an accuracy of 83.56% and 91.84%, respectively. Therefore



(a) Unprotected PUF-0.



(b) Protected PUF-0.

Fig. 15 Superimposing 200 traces of unprotected and DRILL-protected PUF-0 implementations. The purple line is when the response Flip-Flops register the responses

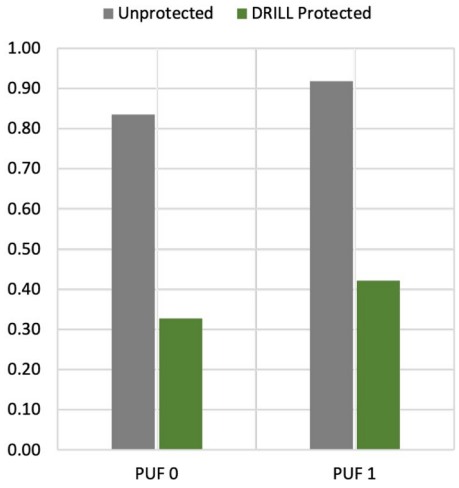


Fig. 16 Accuracy of attacking unprotected and DRILL-protected Multi-bit Parallel PUFs. 5000 traces were used for training

it can be said that parallel PUFs are still vulnerable to power-based modeling attacks. Turning the attention to the DRILL-protected PUF instances, the figure shows a significant drop in the accuracy of attack falling to 32.72% and 42.22% for PUF-0 and PUF-1, respectively.

Table 6 shows the accuracy of the attacks on the targeted PUFs when different responses were generated. The first observation from this table is that both unprotected PUF instances display a higher attack accuracy for the responses of 00 and 11. This is likely due to the fact that these responses will have the least leakage (for 00) and the most leakage (for 11) with makes their responses stand out from one another and from the 01 and 10 responses. This observation turns the attention to the DRILL-protected individual responses. As depicted, for both PUF-0 and PUF-1 DRILL-protected circuits no individual response will be predicted over another and therefore DRILL is effective in preventing the attack.

Table 6 Multi-bit Parallel PUF predictions of individual responses from implemented PUFs for when 5000 traces were used for training

PUF Inst.	Response	Percent Correct	
		Unprotected	Protected
PUF-0	00	0.9439	0.3349
	01	0.7039	0.3641
	10	0.6258	0.3255
	11	0.9672	0.29
PUF-1	00	0.9308	0.4826
	01	0.8825	0.3791
	10	0.9021	0.4201
	11	0.9464	0.4097

Table 7 Maximal SNR of Multi-bit parallel PUFs implemented within the FPGA

PUF Inst.		Unprotected	DRILL-Protected
PUF-0	Resp. Bit 0	0.175	0.021
	Resp. Bit 1	0.136	0.011
PUF-1	Resp. Bit 0	0.138	0.069
	Resp. Bit 1	0.138	0.003
Average		0.147	0.026

The SNR for each of the individual bits for each PUF is shown in Table 7. Investigating the SNR of the individual response bits is important to show the effectiveness of DRILL. As shown in this table, DRILL significantly lowers the SNR.

The takeaways from these observations for the Multi-bit Parallel PUF results are that these PUFs are vulnerable to modeling attacks, yet are highly resilient against such attacks when equipped with DRILL.

7.7 Results of Attacks on a Noisy Circuitry

In this section, we explore the attacks on PUFs implemented in intentionally noisy FPGA environments. This is to show the efficiency of the attacks on PUFs embedded in the circuitries with more logic where more operations are performed on the FPGA. To realize such a circuit, the FPGA implementation shown in Fig. 5 was expanded to include logics driven by the device's 48 MHz clock to access 18 external pins on the FPGA pseudo-randomly to induce a higher level of noise within the device. In this case, we consider a sophisticated attacker who may perform repeated queries to the PUF and subsequently average the result in order to increase the SNR. Table 8 shows the SNR of averaged traces and the resulting accuracy of attacking such noisy traces in our FPGA implementation. As depicted, the SNR increases when increasing the

Table 8 Maximal SNR of Attacks on the noisy FPGA implementation

PUF Type	**Number of Averaged Traces**	**SNR**	**Attack Accuracy**
Unprotected	1	0.244	87.52%
	2	0.349	90.22%
	4	0.478	92.86%
	8	0.593	95.97%
DRILL Protected	1	0.013	56.69%
	2	0.006	54.19%
	4	0.027	59.61%
	8	0.041	63.37%

number of traces, contributing in the averaging process going from 0.244 for a single trace to 0.593 for eight traces. The accuracy of the attack increases with the number of averaged traces to an accuracy of 95.97%. Whereas the maximal SNR for the DRILL protected PUFs is 0.041 and an accuracy of 63.37%.

The takeaway point of these results is that addition of independent noise is not a countermeasure to the Cross-PUF attack as the attack is still successful in our noisy environment. Hence a countermeasure like DRILL remains necessary even in presence of noisy environment. A secondary takeaway point is that the averaging increases the attack accuracy by increasing the SNR, thus reinforcing the necessity of an efficient countermeasure like DRILL.

8 Conclusion and Future Directions

This work investigated the arbiter-PUFs implemented on FPGA fabric to power-based modeling attacks. These investigations verified that the *Self-PUF* and *Cross-PUF* attacks, previously observed in simulation, are real threats in real silicon. We also showed that these power-based modeling attacks are a concern for multi-bit parallel PUF instances. The high accuracy of these attacks warranted mitigation. We showed that the Dual Rail/Random initialization logic-based countermeasure, so-called DRILL, despite being lightweight, is effective in mitigating the power-based modeling attacks by reducing the SNR of the observed power traces. In our study, we consider a situation where the PUF is running alone (no other IPs work in parallel), which is usually a safe decision to maximize the PUF reliability. We also made observations when various noise inducing circuits were added to the device and showed that an attack of this nature is still possible. In the future we will endeavor to perform the same investigations on ASIC implemented PUFs from the same GSDII file. We will also consider the impact of device aging on the Cross-PUF attacks and the proposed countermeasure on real-silicon. We will also investigate the possibility of cross-PUF attacks by exploiting electromagnetic emanations, in addition to power analyses.

Funding This work has partly benefited from the bilateral MESRI-BMBF project “APRIORI” from the ANR Cybersecurity 2020 call. It is also part of the Horizon 2020 “SPARTA” project under grant agreement number 830892. It was also benefited from National Science Foundation Award (grant number: 1920079).

Data Availability All data generated or analyzed during this study are within the paper.

Declarations

Competing Interests The authors declare that there are no competing interests.

Conflict of Interests The authors declare that they have no conflict of interest.

References

1. Aghaie A, Moradi A (2021) Inconsistency of Simulation and Practice in Delay-based Strong PUFs. Cryptology ePrint Archive, Report 2021/482. https://ia.cr/2021/482
2. Anusha G, Kumar A, Kandpal K (2020) A fully on-chip low-dropout regulator for SoC applications. Procedia Comput Sci 171:1009–1017
3. Bassham L et al (2010) A statistical test suite for random and pseudorandom number generators for cryptographic applications. URL https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=906762
4. Becker GT, Kumar R (2014) Active and passive side-channel attacks on delay based PUF designs. International Association for Cryptologic Research Cryptology Archive p 287
5. Becker GT (2015) The gap between promise and reality: On the insecurity of XOR arbiter PUFs. In: Proc. International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp 535–555
6. Bruneau N, Danger JL, Facon A, Guilley S, Hamaguchi S, Hori Y, Kang Y, Schaub A (2018) Development of the unified security requirements of PUFs during the standardization process. In: Proc. International Conference on Security for Information Technology and Communications. Springer, pp 314–330
7. Chatterjee D, Mukhopadhyay D, Hazra A (2020) Interpose PUF can be PAC Learned. International Association for Cryptologic Research Cryptology ePrint Archive p 471
8. Cherif Z, Danger J-L, Guilley S, Bossuet L (2012) An easy-to-design PUF based on a single oscillator: the loop PUF. In: Proc. 15th Euromicro Conference on Digital System Design, pp 156–162
9. Delvaux J, Verbauwhede I (2013) Side channel modeling attacks on 65 nm arbiter PUFs exploiting CMOS device noise. In: Proc. Int’l Symp. on Hardware-Oriented Security and Trust (HOST), pp 137–142
10. Ebrahimabadi M, Younis M, Karimi N (2022) A puf-based modeling-attack resilient authentication protocol for IoT devices. IEEE Internet Things J 9(5):3684–3703
11. Eastland N (2021) Structure of an FPGA. URL https://digilent.com/blog/structure-of-an-fpga/
12. Fukushima K et al (2016) Delay PUF assessment method based on side-channel and modeling analyzes: The final piece of all-in-one assessment methodology. In: IEEE Trustcom/BigDataSE/ISPA, pp 201–207
13. Gassend B, Clarke D, van Dijk M, Devadas S (2002) Controlled physical random functions. In: Proc. 18th Annual Computer Security Applications Conf., pp 149–160. https://doi.org/10.1109/CSAC.2002.1176287
14. Gassend B, Clarke D, van Dijk M, Devadas S (2002) Silicon physical random functions. In: Proc. ACM Conference on Computer and Communications Security, pp 148–160

15. Guntur H, Ishii J, Satoh A (2014) Side-channel attack user reference architecture board SAKURA-G. In: Proc. IEEE 3rd Global Conference on Consumer Electronics, GCCE. Tokyo, Japan, 7-10 October 2014, pp 271–274. https://doi.org/10.1109/GCCE.2014.7031104
16. Herder C, Yu M, Koushanfar F, Devadas S (2014) Physical unclonable functions and applications: A tutorial. Proc IEEE 102(8):1126–1141
17. ISO/IEC 20897-1 (2020) Information security, cybersecurity and privacy protection – Physically unclonable functions – Part 1: Security requirements. https://www.iso.org/standard/76353.html
18. Jiang Q, Zhang X, Zhang N, Tian Y, Ma X, Ma JQ (2019) Two-factor authentication protocol using physical unclonable function for IoV. In: Proc. IEEE/CIC International Conference on Communications in China (ICCC), pp 195–200
19. Karimi N, Danger J-L, Guilley S (2018) Impact of aging on the reliability of delay PUFs. J Electron Test: Theory Appl 34(5):571–586
20. Kim SJ (2021) Integrated and Distributed Digital Low-Drop-Out Regulators with Event-Driven Controls and Side-Channel Attack Resistance. Columbia University
21. Koushanfar F (2011) Integrated circuits metering for piracy protection and digital rights management: An overview. In: Proceedings of the 21st Edition of the Great Lakes Symposium on Great Lakes Symposium on VLSI, Association for Computing Machinery, New York, NY, USA, p 449–454. https://doi.org/10.1145/1973009.1973110
22. Kroeger T, Cheng W, Guilley S, Danger J-L, Karimi N (2020) Cross-PUF attacks on arbiter-PUFs through their power side-channel. In: Proc. IEEE International Test Conference (ITC), pp 1–5
23. Kroeger T, Cheng W, Guilley S, Danger J-L, Karimi N (2021) Enhancing the resiliency of multi-bit parallel arbiter-PUF and its derivatives against power attacks. In: Bhasin S, De Santis F (eds) Constructive side-channel analysis and secure design. COSADE 2021. Lecture notes in computer science, vol 12910, pp 303–321
24. Kroeger T, Cheng W, Guilley S, Danger JL, Karimi N (2021) Making obfuscated PUFs secure against power side-channel based modeling attacks. In: Proc. Design Automation and Test Europe (DATE)
25. Kroeger T, Cheng W, Guilley S, Danger J-L, Karimi N (2022) Assessment and mitigation of power side-channel-based cross-PUF attacks on arbiter-PUFs and their derivatives. IEEE Trans Very Large Scale Integr VLSI Syst 30(2):187–200. https://doi.org/10.1109/TVLSI.2021.3129141
26. Labrado C, Thapliyal H, Mohanty SP (2022) Fortifying Vehicular Security through Low Overhead Physically Unclonable Functions. ACM J Emerg Technol Comput Syst 18(1):8:1–8:18. https://doi.org/10.1145/3442443
27. Mahmoud A, Rührmair U, Majzoobi M, Koushanfar F (2013) Combined modeling and side channel attacks on strong PUFs. International Association for Cryptologic Research Cryptology ePrint Archive 2013:632
28. Mangard S, Oswald E, Popp T (2006) Power Analysis Attacks: Revealing the Secrets of Smart Cards. Springer
29. Maiti A, Gunreddy V, Schaumont P (2013) A Systematic Method to Evaluate and Compare the Performance of Physical Unclonable Functions. Springer, New York, New York, NY, pp 245–267
30. Majzoobi M, Koushanfar F, Devadas S (2010) FPGA PUF using programmable delay lines. In: Proc. IEEE Int'l Workshop on Information Forensics and Security, pp 1–6
31. Mars A, Adi W (2018) New Concept for Physically-Secured E-Coins Circulations. In: Adaptive Hardware and Systems, pp 333–338
32. Nagata M, Miki T, Miura N (2022) Physical attack protection techniques for ic chip level hardware security. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 30(1):5–14
33. Nangate 45 nm open cell library. "http://www.nangate.com"
34. Nguyen PH et al (2019) The Interpose PUF: Secure PUF Design against State-of-the-art Machine Learning Attacks. Trans on Cryptographic Hardware and Embedded Systems (CHES) p 243–290
35. Pappu R, Recht B, Taylor J, Gershenfeld N (2002) Physical one-way functions. Science 297(5589):2026–2030. https://doi.org/10.1126/science.1074376
36. Rührmair U et al (2010) Modeling Attacks on Physical Unclonable Functions. In: ACM Conference on Computer and Communications Security, pp 237–249
37. Rührmair U, Sölter J (2014) PUF modeling attacks: An introduction and overview. In: Proc. Design Automation and Test Europe (DATE), pp 1–6
38. Rührmair U, Xu X, Sölter J, Mahmoud A, Majzoobi M, Koushanfar F, Burleson W (2014) Efficient power and timing side channels for physical unclonable functions. In: Batina L, Robshaw M (eds) Cryptographic hardware and embedded systems – CHES 2014. Lecture notes in computer science, vol 8731. Springer, pp 476–492
39. Santikellur P, Bhattacharyay A, Chakraborty RS (2019) Deep learning based model building attacks on arbiter PUF compositions. Cryptology ePrint Archive
40. Soybali M, Ors B, Saldamli G (2011) Implementation of a PUF circuit on a FPGA. In: Proc. International Conference on New Technologies, Mobility and Security, pp 1–5
41. Spartan-6 FPGA Family. URL https://www.xilinx.com/products/silicon-devices/fpga/spartan-6.html
42. Tebelmann L, Danger JL, Pehl M (2020) Self-secured PUF: protecting the loop PUF by masking. Constructive Side-Channel Analysis and Secure Design (COSADE) 12244:293–314
43. Wisiol N, Mühl C, Pirnay N, Nguyen PH, Margraf M, Seifert J-P, van Dijk M, Rührmair U (2020) Splitting the interpose PUF: A novel modeling attack strategy. Trans on Cryptographic Hardware and Embedded Systems (CHES) 3:97–120
44. Yu W (2017) Exploiting on-chip voltage regulators as a countermeasure against power analysis attack. PhD thesis, University of South Florida. https://digitalcommons.usf.edu/cgi/viewcontent.cgi?article=8183&context=etd
45. Yu W, Uzun OA, Köse S (2015) Leveraging on-chip voltage regulators as a countermeasure against side-channel attacks. In: Proc. 52nd Annual Design Automation Conf., ACM, pp 1–6

Trevor Kroeger received his B.S. degree in Computer Engineering from the University of Denver (2011), and his M.S. Degree in Cyber Security at New York University (2015). In 2022 he obtained his PhD in Computer Engineering at the University of Maryland Baltimore County focusing on Hardware Security specifically the vulnerabilities of PUFs. He has worked in industry designing FPGAs for aerospace applications, and developed System on Chip architectures for distributed nodes in large scale system frameworks. He also performs communications propagation and clutter analysis, and data collection from unmanned systems.

Wei Cheng is now Postdoc Researcher at Secure-IC S.A.S., and also invited Associate Researcher at Télécom Paris. He obtained his Ph.D. degree in Information and Communications in December, 2021 from Télécom Paris & Institut Polytechnique de Paris, France. Before this,

he received the B.S. degree from Wuhan University in 2014 and the M.E. degree from the Institute of Information Engineering, CAS in 2017. His research interests include information theory, side-channel analysis, and related countermeasures (mainly on code-based masking, including inner product masking, direct sum masking, polynomial masking and other variants) of embedded systems and secure implementations. He also works on Machine Learning-based analysis on Physical Unclonable Functions (PUFs). He has received the ICE (Information, Communication and Electronics) PhD Prize of Institut Polytechnique de Paris, 2022.

Jean-Luc Danger is full Professor at Télécom Paris. He is the head of the digital electronic system research team involved in Research in security/safety of embedded systems, configurable architectures, and implementation of complex algorithms in ASICs or FPGAs. He authored more than 250 scientific publications and patents in architectures of embedded systems and security. He received his engineering degree in Electrical Engineering from École Supérieure d'Électricité in 1981. After 12 years in industrial laboratories (namely PHILIPS, NOKIA), he joined Télécom ParisTech in 1993 where he became full professor in 2002. He is a co-founder of Secure-IC. His personal research interests are trusted computing in embedded systems, random number generation, and protected implementations in novel technologies.

Sylvain Guilley is General Manager and Chief Technology Officer at Secure-IC, a company offering security for embedded systems. Secure-IC's flagship technology is the multi-certified SECURYZR® integrated Secure Element (iSE). Within Secure-IC, he is also director of "Threat Analysis" and "Think Ahead" business lines, which develop respectively security evaluation tools and advanced research. Sylvain is also professor at Télécom-Paris, associate research at École Normale Supérieure (ENS), and adjunct professor at the Chinese Academy of Sciences (CAS). His research interests are trusted computing, cyber-physical security, secure prototyping in FPGA and ASIC, and formal/mathematical methods. Since 2012, he organizes the PROOFS workshop, which brings together researchers whose objective is to increase the trust in the security of embedded systems. He is also lead editor of international standards, such as ISO/IEC 20897 (Physically Unclonable Functions), ISO/IEC 20085 (Calibration of non-invasive testing tools), ISO/IEC 24485 (White Box Cryptography), and ISO/IEC 17825 (detection of side-channel leakage). He is "High Level Principles for Design/Architecture" team leader for the drafting of Singapore TR68-3 standard on Cyber-Security of Autonomous Vehicles. Sylvain is associate editor of the Journal of Cryptography Engineering (JCEN, Springer). He has coauthored 250+ research papers and filed 40+ patents. He is member of the IACR and senior member of the IEEE and of the CryptArchi club. He is an alumni of Ecole Polytechnique and Télécom-ParisTech.

Naghmeh Karimi received the B.Sc., M.Sc., and Ph.D. degrees in Computer Engineering from the University of Tehran, Iran in 1997, 2002, and 2010, respectively. She was a visiting researcher at Yale University, USA between 2007 and 2009, and a post-doctoral researcher at Duke University, USA during 2011-2012. She has been a visiting assistant professor at New York University and Rutgers University between 2012 and 2016. She joined University of Maryland Baltimore County as an assistant professor in 2017 where she leads the SECure, REliable and Trusted Systems (SECRETS) research lab. She has published three book chapters and authored/co-authored more than 70 papers in referred conference proceedings and journal manuscripts. She serves as an Editor of the Springer Journal of Electronic Testing: Theory and Applications (JETTA) and IEEE Design & Test Journal. She has been the corresponding guest editor of the Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS); special issue in Hardware Security in Emerging Technologies in 2021. Her current research interests include hardware security, VLSI testing, design-for-trust, design-for-testability, and design-for-reliability. She is a recipient of the National Science Foundation CAREER Award in 2020.