IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

Received 11 January 2019; revised 9 August 2019; accepted 11 August 2019.
Date of publication 5 September 2019; date of current version 6 December 2021.

Digital Object Identifier 10.1109/TETC.2019.2935465

A Flip-Flop Based Arbiter Physical Unclonable Function (APUF) Design with High Entropy and Uniqueness for FPGA Implementation

CHONGYAN GU, (Member, IEEE), WEIQIANG LIU, (Senior Member, IEEE), YIJUN CUI, NEIL HANLEY, MÁIRE O'NEILL, (Senior Member, IEEE), AND FABRIZIO LOMBARDI, (Fellow, IEEE)

C. Gu, N. Hanley, and M. O'Neill are with the Centre for Secure Information Technologies, Institute of Electronics, Communications & Information Technology, Queen's University Belfast, Belfast BT3 9DT, United Kingdom
W. Liu and Y. Cui are with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing 211106, China
F. Lombardi is with the Department of ECE, Northeastern University, Boston MA02115, USA
CORRESPONDING AUTHOR: W. LIU (liuweiqiang@nuaa.edu.cn)

ABSTRACT A PUF is a physical security primitive that allows to extract intrinsic digital identifiers from electronic devices. It is a promising candidate to improve security in lightweight devices targeted at IoT applications due to its low cost nature. The Arbiter PUF or APUF has been widely studied in the technical literature. However it often suffers from disadvantages such as poor uniqueness and reliability, particularly when implemented on FPGAs due to physical layout restrictions. To address these problems, a new design known as FF-APUF has been proposed; it offers a compact architecture, combined with good uniqueness and reliability properties, and is well suited to FPGA implementation. Many PUF designs have been shown to be vulnerable to machine learning (ML) based modelling attacks. In this paper, initial tests show that to attack the FF-APUF design requires more effort for the adversary than a conventional APUF design. A comprehensive analysis of the experimental results for the FF-APUF design is presented to show this outcome. An improved APUF design with a balanced routing, and the proposed FF-APUF design are both implemented on an Xilinx Artix-7 FPGA at 28 nm technology. The empirical min-entropy of the FF-APUF design across different devices is shown to be more than twice that of the conventional APUF design.

INDEX TERMS FPGAs, PUFs, uniqueness, reliability, entropy

I. INTRODUCTION

With the increasing emergence of pervasive electronic devices, the Internet of things (IoT) has emerged as a new technology platform with great potential benefits leading to a projected 50 billion connected devices by 2020 [1]. The large amount of data generated by these devices and sensors requires the use of smart, autonomous machine to machine (M2M) communication. However, security and privacy, as well as newly enabled attacks using malicious or tampered devices such as the IoT based distributed denial-of-service (DDoS) attacks [2], pose substantial challenges. The computational capabilities of IoT devices are very diverse, from passively powered wearable health-care devices or long-life battery powered embedded sensors, to powerful edge servers such as might be found in autonomous vehicles. Additionally, the physical accessibility of the IoT device to an attacker will also vary greatly, leading to a multitude of attack surfaces. Hence, the global deployment of IoT devices could lead to an increasing threat to private and sensitive information. This has led to calls for cryptographic capability in IoT devices to protect user privacy and data security. However, conventional cryptographic approaches require complex computation which are not always suitable for IoT applications, due to the high power requirements and silcon area/memory overhead. A low-cost security approach is imperative to secure lightweight IoT devices.

A physical unclonable function (PUF) is a security primitive that utilizes manufacturing process variations to generate a unique digital fingerprint intrinsic to an electronic device, such as application-specific integrated circuit (ASICs) or field programmable gate array (FPGAs). While FPGAs were originally largely used for proof of concept or prototype

designs, due to their flexibility, increased logic density and faster time to market they are now increasingly being used for many applications as end-product technology. For examples, Intel FPGAs are already being used for emerging IoT applications such as Smart City infrastructure, Smart Grid and data center acceleration [3]. However, an efficient and lightweight security approach for low-cost FPGA-based IoT applications has not yet been fully addressed.

During semiconductor fabrication, manufacturing variations are reduced to ensure stable circuit operation. However it is not possible to entirely remove these variations and they can be utilised in PUF circuitry for security applications. Such a primitive has a number of desirable properties, such as the ability to provide low-cost authentication of an integrated circuit (IC) or a variability aware circuit that returns a specific response to an input challenge. Since no two PUFs are identical, the same $m-$bit input *(challenge)* string produces a different $N-$bit output *(response)* on different devices. They are inherently difficult to clone as individual manufacturing variations cannot be reproduced, which can also provide specific tamper-evident properties. These features provide advantages over current state-of-the-art technologies for a number of applications, such as lightweight secure authentication.

PUF architectures can be broadly categorized into Weak PUF (WPUF) and Strong PUF (SPUF) designs [4], based on the number of challenge response pair (CRPs) that capture information on the underlying variation.[1] Arbiter PUFs (APUFs) has been proposed by [5], [6] and are one of the most widely studied SPUFs architectures. However, conventional APUF designs suffer from poor uniqueness and repeatability properties. Moreover, they are difficult to implement on FPGA. To address the above limitations, a previous work [7] proposed a new robust FPGA-based strong flip flop based Arbiter PUF (FF-APUF) design. The experimental results on Xilinx Artix-7 FPGAs showed improvements in both uniqueness and reliability.

Additionally, APUFs have been shown to be vulnerable in particular to machine learning (ML) based modelling attacks. To prevent modelling attacks, many countermeasures have been proposed, such increasing the circuit complexity of PUF designs or protocol level protections. PUF-based authentication schemes tend to be resource inefficient for some lightweight applications as might be used for IoT devices. Moreover, it is shown in [8] that some modelling resistant methods [9], [10] can be vulnerable instead to protocol based attacks. Increasing the circuit complexity of PUF designs can be an effective, and sometimes low-cost, approach to address modelling attacks. However, these approaches reduce the robustness of PUF designs and have been shown to be less effective against modelling attacks than initially thought. In this paper, increased circuit complexity is used to enhance the security of the FF-APUF design; the results indicate a stronger modelling attack resistance ability than conventional APUF design.

[1]Weak and Strong here do *not* refer to the security strength of the circuit.

To be used as a security primitive, the PUF design must be unpredictable to an adversar, i.e., the responses to unseen challenges must not be predictable so as not to allow the PUF to be emulated in software. An analysis of entropy is commonly utilised to evaluate the unpredictability of a PUFs response. In this paper, an analysis of the empirical entropy is provided from the FPGA measurements. For a fair comparison, an improved conventional APUF design with a balanced arbiter is also implemented on the same FPGA testbed, with a comparison of the entropy analysis between both designs presented.

More specifically, the contributions and differences between this paper and the previous work [7] are summarized as follows:

- A conventional APUF design with a balanced routing is presented and implemented on Xilinx Artix-7 FPGAs. Both the previously proposed FF-APUF [7] and the improved APUF achieve better uniqueness results than the previous work [11] on FPGA.
- The two most widely used machine learning based modelling attacks, linear regression (LR) and covariance matrix adaptation evolution strategies (CMA-ES), are utilized to evaluate the resistance of the FF-APUF design. The results show that the FF-APUF is more difficult to attack than the APUF using the different modelling attack approaches.
- A comprehensive entropy evaluation, which includes conditional Shannon entropy, conditional min-entropy and min-entropy, is presented for both the FF-APUF and APUF designs. The results of the FF-APUF show better performance than the APUF design.

The rest of this paper is organized as follows. Section II reviews related previous works on SPUF designs and modelling attacks. Section III introduces the conventional APUF design and mathematical delay model. Section IV presents the circuit design, delay model and theoretical entropy analysis of the FF-APUF design. Implementation details and experimental results of both the improved conventional APUF and strong FF-APUF designs are given in Sections V and VI respectively, with modelling attack analysis of both designs also presented. Finally, a conclusion is drawn in Section VII.

II. REVIEW

Many different PUF architectures on both ASIC and FPGA implementations have been proposed, such as ring oscillator PUF (RO PUF) designs [12], [13], [14], memory-based PUF designs [4], [15]–[17], and APUF designs [5], [18], [19]. A comprehensive review of PUF designs can be found in [20].

The APUF is one of the most widely studied SPUF designs. It suffers from poor uniqueness and repeatability and is difficult to implement on an FPGA. Although previous designs proposed by Lee *et al.* [18] and Lim *et al.* [19] based on ASIC circuits improve upon these features, an implementation on FPGAs is still problematic. The routing of APUFs

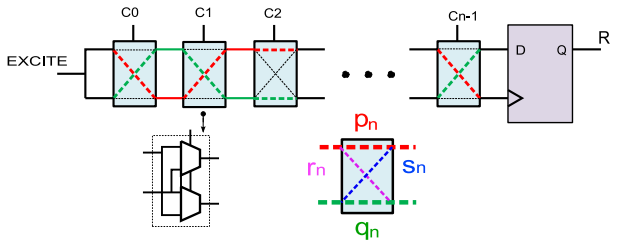


FIGURE 1. **The APUF design [5].**

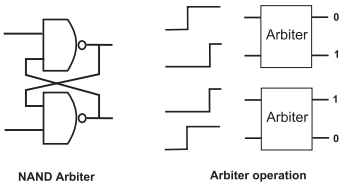


FIGURE 2. **The operation of a NAND gate arbiter.**

on FPGAs, unlike a manual place and route for ASICs, is restricted by the already fabricated circuit layout. Therefore, many previous designs are difficult to implement on FPGA due to the difficulty of implementation of balanced delay lines. Although Majzoobi *et al.* [21] and Hori *et al.* [11] have implemented APUFs on FPGAs, they introduced an additional tuning circuit or reported results with low uniqueness.

ML based modelling attacks, e.g., [22]–[25], have been reported to successfully attack a wide range of APUF designs using a software model to reveal the variability in PUF circuit. The response bits of APUF design can be individually attacked by constructing a separate linear additive delay model for each bit. To prevent modelling attacks, many countermeasures have been proposed such as PUF-based authentication protocols [9], [10], [26], which themselves have been shown to be vulnerable to protocol based attacks [8].

The XORed APUF proposed by Suh *et al.* [27], feed-forward APUF proposed by Lee *et al.* [18] and lightweight APUF proposed by Majzoobi *et al.* [28], increase the resistance of APUF against modelling attacks. However, different variants of the original attack have been proposed to break these designs when given sufficient CRPs as shown by Ruhrmair *et al.* [22]. To address this, a non-linear PUF circuit based on voltage transfer characteristics (VTC) [29], and a current mirrors based PUF [30], have been proposed to be specifically resistant to modelling attacks. However, these techniques have been simulated for ASICs and it has been shown that they are not suitable for FPGA implementations. Properties that designers should meet when designing ML resistant strong PUF designs are suggested in [31], however a practical instantiation remains unsolved. A multi-arbiter scheme is proposed in [32], based on the insertion of either a four-flip-flop or SR latch arbiter after each stage in the configurable paths, thus improving the uniqueness and reliability of the APUF design. However, the resistance of this approach to modelling attacks has not been reported.

III. CONVENTIONAL APUF DESIGN

Among previously published PUF designs, the APUF proposed by Gassend *et al.* [5] has been widely studied as a strong PUF; it consists of two parallel n-stage multiplexer chains that feed into an arbiter stage to form a 1-bit of an N-bit PUF as shown in Figure 1. Two Muxes are configured as either a cross- or straight-through connection based on the input challenge bit. Then an arbiter, such as an flip flop (FF), compares the arrival time of the two inputs to determine the response bit based on the first arrival, which should differ between devices due to manufacturing variability.

For the conventional APUF, an FF arbiter is employed to determine the faster delay path. It was shown that using an FF for the arbiter introduces a 10 percent skew in the routing path [19]. Hence, an arbiter consisting of cross coupled NAND gates is employed in this work as shown in Figure 2.

An APUF can be modeled using a linear additive model and can be derived by considering the delay difference in each stage. It has been shown that SPUFs made of linear circuits can be successfully attacked [22]–[25]. The additive delay model of the APUF circuit [19] can be described as follows. $\Delta(n)$, denotes the final delay difference between the two paths selected by the challenge, represented by Eq. (1)

$$\Delta(n) = \boldsymbol{P} \cdot \boldsymbol{\omega}^T, \tag{1}$$

where $\boldsymbol{P} = (p_0, p_1, \ldots, p_n)$ is a parity check vector incorporating challenge information, and $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_{n+1})$ is a constant vector of delay information. The delay differences at each stage are given as follows:

$$\begin{aligned} \omega_1 &= \alpha_1, \\ \omega_i &= \alpha_i + \beta_{(i-1)}, \text{for } 2 \leq i \leq n, \\ \omega_{n+1} &= \beta_n. \end{aligned} \tag{2}$$

The parity check, p_k, of the challenge bits, C_i, is defined in Eq. (3)

$$p_k = \prod_{i=k+1}^{n} 1 - 2 \cdot C_i. \tag{3}$$

In the constant vector, $\boldsymbol{\omega}$, α_n and β_n can be calculated using Eqs. (4) and (5), where p_n, q_n, r_n, s_n represent the delays of path routing, including the upper straight, upper cross, lower straight and lower cross paths shown in Figure 1

$$\alpha_n = \frac{p_n - q_n - r_n + s_n}{2} \tag{4}$$

$$\beta_n = \frac{p_n - q_n + r_n - s_n}{2}. \tag{5}$$

$\boldsymbol{\omega}$ includes information on the delay caused by manufacturing process variations in different APUF stages. The final delay difference, $\Delta(n)$, is the product of the challenge parity vector $\vec{\boldsymbol{P}}$ and delay difference vector $\boldsymbol{\omega}$. If $\Delta(n)$ is greater than 0, the bit response $\boldsymbol{r}$ is 1, otherwise it is 0.

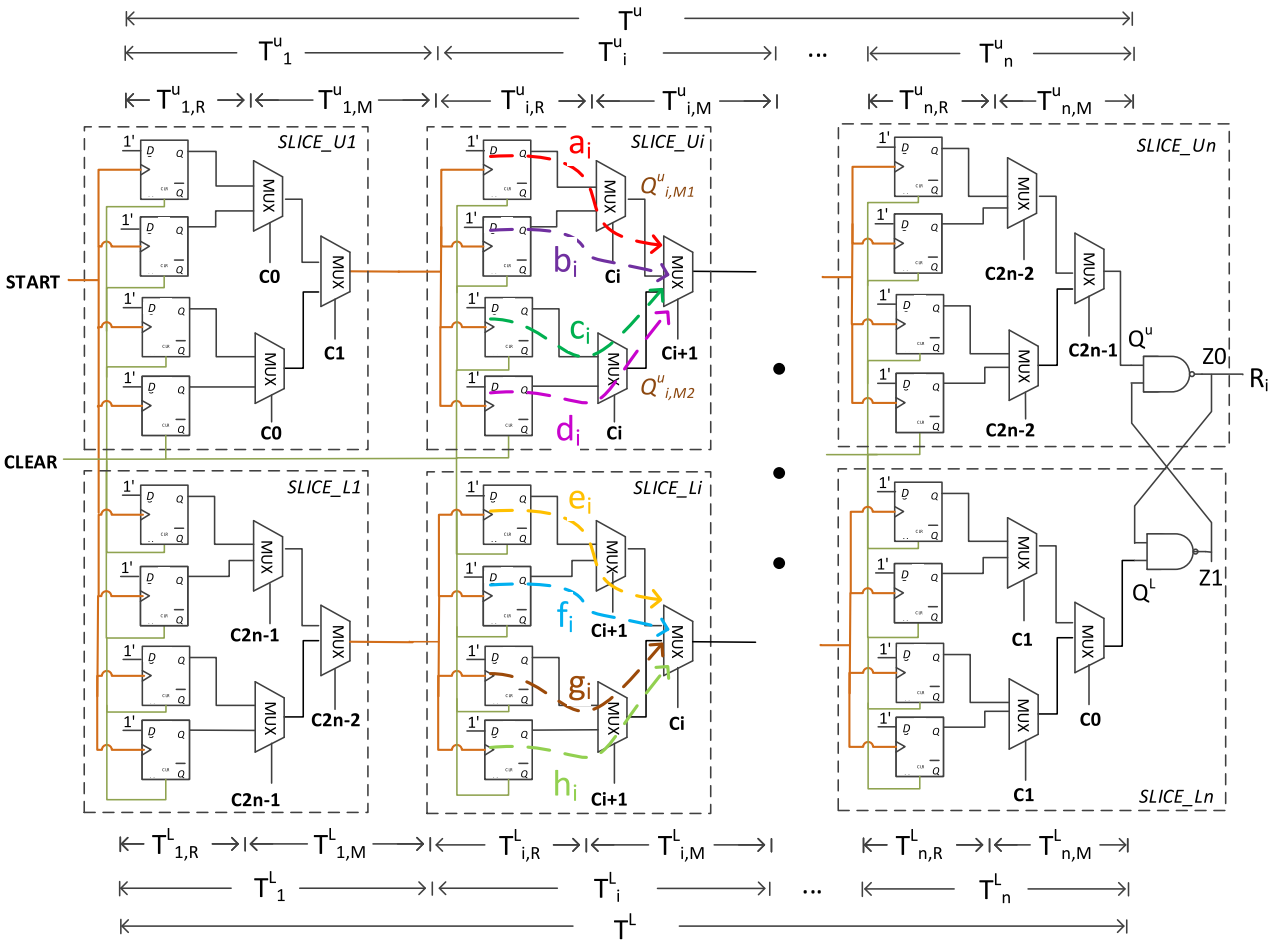


FIGURE 3. The FF-APUF design.

As a result of the linear architecture of SPUF, the response bits of a SPUF design can be attacked individually by building a separate linear additive delay model for each bit. If successful, this breaks the security of the SPUF as well as any protocol built on it.

IV. THE PROPOSED FF-APUF DESIGN

A. CIRCUIT DESIGN

A previously proposed FF-APUF circuit design [7] consists of an array of N asynchronous elementary 1-bit cells to generate an N-bit response. The design of a 1-bit response cell is shown in Figure 3, and consists of Muxes and FFs elements. To generate a single bit response, R_i, $3n$ multiplexer (Mux) gates and $4n$ FFs are cascaded in one of two paths to generate a delay path, where n is the number of stages and $m = 3n$ is the bit length of a challenge. To balance the routing to maximise variability, 3 Muxes are utilised for the FF-APUF design. Additional pre-processing steps can be applied to the challenges if required, such as differentiating the input challenges or applying an input network [28]. The FFs of *SLICE_U1* and *SLICE_L1* are first reset by *CLEAR* and then activated by the rising edge of the *START* signal (fed into the clock port). Three Mux gates in each slice are utilised to select one of the four FFs to form the delay path by the challenge bit, C_i. The result of each PUF cell is fed into the clock port of the next cell up to the last cell, *SLICE_Un* and *SLICE_Ln*. The last cell additionally contains cross-coupled NAND gates as an arbiter to determine the faster delay path (T^U or T^L) and returning an output of either one or zero. T^U and T^L denote the upper and lower delay paths used in the generation of each 1-bit response, respectively. To generate an N-bit PUF response, the design is replicated N times. Compared to the conventional APUF design, the proposed FF-APUF design is more flexible in path selection options.

The timing diagram of a 1-bit FF-APUF design is shown in Figure 4. The *CLEAR* signal is first activated to reset the

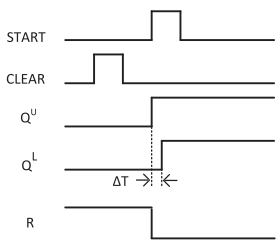


FIGURE 4. The timing diagram for the FF-APUF design.

circuit. When the rising edge of the *START* signal occurs, the delay paths are activated. The output signal, R_i, is 0 when Q^U and Q^L are 1-0 due to the faster arrival time of the delay path T^U (and vice versa for T^L).

Delay path comparison from different cells can be employed to reduce area utilisation. However, this leads to a dependency between different bits which requires additional consideration during the design stage and is not used in this work.

B. DELAY MODEL

The delay model of the FF-APUF design can be represented as follows:

$$\Delta T = T^U - T^L = \sum_{i=1}^{n} T_i^U + \sum_{i=1}^{n} T_i^L \underset{r=1}{\overset{r=0}{\gtrless}} 0. \tag{6}$$

T_i^U is derived as

$$\begin{aligned} T_i^U &= \frac{1-C_i}{2}\cdot Q_{i,M1}^U + \frac{1+C_i}{2}\cdot Q_{i,M2}^U \\ &= \frac{1-C_i}{2}\cdot\left(\frac{1-C_{i+1}}{2}\cdot a_i + \frac{1+C_{i+1}}{2}\cdot b_i\right) \\ &\quad + \frac{1+C_i}{2}\cdot\left(\frac{1-C_{i+1}}{2}\cdot c_i + \frac{1+C_{i+1}}{2}\cdot d_i\right) \\ &= \frac{1}{4}(1-C_i)(1-C_{i+1})a_i + \frac{1}{4}(1-C_i)(1+C_{i+1})b_i \\ &\quad + \frac{1}{4}(1+C_i)(1-C_{i+1})c_i + \frac{1}{4}(1+C_i)(1+C_{i+1})d_i, \end{aligned} \tag{7}$$

where, a_n, b_n, c_n, d_n are the delay segments from the top delay path, and e_n, f_n, g_n, h_n are the delay segments from the bottom delay path as shown in Figure 3.

T^U and T^L can be represented as

$$\begin{aligned} T^U &= \frac{1}{4}\cdot\{\vec{P_A}\cdot\vec{\phi_A}^T + \vec{P_B}\cdot\vec{\phi_B}^T \\ &\quad + \vec{P_C}\cdot\vec{\phi_C}^T + \vec{P_D}\cdot\vec{\phi_D}^T\} \\ T^L &= \frac{1}{4}\cdot\{\vec{P_E}\cdot\vec{\phi_E}^T + \vec{P_F}\cdot\vec{\phi_F}^T \\ &\quad + \vec{P_G}\cdot\vec{\phi_G}^T + \vec{P_H}\cdot\vec{\phi_H}^T\}, \end{aligned} \tag{8}$$

where, the challenge vector

$$\vec{P} = [\vec{P_A}, \vec{P_B}, \vec{P_C}, \vec{P_D}, \vec{P_E}, \vec{P_F}, \vec{P_G}, \vec{P_H}],$$

is calculated by Eq. (9), and the constant vector

$$\vec{\phi} = [\vec{\phi_A}, \vec{\phi_B}, \vec{\phi_C}, \vec{\phi_D}, \vec{\phi_E}, \vec{\phi_F}, \vec{\phi_G}, \vec{\phi_H}],$$

is represented as Eq. (10)

$$\begin{aligned} \vec{P_A} &= [(1+C_0)(1+C_1), (1+C_2)(1+C_3), \\ &\quad \ldots, (1+C_{2n-2})(1+C_{2n-1})] \\ \vec{P_B} &= [(1-C_0)(1+C_1), (1-C_2)(1+C_3), \\ &\quad \ldots, (1-C_{2n-2})(1+C_{2n-1})] \\ \vec{P_C} &= [(1+C_0)(1-C_1), (1+C_2)(1-C_3), \\ &\quad \ldots, (1+C_{2n-2})(1-C_{2n-1})] \\ \vec{P_D} &= [(1-C_0)(1-C_1), (1-C_2)(1-C_3), \\ &\quad \ldots, (1-C_{2n-2})(1-C_{2n-1})] \\ \vec{P_E} &= [(1+C_{2n-1})(1+C_{2n-2}), (1+C_{2n-3})(1+C_{2n-4}), \\ &\quad \ldots, (1+C_1)(1+C_0)] \\ \vec{P_F} &= [(1-C_{2n-1})(1+C_{2n-2}), (1-C_{2n-3})(1+C_{2n-4}), \\ &\quad \ldots, (1-C_1)(1+C_0)] \\ \vec{P_G} &= [(1+C_{2n-1})(1-C_{2n-2}), (1+C_{2n-3})(1-C_{2n-4}), \\ &\quad \ldots, (1+C_1)(1-C_0)] \\ \vec{P_H} &= [(1-C_{2n-1})(1-C_{2n-2}), (1-C_{2n-3})(1-C_{2n-4}), \\ &\quad \ldots, (1-C_1)(1-C_0)] \end{aligned} \tag{9}$$

$$\begin{aligned} &\phi_A = [a_0, a_1, \ldots, a_n],\ \phi_B = [b_0, b_1, \ldots, b_n] \\ &\phi_C = [c_0, c_1, \ldots, c_n],\ \phi_D = [d_0, d_1, \ldots, d_n] \\ &\phi_E = [e_0, e_1, \ldots, e_n],\ \phi_F = [f_0, f_1, \ldots, f_n] \\ &\phi_G = [g_0, g_1, \ldots, g_n],\ \phi_H = [h_0, h_1, \ldots, h_n]. \end{aligned} \tag{10}$$

C. COMPLEXITY ANALYSIS

The complexity of a PUF impacts the efficiency of modelling attacks by an adversary, i.e., the more complex the underlying PUF architecture, the more difficult the adversary to break a PUF. We utilize Shannon entropy, as given in Eq. (11), to assess the complexity of each stage. It allows us to compare the uncertainty introduced by process variations in each stage between the two designs

$$\begin{aligned} \mathrm{H}(\phi_f)_i &= -\sum_{j=1}^{m^2} p_j \cdot \log_2(p_j) \\ &= -\sum_{j=1}^{m^2} \frac{1}{m^2}\cdot\log_2\left(\frac{1}{m^2}\right) \\ &= \log_2(m^2) \\ &= 2\cdot\log_2(m), \end{aligned} \tag{11}$$

ϕ_f represents a 1-bit output of the FF-APUF design, and p_j is the probability of a given pair of FFs (i.e., delay) at each stage being selected by the challenge. As the challenge is assumed to be uniformly random, $p_j = (\frac{1}{m^2}) \quad \forall j$, for m FFs ($m = 4$ for the FF-APUF design) in the upper and lower cells of each stage, there are m^2 combinations in total, where m^2

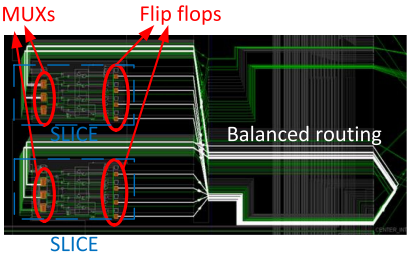


FIGURE 5. The Place & Route result of the FF-APUF design (one stage).

represents the number of combinations for the delay routes at each stage. Hence, the ideal Shannon entropy of the FF-APUF design at each stage is $\mathrm{H}(\phi_f)_i = 2 \cdot \log_2(4) = 4$. The relationship between the Shannon entropy provided by the path choices between the conventional APUF design and FF-APUF design at each stage in an ideal case is as follows:

$$\mathrm{H}(\phi_f)_i = 4 \cdot \mathrm{H}(\phi_a)_i. \tag{12}$$

Therefore, the Shannon entropy of the 1-bit circuit design with n-stages is

$$\mathrm{H}(\phi_f) = \prod_{i=1}^{n} \mathrm{H}(\phi_f)_i, \tag{13}$$

while the relationship between the conventional APUF and the FF-APUF is

$$\begin{aligned} \mathrm{H}(\phi_f) &= \prod_{i=1}^{n} 4 \cdot \mathrm{H}(\phi_a)_i \\ &= \left(4 \cdot \mathrm{H}(\phi_a)_i\right)^n \\ &= 4^n \cdot \mathrm{H}(\phi_a)_i. \end{aligned} \tag{14}$$

For the conventional APUF, the 2 Muxes at each stage provide 2 possible delay paths, so resulting in 2 combinations. According to Eq. (11), the Shannon entropy for each stage of the conventional APUF is equal to $\mathrm{H}(\phi_a)_i = \log_2(m^2) = \log_2(2) = 1$. Hence, Eq. (14) can be transformed as

$$\mathrm{H}(\phi_f) = 4^n \cdot \mathrm{H}(\phi_a)_i. \tag{15}$$

From the above analysis, it can be seen that the Shannon entropy provided by the available routing paths of the FF-APUF circuit design is 4^n times higher than the conventional APUF design, so a resource/complexity trade off can be explored depending on the target application.

V. FPGA IMPLEMENTATION

As previously mentioned, the conventional APUF design is non-trivial when implemented on FPGAs due to the routing restrictions, which can significantly bias the results. In this section, the FPGA implementations of both the FF-APUF and APUF designs are presented. An implementation of a

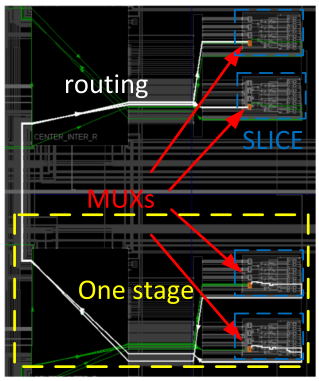


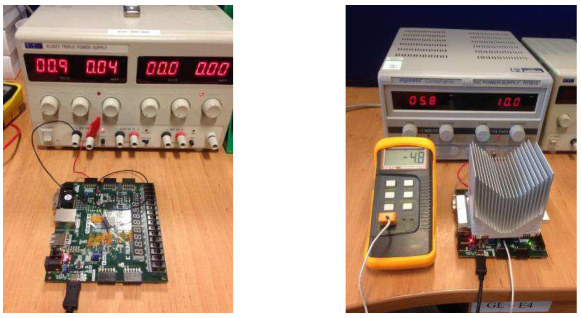
FIGURE 6. The Place & Route result of the improved conventional APUF design (one stage).

64-stage conventional APUF design on the Nexys4 board with a Xilinx Artix-7 FPGA has been proposed in [11]. In this implementation, each of the two delay chains requires 64 slices, with the arbiter (an FF) placed in an additional, separate slice. Hence, to generate a 1-bit response, 129 slices are used in total.

In this work, the improved conventional APUF uses a similar implementation strategy on the same Nexys4 board except cross coupled NAND gates are used for the arbiter instead of the FF because the FF can introduce a significant response skew as discussed in Section III. To generate a 1-bit response of the improved conventional APUF, 128 slices are used for the Muxes in two delay paths and the cross-coupled NAND gates.

The Xilinx Artix-7 XC7A100T has 15,850 logic slices, each consisting of four 6-input look up table (LUTs) and 4 FFs. The Nexys4 board has an oscillator running at 100 MHz to provide a clock input to the FPGA, and communications with the PC over UART run at 115,200 bps. A 64-bit FF-APUF design is implemented by constraining and balancing routing using TCL scripts as part of the Xilinx Vivado design flow. The generation of a 1-bit response for the FF-APUF design requires $64 \times 2 = 128$ slices to implement 4 FFs and 3 Muxes per slice, with the last two slices also implementing the extra cross-coupled NAND gates as an arbiter. Each slice of the Xilinx Artix-7 FPGA has 4 FFs to allow for the 4 FFs of the FF-APUF cell to be placed in a single slice. Note, in the previously published work [7], 22 stages (44 slices, 64-bit challenges) were employed. For fair comparison between the FF-APUF and the improved conventional APUF, 64 stages are utilized for both designs here.

Figure 5 shows a balanced routing map of one stage of the FF-APUF design, as implemented on two slices (one CLB). The balanced routing of each slice ensures a robust response result. Figure 6 shows the placement and routing of the improved conventional APUF design at each stage on the same Xilinx Artix-7 FPGA, as also implemented on two slices. The Mux and cross-coupled NAND gates are implemented by LUTs. At each stage of the APUF design, the upper/lower delay segment, the Mux, requires one LUT of a slice as implementation. To balance the



(a) (b)

FIGURE 7. Experimental setup.

routing, the Mux has to be implemented separately in the slices, so other resources of that slice are unused. Normally, it is suggested to reserve these slice resources for PUF designs in order to reduce the impact of cross-coupling from other, non-PUF circuitry. At each stage of the FF-APUF design, the upper/lower delay segment can be compactly implemented in one slice, involving 4 FFs and 3 LUTs. Hence, both the FF-APUF and APUF designs require the same slice resources.

VI. EXPERIMENTAL RESULTS

The 64-bit FF-APUF design is implemented in 11 low-cost Digilent Nexys4 boards, each containing a Xilinx Artix-7 XC7A100T FPGA (28 nm technology).

To increase the number of test devices and so obtain a more granular set of results, two different floor plan strategies on each FPGA are employed as proposed in [33], in order to emulate 22 devices in the experiment. One board is modified to conduct voltage experiments to allow for the variation of the core voltage (1.0v) $\pm 10\%$ as shown in Figure 7(a). Additionally, the experiments are run for one board while varying the ambient temperature in a range from 0^oC to 75^oC using a thermometric plate platform shown in Figure 7(b).

A. UNIQUENESS

The uniqueness metric measures the inter-chip variation by evaluating the differentiation of a particular PUF circuit design among k different designs. Ideally, when the same PUF circuit is implemented on multiple devices it should produce an average inter-chip Hamming distance (HD) of approximately 50 percent when comparing the responses between the two devices for the same challenge; e.g., half the response bits are expected to differ between any two devices even though the same challenge is provided as input to the same circuit. A percentage-based figure-of-merit for uniqueness based on average inter-chip HD can be defined. If two PUFs, Φ_i and Φ_j, implement the same PUF circuit and have N-bit responses R_i and R_j to the same challenge, C, then the uniqueness is given by the average inter-chip HD among the k devices and is defined as

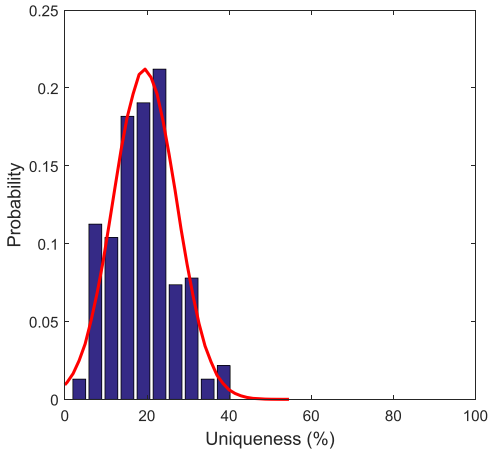


FIGURE 8. The uniqueness result for the improved APUF design.

$$\text{Uniqueness} = \frac{2}{k(k-1)} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} \frac{\text{HD}(R_i, R_j)}{n} \times 100. \tag{16}$$

Eq. (16) is an average of all possible pair-wise average HDs among k devices, and gives an estimate of the expected inter-chip variation in terms of PUF responses for the same challenge. The uniqueness experiment is carried out over normal operating conditions, in which the room temperature is 20^oC and the core supply voltage of a Xilinx Artix-7 FPGA is 1.0 V. The HD values are computed from the 64-bit $(n = 64)$ responses that are collected from $22(k = 22)$ devices.

Figure 8 shows a histogram of the uniqueness results for the improved conventional APUF design, which has an empirical mean of 19.46 percent and a standard deviation (STD) of 7.73 percent, an improvement over previously reported results of 9.42 percent in [11]. The improved results are partially caused by the utilization of a D-latch for the arbiter in the implementation in [11]; this can cause a significant bias in the response as discussed in [19]. In this work, two cross coupled NAND gates are employed to reduce the skew while keeping the routing balance to improve the randomness source of the response. However, it is difficult to further improve this metric due to the limited process variations from the Mux of the APUF design. Compared to the APUF design, the FFs of the FF-APUF design contribute to a significant variability between the paths relative to the Muxes; this is because the FF is a coarse-grained component. Figure 9 shows the uniqueness result of the FF-APUF design, which is 41.53 percent with a STD of 7.52 percent. Compared to the results of 9.42 percent [11] and 19.46 percent of the improved conventional APUF in this work, the FF-APUF design achieves a significant improvement in the capability to distinguish between a population of different (identical) devices.

Many improvements have been developed for APUF to achieve better uniqueness. However, most of them were proposed for ASICs, [30], [34], [35]. It has proven difficult to

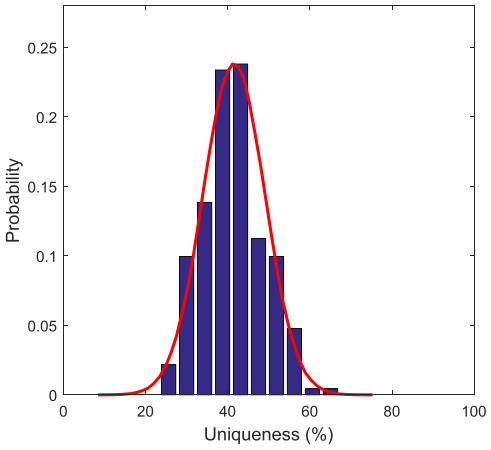


FIGURE 9. **The uniqueness result for the FF-APUF design.**

achieve a high uniqueness for APUF on FPGAs due to unbalanced routing [11], [21]. To counter this, efforts on FPGA based APUF designs have also been proposed, such as those proposed in [21], [35]. However, they are based on programmable delay logic (PDL) circuits which have to utilise a tuning circuit to improve uniqueness. In this paper, the uniqueness of APUF design has been improved from 9 percent to approximate 20 percent using a balanced routing strategy without any extra hardware resource consumption, with the proposed FF-APUF design achieving a uniqueness of 42 percent.

B. RELIABILITY

Ideally, a given PUF design implemented on a device should be able to perfectly reproduce its output whenever it is queried with a challenge. However, environmental changes, such as temperature and power supply voltage variations, as well as the metastability in PUF circuits, induce noise in the responses. Therefore, a reliability metric is utilised to quantify the ability of a PUF design to reproduce a response. For a device Φ_i, reliability is represented as a single value by finding the average intra-chip HD of s response samples, $R_i^{'}$; this is taken at different operating conditions compared to a baseline $N-$bit reference response, R_i, taken at nominal operating conditions. The average intra-chip HD is defined as follows:

$$\mathsf{HD}_{\mathsf{INTRA}} = \frac{1}{s}\sum_{t=1}^{s}\frac{\mathsf{HD}(R_i, R_{i,t}^{'})}{N} \times 100, \tag{17}$$

where $R(i,t)^{'}$ is the tth sample of $R_i^{'}$. The reliability can be represented as

$$\mathsf{Reliability} = 100 - \mathsf{HD}_{\mathsf{INTRA}}. \tag{18}$$

The ideal value for reliability is as close to 100 percent as possible; subsequent circuit components such as error correction also benefit from a high reliability measure.

For the reliability test, 1,000 responses per temperature value at the same challenge are collected and averaged as the final response. The response of the nominal temperature 20^oC is utilized as a flag for comparison with the responses from different

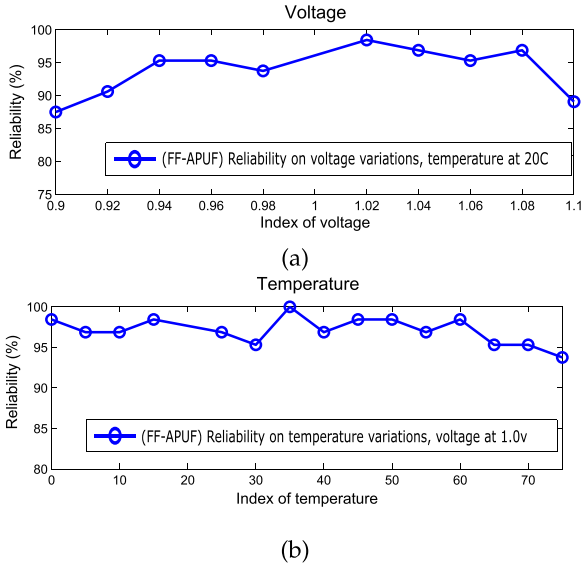


FIGURE 10. **The reliability result of the FF-APUF design: (a) Under voltage variations and (b) under temperature variations.**

temperatures. Figure 10 gives the reliability results of the FF-APUF design, which averages to 97.10 percent over the environmental temperature range under test of 0^oC to 75^oC, and 93.90 percent over a $\pm 10\%$ variation in supply voltage. Hence, the FF-APUF design has a good reliability result across a range of conditions. Moreover, the voltage variation has a greater effect on reliability than temperature for the FF-APUF design. Hence, this test is also carried out for the improved conventional APUF design; the reliability results are shown in Figure 11. The average reliability result of the improved conventional APUF design is 97.03 percent over a $\pm 10\%$ variation in supply voltage. This is higher than the reliability of 93.30 percent for the FF-APUF design, which is likely due to the low uniqueness of the improved conventional APUF design. As there is a significant bias in the bits of the response, these bits are not as heavily influenced by the variations in voltage/temperature as the bits in the FF-APUF design.

Table 1 compares the performance of the FF-APUF design, improved conventional PUF design, and previously proposed PUF designs on both FPGA and ASIC platforms. While direct comparison between various designs is not straightforward, particularly between FPGA and ASIC platforms, it gives an overview of the expected uniqueness and

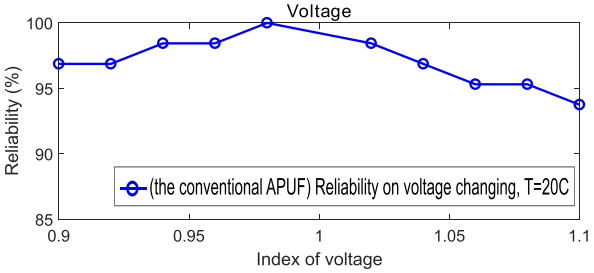


FIGURE 11. **The reliability result of the improved conventional APUF design.**

TABLE 1. A comparison of hardware resource consumption and metrics of different PUF designs

PUF design	U^acute; (ideal 50%)	R^acute; (ideal 100%)	Technologies	Response	Consumption
SRAM PUF [4]	49.97%	$> 88\%^{t}$	FPGA	128	4600 SRAM memory bits
Latch PUF [36]	50.55%	96.96%	0.13um CMOS	128	1 latch for each ID cell
Latch PUF [37]	46%	$> 87\%^{t}$	Spartan 3	128	2×128 slices
Flip-flop PUF [38]	$\approx 50\%^{*}$	$> 95\%^{v}$	Virtex 2	4096	4096 flip flops
Flip-flop PUF [39]	36%	$> 87\%^{t}$	ASIC	1024	1024 flip flops
Buskeeper PUF [33]	49%	$> 80\%^{t}$, $> 95\%^{v}$	TSMC 65 nm	192	$1GE$[a]
Butterfly PUF [40]	$\approx 50\%^{*}$	94%	Virtex 5	64	130 slices
Compact PUF [15]	48.52%	93%	Spartan 6	128	128 slices
Ultra-compact PUF ID generator [16]	49.93%	93.96%	Spartan 6	128	40 slices
ring oscillator (RO) PUF [27]	46.15%	99.52%	Virtex 4	128	16×64 *array*[b]
configurable ring oscillator (CRO) PUF [12]	43.50%	$> 96\%^{t}$, $\approx 100\%^{*}$	Spartan 3	127	64 slices for ROs except counters
BR PUF [41]	14.80%	99.20%	SPICE simulation	64 stages	64×COMB[c]
APUF [18], [19]	23%	$95.20\%^{t}$, $96.30\%^{v}$	TSMC 180 nm	64	$1212um \times 1212um$
Subthreshold APUF [34]	$\approx 50\%$	$98.10\%^{v}$	45 nm	64	$36um \times 50um$
Feedforward APUF [19]	38%	90.20%	TSMC 180 nm	-	-
APUF [42]	4.70%	-	Virtex5	5000	a standard APUF
2-1 Double APUF [42]	46.40%	-	Virtex5	5000	double size of an APUF
3-1 Double APUF [42]	50.20%	-	Virtex5	5000	triple size of an APUF
APUF [35]	45.25%	96%	Spartan3	64	PDLs with tuning circuits
APUF [43]	36.75% (ideal 100%)	98.28%	Virtex5	64	129 slices
APUF [44]	7.20%	99.76%	Virtex5	128	-
APUF [11]	9.42%	-	Artix7, Spartan6, Kintex	64	129 slices
Improved APUF	19.46%	$97.03\%^{v}$	28 nm Artix7	64	128 slices
FF-APUF	41.53%	$97.10\%^{t}$, $93.90\%^{v}$	28 nm Artix7	64	128 Slices

[a] *GE represented gate equivalent.*
[b] 16×64*array = 1024 ROs; each RO consisting of 5 inverters and 1 AND.*
[c] *COMB = 2NOR + 1MUX + 1DEMUX.*
[t] *under temperature variation.* [v] *under supply voltage variation.* [*] *required post-processing.*

reliability results. As previously mentioned, it is difficult for the conventional APUF design to achieve a better uniqueness on FPGAs due to the unbalanced routing. The uniqueness of [43] is 36.75 percent in a different evaluation method, in which the ideal uniqueness value is 100 percent other than 50 percent, that is commonly utilized. The uniqueness results for the conventional APUF implemented on FPGA are 4.70 percent by [42], 7.20 percent by [44] and 9.42 percent by [11], respectively. The 2-1 double APUF and 3-1 double APUF, which mix two or three APUFs to generate one 1-bit response, achieve higher uniqueness [42] but have to sacrifice reliability. [35] achieves higher uniqueness results when introducing tuning circuits. The FF-APUF design exhibits a better uniqueness result compared to the conventional APUF designs in both FPGA and ASIC; moreover, it achieves good reliability results on FPGA where the FF-APUF design achieves comparable reliability results as the conventional PUF design on an ASIC. In terms of hardware resource consumption, the FF-APUF also achieves efficient resource usage, and occupies the same number of slices on FPGA compared to the improved conventional APUF.

C. ENTROPY

A Strong PUF is expected to be unpredictable such that an adversary cannot efficiently predict the output to an unknown challenge given an observed number of CRPs. To verify this uncertainty, entropy is commonly utilised to quantify the level of unpredictability of a PUF design. In this section, the entropy from the responses of both designs is assessed. A number of works suggest assessment measures of the empirical entropy as generated by the PUF responses, e.g., the context-tree weighting (CTW) method [45], as well as standardized randomness tests such as the Diehard test and the NIST test suite [46]. However, Maes *et al.* [47] have pointed out that both these methods only offer a low level of confidence on the outcomes due to the limited bit length of the available responses. The conditional Shannon entropy and min-entropy have been introduced by Stefan *et al.* [48]; they are considered more precise than the previous methods as they consider the dependency between individual bits of the PUF responses. Hence, these two methods are utilised in this work. Although these methods provide a more accurate assessment for a single PUF response, the cross-relationship between responses from different devices cannot be directly assessed. Therefore, a min-entropy method following the NIST specification 800-90 [49] for binary sources is widely used [50].

1) CONDITIONAL ENTROPY

Given a challenge $c \in C$, the adversary tries to predict the response $r \in R$. $R(c)$ represents the response of challenge c, and

```
Challenge(0, :) :   0 1 0 1 0 ... 0 1 0 0 1 1
Challenge(1, :) :   1 1 0 1 0 ... 0 1 0 0 1 1
Challenge(2, :) :   0 0 0 1 0 ... 0 1 0 0 1 1
Challenge(3, :) :   0 1 1 1 0 ... 0 1 0 0 1 1
Challenge(4, :) :   0 1 0 0 0 ... 0 1 0 0 1 1
```

FIGURE 12. An example of the challenge generation by Algorithm 1.

$W(c)$ is for the set of all responses of the PUF except r. The conditional entropy can then be computed as follows [48]:

$$\mathrm{H}(R|W) = -\sum_{c\in C} \Pr[Y(c), W(c)] \cdot log_2(\Pr[R(c)|W(c)]). \tag{19}$$

The conditional min-entropy can be calculated as

$$\mathsf{H}_{\mathsf{min}}(R|W) = -log_2(\mathsf{max}(\Pr[R(c)|W(c)])). \tag{20}$$

In this work, for both the improved conventional APUF and FF-APUF designs, the delay stages are controlled by challenges. Each delay segment contributes to the final output value. The adversary's aim is to guess the output with a probability of success greater than 0.5 after collecting a sufficient number of CRPs. The conditional entropy and conditional min-entropy quantify the average and worst-case information from the PUF design that the adversary cannot predict. However, the response space of the data set, 2^{64} for the 64 stages APUF design under test, is too large to directly calculate the conditional entropy as per Eqs. (19) and (20). Hence, a subset including the 'worst case' for the responses is chosen to reduce the size of $\mathrm{W}(c)$.

Algorithm 1. Subset Challenge Generation Algorithm

```
procedure SUBSET–GENERATION
    challenge[0][:] = randi((0, 1), n)
    % n is the response bit number
    for i from 1 to n do
        challenge_tmp = challenge[0][:]
        challenge_tmp[i] = (∼ challenge[0][i])
        challenge[i][0] = challenge_tmp
    end for
end procedure
```

The operation of the collected subset CRPs in this experiment is outlined in Algorithm 1. A challenge subset is selected by a random challenge having the relationship of HD$\leqslant$1, $\chi = 64$ and $\chi = 64 \times 3 = 192$ for the improved conventional APUF and FF-APUF designs respectively. At first, a random $n-$bit challenge $challenge[0][:]$ is chosen. Then a data set of n, $n-$bit challenges (from $challenge[1][:]$ to $challenge[n][:]$), having HD$\leqslant$1 to the challenge $challenge[0][:]$, is chosen as a challenge subset. Figure 12 shows an example of the challenge generation by Algorithm 1.

The conditional Shannon entropy and the conditional min-entropy results of both the improved conventional APUF design and FF-APUF design are shown in Table 2. The FF-APUF design has both higher conditional Shannon entropy and conditional min-entropy than the improved conventional APUF design.

TABLE 2. A comparison of the entropy analysis between the improved conventional APUF and FF-APUF designs

Metrics	Improved APUF	FF-APUF
Max probability[a]	0.04	0.10
Conditional Shannon entropy[a]	0.88	0.90
Conditional min-entropy[a]	0.60	0.61
Max probability[b]	1	0.77
Min-entropy[b]	0.23	0.54

[a] *Samples are generated from one device.*
[b] *Samples are generated over 22 devices.*

2) MIN-ENTROPY

The min-entropy measures the 'worst case' scenario for the unpredictability of random data. The method for binary sources as outlined in NIST specification 800-90 [49] is utilised to evaluate the min-entropy of both the PUF designs. The N-bit responses of k devices have an occurrence probability for each bit. p_0 and p_1 represent the values of '0' and '1', respectively. Given $p_{i,\mathsf{max}} = \mathsf{max}(p_{i,0}, p_{i,1})$, the formula to calculate the min-entropy is as follows:

$$\mathrm{H}_{\mathsf{min}} = -\frac{1}{N}\sum_{i=1}^{N} log_2(p_{i,\mathsf{max}}). \tag{21}$$

For the improved conventional APUF design and the FF-APUF design, the Hamming weight (HW) of a single position on 22 devices ($k = 22$) of a 64-bit response is computed. The $\mathrm{p}_{\mathsf{max}}$ is then derived from Eq. (22). Table 2 gives the min-entropy results of both the improved conventional APUF and FF-APUF designs, 0.23 and 0.54, respectively. The FF-APUF design has a higher unpredictability lower bound than the improved conventional APUF design, proving the higher inherent uniqueness of the FF-APUF design

$$\begin{aligned} \mathsf{HW}_i > k/2 &\Rightarrow p_{i\ max} = \mathsf{HW}_i/k \\ \mathsf{HW}_i \leq k/2 &\Rightarrow p_{i\ max} = (k - \mathsf{HW}_i)/k. \end{aligned} \tag{22}$$

Figure 13 shows how the per-bit min-entropy of the improved conventional APUF design and the FF-APUF design over an increasing number of devices k. The accuracy of the min-entropy estimation increases with the number of devices.

D. MODELLING ATTACK

To evaluate the modelling attack resistance of the proposed FF-APUF, ML algorithms based modelling attacks have been applied. The modelling attacks utilised in this paper are setup following the delay model introduced in Section IV-B. LR is one of the most efficient modelling attacks to break APUF design [22], [51]. In this work, LR attack is executed to evaluate the FF-APUF design.

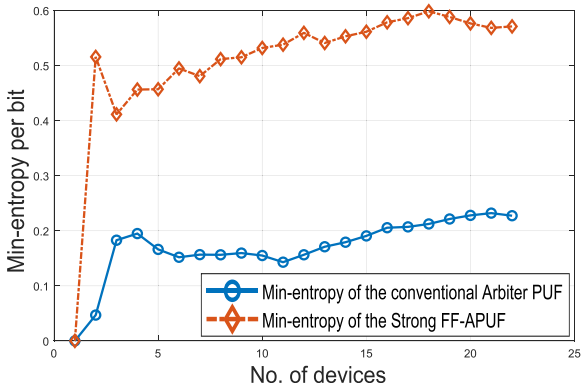


FIGURE 13. The min-entropy results of both the improved conventional APUF and FF-APUF designs.

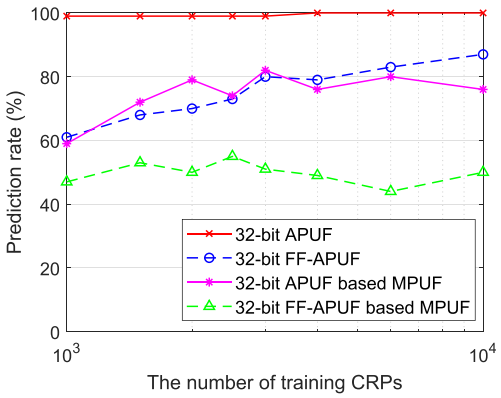


FIGURE 15. The CMA-ES attack on both the APUF and FF-APUF designs as well as two MPUF designs based on both.

The works [22], [51] are utilised to set up an FF-APUF design and an improved APUF design through simulation based on the two previously discussed delay models, with the delay model of the APUF shown in Section III and the delay model of the FF-APUF shown in Section IV-B. For LR attacks, the approach of [51] is implemented to build an adversarial model to test the security performance of the FF-APUF design.

Figure 14 presents the LR based modelling attack results for both the APUF and FF-APUF designs. The solid lines show the prediction rates of the LR attack for the APUF design with responses of 64-bit and 256-bit, respectively, while the dashed lines show the results of the same configuration for the FF-APUF design. Compared to the APUF design, the FF-APUF design is more difficult to predict with the same numbers of training data. The computational attack time when using LR for the APUF design is less than 1 second with responses sizes of 64-bit and 256-bit, while the results of the same configuration for the FF-APUF design are approximately 34 and 79 seconds respectively. Hence only data collection time needs to be considered for this attack. For the 256-bit FF-APUF design, the adversary requires approximately 80 times greater training set size for an equivalent success rate compared to the conventional APUF design. Hence, the FF-APUF design demonstrates a significantly better modelling attack resistance.

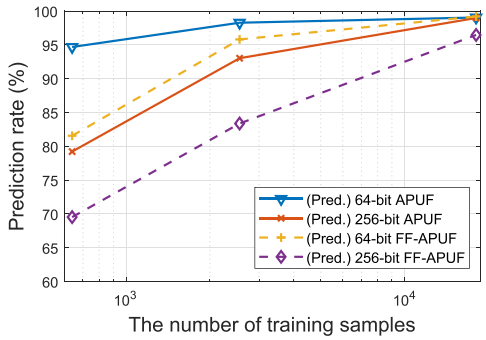


FIGURE 14. The LR attack on both the APUF and FF-APUF designs.

To improve the modelling attack resistance, various methods can be employed. For example, an efficient approach is to increase the number of FFs (m) at each stage. Some previously published approaches for the APUF can also be applied for the proposed FF-APUF, such as the XORed [27], feed-forward [18] and lightweight APUF extension [28]. However, these approaches have been attacked successfully [22], [25]. Recently, Ma *et al.* [52] proposed a Multi-PUF (MPUF) design that utilises a Weak PUF to obfuscate the challenges to a Strong PUF and have shown that it is harder to model than both the conventional APUF and XORed APUF designs using ML attacks. The MPUF strategy can also be utilised to the FF-APUF design to enhance its modelling attack resistance. CMA-ES is another efficient modelling attack method, widely utilised for APUF attacks. It has been shown that the reliability based CMA-ES attack outperformed the LR attack for XORed APUF designs. To present the results of a comprehensive modelling attack resistance for the proposed FF-APUF design, the CMA-ES method is also applied here. Figure 15 presents the CMA-ES attack on both the APUF and FF-APUF designs as well as two MPUF designs based on both. The proposed FF-APUF design again has lower prediction rates than the conventional APUF design. The FF-APUF based MPUF design also achieves better modelling attack resistance than the APUF based MPUF design.

VII. CONCLUSION

In this paper, a theoretical circuit complexity analysis, a modelling attack resistance analysis and an experimental entropy analysis of both previously proposed FPGA-based Strong FF-APUF and improved conventional APUF designs have been presented. The experimental evaluation of the improved conventional APUF design shows the uniqueness and reliability of 19.46 and 97.03 percent respectively, higher than that of the work [11]. The experimental min-entropy of the FF-APUF design, 0.54, is more than twice than 0.23 of the improved conventional APUF design. The conditional Shannon entropy and the conditional min-entropy of the FF-APUF design, 0.90 and 0.61, are also higher than 0.88 and

0.60 of the improved conventional APUF design. The performance evaluation results for the FF-APUF design show uniqueness and reliability results of 41.53 percent and (93.30 percent(voltage), 97.10 percent(temperature)), respectively. The FF-APUF is also shown to be more resistant to the modelling attacks, including LR and CMA-ES, than the conventional APUF design. The compact placement of the circuit also ensures that the design incurs in least hardware reported in the literature to date. The FPGA-based Strong FF-APUF design significantly improves upon the previous APUF designs, and has potential to be the basis for CRP-based authentication applications in the IoT.

ACKNOWLEDGMENTS

This work was partly supported by the Institute for Information & communications Technology Promotion(IITP) grant funded by the Korean government(MSIT) (No. 2016-0-00399, Study on secure key hiding technology for IoT devices [KeyHAS Project]), by the Engineering and Physical Sciences Research Council (EPSRC) (EP/N508664/-CSIT2), by the Fundamental Research Funds for the Central Universities China (NE2019102), National Natural Science Foundation China (61771239 and 61871216) and the Six Talent Peaks Project in Jiangsu Province (2018-XYDXX-009).

REFERENCES

[1] S. M. Ali. (2019). Industrial edge networking components report. [Online]. Available: https://www:cisco:com/c/dam/en_us/solutions/iot/ihs-report:pdf?oid=wprit017016
[2] KrebsonSecurity. (2016). DDoS on dyn impacts twitter, spotify, reddit. [Online]. Available: https://krebsonsecurity:com/2016/10/ddos-on-dyn-impacts-twitter-spotify-reddit/
[3] Intel. (2019). Internet of things - SoC and FPGA solutions." [Online]. Available: https://www:altera:com/solutions/technology/iot/overview:html
[4] J. Guajardo, S. S. Kumar, G.-J. Schrijen, and P. Tuyls, *FPGA Intrinsic PUFs and their Use for IP Protection*, P. Paillier and I. Verbauwhede, Eds. Berlin, Germany: Springer, 2007.
[5] B. Gassend, D. Clarke, M. van Dijk, and S. Devadas, "Silicon physical random functions," in *Proc. 9th Conf. Comput. Commun. Secur.*, 2002, pp. 148–160.
[6] B. Gassend, D. Lim, D. Clarke, M. Van Dijk, and S. Devadas, "Identification and authentication of integrated circuits," *ACM Concurrency Comput.: Practice Experience*, vol. 16, no. 11, pp. 1077–1098, 2004.
[7] C. Gu, Y. Cui, N. Hanley, and M. O'Neill, "Novel lightweight FF-APUF design for FPGA," in *Proc. 29th Int. Conf. Syst.-on-Chip*, 2016, pp. 75–80.
[8] J. Delvaux, "Machine-learning attacks on PolyPUFs, OB-PUFs, RPUFs, LHS-PUFs, and PUFU" FSMs," *IEEE Trans. Info. Forensics Security*, vol. 14, no. 8, pp. 2043–2058, Aug. 2014.
[9] M.-D. Yu, M. Hiller, J. Delvaux, R. Sowell, S. Devadas, and I. Verbauwhede, "A lockdown technique to prevent machine learning on PUFs for lightweight authentication," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 2, no. 3, pp. 146–159, Jul.–Sep. 2016.
[10] Y. Gao, S. F. Al-Sarawi, D. Abbott, A. Sadeghi, and D. C. Ranasinghe, "Modeling attack resilient reconfigurable latent obfuscation technique for PUF based lightweight authentication," Cornell University, Tech. Rep. 1706.06232, 2017.
[11] Y. Hori, H. Kang, T. Katashita, A. Satoh, S. Kawamura, and K. Kobara, "Evaluation of physical unclonable functions for 28-nm process field-programmable gate arrays," *J. Inf. Process.*, vol. 22, no. 2, pp. 344–356, 2014.
[12] D. Merli, F. Stumpf, and C. Eckert, "Improving the quality of ring oscillator PUFs on FPGAs," in *Proc. 5th Workshop Embedded Syst. Secur.*, 2010, Art. no. 9.
[13] J. Murphy, M. O'Neill, F. Burns, A. Bystrov, A. Yakovlev, and B. Halak, "Self-timed physically unclonable functions," in *Proc 5th IFIP Int. Conf. New Technol.*, May 2012, pp. 1–5.
[14] H. Yu, P. H. W. Leong, and Q. Xu, "An FPGA chip identification generator using configurable ring oscillator," in *Proc. Int. Conf. Field-Programmable Technol.*, Dec. 2010, pp. 312–315.
[15] C. Gu, J. Murphy, and M. O'Neill, "A unique and robust single slice FPGA identification generator," in *Proc. Int. Symp. Circuits Syst.*, Jun. 2014, pp. 1223–1226.
[16] C. Gu and M. O'Neill, "Ultra-compact and robust FPGA-based PUF identification generator," in *Proc. Int. Symp. Circuits Syst.*, May 2015, pp. 934–937.
[17] C. Gu, N. Hanley, and M. O'neill, "Improved reliability of FPGA-based PUF identification generator design," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 10, no. 3, pp. 20:1–20:23, May 2017.
[18] J. W. Lee, D. Lim, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *Proc. Symp. VLSI Circuits*, 2004, pp. 176–179.
[19] D. Lim, J. W. Lee, B. Gassend, G. E. Suh, M. Van Dijk, and S. Devadas, "Extracting secret keys from integrated circuits," *IEEE Trans. VLSI Syst.*, vol. 13, no. 10, pp. 1200–1205, Oct. 2005.
[20] C. H. Chang, Y. Zheng, and L. Zhang, "A retrospective and a look forward: Fifteen years of physical unclonable function advancement," *IEEE Circuits Syst. Mag.*, vol. 17, no. 3, pp. 32–62, Jul.–Sep. 2017.
[21] S. Devadas, A. Kharaya, F. Koushanfar, and M. Majzoobi, "Automated design, implementation, and evaluation of arbiter-based PUF on FPGA using programmable delay lines," RICE Digital Scholarship Archive, Tech. Rep. 2014.
[22] U. Rührmair, J. Sölter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas, "PUF modeling attacks on simulated and silicon data," *IEEE Trans. Inf. Forensics Secur.*, vol. 8, no. 11, pp. 1876–1891, Nov. 2013.
[23] J. Tobisch and G. T. Becker, "On the scaling of machine learning attacks on PUFs with application to noise bifurcation," in *Proc. Int. Workshop Radio Frequency Identification: Secur. Privacy Issues*, 2015, pp. 17–31.
[24] G. T. Becker, "The gap between promise and reality: On the insecurity of XOR arbiter PUFs," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2015, pp. 535–555.
[25] G. T. Becker, "On the pitfalls of using arbiter-PUFs as building blocks," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 34, no. 8, pp. 1295–1307, Aug. 2015.
[26] S. S. Zalivaka, A. A. Ivaniuk, and C.-H. Chang, "Low-cost fortification of arbiter PUF against modeling attack," in *Proc. Int. Symp. Circuits Syst.*, 2017, pp. 1–4.
[27] G. E. Suh and S. Devadas, "Physical unclonable functions for device authentication and secret key generation," in *Proc. 44th Annu. Des. Autom. Conf.*, 2007, pp. 9–14.
[28] M. Majzoobi, F. Koushanfar, and M. Potkonjak, "Lightweight secure PUFs," in *Proc. Int. Conf. Comput.-Aided Des.*, 2008, pp. 670–673.
[29] A. Vijayakumar and S. Kundu, "A novel modeling attack resistant PUF design based on non-linear voltage transfer characteristics," in *Proc. Des. Autom. Test Eur. Conf. Exhib.*, Mar. 2015, pp. 653–658.
[30] R. Kumar and W. Burleson, "On design of a highly secure PUF based on non-linear current mirrors," in *Proc. Int. Symp. Hardware-Oriented Secur. Trust*, May 2014, pp. 38–43.
[31] A. Vijayakumar, V. C. Patil, C. B. Prado, and S. Kundu, "Machine learning resistant strong PUF: possible or a pipe dream?" in *Proc. Int. Symp. Hardware Oriented Secur. Trust*, May 2016, pp. 19–24.
[32] S. S. Zalivaka, A. V. Puchkov, V. P. Klybik, A. A. Ivaniuk, and C. H. Chang, "Multi-valued arbiters for quality enhancement of PUF responses on FPGA implementation," in *Proc. 21st Asia South Pacific Des. Autom. Conf.*, Jan. 2016, pp. 533–538.
[33] P. Simons, E. van der Sluis, and V. van der Leest, "Buskeeper PUFs, a promising alternative to D flip-flop PUFs," in *Proc. Int. Symp. Hardware-Oriented Secur. Trust*, Jun. 2012, pp. 7–12.
[34] L. Lin, D. Holcomb, D. K. Krishnappa, P. Shabadi, and W. Burleson, "Low-power sub-threshold design of secure physical unclonable functions," in *Proc. Int. Symp. Low Power Electron. Des.*, Aug. 2010, pp. 43–48.
[35] D. P. Sahoo, R. S. Chakraborty, and D. Mukhopadhyay, "Towards ideal arbiter PUF design on Xilinx FPGA: A practitioner's perspective," in *Proc. Euromicro Conf. Digital Syst. Des.*, Aug. 2015, pp. 559–562.
[36] Y. Su, J. Holleman, and B. P. Otis, "A digital 1.6 pJ/bit chip identification circuit using process variations," *IEEE J. Solid-State Circuits*, vol. 43, no. 1, pp. 69–77, Jan. 2008.
[37] D. Yamamoto, K. Sakiyama, M. Iwamoto, K. Ohta, T. Ochiai, M. Takenaka, and K. Itoh, "Uniqueness enhancement of PUF responses based on the locations of random outputting RS latches," in *Proc. Int. Workshop Cryptographic Hardware Embedded Syst.*, 2011, pp. 390–406.

[38] R. Maes, P. Tuyls, and I. Verbauwhede, "Intrinsic PUFs from Flip-flops on reconfigurable devices," in *Proc. 3rd Benelux Workshop Inf. Syst. Secur.*, 2008, Art. no. 17.

[39] V. Van der Leest, G.-J. Schrijen, H. Handschuh, and P. Tuyls, "Hardware intrinsic security from D flip-flops," in *Proc. 5th Workshop Scalable Trusted Comput.*, 2010, pp. 53–62.

[40] S. S. Kumar, J. Guajardo, R. Maes, G.-J. Schrijen, and P. Tuyls, "The butterfly PUF protecting IP on every FPGA," in *Proc. Int. Symp. Hardware-Oriented Secur. Trust*, 2008, pp. 67–70.

[41] Q. Chen, G. Csaba, P. Lugli, U. Schlichtmann, and U. Rührmair, "The bistable ring PUF: A new architecture for strong physical unclonable functions," in *Proc. Int Symp. Hardware-Oriented Secur. Trust*, Jun. 2011, pp. 134–141.

[42] T. Machida, D. Yamamoto, M. Iwamoto, and K. Sakiyama, "Implementation of double arbiter PUF and its performance evaluation on FPGA," in *Proc. 20th Asia South Pacific Des. Autom. Conf.*, 2015, pp. 6–7.

[43] Y. Hori, T. Yoshida, T. Katashita, and A. Satoh, "Quantitative and statistical performance evaluation of arbiter physical unclonable functions on FPGAs," in *Proc. Int. Conf. Reconfigurable Comput. FPGAs*, Dec. 2010, pp. 298–303.

[44] A. Maiti, V. Gunreddy, and P. Schaumont, *A Systematic Method to Evaluate and Compare the Performance of Physical Unclonable Functions*. New York, NY, USA: Springer, 2013, pp. 245–267.

[45] F. M. Willems, Y. M. Shtarkov, and T. J. Tjalkens, "The context-tree weighting method: Basic properties," *IEEE Trans. Inf. Forensics Secur.*, vol. 41, no. 3, pp. 653–664, May 1995.

[46] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, and E. Barker, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," DTIC Document, Tech. Rep., 2001.

[47] R. Maes and I. Verbauwhede, "Physically unclonable functions: A study on the state of the art and future research directions," in *Towards Hardware-Intrinsic Security*. Berlin, Germany: Springer, 2010, pp. 3–37.

[48] S. Katzenbeisser, Ü. Kocabaş, V. Rožić, A.-R. Sadeghi, I. Verbauwhede, and C. Wachsmann, "PUFs: Myth, fact or busted? A security evaluation of physically unclonable functions (PUFs) cast in silicon," in *Proc Int. Workshop Cryptographic Hardware Embedded Syst.*, 2012, pp. 283–301.

[49] E. B. Barker and J. M. Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators (Revised)*. US Department of Commerce, Technology Administration, National Institute of Standards and Technology, Computer Security Division, Information Technology Laboratory, 2007.

[50] M. Claes, V. van der Leest, and A. Braeken, "Comparison of SRAM and FF PUF in 65nm technology," in *Proc. Nordic Conf. Secure IT Syst.*, 2011, pp. 47–64.

[51] U. Rührmair, F. Sehnke, J. Sölter, G. Dror, S. Devadas, and J. Schmidhuber, "Modeling attacks on physical unclonable functions," in *Proc. 17th Conf. Comput. Commun. Secur.*, 2010, pp. 237–249.

[52] Q. Ma, C. Gu, N. Hanley, C. Wang, W. Liu, and M. O'Neill, "A machine learning attack resistant multi-PUF design on FPGA," in *Proc. IEEE 23rd Asia South Pacific Des. Autom. Conf.*, Jan. 2018, pp. 97–104.

CHONGYAN GU (S'14–M'16) received the MSc degree with distinction in data communications from the University of Sheffield, Sheffield, United Kingdom, in 2006, and the PhD degree from Queen's University Belfast, Belfast, United Kingdom, in 2016. Currently, she is a lecturer with the Center for Secure Information Technologies (CSIT), Queen's University Belfast (QUB), United Kingdom. She was a research fellow in CSIT, QUB, United Kingdom. Before joining QUB, she was an electronic engineer with Vehicle Security and Communication System Design, GAC Mitsubishi Corporation, China. Her current research interests include hardware security and trust, physical unclonable functions, true random number generator, hardware Trojan detection, logic obfuscation circuit, and machine learning. She is a member of the IEEE.

WEIQIANG LIU (M'12-SM'15) received the BSc degree in information engineering from the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 2006, and the PhD degree in electronic engineering from the Queen's University Belfast (QUB), Belfast, United Kingdom, in 2012. In Dec. 2013, he joined the College of Electronic and Information Engineering, NUAA, where he is currently an associate professor. He has published one research book by Artech House and more than 80 leading journal and conference papers. His papers were Best Paper Candidates of ACM GLSVLSI 2015 and IEEE ISCAS 2011. One of his papers was the Most Popular Article of the *IEEE Transactions on Computers* (TC) in July 2017 and one was selected as the Feature Paper of IEEE TC in the 2017 December issue. He serves as an Steering Committee member of the *IEEE Transactions on Multi-Scale Computing Systems*, the associate editor of the *IEEE Transactions on Computers* and the *IEEE Transactions on Emerging Topics in Computing* (TETC), and the guest editors of the *IEEE Transactions on Emerging Topics in Computing* and the *Microelectronics Journal*. He has been a technical program committee member for several international conferences including ARITH, ASP-DAC, ASAP, ISCAS, ISVLSI, NANOARCH, SiPS, AisaHOST and ICONIP. He is a member of CASCOM and VSA technical committee of IEEE CAS Society. His research interests include approximate computing, computer arithmetic, and hardware security. He is a senior member of the IEEE.

YIJUN CUI received the BE degree in information engineering from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2010, where he is currently working toward the PhD degree. He was visiting PhD student with the Data Security System Group, Centre of Secure Information Technologies, Queen's University Belfast, United Kingdom. His current research interests include physically unclonable functions and hardware security.

NEIL HANLEY received the BEng (first-class honours) and PhD degrees in electrical and electronic engineering from the University College Cork, Cork, Ireland, in 2006 and 2014, respectively. He is currently a principal engineer with CSIT, Queen's University Belfast. His research interests include secure hardware architectures for quantum-safe cryptography, physically unclonable functions and their applications, and securing embedded systems from side-channel attacks.

MÁIRE O'NEILL (M'03-SM'11) is currently director of the UK Research Institute in Secure Hardware and Embedded Systems (RISE). She is chair of Information Security and is research director of Data Security Systems, Centre for Secure Information Technologies (CSIT), Queen's University Belfast. She also leads the EU H2020 SAFEcrypto (Secure architectures for Future Emerging Cryptography) project (www.safecrypto.eu). She previously held an EPSRC Leadership Fellowship (2008-2014) and was a former holder of a Royal Academy of Engineering research fellowship (2003-2008). She has received numerous awards for her research work which include a 2014 Royal Academy of Engineering Silver Medal and British Female Inventor of the Year 2007. She has authored two research books and has more than 130 peer-reviewed conference and journal publications. She is an associate editor of the *IEEE Transactions on Computers* and the *IEEE Transactions on Emerging Topics in Computing* and is an IEEE Circuits and Systems for Communications Technical committee member. She is a member of the Royal Irish Academy and a fellow of the Irish Academy of Engineering. She is also a senior member of the IEEE and a member of the IET and IACR. Her research interests include hardware cryptographic architectures, lightweight cryptography, side channel analysis, physical unclonable functions, post-quantum cryptography, and quantum-dot cellular automata circuit design.

FABRIZIO LOMBARDI received the BSc (Hons) degree in electronic engineering from the University of Essex, United Kingdom, in 1977, the master's degree in microwaves and modern optics and the diploma degree in microwave engineering from the Microwave Research Unit, University College London, in 1978, and the PhD degree from the University of London, in 1982. In 1977, he joined the Microwave Research Unit, University College London. He is currently the holder of the International Test Conference Endowed Chair Professorship, North-eastern University, Boston. His research interests include bioinspired and nanomanufacturing/computing, VLSI design, testing, and fault/defect tolerance of digital systems. He has extensively published in these areas and coauthored/edited seven books. He is a fellow of the IEEE.